macromedia®

DREAMWEAVER® 8

Guide des API de Dreamweaver

Remerciements

Gestion de projet : Charles Nadeau, Robert Berry

Rédaction : Anne Sandstrom

Mise en forme : Anne Szabla, John Hammett

Gestion de la production et de l'édition : Patrice O'Neill et Rosana Francescato

Conception et production : Adam Barnett, Aaron Begley, Paul Benkman, John Francis, Geeta Karmarkar, Paul Rangel, Arena Reed, Mario Reynoso

Gestion de la localisation : Melissa Baerwald

Remerciements particuliers à Jay London, Raymond Lim, Alain Dumesny, Masayo Noda, Kristin Conradi, Yuko Yagi, ainsi qu'à tous les membres des équipes techniques et d'assurance qualité de Dreamweaver.

Première édition : Septembre 2005

Macromedia, Inc.
601 Townsend St.
San Francisco, CA 94103

Table des matières

Introduction

Le *Guide des API de Dreamweaver* décrit deux interfaces de programmation d'applications (API) qui vous permettent d'effectuer diverses tâches de prise en charge lorsque vous développez des extensions Macromedia Dreamweaver 8 et ajoutez des codes de programme à vos pages Web Dreamweaver. Ces deux API sont l'API des utilitaires et l'API JavaScript. L'API des utilitaires contient des sous-ensembles de fonctions liées qui permettent d'effectuer des types de tâches spécifiques. Il comprend les sous-ensembles suivants :

- l'API d'E/S des fichiers, qui permet un accès en lecture et en écriture aux fichiers du système local ;
- l'API HTTP, qui permet d'envoyer et de recevoir des informations à partir d'un serveur Web ;
- l'API de Design Notes, qui permet de stocker et d'extraire les notes relatives aux documents ;
- l'API d'intégration de Fireworks, qui permet de communiquer avec Macromedia Fireworks ;
- l'intégration Flash, qui contient des informations concernant l'ajout d'éléments Flash à l'interface utilisateur Dreamweaver et sur l'API des objets Flash (qui permet de créer des objets pour le contenu Macromedia Flash) ;
- l'API de base de données, qui permet d'accéder aux informations stockées dans les bases de données et de gérer les connexions à ces bases de données ;
- l'API de connectivité à une base de données, qui permet de créer un nouveau type de connexion et les boîtes de dialogue correspondantes pour les modèles de serveur nouveaux et existants ;
- l'API JavaBeans, qui extrait les noms de classe, les méthodes, les propriétés et les évènements définis pour JavaBeans ;
- l'API d'intégration de contrôle source, qui permet d'écrire des bibliothèques partagées, pour étendre la fonction Archiver/Extraire de Dreamweaver.

L'API JavaScript permet d'effectuer plusieurs tâches mineures généralement effectuées par l'utilisateur lors de la création ou de la modification de documents Dreamweaver. Ces fonctions d'API sont regroupées selon les parties de l'interface utilisateur de Dreamweaver auxquelles elles se rapportent. Ainsi, l'API JavaScript comprend les fonctions relatives à l'espace de travail, aux documents, à la conception, etc. Ces fonctions vous permettent d'effectuer des tâches telles que : ouvrir un nouveau document, obtenir ou définir une taille de police, trouver l'occurrence d'une chaîne de recherche en code HTML, afficher une barre d'outils, etc.

Arrière-plan

Ce manuel suppose une bonne maîtrise de Dreamweaver, HTML, XML, de la programmation JavaScript et, le cas échéant, de la programmation C. Pour rédiger vos propres extensions afin de créer des applications Web, vous devez connaître les langages de script côté serveur et au moins l'une des plates-formes suivantes : Active Server Pages (ASP), ASP.net, PHP: Hypertext Preprocessor (PHP), ColdFusion ou Java Server Pages (JSP).

Extension de Dreamweaver

Pour en savoir plus sur la plate-forme Dreamweaver et l'API permettant de développer des extensions de Dreamweaver, voir *Extension de Dreamweaver*. Le manuel *Extension de Dreamweaver* décrit les fonctions des API que Dreamweaver appelle pour implémenter les objets, menus, panneaux flottants, comportements de serveur, etc., qui forment les diverses fonctionnalités de Dreamweaver. Ces API permettent d'ajouter des objets, des menus, des panneaux flottants et d'autres fonctions au produit. *Extension de Dreamweaver* explique également comment personnaliser Dreamweaver en modifiant et en ajoutant des balises à divers fichiers HTML et XML, de façon à ajouter des types d'éléments, des éléments de menus, etc.

Ressources supplémentaires pour les créateurs d'extensions

Pour entrer en contact avec d'autres développeurs d'extensions, vous pouvez rejoindre le forum de discussion consacré à l'extensibilité de Dreamweaver. Pour ce faire, il vous suffit de vous rendre à l'adresse suivante : www.macromedia.com/go/extending_newsgrp.

Nouvelles fonctions de Dreamweaver 8

Les fonctions suivantes ont été ajoutées à l'API JavaScript de Dreamweaver 8. Les en-têtes désignent les chapitres et les sections qui contiennent les nouvelles fonctions :

Application

Les fonctions suivantes (fonctions globales relatives aux applications) ont été ajoutées au chapitre Application.

Application globale

- `dreamweaver.showPasteSpecialDialog()`, page 165
- `dreamweaver.showPreferencesDialog()`, page 165 (nouvel argument ajouté)

Espace de travail

Les nouvelles fonctions Fenêtre, Fractionnement de code et barre d'outils du mode Code ci-dessous ont été ajoutées au chapitre Espace de travail.

Fenêtre

- `dreamweaver.cascade()`, page 239 (prise en charge supplémentaire pour Macintosh)
- `dreamweaver.tileHorizontally()`, page 246 (prise en charge supplémentaire pour Macintosh)
- `dreamweaver.tileVertically()`, page 247 (prise en charge supplémentaire pour Macintosh)

Fractionnement de code

- `dom.collapseFullTag()`, page 249
- `dom.collapseSelectedCodeFragment()`, page 250
- `dom.collapseSelectedCodeFragmentInverse()`, page 251
- `dom.expandAllCodeFragments()`, page 252
- `dom.expandSelectedCodeFragments()`, page 252
- `dreamweaver.htmlInspector.collapseFullTag()`, page 253
- `dreamweaver.htmlInspector.collapseSelectedCodeFragment()`, page 254
- `dreamweaver.htmlInspector.collapseSelectedCodeFragmentInverse()`, page 255
- `dreamweaver.htmlInspector.expandAllCodeFragments()`, page 255

- dreamweaver.htmlInspector.expandSelectedCodeFragments(), page 256

Barre d'outils du mode Code

- dom.getOpenPathName(), page 257
- dom.getShowHiddenCharacters(), page 257
- dom.setShowHiddenCharacters(), page 258
- dom.source.applyComment(), page 258
- dom.source.removeComment(), page 259
- dreamweaver.htmlInspector.getShowHiddenCharacters(), page 260
- dreamweaver.htmlInspector.setShowHiddenCharacters(), page 261

Site

Les nouvelles fonctions suivantes (fonctions relatives aux sites) ont été ajoutées au chapitre Site.

Site

- dom.getSiteURLPrefixFromDoc(), page 264
- dom.localPathToSiteRelative(), page 265
- dom.siteRelativeToLocalPath(), page 265
- dreamweaver.compareFiles(), page 266
- dreamweaver.siteSyncDialog.compare(), page 268
- dreamweaver.siteSyncDialog.markDelete(), page 268
- dreamweaver.siteSyncDialog.markGet(), page 269
- dreamweaver.siteSyncDialog.markIgnore(), page 269
- dreamweaver.siteSyncDialog.markPut(), page 269
- dreamweaver.siteSyncDialog.markSynced(), page 270
- site.compareFiles(), page 275
- site.getAppURLPrefixForSite(), page 282
- site.getSiteURLPrefix(), page 288
- site.serverActivity(), page 298
- site.siteRelativeToLocalPath(), page 302

Document

Les nouvelles fonctions suivantes (fonctions relatives aux manipulations de fichiers) ont été ajoutées au chapitre Document.

Manipulation de fichiers

- `dreamweaver.getNewDocumentDOM()`, page 321 (nouvel argument ajouté)
- `MMXSLT.getXMLSchema()`, page 362
- `MMXSLT.getXMLSourceURI()`, page 362
- `MMXSLT.launchXMLSourceDialog()`, page 363

Contenu de page

Les nouvelles fonctions suivantes (fonctions relatives au Presse-papiers) ont été ajoutées au chapitre Page.

Presse-papiers

- `dreamweaver.clipPaste()`, page 394 (nouvel argument ajouté)

Conception

Les nouvelles fonctions CSS, mode Mise en forme et Zoom ci-dessous ont été ajoutées au chapitre Conception :

CSS

- `cssStylePalette.getInternetExplorerRendering()`, page 431
- `cssStylePalette.setInternetExplorerRendering()`, page 432
- `dom.getElementView()`, page 433
- `dom.getShowDivBackgrounds()`, page 434
- `dom.getShowDivBoxModel()`, page 435
- `dom.getShowDivOutlines()`, page 435
- `dom.resetAllElementViews()`, page 436
- `dom.setElementView()`, page 437
- `dom.setShowDivBackgrounds()`, page 438
- `dom.setShowDivBoxModel()`, page 438
- `dom.setShowDivOutlines()`, page 439

- dreamweaver.cssStylePalette.applySelectedStyle(), page 440 (nouvel argument ajouté)
- dreamweaver.cssStylePalette.deleteSelectedStyle(), page 441 (nouvel argument ajouté)
- dreamweaver.cssStylePalette.duplicateSelectedStyle(), page 442 (nouvel argument ajouté)
- dreamweaver.cssStylePalette.editSelectedStyle(), page 443 (nouvel argument ajouté)
- dreamweaver.cssStylePalette.editSelectedStyleInCodeview(), page 443
- dreamweaver.cssStylePalette.getDisplayStyles(), page 444
- dreamweaver.cssStylePalette.renameSelectedStyle(), page 448
- dreamweaver.cssStylePalette.setDisplayStyles(), page 449
- dreamweaver.getBlockVisBoxModelColors(), page 450
- dreamweaver.getBlockVisOutlineProperties(), page 451
- dreamweaver.getDivBackgroundColors(), page 451
- dreamweaver.setBlockVisOutlineProperties(), page 452
- dreamweaver.setDivBackgroundColors(), page 453

mode Mise en forme

- dom.getShowBlockBackgrounds(), page 474
- dom.getShowBlockBorders(), page 474
- dom.getShowBlockIDs(), page 475
- dom.getShowBoxModel(), page 476
- dom.setShowBlockBackgrounds(), page 476
- dom.setShowBlockBorders(), page 477
- dom.setShowBlockIDs(), page 477
- dom.setShowBoxModel(), page 478

Zoom

- dreamweaver.activeViewScale(), page 479
- dreamweaver.fitAll(), page 480
- dreamweaver.fitSelection(), page 481
- dreamweaver.fitWidth(), page 481
- dreamweaver.zoomIn(), page 482

- dreamweaver.zoomOut(), page 482

Guide

- dom.clearGuides(), page 483
- dom.createHorizontalGuide(), page 484
- dom.createVerticalGuide(), page 485
- dom.deleteHorizontalGuide(), page 486
- dom.deleteVerticalGuide(), page 486
- dom.guidesColor, page 487
- dom.guidesDistanceColor, page 488
- dom.guidesLocked, page 488
- dom.guidesSnapToElements, page 489
- dom.guidesVisible, page 489
- dom.hasGuides(), page 491
- dom.hasHorizontalGuide(), page 491
- dom.hasVerticalGuide(), page 492

Activateurs

Les nouvelles fonctions suivantes ont été ajoutées au chapitre Activateurs.

- dreamweaver.canFitSelection(), page 586
- dreamweaver.canPasteSpecial(), page 586
- dreamweaver.canZoom(), page 592
- dreamweaver.cssStylePalette.canApplySelectedStyle(), page 593 (nouvel argument ajouté)
- dreamweaver.cssStylePalette.canDeleteSelectedStyle(), page 593 (nouvel argument ajouté)
- dreamweaver.cssStylePalette.canDuplicateSelectedStyle(), page 594 (nouvel argument ajouté)
- dreamweaver.cssStylePalette.canEditSelectedStyle(), page 595 (nouvel argument ajouté)
- dreamweaver.cssStylePalette.canEditSelectedStyleInCodeview(), page 595
- dreamweaver.cssStylePalette.canRenameSelectedStyle(), page 596
- dreamweaver.siteSyncDialog.canCompare(), page 600
- dreamweaver.siteSyncDialog.canMarkDelete(), page 601

- `dreamweaver.siteSyncDialog.canMarkGet()`, page 601
- `dreamweaver.siteSyncDialog.canMarkIgnore()`, page 602
- `dreamweaver.siteSyncDialog.canMarkPut()`, page 602
- `dreamweaver.siteSyncDialog.canMarkSynced()`, page 603
- `site.canCompareFiles()`, page 607

Fonctions supprimées

Les fonctions suivantes ont été supprimées de l'API Dreamweaver 8, car les fonctions associées ont été supprimées du produit.

Errata

Vous trouverez une liste des problèmes connus dans la section Extensibility (Extension) du centre de support de Dreamweaver (www.macromedia.com/go/extending_errata).

Conventions utilisées dans ce manuel

Ce manuel utilise les conventions typographiques suivantes :

- La police de `code` indique des fragments de code et des constantes d'API, notamment des noms de classe, des noms de méthodes, des noms de fonctions, des noms de type, des scripts, des instructions SQL et des noms de balises et d'attributs HTML et XML.
- La police de *`code en italique`* identifie les éléments remplaçables dans le code.
- Le symbole de continuation (¬) indique qu'une longue ligne de code a été fractionnée sur deux lignes ou plus. En raison des limites de marge du format de ce manuel, une ligne de code continue doit ici être coupée. Lorsque vous copiez les lignes de code, supprimez le symbole de continuation et entrez-les comme une seule ligne.
- Les accolades ({ }) placées avant et après un argument de fonction indiquent que cet argument est facultatif.
- Le nom des fonctions ayant le préfixe `dreamweaver.`*`nomfonc`* peut être abrégé en `dw.`*`nomfonc`* lorsque vous écrivez le code. Ce manuel utilise le préfixe `dreamweaver.` complet dans les définitions de fonctions et dans l'index. Toutefois, dans de nombreux exemples, le préfixe `dw.` est utilisé.

Conventions de noms utilisées dans ce manuel :

- Vous — le développeur responsable de la rédaction des extensions
- L'utilisateur — la personne utilisant Dreamweaver

PARTIE 1

API d'utilitaire

1

Etudiez les fonctions utilitaires de Macromedia Dreamweaver 8 qui vous permettent d'accéder aux fichiers locaux ainsi qu'aux fichiers placés sur un site, de travailler avec des objets Macromedia Fireworks et Macromedia Flash, de gérer des connexions aux bases de données, de créer des types de connexion aux bases de données, d'accéder aux composants JavaBeans et d'intégrer Dreamweaver à divers systèmes de commande source.

CHAPITRE 1

API d'E/S des fichiers

1

Macromedia Dreamweaver 8 inclut une bibliothèque partagée C, appelée DWfile, qui donne aux auteurs d'objets, de commandes, de comportements, de traducteurs de données, de panneaux flottants et d'inspecteurs de propriétés la possibilité de lire et d'écrire des fichiers sur le système de fichiers local. Ce chapitre décrit l'API d'entrée/sortie des fichiers et son utilisation.

Pour obtenir des informations générales sur la façon dont les bibliothèques C interagissent avec l'interpréteur JavaScript dans Dreamweaver, voir Extensibilité de niveau C dans le manuel *Extension de Dreamweaver*.

Accès aux fichiers de configuration

Sur les plates-formes Microsoft Windows 2000, Windows XP et Mac OS X, les utilisateurs disposent de leur propre copie des fichiers de configuration. Chaque fois que Dreamweaver écrit dans un fichier de configuration, il le fait dans le dossier Configuration de l'utilisateur. De même, lorsqu'il lit un fichier de configuration, Dreamweaver commence par rechercher ce fichier dans le dossier Configuration de l'utilisateur, puis dans le dossier Configuration de l'application. Les fonctions DWfile procèdent de la même manière. En d'autres termes, si une extension lit ou écrit un fichier dans le dossier Configuration de l'application, elle se reporte aussi au dossier Configuration de l'utilisateur. Pour plus d'informations sur les dossiers Configuration dans un environnement multiutilisateur, voir *Extension de Dreamweaver*.

L'API d'E/S des fichiers

Toutes les fonctions de l'API d'E/S des fichiers sont des méthodes associées à l'objet `DWfile`.

DWfile.copy()

Disponibilité

Dreamweaver 3.

Description

Cette fonction copie le fichier spécifié vers un nouvel emplacement.

Arguments

originalURL, copyURL

- L'argument *originalURL*, exprimé sous la forme d'une URL de type file://, représente le fichier que vous souhaitez copier.
- L'argument *copyURL*, exprimé sous la forme d'une URL de type file://, représente l'emplacement où vous souhaitez enregistrer le fichier copié.

Valeurs renvoyées

Valeur booléenne : `true` si la copie réussit et `false` dans le cas contraire.

Exemple

Le code suivant copie un fichier appelé myconfig.cfg vers myconfig_backup.cfg :

```
var fileURL = "file:///c|/Config/myconfig.cfg";
var newURL ="file:///c|/Config/myconfig_backup.cfg";
DWfile.copy(fileURL, newURL);
```

DWfile.createFolder()

Disponibilité

Dreamweaver 2.

Description

Cette fonction crée un dossier à l'emplacement spécifié.

Arguments

folderURL

- L'argument *folderURL*, exprimé sous la forme d'une URL de type file://, représente l'emplacement auquel vous souhaitez créer le dossier.

Valeurs renvoyées

Valeur booléenne : `true` si la création du dossier a réussi et `false` dans le cas contraire.

Exemple

Le code suivant tente de créer un dossier nommé tempFolder à la racine du lecteur C et affiche un message d'avertissement indiquant si l'opération a réussi.

```
var folderURL = "file:///c|/tempFolder";
if (DWfile.createFolder(folderURL)){
  alert("Created " + folderURL);
}else{
  alert("Unable to create " + folderURL);
}
```

DWfile.exists()

Disponibilité

Dreamweaver 2.

Description

Cette fonction vérifie l'existence du fichier spécifié.

Arguments

fileURL

- L'argument *fileURL*, exprimé sous la forme d'une URL de type file://, représente le fichier requis.

Valeurs renvoyées

Valeur booléenne : `true` si le fichier existe et `false` dans le cas contraire.

Exemple

Le code suivant recherche le fichier mydata.txt et affiche un message d'avertissement indiquant à l'utilisateur si le fichier existe :

```
var fileURL = "file:///c|/temp/mydata.txt";
if (DWfile.exists(fileURL)){
  alert(fileURL + " exists!");
}else{
  alert(fileURL + " does not exist.");
}
```

DWfile.getAttributes()

Disponibilité

Dreamweaver 2.

Description

Cette fonction obtient les attributs du fichier ou dossier spécifié.

Arguments

fileURL

- L'argument *fileURL*, exprimé sous la forme d'une URL de type file://, représente le fichier ou dossier dont vous souhaitez obtenir les attributs.

Valeurs renvoyées

Chaîne représentant les attributs du fichier ou du dossier spécifié. Si le fichier ou le dossier n'existe pas, cette fonction renvoie la valeur `null`. Les caractères suivants de la chaîne représentent les attributs :

- `R` signifie lecture seule.
- `D` signifie dossier.
- `H` signifie masqué.
- `S` indique un fichier ou dossier système.

Exemple

Le code suivant obtient les attributs du fichier mydata.txt et affiche un message d'avertissement si le fichier est en lecture seule :

```
var fileURL = "file:///c|/temp/mydata.txt";
var str = DWfile.getAttributes(fileURL);
if (str && (str.indexOf("R") != -1)){
  alert(fileURL + " is read only!");
}
```

DWfile.getModificationDate()

Disponibilité

Dreamweaver 2.

Description

Cette fonction renvoie l'heure à laquelle le fichier a été modifié pour la dernière fois.

Arguments

fileURL

- L'argument *fileURL*, exprimé sous la forme d'une URL de type file://, représente le fichier dont vous vérifiez l'heure de la dernière modification.

Valeurs renvoyées

Chaîne qui contient un nombre hexadécimal représentant le nombre d'unités de temps écoulées depuis une base de temps donnée. La signification exacte des unités de temps et de la base de temps dépend de la plate-forme ; sous Windows, par exemple, une unité de temps est égale à 100 ns et la base de temps est le 1er janvier 1600.

Exemple

Comme la valeur renvoyée par cette fonction n'est pas une date et une heure identifiables et qu'elle dépend de la plate-forme employée, il est utile d'appeler la fonction deux fois pour comparer les valeurs renvoyées. L'exemple de code suivant renvoie les dates de modification des fichiers file1.txt et file2.txt et affiche un message d'avertissement indiquant le fichier le plus récent :

```
var file1 = "file:///c|/temp/file1.txt";
var file2 = "file:///c|/temp/file2.txt";
var time1 = DWfile.getModificationDate(file1);
var time2 = DWfile.getModificationDate(file2);
if (time1 == time2){
  alert("file1 and file2 were saved at the same time");
}else if (time1 < time2){
  alert("file1 older that file2");
}else{
  alert("file1 is newer than file2");
}
```

DWfile.getCreationDate()

Disponibilité

Dreamweaver 4.

Description

Cette fonction renvoie l'heure à laquelle le fichier a été créé.

Arguments

`fileURL`

- L'argument *`fileURL`*, exprimé sous la forme d'une URL de type file://, représente le fichier dont vous vérifiez l'heure de création.

Valeurs renvoyées

Chaîne qui contient un nombre hexadécimal représentant le nombre d'unités de temps écoulées depuis une base de temps donnée. La signification exacte des unités de temps et de la base de temps dépend de la plate-forme ; sous Windows, par exemple, une unité de temps est égale à 100 ns et la base de temps est le 1er janvier 1600.

Exemple

Vous pouvez appeler cette fonction ainsi que la fonction `DWfile.getModificationDate()` pour un fichier afin de comparer les dates de modification et de création :

```
var file1 = "file:///c|/temp/file1.txt";
var time1 = DWfile.getCreationDate(file1);
var time2 = DWfile.getModificationDate(file1);
if (time1 == time2){
  alert("file1 has not been modified since it was created");
}else if (time1 < time2){
  alert("file1 was last modified on " + time2);
}
```

DWfile.getCreationDateObj()

Disponibilité

Dreamweaver MX.

Description

Cette fonction obtient l'objet JavaScript représentant l'heure de création du fichier.

Arguments

`fileURL`

- L'argument *`fileURL`*, exprimé sous la forme d'une URL de type file://, représente le fichier dont vous vérifiez l'heure de création.

Valeurs renvoyées

Obtient un objet de `date` JavaScript représentant la date et l'heure de création du fichier spécifié.

DWfile.getModificationDateObj()

Disponibilité

Dreamweaver MX.

Description

Cette fonction obtient l'objet de `date` JavaScript représentant l'heure de la dernière modification du fichier.

Arguments

`fileURL`

- L'argument *`fileURL`*, exprimé sous la forme d'une URL de type file://, représente le fichier dont vous vérifiez l'heure de modification la plus récente.

Valeurs renvoyées

Obtient un objet de `date` JavaScript représentant la date et l'heure de la dernière modification du fichier spécifié.

DWfile.getSize()

Disponibilité

Dreamweaver MX.

Description

Cette fonction obtient la taille du fichier spécifié.

Arguments

`fileURL`

- L'argument *`fileURL`*, exprimé sous la forme d'une URL de type file://, représente le fichier dont vous vérifiez la taille.

Valeurs renvoyées

Nombre entier qui représente la taille réelle du fichier spécifié, exprimée en octets.

DWfile.listFolder()

Disponibilité

Dreamweaver 2.

Description

Cette fonction obtient une liste du contenu du dossier spécifié.

Arguments

`folderURL, {constraint}`

- L'argument *folderURL* est le dossier dont vous souhaitez obtenir le contenu, exprimé sous la forme d'une URL de type file:// et d'un masque de fichier facultatif composé de caractères génériques. Les caractères génériques valides sont les astérisques (*), qui représentent un ou plusieurs caractères, et les points d'interrogation (?), qui représentent un seul caractère.
- L'argument *constraint*, s'il est fourni, doit être soit `"files"` (renvoyer uniquement les fichiers), soit `"directories"` (renvoyer uniquement les dossiers). Si cet argument n'est pas spécifié, la fonction renvoie aussi bien des fichiers que des dossiers.

Valeurs renvoyées

Tableau de chaînes représentant le contenu du dossier.

Exemple

Le code suivant obtient une liste de tous les fichiers texte (TXT) du dossier C:/Temp et affiche la liste dans un message d'avertissement.

```
var folderURL = "file:///c|/temp";
var fileMask = "*.txt";
var list = DWfile.listFolder(folderURL + "/" + fileMask, "files");
if (list){
  alert(folderURL + " contains: " + list.join("\n"));
}
```

DWfile.read()

Disponibilité

Dreamweaver 2.

Description

Cette fonction lit le contenu du fichier spécifié dans une chaîne.

Arguments

fileURL

- L'argument *fileURL*, exprimé sous la forme d'une URL de type file://, représente le fichier que vous souhaitez lire.

Valeurs renvoyées

Soit une chaîne indiquant le contenu du fichier, soit la valeur `null` si la lecture échoue.

Exemple

Le code suivant lit le fichier mydata.txt et, s'il réussit, affiche un message d'avertissement renfermant le contenu du fichier :

```
var fileURL = "file:///c|/temp/mydata.txt";
var str = DWfile.read(fileURL);
if (str){
  alert(fileURL + " contains: " + str);
}
```

DWfile.remove()

Disponibilité

Dreamweaver 3.

Description

Cette fonction permet de supprimer le fichier spécifié.

Arguments

fileURL

- L'argument *fileURL*, exprimé sous la forme d'une URL de type file://, représente le fichier que vous souhaitez supprimer.

Valeurs renvoyées

Valeur booléenne : `true` si l'opération réussit et `false` dans le cas contraire.

Exemple

L'exemple suivant utilise la fonction `DWfile.getAttributes()` pour déterminer si le fichier est accessible en lecture seule et la fonction `confirm()` pour afficher à l'utilisateur une boîte de dialogue de type Oui/Non :

```
function deleteFile(){
  var delAnyway = false;
  var selIndex = document.theForm.menu.selectedIndex;

  var selFile = document.theForm.menu.options[selIndex].value;
  if (DWfile.getAttributes(selFile).indexOf('R') != -1){
    delAnyway = confirm('This file is read-only. Delete anyway?');
    if (delAnyway){
      DWfile.remove(selFile);
    }
  }
}
```

DWfile.setAttributes()

Disponibilité

Dreamweaver MX.

Description

Cette fonction définit les attributs système d'un fichier donné.

Arguments

fileURL, strAttrs

- L'argument *fileURL*, exprimé sous la forme d'une URL de type file://, identifie le fichier dont vous définissez les attributs.
- L'argument *strAttrs* spécifie les attributs système du fichier identifié par l'argument *fileURL*. Le tableau suivant décrit les valeurs d'attribut valides et leur signification :

Valeur d'attribut	Description
R	Accessible en lecture seule
W	Accessible en écriture (annule R)
H	Masqué
V	Visible (annule H)

Les valeurs acceptables pour la chaîne strAttrs sont : R, W, H, V, RH, RV, WH ou WV.

N'utilisez pas R et W conjointement, car ces attributs s'excluent l'un l'autre. Si vous les associez, R perd tout son sens et le fichier est défini comme étant accessible en écriture (W). N'utilisez pas H et V conjointement, car ils s'excluent aussi l'un l'autre. Si vous les associez, H perd tout son sens et le fichier est défini comme étant visible (V).

Si vous spécifiez l'attribut H ou V sans indiquer d'attribut de lecture/écriture R ou W, l'attribut de lecture/écriture existant pour le fichier reste inchangé. De même, si vous spécifiez l'attribut R ou W sans spécifier un attribut de visibilité H ou V, l'attribut de visibilité existant pour le fichier reste inchangé.

Valeurs renvoyées

Aucune.

DWfile.write()

Disponibilité

Dreamweaver 2.

Description

Cette fonction rédige la chaîne spécifiée dans le fichier spécifié. Si le fichier spécifié n'existe pas, il est créé.

Arguments

fileURL, text, {mode}

- L'argument *fileURL*, exprimé sous la forme d'une URL de type file://, représente le fichier dans lequel vous écrivez une chaîne.
- L'argument *text* est la chaîne qui doit être écrite.
- L'argument *mode*, s'il est fourni, doit être `"append"`. Si cet argument est omis, le contenu du fichier est écrasé par la chaîne.

Valeurs renvoyées

Valeur booléenne : `true` si l'écriture de la chaîne dans le fichier a réussi et `false` dans le cas contraire.

Exemple

Le code suivant tente d'écrire la chaîne `"xxx"` dans le fichier mydata.txt et affiche un message d'avertissement si l'opération d'écriture réussit. Il essaie ensuite d'annexer la chaîne `"aaa"` au fichier et affiche un deuxième message d'avertissement si cette opération réussit. Après l'exécution de ce script, le fichier mydata.txt contient uniquement le texte `xxxaaa`.

```
var fileURL = "file:///c|/temp/mydata.txt";
if (DWfile.write(fileURL, "xxx")){
  alert("Wrote xxx to " + fileURL);
}
if (DWfile.write(fileURL, "aaa", "append")){
  alert("Appended aaa to " + fileURL);
}
```

CHAPITRE 2

API HTTP

2

Les extensions ne fonctionnent pas uniquement dans le système de fichiers local. Macromedia Dreamweaver 8 permet d'échanger des informations avec un serveur web via le protocole HTTP (Hypertext Transfer Protocol). Ce chapitre décrit l'API HTTP et son utilisation .

Fonctionnement de l'API HTTP

Toutes les fonctions de l'API HTTP sont des méthodes associées à l'objet `MMHttp`. La plupart d'entre elles acceptent au moins une URL comme argument et la plupart renvoient un objet. Le port par défaut pour les arguments URL est 80. Pour spécifier un port différent, ajoutez deux points (:) et le numéro de port à la suite de l'URL, comme dans l'exemple suivant :

```
MMHttp.getText("http://www.myserver.com:8025");
```

Pour les fonctions qui renvoient un objet, cet objet possède deux propriétés : `statusCode` et `data`.

La propriété `statusCode` indique l'état de l'opération ; les valeurs possibles sont notamment :

- 200: Status OK (état OK)
- 400: Unintelligible request (requête incompréhensible)
- 404: Requested URL not found (URL demandée introuvable)
- 405: Server does not support requested method (le serveur ne prend pas en charge la méthode demandée)
- 500: Unknown server error (erreur de serveur inconnue)
- 503: Server capacity reached (capacité du serveur atteinte)

Pour obtenir une liste complète des codes d'état pour votre serveur, consultez votre fournisseur d'accès Internet ou votre administrateur système.

La valeur de la propriété `data` varie selon la fonction ; les valeurs possibles sont spécifiées dans les listes des fonctions individuelles.

Les fonctions qui renvoient un objet ont également une version de rappel (« callback »). Les fonctions de rappel permettent aux autres fonctions de s'exécuter pendant que le serveur Web traite une requête HTTP. Ceci est utile si vous effectuez plusieurs requêtes HTTP à partir de Dreamweaver. La version de rappel d'une fonction transmet directement son ID et sa valeur de renvoi à la fonction spécifiée sous forme de premier argument.

L'API HTTP

Cette section présente en détail les fonctions qui sont des méthodes de l'objet `MMHttp`.

MMHttp.clearServerScriptsFolder()

Disponibilité

Dreamweaver MX.

Description

Supprime le dossier _mmServerScripts (et tous ses fichiers) sous le dossier racine du site en cours, qu'il soit local ou distant. Le dossier _mmServerScripts se trouve dans le dossier Configuration/Connections/Scripts/*`server-model`*/_mmDBScripts.

Arguments

`serverScriptsfolder`

- L'argument *`serverScriptsfolder`* est une chaîne qui nomme un dossier donné, en liaison avec le dossier Configuration du serveur d'application, à partir duquel vous pouvez extraire et supprimer les scripts de serveur.

Valeurs renvoyées

Un objet représentant la réponse du serveur. La propriété `data` de cet objet est une chaîne englobant le contenu des scripts supprimés. Si une erreur se produit, Dreamweaver la consigne dans la propriété `statusCode` de l'objet renvoyé.

Exemple

Le code suivant, dans un fichier de commandes de menu du dossier Configuration/Menus, supprime tous les fichiers du dossier _mmServerScripts lorsque celle-ci est appelée depuis un menu :

```
<!-- MENU-LOCATION=NONE -->
<html>
<head>
<TITLE>Clear Server Scripts</TITLE>
<SCRIPT SRC="ClearServerScripts.js"></SCRIPT>
```

```
<SCRIPT LANGUAGE="javascript">
</SCRIPT>
<body onLoad="MMHttp.clearServerScriptsFolder()">
</body>
</html>
```

MMHttp.clearTemp()

Description

Cette fonction supprime tous les fichiers du dossier Configuration/Temp situé dans le dossier de l'application Dreamweaver.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

Le code suivant, lorsqu'il est enregistré dans un fichier du dossier Configuration/Shutdown, supprime tous les fichiers du dossier Configuration/Temp lorsque l'utilisateur quitte Dreamweaver :

```
<html>
<head>
<title>Clean Up Temp Files on Shutdown</title>
</head>
<body onLoad="MMHttp.clearTemp()">
</body>
</html>
```

MMHttp.getFile()

Description

Cette fonction obtient le fichier situé à l'URL spécifiée et l'enregistre dans le dossier Configuration/Temp situé dans le dossier de l'application Dreamweaver. Dreamweaver crée automatiquement des sous-dossiers qui reproduisent la structure de dossiers du serveur ; par exemple, si le fichier spécifié est dans www.dreamcentral.com/people/index.html, Dreamweaver enregistre le fichier index.html dans le sous-dossier People du dossier www.dreamcentral.com.

Arguments

URL, {prompt}, {saveURL}, {titleBarLabel}

- L'argument *URL* est une URL absolue sur un serveur web ; si http:// n'est pas indiqué dans l'URL, Dreamweaver considère qu'il s'agit du protocole HTTP.
- L'argument facultatif *prompt* est une valeur booléenne spécifiant s'il faut inviter l'utilisateur à enregistrer le fichier. Si *saveURL* est en dehors du dossier Configuration/Temp, une valeur de *prompt* égale à `false` n'est pas prise en compte pour des raisons de sécurité.
- L'argument facultatif *saveURL* est l'emplacement sur le disque dur de l'utilisateur où le fichier doit être enregistré, exprimé sous la forme d'une URL de type file://. Si l'argument prompt a pour valeur `true` ou si *saveURL* est en dehors du dossier Configuration/Temp, l'utilisateur peut remplacer *saveURL* dans la boîte de dialogue d'enregistrement.
- L'argument facultatif *titleBarLabel* est le libellé qui doit figurer dans la barre de titre de la boîte de dialogue d'enregistrement.

Valeurs renvoyées

Un objet représentant la réponse du serveur. La propriété `data` de cet objet est une chaîne contenant l'emplacement où le fichier a été enregistré, exprimé sous la forme d'une URL de type file://. Normalement, la propriété `statusCode` de l'objet contient le code d'état envoyé par le serveur. Toutefois, si une erreur de disque se produit lors de l'enregistrement du fichier sur le lecteur local, la propriété `statusCode` contient un entier représentant l'un des codes d'erreur suivants en cas d'échec de l'opération :

- 1: Unspecified error (erreur inconnue)
- 2: File not found (fichier introuvable)
- 3: Invalid path (chemin non valide)
- 4: Number of open files limit reached (la limite du nombre de fichiers ouverts est atteinte)
- 5: Access denied (accès refusé)
- 6: Invalid file handle (identificateur de fichier non valide)
- 7: Cannot remove current working directory (impossible de supprimer le répertoire de travail en cours)
- 8: No more folder entries (plus d'entrées de dossier)
- 9: Error setting file pointer (erreur lors de la définition du pointeur de fichier)
- 10: Hardware error (erreur matérielle)
- 11: Sharing violation (violation de partage)
- 12: Lock violation (violation de verrouillage)
- 13: Disk full (disque saturé)

- 14: End of file reached (fin du fichier atteinte)

Exemple

Le code suivant obtient un fichier HTML, enregistre tous les fichiers dans le dossier Configuration/Temp, puis ouvre la copie locale du fichier HTML dans un navigateur :

```
var httpReply = MMHttp.getFile("http://www.dreamcentral.com/¬
people/profiles/scott.html",
false);
if (httpReply.statusCode == 200){
  var saveLoc = httpReply.data;
  dw.browseDocument(saveLoc);
}
```

MMHttp.getFileCallback()

Description

Cette fonction obtient le fichier situé à l'URL spécifiée, l'enregistre dans le dossier Configuration/Temp du dossier de l'application Dreamweaver, puis appelle la fonction spécifiée avec l'ID et le résultat de la requête. Lorsque le fichier est enregistré localement, Dreamweaver crée automatiquement des sous-dossiers qui reproduisent la structure de dossiers du serveur ; par exemple, si le fichier spécifié est dans www.dreamcentral.com/people/index.html, Dreamweaver enregistre le fichier index.html dans le sous-dossier People du dossier www.dreamcentral.com.

Arguments

callbackFunction, URL, {prompt}, {saveURL}, {titleBarLabel}

- L'argument *callbackFunction* est le nom de la fonction JavaScript à appeler lorsque la requête HTTP est terminée.
- L'argument *URL* est une URL absolue sur un serveur web ; si http:// n'est pas indiqué dans l'URL, Dreamweaver considère qu'il s'agit du protocole HTTP.
- L'argument facultatif *prompt* est une valeur booléenne spécifiant s'il faut inviter l'utilisateur à enregistrer le fichier. Si l'argument *saveURL* spécifie un emplacement en dehors du dossier Configuration/Temp, une valeur de *prompt* égale à `false` n'est pas prise en compte pour des raisons de sécurité.
- L'argument facultatif *saveURL* est l'emplacement sur le disque dur de l'utilisateur où le fichier doit être enregistré, exprimé sous la forme d'une URL de type file://. Si l'argument prompt a pour valeur `true` ou si *saveURL* est en dehors du dossier Configuration/Temp, l'utilisateur peut remplacer *saveURL* dans la boîte de dialogue d'enregistrement.

- L'argument facultatif *titleBarLabel* est le libellé qui doit figurer dans la barre de titre de la boîte de dialogue d'enregistrement.

Valeurs renvoyées

Un objet représentant la réponse du serveur. La propriété `data` de cet objet est une chaîne contenant l'emplacement où le fichier a été enregistré, exprimé sous la forme d'une URL de type file:///. Normalement, la propriété `statusCode` de l'objet contient le code d'état envoyé par le serveur. Toutefois, si une erreur disque se produit lors de l'enregistrement du fichier sur le lecteur local, la propriété `statusCode` contient un nombre entier représentant un code d'erreur. Voir MMHttp.getFile() pour une liste des codes d'erreur possibles.

MMHttp.getText()

Disponibilité

Macromedia Dreamweaver UltraDev 4, améliorée dans la version Dreamweaver MX.

Description

Extrait le contenu du document situé à l'URL spécifiée.

Arguments

URL, {serverScriptsFolder}

- L'argument *URL* est une URL absolue sur un serveur web. Si http:// n'est pas indiqué dans l'URL, Dreamweaver considère qu'il s'agit du protocole HTTP.
- L'argument *serverScriptsFolder* est une chaîne facultative qui nomme un dossier spécifique, lié au dossier Configuration du serveur d'application, à partir duquel vous souhaitez extraire les scripts de serveur. Pour extraire les scripts, Dreamweaver utilise le protocole de transfert approprié (par exemple FTP, WebDAV ou Remote File System). Dreamweaver copie ces fichiers dans le sous-dossier _mmServerScripts dans le dossier racine du site en cours.

Si une erreur se produit, Dreamweaver la consigne dans la propriété `statusCode` de l'objet renvoyé.

MMHttp.getTextCallback()

Disponibilité

Dreamweaver UltraDev 4, amélioré dans Dreamweaver MX.

Description

Extrait le contenu du document situé à l'URL spécifiée et le transmet à la fonction spécifiée.

Arguments

callbackFunc, URL, {serverScriptsFolder}

- L'argument *callbackFunc* est la fonction JavaScript à appeler lorsque la requête HTTP est terminée.
- L'argument *URL* est une URL absolue sur un serveur web ; si http:// n'est pas indiqué dans l'URL, Dreamweaver considère qu'il s'agit du protocole HTTP.
- L'argument *serverScriptsFolder* est une chaîne facultative qui nomme un dossier spécifique, lié au dossier Configuration du serveur d'application, à partir duquel vous souhaitez extraire les scripts de serveur. Pour extraire les scripts, Dreamweaver utilise le protocole de transfert approprié (par exemple FTP, WebDAV ou Remote File System). Dreamweaver extrait ces fichiers et les transmet à la fonction identifiée par *callbackFunc*.

Si une erreur survient, Dreamweaver MX la consigne dans la propriété `statusCode` de l'objet renvoyé.

MMHttp.postText()

Disponibilité

Dreamweaver UltraDev 4, améliorée dans la version Dreamweaver MX.

Description

Exécute un envoi HTTP des données définies à l'URL spécifiée. En règle générale, les données associées à une opération d'envoi se présentent sous la forme de texte codé en formulaire, mais il peut s'agir de tout type de données que le serveur peut accepter.

Arguments

URL, dataToPost, {contentType}, {serverScriptsFolder}

- L'argument *URL* est une URL absolue sur un serveur web ; si http:// n'est pas indiqué dans l'URL, Dreamweaver considère qu'il s'agit du protocole HTTP.
- L'argument *dataToPost* représente les données à envoyer. Si le troisième argument est `"application/x-www-form-urlencoded"` ou s'il n'est pas spécifié, *dataToPost* doit être codé en formulaire conformément à la section 8.2.1 de la spécification RFC 1866 (disponible à l'adresse www.faqs.org/rfcs/rfc1866.html).
- L'argument facultatif *contentType* est le type de contenu des données à envoyer. S'il n'est pas spécifié, il prend par défaut la valeur `" application/x-www-form-urlencoded "`.

- L'argument *serverScriptsFolder* est une chaîne facultative qui nomme un dossier spécifique, lié au dossier Configuration du serveur d'application, vers lequel vous souhaitez envoyer les données. Pour envoyer les données, Dreamweaver utilise le protocole de transfert approprié (par exemple FTP, WebDAV ou Remote File System).

Si une erreur se produit, Dreamweaver la consigne dans la propriété `statusCode` de l'objet renvoyé.

Exemple

Dans l'exemple suivant d'appel de la fonction `MMHttp.postText()`, supposons qu'un développeur a placé le fichier myScripts.cfm dans un dossier nommé DeployScripts, qui se trouve dans le dossier Configuration sur l'ordinateur local :

```
MMHttp.postText(
    "http://ultraqa8/DeployScripts/myScripts.cfm",
    "arg1=Foo",
    "application/x-www-form-urlencoded",
    "Configuration/DeployScripts/"
)
```

Voici ce qui se produit lorsque Dreamweaver effectue cet appel de fonction :

1. Le fichier myScripts.cfm du dossier Configuration/DeployScripts de l'ordinateur local est copié dans un autre dossier nommé DeployScripts, qui constitue un sous-dossier du dossier racine du site Web ultraqa8. Pour déployer les fichiers, Dreamweaver utilise le protocole spécifié dans les propriétés de configuration du site.
2. Dreamweaver utilise le protocole HTTP pour envoyer les données `arg1=Foo` vers le serveur Web.
3. En réponse à la requête d'envoi, le serveur Web sur ultraqa8 exécute le script myScripts.cfm à l'aide des données `arg1`.

MMHttp.postTextCallback()

Disponibilité

Dreamweaver UltraDev 4, améliorée dans la version Dreamweaver MX.

Description

Exécute un envoi HTTP du texte à l'URL spécifiée et transmet la réponse du serveur à la fonction spécifiée. En règle générale, les données associées à une opération d'envoi se présentent sous la forme de texte codé en formulaire, mais il peut s'agir de tout type de données que le serveur peut accepter.

Arguments

callbackFunc, URL, dataToPost, {contentType}, {serverScriptsFolder}

- L'argument *callbackFunc* est le nom de la fonction JavaScript à appeler lorsque la requête HTTP est terminée.
- L'argument *URL* est une URL absolue sur un serveur web ; si http:// n'est pas indiqué dans l'URL, Dreamweaver considère qu'il s'agit du protocole HTTP.
- L'argument *dataToPost* représente les données à envoyer. Si le troisième argument est « application/x-www-form-urlencoded » ou s'il n'est pas spécifié, *data* doit être codé en formulaire conformément à la section 8.2.1 de la spécification RFC 1866 (disponible à l'adresse www.faqs.org/rfcs/rfc1866.html).
- L'argument facultatif *contentType* correspond au type de contenu des données à envoyer. S'il n'est pas spécifié, il prend par défaut la valeur " application/x-www-form-urlencoded ".
- L'argument *serverScriptsFolder* est une chaîne facultative. Il nomme un dossier donné, en liaison avec le dossier Configuration du serveur d'application sur lequel vous voulez envoyer les données. Pour envoyer les données, Dreamweaver utilise le protocole de transfert approprié (par exemple FTP, WebDAV ou Remote File System). Dreamweaver extrait ces données et les transmet à la fonction identifiée par *callbackFunc*.

Si une erreur se produit, Dreamweaver la consigne dans la propriété statusCode de l'objet renvoyé.

CHAPITRE 3

API de Design Notes

3

Macromedia Dreamweaver 8, Macromedia Fireworks et Macromedia Flash permettent aux concepteurs et développeurs de sites Web de stocker et de récupérer des informations complémentaires sur les documents (telles que des commentaires de révision, des notes de modification ou le fichier source d'un document GIF ou JPEG) dans des fichiers appelés Design Notes.

MMNotes est une bibliothèque C partagée qui permet aux auteurs d'extensions de lire et d'écrire dans les fichiers Design Notes. A l'instar de la bibliothèque partagée DWfile, MMNotes possède une API JavaScript qui permet d'appeler les fonctions contenues dans la bibliothèque à partir d'objets, de commandes, de comportements, de panneaux flottants, d'inspecteurs de propriétés et de traducteurs de données.

MMNotes possède également une API C qui donne à d'autres applications la possibilité de lire et d'écrire dans les fichiers Design Notes. La bibliothèque partagée MMNotes peut être utilisée indépendamment de Dreamweaver, que celui-ci soit installé ou non.

Pour plus d'informations sur l'utilisation de la fonctionnalité Design Notes dans Dreamweaver, voir le manuel *Utilisation de Dreamweaver*.

Fonctionnement de Design Notes

Chaque fichier Design Notes stocke des informations relatives à un seul document. Si un fichier Design Notes est associé à un ou plusieurs documents dans un dossier, Dreamweaver crée un sous-dossier _notes pour y stocker les fichiers Design Notes. Le dossier _notes et les fichiers Design Notes qu'il contient ne sont pas visibles dans la fenêtre Site, mais ils s'affichent dans le Finder Macintosh ou dans l'Explorateur Windows. Un nom de fichier Design Notes est composé du nom du fichier principal suivi de l'extension .mno. Par exemple, le fichier Design Notes associé à avocado8.gif est avocado8.gif.mno.

Les fichiers Design Notes sont des fichiers XML stockant des informations sous la forme d'une série de paires clé/valeur. La clé décrit le type des informations stockées et la valeur représente les informations. Les clés sont limitées à 64 caractères.

L'exemple suivant illustre le fichier Design Notes associé au fichier foghorn.gif.mno :

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<info>
  <infoitem key="FW_source" value="file:///C|sites/¬
  dreamcentral/images/sourceFiles/foghorn.png" />
  <infoitem key="Author" value="Heidi B." />
  <infoitem key="Status" value="Final draft, approved ¬
  by Jay L." />
</info>
```

API JavaScript de Design Notes

Toutes les fonctions de l'API JavaScript de Design Notes sont des méthodes associées à l'objet `MMNotes`.

MMNotes.close()

Description

Cette fonction ferme le fichier Design Notes spécifié et enregistre les modifications éventuelles. Si toutes les paires clé/valeur ont été supprimées, Dreamweaver supprime le fichier Design Notes. S'il s'agit du dernier fichier Design Notes dans le dossier _notes, Dreamweaver supprime également le dossier.

REMARQUE	Appelez toujours la fonction `MMNotes.close()` une fois le travail sur les Design Notes terminé, afin que Dreamweaver écrive sur le fichier.

Arguments

`fileHandle`

- L'argument *`fileHandle`* est l'identificateur de fichier renvoyé par la fonction `MMNotes.open()`.

Valeurs renvoyées

Aucune.

Exemple

Voir *MMNotes.set()*, page 45.

MMNotes.filePathToLocalURL()

Description

Cette fonction convertit le chemin d'accès du lecteur local spécifié en une URL de type file://.

Arguments

drivePath

- L'argument *drivePath* est une chaîne contenant le chemin d'accès complet du lecteur.

Valeurs renvoyées

Une chaîne contenant l'URL de type file:// du fichier spécifié.

Exemple

Un appel à la fonction `MMNotes.filePathToLocalURL('C:/sites/webdev/index.htm')` renvoie `"file:///c|sites/webdev/index.htm"`.

MMNotes.get()

Description

Cette fonction obtient la valeur de la clé spécifiée dans le fichier Design Notes indiqué.

Arguments

fileHandle, keyName

- L'argument *fileHandle* est l'identificateur de fichier renvoyé par `MMNotes.open()`.
- L'argument *keyName* est une chaîne contenant le nom de la clé.

Valeurs renvoyées

Une chaîne contenant la valeur de la clé.

Exemple

Voir *MMNotes.getKeys()*, page 42.

MMNotes.getKeyCount()

Description

Cette fonction obtient le nombre de paires clé/valeur du fichier Design Notes spécifié.

Arguments

fileHandle

- L'argument *fileHandle* est l'identificateur de fichier renvoyé par la fonction `MMNotes.open()`.

Valeurs renvoyées

Un nombre entier représentant le nombre de paires clé/valeur du fichier Design Notes spécifié.

MMNotes.getKeys()

Description

Cette fonction obtient une liste de toutes les clés d'un fichier Design Notes.

Arguments

fileHandle

- L'argument *fileHandle* est l'identificateur de fichier renvoyé par la fonction `MMNotes.open()`.

Valeurs renvoyées

Un tableau de chaînes, chacune d'elles contenant le nom d'une clé.

Exemple

Le code suivant peut être utilisé dans un panneau flottant personnalisé afin d'afficher les informations Design Notes relatives au document actif :

```
var noteHandle = MMNotes.open(dw.getDocumentDOM().URL);
var theKeys = MMNotes.getKeys(noteHandle);
var noteString = "";
var theValue = "";
for (var i=0; i < theKeys.length; i++){
  theValue = MMNotes.get(noteHandle,theKeys[i]);
  noteString +=  theKeys[i] + " = " theValue + "\n";
}
document.theForm.bigTextField.value = noteString;
// always close noteHandle
MMNotes.close(noteHandle);
```

MMNotes.getSiteRootForFile()

Description

Cette fonction détermine la racine du site pour le fichier Design Notes spécifié.

Arguments

fileURL

- L'argument *fileURL*, exprimé sous la forme d'une URL de type file://, est le chemin d'un fichier local.

Valeurs renvoyées

Une chaîne contenant le chemin du dossier racine local pour le site, exprimé sous la forme d'une URL de type file://, ou une chaîne vide si Dreamweaver n'est pas installé ou si le fichier Design Notes se trouve en dehors de tout site défini par Dreamweaver. Cette fonction recherche tous les sites définis dans Dreamweaver.

MMNotes.getVersionName()

Description

Cette fonction obtient le nom de version de la bibliothèque partagée MMNotes indiquant l'application qui l'a implémentée.

Arguments

Aucun.

Valeurs renvoyées

Une chaîne contenant le nom de l'application qui a implémenté la bibliothèque partagée MMNotes.

Exemple

L'appel de la fonction `MMNotes.getVersionName()` à partir d'une commande, d'un objet, d'un comportement, d'un inspecteur de propriétés, d'un panneau flottant ou d'un traducteur de données Dreamweaver renvoie `"Dreamweaver"`. L'appel de la fonction `MMNotes.getVersionName()` à partir de Fireworks renvoie également la valeur `"Dreamweaver"` étant donné que Fireworks utilise la même version de la bibliothèque (celle qui a été créée par l'équipe technique de Dreamweaver).

MMNotes.getVersionNum()

Description

Cette fonction obtient le numéro de version de la bibliothèque partagée MMNotes.

Arguments

Aucun.

Valeurs renvoyées

Une chaîne contenant le numéro de version.

MMNotes.localURLToFilePath()

Description

Cette fonction convertit l'URL de type file:// spécifiée en un chemin d'accès du lecteur local.

Arguments

fileURL

- L'argument *fileURL*, exprimé sous la forme d'une URL de type file://, est le chemin d'un fichier local.

Valeurs renvoyées

Une chaîne contenant le chemin d'accès du lecteur local pour le fichier spécifié.

Exemple

Un appel à la fonction `MMNotes.localURLToFilePath('file:///MacintoshHD/images/moon.gif')` renvoie `"MacintoshHD:images:moon.gif"`.

MMNotes.open()

Description

Cette fonction ouvre le fichier Design Notes associé au fichier spécifié ou crée un fichier Design Notes s'il n'en existe pas.

Arguments

filePath, [bForceCreate]

- L'argument *filePath*, exprimé sous la forme d'une URL de type file://, est le chemin du fichier principal auquel le fichier Design Notes est associé.
- L'argument *bForceCreate* est une valeur booléenne indiquant si la note doit être créée même si la fonctionnalité Design Notes est désactivée pour le site ou si l'argument *filePath* n'est associé à aucun site.

Valeurs renvoyées

L'identificateur du fichier Design Notes ou zéro (0) si le fichier n'a pas été ouvert ni créé.

Exemple

Voir *MMNotes.set()*, page 45.

MMNotes.remove()

Description

Cette fonction supprime la clé spécifiée (et sa valeur) du fichier Design Notes indiqué.

Arguments

fileHandle, keyName

- L'argument *fileHandle* est l'identificateur de fichier renvoyé par la fonction `MMNotes.open()`.
- L'argument *keyName* est une chaîne contenant le nom de la clé à supprimer.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec.

MMNotes.set()

Description

Cette fonction crée ou met à jour une paire clé/valeur dans un fichier Design Notes.

Arguments

fileHandle, keyName, valueString

- L'argument *fileHandle* est l'identificateur de fichier renvoyé par la fonction `MMNotes.open()`.
- L'argument *keyName* est une chaîne contenant le nom de la clé.
- L'argument *valueString* est une chaîne contenant la valeur.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec.

Exemple

L'exemple suivant ouvre le fichier Design Notes associé à un fichier situé sur le site dreamcentral appelé peakhike99/index.html, ajoute une nouvelle paire clé/valeur, modifie la valeur d'une clé existante, puis ferme le fichier de Design Notes :.

```
var noteHandle = MMNotes.open('file:///c|/sites/dreamcentral/
peakhike99/index.html',true);
if(noteHandle > 0){
  MMNotes.set(noteHandle,"Author","M. G. Miller");
  MMNotes.set(noteHandle,"Last Changed","August 28, 1999");
  MMNotes.close(noteHandle);
}
```

API Cde Design Notes

Outre l'API JavaScript, la bibliothèque partagée MMNotes affiche une API C permettant aux autres applications de créer des fichiers Design Notes. Il n'est pas nécessaire d'appeler les fonctions C directement si vous utilisez la bibliothèque partagée MMNotes dans Dreamweaver ; les versions JavaScript de ces fonctions les appellent pour vous.

Cette section décrit ces fonctions, leurs arguments et les valeurs qu'elles renvoient. Les définitions de toutes les fonctions et de tous les types de données sont disponibles dans le fichier MMInfo.h du dossier Extending/c_files dans le dossier de l'application Dreamweaver.

void CloseNotesFile()

Description

Cette fonction ferme le fichier Design Notes spécifié et enregistre les modifications éventuelles. Si toutes les paires clé/valeur ont été supprimées du fichier Design Notes, Dreamweaver supprime ce dernier. Dreamweaver supprime le dossier _notes lorsque le dernier fichier Design Notes est supprimé.

Arguments

`FileHandle noteHandle`

- L'argument `noteHandle` est l'identificateur de fichier renvoyé par la fonction `OpenNotesFile()`.

Valeurs renvoyées

Aucune.

BOOL FilePathToLocalURL()

Description

Cette fonction convertit le chemin d'accès du lecteur local spécifié en une URL de type file://.

Arguments

`const char* drivePath, char* localURLBuf, int localURLMaxLen`

- L'argument `drivePath` est une chaîne contenant le chemin d'accès complet du lecteur.
- L'argument `localURLBuf` est la zone de mémoire tampon où l'URL de type file:// est stockée.
- L'argument `localURLMaxLen` est la taille maximale de `localURLBuf`.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec. L'argument *`localURLBuf`* reçoit la valeur de l'URL de type file://.

BOOL GetNote()

Description

Cette fonction obtient la valeur de la clé spécifiée dans le fichier Design Notes indiqué.

Arguments

`FileHandle` *`noteHandle`*`, const char` *`keyName[64]`*`, char*` *`valueBuf`*`, int` *`valueBufLength`*

- L'argument *`noteHandle`* est l'identificateur de fichier renvoyé par la fonction `OpenNotesFile()`.
- L'argument *`keyName[64]`* est une chaîne contenant le nom de la clé.
- L'argument *`valueBuf`* est la zone de mémoire tampon où la valeur est stockée.
- L'argument *`valueBufLength`* est le nombre entier renvoyé par `GetNoteLength(`*`noteHandle`*`,` *`keyName`*`)`, indiquant la longueur maximale de la mémoire tampon des valeurs.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec. L'argument *`valueBuf`* reçoit la valeur de la clé.

Exemple

Le code suivant obtient la valeur de la clé `comments` dans le fichier Design Notes associé au fichier welcome.html :

```
FileHandle noteHandle = OpenNotesFile("file:///c|/sites/avocado8/¬
iwjs/welcome.html");
if(noteHandle > 0){
  int valueLength = GetNoteLength( noteHandle, "comments");
  char* valueBuffer = new char[valueLength + 1];
  GetNote(noteHandle, "comments", valueBuffer, valueLength + 1);
  printf("Comments: %s",valueBuffer);
  CloseNotesFile(noteHandle);
}
```

int GetNoteLength()

Description

Cette fonction obtient la longueur de la valeur associée à la clé spécifiée.

Arguments

`FileHandle noteHandle, const char keyName[64]`

- L'argument `noteHandle` est l'identificateur de fichier renvoyé par la fonction `OpenNotesFile()`.
- L'argument `keyName[64]` est une chaîne contenant le nom de la clé.

Valeurs renvoyées

Un nombre entier représentant la longueur de la valeur.

Exemple

Voir *BOOL GetNote()*, page 47.

int GetNotesKeyCount()

Description

Cette fonction obtient le nombre de paires clé/valeur du fichier Design Notes spécifié.

Arguments

`FileHandle noteHandle`

- L'argument `noteHandle` est l'identificateur de fichier renvoyé par la fonction `OpenNotesFile()`.

Valeurs renvoyées

Un nombre entier représentant le nombre de paires clé/valeur du fichier Design Notes spécifié.

BOOL GetNotesKeys()

Description

Cette fonction renvoie une liste de toutes les clés d'un fichier Design Notes.

Arguments

`FileHandle noteHandle, char* keyBufArray[64], int keyArrayMaxLen`

- L'argument `noteHandle` est l'identificateur de fichier renvoyé par `OpenNotesFile()`.

- L'argument *keyBufArray[64]* est le tableau en mémoire tampon où les clés sont stockées.
- L'argument *keyArrayMaxLen* est le nombre entier renvoyé par `GetNotesKeyCount(`*noteHandle*`)`, indiquant le nombre maximum d'éléments contenus dans le tableau en mémoire tampon des clés.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec. L'argument *keyBufArray* reçoit les noms de clé.

Exemple

Le code suivant imprime les noms de clé et les valeurs de toutes les clés du fichier Design Notes associé au fichier welcome.html :

```
typedef char[64] InfoKey;
FileHandle noteHandle = OpenNotesFile("file:///c|/sites/avocado8/¬
iwjs/welcome.html");
if (noteHandle > 0){
  int keyCount = GetNotesKeyCount(noteHandle);
  if (keyCount <= 0)
    return;
  InfoKey* keys = new InfoKey[keyCount];
  BOOL succeeded = GetNotesKeys(noteHandle, keys, keyCount);

  if (succeeded){
    for (int i=0; i < keyCount; i++){
      printf("Key is: %s\n", keys[i]);
      printf("Value is: %s\n\n", GetNote(noteHandle, keys[i]);
    }
  }
  delete []keys;
}
CloseNotesFile(noteHandle);
```

BOOL GetSiteRootForFile()

Description

Cette fonction détermine la racine du site pour le fichier Design Notes spécifié.

Arguments

```
const char* filePath, char* siteRootBuf, int siteRootBufMaxLen,
  {InfoPrefs* infoPrefs}
```

- L'argument *filePath* est l'URL de type file:// du fichier dont vous souhaitez obtenir la racine du site.
- L'argument *siteRootBuf* est la zone de mémoire tampon où la racine du site est stockée.

- L'argument *siteRootBufMaxLen* est la taille maximale de la mémoire tampon référencée par *siteRootBuf* .
- L'argument facultatif *infoPrefs* est une référence à un `struct` dans lequel les préférences du site sont stockées.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec. L'argument *siteRootBuf* reçoit l'adresse de la mémoire tampon qui stocke la racine du site. Si vous spécifiez l'argument *infoPrefs*, la fonction renvoie également les préférences de Design Notes pour le site. Le struct `InfoPrefs` possède deux variables : `bUseDesignNotes` et `bUploadDesignNotes`, toutes deux de type `BOOL`.

BOOL GetVersionName()

Description

Cette fonction renvoie le nom de version de la bibliothèque partagée MMNotes indiquant l'application qui l'a implémentée.

Arguments

`char* versionNameBuf, int versionNameBufMaxLen`

- L'argument *versionNameBuf* est la zone de mémoire tampon où le nom de version est stocké.
- L'argument *versionNameBufMaxLen* est la taille maximale de la mémoire tampon référencée par *versionNameBuf*.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec. Dreamweaver stocke `Dreamweaver` dans l'argument *versionNameBuf*.

BOOL GetVersionNum()

Description

Cette fonction renvoie le numéro de version de la bibliothèque partagée MMNotes, ce qui vous permet de savoir si certaines fonctions sont disponibles.

Arguments

`char* versionNumBuf, int versionNumBufMaxLen`

- L'argument *versionNumBuf* est la zone de mémoire tampon où le numéro de version est stocké.
- L'argument *versionNumBufMaxLen* est la taille maximale de la mémoire tampon référencée par *versionNumBuf* .

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec. L'argument *versionNumBuf* stocke le numéro de version.

BOOL LocalURLToFilePath()

Description

Cette fonction convertit l'URL de type file:// spécifiée en un chemin d'accès du lecteur local.

Arguments

`const char* localURL, char* drivePathBuf, int drivePathMaxLen`

- L'argument *localURL*, exprimé sous la forme d'une URL de type file://, est le chemin d'un fichier local.
- L'argument *drivePathBuf* est la zone de mémoire tampon où le chemin d'accès du lecteur local est stocké.
- L'argument *drivePathMaxLen* est la taille maximale de la mémoire tampon référencée par *drivePathBuf*.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec. L'argument *drivePathBuf* reçoit le chemin de fichier local.

FileHandle OpenNotesFile()

Description

Cette fonction ouvre le fichier Design Notes associé au fichier spécifié ou crée un fichier Design Notes s'il n'en existe pas.

Arguments

`const char* localFileURL, {BOOL bForceCreate}`

- L'argument *localFileURL*, exprimé sous la forme d'une URL de type file://, est une chaîne contenant le chemin du fichier principal auquel le fichier Design Notes est associé.

- L'argument *bForceCreate* est une valeur booléenne indiquant si le fichier Design Notes doit être créé même si la fonctionnalité Design Notes est désactivée pour le site ou si le chemin spécifié par l'argument *localFileURL* n'est associé à aucun site.

FileHandle OpenNotesFilewithOpenFlags()

Description

Cette fonction ouvre le fichier Design Notes associé au fichier spécifié ou crée un fichier Design Notes s'il n'en existe pas. Vous pouvez ouvrir le fichier en mode lecture seule.

Arguments

```
const char* localFileURL, {BOOL bForceCreate}, {BOOL bReadOnly}
```

- L'argument *localFileURL*, exprimé sous la forme d'une URL de type file://, est une chaîne contenant le chemin du fichier principal auquel le fichier Design Notes est associé.
- L'argument *bForceCreate* est une valeur booléenne indiquant si le fichier Design Notes doit être créé même si la fonctionnalité Design Notes est désactivée pour le site ou si le chemin n'est associé à aucun site. La valeur par défaut est `false`. Cet argument est facultatif, mais vous devez le définir si vous spécifiez le troisième argument.
- L'argument facultatif *bReadOnly* est une valeur booléenne indiquant si le fichier doit être ouvert en mode lecture seule. La valeur par défaut est `false`. Vous pouvez spécifier l'argument *bReadOnly* disponible à partir de la version 2 du fichier MMNotes.dll.

BOOL RemoveNote()

Description

Cette fonction supprime la clé spécifiée (et sa valeur) du fichier Design Notes indiqué.

Arguments

```
FileHandle noteHandle, const char keyName[64]
```

- L'argument *noteHandle* est l'identificateur de fichier renvoyé par la fonction `OpenNotesFile()`.
- L'argument *keyName[64]* est une chaîne contenant le nom de la clé à supprimer.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec.

BOOL SetNote()

Description

Cette fonction crée ou met à jour une paire clé/valeur dans un fichier Design Notes.

Arguments

`FileHandle noteHandle, const char keyName[64], const char* value`

- L'argument *`noteHandle`* est l'identificateur de fichier renvoyé par la fonction `OpenNotesFile()`.
- L'argument *`keyName[64]`* est une chaîne contenant le nom de la clé.
- L'argument *`value`* est une chaîne contenant la valeur.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'opération a réussi ; `false` indique un échec.

CHAPITRE 4

Intégration de Fireworks

4

FWLaunch est une bibliothèque C partagée qui donne aux auteurs d'objets, de commandes, de comportements et d'inspecteurs de propriétés la possibilité de communiquer avec Macromedia Fireworks. A l'aide de FWLaunch, vous rédigez du code JavaScript pour ouvrir l'interface utilisateur (IU) de Fireworks et fournissez des commandes à Fireworks via son propre API JavaScript, au sujet duquel vous trouverez plus d'informations dans le manuel *Extension de Fireworks*. Pour des informations générales sur l'interaction des bibliothèques C avec l'interpréteur JavaScript dans Macromedia Dreamweaver 8, voir *Extension de Dreamweaver* pour plus de détails sur les extensions C.

L'API FWLaunch

L'objet FWLaunch permet aux extensions d'ouvrir Fireworks, d'exécuter des opérations Fireworks à l'aide de l'API JavaScript de Fireworks et de renvoyer les valeurs à Dreamweaver. Ce chapitre décrit l'API de communication FWLaunch et son utilisation.

FWLaunch.bringDWToFront()

Disponibilité

Dreamweaver 3, Fireworks 3

Description

Cette fonction affiche Dreamweaver au premier plan.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

FWLaunch.bringFWToFront()

Disponibilité

Dreamweaver 3, Fireworks 3

Description

Cette fonction permet d'afficher Fireworks au premier plan s'il est en cours d'exécution.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

FWLaunch.execJsInFireworks()

Disponibilité

Dreamweaver 3, Fireworks 3

Description

Cette fonction transmet l'élément JavaScript spécifié (ou une référence à un fichier JavaScript) à Fireworks en vue de son exécution.

Arguments

`javascriptOrFileURL`

- L'argument `javascriptOrFileURL`, exprimé sous la forme d'une URL de type file://, est soit une chaîne de JavaScript littéral, soit le chemin d'accès d'un fichier JavaScript.

Valeurs renvoyées

Un objet cookie si le code JavaScript a été transmis avec succès ou un code d'erreur non nul indiquant que l'une des erreurs suivantes s'est produite :

- Utilisation non valide, ce qui indique que le chemin du fichier JS ou JSF n'est pas valide ou que l'argument `javascriptOrFileURL` est spécifié comme ayant la valeur `null` ou comme une chaîne vide.
- Erreur d'E/S de fichier, ce qui indique que Fireworks ne peut pas créer de fichier réponse car le disque est saturé.
- Erreur de notification de Dreamweaver ; l'utilisateur n'exécute pas une version valide de Dreamweaver (version 3 ou ultérieure).

- Erreur de lancement de Fireworks, ce qui indique que la fonction n'ouvre pas une version valide de Fireworks (version 3 ou ultérieure).
- L'utilisateur a annulé l'opération.

FWLaunch.getJsResponse()

Disponibilité

Dreamweaver 3, Fireworks 3

Description

Cette fonction détermine si Fireworks est toujours en train d'exécuter le code JavaScript qui lui a été transmis par la fonction `FWLaunch.execJsInFireworks()`, que l'exécution du script se soit terminée avec succès ou qu'une erreur se soit produite.

Arguments

`progressTrackerCookie`

- L'argument `progressTrackerCookie` est l'objet cookie renvoyé par la fonction `FWLaunch.execJsInFireworks()`.

Valeurs renvoyées

Une chaîne contenant le résultat du script transmis à `FWLaunch.execJsInFireworks()` si l'opération se termine avec succès, la valeur `null` si Fireworks est toujours en train d'exécuter le code JavaScript, ou un code d'erreur non nul indiquant que l'une des erreurs suivantes s'est produite :

- Utilisation non valide, ce qui indique qu'une erreur JavaScript s'est produite pendant que Fireworks exécutait le script.
- Erreur d'E/S de fichier, ce qui indique que Fireworks ne peut pas créer de fichier réponse car le disque est saturé.
- Erreur de notification de Dreamweaver ; l'utilisateur n'exécute pas une version valide de Dreamweaver (version 3 ou ultérieure).
- Erreur de lancement de Fireworks, ce qui indique que la fonction n'ouvre pas une version valide de Fireworks (version 3 ou ultérieure).
- L'utilisateur a annulé l'opération.

Exemple

Le code suivant transmet la chaîne `"prompt('Please enter your name:')"` à la fonction `FWLaunch.execJsInFireworks()`, puis il vérifie le résultat :

```
var progressCookie = FWLaunch.execJsInFireworks("prompt('Please enter your
  name:')");
var doneFlag = false;
while (!doneFlag){
  // check for completion every 1/2 second
  setTimeout('checkForCompletion()',500);
}

function checkForCompletion(){
  if (progressCookie != null) {
    var response = FWLaunch.getJsResponse(progressCookie);
    if (response != null) {
      if (typeof(response) == "number") {
        // error or user-cancel, time to close the window
        // and let the user know we got an error
        window.close();
        alert("An error occurred.");
      }else{
        // got a valid response!
        alert("Nice to meet you, " + response);
        window.close();
      }
        doneFlag = true;
    }
  }
}
```

FWLaunch.mayLaunchFireworks()

Disponibilité

Dreamweaver 2, Fireworks 2.

Description

Cette fonction détermine s'il est possible d'ouvrir une session d'optimisation de Fireworks.

Arguments

Aucun.

Valeurs renvoyées

Une valeur booléenne indiquant si la plate-forme est Windows ou Macintosh ; sur Macintosh, la valeur indique si une autre session d'optimisation de Fireworks est déjà en cours d'exécution.

FWLaunch.optimizeInFireworks()

Disponibilité

Dreamweaver 2, Fireworks 2.

Description

Cette fonction ouvre une session d'optimisation de Fireworks pour l'image spécifiée.

Arguments

docURL, imageURL, {targetWidth}, {targetHeight}

- L'argument *docURL* est le chemin d'accès du document actif, exprimé sous la forme d'une URL de type file:///.
- L'argument *imageURL* est le chemin de l'image sélectionnée. Si le chemin est relatif, sa référence sera le chemin spécifié dans l'argument *docURL*.
- L'argument *targetWidth* (facultatif) définit la largeur par rapport à laquelle l'image doit être redimensionnée.
- L'argument *targetHeight* (facultatif) définit la hauteur par rapport à laquelle l'image doit être redimensionnée.

Valeurs renvoyées

Zéro (0) si une session d'optimisation de Fireworks a été lancée avec succès pour l'image spécifiée ; sinon, un code d'erreur non nul indiquant que l'une des erreurs suivantes s'est produite :

- Utilisation non valide, ce qui indique que l'argument *docURL*, l'argument *imageURL*, ou les deux, ont été spécifiés comme étant `null` ou sous forme d'une chaîne vide.
- Erreur d'E/S de fichier, ce qui indique que Fireworks ne peut pas créer de fichier réponse car le disque est saturé.
- Erreur de notification de Dreamweaver ; l'utilisateur n'exécute pas une version valide de Dreamweaver (version 2 ou ultérieure).
- Erreur lors du démarrage de Fireworks, ce qui indique que la fonction n'ouvre pas une version valide de Fireworks (version 2 ou ultérieure).
- L'utilisateur a annulé l'opération.

FWLaunch.validateFireworks()

Disponibilité

Dreamweaver 2, Fireworks 2.

Description

Cette fonction recherche la version spécifiée de Fireworks sur le disque dur de l'utilisateur.

Arguments

{versionNumber}

- L'argument *versionNumber* est un nombre à virgule flottante supérieur ou égal à 2 ; il est facultatif et représente la version requise de Fireworks. Si cet argument n'est pas défini, il prend par défaut la valeur 2.

Valeurs renvoyées

Une valeur booléenne indiquant si la version spécifiée de Fireworks a été trouvée.

Exemple

Le code suivant vérifie si Fireworks est installé :

```
if (FWLaunch.validateFireworks(6.0)){
  alert("Fireworks 6.0 or later is installed.");
}else{
  alert("Fireworks 6.0 is not installed.");
}
```

Un exemple simple de communication FWLaunch

La commande suivante demande à Fireworks d'inviter l'utilisateur à entrer son nom, puis renvoie le nom à Dreamweaver.

```
<html>
<head>
<title>Prompt in Fireworks</title>
<meta http-equiv="Content-Type" content="text/html; ¬
charset=iso-8859-1">
<script>

function commandButtons(){
  return new Array("Prompt", "promptInFireworks()", "Cancel", ¬
  "readyToCancel()", "Close","window.close()");
}

var gCancelClicked = false;
var gProgressTrackerCookie = null;
```

```
function readyToCancel() {
  gCancelClicked = true;
}

function promptInFireworks() {
  var isFireworks3 = FWLaunch.validateFireworks(3.0);
  if (!isFireworks3) {
    alert("You must have Fireworks 3.0 or later to use this ¬
    command");
  return;
}

  // Tell Fireworks to execute the prompt() method.
  gProgressTrackerCookie = FWLaunch.execJsInFireworks¬
  ("prompt('Please enter your name:')");

  // null means it wasn't launched, a number means an error code
  if (gProgressTrackerCookie == null || ¬
  typeof(gProgressTrackerCookie) == "number") {
    window.close();
    alert("an error occurred");
    gProgressTrackerCookie = null;
  } else {
    // bring Fireworks to the front
    FWLaunch.bringFWToFront();
    // start the checking to see if Fireworks is done yet
    checkOneMoreTime();
  }
}

function checkOneMoreTime() {
  // Call checkJsResponse() every 1/2 second to see if Fireworks
   // is done yet
  window.setTimeout("checkJsResponse();", 500);
}

function checkJsResponse() {
  var response = null;

  // The user clicked the cancel button, close the window
  if (gCancelClicked) {
    window.close();
    alert("cancel clicked");
  } else {
    // We're still going, ask Fireworks how it's doing
    if (gProgressTrackerCookie != null)
      response = ¬
      FWLaunch.getJsResponse(gProgressTrackerCookie);
```

```
    if (response == null) {
      // still waiting for a response, call us again in 1/2 a
      // second
      checkOneMoreTime();
    } else if (typeof(response) == "number") {
      // if the response was a number, it means an error
      // occurred
      // the user cancelled in Fireworks
      window.close();
      alert("an error occurred.");

    } else {
      // got a valid response!  This return value might not
      // always be a useful one, since not all functions in
      // Fireworks return a string, but we know this one does,
      // so we can show the user what we got.
      window.close();
      FWLaunch.bringDWToFront();  // bring Dreamweaver to the ¬
      front
      alert("Nice to meet you, " + response + "!");
    }
  }
}

</script>
</head>
<body>
<form>
<table width="313" nowrap>
<tr>
<td>This command asks Fireworks to execute the prompt() ¬
function. When you click Prompt, Fireworks comes forward and ¬
asks you to enter a value into a dialog box. That value is then ¬
returned to Dreamweaver and displayed in an alert.</td>
</tr>
</table>
</form>
</body>
</html>
```

CHAPITRE 5

Intégration de Flash

5

Macromedia Dreamweaver 8 prend maintenant en charge les éléments Flash, en plus de l'API d'objet Flash, qui s'appuie sur le modèle Flash Generator pour créer de nouveaux objets Flash. Ce chapitre décrit comment utiliser les éléments Flash (fichiers SWC), et explique en détail comment créer des objets Flash (fichiers SWF) à partir des modèles Flash Generator (fichiers SWT).

Pour plus d'informations sur l'ajout de contenu Flash à des objets ou des commandes Dreamweaver, voir *Extension de Dreamweaver*.

Fonctionnement des éléments Flash

Les éléments Flash se présentent sous forme de fichiers SWC. Un fichier SWC est un clip vidéo compilé généré par Flash pour être utilisé par Macromedia et des produits tiers. Dreamweaver peut rendre ces composants accessibles aux utilisateurs via la barre Insérer, le menu Insérer ou une barre d'outils. Vous créez des éléments Flash à l'aide de l'outil de création Web Flash, mais Dreamweaver peut analyser les propriétés d'un élément Flash et les exprimer via la balise `param` (enfant de la balise `object`). Les utilisateurs peuvent ensuite modifier les attributs de la balise `param`, de façon à changer les propriétés de l'élément au moment de l'édition (pour plus d'informations sur l'utilisation des propriétés des composants dans Dreamweaver, voir *Utilisation de Dreamweaver*).

Insertion d'éléments Flash

Les éléments Flash sont installés via Extension Manager. Dreamweaver ajoute des éléments Flash aux documents de la même façon que les objets de la barre Insérer ou du menu Insérer (pour plus d'informations sur l'utilisation d'objets Dreamweaver, voir Objets de la barre Insérer dans *Extension de Dreamweaver*). Les utilisateurs peuvent ajouter des chaînes de code aux documents en cliquant sur les objets de la barre Insérer ou en sélectionnant les options dans le menu Insérer. Les utilisateurs peuvent accéder aux éléments Flash via la barre Insérer ou le menu Insérer (ce qui signifie que vous pouvez ajouter à la barre Insérer ou au menu Insérer un fichier d'élément Flash valide déjà installé dans le dossier Configuration/Objects/FlashElements ou l'un de ses sous-dossiers). Les développeurs d'extensions peuvent utiliser la fonction JavaScript dom.insertFlashElement() du fichier de définition de l'objet pour ajouter des éléments Flash disponibles à un document. Lorsque l'utilisateur sélectionne l'objet de l'élément Flash, Dreamweaver ouvre le fichier SWC, qui contient le contenu Flash (fichier SWF) et un fichier détaillant les paramètres que l'utilisateur peut modifier. Dreamweaver insère ensuite le fichier SWF dans le document de l'utilisateur.

Ajout d'un élément Flash à la barre Insérer

De même que pour tout autre objet, l'ajout d'un élément Flash à la barre Insérer se fait via la balise `button`. Toutefois, une balise `button` pour un élément Flash doit avoir des attributs `file` et `command` pour être correctement ajoutée au document (pour plus d'informations sur la balise button, voir Objets de la barre Insérer dans *Extension de Dreamweaver*). Les attributs `file` permettent d'indiquer à Dreamweaver où se trouve le fichier source de l'élément par rapport au dossier Objects. Utilisez ensuite l'attribut `command` pour indiquer à Dreamweaver d'utiliser la fonction `dom.insertFlashElement()` lorsque l'utilisateur clique sur le bouton de la barre Insérer.

L'exemple suivant représente le code qui doit être écrit dans le fichier inserbar.xml (en tant qu'enfant de la balise `category` ou `menubutton` appropriée, selon l'endroit où vous souhaitez voir apparaître le bouton de l'élément Flash) :

```
<button id="FlashElement_Nav"
name="Navigation"
file="FlashElements\nav.swc"
command="dw.getDocumentDOM().insertFlashElement('nav.swc')" />
```

REMARQUE L'image sur la barre Insérer pour l'élément Flash est déterminée au sein du fichier SWC. En outre, la balise *`button`* pour un élément Flash doit avoir un attribut de fichier défini.

Ajout d'un élément Flash à un menu

Un élément Flash peut également être situé dans le menu Insérer ou dans d'autres menus de Dreamweaver. Utilisez la fonction JavaScript dom.insertFlashElement() avec le format de fichier menus.xml (voir Menus et commandes de menus dans *Extension de Dreamweaver*) pour indiquer l'emplacement de l'élément de menu Flash. L'exemple suivant est un code qui permet, au sein du fichier menus.xml, d'intégrer l'élément Navigation Flash dans le menu Insertion > Elément Flash :

```
<menuitem name="Navigation"
key=""command="dw.getDocumentDOM().insertFlashElement('nav.swc')"
enabled="(dw.getFocus() != 'browser') && (dw.getDocumentDOM() != null && ¬
  dw.getDocumentDOM().getParseMode() == 'html')"
id="DWMenu_Insert_FlashElement_Nav" />
```

L'API des objets Flash

L'API des objets Flash permet aux développeurs d'extensions de construire des objets pour créer un contenu Macromedia Flash simple via Flash Generator. Cette API fournit un moyen de définir des paramètres dans un modèle Flash Generator pour réaliser un fichier SWF ou d'image. Elle permet de créer de nouveaux objets Flash mais aussi de lire et de manipuler des objets Flash existants. Les fonctionnalités bouton Flash et texte Flash sont construites à l'aide de cette API.

Le fichier de modèle Flash Generator SWT contient toutes les informations indispensables à la construction d'un fichier d'objet Flash. Ces fonctions d'API vous permettent de créer un nouveau fichier SWF (ou fichier d'image) à partir d'un fichier SWT en remplaçant les paramètres du fichier SWT par des valeurs réelles. Pour plus d'informations sur Flash, voir le manuel correspondant. Les fonctions suivantes sont des méthodes de l'objet `SWFFile`.

SWFFile.createFile()

Description

Cette fonction génère un nouveau fichier Objet Flash à partir du modèle et du tableau des paramètres spécifiés. Elle crée également une version GIF, PNG, JPEG et MOV du titre si les noms de fichier sous ces formats sont précisés.

Pour pouvoir spécifier un paramètre facultatif placé après des paramètres facultatifs que vous ne voulez pas spécifier, vous devez insérer des chaînes vides dans ces paramètres. Par exemple, si vous souhaitez spécifier un fichier PNG, mais pas de fichier GIF, vous devez insérer une chaîne vide avant de spécifier le nom du fichier PNG.

Arguments

templateFile, *templateParams*, *swfFileName*, *{gifFileName}*, *{pngFileName}*, *{jpgFileName}*, *{movFileName}*, *{generatorParams}*

- L'argument *templateFile* est le chemin d'accès du fichier modèle, exprimé sous la forme d'une URL de type file://. Il peut s'agir d'un fichier SWT.
- L'argument *templateParams* est un tableau de paires nom/valeur dans lequel les noms identifient les paramètres du fichier SWT et les valeurs correspondent à la définition que vous voulez leur attribuer. Pour que Dreamweaver puisse reconnaître un fichier SWF comme objet Flash, le premier paramètre doit être `"dwType"`. Sa valeur doit être une chaîne représentant le nom du type d'objet, telle que `"Flash Text"`.
- L'argument *swfFileName*, exprimé sous forme d'une URL de type file://, correspond au nom de fichier de sortie d'un fichier SWF ou d'une chaîne vide à ignorer.
- L'argument *gifFileName*, exprimé sous forme d'une URL de type file://, correspond au nom de fichier de sortie d'un fichier GIF. Cet argument est facultatif.
- L'argument *pngFileName*, exprimé sous forme d'une URL de type file://, correspond au nom de fichier de sortie d'un fichier PNG. Cet argument est facultatif.
- L'argument *jpgFileName*, exprimé sous forme d'une URL de type file://, correspond au nom de fichier de sortie d'un fichier JPEG. Cet argument est facultatif.
- L'argument *movFileName*, exprimé sous forme d'une URL de type file://, correspond au nom de fichier de sortie d'un fichier QuickTime. Cet argument est facultatif.
- L'argument *generatorParams* est un tableau de chaînes représentant les indicateurs facultatifs de la ligne de commande de Generator. Cet argument est facultatif. Les éléments de données de toutes les balises doivent suivre dans le tableau. Les indicateurs couramment utilisés sont répertoriés dans le tableau suivant :

Indicateur d'option	Données	Description	Exemple
`-defaultsize`	Largeur, hauteur	Définit la taille de l'image de sortie en fonction des largeur et hauteur indiquées.	`"-defaultsize", "640", "480"`
`-exactFit`	Aucun	Etend le contenu de l'image de sortie pour qu'il s'adapte exactement à la taille de sortie indiquée.	`"-exactFit"`

Valeurs renvoyées

Chaîne qui contient l'une des valeurs suivantes :

- `"noError"` signifie que l'appel s'est déroulé sans anomalie.

- `"invalidTemplateFile"` signifie que le fichier de modèle choisi était incorrect ou introuvable.
- `"invalidOutputFile"` signifie qu'au moins un des noms de fichier de sortie spécifiés était incorrect.
- `"invalidData"` signifie qu'une ou plusieurs des paires nom/valeur *`templateParams`* étaient incorrectes.
- `"initGeneratorFailed"` signifie que Generator n'a pas pu être initialisé.
- `"outOfMemory"` signifie que l'opération n'a pas pu se terminer correctement faute de mémoire.
- `"unknownError"` signifie qu'une erreur inconnue s'est produite.

Exemple

Le code JavaScript suivant crée un fichier objet Flash de type `"monType"`, qui remplace toutes les occurrences de la chaîne `"text"` à l'intérieur du fichier de modèle par la chaîne `"Hello World"`. Il crée un fichier GIF et un fichier SWF.

```
var params = new Array;
params[0] = "dwType";
params[1] = "myType";
params[2] = "text";
params[3] = "Hello World";
errorString = SWFFile.createFile( "file:///MyMac/test.swt", ¬
params, "file:///MyMac/test.swf", "file:///MyMac/test.gif");
```

SWFFile.getNaturalSize()

Description

Cette fonction renvoie la taille naturelle de tout contenu Flash.

Arguments

`fileName`

- L'argument *`fileName`*, exprimé sous la forme d'une URL de type file://, correspond au chemin d'accès au contenu Flash.

Valeurs renvoyées

Un tableau contenant deux éléments qui représentent la largeur et la hauteur du contenu Flash ou une valeur `null` si le fichier n'est pas un fichier Flash.

SWFFile.getObjectType()

Description

Cette fonction renvoie le type d'objet Flash, c'est-à-dire la valeur transmise dans le paramètre `dwType` lorsque le fichier a été créé par la fonction `SWFFile.createFile()`.

Arguments

`fileName`

- L'argument *`fileName`*, exprimé sous la forme d'une URL de type file://, correspond au chemin d'accès à un fichier Objet Flash. Il s'agit généralement d'un fichier SWF.

Valeurs renvoyées

Une chaîne représentant le type d'objet ou contenant la valeur `null` si le fichier n'est pas un fichier Objet Flash ou s'il est introuvable.

Exemple

Le code suivant vérifie si le fichier test.swf est un objet Flash de type `monType` :

```
if ( SWFFile.getObjectType("file:///MyMac/test.swf") == ¬
"myType" ){
  alert ("This is a myType object.");
}else{
  alert ("This is not a myType object.");
}
```

SWFFile.readFile()

Description

Cette fonction lit un fichier Objet Flash.

Arguments

`fileName`

- L'argument *`fileName`*, exprimé sous la forme d'une URL de type file://, correspond au chemin d'accès à un fichier Objet Flash.

Valeurs renvoyées

Un tableau de chaînes dans lequel le premier élément est le chemin d'accès complet du fichier modèle SWT. Les chaînes suivantes représentent les paramètres (paires nom/valeur) de l'objet. Dans le tableau, chaque nom est suivi de sa valeur. La première paire nom/valeur est `"dwType"`, suivie par sa valeur. La fonction renvoie une valeur `null` si le fichier est introuvable ou s'il ne s'agit pas d'un fichier Objet Flash.

Exemple

L'appel du code `var params = SWFFile.readFile("file:///MyMac/test.swf")` renvoie les valeurs suivantes dans le tableau des paramètres :

```
"file:///MyMac/test.swt"  // template file used to create this .swf file
"dwType"                  // first parameter
"myType"                  // first parameter value
"text"                    // second parameter
"Hello World"             // second parameter value
```

CHAPITRE 6

API de base de données

6

Les fonctions de l'API de base de données permettent de gérer les connexions aux bases de données et d'accéder aux informations stockées dans ces dernières. L'API de base de données est divisé en deux fonctions distinctes : la gestion des connexions et l'accès aux bases de données.

La gestion des connexions aux bases de données permet, par exemple, d'obtenir le nom d'utilisateur et le mot de passe nécessaires pour établir une connexion à une base de données ou d'ouvrir une boîte de dialogue de connexion à une base de données.

L'accès aux informations de base de données permet par exemple d'extraire les métadonnées qui décrivent le schéma ou la structure d'une base de données. Ces métadonnées incluent des informations telles que les noms des tables, des colonnes, des procédures stockées et des affichages. Vous pouvez également afficher les résultats de l'exécution d'une requête de base de données ou d'une procédure stockée. Lorsque vous accédez à une base de données par le biais de cette API, vous utilisez des instructions SQL (Structured Query Language).

Les fonctions de l'API de base de données sont utilisées au moment de la conception, lorsque les utilisateurs développent leurs applications Web, et non au moment de l'exécution, c'est-à-dire lorsque l'application Web est déployée.

Il est possible d'utiliser ces fonctions dans n'importe quelle extension. En fait, les fonctions des API de comportement de serveur, de format des données et de source de données de Macromedia Dreamweaver 8 utilisent toutes des fonctions de base de données.

Fonctionnement de l'API de bases de données

L'exemple suivant illustre la façon dont la fonction de comportement de serveur, `getDynamicBindings()`, est définie pour Recordset.js. Cet exemple utilise la fonction `MMDB.getColumnAndTypeList()` :

```
function getDynamicBindings(ss)
{
```

```
  var serverModel = dw.getDocumentDOM().serverModel.getServerName();
  var bindingsAndTypeArray = new Array();
  var connName  =  ss.connectionName;
  var statement = ss.source;
  var rsName    = ss.rsName;

  // remove SQL comments
  statement = statement.replace(/\/\*[\S\s]*?\*\//g, " ");
  var bIsSimple = ParseSimpleSQL(statement);
  statement = stripCFIFSimple(statement);

  if (bIsSimple) {
  statement = RemoveWhereClause(statement,false);
  } else {
  var pa = new Array();

    if (ss.ParamArray != null) {
    for (var i = 0; i < ss.ParamArray.length; i++) {
      pa[i] = new Array();
      pa[i][0] = ss.ParamArray[i].name;
      pa[i][1] = ss.ParamArray[i].value;
    }
    }

  var statement = replaceParamsWithVals(statement, pa, serverModel);
}
  bindingsAndTypeArray = MMDB.getColumnAndTypeList(connName, statement);
  return bindingsAndTypeArray;
}
```

Fonctions de connexion à une base de données

Les fonctions de connexion à une base de données vous permettent d'établir et de gérer toutes les connexions, y compris les connexions ADO de Dreamweaver, ColdFusion et JDBC. Ces fonctions interagissent avec le Gestionnaire de connexions uniquement ; elles n'accèdent pas aux bases de données. Pour les fonctions qui accèdent aux bases de données, voir *Fonctions d'accès à la base de données*, page 89.

MMDB.deleteConnection()

Disponibilité

Dreamweaver MX.

Description

Cette fonction permet de supprimer la connexion à la base de données nommée.

Arguments

connName

- L'argument *connName* est le nom de la connexion à la base de données tel qu'il est spécifié dans le Gestionnaire de connexions. Cet argument identifie la connexion à la base de données à supprimer en fonction de son nom.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant supprime une connexion à une base de données :

```
function clickedDelete()
{
  var selectedObj = dw.serverComponents.getSelectedNode();
  if (selectedObj && selectedObj.objectType=="Connection")
  {
    var connRec = MMDB.getConnection(selectedObj.name);
    if (connRec)
    {
      MMDB.deleteConnection(selectedObj.name);
      dw.serverComponents.refresh();
    }
  }
}
```

MMDB.getColdFusionDsnList()

Disponibilité

Dreamweaver UltraDev 4.

Description

Cette fonction extrait les noms des sources de données (DSN) ColdFusion du serveur du site, en utilisant les fonctions `getRDSUserName()` et `getRDSPassword()`.

Arguments

Aucun.

Valeurs renvoyées

Tableau contenant les DSN ColdFusion définis sur le serveur pour le site en cours.

MMDB.getConnection()

Disponibilité

Dreamweaver UltraDev 4, améliorée dans la version Dreamweaver MX.

Description

Extrait un objet de connexion nommé.

Arguments

name

- L'argument *name* est une variable de chaîne qui spécifie le nom de la connexion à laquelle vous souhaitez faire référence.

Valeurs renvoyées

Référence à un objet de connexion nommé. Les objets de connexion ont les propriétés suivantes :

Propriété	Description
`name`	Nom de connexion
`type`	Indique, en cas de `useHTTP false`, quel fichier DLL utiliser pour se connecter à une base de données en exécution.
`string`	Chaîne de connexion ADO d'exécution ou URL JDBC
`dsn`	DSN ColdFusion
`driver`	Pilote JDBC d'exécution
`username`	Nom d'utilisateur d'exécution
`password`	Mot de passe d'exécution
`useHTTP`	Chaîne qui contient la valeur `true` ou `false`, indiquant si vous devez utiliser un pilote distant (connexion HTTP) au moment de la conception ou un pilote local (DLL).
`includePattern`	Expression régulière utilisée pour trouver l'instruction d'inclusion de fichier sur la page pendant Live Data et Aperçu dans le navigateur
`variables`	Tableau de noms de variables de pages et leurs valeurs correspondantes, utilisé pendant Live Data et Aperçu dans le navigateur
`catalog`	Utilisé pour restreindre les métadonnées qui apparaissent (pour plus d'informations, voir *MMDB.getProcedures()*, page 93)
`schema`	Utilisé pour restreindre les métadonnées qui apparaissent (pour plus d'informations, voir *MMDB.getProcedures()*, page 93)
`filename`	Nom de fichier de boîte de dialogue qui était utilisé pour créer la connexion

REMARQUE Il s'agit des propriétés standard implémentées par Dreamweaver. Les développeurs peuvent définir leurs propres types de connexion et ajouter de nouvelles propriétés à cet ensemble standard, ou bien fournir un ensemble différent de propriétés.

MMDB.getConnectionList()

Disponibilité

Dreamweaver UltraDev 1.

Description

Cette fonction extrait une liste de toutes les chaînes de connexion définies dans le Gestionnaire de connexions.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes, chaque chaîne correspondant au nom d'une connexion tel qu'il apparaît dans le Gestionnaire de connexions.

Exemple

La fonction `MMDB.getConnectionList()` peut renvoyer les chaînes `["EmpDB", "Test", "TestEmp"]`.

MMDB.getConnectionName()

Disponibilité

Dreamweaver UltraDev 1.

Description

Extrait un nom de connexion correspondant à la chaîne de connexion spécifiée.
Cette fonction est utile lorsque vous devez resélectionner un nom de connexion dans l'interface utilisateur (IU) à partir des données de la page.

Si vous avez une chaîne de connexion faisant référence à deux pilotes différents, vous pouvez spécifier à la fois la chaîne de connexion et le pilote correspondant au nom de connexion que vous souhaitez obtenir. Par exemple, vous pourriez avoir deux connexions :

- Connexion 1 possède les propriétés suivantes :

  ```
  ConnectionString="jdbc:inetdae:velcro-qa-5:1433?database=pubs"
  DriverName="com.inet.tds.TdsDriver"
  ```

- Connexion 2 possède les propriétés suivantes :

  ```
  ConnectionString="jdbc:inetdae:velcro-qa-5:1433?database=pubs"
  DriverName="com.inet.tds.TdsDriver2"
  ```

Les chaînes de connexion de Connexion 1 et Connexion 2 sont identiques. Connexion 2 établit une connexion avec une version plus récente de `TdsDriver`. Vous devez transmettre le nom du pilote à cette fonction pour définir complètement le nom de connexion que vous souhaitez obtenir.

Arguments

`connString, {driverName}`

- L'argument *`connString`* est la chaîne de connexion qui extrait le nom de la connexion.
- L'argument facultatif *`driverName`* définit l'argument *`connString`* de manière plus précise.

Valeurs renvoyées

Chaîne de nom de connexion correspondant à la chaîne de connexion.

Exemple

Le code suivant renvoie la chaîne `"EmpDB"` :

```
var connectionName = MMDB.getConnectionName ¬
("dsn=EmpDB;uid=;pwd=");
```

MMDB.getConnectionString()

Disponibilité

Dreamweaver UltraDev 1.

Description

Extrait la chaîne de connexion associée à la connexion nommée.

Arguments

`connName`

- L'argument *`connName`* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.

Valeurs renvoyées

Chaîne de connexion qui correspond à la connexion nommée.

Exemple

Le code `var connectionString = MMDB.getConnectionString ("EmpDB")` renvoie différentes chaînes pour une connexion ADO ou JDBC.

- Pour une connexion ADO, la chaîne suivante peut renvoyer :

  ```
  "dsn=EmpDB;uid=;pwd=";
  ```

- Pour une connexion JDBC, la chaîne suivante peut renvoyer :

  ```
  "jdbc:inetdae:192.168.64.49:1433?database=pubs&user=JoeUser&¬
  password=joesSecret"
  ```

MMDB.getDriverName()

Disponibilité

Dreamweaver UltraDev 1.

Description

Extrait le nom du pilote associé à la connexion spécifiée. Seules les connexions JDBC ont des noms de pilote.

Arguments

connName

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.

Valeurs renvoyées

Chaîne contenant le nom du pilote.

Exemple

L'instruction `MMDB.getDriverName ("EmpDB");` peut renvoyer la chaîne suivante :

```
"jdbc/oracle/driver/JdbcOracle"
```

MMDB.getDriverUrlTemplateList() (déconseillée)

Disponibilité

Dreamweaver UltraDev 4, déconseillée dans Dreamweaver MX.

REMARQUE	Pour Dreamweaver UltraDev 4, la liste des pilotes JDBC est stockée dans le fichier connections.xml résidant dans le dossier Configuration/Connections. Tous les pilotes ont un modèle d'URL associé. Cette fonction renvoie la liste des pilotes JDBC.

Pour Dreamweaver MX (ou version ultérieure), ces pilotes et les modèles d'URL sont codés dans les boîtes de dialogue JDBC. En outre, cette fonction est une définition de fonction vide utilisée pour éliminer les erreurs de fonctions non définies. L'exemple suivant indique la manière dont un lecteur JDBC et un modèle d'URL sont codés :

```
var DEFAULT_DRIVER = "COM.ibm.db2.jdbc.app.DB2Driver";
var DEFAULT_TEMPLATE = "jdbc:db2:[database name]";
```

Dreamweaver dispose d'une boîte de dialogue pour chaque paire pilote/modèle d'URL.

En résumé, les développeurs qui utilisent Dreamweaver UltraDev 4 doivent ajouter une nouvelle entrée à XML et ceux qui utilisent Dreamweaver MX (ou version supérieure) doivent mettre en œuvre une nouvelle boîte de dialogue.

Description

Extrait les pilotes JDBC et les modèles d'URL respectifs.

Arguments

Aucun.

Valeurs renvoyées

Tableau contenant les pilotes JDBC détectés sur le système de l'utilisateur et leurs modèles d'URL respectifs, s'ils sont spécifiés. Le tableau dispose d'un nombre pair d'éléments contenant : `Driver1`, `UrlTemplate1`, `Driver2`, `UrlTemplate2`, etc.

MMDB.getLocalDsnList()

Disponibilité

Dreamweaver UltraDev 4.

Description

Extrait les DSN ODBC définis dans le système de l'utilisateur.

Arguments

Aucun.

Valeurs renvoyées

Tableau contenant les DSN ODBC définis sur le système de l'utilisateur.

MMDB.getPassword()

Disponibilité

Dreamweaver UltraDev 1.

Description

Extrait le mot de passe utilisé pour la connexion spécifiée.

Arguments

connName

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.

Valeurs renvoyées

Une chaîne de mot de passe associée au nom de connexion.

Exemple

L'instruction `MMDB.getPassword ("EmpDB");` peut renvoyer `"joessecret"`.

MMDB.getRDSPassword()

Disponibilité

Dreamweaver UltraDev 4.

Description

Extrait le mot de passe Remote Development Services (RDS) (à utiliser avec les connexions ColdFusion).

Arguments

Aucun.

Valeurs renvoyées

Chaîne contenant le mot de passe RDS.

MMDB.getRDSUserName()

Disponibilité

Dreamweaver UltraDev 4.

Description

Extrait le nom d'utilisateur RDS (à utiliser avec les connexions ColdFusion).

Arguments

Aucun.

Valeurs renvoyées

Une chaîne contenant le nom d'utilisateur RDS.

MMDB.getRemoteDsnList()

Disponibilité

Dreamweaver UltraDev 4, améliorée dans la version Dreamweaver MX.

Description

Extrait les DSN ODBC du serveur de site. Les fonctions `getRDSUserName()` et `getRDSPassword()` sont utilisées lorsque le modèle de serveur du site en cours est ColdFusion. Cette fonction offre aux développeurs la possibilité de spécifier une chaîne de paramètre URL à annexer à l'URL Remote Connectivity générée par `MMDB.getRemoteDsnList()`. Si le développeur fournit une chaîne de paramètre, cette fonction la transmet aux scripts de connectivité HTTP.

Arguments

`[urlParams]`

- L'argument facultatif *`urlParams`* est une chaîne contenant une liste des expressions *`name=value`*, séparées par des esperluettes (&). Les valeurs ne doivent pas être entourées de guillemets. Certains caractères, tels que l'espace dans la valeur « `Hello World` », doivent être codés. Voici un exemple d'argument valide que vous pouvez transmettre dans `MMDB.getRemoteDsnList()` :

  ```
  a=1&b=Hello%20World
  ```

Valeurs renvoyées

Renvoie un tableau contenant les DSN ODBC définis sur le serveur pour le site en cours.

MMDB.getRuntimeConnectionType()

Disponibilité

Dreamweaver UltraDev 1.

Description

Renvoie le type de connexion d'exécution du nom de connexion spécifié.

Arguments

connName

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.

Valeurs renvoyées

Chaîne correspondant au type de connexion. Cette fonction peut renvoyer une des valeurs suivantes : `"ADO"`, `"ADODSN"`, `"JDBC"` ou `"CFDSN"`.

Exemple

Le code suivant renvoie la chaîne `"ADO"` pour une connexion ADO :

```
var connectionType = MMDB.getRuntimeConnectionType ("EmpDB")
```

MMDB.getUserName()

Disponibilité

Dreamweaver UltraDev 1.

Description

Renvoie un nom d'utilisateur pour la connexion spécifiée.

Arguments

connName

- *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.

Valeurs renvoyées

Chaîne de nom d'utilisateur associée au nom de connexion.

Exemple

L'instruction `MMDB.getUserName ("EmpDB");` peut renvoyer `"amit"`.

MMDB.hasConnectionWithName()

Disponibilité

Dreamweaver UltraDev 4.

Description

Détermine l'existence de la connexion d'un nom donné.

Arguments

`name`

- L'argument *`name`* est le nom de la connexion.

Valeurs renvoyées

Renvoie une valeur booléenne : `true` indique l'existence d'une connexion ayant le nom spécifié ; `false` indique le contraire.

MMDB.needToPromptForRdsInfo()

Disponibilité

Dreamweaver MX.

Description

Détermine si Dreamweaver doit ouvrir la boîte de dialogue des informations de connexion RDS.

Arguments

`bForce`

- L'argument *`bForce`* est une valeur booléenne ; `true` indique que l'utilisateur ayant précédemment annulé la boîte de dialogue RDS doit toujours être invité à saisir les informations de connexion RDS.

Valeurs renvoyées

Valeur booléenne : `true` indique que l'utilisateur doit être invité à saisir les informations de connexion RDS ; `false` indique le contraire.

MMDB.needToRefreshColdFusionDsnList()

Disponibilité

Dreamweaver MX.

Description

Ordonne au Gestionnaire de connexions de vider la mémoire cache et d'extraire la liste des sources des données ColdFusion du serveur d'application la prochaine fois qu'un utilisateur demande la liste.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

MMDB.popupConnection()

Disponibilité

Dreamweaver MX.

Description

Cette fonction appelle une boîte de dialogue de connexion. Cette fonction a les trois signatures suivantes :

- Si la liste d'arguments ne comporte que *`dialogFileName`* (une chaîne), `popupConnection()` provoque le lancement de la boîte de dialogue de connexion dans Dreamweaver, pour que vous puissiez y définir une nouvelle connexion.
- Si la liste d'arguments ne comporte que *`connRec`* (une référence de connexion), `popupConnection()` provoque le lancement de la boîte de dialogue de connexion en mode d'édition dans Dreamweaver, pour que vous puissiez y modifier la connexion nommée. Dans ce mode, la zone de texte du nom s'affiche en grisé.

- Si la liste d'arguments ne comporte que *connRec* et la valeur booléenne *bDuplicate*, popupConnection() provoque le lancement de la boîte de dialogue en mode dupliqué dans Dreamweaver. Dans ce mode, la zone de texte du nom s'affiche en grisé et les propriétés restantes sont copiées pour définir une connexion dupliquée.

Arguments

dialogFileName

ou

connRec

ou

connrec, bDuplicate

- L'argument *dialogFileName* est une chaîne qui contient le nom d'un fichier HTML résidant dans le dossier Configuration/Connections/*server-model*. Ce fichier HTML définit la boîte de dialogue qui crée une connexion. Il doit implémenter trois fonctions API JavaScript : findConnection(), inspectConnection() et applyConnection(). En général, vous créez un fichier JavaScript qui implémente ces fonctions, puis vous intégrez ce fichier au fichier HTML. (Pour plus d'informations sur la création d'une connexion, voir *API de connectivité à une base de données*, page 105).
- *connRec* est une référence à un objet de connexion existant.
- *bDuplicate* est une valeur booléenne.

Valeurs renvoyées

Aucune. La boîte de dialogue de connexion définie s'ouvre.

MMDB.setRDSPassword()

Disponibilité

Dreamweaver UltraDev 4.

Description

Définit le mot de passe RDS.

Arguments

password

- L'argument *password* est une chaîne contenant le mot de passe RDS.

Valeurs renvoyées

Aucune.

MMDB.setRDSUserName()

Disponibilité

Dreamweaver UltraDev 4.

Description

Définit le nom d'utilisateur RDS.

Arguments

username

- L'argument *username* est le nom d'un utilisateur RDS valide.

Valeurs renvoyées

Aucune.

MMDB.showColdFusionAdmin()

Disponibilité

Dreamweaver MX.

Description

Affiche la boîte de dialogue ColdFusion Administrator.

Arguments

Aucun.

Valeurs renvoyées

Aucune. La boîte de dialogue ColdFusion Administrator apparaît.

MMDB.showConnectionMgrDialog()

Disponibilité

Dreamweaver UltraDev 1.

Description

Affiche la boîte de dialogue Gestionnaire de connexions.

Arguments

Aucun.

Valeurs renvoyées

Aucune. La boîte de dialogue Gestionnaire de connexions s'affiche.

MMDB.showOdbcDialog()

Disponibilité

Dreamweaver UltraDev 4 (Windows uniquement).

Description

Affiche la boîte de dialogue d'administration ODBC système ou Administrateur de source de données ODBC.

Arguments

Aucun.

Valeurs renvoyées

Aucune. La boîte de dialogue d'administration ODBC système ou Administrateur de source de données ODBC apparaît.

MMDB.showRdsUserDialog()

Disponibilité

Dreamweaver UltraDev 4.

Description

Affiche la boîte de dialogue demandant le nom d'utilisateur et du mot de passe RDS.

Arguments

username, password

- L'argument *username* est le nom d'utilisateur initial.
- *password* est le mot de passe initial.

Valeurs renvoyées

Objet contenant les nouvelles valeurs dans les propriétés `username` et `password`. Si l'une des propriétés n'est pas définie, ceci indique que l'utilisateur a annulé la boîte de dialogue.

MMDB.showRestrictDialog()

Disponibilité

Dreamweaver UltraDev 4.

Description

Affiche la boîte de dialogue Restreindre.

Arguments

catalog, schema

- L'argument *catalog* est la valeur de catalogue initiale.
- L'argument *schema* est la valeur de schéma initiale.

Valeurs renvoyées

Objet contenant les nouvelles valeurs dans les propriétés `catalog` et `schema`. Si l'une des propriétés n'est pas définie, ceci indique que l'utilisateur a annulé la boîte de dialogue.

MMDB.testConnection()

Disponibilité

Dreamweaver UltraDev 4.

Description

Teste les paramètres de connexion. Affiche une boîte de dialogue modale qui décrit les résultats.

Arguments

serverPropertiesArray

Cette fonction attend un seul argument, un objet de tableau contenant les valeurs de la liste suivante adaptées au modèle de serveur en cours. Pour les propriétés qui ne s'appliquent pas à la connexion testée, laissez-les vides (“”).

- *type* indique, lorsque *useHTTP* est une valeur `false`, quelle DLL utiliser pour se connecter à une base de données au moment de la conception, pour tester les paramètres de connexion.
- *string* est la chaîne de connexion ADO ou l'URL JDBC.
- *dsn* est le nom de la source de données.
- *driver* est le pilote JDBC.
- *username* est le nom d'utilisateur.

- L'argument *password* est le mot de passe.
- L'argument *useHTTP* est une valeur booléenne. Une valeur `true` spécifie que Dreamweaver doit utiliser une connexion HTTP au moment de la conception ; dans le cas contraire, Dreamweaver utilise une DLL.

Valeurs renvoyées

Valeur booléenne : `true` si le test de connexion réussit ; `false` dans le cas contraire.

Fonctions d'accès à la base de données

Les fonctions d'accès à la base de données vous permettent de faire une recherche dans la base de données. Pour l'ensemble des fonctions qui gèrent une connexion à une base de données, voir *Fonctions de connexion à une base de données*, page 72.

La liste suivante décrit certains des arguments communs à toutes les fonctions disponibles :

- La plupart des fonctions d'accès à une base de données utilisent un nom de connexion comme argument. Pour obtenir une liste des noms de connexion valides, utilisez le Gestionnaire de connexions ou la fonction `MMDB.getConnectionList()`. Cette dernière vous permet d'obtenir par programmation une liste de tous les noms de connexion.
- Les procédures stockées exigent souvent des paramètres. Il existe deux façons de spécifier des valeurs de paramètre pour les fonctions d'accès aux bases de données. En premier lieu, vous pouvez fournir un tableau de valeurs de paramètre (*paramValuesArray*). Si vous ne spécifiez que des valeurs de paramètre, celles-ci doivent être dans l'ordre dans lequel la procédure stockée les demande. Spécifiez ensuite les valeurs de paramètre pour fournir un tableau des noms de paramètre (*paramNameArray*). Vous pouvez utiliser la fonction `MMDB.getSPParamsAsString()` pour obtenir les paramètres de la procédure stockée. Si vous fournissez des noms de paramètres, les valeurs spécifiées dans *paramValuesArray* doivent être dans l'ordre dans lequel les noms ont été spécifiés dans *paramNameArray*.

MMDB.getColumnAndTypeList()

Disponibilité

Dreamweaver UltraDev 1.

Description

Cette fonction extrait une liste de colonnes et leurs types respectifs d'une déclaration SQL `SELECT` exécutée.

Arguments

connName, statement

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *statement* est la déclaration SQL `SELECT` à exécuter.

Valeurs renvoyées

Un tableau de chaînes qui représente une liste de colonnes (et leur type) qui correspondent à la déclaration `SELECT`, ou bien une erreur si la déclaration SQL n'est pas valide ou si la connexion n'a pas pu s'établir.

Exemple

Le code `var columnArray = MMDB.getColumnAndTypeList("EmpDB","Select * from Employees")` renvoie le tableau de chaînes suivant :

```
columnArray[0] = "EmpName"
columnArray[1] = "varchar"
columnArray[2] = "EmpFirstName"
columnArray[3] = "varchar"
columnArray[4] = "Age"
columnArray[5] = "integer"
```

MMDB.getColumnList()

Disponibilité

Dreamweaver UltraDev 1.

Description

Cette fonction extrait la liste de colonnes d'une déclaration SQL `SELECT` exécutée.

Arguments

connName, statement

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *statement* est la déclaration SQL `SELECT` à exécuter.

Valeurs renvoyées

Un tableau de chaînes qui représente une liste de colonnes correspondant à la déclaration `SELECT`, ou une erreur si la déclaration SQL n'est pas valide ou si la connexion n'a pas pu s'établir.

Exemple

Le code `var columnArray = MMDB.getColumnList("EmpDB","Select * from Employees")` renvoie le tableau de chaînes suivant :

```
columnArray[0] = "EmpName"
columnArray[1] = "EmpFirstName"
columnArray[2] = "Age"
```

MMDB.getColumns()

Disponibilité

Dreamweaver MX, arguments mis à jour dans MX 2004.

Description

Renvoie un tableau d'objets qui décrivent les colonnes du tableau spécifié.

Arguments

`connName, tableName`

- L'argument *`connName`* est le nom de la connexion. Cette valeur identifie la connexion qui contient la chaîne que Dreamweaver doit utiliser pour établir une connexion de base de données à une source de données active.
- *`tableName`* est le tableau à interroger.

Valeurs renvoyées

Un tableau d'objets, à raison d'un objet par colonne. Chaque objet définit les trois propriétés suivantes pour la colonne à laquelle il est associé.

Nom de propriété	Description
`name`	Nom de la colonne (par exemple, `price`)
`datatype`	Type de données de la colonne (par exemple, `small money`)
`definedsize`	Taille définie de la colonne (par exemple, `8`)
nullable	Indique si la colonne peut contenir des valeurs `null`

Exemple

L'exemple suivant utilise `MMDB.getColumns()` pour définir la valeur du texte de l'info bulle :

```
var columnNameObjs = MMDB.getColumns(connName,tableName);
var databaseType   = MMDB.getDatabaseType(connName);

  for (i = 0; i < columnNameObjs.length; i++)
  {
    var columnObj = columnNameObjs[i];
    var columnName = columnObj.name;
    var typename = columnObj.datatype;
    if (dwscripts.isNumber(typename))
    {
      // it already is a num
      typename = dwscripts.getDBColumnTypeAsString(typename,
  databaseType);
    }

    var tooltiptext = typename;
  }
```

MMDB.getColumnsOfTable()

Disponibilité

Dreamweaver UltraDev 1.

Description

Cette fonction extrait une liste de toutes les colonnes du tableau spécifié.

Arguments

connName, tableName

- *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *tableName* est le nom d'une table de la base de données désignée par l'argument *connName*.

Valeurs renvoyées

Un tableau de chaînes dont chaque chaîne est le nom d'une colonne du tableau.

Exemple

L'instruction `MMDB.getColumnsOfTable ("EmpDB","Employees");` renvoie les chaînes suivantes :

```
["EmpID", "FirstName", "LastName"]
```

MMDB.getPrimaryKeys()

Disponibilité

Dreamweaver MX.

Description

Renvoie les noms de colonne qui s'associent pour former la clé primaire de la table nommée. Une clé primaire sert d'identificateur unique pour une ligne de base de données et se compose d'une colonne minimum.

Arguments

connName, tableName

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *tableName* est le nom de la table pour laquelle vous souhaitez restituer l'ensemble des colonnes comprenant la clé primaire de cette table.

Valeurs renvoyées

Tableau de chaînes. Le tableau contient une chaîne pour chaque colonne comprenant la clé primaire.

Exemple

L'exemple suivant renvoie la clé primaire de la table spécifiée.

```
var connName    = componentRec.parent.parent.parent.name;
var tableName   = componentRec.name;
var primaryKeys = MMDB.getPrimaryKeys(connName,tableName);
```

MMDB.getProcedures()

Disponibilité

Dreamweaver MX.

Description

Cette fonction renvoie un tableau d'objets de procédure associés à une connexion nommée.

Arguments

connName

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.

Valeurs renvoyées

Un tableau d'objets de procédure dans lequel chaque objet de procédure a les trois propriétés suivantes :

Nom de propriété	Description
`schema*`	Nom du schéma associé à l'objet. Cette propriété identifie l'utilisateur associé à la procédure stockée dans la base de données SQL et à laquelle accède `getProcedures()`. La base de données à laquelle accède cette fonction dépend du type de connexion. • Pour les connexions ODBC, la source de données ODBC définit la base de données. Le DSN est spécifié par la propriété `dsn` dans l'objet de connexion (*connName*) que vous transmettez à la fonction `getProcedures()`. • Pour les connexion à la BD OLE, la chaîne de connexion donne un nom à la base de données.
`catalog`	Nom du catalogue associé à l'objet (qualificatif de propriétaire). La valeur de la propriété `catalog` est définie par un attribut du pilote de la BD OLE. Cet attribut du pilote définit une propriété `user.database` par défaut à utiliser lorsque la chaîne de connexion à la BD OLE n'indique pas de base de données.
`procedure`	Nom de la procédure.

* Dreamweaver permet de se connecter aux tables de la base de données et de les acquérir lorsque vous modifiez un ensemble d'enregistrements. Si la base de données comporte de nombreuses tables, leur extraction risque de prendre beaucoup de temps sur certains systèmes. Si la base de données contient un schéma ou un catalogue, vous pouvez l'utiliser pour filtrer le contenu de la base de données pendant le processus de conception par Dreamweaver. Les schémas et les catalogues doivent être créés au préalable, afin de permettre leur application dans Dreamweaver. Consultez la documentation de la base de données ou votre administrateur système.

Exemple

Le code suivant extrait une liste de procédures :

```
var procObjects   = MMDB.getProcedures(connectionName);
for (i = 0; i < procObjects.length; i++)
{
  var thisProcedure = procObjects[i]
  thisSchema =  Trim(thisProcedure.schema)
```

```
  if (thisSchema.length == 0)
  {
  thisSchema = Trim(thisProcedure.catalog)
  }
  if (thisSchema.length > 0)
  {
  thisSchema += "."
  }

  var procName = String(thisSchema + thisProcedure.procedure);
 }
```

MMDB.getSPColumnList()

Disponibilité

Dreamweaver UltraDev 1.

Description

Cette fonction extrait une liste de colonnes de jeu de résultats générées par un appel à la procédure stockée spécifiée.

Arguments

connName, statement, paramValuesArray

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *statement* est le nom de la procédure stockée qui renvoie le jeu de résultats lorsqu'elle est exécutée.
- L'argument *paramValuesArray* est un tableau contenant une liste de valeurs tests de paramètre au moment de la conception. Spécifiez les valeurs de paramètre dans l'ordre attendu par la procédure stockée. Vous pouvez utiliser la fonction `MMDB.getSPParamsAsString()` pour obtenir les paramètres de la procédure stockée.

Valeurs renvoyées

Tableau de chaînes représentant la liste des colonnes. Cette fonction renvoie une erreur lorsque l'instruction SQL ou la chaîne de connexion est incorrecte.

Exemple

Le code suivant pourrait renvoyer une liste de colonnes de jeux de résultats générées à partir de la procédure stockée exécutée, `getNewEmployeesMakingAtLeast` :

```
var paramValueArray = new Array("2/1/2000", "50000")
var columnArray = MMDB.getSPColumnList("EmpDB", ¬
"getNewEmployeesMakingAtLeast", paramValueArray)
```

Les valeurs suivantes renvoient :

```
columnArray[0] = "EmpID", columnArray[1] = "LastName", ¬
columnArray[2] ="startDate", columnArray[3] = "salary"
```

MMDB.getSPColumnListNamedParams()

Disponibilité

Dreamweaver UltraDev 1.

Description

Cette fonction extrait une liste de colonnes de jeu de résultats générées par un appel à la procédure stockée spécifiée.

Arguments

`connName, statement, paramNameArray, paramValuesArray`

- L'argument *`connName`* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *`statement`* est le nom de la procédure stockée qui renvoie le jeu de résultats lorsqu'elle est exécutée.
- L'argument *`paramNameArray`* est un tableau contenant une liste de noms de paramètres. Vous pouvez utiliser la fonction `MMDB.getSPParamsAsString()` pour obtenir les paramètres de la procédure stockée.
- L'argument *`paramValuesArray`* est un tableau contenant une liste de valeurs tests de paramètre au moment de la conception. Vous pouvez spécifier si oui ou non la procédure requiert des paramètres pendant l'exécution. Si vous avez fourni des noms de paramètre dans *`paramNameArray`*, spécifiez les valeurs de paramètre dans l'ordre dans lequel leurs noms apparaissent dans *`paramNameArray`*. Si vous n'avez pas indiqué *`paramNameArray`*, spécifiez les valeurs dans l'ordre attendu par la procédure stockée.

Valeurs renvoyées

Tableau de chaînes représentant la liste des colonnes. Cette fonction renvoie une erreur lorsque l'instruction SQL ou la chaîne de connexion est incorrecte.

Exemple

Le code suivant pourrait renvoyer une liste de colonnes de jeux de résultats générées à partir de la procédure stockée exécutée, `getNewEmployeesMakingAtLeast` :

```
var paramNameArray = new Array("startDate", "salary")
var paramValueArray = new Array("2/1/2000", "50000")
var columnArray = MMDB.getSPColumnListNamedParams("EmpDB", ¬
"getNewEmployeesMakingAtLeast", paramNameArray, paramValueArray)
```

Les valeurs suivantes renvoient :

```
columnArray[0] = "EmpID", columnArray[1] = "LastName",¬
columnArray[2] ="startDate", columnArray[3] = "salary"
```

MMDB.getSPParameters()

Disponibilité

Dreamweaver MX.

Description

Cette fonction renvoie un tableau d'objets de paramètre pour une procédure nommée.

Arguments

`connName, procName`

- L'argument *`connName`* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *`procName`* est le nom de la procédure.

Valeurs renvoyées

Tableau d'objets de paramètre, chacun d'entre eux spécifiant l'ensemble de propriétés suivant :

Nom de la propriété	Description
`name`	Nom du paramètre (par exemple, `@@lolimit`)
`datatype`	Type de données du paramètre (par exemple, `smallmoney`)
`direction`	Direction du paramètre : `1`- Le paramètre est utilisé uniquement pour l'entrée. `2`- Le paramètre est utilisé uniquement pour la sortie. Dans ce cas, vous transmettez le paramètre par référence et la méthode place une valeur dedans. Vous pouvez utiliser la valeur une fois la méthode renvoyée. `3`- Le paramètre est utilisé pour l'entrée et la sortie. `4`- Le paramètre contient une valeur de retour.

Exemple

L'exemple suivant extrait les objets de paramètre pour la procédure spécifiée et crée une info bulle pour chaque objet qui utilise ses propriétés.

```
var paramNameObjs = MMDB.getSPParameters(connName,procName);
for (i = 0; i < paramNameObjs.length; i++)
{
  var paramObj = paramNameObjs[i];
  var tooltiptext = paramObj.datatype;
  tooltiptext+=" ";
  tooltiptext+=GetDirString(paramObj.directiontype);
}
```

MMDB.getSPParamsAsString()

Disponibilité

Dreamweaver UltraDev 1.

Description

Cette fonction extrait une chaîne délimitée par virgules contenant la liste des paramètres pris par la procédure stockée.

Arguments

connName, procName

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *procName* est le nom de la procédure stockée.

Valeurs renvoyées

Chaîne délimitée par virgules contenant la liste des paramètres requis par la procédure stockée. Les noms, la direction et le type de données des paramètres sont inclus, séparés par des points-virgules (;).

Exemple

Le code `MMDB.getSPParamsAsString ("EmpDB","getNewEmployeesMakingAtLeast")` peut renvoyer une chaîne de nom de formulaire `startDate;direction:in;datatype:date, salary;direction:in;datatype:integer`

Dans cet exemple, la procédure stockée `getNewEmployeesMakingAtLeast` a deux paramètres : `startDate` et `Salary`. Pour `startDate`, la direction est `in` et le type de données est `date`. Pour `salary`, la direction est `in` et le type de données est `date`.

MMDB.getTables()

Disponibilité

Dreamweaver UltraDev 1.

Description

Extrait une liste de toutes les tables définies pour la base de données spécifiée. Chaque objet a trois propriétés : `table`, `schema` et `catalog`.

Arguments

`connName`

- L'argument *`connName`* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.

Valeurs renvoyées

Un tableau d'objets dans lequel chaque objet a trois propriétés : `table`, `schema` et `catalog`. `Table` est le nom de la table. `Schema` est le nom du schéma qui contient la table. `Catalog` est le catalogue qui contient la table.

Exemple

L'instruction `MMDB.getTables ("EmpDB");` pourrait produire un tableau de deux objets. Les propriétés du premier objet peuvent ressembler à l'exemple suivant :

```
object1[table:"Employees", schema:"personnel", catalog:"syscat"]
```

Les propriétés du deuxième objet peuvent ressembler à l'exemple suivant :

```
object2[table:"Departments", schema:"demo", catalog:"syscat2"]
```

MMDB.getViews()

Disponibilité

Dreamweaver UltraDev 4.

Description

Extrait une liste de tous les modes d'affichage définis pour la base de données spécifiée. Chaque objet mode d'affichage a les propriétés `catalog`, `schema` et `view`.

Arguments

connName

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.

Valeurs renvoyées

Un table d'objets d'affichage dans lequel chaque objet a trois propriétés : `catalog`, `schema` et `view`. `Catalog` ou `schema` permet de restreindre/filtrer le nombre de modes d'affichage rattachés à un nom de schéma individuel ou à un nom de catalogue défini comme faisant partie des informations de connexion.

Exemple

L'exemple suivant renvoie les modes pour une valeur de connexion donnée, `CONN_LIST.getValue()` :

```
var viewObjects = MMDB.getViews(CONN_LIST.getValue())
for (i = 0; i < viewObjects.length; i++)
{
  thisView = viewObjects[i]
  thisSchema = Trim(thisView.schema)
  if (thisSchema.length == 0)
  {
    thisSchema = Trim(thisView.catalog)
  }
  if (thisSchema.length > 0)
  {
    thisSchema += "."
  }
  views.push(String(thisSchema + thisView.view))
}
```

MMDB.showResultset()

Disponibilité

Dreamweaver UltraDev 1.

Description

Affiche une boîte de dialogue contenant les résultats de l'exécution de la déclaration SQL spécifiée. La boîte de dialogue contient une grille dont l'en-tête reflète les informations de colonnes qui décrivent le jeu de résultats. Si la chaîne de connexion ou l'instruction SQL n'est pas valide, une erreur apparaît. Cette fonction valide l'instruction SQL.

Arguments

connName, SQLstatement

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *SQLstatement* désigne l'instruction SQL `SELECT`.

Valeurs renvoyées

Aucune. Cette fonction renvoie une erreur lorsque l'instruction SQL ou la chaîne de connexion est incorrecte.

Exemple

Le code suivant affiche le résultat de l'instruction SQL exécutée :

```
MMDB.showResultset("EmpDB","Select EmpName,EmpFirstName,Age ¬
from Employees")
```

MMDB.showSPResultset()

Disponibilité

Dreamweaver UltraDev 1.

Description

Affiche une boîte de dialogue contenant les résultats de l'exécution de la procédure stockée spécifiée. La boîte de dialogue contient une grille dont l'en-tête reflète les informations de colonne qui décrivent le jeu de résultats. Si la chaîne de connexion ou la procédure stockée n'est pas valide, une erreur apparaît. Cette fonction valide la procédure stockée.

Arguments

connName, procName, paramValuesArray

- L'argument *connName* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *procName* est le nom de la procédure stockée à exécuter.
- *paramValuesArrayargument* est un tableau contenant une liste de valeurs tests de paramètre au moment de la conception. Spécifiez les valeurs de paramètre dans l'ordre attendu par la procédure stockée. Vous pouvez utiliser la fonction `MMDB.getSPParamsAsString()` pour obtenir les paramètres de la procédure stockée.

Valeurs renvoyées

Cette fonction renvoie une erreur lorsque l'instruction SQL ou la chaîne de connexion est incorrecte, sinon, elle ne renvoie rien.

Exemple

Le code suivant affiche le résultat de la procédure stockée exécutée :

```
var paramValueArray = new Array("2/1/2000", "50000")
MMDB.showSPResultset("EmpDB", "getNewEmployeesMakingAtLleast", ¬
paramValueArray)
```

MMDB.showSPResultsetNamedParams()

Disponibilité

Dreamweaver UltraDev 1.

Description

Affiche une boîte de dialogue contenant le jeu de résultats de la procédure stockée spécifiée. La boîte de dialogue contient une grille dont l'en-tête reflète les informations de colonne qui décrivent le jeu de résultats. Si la chaîne de connexion ou la procédure stockée n'est pas valide, une erreur apparaît. Cette fonction valide la procédure stockée. Cette fonction diffère de `MMDB.showSPResultset()`, car vous pouvez spécifier les valeurs de paramètre par leur nom, au lieu de l'ordre attendu par la procédure stockée.

Arguments

`connName, procName, paramNameArray, paramValuesArray`

- L'argument *`connName`* est un nom de connexion spécifié dans le Gestionnaire de connexions. Il identifie la chaîne de connexion que Dreamweaver doit utiliser pour connecter la base de données à une source de données active.
- L'argument *`procName`* est le nom de la procédure stockée qui renvoie le jeu de résultats lorsqu' elle est exécutée.
- L'argument *`paramNameArray`* est un tableau contenant une liste de noms de paramètres. Vous pouvez utiliser la fonction `MMDB.getSPParamsAsString()` pour obtenir les paramètres de la procédure stockée.
- L'argument *`paramValuesArray`* est un tableau contenant une liste de valeurs tests de paramètre au moment de la conception.

Valeurs renvoyées

Cette fonction renvoie une erreur lorsque l'instruction SQL ou la chaîne de connexion est incorrecte, sinon, elle ne renvoie rien.

Exemple

Le code suivant affiche le résultat de la procédure stockée exécutée :

```
var paramNameArray = new Array("startDate", "salary")
var paramValueArray = new Array("2/1/2000", "50000")
MMDB.showSPResultsetNamedParams("EmpDB","getNewEmployees¬
MakingAtLeast", paramNameArray, paramValueArray)
```

CHAPITRE 7

API de connectivité à une base de données

7

En tant que développeur, vous pouvez créer de nouveaux types de connexions et leurs boîtes de dialogue correspondantes pour les modèles de serveur nouveaux et existants de Macromedia Dreamweaver 8. Lorsqu'un utilisateur crée ensuite un site pour élaborer des pages, il ou elle crée un nouvel objet de connexion après avoir sélectionné le type particulier de connexion que vous avez créé.

L'utilisateur peut sélectionner votre nouveau type de connexion de plusieurs manières :

- Il peut cliquer sur le bouton Plus (+) et sélectionner Jeu d'enregistrements dans le panneau Application. Il peut ensuite agrandir la fenêtre de la liste Connexion dans la boîte de dialogue Jeu d'enregistrements.
- Il peut cliquer sur le bouton Plus (+) et sélectionner Nom de la source de données dans l'onglet Base de données du panneau Bases de données.

Développement d'un nouveau type de connexion

Les étapes suivantes expliquent le processus de création d'un nouveau type de connexion :

1. Définissez la mise en forme de la boîte de dialogue de connexion.

   Créez un fichier HTML mettant en forme l'interface utilisateur de votre boîte de dialogue de connexion. Donnez un nom à ce fichier en utilisant le nom de la connexion (par exemple myConnection.htm). Pour plus d'informations sur la création d'une boîte de dialogue, voir *Bien démarrer avec Dreamweaver*.

   Vérifiez que ce fichier HTML inclut une référence au fichier de mise en œuvre de JavaScript défini à l'étape 2, *Créez un fichier JavaScript qui implémente au moins les éléments suivants :*, page 106, comme le montre l'exemple suivant :

   ```
   <head>
     <script SRC="../myConnectionImpl.js"></script>
   </head>
   ```

Stockez ce fichier HTML, qui définit votre boîte de dialogue de connexion, dans le dossier Configuration/Connections/*server-model*/*platform* (où la *plate-forme* est Windows ou Macintosh).

Par exemple, la boîte de dialogue de connexion ADO par défaut pour un document ASP JavaScript sur une plate-forme Windows est stockée dans le dossier ASP_Js/Win et est intitulée Connection_ado_conn_string.htm.

REMARQUE	Pendant l'exécution, Macromedia Dreamweaver établit de manière dynamique la liste des types de connexion disponibles dans l'ensemble des boîtes de dialogue présentes dans le dossier ASP_Js/Win.

Le dossier Configuration/ServerModels contient des fichiers HTML qui définissent chaque modèle de serveur. A l'intérieur de chaque fichier HTML se trouve la fonction `getServerModelFolderName()`, qui renvoie le nom du dossier associé au modèle de serveur. L'exemple suivant indique la fonction pour le type de document ASP JavaScript :

```
function getServerModelFolderName()
{
  return "ASP_JS";
}
```

Vous pouvez également consulter le fichier MMDocumentTypes.xml, situé dans le dossier Configuration/DocumentTypes, pour déterminer la correspondance entre les modèles de serveur et les types de documents.

2. Créez un fichier JavaScript qui implémente au moins les éléments suivants :

Elément	Description	Exemples
Un ensemble de variables	Chacune d'entre elles définit une propriété de connexion spécifique	Type de connexion, nom de la source de données, etc.
Un ensemble de boutons	Tous les boutons apparaissent dans la boîte de dialogue de connexion.	Tester, Aide, etc. (OK et Annuler sont automatiquement inclus)

Fonctions de connectivité	Ensemble, ces fonctions définissent l'API de connectivité	`findConnection()`
		`applyConnection()`
		`inspectConnection()`

Vous pouvez choisir n'importe quel nom pour ce fichier de mise en œuvre, mais il doit comporter une extension .js (par exemple, myConnectionImpl.js). Vous pouvez stocker ce fichier de mise en œuvre sur votre ordinateur local ou distant. Si vous le souhaitez, vous pouvez également le stocker dans le sous-dossier approprié du dossier Configuration/Connections.

REMARQUE

Le fichier HTML défini à l'étape 1 *Définissez la mise en forme de la boîte de dialogue de connexion., page 105*, doit inclure ce fichier de mise en œuvre du type de connexion.

Ces deux étapes constituent les conditions minimales pour créer une nouvelle boîte de dialogue de connexion, sauf si vous avez besoin de définir des paramètres de connexion autres que ceux fournis dans le fichier standard connection_includefile.edml.

REMARQUE

Le titre de la boîte de dialogue que voit l'utilisateur se trouve dans la balise `title`, spécifiée dans le document HTML.

Les fonctions répertoriées dans la section suivante permettent de créer une boîte de dialogue de connexion. En plus d'implémenter les appels pour la génération des fichiers inclus réservés à l'utilisateur, vous pouvez enregistrer votre type de connectivité dans la section du modèle de serveur du fichier XML de connexion.

Pour plus d'informations sur l'API de connectivité à une base de données, associée à la création d'une nouvelle connexion, voir *Fonctions de connexion à une base de données, page 72*.

L'API de connexion

Pour créer un nouveau type de connexion, y compris la boîte de dialogue avec laquelle les utilisateurs interagissent, vous devez implémenter les trois fonctions suivantes : `findConnection()`, `inspectConnection()` et `applyConnection()`. Ecrivez ces trois fonctions et incluez-les dans le fichier de mise en œuvre JavaScript associé à votre nouveau type de connexion (voir l'étape 2 *Créez un fichier JavaScript qui implémente au moins les éléments suivants :, page 106*).

La fonction `applyConnection()` renvoie une source HTML dans un fichier inclus. Consultez les exemples de source HTML dans le *Fichier inclus généré*, page 111. La fonction `findConnection()` prend la source HTML et en extrait les propriétés. Vous pouvez mettre en œuvre `findConnection()` pour utiliser les modèles de recherche dans les fichiers XML afin d'extraire les informations renvoyées à partir de `applyConnection()`. Si vous souhaitez voir un exemple de ce type de mise en œuvre, étudiez les deux fichiers JavaScript suivants :

- `connection_ado_conn_string.js` se trouve dans le dossier Configuration/Connections/ASP_Js.
- `connection_common.js` se trouve dans le dossierConfiguration/Connections/Shared.

Lorsque l'utilisateur ouvre un site, Dreamweaver parcourt tous les fichiers dans le dossier Connections, les ouvre et transmet leur contenu à la fonction `findConnection()`. Si le contenu d'un fichier correspond aux critères d'une connexion valide, `findConnection()` renvoie un objet de connexion. Dreamweaver répertorie ensuite tous les objets de connexion dans le panneau Explorateur de base de données.

Lorsque l'utilisateur ouvre une boîte de dialogue de connexion et choisit de créer une nouvelle connexion ou de dupliquer ou encore de modifier une connexion existante, Dreamweaver déclenche la fonction `inspectConnection()` et retransmet le même objet de connexion créé par `findConnection()`. Ce processus permet à Dreamweaver de renseigner la boîte de dialogue en utilisant les informations de connexion.

Lorsque l'utilisateur clique sur OK dans une boîte de dialogue de connexion, Dreamweaver déclenche la fonction `applyConnection()` pour construire la page HTML, placée dans le fichier inclus de connexion résidant dans le dossier Configuration/Connections. La fonction `applyConnection()` renvoie une chaîne vide qui indique une erreur dans l'un des champs. La boîte de dialogue ne doit pas être fermée. Le fichier inclus a un type d'extension de fichier par défaut pour le modèle de serveur en cours.

Lorsque l'utilisateur ajoute à une page un comportement de serveur qui utilise la connexion, tel qu'un jeu d'enregistrements ou une procédure stockée, Dreamweaver ajoute une instruction à la page qui comprend le fichier inclus de connexion.

findConnection()

Disponibilité

Dreamweaver UltraDev 4.

Description

Dreamweaver appelle cette fonction pour détecter une connexion dans la source HTML spécifiée et analyser les paramètres de la connexion. Si le contenu de ce fichier source respecte les critères permettant une connexion valide, `findConnection()` renvoie un objet de connexion ; dans le cas contraire, cette fonction renvoie une valeur `null`.

Argument

`htmlSource`

`htmlSource` est la source HTML d'une connexion.

Valeurs renvoyées

Objet de connexion qui fournit les valeurs d'une combinaison particulière de propriétés répertoriées dans le tableau suivant. Les propriétés pour lesquelles cette fonction renvoie une valeur dépendent du type de document.

Propriété	Description
`name`	Nom de la connexion
`type`	Si `useHTTP` est `false`, indique quelle DLL doit être utilisée pour la connexion à des bases de données au moment de l'exécution.
`string`	Chaîne de connexion d'exécution. Pour ADO, il s'agit d'une chaîne de paramètres de connexion ; pour JDBC, il s'agit d'une URL de connexion.
`dsn`	Nom de la source de données utilisé pour les connexions de d'exécution ODBC ou Cold Fusion
`driver`	Nom d'un pilote JDBC utilisé pendant l'exécution
`username`	Nom d'utilisateur employé pour la connexion d'exécution
`password`	Mot de passe utilisé pour la connexion d'exécution
`designtimeString`	Chaîne de connexion au moment de la conception (voir `string`)
`designtimeDsn`	Nom de la source de données au moment de la conception (voir `dsn`)
`designtimeDriver`	Nom d'un pilote JDBC utilisé au moment de la conception
`designtimeUsername`	Nom de l'utilisateur employé pour la connexion au moment de la conception
`designtimePassword`	Mot de passe utilisé pour la connexion au moment de la conception
`designtimeType`	Type de connexion utilisée au moment de la conception

Propriété	Description
`usesDesigntimeInfo`	En cas de valeur `false`, Dreamweaver utilise les propriétés d'exécution au moment de la conception ; dans le cas contraire, Dreamweaver utilise les propriétés au moment de la conception.
`useHTTP`	Chaîne contenant `true` ou `false` : `true` indique d'utiliser la connexion HTTP au moment de la conception ; `false` indique d'utiliser DLL
`includePattern`	Expression régulière utilisée pour trouver l'instruction d'inclusion de fichier sur la page pendant Live Data et Aperçu dans le navigateur
`variables`	Objet ayant une propriété pour chaque variable de page définie sur sa valeur correspondante. Cet objet est utilisé pendant Live Data et Aperçu dans le navigateur.
`catalog`	Chaîne contenant un identificateur de base de données qui restreint la quantité de métadonnées qui apparaissent
`schema`	Chaîne contenant un identificateur de base de données qui restreint la quantité de métadonnées qui apparaissent
`filename`	Nom de la boîte de dialogue utilisée pour créer la connexion

Si une connexion n'est pas trouvée dans htmlSource, une valeur `null` est renvoyée.

REMARQUE Les développeurs peuvent ajouter des propriétés personnalisées (par exemple, métadonnées) à la source HTML, qui renvoie `applyConnection()` avec les propriétés standard.

inspectConnection()

Disponibilité

Dreamweaver UltraDev 4.

Description

Dreamweaver appelle cette fonction pour initialiser les données de la boîte de dialogue pour définir une connexion lorsque l'utilisateur modifie une connexion existante. Ce processus permet à Dreamweaver de renseigner la boîte de dialogue en utilisant les informations de connexion appropriées.

Argument

paramètres

L'argument *parameters* correspond à l'objet renvoyé par la fonction `findConnection()`.

Valeurs renvoyées

Aucune.

applyConnection()

Disponibilité

Dreamweaver UltraDev 4.

Description

Dreamweaver déclenche cette fonction lorsque l'utilisateur clique sur OK dans la boîte de dialogue de connexion. La fonction `applyConnection()` génère la source HTML pour une connexion. Dreamweaver écrit le HTML dans le fichier inclus Configuration/Connections/*connection-name.ext*, où *connection-name* est le nom de votre connexion (voir *Définissez la mise en forme de la boîte de dialogue de connexion.*, page 105) et .ext est l'extension par défaut associée au modèle de serveur.

Arguments

Aucun.

Valeurs renvoyées

Source HTML pour une connexion. Dreamweaver ferme également la boîte de dialogue de connexion. Si une erreur de validation d'un champ se produit, `applyConnection()` affiche un message d'erreur et renvoie une chaîne vide pour indiquer que la boîte de dialogue doit rester ouverte.

Fichier inclus généré

Le fichier inclus généré par `applyConnection()` déclare toutes les propriétés d'une connexion. Le nom du fichier inclus correspond au nom de connexion avec l'extension de fichier définie pour le modèle de serveur associé au site en cours.

REMARQUE

Les connexions étant partagées, définissez la valeur `allowMultiple` sur `false`. Le fichier de connexion est ainsi inclus dans le document une fois seulement et le script de serveur reste dans la page si un autre comportement de serveur l'utilise.

Les sections suivantes illustrent certains exemples de fichiers inclus générés par `applyConnection()` pour divers modèles de serveur par défaut.

REMARQUE

Pour créer un nouveau format de fichier inclus de connexion, vous devez définir un nouveau fichier de correspondance EDML, qui doit ressembler à connection_includefile.edml, comme dans *Fichier de définition pour votre type de connexion*, page 113.

ASP JavaScript

Le fichier inclus ASP et JavaScript doit être nommé MyConnection1.asp, où MyConnection1 est le nom de la connexion. L'exemple suivant est un fichier inclus pour une chaîne de connexion ADO :

```
<%
  // Filename="Connection_ado_conn_string.htm"
  // Type="ADO"
  // HTTP="true"
  // Catalog=""
  // Schema=""
  var MM_MyConnection1_STRING = "dsn=pubs";
%>
```

Le fichier du comportement de serveur inclut cette connexion en utilisant l'instruction d'inclusion de fichier relative, comme le montre l'illustration suivante :

```
<!--#include file="../Connections/MyConnection1.asp"-->
```

ColdFusion

Lorsque vous utilisez UltraDev 4 ColdFusion, Dreamweaver s'appuie sur un fichier inclus ColdFusion pour extraire une liste des sources de données.

REMARQUE

Avec Dreamweaver ColdFusion standard, Dreamweaver ignore tous les fichiers inclus et utilise les RDS pour récupérer la liste des sources de données à partir de ColdFusion.

Le fichier inclus UltraDev 4 ColdFusion doit être nommé MyConnection1.cfm, où MyConnection1 est le nom de votre connexion. L'exemple suivant illustre le fichier inclus pour une connexion ColdFusion à un tableau de produits :

```
<!-- FileName="Connection_cf_dsn.htm" "dsn=products" -->
<!-- Type="ADO" -->
<!-- Catalog="" -->
```

```
<!-- Schema="" -->
<!-- HTTP="false" -->
<CFSET MM_MyConnection1_DSN       = "products">
<CFSET MM_MyConnection1_USERNAME  = "">
<CFSET MM_Product_USERNAME        = "">
<CFSET MM_MyConnection1_PASSWORD  = "">
```

Le fichier du comportement de serveur inclut cette connexion en utilisant l'instruction `cfinclude`, comme le montre l'illustration suivante :

```
<cfinclude template="Connections/MyConnection1.cfm">
```

JSP

Le fichier inclus JSP doit être nommé MyConnection1.jsp, où MyConnection1 est le nom de votre connexion. L'exemple suivant est le fichier inclus pour une connexion JDBC à une base de données :

```
<%
  // Filename="Connection_jdbc_conn1.htm"
  // Type="JDBC"
  // HTTP="false"
  // Catalog=""
  // Schema=""
  String MM_MyConnection1_DRIVER    = "com.inet.tds.TdsDriver";
  String MM_MyConnection1_USERNAME  = "testadmin";
  String MM_MyConnection1_PASSWORD  = "velcro";
  String MM_MyConnection1_URL       = "jdbc:server:test-
  3:1433?database=pubs";
%>
```

Le fichier du comportement de serveur inclut cette connexion en utilisant l'instruction d'inclusion de fichier relative, comme le montre l'illustration suivante :

```
<%@ include file="Connections/MyConnection1.jsp" %>
```

Fichier de définition pour votre type de connexion

Pour tous les modèles de serveur, il existe un fichier connection_includefile.edml qui définit le type de connexion et associe les propriétés définies dans le fichier inclus aux éléments de l'interface Dreamweaver.

Par défaut, Dreamweaver fournit sept fichiers de définition, un pour chacun des modèles de serveur prédéfinis, comme l'illustre le tableau suivant.

Modèle de serveur	Sous-dossier du dossier Configuration/Connections
ASP JavaScript	ASP_Js
ASP.NET CSharp	ASP.NET_Csharp
ASP.NET VBScript	ASP.NET_VB
ASP VBScript	ASP_Vbs
ColdFusion	ColdFusion
Page JSP (JavaServer Page)	JSP
PHP MySql	PHP_MySql

Dreamweaver utilise les paramètres `quickSearch` et `searchPattern` pour reconnaître les blocs de connexion et le paramètre `insertText` afin de créer des blocs de connexion. Pour plus d'informations sur les balises et les attributs EDML et sur les modèles de recherche d'expression régulière, voir Comportements de serveur dans *Extension de Dreamweaver*.

REMARQUE Si vous changez le format de votre fichier inclus ou si vous définissez un fichier inclus pour un nouveau modèle de serveur, vous devez associer les paramètres de connexion avec l'interface utilisateur de Dreamweaver, Live Data et Aperçu dans le navigateur. L'exemple suivant de fichier EDML, associé au modèle de serveur ASP JS par défaut, met en correspondance toutes les variables de page de connexion avec leurs valeurs dynamiques respectives avant d'envoyer la page au serveur. Pour plus d'informations sur EDML et les modèles de recherche d'expression régulière, voir Comportements de serveur dans ***Extension de Dreamweaver***.

```
<participant name="connection_includefile" version="5.0">
  <quickSearch>
    <![CDATA[// HTTP=]]></quickSearch>
  <insertText location="">
<![CDATA[<%
// FileName="@@filename@@"
// Type="@@type@@" @@designtimeString@@
// DesigntimeType="@@designtimeType@@"
// HTTP="@@http@@"
// Catalog="@@catalog@@"
// Schema="@@schema@@"
var MM_@@cname@@_STRING = @@string@@
%>
]]>
  </insertText>
  <searchPatterns whereToSearch="directive">
    <searchPattern paramNames="filename">
```

```
      <![CDATA[/\/\/\s*FileName="([^"]*)"/]]></searchPattern>
    <searchPattern paramNames="type,designtimeString">
      <![CDATA[/\/\/\s+Type="(\w*)"([^\r\n]*)/]]></searchPattern>
    <searchPattern paramNames="designtimeType" isOptional="true">
      <![CDATA[/\/\/\s*DesigntimeType="(\w*)"/]]></searchPattern>
    <searchPattern paramNames="http">
      <![CDATA[/\/\/\s*HTTP="(\w+)"/]]></searchPattern>
    <searchPattern paramNames="catalog">
      <![CDATA[/\/\/\s*Catalog="(\w*)"/]]></searchPattern>
    <searchPattern paramNames="schema">
      <![CDATA[/\/\/\s*Schema="(\w*)"/]]></searchPattern>
    <searchPattern paramNames="cname,string">
      <![CDATA[/var\s+MM_(\w*)_STRING\s*=\s*([^\r\n]+)/]]></searchPattern>
  </searchPatterns>
</participant>
```

Les expressions d'un fichier EDML, telles que `@@filename@@` dans cet exemple, associent les valeurs du fichier inclus avec les propriétés d'un objet de connexion. Les propriétés des objets de connexion sont définies dans le fichier de mise en œuvre JavaScript.

Toutes les boîtes de dialogue de connexion par défaut de Dreamweaver utilisent le fichier de correspondance connection_includefile.edml. Pour permettre à Dreamweaver de trouver ce fichier, son nom est défini dans le fichier de mise en œuvre JavaScript, comme le montre l'illustration suivante :

```
var PARTICIPANT_FILE = "connection_includefile";
```

Lors de la création d'un type de connexion personnalisée, vous pouvez utiliser n'importe quel fichier de correspondance dans vos boîtes de dialogue personnalisées. Si vous créez un fichier de correspondance, vous pouvez utiliser un nom différent de connection_includefile pour votre fichier EDML. Si vous utilisez un autre nom, vous devez l'utiliser dans votre fichier de mise en œuvre JavaScript lorsque vous indiquez la valeur assignée à la variable `PARTICIPANT_FILE`, comme le montre l'illustration suivante :

```
var PARTICIPANT_FILE = "myConnection_mappingfile";
```

CHAPITRE 8

API JavaBeans

8

Ce chapitre présente les API des composants JavaBeans. Les fonctions `MMJB*()` sont des accroches JavaScript qui appellent des introspections Java pour la prise en charge des JavaBeans. Ces fonctions extraient des noms de classe, des méthodes, des propriétés et des événements à partir des JavaBeans, qui peuvent être affichés dans l'interface utilisateur Dreamweaver. Pour utiliser ces fonctions JavaScript et permettre à Dreamweaver 8 d'accéder à des JavaBeans, ces derniers doivent se trouver dans le dossier Configuration/Classes.

REMARQUE

Les arguments de fonction décrits dans ce chapitre contiennent parfois un argument appelé `packageName.className`, qui représente une valeur unique.

L'API JavaBeans

Les fonctions suivantes sont des méthodes de l'objet `MMJB`.

MMJB.getClasses()

Disponibilité

Dreamweaver UltraDev 4.

Description

Lit tous les noms de classe des JavaBeans dans le dossier Configuration/Classes.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes des noms de classe résidant dans le dossier Configuration/Classes ; une erreur renvoie un tableau vide.

MMJB.getClassesFromPackage()

Disponibilité

Dreamweaver UltraDev 4.

Description

Lit toutes les classes JavaBeans du paquet.

Arguments

packageName.pathName

- L'argument *packageName.pathName* correspond au chemin du paquet. Il doit s'agir d'une archive Java JAR ou ZIP (par exemple, `C:/jdbcdrivers/Una2000_Enterprise.zip`).

Valeurs renvoyées

Tableau de chaînes des noms de classe à l'intérieur du fichier JARou ZIP ; une erreur renvoie un tableau vide.

MMJB.getErrorMessage()

Disponibilité

Dreamweaver UltraDev 4.

Description

Extrait le dernier message d'erreur de Dreamweaver envoyé pendant l'utilisation de l'interface MMJB.

Arguments

Aucun.

Valeurs renvoyées

Une chaîne du message Dreamweaver de la dernière erreur.

MMJB.getEvents()

Disponibilité

Dreamweaver UltraDev 4, améliorée dans la version Dreamweaver MX.

Description

Introspecte la classe des JavaBeans et en renvoie les événements.

Arguments

packageName.className, {packagePath}

- L'argument *packageName.className* est le nom de la classe. La classe doit être située dans une archive Java JAR ou ZIP. Si *packagePath* est omis, l'archive doit être située dans le `classpath` de votre système, ou doit être un fichier de classe installé dans le dossier Configuration/Classes.
- L'argument *PackagePath* est une chaîne facultative qui pointe vers l'emplacement de l'archive Java JAR ou ZIP qui contient *className*.

Valeurs renvoyées

Un tableau de chaînes des évènements associés à *className* ; une erreur renvoie un tableau vide.

MMJB.getIndexedProperties()

Disponibilité

Dreamweaver UltraDev 4, améliorée dans la version Dreamweaver MX.

Description

Introspecte la classe des JavaBeans et en renvoie les propriétés indexées, qui correspondent à des modèles de conception dont le comportement est identique à celui des ensembles.

Arguments

packageName.className, {packagePath}

- L'argument *packageName.className* est le nom de la classe. La classe doit être située dans une archive Java JAR ou ZIP. Si *packagePath* est omis, l'archive doit être située dans le `classpath` de votre système, ou doit être un fichier de classe installé dans le dossier Configuration/Classes.
- L'argument *packagePath*, argument facultatif, est une chaîne qui pointe vers l'archive Java JAR ou ZIP contenant *className*.

Valeurs renvoyées

Tableau de chaînes relatives aux propriétés indexées de *className* ; une erreur renvoie un tableau vide.

MMJB.getMethods()

Disponibilité

Dreamweaver UltraDev 4, améliorée dans la version Dreamweaver MX.

Description

Introspecte la classe des JavaBeans et en renvoie les méthodes.

Arguments

packageName.className, {*packagePath*}

- L'argument *packageName.className* est le nom de la classe. La classe doit être située dans une archive Java JAR ou ZIP. Si *packagePath* est omis, l'archive doit être située dans le `classpath` de votre système, ou doit être un fichier de classe installé dans le dossier Configuration/Classes.
- L'argument *PackagePath* est une chaîne facultative qui pointe vers l'emplacement de l'archive Java JAR ou ZIP qui contient *className*.

Valeurs renvoyées

Tableau de chaînes relatives aux méthodes associées à *className* ; une erreur renvoie un tableau vide.

MMJB.getProperties()

Disponibilité

Dreamweaver UltraDev 4, améliorée dans la version Dreamweaver MX.

Description

Introspecte la classe des JavaBeans et en renvoie les propriétés.

Arguments

packageName.className, {*packagePath*}

- L'argument *packageName.className* est le nom de la classe. La classe doit être située dans une archive Java JAR ou ZIP. Si *packagePath* est omis, l'archive doit être située dans le `classpath` de votre système, ou doit être un fichier de classe installé dans le dossier Configuration/Classes.
- L'argument *PackagePath* est une chaîne facultative qui pointe vers l'emplacement de l'archive Java JAR ou ZIP qui contient *className*.

Valeurs renvoyées

Tableau de chaînes relatives aux propriétés associées à *`className`* ; une erreur renvoie un tableau vide.

MMJB.getReadProperties()

Disponibilité

Dreamweaver MX.

Description

Extrait les propriétés en lecture seule des JavaBeans prenant en charge les appels d'accès définis.

Arguments

`packageName.className`, {*`packagePath`*}

- L'argument *`packageName.className`* est le nom de la classe. La classe doit être située dans une archive Java JAR ou ZIP. Si *`packagePath`* est omis, l'archive doit être située dans le `classpath` de votre système, ou doit être un fichier de classe installé dans le dossier Configuration/Classes.
- L'argument *`packagePath`*, argument facultatif, est une chaîne qui pointe vers l'archive Java JAR ou ZIP contenant *`className`*.

Valeurs renvoyées

Tableau de chaînes de propriétés en lecture seule associées à *`className`* ; une erreur renvoie un tableau vide.

MMJB.getWriteProperties()

Disponibilité

Dreamweaver MX.

Description

Propriétés en écriture seule pour les JavaBeans prenant en charge les appels de méthodes définies.

Arguments

`packageName.className`, {*`packagePath`*}

- L'argument *packageName.className* est le nom de la classe. La classe doit être située dans une archive Java JAR ou ZIP. Si *packagePath* est omis, l'archive doit être située dans le `classpath` de votre système, ou doit être un fichier de classe installé dans le dossier Configuration/Classes.
- L'argument *packagePath*, argument facultatif, est une chaîne qui pointe vers l'archive Java JAR ou ZIP contenant *className*.

Valeurs renvoyées

Tableau de chaînes relatives aux propriétés en écriture seule associées à *className* ; une erreur renvoie un tableau vide.

CHAPITRE 9

API d'intégration de commande source

9

L'API d'intégration de commande source vous permet de rédiger des bibliothèques partagées afin d'accroître les fonctionnalités d'archivage et d'extraction de Macromedia Dreamweaver 8 à l'aide de systèmes de commande source (tels que Sourcesafe ou CVS).

Vos bibliothèques doivent prendre en charge un minimum de fonctions API pour que Dreamweaver puisse être intégré au système de commande source. En outre, vos bibliothèques doivent se trouver dans le dossier Files/Common Files/Macromedia/2004/Source Control.

Lorsque vous démarrez Dreamweaver, celui-ci charge toutes les bibliothèques. Il détermine les fonctionnalités prises en charge par chaque bibliothèque en appelant la fonction `GetProcAddress()` pour chaque API. Si une adresse n'existe pas, Dreamweaver suppose que la bibliothèque ne prend pas en charge l'API. Si l'adresse existe, Dreamweaver utilise la version de la fonction qui se trouve dans la bibliothèque pour prendre en charge la fonctionnalité. Lorsqu'un utilisateur Dreamweaver définit ou modifie un site, puis choisit l'onglet SCS du serveur Web, les choix correspondant aux DLL chargées depuis le dossier Program Files/Common Files/Macromedia/2004/Source Control s'affichent (en plus des éléments standard) dans l'onglet.

Pour créer un menu Site > Commande source auquel vous pouvez ajouter des éléments personnalisés, ajoutez le code suivant au fichier :

```
<menu name="Source Control" id="DWMenu_MainSite_Site_Source¬
Control"><menuitem dynamic name="None"file="Menus/MM/¬
File_SCSItems.htm" id="DWMenu_MainSite_Site_NewFeatures_¬
Default" />
</menu>
```

Fonctionnement de l'intégration des commandes source avec Dreamweaver

Lorsqu'un utilisateur Dreamweaver choisit des fonctions de connexion au serveur, de transfert de fichiers ou de Design Notes, Dreamweaver appelle la version de la DLL de la fonction API correspondante (`Connect()`, `Disconnect()`, `Get()`, `Put()`, `Checkin()`, `Checkout()`, `Undocheckout()` et `Synchronize()`). La DLL est responsable de la gestion de la requête, notamment de l'affichage des boîtes de dialogue qui rassemblent les informations ou qui permettent à l'utilisateur d'interagir avec la DLL. La DLL affiche également des informations ou des messages d'erreur.

Le système de commande source peut éventuellement prendre en charge les Design Notes et l'archivage et l'extraction. Pour activer les Design Notes dans les systèmes de commande source, l'utilisateur Dreamweaver doit choisir l'onglet Design Notes dans la boîte de dialogue Modifier les sites et cocher la case qui permet d'activer cette fonctionnalité (cette procédure s'applique également aux systèmes FTP et de réseau local). Si le système de commande source ne prend pas en charge les Design Notes et que l'utilisateur souhaite les utiliser, Dreamweaver transporte les fichiers Design Note (.mno) pour gérer les Design Notes (de la même façon qu'avec les systèmes FTP et de réseau local).

Les fonctions d'archivage et d'extraction sont traitées différemment ; si le système de commande source les prend en charge, l'utilisateur ne peut pas éviter leur utilisation dans la boîte de dialogue Design Notes. Si l'utilisateur essaie de court-circuiter le système de commande source, un message d'erreur s'affiche.

Ajout d'une fonctionnalité de système de commande source

Pour ajouter une fonctionnalité de système de commande source à Dreamweaver, rédigez un gestionnaire `GetNewFeatures` qui renvoie un jeu d'éléments de menu et les fonctions C correspondantes. Si, par exemple, vous rédigez une bibliothèque Sourcesafe et que vous souhaitez permettre aux utilisateurs de Dreamweaver de consulter l'historique d'un fichier, vous pouvez rédiger un gestionnaire `GetNewFeatures` qui renvoie l'élément de menu Historique et le nom de la fonction C `history`. Ainsi, sous Windows, si un utilisateur clique avec le bouton droit de la souris sur un fichier, l'élément Historique s'affiche dans le menu. Si l'utilisateur choisit alors l'élément de menu Historique, Dreamweaver appelle la fonction correspondante, qui se charge de transmettre les fichiers sélectionnés à la DLL. La DLL affiche ensuite la boîte de dialogue Historique, ce qui permet à l'utilisateur d'interagir avec elle de la même façon que Sourcesafe.

Fonctions obligatoires de l'API d'intégration de commande source

L'API d'intégration de commande source comporte des fonctions obligatoires et facultatives. Les fonctions répertoriées dans cette section sont obligatoires.

bool SCS_GetAgentInfo()

Description

Demande à la DLL de renvoyer son nom et sa description, qui sont affichés dans la boîte de dialogue Modifier les sites. Le nom apparaît dans le menu déroulant Accès (par exemple, sourcesafe, webdav, perforce) et la description s'affiche juste en dessous du menu.

Arguments

```
char name[32], char version[32], char description[256], const char
  *dwAppVersion
```

- L'argument *name* est le nom du système de commande source. Ce nom s'affiche dans la zone de liste modifiable permettant de sélectionner un système de commande source, dans l'onglet Commande source de la boîte de dialogue Modifier les sites. Le nom ne doit pas compter plus de 32 caractères.

- *version* est une chaîne qui indique la version de la DLL. La version apparaît dans l'onglet Commande source de la boîte de dialogue Modifier les sites. La version ne doit pas compter plus de 32 caractères.
- *description* est une chaîne qui décrit le système de commande source. La description apparaît dans l'onglet Commande source de la boîte de dialogue Modifier les sites. La description ne doit pas compter plus de 256 caractères.
- *dwAppVersion* est une chaîne qui décrit la version de Dreamweaver appelant la DLL. La DLL peut utiliser cette chaîne pour déterminer la version et la langue de Dreamweaver.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_Connect()

Description

Connecte l'utilisateur à son système de commande source. Si la DLL ne dispose pas d'informations de connexion, elle doit afficher une boîte de dialogue invitant l'utilisateur à entrer des informations, et elle doit stocker les données pour une utilisation ultérieure.

Arguments

```
void **connectionData, const char siteName[64]
```

- L'argument *connectionData* est un descripteur des données que l'agent souhaite recevoir de Dreamweaver lorsqu'il appelle d'autres fonctions API.
- *siteName* est une chaîne qui pointe vers le nom du site. Le nom du site ne doit pas compter plus de 64 caractères.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_Disconnect()

Description

Déconnecte l'utilisateur du système de commande source.

Arguments

```
void *connectionData
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_IsConnected()

Description

Détermine l'état de la connexion.

Arguments

`void *connectionData`

- *`connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

int SCS_GetRootFolderLength()

Description

Renvoie la longueur du nom du dossier racine.

Arguments

`void *connectionData`

- *`connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

Valeurs renvoyées

Nombre entier qui indique la longueur du nom du dossier racine. Si la fonction renvoie `< 0`, Dreamweaver considère cette réponse comme une erreur et tente de récupérer le message d'erreur de la DLL si elle est prise en charge.

bool SCS_GetRootFolder()

Description

Renvoie le nom du dossier racine.

Arguments

`void *connectionData, char remotePath[], const int folderLen`

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est une mémoire tampon dans laquelle est enregistré le chemin distant complet du dossier racine.
- *folderLen* est un nombre entier qui indique la longueur de l'argument *remotePath*. Il s'agit de la valeur renvoyée par la fonction *GetRootFolderLength*.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

int SCS_GetFolderListLength()

Description

Renvoie le nombre d'éléments dans le dossier transmis.

Arguments

```
void *connectionData, const char *remotePath
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est le chemin d'accès et le nom complets du dossier distant dont la DLL vérifie le nombre d'éléments.

Valeurs renvoyées

Nombre entier qui indique le nombre d'éléments dans le dossier en cours. Si la fonction renvoie `< 0`, Dreamweaver considère cette réponse comme une erreur et tente de récupérer le message d'erreur de la DLL si elle est prise en charge.

bool SCS_GetFolderList()

Description

Renvoie une liste de fichiers et de dossiers dans le dossier transmis, notamment des informations pertinentes telles que la date de modification, la taille et si l'élément est un dossier ou un fichier.

Arguments

```
void *connectionData, const char *remotePath, itemInfo itemList[ ], const
  int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est le nom du chemin d'accès au dossier distant dont la DLL vérifie le nombre d'éléments.
- *itemList* est une liste pré-allouée de structures `itemInfo` :

name	char[256]	nom de fichier ou de dossier
isFolder	bool	`true` si c'est un dossier, `false` si c'est un fichier
month	int	Composant mois de la date de modification, de 1 à 12
day	int	Composant jour de la date de modification, de 1 à 31
year	int	Composant année de la date de modification, 1900+
hour	int	Composant heure de la date de modification, de 0 à 23
minutes	int	Composant minute de la date de modification, de 0 à 59
seconds	int	Composant seconde de la date de modification, de 0 à 59
type	char[256]	type de fichier (s'il n'est pas défini par la DLL, DW utilise l'extension de fichier pour déterminer le type, comme il le fait à présent)
size	int	En octets

- *numItems* est le nombre d'éléments alloués à l'argument `itemList` (renvoyé par la fonction `GetFolderListLength`).

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_Get()

Description

Extrait une liste de fichiers ou de dossiers et les stocke localement.

Arguments

```
void *connectionData, const char *remotePathList[], const char
  *localPathList[], const int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

- *remotePathList* est une liste des fichiers ou dossiers distants à extraire, exprimée sous la forme de noms et de chemins d'accès complets.
- L'argument *localPathList* est une liste miroir des noms de chemin d'accès aux fichiers ou aux dossiers locaux.
- L'argument *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_Put()

Description

Place une liste de fichiers ou de dossiers locaux dans le système de commande source.

Arguments

```
void *connectionData, const char *localPathList[], const char
  *remotePathList[], const int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- L'argument *localPathList* est la liste des noms de fichiers locaux ou des chemins de fichiers à placer dans le système de commande source.
- L'argument *remotePathList* est une liste miroir de noms de chemin d'accès aux fichiers ou aux dossiers distants.
- L'argument *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_NewFolder()

Description

Crée un nouveau dossier.

Arguments

```
void *connectionData, const char *remotePath
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est le chemin d'accès complet du dossier distant que la DLL crée.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_Delete()

Description

Supprime une liste de fichiers ou de dossiers du système de commande source.

Arguments

`void *connectionData, const char *remotePathList[], const int numItems`

- *`connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- L'argument *`remotePathList`* est une liste de noms de chemin d'accès aux fichiers ou aux dossiers distants à supprimer.
- *`numItems`* est le nombre d'éléments dans *`remotePathList`*.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_Rename()

Description

Renomme ou déplace un fichier ou un dossier selon les valeurs spécifiées dans les arguments *`oldRemotePath`* et *`newRemotePath`*. Ainsi, si les valeurs de *`oldRemotePath`* et `newRemotePath` sont respectivement *`"$/folder1/file1"`* et `"$/folder1/renamefile1"`, le fichier file1 se voit attribuer le nouveau nom renamefile1 et reste dans le dossier folder1.

Si les valeurs de *`oldRemotePath`* et de `newRemotePath` sont respectivement *`"$/folder1/file1"`* et `"$/folder1/subfolder1/file1"`, le fichier file1 est alors déplacé dans le sous-dossier subfolder1.

Pour savoir si l'invocation de cette fonction constitue un déplacement ou l'attribution d'un nouveau nom, vérifiez les chemins parents des deux valeurs d'entrée ; s'ils sont identiques, il s'agit de l'opération « renommer ».

Arguments

`void *connectionData`, *`const char *oldRemotePath`*, *`const char *newRemotePath`*

- L'argument *`connectionData`* est un pointeur vers les données de l'agent transférées à Dreamweaver pendant l'appel de la fonction `Connect()`.

- *oldRemotePath* est le nom du chemin d'accès au fichier ou dossier distant à renommer ou à déplacer.
- *newRemotePath* est le nom du chemin d'accès au nouveau fichier ou dossier distant.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_ItemExists()

Description

Détermine si un fichier ou un dossier existe sur le serveur.

Arguments

*void *connectionData, const char *remotePath*

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est le chemin d'accès d'un fichier ou d'un dossier distant.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

Fonctions facultatives de l'API d'intégration de commande source

L'API d'intégration de commande source comporte des fonctions obligatoires et facultatives. Les fonctions répertoriées dans cette section sont facultatives.

bool SCS_GetConnectionInfo()

Description

Affiche une boîte de dialogue qui permet à l'utilisateur de modifier ou de définir les informations de connexion du site. N'établit pas la connexion. Cette fonction est appelée lorsque l'utilisateur clique sur le bouton Paramètres dans la section Infos distantes de la boîte de dialogue Modifier les sites.

Arguments

*void **connectionData, const char siteName[64]*

- L'argument *connectionData* est un descripteur des données que l'agent veut recevoir de Dreamweaver lorsqu'il appelle d'autres fonctions API.
- L'argument *siteName* est une chaîne qui pointe vers le nom du site. Ce nom ne peut pas comporter plus de 64 caractères.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_SiteDeleted()

Description

Informe la DLL que le site a été supprimé ou qu'il n'est plus lié à ce système de commande source. Cette fonction indique au système de commande source qu'il peut supprimer les informations persistantes du site.

Arguments

const char siteName[64]

- L'argument *siteName* est une chaîne qui pointe vers le nom du site. Ce nom ne peut pas comporter plus de 64 caractères.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_SiteRenamed()

Description

Notifie à la DLL que l'utilisateur a renommé le site, pour qu'il puisse mettre à jour les informations persistantes relatives à ce site.

Arguments

const char oldSiteName[64], const char newSiteName[64]

- L'argument *oldSiteName* est une chaîne qui pointe vers le nom initial du site, avant qu'il ne soit renommé. Ce nom ne peut pas comporter plus de 64 caractères.
- *newSiteName* est une chaîne qui pointe vers le nouveau nom du site, après qu'il a été renommé. Ce nom ne peut pas comporter plus de 64 caractères.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

int SCS_GetNumNewFeatures()

Description

Renvoie le nombre de nouvelles fonctions à ajouter à Dreamweaver (comme Historique de fichier, Différences, etc.).

Arguments

Aucun.

Valeurs renvoyées

Nombre entier qui indique le nombre de nouvelles fonctionnalités à ajouter à Dreamweaver. Si la fonction renvoie `< 0`, Dreamweaver considère cette réponse comme une erreur et tente de récupérer le message d'erreur de la DLL si elle est prise en charge.

bool SCS_GetNewFeatures()

Description

Renvoie une liste d'éléments à ajouter aux menus principaux et contextuels de Dreamweaver. Par exemple, la DLL Sourcesafe peut ajouter Historique et Différences de fichiers au menu principal.

Arguments

```
char menuItemList[][32], scFunction functionList[], scFunction
  enablerList[], const int numNewFeatures
```

- L'argument *`menuItemList`* est une liste de chaînes complétée par la DLL et indiquant les éléments à ajouter aux menus principaux et contextuels. Chaque chaîne peut contenir 32 caractères maximum.
- *`functionList`* est renseigné par la DLL ; il indique les routines de la DLL à appeler lorsque l'utilisateur choisit l'élément de menu correspondant.
- *`enablerList`* est renseigné par la DLL ; il indique les routines de la DLL à appeler lorsque Dreamweaver a besoin de déterminer si l'élément de menu correspondant est activé.
- *`numNewFeatures`* est le nombre d'éléments ajoutés par la DLL ; cette valeur est récupérée en appelant la fonction `GetNumNewFeatures()`.

La signature de fonction suivante définit les fonctions et les activateurs transmis par appel de la fonction `SCS_GetNewFeatures()` dans les arguments `functionlist` et `enablerList`.

```
bool (*scFunction)(void *connectionData, const char *remotePathList[],
  const char *localPathList[], const int numItems)
```

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_GetCheckoutName()

Description

Renvoie le nom d'extraction de l'utilisateur en cours. Si le système de commande source ne prend pas en charge cette fonction et que l'utilisateur a activé l'extraction, la fonction utilise la fonctionnalité interne d'archivage et d'extraction de Dreamweaver qui transporte les fichiers LCK depuis et vers le système de commande source.

Arguments

```
void *connectionData, char checkOutName[64], char emailAddress[64]
```

- *`connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *`checkOutName`* est le nom de l'utilisateur en cours.
- *`emailAddress`* est l'adresse électronique de l'utilisateur en cours.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_Checkin()

Description

Archive une liste de fichiers ou de dossiers locaux dans le système de commande source. La DLL doit configurer le fichier en lecture seule. Si le système de commande source ne prend pas en charge cette fonction et que l'utilisateur a activé l'extraction, la fonction utilise la fonctionnalité interne d'archivage et d'extraction de Dreamweaver qui transporte les fichiers LCK depuis et vers le système de commande source.

Arguments

```
void *connectionData, const char *localPathList[], const char
  *remotePathList[], bool successList[], const int numItems
```

- *`connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- L'argument *`localPathList`* est une liste de noms de chemin d'accès aux fichiers ou aux dossiers locaux à archiver.

- L'argument *remotePathList* est une liste miroir de noms de chemin d'accès aux fichiers ou aux dossiers distants.
- *successList* est une liste de valeurs booléennes complétée par la DLL pour permettre à Dreamweaver de connaître les fichiers dont l'archivage a réussi.
- *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_Checkout()

Description

Extrait une liste de fichiers ou de dossiers locaux du système de commande source. La DLL se charge d'accorder les droits d'accès en écriture au fichier. Si le système de commande source ne prend pas en charge cette fonction et que l'utilisateur a activé l'extraction, la fonction utilise la fonctionnalité interne d'archivage et d'extraction de Dreamweaver qui transporte les fichiers LCK depuis et vers le système de commande source.

Arguments

```
void *connectionData, const char *remotePathList[], const char
  *localPathList[], bool successList[], const int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePathList* est une liste de noms de chemin d'accès aux fichiers et aux dossiers distants à extraire.
- L'argument *localPathList* est une liste miroir des noms de chemin d'accès aux fichiers ou aux dossiers locaux.
- *successList* est une liste de valeurs booléennes complétée par la DLL pour permettre à Dreamweaver de connaître les fichiers dont l'extraction a réussi.
- L'argument *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_UndoCheckout()

Description

Annule l'état d'extraction d'une liste de fichiers ou de dossiers. La DLL doit configurer le fichier en lecture seule. Si le système de commande source ne prend pas en charge cette fonction et que l'utilisateur a activé l'extraction, la fonction utilise la fonctionnalité interne d'archivage et d'extraction de Dreamweaver qui transporte les fichiers LCK depuis et vers le système de commande source.

Arguments

```
void *connectionData, const char *remotePathList[], const char
  *localPathList[], bool successList[], const int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePathList* est une liste de chemins aux fichiers ou dossiers distants dont l'extraction doit être annulée.
- L'argument *localPathList* est une liste miroir des noms de chemin d'accès aux fichiers ou aux dossiers locaux.
- *successList* est une liste de valeurs booléennes complétée par la DLL pour permettre à Dreamweaver de connaître les fichiers dont l'extraction a été annulée avec succès.
- L'argument *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

int SCS_GetNumCheckedOut()

Description

Renvoie le nombre d'utilisateurs dont un fichier a été extrait.

Arguments

```
void *connectionData, const char *remotePath
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est le nom du chemin d'accès au fichier ou dossier distant à vérifier pour connaître le nombre d'utilisateurs ayant procédé à son extraction.

Valeurs renvoyées

Nombre entier qui représente le nombre de personnes disposant du fichier extrait. Si la fonction renvoie `< 0`, Dreamweaver considère cette réponse comme une erreur et tente de récupérer le message d'erreur de la DLL si elle est prise en charge.

bool SCS_GetFileCheckoutList()

Description

Renvoie une liste d'utilisateurs dont un fichier a été extrait. Si cette liste est vide, c'est que personne n'a de fichier extrait.

Arguments

```
void *connectionData, const char *remotePath, char checkOutList[][64], char
  emailAddressList[][64], const int numCheckedOut
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est le nom du chemin d'accès au fichier ou dossier distant à vérifier pour connaître le nombre d'utilisateurs ayant procédé à son extraction.
- *checkOutList* est une liste de chaînes qui correspond aux utilisateurs disposant du fichier extrait. Chaque chaîne ne doit pas avoir plus de 64 caractères.
- *emailAddressList* est une liste de chaînes correspondant aux adresses électroniques des utilisateurs. Chaque adresse ne doit pas dépasser 64 caractères.
- *numCheckedOut* est le nombre de personnes qui ont le fichier extrait. Il est renvoyé par `GetNumCheckedOut()`.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

int SCS_GetErrorMessageLength()

Description

Renvoie la longueur du message d'erreur interne en cours de la DLL. Permet d'allouer la mémoire tampon transmise dans la fonction `GetErrorMessage()`. Cette fonction doit être appelée uniquement si une fonction d'API renvoie la valeur `false` ou `<0`, ce qui indique un échec.

Arguments

```
void *connectionData
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

Valeurs renvoyées

Nombre entier représentant la longueur du message d'erreur.

bool SCS_GetErrorMessage()

Description

Renvoie le dernier message d'erreur. Si vous implémentez `getErrorMessage()`, Dreamweaver l'appelle à chaque fois qu'une de vos fonctions d'API renvoie `false`.

Si une routine renvoie `-1` ou `false`, cela indique qu'un message d'erreur doit être disponible.

Arguments

*void *connectionData, char errorMsg[], const int *msgLength*

connectionData est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

- *errorMsg* est une chaîne pré-allouée de la DLL dans laquelle vient se placer le message d'erreur.
- *msgLength* est la longueur de la mémoire tampon représentée par l'argument *errorMsg[]*.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

int SCS_GetNoteCount()

Description

Renvoie le nombre de clés Design Note pour le chemin de dossier ou de fichier distant spécifié. Si cela n'est pas pris en charge par le système de commande source, Dreamweaver obtient ces informations du fichier compagnon MNO.

Arguments

*void *connectionData, const char *remotePath*

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est le nom du chemin d'accès au fichier ou dossier distant dont la DLL vérifie le nombre de Design Notes jointes.

Valeurs renvoyées

Nombre entier qui indique le nombre de Design Notes associées au fichier. Si la fonction renvoie `< 0`, Dreamweaver considère cette réponse comme une erreur et tente de récupérer le message d'erreur de la DLL si elle est prise en charge.

int SCS_GetMaxNoteLength()

Description

Renvoie la longueur de la Design Note la plus longue pour le fichier ou le dossier spécifié. Si cette fonction n'est pas prise en charge par le système de commande source, Dreamweaver obtient ces informations du fichier MNO.

Arguments

```
void *connectionData, const char *remotePath
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est le nom du chemin d'accès au fichier ou dossier distant dont la DLL vérifie la Design Note la plus longue.

Valeurs renvoyées

Nombre entier qui indique la taille de la Design Note la plus longue associée au fichier. Si la fonction renvoie `< 0`, Dreamweaver considère cette réponse comme une erreur et tente de récupérer le message d'erreur de la DLL si elle est prise en charge.

bool SCS_GetDesignNotes()

Description

Récupère des paires clé-valeur des méta-informations pour le fichier ou le dossier spécifié. Si cette fonction n'est pas prise en charge par le système de commande source, Dreamweaver récupère ces informations dans le fichier MNO correspondant.

Arguments

```
void *connectionData, const char *remotePath, char keyList[][64],
  char *valueList[], bool showColumnList[], const int noteCount,
  const int noteLength
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

- *remotePath* est le nom du chemin d'accès au fichier ou dossier distant dont la DLL vérifie le nombre d'éléments.
- L'argument *keyList* est une liste de clés de Design Note, comme `"Status"`.
- *valueList* est une liste de valeurs de Design Note correspondant aux clés de Design Note, comme `"Awaiting Signoff"`.
- L'argument *showColumnList* est une liste de valeurs booléennes correspondant aux clés de Design Note, qui indiquent si Dreamweaver peut afficher une clé sous forme de colonne dans le panneau Site.
- *noteCount* correspond au nombre de Design Notes jointes à un fichier ou dossier ; cette valeur est renvoyée par la fonction `GetNoteCount()`.
- *noteLength* est la longueur maximale d'une Design Note ; cette valeur est renvoyée par la fonction `GetMaxNoteLength()`.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_SetDesignNotes()

Description

Enregistre les paires clé-valeur dans les méta-informations du fichier ou du dossier spécifié. Cela remplace le jeu de méta-informations du fichier. Si cette fonction n'est pas prise en charge par le système de commande source, Dreamweaver enregistre les Design Notes dans des fichiers MNO.

Arguments

```
void *connectionData, const char *remotePath, const char keyList[][64],
  const char *valueList[], bool showColumnList[], const int noteCount,
  const int noteLength
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePath* est le nom du chemin d'accès au fichier ou dossier distant dont la DLL vérifie le nombre d'éléments.
- L'argument *keyList* est une liste de clés de Design Note, comme `"Status"`.
- *valueList* est une liste de valeurs de Design Note correspondant aux clés de Design Note, comme `"Awaiting Signoff"`.
- L'argument *showColumnList* est une liste de valeurs booléennes correspondant aux clés de Design Note, qui indiquent si Dreamweaver peut afficher une clé sous forme de colonne dans le panneau Site.

- *noteCount* correspond au nombre de Design Notes jointes à un fichier ou dossier ; ce nombre permet à la DLL de connaître la taille des listes spécifiées. Si *noteCount* a pour valeur 0, toutes les Design Notes sont supprimées du fichier.
- *noteLength* est la longueur de la Design Note la plus longue pour le fichier ou le dossier spécifié.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_IsRemoteNewer()

Description

Vérifie chaque chemin distant spécifié pour voir si la copie distante est plus récente.
Si cette fonction n'est pas prise en charge par le système de commande source, Dreamweaver utilise son algorithme interne `isRemoteNewer`.

Arguments

```
void *connectionData, const char *remotePathList[],
  const char *localPathList[], int remoteIsNewerList[], const int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePathList* est une liste de noms de chemin d'accès aux fichiers ou aux dossiers à comparer pour connaître ceux dotés des états les plus récents.
- L'argument *localPathList* est une liste miroir des noms de chemin d'accès aux fichiers ou aux dossiers locaux.
- *remoteIsNewerList* est une liste de nombres entiers complétée par la DLL pour permettre à Dreamweaver d'identifier le fichier le plus récent du côté distant. Les valeurs suivantes sont valides : 1 indique que la version distante est la plus récente ; -1 indique que la version locale est la plus récente ; 0 indique que les deux versions sont identiques.
- L'argument *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

Activateurs

Si les activateurs facultatifs ne sont pas pris en charge par le système de commande source ou que l'application n'est pas connectée au serveur, Dreamweaver détermine le moment où les éléments de menu sont activés, en fonction des informations dont il dispose concernant les fichiers distants.

bool SCS_canConnect()

Description

Indique si l'élément de menu Connecter doit être activé.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_canGet()

Description

Indique si l'élément de menu Acquérir doit être activé.

Arguments

```
void *connectionData, const char *remotePathList[], const char
  *localPathList[], const int numItems
```

- *`connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *`remotePathList`* est une liste de noms de chemin d'accès aux fichiers et aux dossier distant à obtenir.
- L'argument *`localPathList`* est une liste miroir des noms de chemin d'accès aux fichiers ou aux dossiers locaux.
- L'argument *`numItems`* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_canCheckout()

Description

Indique si l'élément de menu Extraire doit être activé.

Arguments

```
void *connectionData, const char *remotePathList[], const char
  *localPathList[], const int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- L'argument *remotePathList* est une liste de noms de chemin d'accès aux fichiers et aux dossiers distant à extraire.
- L'argument *localPathList* est une liste miroir des noms de chemin d'accès aux fichiers ou aux dossiers locaux.
- L'argument *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_canPut()

Description

Indique si l'élément de menu Placer doit être activé.

Arguments

```
void *connectionData, const char *localPathList[], const char
  *remotePathList[], const int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *localPathList* est une liste de noms de chemin d'accès aux fichiers ou aux dossiers à placer dans le système de commande source.
- *remotePathList* est une liste miroir de noms de chemin d'accès aux fichiers ou aux dossiers à placer dans le système de commande source.
- L'argument *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_canCheckin()

Description

Indique si l'élément de menu Archiver doit être activé.

Arguments

```
void *connectionData, const char *localPathList[], const char
  *remotePathList[], const int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- L'argument *localPathList* est une liste de noms de chemin d'accès aux fichiers ou aux dossiers locaux à archiver.
- L'argument *remotePathList* est une liste miroir de noms de chemin d'accès aux fichiers ou aux dossiers distants.
- L'argument *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_CanUndoCheckout()

Description

Indique si l'élément de menu Annuler l'extraction doit être activé.

Arguments

```
void *connectionData, const char *remotePathList[], const char
  *localPathList[], const int numItems
```

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- L'argument *remotePathList* est une liste de noms de chemin d'accès aux fichiers et aux dossiers distant à extraire.
- *localPathList* est une liste de noms de chemin d'accès aux fichiers ou aux dossiers locaux à placer dans le système de commande source.
- L'argument *numItems* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_canNewFolder()

Description

Indique si l'élément de menu Nouveau dossier doit être activé.

Arguments

`void *connectionData, const char *remotePath`

- *`connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *`remotePath`* est une liste de noms de chemin d'accès aux fichiers ou aux dossiers distants que l'utilisateur a sélectionné pour indiquer l'emplacement de création du nouveau dossier.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_canDelete()

Description

Indique si l'élément de menu Supprimer doit être activé.

Arguments

`void *connectionData, const char *remotePathList[], const int numItems`

- *`connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- L'argument *`remotePathList`* est une liste de noms de chemin d'accès aux fichiers ou aux dossiers distants à supprimer.
- L'argument *`numItems`* est le nombre d'éléments dans chaque liste.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_canRename()

Description

Indique si l'élément de menu Renommer doit être activé.

Arguments

*void *connectionData, const char *remotePath*

- *connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.
- *remotePathList* est une liste des noms de chemin d'accès aux fichiers ou aux dossiers distants qui peuvent être renommés.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

bool SCS_BeforeGet()

Description

Dreamweaver appelle cette fonction avant d'acquérir ou d'extraire un ou plusieurs fichiers. Cette fonction permet à la DLL d'effectuer une opération sur un groupe de fichiers, telle que l'ajout d'un commentaire d'extraction.

Arguments

**connectionData*

- **connectionData* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

Exemple

Pour acquérir un groupe de fichiers, Dreamweaver effectue des appels vers la DLL dans l'ordre suivant :

```
SCS_BeforeGet(connectionData);
SCS_Get(connectionData,remotePathList1,localPathList1,¬
successList1);
SCS_Get(connectionData,remotePathList2,localPathList2,¬
successList2);
SCS_Get(connectionData,remotePathList3,localPathList3,¬
successList3);
SCS_AfterGet(connectionData);
```

bool SCS_BeforePut()

Description

Dreamweaver appelle cette fonction avant de placer ou d'archiver un ou plusieurs fichiers. Cette fonction permet à la DLL d'effectuer une opération sur un groupe de fichiers, telle que l'ajout d'un commentaire d'archivage.

Arguments

`*connectionData`

- *`*connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

Exemple

Pour acquérir un groupe de fichiers, Dreamweaver effectue des appels vers la DLL dans l'ordre suivant :

```
SCS_BeforePut(connectionData);
SCS_Put(connectionData,localPathList1,remotePathList1,¬
successList1);
SCS_Put(connectionData,localPathList2,remotePathList2,¬
successList2);
SCS_Put(connectionData,localPathList3,remotePathList3,¬
successList3);
SCS_AfterPut(connectionData);
```

bool SCS_AfterGet()

Description

Dreamweaver appelle cette fonction après avoir acquis ou extrait un ou plusieurs fichiers. Cette fonction permet à la DLL d'effectuer n'importe quelle opération après l'acquisition ou l'extraction d'un lot, telle que la création d'une boîte de dialogue de résumé.

Arguments

`*connectionData`

- *`*connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

Exemple

Voir *bool SCS_BeforeGet()*, page 147.

bool SCS_AfterPut()

Description

Dreamweaver appelle cette fonction après avoir placé ou archivé un ou plusieurs fichiers. Cette fonction permet à la DLL d'effectuer n'importe quelle opération après le placement ou l'archivage d'un lot, telle que la création d'une boîte de dialogue de résumé.

Arguments

`*connectionData`

- *`*connectionData`* est un argument qui pointe vers les données de l'agent transmises à Dreamweaver pendant l'appel `Connect()`.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

Exemple

Voir *bool SCS_BeforePut()*, page 148.

PARTIE 2

API JavaScript

2

Utilisez l'une des principales fonctions Javascript disponibles (Macromedia Dreamweaver 8 en propose plus de 600), qui encapsulent les tâches exécutées par les utilisateurs lors de la création ou de la modification d'un document dans Dreamweaver. Vous pouvez utiliser ces fonctions pour réaliser presque toutes les opérations que l'utilisateur peut effectuer à l'aide des menus, des panneaux flottants, des inspecteurs de propriétés, du panneau Site ou de la fenêtre de document.

CHAPITRE 10

Application

10

Les fonctions relatives aux applications effectuent des opérations ayant trait à la façon dont Macromedia Dreamweaver 8 interagit avec d'autres applications ou des tâches Dreamweaver indépendantes des documents (définition des préférences ou fermeture de Dreamweaver, par exemple).

Fonctions relatives aux applications externes

Ces fonctions permettent d'effectuer des opérations relatives à l'application Macromedia Flash, aux navigateurs et aux éditeurs externes, définis respectivement dans les catégories Aperçu dans le navigateur et Editeurs externes des Préférences. Elles permettent d'obtenir des informations sur les applications externes et d'ouvrir des fichiers dans ces applications.

dreamweaver.browseDocument()

Disponibilité

Dreamweaver 2, améliorée dans les versions 3 et 4.

Description

Ouvre l'URL spécifiée dans le navigateur spécifié.

Arguments

fileName, {browser}

- L'argument *fileName* correspond au nom du fichier à ouvrir, exprimé sous la forme d'une URL absolue.

REMARQUE	Certains navigateurs ne sont pas en mesure de trouver un fichier dont l'URL contient une ancre (« Configuration/ExtensionHelp/browseHelp.htm#helpyou », par exemple).

- L'argument *browser*, ajouté à Dreamweaver 3, définit un navigateur. Cet argument peut être le nom d'un navigateur tel qu'il est défini dans la catégorie Aperçu dans le navigateur des Préférences ou être `'primary'` ou `'secondary'`. Si l'argument n'est pas défini, l'URL s'ouvre dans le navigateur principal de l'utilisateur.

Valeurs renvoyées

Aucune.

Exemple

La fonction suivante utilise la fonction `dreamweaver.browseDocument()` pour ouvrir la page d'accueil Hotwired dans un navigateur :

```
function goToHotwired(){
  dreamweaver.browseDocument('http://www.hotwired.com/');
}
```

Dans Dreamweaver 4, vous pouvez faire en sorte que cette opération ouvre le document dans Microsoft Internet Explorer à l'aide du code suivant :

```
function goToHotwired(){
  var prevBrowsers = dw.getBrowserList();
  var theBrowser = "";
  for (var i=1; i < prevBrowsers.length; i+2){
    if (prevBrowsers[i].indexOf('Iexplore.exe') != -1){
      theBrowser = prevBrowsers[i];
      break;
    }
  }
  dw.browseDocument('http://www.hotwired.com/',theBrowser);
}
```

Pour plus d'informations sur la fonction `dreamweaver.getBrowserList()`, voir *dreamweaver.getBrowserList()*, page 155.

dreamweaver.getBrowserList()

Disponibilité

Dreamweaver 3.

Description

Obtient la liste des navigateurs définis dans le sous-menu Fichier > Aperçu dans le navigateur.

Arguments

Aucun.

Valeurs renvoyées

Tableau contenant autant de paires de chaînes que de navigateurs définis. La première chaîne de la paire représente le nom du navigateur, et la seconde son emplacement sur l'ordinateur de l'utilisateur, exprimé sous la forme d'une URL de type file://. Si aucun navigateur n'est défini, la fonction ne renvoie rien.

dreamweaver.getExtensionEditorList()

Disponibilité

Dreamweaver 3

Description

Obtient la liste des éditeurs définis dans la catégorie Editeurs externes des préférences pour le fichier spécifié.

Arguments

`fileURL`

- L'argument *`fileURL`* peut être une URL complète de type file://, un nom de fichier ou une extension de fichier (point compris).

Valeurs renvoyées

Tableau contenant autant de paires de chaînes que d'éditeurs définis. La première chaîne de la paire représente le nom de l'éditeur, et la seconde son emplacement sur l'ordinateur de l'utilisateur, exprimé sous la forme d'une URL de type file://. Si aucun éditeur n'est défini dans les préférences, la fonction renvoie un tableau contenant une chaîne vide.

Exemple

Un appel à la fonction `dreamweaver.getExtensionEditorList(".gif")` pourrait renvoyer un tableau contenant les chaînes suivantes :

- `"Fireworks 3"`
- `"file:///C|/Program Files/Macromedia/Fireworks 3/Fireworks 3.exe"`

dreamweaver.getExternalTextEditor()

Disponibilité

Dreamweaver 4.

Description

Affiche le nom de l'éditeur de texte externe actuellement configuré.

Arguments

Aucun.

Valeurs renvoyées

Une chaîne contenant le nom de l'éditeur de texte approprié pour l'interface utilisateur et non le chemin entier.

dreamweaver.getFlashPath()

Disponibilité

Dreamweaver MX.

Description

Obtient le chemin d'accès complet de l'application Flash MX sous la forme d'une URL de fichier.

Arguments

Aucun.

Valeurs renvoyées

Tableau contenant deux éléments. Elément [0] est une chaîne contenant le nom de l'éditeur Flash MX. Elément [1] est une chaîne contenant le chemin de l'application Flash sur l'ordinateur local, exprimé sous la forme d'une URL de type file:///. Si Flash n'est pas installé, la fonction ne renvoie rien.

Exemple

L'exemple suivant appelle la fonction `dw.getFlashPath()` pour obtenir le chemin d'accès à l'application Flash et transmettre ce dernier sous la forme d'une URL de type file:// à la fonction `dw.openWithApp()` afin d'ouvrir le document avec Flash :

```
var myDoc = dreamweaver.getDocumentDOM();

if (dreamweaver.validateFlash()) {
  var flashArray = dreamweaver.getFlashPath();
  dreamweaver.openWithApp(myDoc.myForm.swfFilePath, flashArray[1]);
}
```

dreamweaver.getPrimaryBrowser()

Disponibilité

Dreamweaver 3.

Description

Obtient le chemin du navigateur principal.

Arguments

Aucun.

Valeurs renvoyées

Une chaîne contenant le chemin d'accès au navigateur principal sur l'ordinateur de l'utilisateur, exprimé sous la forme d'une URL de type file://, ou rien, si aucun navigateur principal n'est défini.

dreamweaver.getPrimaryExtensionEditor()

Disponibilité

Dreamweaver 3.

Description

Obtient l'éditeur principal associé au fichier spécifié.

Arguments

`fileURL`

- L'argument *`fileURL`* correspond au chemin d'accès au fichier à ouvrir, exprimé sous la forme d'une URL de type file://.

Valeurs renvoyées

Tableau contenant une paire de chaînes. La première chaîne de la paire représente le nom de l'éditeur, et la seconde son emplacement sur l'ordinateur de l'utilisateur, exprimé sous la forme d'une URL de type file:///. Si aucun éditeur principal n'est défini, la fonction renvoie un tableau contenant une chaîne vide.

dreamweaver.getSecondaryBrowser()

Disponibilité

Dreamweaver 3.

Description

Obtient le chemin du navigateur secondaire.

Arguments

Aucun.

Valeurs renvoyées

Une chaîne contenant le chemin d'accès au navigateur secondaire sur l'ordinateur de l'utilisateur, exprimé sous la forme d'une URL de type file:///, ou rien, si aucun navigateur secondaire n'est défini.

dreamweaver.openHelpURL()

Disponibilité

Dreamweaver MX.

Description

Ouvre le fichier d'aide spécifié dans le visualisateur d'aide du système d'exploitation.

Dreamweaver affiche le contenu de l'aide dans le visualisateur d'aide standard du système d'exploitation plutôt que dans un navigateur. L'aide est au format HTML, mais elle peut être lue par HTML Help sous Windows ou Help Viewer sur Macintosh OS X.

L'aide complète est contenue dans les quatre types de fichiers suivants. Pour plus d'informations sur les fichiers d'aide, consultez la documentation de votre système d'exploitation.

- Manuel d'aide

Un manuel d'aide se compose de fichiers d'aide HTML, d'images et d'index. Sous Windows, le manuel d'aide est un fichier portant l'extension .chm. Sur Macintosh, le manuel d'aide est un dossier.

Les fichiers du manuel d'aide se trouvent dans le dossier d'aide de Dreamweaver.

- Fichier help.xml

  Le fichier help.xml établit des correspondances entre les ID et le nom des manuels d'aide. Par exemple, le code XML ci-dessous établit une correspondance entre l'ID du manuel d'aide de Dreamweaver et le nom des fichiers contenant cette aide, sous Windows comme sur Macintosh :

  ```
  <?xml version = "1.0" ?>
  <help-books>
  <book-id id="DW_Using" win-mapping="UsingDreamweaver.chm" mac-
  mapping="Dreamweaver Help"/>
  </help-books>
  ```

  Chaque entrée `book-id` comporte les attributs suivants :

  - L'attribut `id` correspond à l'ID de manuel utilisé dans les fichiers help.map et HelpDoc.js.
  - L'attribut `win-mapping` correspond au nom du manuel Windows, soit `"UsingDreamweaver.chm"` dans cet exemple.
  - L'attribut `mac-mapping` correspond au nom du manuel Macintosh, soit `"Dreamweaver Help"` dans cet exemple.

- Fichier help.map

  Le fichier help.map établit une correspondance entre une ID de contenu de l'aide et un manuel d'aide spécifique. Dreamweaver utilise le fichier help.map pour rechercher un contenu d'aide spécifique lorsqu'il appelle l'aide directement.

- Fichier helpDoc.js

  Le fichier helpDoc.js vous permet d'établir une correspondance entre des noms de variables que vous pouvez utiliser à la place de l'ID du manuel et de la chaîne de page. Le fichier helpDoc.js établit une correspondance entre un ID de contenu et une page HTML dans un manuel d'aide spécifique. Dreamweaver utilise le fichier helpDoc.js lorsqu'il appelle l'aide à partir de JavaScript.

Arguments

`bookID`

- L'argument obligatoire `bookID` se présente au format suivant :

  `ID:page`

L'attribut `ID` correspond à l'ID de manuel `bookID` de l'entrée figurant dans le fichier help.xml qui indique le fichier renfermant l'aide à afficher. L'attribut `page` de l'entrée identifie la page à afficher. Les pages sont référencées dans le fichier help.map.

Valeurs renvoyées

`true` en cas de réussite et `false` si Dreamweaver ne peut pas ouvrir le fichier spécifié dans le visualisateur d'aide.

Exemple

```
openHelpURL("DW_Using:index.htm");
```

dreamweaver.openWithApp()

Disponibilité

Dreamweaver 3.

Description

Ouvre le fichier spécifié dans l'application spécifiée.

Arguments

`fileURL, appURL`

- L'argument *`fileURL`* correspond au chemin d'accès au fichier à ouvrir, exprimé sous la forme d'une URL de type file://.
- L'argument *`appURL`* correspond au chemin d'accès à l'application dans laquelle ouvrir le fichier, exprimé sous forme d'une URL de type file://.

Valeurs renvoyées

Aucune.

dreamweaver.openWithBrowseDialog()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Sélectionner un éditeur externe pour permettre à l'utilisateur de sélectionner l'application dans laquelle ouvrir le fichier spécifié.

Arguments

`fileURL`

- L'argument *fileURL* correspond au chemin d'accès au fichier à ouvrir, exprimé sous la forme d'une URL de type file://.

Valeurs renvoyées

Aucune.

dreamweaver.openWithExternalTextEditor()

Disponibilité

Dreamweaver 3.

Description

Ouvre le document actif dans l'éditeur de texte externe défini dans la section Editeurs externes de la boîte de dialogue Préférences.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.openWithImageEditor()

Disponibilité

Dreamweaver 3.

Description

Ouvre le fichier nommé dans l'éditeur d'image spécifié.

REMARQUE	Cette fonction fait appel à un mécanisme spécial d'intégration Macromedia Fireworks qui renvoie les informations au document actif lorsque Fireworks est spécifié comme éditeur d'image. Pour éviter les erreurs lorsqu'aucun document n'est actif, n'appelez jamais cette fonction à partir du panneau Site.

Arguments

fileURL, appURL

- L'argument *fileURL* correspond au chemin d'accès au fichier à ouvrir, exprimé sous la forme d'une URL de type file://.

- L'argument *appURL* correspond au chemin d'accès à l'application dans laquelle ouvrir le fichier, exprimé sous forme d'une URL de type file://.

Valeurs renvoyées

Aucune.

dreamweaver.validateFlash()

Disponibilité

Dreamweaver MX.

Description

Détermine si Flash MX (ou une version ultérieure) est installé sur l'ordinateur local.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si Flash MX (ou une version ultérieure) est installé sur l'ordinateur local ; `false` dans le cas contraire.

Fonctions globales relatives aux applications

Ces fonctions agissent sur l'ensemble d'une application. Elles permettent d'effectuer des tâches comme quitter une application ou accéder aux préférences.

dreamweaver.beep()

Disponibilité

Dreamweaver MX.

Description

Crée un bip système.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant appelle la fonction `dw.beep()` pour attirer l'attention de l'utilisateur sur un message affiché par la fonction `alert()` :

```
beep(){
  if(confirm(“Is your order complete?”)
  {
    dreamweaver.beep();
    alert(“Click OK to submit your order”);
  }
}
```

dreamweaver.getShowDialogsOnInsert()

Disponibilité

Dreamweaver 3.

Description

Vérifie si l'option Afficher la boîte de dialogue lors de l'insertion d'objets est activée dans la catégorie Général des préférences.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne qui indique si cette option est activée.

dreamweaver.quitApplication()

Disponibilité

Dreamweaver 3.

Description

Quitte Dreamweaver lorsque l'exécution du script appelant cette fonction prend fin.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.showAboutBox()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue A propos de.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.showDynamicDataDialog()

Disponibilité

Dreamweaver UltraDev 1.

Description

Affiche la boîte de dialogue Données dynamiques ou Texte dynamique, puis attend que l'utilisateur ferme la boîte de dialogue. Si l'utilisateur clique sur OK, la fonction `showDynamicDataDialog()` renvoie une chaîne à insérer dans le document de l'utilisateur. Cette chaîne est renvoyée par la fonction API de la source des données, c'est-à-dire `generateDynamicDataRef()`, et transmise à la fonction API du format de données, `formatDynamicDataRef()` ; la valeur renvoyée par `formatDynamicDataRef()` est la même que celle renvoyée par la fonction `showDynamicDataDialog()`.

Arguments

`source, {title}`

- L'argument *`source`* est une chaîne contenant le code source qui représente l'objet de données dynamique. Il s'agit de la même chaîne que celle renvoyée après un appel précédent de la même fonction. Cette fonction utilise le contenu de l'argument *`source`* pour initialiser toutes les commandes de la boîte de dialogue afin qu'elles apparaissent exactement telles qu'elles étaient au moment où l'utilisateur a cliqué sur OK pour créer cette chaîne.

Dreamweaver transmet cette chaîne à la fonction `inspectDynamicDataRef()` pour déterminer si la chaîne correspond à l'un des nœuds de l'arborescence. Si la chaîne correspond à un nœud, celui-ci est sélectionné lorsque la boîte de dialogue réapparaît. Vous pouvez également transmettre une chaîne vide qui n'initialise pas la boîte de dialogue. Par exemple, aucune boîte de dialogue n'est initialisée lorsqu'elle est utilisée pour créer un nouvel élément.

- L'argument *`title`*, qui est facultatif, est une chaîne qui contient le texte à afficher dans la barre de titre de la boîte de dialogue. S'il n'est pas défini, Dreamweaver affiche Données dynamiques dans la barre de titre.

Valeurs renvoyées

Chaîne qui représente l'objet de données dynamique, si l'utilisateur clique sur OK.

dreamweaver.showPasteSpecialDialog()

Disponibilité

Dreamweaver 8

Description

Affiche la boîte de dialogue Collage spécial. Si l'utilisateur clique sur OK, la fonction `showPasteSpecialDialog()` effectue le collage.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

```
dw.showPasteSpecialDialog();
```

dreamweaver.showPreferencesDialog()

Disponibilité

Dreamweaver 3. Ajout de l'argument *`strCategory`* dans Dreamweaver 8.

Description

La fonction ouvre la boîte de dialogue Préférences.

Arguments

{strCategory}

- L'argument facultatif *strCategory* doit correspondre à l'une des chaînes suivantes pour ouvrir la catégorie correspondante dans la boîte de dialogue Préférences : `"general"`, `"accessibility"`, `"html colors"` (pour la catégorie Coloration du code), `"html format"` (pour la catégorie Format du code), `"code hints"`, `"html rewriting"` (pour la catégorie Correction du code), `“copyPaste”`, `"css styles"`, `"file compare"`, `"external editors"` (pour la catégorie Types de fichiers/Editeurs), `"fonts"`, `"highlighting"`, `"invisible elements"`, `"layers"`, `"layout mode"`, `"new document"`, `"floaters"` (pour la catégorie Panneaux), `"browsers"` (pour la catégorie Aperçu dans le navigateur), `"site ftp"` (pour la catégorie Site), `"status bar"` et `"validator"`. Si Dreamweaver ne reconnaît pas dans l'argument un nom de volet valide ou qu'aucun argument n'est défini, la boîte de dialogue ouvre le dernier volet actif.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant ouvre la boîte de dialogue Préférences et sélectionne la catégorie Coloration du code :

```
dw.showPreferencesDialog(“html colors”);
```

dreamweaver.showTagChooser()

Disponibilité

Dreamweaver MX.

Description

Active ou désactive l'affichage de la boîte de dialogue Sélecteur de balise afin que les utilisateurs insèrent des balises en mode Code. La fonction affiche la boîte de dialogue Sélecteur de balise au-dessus de toutes les autres fenêtres Dreamweaver. Si la boîte de dialogue n'est pas visible, la fonction l'ouvre, l'affiche au premier plan et en fait l'élément actif. Si le sélecteur de balise est visible, la fonction masque la boîte de dialogue.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

CHAPITRE 11

Espace de travail

11

Les fonctions API de l'espace de travail créent des éléments dans l'espace de travail Macromedia Dreamweaver 8 ou effectuent des actions dessus. Elles permettent d'effectuer diverses tâches dont voici un aperçu : refaire les étapes qui s'affichent dans le panneau Historique, placer un objet dans la barre Insérer, se déplacer à l'aide des fonctions relatives au clavier, recharger des menus, manipuler des fenêtres de résultats indépendantes ou intégrées, définir des options, positionner une barre d'outils et obtenir la sélection ou définir l'élément actif.

Fonctions relatives à l'historique

Ces fonctions permettent d'annuler, de refaire, d'enregistrer et de lire toutes les étapes qui s'affichent dans le panneau Historique. Par étape, on entend une modification apportée à un document (ou à une sélection dans le document) pouvant être reproduite. Les méthodes de l'objet `dreamweaver.historyPalette` permettent de contrôler ou d'agir sur la sélection dans le panneau Historique et non dans le document actif.

dom.redo()

Disponibilité

Dreamweaver 3.

Description

Répète l'étape qui vient juste d'être annulée dans le document.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canRedo()*, page 578.

dom.undo()

Disponibilité

Dreamweaver 3.

Description

Annule l'étape qui vient d'être exécutée dans le document.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canUndo()*, page 581 .

dreamweaver.getRedoText()

Disponibilité

Dreamweaver 3.

Description

Obtient le texte associé à l'opération de modification à répéter si l'utilisateur choisit Edition > Répéter ou s'il appuie sur Ctrl+Y (Windows) ou sur Commande+Y (Macintosh).

Arguments

Aucun.

Valeurs renvoyées

Chaîne contenant le texte associé à l'opération de modification à répéter.

Exemple

Si la dernière action de l'utilisateur a consisté à mettre le texte sélectionné en gras, un appel à la fonction `dreamweaver.getRedoText()` renvoie `"Repeat Apply Bold"`.

dreamweaver.getUndoText()

Disponibilité

Dreamweaver 3.

Description

Obtient le texte associé à l'opération de modification à annuler si l'utilisateur choisit Edition > Annuler ou s'il appuie sur Ctrl+Z (Windows) ou sur Commande+Z (Macintosh).

Arguments

Aucun.

Valeurs renvoyées

Chaîne contenant le texte associé à l'opération de modification à annuler.

Exemple

Si la dernière action de l'utilisateur a consisté à appliquer un style CSS (cascading style sheet) à un texte sélectionné, un appel à la fonction `dreamweaver.getUndoText()` renvoie `"Undo Apply <span>"`.

dreamweaver.playRecordedCommand()

Disponibilité

Dreamweaver 3.

Description

Exécute la commande mémorisée dans le document actif.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canPlayRecordedCommand()*, page 587.

dreamweaver.redo()

Disponibilité

Dreamweaver 3.

Description

Répète l'étape qui vient juste d'être annulée dans la fenêtre de document, la boîte de dialogue, le panneau flottant ou le panneau Site en cours.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canRedo()*, page 588.

dreamweaver.startRecording()

Disponibilité

Dreamweaver 3.

Description

Démarre la mémorisation des étapes dans le document actif ; la commande précédemment mémorisée est immédiatement éliminée.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.isRecording()*, page 597 (doit renvoyer la valeur `false`).

dreamweaver.stopRecording()

Disponibilité

Dreamweaver 3.

Description

Arrête la mémorisation sans intervention de l'utilisateur.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.isRecording()*, page 597 (doit renvoyer la valeur `true`).

dreamweaver.undo()

Disponibilité

Dreamweaver 3.

Description

Annule l'étape précédente dans la fenêtre de document, la boîte de dialogue, le panneau flottant ou le panneau Site actif.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canUndo()*, page 581 .

dreamweaver.historyPalette.clearSteps()

Disponibilité

Dreamweaver 3.

Description

Efface toutes les étapes du panneau Historique et désactive les éléments de menu Annuler et Rétablir.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.historyPalette.copySteps()

Disponibilité

Dreamweaver 3.

Description

Copie les étapes spécifiées de l'historique dans le Presse-papiers. Dreamweaver prévient l'utilisateur de la possibilité de conséquences inattendues au cas où les étapes spécifiées contiendraient une action impossible à reproduire.

Arguments

arrayOfIndices

- L'argument *arrayOfIndices* est un tableau d'index de position dans le panneau Historique.

Valeurs renvoyées

Chaîne qui contient le code JavaScript correspondant aux étapes spécifiées de l'historique.

Exemple

L'exemple suivant copie les quatre premières étapes dans le panneau Historique :

```
dreamweaver.historyPalette.copySteps([0,1,2,3]);
```

dreamweaver.historyPalette.getSelectedSteps()

Disponibilité

Dreamweaver 3.

Description

Détermine quelle section du panneau Historique est sélectionnée.

Arguments

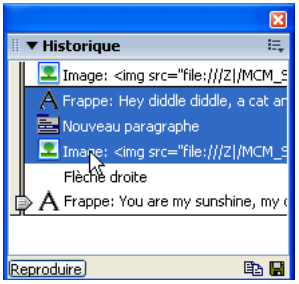
Aucun.

Valeurs renvoyées

Tableau qui contient les index de position de toutes les étapes sélectionnées. La première position est la position 0 (zéro).

Exemple

Si la deuxième, la troisième et la quatrième étapes sont sélectionnées dans le panneau Historique, comme le montre la figure ci-après, un appel à la fonction `dreamweaver.historyPalette.getSelectedSteps()` renvoie `[1,2,3]` :



dreamweaver.historyPalette.getStepCount()

Disponibilité

Dreamweaver 3.

Description

Obtient le nombre d'étapes figurant dans le panneau Historique.

Arguments

Aucun.

Valeurs renvoyées

Nombre entier qui représente le nombre d'étapes répertoriées actuellement dans le panneau Historique.

dreamweaver.historyPalette.getStepsAsJavaScript()

Disponibilité

Dreamweaver 3.

Description

Obtient l'équivalent JavaScript des étapes sélectionnées.

Arguments

arrayOfIndices

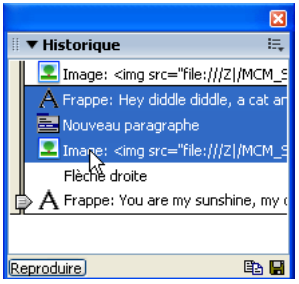
- L'argument *arrayOfIndices* est un tableau d'index de position dans le panneau Historique.

Valeurs renvoyées

Chaîne qui contient le code JavaScript correspondant aux étapes spécifiées de l'historique.

Exemple

Si les trois étapes indiquées dans l'illustration ci-après sont sélectionnées dans le panneau Historique, un appel à la fonction `dreamweaver.historyPalette.getStepsAsJavaScript(dw.historyPalette.getSelectedSteps())` renvoie `"dw.getDocumentDOM().insertText('Lon lon lon, le chat et le violon, par-dessus la lune la vache a fait un bond.');\ndw.getDocumentDOM().newBlock();\ndw.getDocumentDOM().insertHTML('<img src=\ "../wdw99/50browsers/images/sun.gif\">', true);\n"`:



dreamweaver.historyPalette.getUndoState()

Disponibilité

Dreamweaver 3.

Description

Obtient l'état d'annulation en cours.

Arguments

Aucun.

Valeurs renvoyées

La position du marqueur d'annulation dans le panneau Historique.

dreamweaver.historyPalette.replaySteps()

Disponibilité

Dreamweaver 3.

Description

Réexécute les étapes spécifiées de l'historique dans le document actif. Dreamweaver prévient l'utilisateur de la possibilité de conséquences inattendues dans le cas où les étapes spécifiées contiendraient une action qui ne peut pas être reproduite.

Arguments

`arrayOfIndices`

- L'argument *`arrayOfIndices`* est un tableau d'index de position dans le panneau Historique.

Valeurs renvoyées

Chaîne qui contient le code JavaScript correspondant aux étapes spécifiées de l'historique.

Exemple

Un appel à la fonction `dreamweaver.historyPalette.replaySteps([0,2,3])` réexécute la première, la troisième et la quatrième étapes dans le panneau Historique.

dreamweaver.historyPalette.saveAsCommand()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Enregistrer comme commande et permet à l'utilisateur d'enregistrer les étapes spécifiées sous forme de commande. Dreamweaver prévient l'utilisateur de la possibilité de conséquences inattendues dans le cas où les étapes spécifiées contiendraient une action qui ne peut pas être reproduite.

Arguments

arrayOfIndices

- L'argument *arrayOfIndices* est un tableau d'index de position dans le panneau Historique.

Valeurs renvoyées

Chaîne qui contient le code JavaScript correspondant aux étapes spécifiées de l'historique.

Exemple

L'exemple suivant enregistre la quatrième, la sixième et la huitième étapes du panneau Historique comme une commande :

```
dreamweaver.historyPalette.saveAsCommand([3,5,7]);
```

dreamweaver.historyPalette.setSelectedSteps()

Disponibilité

Dreamweaver 3.

Description

Sélectionne les étapes spécifiées dans le panneau Historique.

Arguments

arrayOfIndices

- L'argument *arrayOfIndices* est un tableau d'index de position dans le panneau Historique. Si aucun argument n'est défini, aucune étape n'est sélectionnée.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant sélectionne la première, la deuxième et la troisième étapes dans le panneau Historique :

```
dreamweaver.historyPalette.setSelectedSteps([0,1,2]);
```

dreamweaver.historyPalette.setUndoState()

Disponibilité

Dreamweaver 3.

Description

Effectue le nombre d'annulations et de répétitions nécessaires pour arriver à l'état d'annulation spécifié.

Arguments

undoState

- L'argument *undoState* correspond à l'objet renvoyé par la fonction `dreamweaver.historyPalette.getUndoState()`.

Valeurs renvoyées

Aucune.

Fonctions d'insertion d'objets

Les fonctions d'insertion d'objets permettent d'effectuer des tâches ayant trait aux objets de la barre Insérer ou du menu Insertion.

dom.insertFlashElement()

Disponibilité

Dreamweaver MX 2004.

Description

Insère l'élément Flash (fichier SWC) spécifié dans le document actif. Cette fonction suppose que l'élément Flash a été ajouté à la barre Insérer et que le fichier du composant se trouve dans le dossier ou le sous-dossier Configuration/Objects/FlashElements.

Arguments

swcFilename

- La chaîne *swcFilename* indique le chemin d'accès et le nom du composant Flash désiré par rapport au dossier Configuration/Objects/FlashElements.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant insère le composant Flash de la barre de navigation figurant dans le dossier Components/Objects/FlashElements/Navigation dans le document actif :

```
dom.insertFlashElement("\Navigation\navBar.swc");
```

dreamweaver.objectPalette.getMenuDefault()

Disponibilité

Dreamweaver MX 2004.

Description

Extrait la chaîne d'ID de l'élément par défaut pour le menu associé.

Arguments

menuId

- L'argument *menuId* correspond à la chaîne définissant le menu dans le fichier insertbar.xml.

Valeurs renvoyées

Valeur de la chaîne définissant l'ID de l'élément par défaut.

Exemple

L'exemple suivant attribue l'objet par défaut actif du menu Support à la variable `defID` :

```
var defId = dw.objectPalette.getMenuDefault("DW_Media");
```

dreamweaver.objectPalette.setMenuDefault()

Disponibilité

Dreamweaver MX 2004.

Description

Définit l'objet par défaut d'un menu déroulant. L'icône de l'objet par défaut représente le menu déroulant spécifié dans la barre Insérer. L'utilisateur peut cliquer sur l'objet par défaut pour l'insérer ou cliquer sur la flèche figurant en regard pour ouvrir le menu déroulant correspondant afin d'afficher les autres objets de ce menu. Dreamweaver prendra en compte la nouvelle valeur par défaut la prochaine fois que l'utilisateur ouvrira Dreamweaver ou utilisera la commande Recharger extensions.

Arguments

menuId, defaultId

- L'argument *menuId* correspond à la chaîne définissant le menu dans le fichier insertbar.xml.
- L'argument *defaultId* correspond à la chaîne définissant le nouvel objet par défaut dans le champ insertbar.xml.

Valeurs renvoyées

Valeur booléenne : `true` si la configuration de la nouvelle valeur par défaut réussit et `false` dans le cas contraire.

Exemple

L'exemple suivant définit l'objet Flash en qualité d'objet par défaut pour le menu Support :

```
dw.objectPalette.setMenuDefault("DW_Media", "DW_Flash");
```

dreamweaver.reloadObjects()

Disponibilité

Dreamweaver MX 2004.

Description

Recharge tous les objets de la barre Insérer. Cette fonction revient à maintenir la touche Ctrl (Windows) ou Alt (Macintosh) enfoncée tout en cliquant dans le menu des catégories de la barre Insérer et à sélectionner l'option de menu Recharger extensions.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le chargement des objets a réussi et `false` dans le cas contraire.

Fonctions relatives au clavier

Les fonctions relatives au clavier émulent les opérations de déplacement au sein d'un document effectuées à l'aide des touches de direction RET. ARR, SUPPR., PG. PREC et PG. SUIV. Parallèlement aux fonctions générales, telles que `arrowLeft()` (équivalent à la touche de direction GAUCHE) et `backspaceKey()` (équivalent à la touche RET. ARR), Dreamweaver propose également des méthodes permettant d'accéder soit au mot ou paragraphe suivant ou précédent, soit au début ou à la fin d'une ligne ou d'un document.

dom.arrowDown()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion vers le bas du nombre de lignes spécifié.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument *nTimes* correspond au nombre de fois où le point d'insertion doit se déplacer vers le bas. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.arrowLeft()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion vers la gauche du nombre de colonnes spécifié.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* correspond au nombre de fois où le point d'insertion doit se déplacer vers la gauche. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.arrowRight()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion vers la droite du nombre de colonnes spécifié.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* correspond au nombre de fois où le point d'insertion doit se déplacer vers la droite. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.arrowUp()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion vers le haut du nombre de lignes spécifié.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* correspond au nombre de fois où le point d'insertion doit se déplacer vers le haut. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.backspaceKey()

Disponibilité

Dreamweaver 3.

Description

Cette fonction revient à appuyer sur la touche RET. ARR un nombre de fois donné. Le résultat obtenu diffère selon qu'il y ait une sélection en cours ou simplement un point d'insertion.

Arguments

{nTimes}

- L'argument facultatif *nTimes* correspond au nombre de fois où une opération de retour arrière doit avoir lieu. Si l'argument n'est pas défini, il prend par défaut la valeur 1.

Valeurs renvoyées

Aucune.

dom.deleteKey()

Disponibilité

Dreamweaver 3.

Description

Cette fonction revient à appuyer sur la touche SUPPR. un nombre de fois donné. Le résultat obtenu diffère selon qu'il y ait une sélection en cours ou simplement un point d'insertion.

Arguments

{nTimes}

- L'argument facultatif *nTimes* correspond au nombre de fois où une opération de suppression doit avoir lieu. Si l'argument n'est pas défini, il prend par défaut la valeur 1.

Valeurs renvoyées

Aucune.

dom.endOfDocument()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion à la fin du document, dans la fenêtre active (selon le cas, après le dernier contenu visible dans la fenêtre de document ou après la balise `HTML` de fermeture dans l'inspecteur de code).

Arguments

`{bShiftIsDown}`

- L'argument facultatif *`bShiftIsDown`* est une valeur booléenne qui indique s'il faut étendre la sélection. Si l'argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.endOfLine()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion à la fin de la ligne.

Arguments

`{bShiftIsDown}`

- L'argument facultatif *`bShiftIsDown`* est une valeur booléenne qui indique s'il faut étendre la sélection. Si l'argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.nextParagraph()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion au début du paragraphe suivant ou saute plusieurs paragraphes si l'argument *`nTimes`* est supérieur à 1.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* correspond au nombre de paragraphes postérieurs que le point d'insertion doit sauter. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.nextWord()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion au début du mot suivant ou saute plusieurs mots si l'argument *nTimes* est supérieur à 1.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* correspond au nombre de mots postérieurs que le point d'insertion doit sauter. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.pageDown()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion d'une page vers le bas (équivalent à la touche PG. SUIV).

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* correspond au nombre de pages postérieures que le point d'insertion doit sauter. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.pageUp()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion d'une page vers le haut (équivalent à la touche PG. PREC).

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* correspond au nombre de pages antérieures que le point d'insertion doit sauter. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.previousParagraph()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion au début du paragraphe suivant ou saute plusieurs paragraphes si l'argument *nTimes* est supérieur à 1.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* correspond au nombre de paragraphes antérieurs que le point d'insertion doit sauter. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.previousWord()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion au début du mot précédent ou saute plusieurs mots si l'argument *nTimes* est supérieur à 1.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* correspond au nombre de mots antérieurs que le point d'insertion doit sauter. Si cet argument n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.startOfDocument()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion au début du document (selon le cas, avant le premier contenu visible dans la fenêtre de document ou avant la balise HTML d'ouverture dans l'inspecteur de code).

Arguments

{bShiftIsDown}

- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si l'argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.startOfLine()

Disponibilité

Dreamweaver 3.

Description

Déplace le point d'insertion au début de la ligne.

Arguments

{bShiftIsDown}

- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique s'il faut étendre la sélection. Si l'argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dreamweaver.mapKeyCodeToChar()

Disponibilité

Dreamweaver 4.

Description

Convertit un code de touche tel qu'il est indiqué dans le champ `keyCode` de l'objet événement en caractère. Vous devez vérifier si le code de touche est une touche spéciale, telle que les touches ORIG, PG. SUIV, etc. S'il ne s'agit pas d'une touche spéciale, cette méthode peut être utilisée pour traduire le code en code de caractère affichable à l'utilisateur.

Arguments

keyCode

- L'argument *keyCode* est le code de touche à traduire en caractère.

Valeurs renvoyées

Aucune.

Fonctions relatives aux menus

Ces fonctions permettent d'optimiser et de recharger les menus dans Dreamweaver. Les fonctions `dreamweaver.getMenuNeedsUpdating()` et `dreamweaver.notifyMenuUpdated()` sont spécialement conçues pour empêcher l'exécution de routines de mise à jour superflues sur les menus dynamiques intégrés à Dreamweaver. Pour plus d'informations, voir dreamweaver.getMenuNeedsUpdating() et dreamweaver.notifyMenuUpdated().

dreamweaver.getMenuNeedsUpdating()

Disponibilité

Dreamweaver 3.

Description

Vérifie si le menu spécifié doit être mis à jour.

Arguments

`menuId`

- L'argument *`menuId`* est une chaîne contenant la valeur de l'attribut `id` de l'élément de menu, tel qu'il est défini dans le fichier menus.xml.

Valeurs renvoyées

Valeur booléenne qui indique si le menu doit être mis à jour. Cette fonction renvoie la valeur `false` uniquement si la fonction `dreamweaver.notifyMenuUpdated()` a été appelée avec cet argument *`menuId`* et si la valeur renvoyée par *`menuListFunction`* n'a pas changé. Pour plus d'informations, voir *dreamweaver.notifyMenuUpdated()*, page 188.

dreamweaver.notifyMenuUpdated()

Disponibilité

Dreamweaver 3.

Description

Prévient Dreamweaver lorsque le menu spécifié doit être mis à jour.

Arguments

menuId, menuListFunction

- L'argument *menuId* est une chaîne contenant la valeur de l'attribut `id` de l'élément de menu, tel qu'il est défini dans le fichier menus.xml.
- L'argument *menuListFunction* doit correspondre à l'une des chaînes suivantes : `"dw.cssStylePalette.getStyles()"`, `"dw.getDocumentDOM().getFrameNames()"`, `"dw.getDocumentDOM().getEditableRegionList"`, `"dw.getBrowserList()"`, `"dw.getRecentFileList()"`, `"dw.getTranslatorList()"`, `"dw.getFontList()"`, `"dw.getDocumentList()"`, `"dw.htmlStylePalette.getStyles()"` ou `"site.getSites()"`.

Valeurs renvoyées

Aucune.

dreamweaver.reloadMenus()

Disponibilité

Dreamweaver 3.

Description

Recharge la structure de menus à partir du fichier menus.xml du dossier Configuration.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Fonctions de la fenêtre de résultats

Les fonctions de la fenêtre de résultats vous permettent de créer une fenêtre indépendante contenant des colonnes de données formatées ou de dialoguer avec les fenêtres intégrées du groupe de panneaux Résultats.

Création d'une fenêtre de résultats indépendante

Ces fonctions créent des fenêtres personnalisées qui ressemblent au résultat de la fenêtre Débogueur JavaScript.

dreamweaver.createResultsWindow()

Disponibilité

Dreamweaver 4.

Description

Crée une nouvelle fenêtre de résultats et renvoie une référence d'objet JavaScript à la fenêtre.

Arguments

strName, arrColumns

- L'argument *strName* est la chaîne à utiliser pour le titre de la fenêtre.
- L'argument *arrColumns* est un tableau de noms de colonnes à utiliser dans le contrôle de liste.

Valeurs renvoyées

Référence d'objet à la fenêtre créée.

dreamweaver.showResults()

Disponibilité

Dreamweaver MX 2004.

Description

Ouvre le panneau flottant de résultats spécifié et sélectionne l'élément.

REMARQUE	Cette fonction est uniquement prise en charge dans `les` fenêtres Validation, Vérification du navigateur cible et Rapports du site du panneau flottant de résultats.

Arguments

floaterName, floaterIndex

- L'argument *floaterName* est une chaîne indiquant le panneau flottant de résultats à ouvrir. Les valeurs autorisées sont `'validation'`, `'btc'` ou `'reports'`.
- L'argument *floaterIndex* est un nombre ou une chaîne. Utilisez un nombre pour spécifier l'index d'un élément à sélectionner dans le panneau de résultats. Utilisez une chaîne pour spécifier l'URL d'un document. Dans le cas d'une URL, la fonction sélectionne le premier élément visible de ce document.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant vérifie la présence d'erreurs au début de la sélection en cours dans le document. Si des erreurs sont détectées, elles sont affichées dans la fenêtre spécifiée (`floaterName`) du panneau de résultats. Dans le cas contraire, il ouvre la fenêtre Vérification du navigateur cible du panneau de résultats et affiche le premier élément visible du document.

```
var offset = dw.getDocumentDOM().source.getSelection()[0];
var errors =
  dw.getDocumentDOM().source.getValidationErrorsForOffset(offset);
if ( errors && errors.length > 0 )
  dw.showResults( errors[0].floaterName, errors[0].floaterIndex );
else
  dw.showResults('btc', dw.getDocumentDOM().URL);
```

resWin.addItem()

Disponibilité

Dreamweaver 4.

Description

Ajoute un nouvel élément à la fenêtre de résultats.

REMARQUE

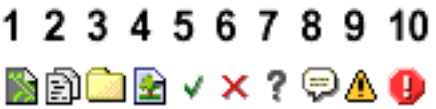
N'utilisez cette fonction que sur des fenêtres de résultats indépendantes créées à l'aide de dreamweaver.createResultsWindow(). Il est impossible d'utiliser resWin.addItem() avec des fenêtres de résultats intégrées, notamment les fenêtres Validation, Vérifier le navigateur cible ou Rapports du site.

Arguments

`resultWindowObj, strIcon, strDesc, itemData, iStartSel, iEndSel, colNdata`

- L'argument *`resultWindowObj`* correspond à l'objet renvoyé par la fonction `createResultsWindow()`.

- L'argument *strIcon* est une chaîne indiquant le chemin d'accès à l'icône à utiliser. Pour afficher une icône intégrée, utilisez une valeur comprise entre 1 et 10 plutôt que le chemin d'accès complet à l'icône. Utilisez la valeur 0 pour n'afficher aucune icône. L'illustration ci-dessous indique les icônes correspondant aux chiffres allant de 1 à 10 :



- L'argument *strDesc* correspond à la description détaillée de l'élément. Indiquez "0" s'il n'y a pas de description.
- L'argument *itemData* est une chaîne qui peut vous servir à stocker des données particulières relatives à l'élément à ajouter (numéro de ligne du document, par exemple).
- L'argument *iStartSel* indique le début du décalage de la sélection dans le fichier. Entrez la valeur null s'il n'y a pas de décalage.
- L'argument *iEndSel* indique la fin du décalage de la sélection dans le fichier. Entrez la valeur null s'il n'y a pas de décalage.
- L'argument *colNdata* est un tableau de chaînes qui contient les données de chaque colonne (par exemple, tableau de 3 chaînes s'il existe 3 colonnes).

Valeurs renvoyées

Valeur booléenne : true si l'ajout de l'élément a réussi et false dans le cas contraire.

Exemple

L'exemple suivant crée une fenêtre de résultats du nom de resWin dont les colonnes s'intitulent Frodo, Sam et Gollum. L'appel de la fonction resWin.addItem() ajoute une icône de dossier ainsi que les trois chaînes (msg1, msg2 et msg3) dans les trois colonnes définies pour la fenêtre.

```
var resWin = dw.createResultsWindow("Test Window", ["Frodo", "Sam",
  "Gollum"]);
resWin.addItem(resWin, "3", "Description", null, null, null, ["msg1",
  "msg2", "msg3"]);
```

resWin.addResultItem()

Disponibilité

Dreamweaver 4.

Description

Ajoute une nouvelle entrée dans la fenêtre de résultats en cours, en fonction des informations figurant dans le fichier traité par la fonction `processfile()`.

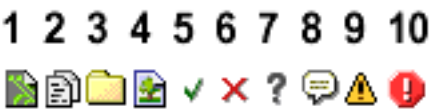
REMARQUE N'utilisez cette fonction que sur la fenêtre de résultats intégrée des rapports de site (dreamweaver.resultsPalette.siteReports). Il est impossible d'utiliser resWin.addResultItem() avec les autres fenêtres de résultats intégrées ou les fenêtres de résultats indépendantes créées avec dreamweaver.createResultsWindow().

Cette fonction est uniquement disponible dans le rappel `processFile()` d'un rapport de site. Pour plus d'informations sur les rapports de site, voir la rubrique « Rapports » dans l'aide *Extension de Dreamweaver*.

Arguments

`strFilePath, strIcon, strDisplay, strDesc, {iLineNo}, {iStartSel}, {iEndSel}`

- L'argument *`strFilePath`* est le chemin d'accès complet à l'URL du fichier à traiter.
- L'argument *`strIcon`* est le chemin d'accès à l'icône à utiliser. Pour afficher une icône intégrée, utilisez une valeur comprise entre 1 et 10 plutôt que le chemin d'accès complet à l'icône (utilisez la valeur 0 pour n'afficher aucune icône). L'illustration ci-dessous indique les icônes correspondant aux chiffres allant de 1 à 10 :



- L'argument *`strDisplay`* est la chaîne à afficher à l'utilisateur dans la première colonne de la fenêtre de résultats (généralement le nom de fichier).
- L'argument *`strDesc`* est la description qui accompagne l'entrée.
- L'argument *`iLineNo`* est le nombre de lignes figurant dans le fichier (facultatif).
- L'argument *`iStartSel`* indique le début du décalage dans le fichier (facultatif, mais s'il est présent, il faut également utiliser l'argument *`iEndSel`*).
- L'argument *`iEndSel`* indique la fin du décalage dans le fichier (obligatoire si l'argument *`iStartSel`* est utilisé).

Valeurs renvoyées

Aucune.

resWin.getItem()

Disponibilité

Dreamweaver 4.

Description

Extrait un tableau d'éléments qui incluent le nom de la commande qui a ajouté l'élément et les mêmes chaînes que celles transmises à la fonction `addItem()`.

Arguments

`itemIndex`

- L'argument *`itemIndex`* correspond à l'index de l'élément dont les données doivent être extraites.

Renvoie

Tableau de chaînes. Le premier élément du tableau correspond au nom de la commande qui a ajouté l'élément ; les autres éléments correspondent aux mêmes chaînes que celles transmises à la fonction `addItem()`.

resWin.getItemCount()

Disponibilité

Dreamweaver 4.

Description

Extrait le nombre d'éléments contenus dans la liste.

Arguments

Aucun.

Renvoie

Le nombre d'éléments contenus dans la liste.

resWin.getSelectedItem()

Disponibilité

Dreamweaver 4.

Description

Extrait l'index de l'élément sélectionné.

Arguments

Aucun.

Renvoie

L'index de l'élément sélectionné.

resWin.setButtons()

Disponibilité

Dreamweaver 4.

Description

Définit les boutons spécifiés par l'argument *arrButtons*.

Arguments

cmdDoc, *arrButtons*

- L'argument *cmdDoc* est un objet document représentant la commande utilisée pour appeler la fonction. Les commandes doivent utiliser le mot-clé *this*.
- L'argument *arrButtons* est un tableau de chaînes correspondant au texte du bouton et au code JavaScript à exécuter lorsque l'utilisateur clique sur le bouton en question. Cet argument fonctionne de la même manière que la fonction `commandButtons()` pour les commandes. Vous ne pouvez définir que deux boutons dans la fenêtre.

Renvoie

Aucune.

resWin.setCallbackCommands()

Disponibilité

Dreamweaver 4.

Description

Indique à la fenêtre de résultats sur quelles commandes la méthode `processFile()` doit être appelée. Si cette fonction n'est pas appelée, la commande qui a créé la fenêtre de résultats est appelée.

Arguments

arrCmdNames

- L'argument *arrCmdNames* est un tableau de noms de commandes sur lesquelles appeler la fonction `processFile()`.

Valeurs renvoyées

Aucune.

resWin.setColumnWidths()

Disponibilité

Dreamweaver 4.

Description

Définit la largeur de chaque colonne.

Arguments

arrWidth

- L'argument *arrWidth* est un tableau de nombres entiers représentant les largeurs à utiliser pour chaque colonne de la commande.

Valeurs renvoyées

Aucune.

resWin.setFileList()

Disponibilité

Dreamweaver 4.

Description

Indique à la fenêtre de résultats une liste de fichiers, de dossiers ou des deux sur lesquels appeler un ensemble de commandes à traiter.

Arguments

arrFilePaths, bRecursive

- L'argument *arrFilePaths* est un tableau de chemins de fichier ou de dossier à répéter.
- L'argument *bRecursive* est une valeur booléenne indiquant si la répétition doit être récurrente (`true`) ou non (`false`).

Valeurs renvoyées

Aucune.

resWin.setSelectedItem()

Disponibilité

Dreamweaver 4.

Description

Définit l'élément sélectionné sur l'élément spécifié par *itemIndex*.

Arguments

itemIndex

- L'index de l'élément à sélectionner contenu dans la liste.

Renvoie

L'index de l'ancien élément sélectionné

resWin.setTitle()

Disponibilité

Dreamweaver 4.

Description

Définit le titre de la fenêtre.

Arguments

strTitle

- L'argument *strTitle* est le nouveau titre du panneau flottant.

Valeurs renvoyées

Aucune.

resWin.startProcessing()

Disponibilité

Dreamweaver 4.

Description

Lance le traitement du fichier.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

resWin.stopProcessing()

Disponibilité

Dreamweaver 4.

Description

Arrête le traitement du fichier.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Utilisation du groupe de panneaux Résultats intégré

Ces fonctions produisent la sortie dans le groupe de panneaux Résultats. Le groupe de panneaux Résultats affiche des rapports à onglets sur les recherches, la validation de la source, les rapports au niveau du site, les cibles de navigateur, les rapports de console, les connexions FTP et les vérifications de liens.

Utilisation de panneaux enfants spécifiques

Les panneaux enfants suivants sont des fenêtres de résultats intégrées qui existent toujours dans l'interface Dreamweaver et auxquelles il est possible d'accéder directement. Ces panneaux étant des fenêtres de résultats, vous pouvez utiliser les méthodes suivantes, définies pour les fenêtres de résultats indépendantes :

- `dreamweaver.resultsPalette.siteReports`
- `dreamweaver.resultsPalette.validator`
- `dreamweaver.resultsPalette.btc` (panneau Vérification du navigateur cible)

Pour plus d'informations sur l'utilisation des méthodes `resWin`, voir *Création d'une fenêtre de résultats indépendante*, page 189.

Utilisation du panneau enfant actif

Les fonctions API générales suivantes s'appliquent au panneau enfant actif. Certains panneaux enfants peuvent ignorer certaines de ces fonctions. Si le panneau enfant actif ne prend pas la fonction en charge, l'appel n'a aucun effet.

dreamweaver.resultsPalette.clear()

Disponibilité

Dreamweaver MX.

Description

Efface le contenu du panneau actif.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.resultsPalette.canClear()*, page 597.

dreamweaver.resultsPalette.Copy()

Disponibilité

Dreamweaver MX.

Description

Envoie un message copié à la fenêtre active (souvent utilisé pour la fenêtre de connexion FTP).

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.resultsPalette.canCopy()*, page 598.

dreamweaver.resultsPalette.cut()

Disponibilité

Dreamweaver MX.

Description

Envoie un message coupé à la fenêtre active (souvent utilisé pour la fenêtre de connexion FTP).

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.resultsPalette.canCut()*, page 598.

dreamweaver.resultsPalette.Paste()

Disponibilité

Dreamweaver MX.

Description

Envoie un message collé à la fenêtre active (souvent utilisé pour la fenêtre de connexion FTP).

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.resultsPalette.canPaste()*, page 599.

dreamweaver.resultsPalette.openInBrowser

Disponibilité

Dreamweaver MX.

Description

Envoie un rapport (Rapports de site, Vérification du navigateur cible, Validation et Vérificateur de lien) vers le navigateur par défaut.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.resultsPalette.canOpenInBrowser()*, page 599.

dreamweaver.resultsPalette.openInEditor()

Disponibilité

Dreamweaver MX.

Description

Saute à la ligne sélectionnée pour les rapports spécifiques (Rapports du site, Vérification du navigateur cible, Validation et Vérificateur de lien) et ouvre le document dans l'éditeur.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.resultsPalette.canOpenInEditor()*, page 599.

dreamweaver.resultsPalette.save()

Disponibilité

Dreamweaver MX.

Description

Ouvre la boîte de dialogue Enregistrer pour une fenêtre qui prend en charge la fonction Enregistrer (Rapports du site, Vérification du navigateur cible, Validation et Vérificateur de lien).

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.resultsPalette.canSave()*, page 600.

dreamweaver.resultsPalette.selectAll()

Disponibilité

Dreamweaver MX.

Description

Envoie une commande Sélectionner tout vers la fenêtre active.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.resultsPalette.canSelectAll()*, page 600.

Débogage de serveur

Dreamweaver peut solliciter des fichiers à partir de ColdFusion et afficher la réponse dans son navigateur incorporé. Lorsque le serveur renvoie la réponse, Dreamweaver recherche un paquet de XML ayant une signature connue, dans la réponse. Si Dreamweaver trouve XML avec cette signature, il traite XML et affiche les informations contenues dans une commande d'arborescence. Cette arborescence affiche des informations sur les éléments suivants :

- l'ensemble des modèles, balises personnalisées et fichiers inclus utilisés pour créer la page CFM rendue ;

- Les exceptions
- les requêtes SQL ;
- les requêtes d'objets ;
- les variables ;
- la plage de tracé.

En outre, le panneau Débogage du serveur peut afficher des données de débogage provenant d'autres modèles de serveur. Pour configurer Dreamweaver afin que le programme débogue les autres modèles de serveur, utilisez la fonction `dreamweaver.resultsPalette.debugWindow.addDebugContextData()`.

dreamweaver.resultsPalette.debugWindow.addDebugContextData()

Disponibilité

Dreamweaver MX.

Description

Interprète un fichier XML personnalisé renvoyé du serveur spécifié dans la boîte de dialogue Définition du site. Le contenu du fichier XML affiche les données de l'arborescence dans le panneau Débogage du serveur, pour que vous puissiez utiliser le panneau Débogage du serveur en vue d'évaluer le contenu généré par le serveur à partir des différents modèles de serveur.

Arguments

`treedata`

- L'argument *`treedata`* est la chaîne XML renvoyée par le serveur. La chaîne XML doit utiliser la mise en forme suivante :

`server debug node`	Nœud racine pour les données XML de débogage
`debugnode`	Correspond à tous les nœuds
`context`	Nom de l'élément qui apparaît dans la liste de contexte
`icon`	Icône à utiliser pour le nœud d'arborescence
`name`	Nom à afficher
`value`	Valeur à afficher

`timestamp`	Uniquement applicable au nœud de contexte
Les chaînes suivantes sont facultatives :	
`jumptoline`	Etablit un lien vers un numéro de ligne spécifique
`template`	Nom de la partie du fichier de modèle de l'URL
`path`	Chemin du fichier à partir du serveur
`line number`	Numéro de ligne dans le fichier
`start position`	Début du décalage de caractères dans la ligne
`end position`	Fin du décalage de caractères dans la ligne

Exemple :

```
<serverdebuginfo>
  <context>
    <template><![CDATA[/ooo/master.cfm]]></template>
    <path><![CDATA[C:\server\wwwroot\ooo\master.cfm]]></path>
    <timestamp><![CDATA[0:0:0.0]]></timestamp>
  </context>
  <debugnode>
    <name><![CDATA[CGI]]></name>
    <icon><![CDATA[ServerDebugOutput/ColdFusion/CGIVariables.gif]]></icon>
    <debugnode>
      <name><![CDATA[Pubs.name.sourceURL]]></name>
      <icon><![CDATA[ServerDebugOutput/ColdFusion/Variable.gif]]></icon>
      <value><![CDATA[jdbc:macromedia:sqlserver://
  name.macromedia.com:1111;databaseName=Pubs]]></value>
    </debugnode>
  </debugnode>
  <debugnode>
    <name><![CDATA[Element Snippet is undefined in class
  coldfusion.compiler.TagInfoNotFoundException]]></name>
    <icon><![CDATA[ServerDebugOutput/ColdFusion/Exception.gif]]></icon>
    <jumptoline linenumber="3" startposition="2" endposition="20">
      <template><![CDATA[/ooo/master.cfm]]></template>
      <path><![CDATA[C:\Neo\wwwroot\ooo\master.cfm]]></path>
    </jumptoline>
  </debugnode>
</serverdebuginfo>
```

Valeurs renvoyées

Aucune.

Fonctions de bascule

Ces fonctions permettent d'obtenir et de définir un certain nombre d'options qui peuvent être activées ou désactivées.

dom.getEditNoFramesContent()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Modifier > Jeu de cadres > Modifier le contenu sans cadres.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu de `NOFRAMES` est l'affichage actif et `false` dans le cas contraire.

dom.getHideAllVisualAids()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si les assistances visuelles sont définies comme étant masquées.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` pour définir les assistances visuelles comme étant masquées et `false` dans le cas contraire.

dom.getPreventLayerOverlaps()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Empêcher le chevauchement des calques.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` pour activer l'option Empêcher le chevauchement des calques et `false` dans le cas contraire.

dom.getShowAutoIndent()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si l'option de retrait automatique est activée dans la fenêtre de document en mode Code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le retrait automatique est activé et `false` dans le cas contraire.

dom.getShowFrameBorders()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Bordures de cadre.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les bordures de cadre sont visibles et `false` dans le cas contraire.

dom.getShowGrid()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Grille > Afficher.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la grille est visible et `false` dans le cas contraire.

dom.getShowHeadView()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Contenu de l'en-tête.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu de l'en-tête est visible et `false` dans le cas contraire.

dom.getShowInvalidHTML()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si le code HTML incorrect est actuellement en surbrillance dans la fenêtre de document en mode Code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le code HTML incorrect est en surbrillance et `false` dans le cas contraire.

dom.getShowImageMaps()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Cartes graphiques.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les cartes graphiques sont visibles et `false` dans le cas contraire.

dom.getShowInvisibleElements()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Eléments invisibles.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les marqueurs d'éléments invisibles sont visibles et `false` dans le cas contraire.

dom.getShowLayerBorders()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Bordures de calque.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les bordures de calque sont visibles et `false` dans le cas contraire.

dom.getShowLineNumbers()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si les numéros de ligne sont affichés en mode Code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les numéros de ligne sont affichés et `false` dans le cas contraire.

dom.getShowRulers()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Règles > Afficher.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les règles sont visibles et `false` dans le cas contraire.

dom.getShowSyntaxColoring()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si l'option de coloration de la syntaxe est activée dans la fenêtre de document en mode Code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la coloration de la syntaxe est activée et `false` dans le cas contraire.

dom.getShowTableBorders()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Bordures de tableau.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les bordures de tableau sont visibles et `false` dans le cas contraire.

dom.getShowToolbar()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si la barre d'outils est visible.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la barre d'outils est affichée et `false` dans le cas contraire.

dom.getShowTracingImage()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Tracé de l'image > Afficher.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si l'option est activée et `false` dans le cas contraire.

dom.getShowWordWrap()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si l'option de retour automatique à la ligne est activée dans la fenêtre de document en mode Code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le retour automatique à la ligne est activé et `false` dans le cas contraire.

dom.getSnapToGrid()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Grille > Aligner sur.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si l'option Aligner sur la grille est activée et `false` dans le cas contraire.

dom.setEditNoFramesContent()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Modifier > Jeu de cadres > Modifier le contenu sans cadres.

Arguments

`bEditNoFrames`

- L'argument *`bEditNoFrames`* doit avoir une valeur booléenne : `true` active l'option Modifier le contenu sans cadres, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canEditNoFramesContent()*, page 574.

dom.setHideAllVisualAids()

Disponibilité

Dreamweaver 4.

Description

Cette fonction désactive l'affichage de l'ensemble des bordures, cartes graphiques et éléments invisibles, quel que soit leur paramètre individuel dans le menu Affichage.

Arguments

bSet

- L'argument *bSet* doit avoir une valeur booléenne : `true` masque les assistances visuelles tandis que `false` les affiche.

Valeurs renvoyées

Aucune.

dom.setPreventLayerOverlaps()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Empêcher le chevauchement des calques.

Arguments

bPreventLayerOverlaps

- L'argument *bPreventLayerOverlaps* doit avoir une valeur booléenne : `true` active l'option Empêcher le chevauchement des calques, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowFrameBorders()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Bordures de cadre.

Arguments

bShowFrameBorders

- L'argument *bShowFrameBorders* doit avoir une valeur booléenne : `true` active l'option Bordures de cadre, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowGrid()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Grille > Afficher.

Arguments

bShowGrid

- L'argument *bShowGrid* doit avoir une valeur booléenne : `true` active l'option Affichage > Grille > Afficher, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowHeadView()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Contenu de l'en-tête.

Arguments

bShowHead

- L'argument *bShowHead* doit avoir une valeur booléenne : `true` active l'option Contenu de l'en-tête, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowInvalidHTML()

Disponibilité

Dreamweaver 4.

Description

Cette fonction active ou désactive la mise en surbrillance du code HTML incorrect dans la fenêtre de document en mode Code.

Cette fonction détermine si le code HTML incorrect est actuellement en surbrillance.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` si la mise en surbrillance du code HTML incorrect est activée et `false` dans le cas contraire.

Valeurs renvoyées

Aucune.

dom.setShowImageMaps()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Cartes graphiques.

Arguments

bShowImageMaps

- L'argument *bShowImageMaps* est une valeur booléenne : `true` active l'option Cartes graphiques, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowInvisibleElements()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Eléments invisibles.

Arguments

bViewInvisibleElements

- L'argument *bViewInvisibleElements* doit avoir une valeur booléenne : `true` active l'option Eléments invisibles, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowLayerBorders()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Bordures de calque.

Arguments

bShowLayerBorders

- L'argument *bShowLayerBorders* est une valeur booléenne : `true` active l'option Bordures de calque, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowLineNumbers()

Disponibilité

Dreamweaver 4.

Description

Cette fonction affiche ou masque les numéros de ligne dans la fenêtre de document en mode Code.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` pour afficher les numéros de ligne et `false` dans le cas contraire.

Valeurs renvoyées

Aucune.

dom.setShowRulers()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage >Règles > Afficher.

Arguments

bShowRulers

- L'argument *bShowRulers* doit avoir une valeur booléenne : `true` active l'option Afficher, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowSyntaxColoring()

Disponibilité

Dreamweaver 4.

Description

Cette fonction active ou désactive la coloration de la syntaxe dans la fenêtre de document en mode Code.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` pour afficher la coloration de la syntaxe et `false` dans le cas contraire.

Valeurs renvoyées

Aucune.

dom.setShowTableBorders()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Bordures de tableau.

Arguments

bShowTableBorders

- L'argument *bShowTableBorders* doit avoir une valeur booléenne : `true` active l'option Bordures de tableau, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowToolbar()

Disponibilité

Dreamweaver 4.

Description

Cette fonction affiche ou masque la barre d'outils.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` pour afficher la barre d'outils et `false` pour la masquer.

Valeurs renvoyées

Aucune.

dom.setShowTracingImage()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Tracé de l'image > Afficher.

Arguments

bShowTracingImage

- L'argument *bShowTracingImage* doit avoir une valeur booléenne : `true` active l'option Afficher, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dom.setShowWordWrap()

Disponibilité

Dreamweaver 4.

Description

Cette fonction active ou désactive l'option Retour à la ligne dans la fenêtre de document en mode Code.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` pour effectuer un retour à la ligne et `false` dans le cas contraire.

Valeurs renvoyées

Aucune.

dom.setSnapToGrid()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Grille > Aligner sur.

Arguments

bSnapToGrid

- L'argument *bSnapToGrid* doit avoir une valeur booléenne : `true` active l'option Aligner sur, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dreamweaver.getHideAllFloaters()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Masquer les panneaux.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si l'option Masquer les panneaux est activée et `false` si l'option Afficher les panneaux est activée.

dreamweaver.getShowStatusBar()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Affichage > Barre d'état.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la barre d'état est visible et `false` dans le cas contraire.

dreamweaver.htmlInspector.getShowAutoIndent()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si l'option de retrait automatique est activée dans l'inspecteur de code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le retrait automatique est activé et `false` dans le cas contraire.

dreamweaver.htmlInspector.getShowInvalidHTML()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si le code HTML incorrect est actuellement en surbrillance dans l'inspecteur de code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le code HTML incorrect est en surbrillance et `false` dans le cas contraire.

dreamweaver.htmlInspector.getShowLineNumbers()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si les numéros de ligne sont affichés dans l'inspecteur de code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les numéros de ligne sont affichés et `false` dans le cas contraire.

dreamweaver.htmlInspector.getShowSyntaxColoring()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si l'option de coloration de la syntaxe est activée dans l'inspecteur de code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la coloration de la syntaxe est activée et `false` dans le cas contraire.

dreamweaver.htmlInspector.getShowWordWrap()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine si l'option de retour à la ligne est activée dans l'inspecteur de code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le retour automatique à la ligne est activé et `false` dans le cas contraire.

dreamweaver.htmlInspector.setShowAutoIndent()

Disponibilité

Dreamweaver 4.

Description

Cette fonction active ou désactive l'option de retrait automatique dans l'inspecteur de code.

Arguments

`bShow`

- L'argument *bShow* doit avoir une valeur booléenne : true active l'option de retrait automatique, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dreamweaver.htmlInspector.setShowInvalidHTML()

Disponibilité

Dreamweaver 4.

Description

Cette fonction active ou désactive la mise en surbrillance du code HTML incorrect dans l'inspecteur de code.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` pour rendre la mise en surbrillance du code HTML incorrect visible et `false` dans le cas contraire.

Valeurs renvoyées

Aucune.

dreamweaver.htmlInspector.setShowLineNumbers()

Disponibilité

Dreamweaver 4.

Description

Cette fonction affiche ou masque les numéros de ligne dans l'inspecteur de code en mode Code.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` pour afficher les numéros de ligne et `false` pour les masquer.

Valeurs renvoyées

Aucune.

dreamweaver.htmlInspector.setShowSyntaxColoring()

Disponibilité

Dreamweaver 4.

Description

Cette fonction active ou désactive la coloration de la syntaxe dans l'inspecteur de code en mode Code.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` pour rendre la coloration de la syntaxe visible et `false` dans le cas contraire.

Valeurs renvoyées

Aucune.

dreamweaver.htmlInspector.setShowWordWrap()

Disponibilité

Dreamweaver 4.

Description

Cette fonction active ou désactive l'option Retour à la ligne dans l'inspecteur de code.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` active l'option Retour à la ligne, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

dreamweaver.setHideAllFloaters()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active l'option Masquer les panneaux ou Afficher les panneaux.

Arguments

bShowFloatingPalettes

- L'argument *bShowFloatingPalettes* doit avoir une valeur booléenne : `true` active l'option Masquer les panneaux, tandis que `false` active l'option Afficher les panneaux.

Valeurs renvoyées

Aucune.

dreamweaver.setShowStatusBar()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Affichage > Barre d'état.

Arguments

bShowStatusBar

- L'argument *bShowStatusBar* doit avoir une valeur booléenne : `true` active l'option Barre d'état, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

site.getShowDependents()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Afficher les fichiers dépendants.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les fichiers dépendants sont visibles dans la carte du site et `false` dans le cas contraire.

site.getShowHiddenFiles()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Afficher les fichiers identifiés comme masqués.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les fichiers masqués sont visibles dans la carte du site et `false` dans le cas contraire.

site.getShowPageTitles()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Afficher les titres des pages.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les titres de page sont visibles dans la carte du site et `false` dans le cas contraire.

site.getShowToolTips()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient l'état en cours de l'option Info-bulles.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` lorsque les info-bulles sont visibles dans le panneau Site et `false` dans le cas contraire.

site.setShowDependents()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Afficher les fichiers dépendants dans la carte du site.

Arguments

`bShowDependentFiles`

- L'argument *`bShowDependentFiles`* doit avoir une valeur booléenne : `true` active l'option Afficher les fichiers dépendants, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

site.setShowHiddenFiles()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Afficher les fichiers identifiés comme masqués dans la carte du site.

Arguments

`bShowHiddenFiles`

- L'argument *bShowHiddenFiles* doit avoir une valeur booléenne : `true` active l'option Afficher les fichiers identifiés comme masqués, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

site.setShowPageTitles()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Afficher les titres des pages dans la carte du site.

Arguments

bShowPageTitles

- L'argument *bShowPageTitles* doit avoir une valeur booléenne : `true` active l'option Afficher les titres des pages, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canShowPageTitles()*, page 614.

site.setShowToolTips()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active ou désactive l'option Info-bulles.

Arguments

bShowToolTips

- L'argument *bShowToolTips* doit avoir une valeur booléenne : `true` active l'option Info-bulles, tandis que `false` la désactive.

Valeurs renvoyées

Aucune.

Fonctions relatives aux barres d'outils

Les fonctions JavaScript suivantes vous permettent d'obtenir et de définir l'affichage des barres d'outils et de leurs étiquettes, de vous procurer les étiquettes des éléments de barre d'outils dans la fenêtre en cours, de positionner les barres d'outils et d'obtenir leur ID. Pour plus d'informations sur la création ou la modification des barres d'outils, voir la rubrique « Barres d'outils » dans l'aide Extension de Dreamweaver.

dom.forceToolbarUpdate()

Disponibilité

Dreamweaver MX 2004.

Description

Oblige les gestionnaires de mise à jour à s'exécuter sur tous les éléments de la barre d'outils spécifiée.

Arguments

`toolbarID`

- L'argument *`toolbarID`* correspond à l'ID de la barre d'outils dont Dreamweaver doit mettre à jour les éléments.

Valeurs renvoyées

Aucune.

dom.getShowToolbarIconLabels()

Disponibilité

Dreamweaver MX.

Description

Cette fonction détermine si les étiquettes des boutons sont visibles dans la fenêtre de document active. Dreamweaver affiche toujours les étiquettes des commandes autres que les boutons si ces étiquettes sont définies.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les étiquettes des boutons sont visibles dans la fenêtre de document active et `false` dans le cas contraire.

Exemple

L'exemple suivant permet d'afficher les étiquettes des boutons :

```
var dom = dw.getDocumentDom();
if (dom.getShowToolbarIconLabels() == false)
{
  dom.setShowToolbarIconLabels(true);
}
```

dom.getToolbarIdArray()

Disponibilité

Dreamweaver MX.

Description

Cette fonction renvoie un tableau contenant les ID de toutes les barres d'outils de l'application. Utilisez la fonction `dom.getToolbarIdArray()` pour désactiver toutes les barres d'outils et pouvoir ainsi les repositionner en ne rendant visibles que certaines d'entre elles.

Arguments

Aucun.

Valeurs renvoyées

Tableau de tous les ID de barre d'outils.

Exemple

L'exemple suivant stocke le tableau des ID de barre d'outils dans la variable `tb_ids` :

```
var tb_ids = new Array();
tb_ids = dom.getToolbarIdArray();
```

dom.getToolbarItemValue()

Disponibilité

Dreamweaver MX 2004.

Description

Obtient la valeur de l'élément de barre d'outils spécifié.

Arguments

toolbarID, itemID

- L'argument *toolbarID* est une chaîne indiquant l'ID de la barre d'outils contenant l'élément pour lequel vous souhaitez obtenir une valeur.
- L'argument *itemID* est une chaîne qui indique l'ID de l'élément pour lequel vous souhaitez obtenir une valeur.

Valeurs renvoyées

Une chaîne représentant la valeur de l'élément de barre d'outils.

Exemple

L'exemple de fonction `receiveArguments()` suivant figure dans une commande de barre d'outils qui régit le comportement d'un champ de texte Size ; il obtient la valeur du champ Size sous la forme d'un argument, puis lit la valeur du champ Units afin de générer une valeur valide pour la fonction de propriété CSS `font-size` :

```
receiveArguments(newSize){
var dom = dw.getDocumentDOM();
if (newSize != ""){
  dom.applyFontMarkupAsStyle('font-size', newSize +
  dom.getToolbarItemValue("DW_Toolbar_Text","DW_Text_Units"));
  }
else{
  dom.removeFontMarkupAsStyle('font-size');
  }
}
```

dom.getToolbarLabel()

Disponibilité

Dreamweaver MX.

Description

Cette fonction obtient l'étiquette de la barre d'outils spécifiée. Utilisez la fonction `dom.getToolbarLabel()` pour les menus qui affichent ou masquent des barres d'outils.

Arguments

`toolbar_id`

- L'argument *`toolbar_id`* est l'ID de la barre d'outils, qui correspond à la valeur de l'attribut ID de la balise toolbar définie dans le fichier toolbars.xml.

Valeurs renvoyées

La chaîne de nom *`label`* qui est affectée à la balise `toolbar` comme attribut.

Exemple

L'exemple suivant stocke l'étiquette de `myEditbar` dans la variable `label` :

```
var label = dom.getToolbarLabel("myEditbar");
```

dom.getToolbarVisibility()

Disponibilité

Dreamweaver MX.

Description

Cette fonction renvoie une valeur booléenne qui indique si la barre d'outils spécifiée dans l'argument *`toolbar_id`* est visible.

Arguments

`toolbar_id`

- L'argument *`toolbar_id`* est la chaîne d'ID affectée à la barre d'outils.

Valeurs renvoyées

Valeur booléenne : `true` si la barre d'outils est visible et `false` si la barre d'outils n'est pas visible ou n'existe pas.

Exemple

L'exemple suivant vérifie si la barre d'outils `myEditbar` est visible dans la fenêtre de document, puis stocke cette valeur dans la variable `retval` :

```
var retval = dom.getToolbarVisibility("myEditbar");
return retval;
```

dom.setToolbarItemAttribute()

Disponibilité

Dreamweaver MX 2004.

Description

Modifie une valeur parmi les trois attributs image ou l'attribut tooltip d'un élément de la barre d'outils.

Arguments

toolbarID, *toolbarItemId*, *attrName*, *attrValue*

- L'argument *toolbarID* est une chaîne qui indique l'ID de la barre d'outils.
- L'argument *toolbarItemId* est une chaîne qui indique l'ID de l'élément de la barre d'outils.
- L'argument *attrName* est une chaîne qui indique le nom de l'attribut à définir. Les valeurs autorisées sont 'image', 'overImage', 'disabledImage' et 'tooltip'.
- L'argument *attrValue* est une chaîne qui indique la valeur à définir.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant appelle dom.setToolbarItemAttribute() trois fois pour définir les attributs image, imageOver et tooltip de l'élément MyButton sur la barre d'outils portant l'ID DW_Toolbar_Main :

```
var dom = dw.getDocumentDOM();
dom.setToolbarItemAttribute('DW_Toolbar_Main', 'MyButton', 'image',
  'Toolbars/imgs/newimage.gif');
dom.setToolbarItemAttribute('DW_Toolbar_Main', 'MyButton', 'imageOver',
  'Toolbars/imgs/newimageOver.gif');
dom.setToolbarItemAttribute('DW_Toolbar_Main', 'MyButton', 'tooltip', 'One
  fine button');
```

dom.setShowToolbarIconLabels()

Disponibilité

Dreamweaver MX.

Description

Cette fonction indique à Dreamweaver d'afficher les étiquettes des boutons comportant des étiquettes. Dreamweaver affiche toujours les étiquettes des commandes autres que les boutons, si ces étiquettes sont définies.

Arguments

bShow

- L'argument *bShow* doit avoir une valeur booléenne : `true` pour afficher les étiquettes des boutons et `false` dans le cas contraire.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant indique à Dreamweaver d'afficher les étiquettes des boutons sur les barres d'outils :

```
dom.setShowToolbarIconLabels(true);
```

dom.setToolbarPosition()

Disponibilité

Dreamweaver MX.

Description

Cette fonction déplace la barre d'outils spécifiée à la position indiquée.

REMARQUE	Il est impossible de déterminer la position actuelle d'une barre d'outils.

Arguments

toobar_id, position, relative_to

- L'argument *toolbar_id* est l'ID de la barre d'outils, qui correspond à la valeur de l'attribut ID de la balise toolbar définie dans le fichier toolbars.xml.

- L'argument *position* indique l'endroit où Dreamweaver place la barre d'outils par rapport aux autres barres d'outils. Les valeurs possibles de l'argument *position* sont décrites dans la liste suivante :
  - top correspond à la position par défaut. La barre d'outils apparaît en haut de la fenêtre de document.
  - below affiche la barre d'outils au début de la ligne qui se trouve juste en dessous de la barre d'outils spécifiée dans l'argument *relative_to*. Dreamweaver indique une erreur si la barre d'outils ne trouve pas la barre d'outils spécifiée dans l'argument *relative_to*.
  - floating affiche la barre d'outils au-dessus du document, en mode flottant. Dreamweaver place automatiquement la barre d'outils de sorte qu'elle soit décalée par rapport aux autres barres d'outils flottantes. Sur Macintosh, floating est traité de la même façon que top.
- *relative_to*="*toolbar_id*" est obligatoire lorsque l'argument *position* est défini sur la valeur below. Dans les autres cas, cet argument est ignoré. Indique l'ID de la barre d'outils en dessous de laquelle cette barre d'outils doit être placée.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant place myEditbar en dessous de la barre d'outils myPicturebar :

```
dom.setToolbarPosition("myEditbar", "below", "myPicturebar");
```

dom.setToolbarVisibility()

Disponibilité

Dreamweaver MX.

Description

Cette fonction affiche ou masque la barre d'outils spécifiée.

Arguments

toolbar_id, bShow

- L'argument *toolbar_id* est l'ID de la barre d'outils, qui correspond à la valeur de l'attribut ID de la balise toolbar définie dans le fichier toolbars.xml.

- L'argument *bShow* est une valeur booléenne qui indique si la barre d'outils doit être affichée ou masquée. Si *bShow* a pour valeur `true`, la fonction `dom.setToolbarVisibility()` rend la barre d'outils visible. Si *bShow* a pour valeur `false`, la fonction `dom.setToolbarVisibility()` rend la barre d'outils invisible.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant vérifie si la barre d'outils myEditbar est visible dans la fenêtre de document. Si ce n'est pas le cas, il l'affiche :

```
var dom = dw.getDocumentDOM();
if(dom != null && dom.getToolbarVisibility("myEditbar") == false)
{
  dom.setToolbarVisibility("myEditbar", true);
{
```

Fonctions relatives aux fenêtres

Ces fonctions permettent d'agir sur la fenêtre de document et sur les panneaux flottants. Elles permettent d'afficher et de masquer les panneaux flottants, de déterminer la partie active de la fenêtre de document et de définir le document actif. Pour les opérations relatives au panneau Site, voir *Fonctions relatives aux sites*, page 264.

REMARQUE	Certaines fonctions de cette section fonctionnent uniquement sous Windows. Leur description indique si tel est le cas.

dom.getFocus()

Disponibilité

Dreamweaver 3.

Description

Cette fonction détermine quelle partie du document est actuellement active.

Arguments

Aucun.

Valeurs renvoyées

L'une des chaînes suivantes :

- la chaîne `"head"` si la zone `HEAD` est active ;
- la chaîne `"body"` si la zone `BODY` ou `NOFRAMES` est active ;
- la chaîne `"frameset"` si un jeu de cadres ou l'un des cadres qui le composent est sélectionné ;
- la chaîne `"none"` si la zone active ne figure pas dans le document (si elle se trouve dans l'inspecteur de propriétés ou dans un autre panneau flottant, par exemple).

dom.getView()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine quel mode est visible.

Arguments

Aucun.

Valeurs renvoyées

`"design"`, `"code"` ou `"split"`, selon le mode d'affichage visible.

dom.getWindowTitle()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient le titre de la fenêtre contenant le document.

Arguments

Aucun.

Valeurs renvoyées

Chaîne contenant le texte qui apparaît entre les balises `TITLE` du document, ou rien si le document ne figure pas dans une fenêtre ouverte.

dom.setView()

Disponibilité

Dreamweaver 4.

Description

Cette fonction affiche ou masque le mode Création ou Code pour produire un affichage contenant uniquement le mode Création ou Code ou les deux.

Arguments

viewString

- L'argument *viewString* correspond au mode à afficher ; il doit avoir l'une des valeurs suivantes : `"design"`, `"code"` ou `"split"`.

Valeurs renvoyées

Aucune.

dreamweaver.bringAttentionToFloater()

Disponibilité

Dreamweaver MX.

Description

Place au premier plan le panneau ou l'inspecteur spécifié et attire l'attention sur cet élément en le faisant clignoter (fonctionnalité légèrement différente de `dw.toggleFloater()`).

Arguments

floaterName

- L'argument *floaterName* correspond au nom de la fenêtre, du panneau ou de l'inspecteur.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant ouvre le panneau Actifs et le fait clignoter :

```
dw.bringAttentionToFloater("library");
```

dreamweaver.cascade()

Disponibilité

Dreamweaver MX (Windows uniquement), Dreamweaver 8 (prise en charge supplémentaire pour Macintosh).

Description

Affiche en cascade les fenêtres de document en commençant dans l'angle supérieur gauche et en décalant légèrement chaque fenêtre par rapport à la précédente.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant affiche les documents ouverts en cascade :

```
dw.cascade()
```

dreamweaver.getActiveWindow()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient le document figurant dans la fenêtre active.

Arguments

Aucun.

Valeurs renvoyées

Objet document correspondant au document qui figure dans la fenêtre active ou, si le document figure dans un cadre, objet document correspondant au jeu de cadres.

dreamweaver.getDocumentList()

Disponibilité

Dreamweaver 3.

Description

Cette fonction renvoie la liste de tous les documents ouverts.

Arguments

Aucun.

Valeurs renvoyées

Tableau d'objets document, chacun correspondant à une fenêtre de document ouverte. Si une fenêtre de document contient un jeu de cadres, l'objet document désigne le jeu de cadres et non le contenu des cadres.

dreamweaver.getFloaterVisibility()

Disponibilité

Dreamweaver 3.

Description

Cette fonction vérifie si le panneau ou l'inspecteur spécifié est visible.

Arguments

floaterName

- L'argument *floaterName* est le nom du panneau flottant. Si *floaterName* ne correspond à aucun nom de panneau intégré, Dreamweaver recherche dans le dossier Configuration/Floaters un fichier appelé *floaterName*.htm dans lequel *floaterName* correspond au nom d'un panneau flottant.

  Les valeurs de *floaterName* pour les panneaux Dreamweaver intégrés sont les chaînes se trouvant à droite des noms de panneaux dans la liste suivante :

  Actifs = "assets"

  Comportements = "behaviors"

  Liaisons = "data bindings"

  Inspecteur de code = "html"

  Composants = "server components"

  Styles CSS = "css styles"

  Cadres = "frames"

  Historique = "history"

  Barre Insérer = "objects"

Calques = "layers"

Bibliothèque = "library"

Vérificateur de lien (Résultats) = "linkchecker"

Propriétés = "properties"

Référence = "reference"

Rapports (Résultats) = "reports"

Recherche (Résultats) = "search"

Inspecteur de sélections = "selection inspector"

Comportements de serveur = "server behaviors"

Site = "site"

Fichiers du site = "site files"

Carte du site - “site map”

Fragments de code = "snippets"

Vérification du navigateur cible (Résultats) = "btc"

Validation (Résultats) = "validation"

Valeurs renvoyées

Valeur booléenne : `true` si le panneau flottant est visible et se trouve au premier plan, `false` si ce n'est pas le cas ou que Dreamweaver ne trouve pas de panneau flottant nommé *`floaterName`*.

dreamweaver.getFocus()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine la partie de l'application actuellement active.

Arguments

`bAllowFloaters`

- L'argument `bAllowFloaters` doit avoir une valeur booléenne : `true` si la fonction doit renvoyer le nom du panneau flottant (si un panneau flottant est actif) et `false` dans le cas contraire.

Valeurs renvoyées

L'une des chaînes suivantes :

- la chaîne `"document"` si la fenêtre de document est active ;
- la chaîne `"site"` si le panneau Site est actif ;
- la chaîne `"textView"` si le mode Texte est actif ;
- la chaîne `"html"` si l'inspecteur de code est actif ;
- la chaîne `floaterName` si `bAllowFloaters` a pour valeur `true` et si un panneau flottant est actif, `floaterName` correspondant à `"objects"`, `"properties"`, `"launcher"`, `"library"`, `"css styles"`, `"html styles"`, `"behaviors"`, `"timelines"`, `"layers"`, `"frames"`, `"templates"` ou `"history"` ;
- la chaîne `"none"` (Macintosh) si aucune fenêtre de document ni le panneau Site ne sont ouverts.

dreamweaver.getPrimaryView()

Disponibilité

Dreamweaver 4.

Description

Cette fonction détermine le mode d'affichage défini comme mode principal (au premier plan).

Arguments

Aucun.

Valeurs renvoyées

Les chaînes `"design"` ou `"code"`, selon le mode visible ou le volet situé au premier plan dans un affichage à deux volets.

dreamweaver.getSnapDistance()

Disponibilité

Dreamweaver 4.

Description

Cette fonction renvoie la distance d'alignement en pixels.

Arguments

Aucun.

Valeurs renvoyées

Nombre entier représentant la distance d'alignement en pixels. La distance par défaut est 10 pixels ; 0 indique que la fonction d'alignement est désactivée.

dreamweaver.minimizeRestoreAll()

Disponibilité

Dreamweaver 4.

Description

Cette fonction réduit en icône ou restaure toutes les fenêtres dans Dreamweaver.

Arguments

`bMinimize`

- L'argument *`bMinimize`* doit avoir une valeur booléenne : `true` pour réduire les fenêtres en icônes et `false` pour les restaurer.

Valeurs renvoyées

Aucune.

dreamweaver.setActiveWindow()

Disponibilité

Dreamweaver 3.

Description

Cette fonction active la fenêtre contenant le document spécifié.

Arguments

`documentObject, {bActivateFrame}`

- L'argument *`documentObject`* correspond à l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`).

- Facultatif, l'argument *bActivateFrame* n'est applicable que lorsque *documentObject* se trouve à l'intérieur d'un jeu de cadres. L'argument *bActivateFrame* est une valeur booléenne : `true` active le cadre contenant le document ainsi que la fenêtre contenant le jeu de cadres et `false` les désactive.

Valeurs renvoyées

Aucune.

dreamweaver.setFloaterVisibility()

Disponibilité

Dreamweaver 3.

Description

Cette fonction indique s'il faut rendre visible un panneau flottant ou un inspecteur spécifique.

Arguments

floaterName, bIsVisible

- L'argument *floaterName* est le nom du panneau flottant. Si *floaterName* ne correspond à aucun nom de panneau intégré, Dreamweaver recherche dans le dossier Configuration/Floaters un fichier appelé *floaterName*.htm dans lequel *floaterName* correspond au nom d'un panneau flottant. Si Dreamweaver ne trouve aucun panneau flottant portant le nom *floaterName*, cette fonction n'a aucun effet.

  Les valeurs de *floaterName* pour les panneaux Dreamweaver intégrés sont les chaînes se trouvant à droite des noms de panneaux dans la liste suivante :

  Actifs = "assets"

  Comportements = "behaviors"

  Liaisons = "data sources"

  Inspecteur de code = "html"

  Composants = "server components"

  Styles CSS = "css styles"

  Cadres = "frames"

  Historique = "history"

  Styles HTML = "html styles"

  Barre Insérer = "objects"

Calques = "`layers`"

Bibliothèque = "`library`"

Vérificateur de lien (Résultats) = "`linkchecker`"

Propriétés = "`properties`"

Référence = "`reference`"

Rapports (Résultats) = "`reports`"

Recherche (Résultats) = "`search`"

Comportements de serveur = "`server behaviors`"

Site = "`site`"

Fichiers du site = "`site files`"

Carte du site = "`site map`"

Fragments de code = "`snippets`"

Inspecteur de balises = "`tag inspector`"

Vérification du navigateur cible (Résultats) = "`btc`"

Modèles = "`templates`"

Validation (Résultats) = "`validation`"

L'argument *`bIsVisible`* est une valeur booléenne indiquant si le panneau flottant doit être visible.

Valeurs renvoyées

Aucune.

dreamweaver.setPrimaryView()

Disponibilité

Dreamweaver 4.

Description

Cette fonction affiche le mode spécifié en haut de la fenêtre de document.

Arguments

`viewString`

- L'argument *`viewString`* correspond au mode à afficher en haut de la fenêtre de document ; il peut avoir l'une des valeurs suivantes : "`design`" ou "`code`".

Valeurs renvoyées

Aucune.

dreamweaver.setSnapDistance()

Disponibilité

Dreamweaver 4.

Description

Cette fonction définit la distance d'alignement en pixels.

Arguments

snapDistance

- L'argument *snapDistance* est un nombre entier représentant la distance d'alignement en pixels. La valeur par défaut est 10 pixels. Saisissez 0 pour désactiver la fonction d'alignement.

Valeurs renvoyées

Aucune.

dreamweaver.showProperties()

Disponibilité

Dreamweaver 3.

Description

Cette fonction rend l'inspecteur de propriétés visible et l'active.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.tileHorizontally()

Disponibilité

Dreamweaver MX (Windows uniquement), Dreamweaver 8 (prise en charge supplémentaire pour Macintosh).

Description

Organise les fenêtres de document en mosaïque horizontale en les plaçant les unes à côté des autres sans qu'elles ne se chevauchent. Ce processus revient au partage vertical de l'espace de travail.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant affiche les documents ouverts en mosaïque horizontale :

```
dw.tileHorizontally()
```

dreamweaver.tileVertically()

Disponibilité

Dreamweaver MX (Windows uniquement), Dreamweaver 8 (prise en charge supplémentaire pour Macintosh).

Description

Organise les fenêtres de documents en mosaïque verticale en les plaçant les unes derrière les autres sans qu'elles ne se chevauchent. Ce processus revient au partage horizontal de l'espace de travail.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant affiche les documents ouverts en mosaïque verticale :

```
dw.tileVertically()
```

dreamweaver.toggleFloater()

Disponibilité

Dreamweaver 3.

Description

Cette fonction affiche, masque ou place au premier plan le panneau ou l'inspecteur spécifié.

REMARQUE

Cette fonction n'a de sens que dans le fichier menus.xml. Pour afficher, placer au premier plan ou masquer un panneau flottant, utilisez la fonction `dw.setFloaterVisibility()`.

Arguments

`floaterName`

- L'argument *`floaterName`* correspond au nom de la fenêtre. Si le nom de la fenêtre flottante est `reference`, l'état de visibilité du panneau Référence peut être mis à jour par l'utilisateur dans le mode Code. Tous les autres panneaux effectuent un suivi continu de la sélection, mais le panneau Référence n'analyse la sélection dans le mode Code que lorsque l'utilisateur appelle un suivi.

Valeurs renvoyées

Aucune.

dreamweaver.updateReference()

Disponibilité

Dreamweaver 4.

Description

Cette fonction met à jour le panneau flottant Référence. Si le panneau flottant Référence n'est pas visible, `dw.updateReference()` l'affiche et le met à jour.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Fonctions relatives au fractionnement des codes

Ces fonctions vous permettent de fractionner et d'étendre visuellement des codes. Vous pouvez fractionner ou étendre des sélections arbitraires de codes, ou des fragments situés entre les balises d'ouverture et de fermeture. Bien que les fonctions de fractionnement de code existent dans le dom comme dans htmlInspector, les plages réduites sont identiques dans le mode Code et dans Cold Inspector.

dom.collapseFullTag()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si l'élément sélectionné se trouve dans une seule paire de balises d'ouverture et de fermeture ou s'il contient une seule paire de balises d'ouverture et de fermeture en mode Code. Le cas échéant, la fonction fractionne le fragment de code qui commence immédiatement avant la balise d'ouverture et se termine après la balise de fermeture ; dans le cas contraire, elle ne fait rien.

Arguments

`allowCodeFragmentAdjustment`

- L'argument obligatoire *`allowCodeFragmentAdjustment`* est une valeur booléenne. Si défini sur `true`, cet argument n'a aucun effet actuellement ou bien le même effet que la valeur `false`. Si défini sur `false`, Dreamweaver fractionne le code qui commence immédiatement avant la balise d'ouverture et se termine immédiatement après la balise de fermeture, sans le modifier

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant fractionne le fragment de code qui commence juste avant la balise d'ouverture et se termine juste après la balise de fermeture dans la sélection en cours du mode Code :

```
var currentDOM = dw.getDocumentDOM();
currentDOM.collapseFullTag(false);
```

dom.collapseFullTagInverse()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si l'élément sélectionné se trouve dans une seule paire de balises d'ouverture et de fermeture ou s'il contient une seule paire de balises d'ouverture et de fermeture en mode Code. Le cas échéant, elle fractionne le code qui précède la balise d'ouverture et le code qui suit la balise de fermeture ; dans le cas contraire, elle ne fait rien.

Arguments

allowAdjustmentOfCodeFragments

- L'argument obligatoire *allowAdjustmentOfCodeFragments* est une valeur booléenne. Si défini sur `true`, Dreamweaver ajuste les limites du code situé *avant* la balise de d'ouverture et du code situé *après* la balise de fermeture pour effectuer un *fractionnement intelligent* qui conserve la mise en retrait et l'espacement en cours. Si défini sur `false`, Dreamweaver fractionne les fragments de code qui se trouvent *avant* la balise d'ouverture et *après* la balise de fermeture, exactement comme indiqué par la sélection.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant ajuste les limites du code avant la balise d'ouverture et après la balise de fermeture pour effectuer un *fractionnement intelligent* qui conserve la mise en retrait et l'espacement :

```
var currentDOM = dw.getDocumentDOM();
currentDOM.collapseFullTagInverse(true);
```

dom.collapseSelectedCodeFragment()

Disponibilité

Dreamweaver 8.

Description

Fractionne le code sélectionné en mode Code. Si la sélection est déjà réduite, cette fonction ne fait rien.

Arguments

allowCodeFragmentAdjustment

- L'argument obligatoire *allowCodeFragmentAdjustment* est une valeur booléenne. Si défini sur `true`, Dreamweaver modifie les limites de la sélection en cours pour effectuer un *fractionnement intelligent* et conserver la mise en retrait et l'espacement en cours. Si défini sur `false`, Dreamweaver fractionne le fragment de code actuellement sélectionné, exactement comme indiqué par la sélection.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant fractionne le fragment de code sélectionné en mode Code sans le modifier.

```
var currentDOM = dw.getDocumentDOM();
currentDOM.collapseSelectedCodeFragment(false);
```

dom.collapseSelectedCodeFragmentInverse()

Disponibilité

Dreamweaver 8.

Description

Fractionne toutes les portions de code *avant* et *après* le code sélectionné en mode Code.

Arguments

allowAdjustmentOfCodeFragments

- L'argument obligatoire *allowAdjustmentOfCodeFragments* est une valeur booléenne. Si défini sur `true`, Dreamweaver ajuste les limites du code situé *avant* et *après* la sélection en cours pour effectuer un *fractionnement intelligent* qui conserve la mise en retrait et l'espacement en cours. Si défini sur `false`, Dreamweaver fractionne le fragment de code exactement comme indiqué par la sélection.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant modifie puis fractionne toutes les portions de code situées avant et après le code sélectionné en mode Code :

```
var currentDOM = dw.getDocumentDOM();
currentDOM.collapseSelectedCodeFragmentInverse(true);
```

dom.expandAllCodeFragments()

Disponibilité

Dreamweaver 8.

Description

Etend tous les fragments de code réduits en mode Code, y compris les fragments de code réduits imbriqués.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant étend toutes les portions de code réduit en mode Code.

```
var currentDOM = dw.getDocumentDOM();
currentDOM.expandAllCodeFragments();
```

dom.expandSelectedCodeFragments()

Disponibilité

Dreamweaver 8.

Description

Etend tous les fragments de code réduits dans la sélection en cours en mode Code. Si la sélection est déjà étendue, cette fonction ne fait rien.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant étend toutes les portions de code réduit dans la sélection en cours en mode Code.

```
var currentDOM = dw.getDocumentDOM();
currentDOM.expandSelectedCodeFragments();
```

dreamweaver.htmlInspector.collapseFullTag()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si l'élément sélectionné se trouve dans une seule paire de balises d'ouverture et de fermeture ou s'il contient une seule paire de balises d'ouverture et de fermeture dans l'inspecteur de code. Le cas échéant, la fonction fractionne le fragment de code qui commence immédiatement avant la balise d'ouverture et se termine après la balise de fermeture ; dans le cas contraire, elle ne fait rien.

Arguments

`allowACodeFragmentAdjustment`

- L'argument obligatoire *`allowCodeFragmentAdjustment`* est une valeur booléenne. Si défini sur `true`, cet argument n'a aucun effet actuellement ou bien le même effet que la valeur `false`. Si défini sur `false`, Dreamweaver fractionne le code qui commence immédiatement avant la balise d'ouverture et se termine immédiatement après la balise de fermeture, sans le modifier.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant fractionne le fragment de code qui commence juste avant la balise d'ouverture et se termine juste après la balise de fermeture dans la sélection en cours de l'inspecteur de code :

```
dreamweaver.htmlInspector.collapseFullTag(false);
```

dreamweaver.htmlInspector.collapseFullTagInverse()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si l'élément sélectionné se trouve dans une seule paire de balises d'ouverture et de fermeture ou s'il contient une seule paire de balises d'ouverture et de fermeture dans l'inspecteur de code. Le cas échéant, elle fractionne le code situé *avant* la balise d'ouverture et le code situé *après* la balise de fermeture ; dans le cas contraire, elle ne fait rien.

Arguments

allowAdjustmentOfCodeFragments

- L'argument obligatoire *allowAdjustmentOfCodeFragments* est une valeur booléenne. Si défini sur `true`, Dreamweaver ajuste les limites du code situé avant la balise de d'ouverture et du code situé après la balise de fermeture pour effectuer un *fractionnement intelligent* qui conserve la mise en retrait et l'espacement existants. Si défini sur `false`, Dreamweaver fractionne le code situé *avant* la balise d'ouverture et le code situé *après* la balise de fermeture, sans le modifier.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant effectue un *fractionnement intelligent* sur les sections de code qui apparaissent avant la balise d'ouverture et après la balise de fermeture de la sélection en cours :

```
dreamweaver.htmlInspector.collapseFullTagInverse(true);
```

dreamweaver.htmlInspector.collapseSelectedCodeFragment()

Disponibilité

Dreamweaver 8.

Description

Fractionne le code sélectionné dans l'inspecteur de code. Si la sélection est déjà réduite, cette fonction ne fait rien.

Arguments

allowCodeFragmentAdjustment

- L'argument obligatoire *allowCodeFragmentAdjustment* est une valeur booléenne. Si défini sur `true`, Dreamweaver modifie la sélection en cours pour effectuer un *fractionnement intelligent* et conserver la mise en retrait et l'espacement existants. Si défini sur `false`, Dreamweaver fractionne le fragment de code actuellement sélectionné, exactement comme indiqué par la sélection.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant modifie puis fractionne le code sélectionné dans l'inspecteur de code.

```
dreamweaver.htmlInspector.collapseSelectedCodeFragment(true);
```

dreamweaver.htmlInspector.collapseSelectedCodeFragmentInverse()

Disponibilité

Dreamweaver 8.

Description

Cette fonction fractionne toutes les portions de code *avant* et *après* le code sélectionné dans l'inspecteur de code. Si la sélection est déjà réduite, cette fonction ne fait rien.

Arguments

`allowAdjustmentOfCodeFragments`

- L'argument obligatoire *`allowAdjustmentOfCodeFragments`* est une valeur booléenne. Si défini sur `true`, Dreamweaver ajuste les limites des sections de code situé avant et après la sélection en cours pour effectuer un *fractionnement intelligent* qui conserve la mise en retrait et l'espacement en cours. Si défini sur `false`, Dreamweaver fractionne les sections de code exactement comme indiqué par la sélection.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant fractionne toutes les portions de code avant et après le code sélectionné dans l'inspecteur de code, exactement comme indiqué par la sélection.

```
dreamweaver.htmlInspector.collapseSelectedCodeFragmentInverse(false);
```

dreamweaver.htmlInspector.expandAllCodeFragments()

Disponibilité

Dreamweaver 8.

Description

Etend tous les fragments de code réduits dans l'inspecteur de code, y compris les fragments de code réduits imbriqués.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant étend toutes les portions de code réduit dans l'inspecteur de code.

```
dreamweaver.htmlInspector.expandAllCodeFragments();
```

dreamweaver.htmlInspector.expandSelectedCodeFragments()

Disponibilité

Dreamweaver 8.

Description

Etend tous les fragments de code réduits dans la sélection en cours de l'inspecteur de code. Si la sélection est déjà étendue, cette fonction ne fait rien.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant étend toutes les portions de code réduit dans la sélection en cours de l'inspecteur de code.

```
dreamweaver.htmlInspector.expandSelectedCodeFragments();
```

Fonctions relatives aux barres d'outils du mode Code

Ces fonctions permettent d'insérer du texte, de supprimer des commentaires, d'afficher ou de masquer les caractères spéciaux utilisés pour les espaces blancs en mode Code et d'obtenir le chemin d'accès du document actif.

REMARQUE	Deux barres d'outils Codage différentes sont proposées : une pour le mode Code et l'autre pour l'inspecteur de code. Les deux sont personnalisées dans le ficher Configuration/Toolbars/toolbars.xml.

dom.getOpenPathName()

Disponibilité

Dreamweaver 8.

Description

Cette fonction obtient le chemin de fichier absolu du document ouvert.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui correspond au chemin de fichier absolu du document ouvert.

Exemple

L'exemple suivant assigne la chaîne contenant le chemin d'accès du document actuellement ouvert à la variable `fileName` :

```
var fileName = dom.getOpenPathName();
```

dom.getShowHiddenCharacters()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si les caractères spéciaux utilisés pour les espaces blancs sont affichés dans la fenêtre de document en mode Code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les caractères masqués sont affichés ; `false` dans le cas contraire.

Exemple

L'exemple suivant désactive l'affichage des caractères spéciaux utilisés pour les espaces blancs lorsque l'affichage de ces caractères est activé :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.getShowHiddenCharacters()){
  currentDOM.setShowHiddenCharacters(false);
  }
```

dom.setShowHiddenCharacters()

Disponibilité

Dreamweaver 8.

Description

Cette fonction affiche ou masque les caractères spéciaux utilisés pour les espaces blancs dans l'inspecteur de code en mode Code.

Arguments

show

- L'argument obligatoire *show* est une valeur booléenne qui indique s'il faut afficher les caractères masqués.

Valeurs renvoyées

Aucune.

Exemple

See *dom.getShowHiddenCharacters(), page 257*.

dom.source.applyComment()

Disponibilité

Dreamweaver 8.

Description

Insère le texte spécifié dans l'argument *beforeText* avant la sélection en cours et le texte spécifié dans l'argument *afterText* après la sélection en cours. La fonction étend ensuite la sélection en cours de manière à inclure le texte ajouté. Cependant, s'il n'y a pas de sélection, la fonction ne sélectionne aucun élément. Si la valeur du texte spécifié dans l'argument *afterText* est null, la fonction insère le texte spécifié dans l'argument *beforeText* au début de chaque ligne dans la sélection en cours.

Arguments

beforeText

- L'argument *beforeText* est obligatoire. Il indique le texte à insérer au début de la sélection, ou, si la valeur de l'argument *afterText* est null, il indique le texte à insérer au début de chaque ligne dans la sélection.

afterText

- Facultatif, l'argument *afterText* indique le texte à insérer à la fin de la sélection.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant transforme la sélection en cours en un commentaire HTML :

```
dw.getDocumentDOM().source.applyComment('<!--', '-->')
```

dom.source.removeComment()

Disponibilité

Dreamweaver 8.

Description

Cette fonction supprime des commentaires. Si vous n'indiquez aucun argument, elle supprime tous les types de commentaires de la sélection en cours, à l'exception des inclusions côté serveur et des commentaires spécifiques Dreamweaver. En cas de commentaires imbriqués, elle supprime uniquement le commentaire externe. S'il n'y a pas de sélection en cours, elle supprime uniquement le premier commentaire de la ligne sur laquelle se trouve le curseur. Si vous indiquez des arguments, la fonction supprime uniquement les commentaires correspondant aux valeurs spécifiées dans les arguments *beforeText* et *afterText*, même si les commentaires correspondants sont imbriqués dans d'autres types de commentaires.

Arguments

beforeText

- L'argument *beforeText* est facultatif. Il indique le texte permettant d'identifier le début du commentaire à supprimer dans la sélection, ou, si la valeur de l'argument *afterText* est null, il indique le type de commentaire de ligne à supprimer de la sélection en cours.

afterText

- Facultatif, l'argument *afterText* indique le texte permettant d'identifier la fin du commentaire à supprimer dans la sélection.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant supprime un commentaire HTML :

```
dw.getDocumentDOM().source.removeComment('<!--', '-->')
```

dreamweaver.htmlInspector.getShowHiddenCharacters()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si les caractères spéciaux utilisés pour les espaces blancs sont affichés dans l'inspecteur de code en mode Code.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les caractères masqués sont affichés ; `false` dans le cas contraire.

Exemple

L'exemple suivant désactive l'affichage des caractères spéciaux utilisés pour les espaces blancs dans l'inspecteur de code lorsque l'affichage de ces caractères est activé :

```
if (dreamweaver.htmlinspector.getShowHiddenCharacters()){
  dreamweaver.htmlinspector.setShowHiddenCharacters(false);
  }
```

dreamweaver.htmlInspector.setShowHiddenCharacters()

Disponibilité

Dreamweaver 8.

Description

Cette fonction affiche ou masque les caractères spéciaux utilisés pour les espaces blancs dans l'inspecteur de code en mode Code.

Arguments

show

- L'argument obligatoire *show* est une valeur booléenne qui indique s'il faut afficher les caractères masqués utilisés pour les espaces blancs.

Valeurs renvoyées

Valeur booléenne : `true` si les caractères masqués sont affichés ; `false` dans le cas contraire.

Exemple

Voir *dreamweaver.htmlInspector.getShowHiddenCharacters()*, page 260.

CHAPITRE 12

Site

12

Les fonctions relatives aux sites effectuent des tâches ayant trait à la gestion d'un site Web. Il peut s'agir de personnaliser un rapport, de définir un nouveau site, d'archiver ou d'extraire des fichiers, de valider un site, etc.

Fonctions relatives aux rapports

Ces fonctions permettent d'accéder aux fonctionnalités de création de rapports de Macromedia Dreamweaver 8, c'est-à-dire de lancer, de contrôler et de personnaliser le processus de création de rapports. Pour plus d'informations, voir la rubrique « Rapports » dans l'aide Extension de Dreamweaver.

dreamweaver.isReporting()

Disponibilité

Dreamweaver 4.

Description

Vérifie si un processus de création de rapport est en cours.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si un processus est en cours et `false` dans le cas contraire.

dreamweaver.showReportsDialog()

Disponibilité

Dreamweaver 4.

Description

Ouvre la boîte de dialogue Rapports.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Fonctions relatives aux sites

Ces fonctions permettent d'effectuer des opérations sur les fichiers de site ou sur la carte du site. Elles permettent d'effectuer les tâches suivantes :

- créer des liens entre fichiers ;
- obtenir, placer, archiver et extraire des fichiers ;
- sélectionner et désélectionner des fichiers ;
- créer et supprimer des fichiers ;
- obtenir des informations sur les sites définis par l'utilisateur ;
- importer et exporter les informations du site.

dom.getSiteURLPrefixFromDoc()

Disponibilité

Dreamweaver 8.

Description

Cette fonction obtient le préfixe de l'URL du site qui est extrait de l'adresse HTTP définie dans la section Infos locales de la boîte de dialogue Définition du site.

Arguments

Aucun.

Valeurs renvoyées

Chaîne indiquant le préfixe de l'URL du site.

Exemple

L'exemple suivant obtient le préfixe de l'URL du site pour le document actif :

```
var currentDOM = dw.getDocumentDOM();
var sitePrefix = dom.getSiteURLPrefixFromDoc();
```

dom.localPathToSiteRelative()

Disponibilité

Dreamweaver 8.

Description

Cette fonction convertit un chemin d'accès de fichier local en une référence d'URI relative au site.

Arguments

localFilePath

- L'argument obligatoire *localFilePath* est une chaîne contenant le chemin d'accès à un fichier local sur votre ordinateur local.

Valeurs renvoyées

Chaîne indiquant l'URI relative du site.

Exemple

L'exemple suivant

```
var siteRelativeURI =
  site.localPathToSiteRelative("C:\Inetpub\wwwroot\siteA\myFile.cfm")
```

renvoie `"/myWebApp/myFile.cfm"`, en fonction de vos mappages de site et de l'adresse HTTP spécifiée dans la section Infos locales de la boîte de dialogue Définition du site.

dom.siteRelativeToLocalPath()

Disponibilité

Dreamweaver 8.

Description

Cette fonction convertit une référence d'URI relative au site en un chemin d'accès de fichier local.

Arguments

siteRelativeURI

- L'argument obligatoire *siteRelativeURI* est une chaîne contenant l'URI relative du site.

Valeurs renvoyées

Chaîne indiquant le chemin d'accès à un fichier local sur votre ordinateur local.

Exemple

L'exemple suivant

```
var filePath = siteRelativeToLocalPath("/myWebApp/myFile.xml");
```

renvoie `"C:\Inetpub\wwwroot\siteA\myFile.xml"`, en fonction de vos mappages de site et de l'adresse HTTP spécifiée dans la section Infos locales de la boîte de dialogue Définition du site.

dreamweaver.compareFiles()

Disponibilité

Dreamweaver 8.

Description

Cette fonction lance l'outil de comparaison de fichiers installé par l'utilisateur dans la section Diff de la boîte de dialogue Préférences.

Arguments

file1, file2

- L'argument obligatoire *file1* est une chaîne contenant le chemin d'accès complet au premier fichier à comparer.
- L'argument obligatoire `file2` est une chaîne contenant le chemin d'accès complet au second fichier à comparer.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant compare deux fichiers, red.htm et blue.htm :

```
dw.compareFiles(hc:\data\red.htm", "e:\data\blue.htm");
```

dreamweaver.loadSitesFromPrefs()

Disponibilité

Dreamweaver 4.

Description

Charge les informations du site pour tous les sites de la base de registres du système (Windows) ou du fichier de préférences Dreamweaver (Macintosh) dans Dreamweaver. Si un site est connecté à un serveur distant lorsque cette fonction est appelée, il est automatiquement déconnecté.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.saveSitesToPrefs()

Disponibilité

Dreamweaver 4.

Description

Enregistre toutes les informations pour chaque site que l'utilisateur a défini dans la base de registres du système (Windows) ou le fichier de préférences Dreamweaver (Macintosh).

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.siteSyncDialog.compare()

Disponibilité

Dreamweaver 8.

Description

Cette fonction exécute l'application de comparaison de fichiers spécifiée dans la Catégorie de comparaison de fichiers de la boîte de dialogue Préférences pour comparer les fichiers sélectionnés sur les sites distants ou locaux.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.siteSyncDialog.canCompare()*, page 600.

dreamweaver.siteSyncDialog.markDelete()

Disponibilité

Dreamweaver 8.

Description

Cette fonction bascule l'action des éléments sélectionnés sur Supprimer dans la boîte de dialogue Synchronisation du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.siteSyncDialog.canMarkDelete()*, page 601.

dreamweaver.siteSyncDialog.markGet()

Disponibilité

Dreamweaver 8.

Description

Cette fonction bascule l'action des éléments sélectionnés sur Acquérir dans la boîte de dialogue Synchronisation du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.siteSyncDialog.canMarkGet()*, page 601.

dreamweaver.siteSyncDialog.markIgnore()

Disponibilité

Dreamweaver 8.

Description

Cette fonction bascule l'action des éléments sélectionnés sur Ignorer dans la boîte de dialogue Synchronisation du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.siteSyncDialog.canMarkIgnore()*, page 602.

dreamweaver.siteSyncDialog.markPut()

Disponibilité

Dreamweaver 8.

Description

Cette fonction bascule l'action des éléments sélectionnés sur Placer dans la boîte de dialogue Synchronisation du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.siteSyncDialog.canMarkPut()*, page 602.

dreamweaver.siteSyncDialog.markSynced()

Disponibilité

Dreamweaver 8.

Description

Cette fonction bascule l'action des éléments sélectionnés sur Synchroniser dans la boîte de dialogue Synchronisation du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.siteSyncDialog.canMarkSynced()*, page 603.

dreamweaver.siteSyncDialog.toggleShowAllFiles()

Disponibilité

Dreamweaver 8.

Description

Cette fonction permet d'afficher les fichiers que Dreamweaver considère comme identiques sur les sites distants et locaux dans la boîte de dialogue d'aperçu Synchroniser le site. Si la fonction est appelée alors que la case à cocher Afficher tous les fichiers est activée, elle la désactive ; inversement, si la case à cocher Afficher tous les fichiers est désactivée, elle l'active.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.addLinkToExistingFile()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Sélectionner fichier HTML pour permettre à l'utilisateur de sélectionner un fichier, puis crée un lien entre ce dernier et le document sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canAddLink()*, page 604.

site.addLinkToNewFile()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Lier au nouveau fichier pour permettre à l'utilisateur d'indiquer les détails du nouveau fichier, puis crée un lien entre ce dernier et le document sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canAddLink()*, page 604.

site.changeLinkSitewide()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Modifier le lien au niveau du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.changeLink()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Sélectionner fichier HTML pour permettre à l'utilisateur de sélectionner le nouveau fichier à associer au lien.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canChangeLink()*, page 605.

site.checkIn()

Disponibilité

Dreamweaver 3.

Description

Archive les fichiers sélectionnés et traite les fichiers dépendants de l'une des façons suivantes :

- Si l'utilisateur a activé l'option Invite lors de Placer/Archiver dans les préférences, catégorie FTP du site, la boîte de dialogue Fichiers dépendants s'affiche.
- Si l'utilisateur a activé l'option Ne plus afficher ce message dans la boîte de dialogue Fichiers dépendants, puis qu'il a cliqué sur Oui, les fichiers dépendants sont téléchargés et aucune boîte de dialogue ne s'affiche.
- Si l'utilisateur a activé l'option Ne plus afficher ce message dans la boîte de dialogue Fichiers dépendants, puis qu'il a cliqué sur Non, les fichiers dépendants ne sont pas téléchargés et aucune boîte de dialogue ne s'affiche.

Arguments

`siteOrURL`

- L'argument *`siteOrURL`* doit être soit le mot-clé `"site"`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canCheckIn()*, page 605.

site.checkLinks()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Vérificateur de lien et vérifie les liens dans les fichiers spécifiés.

Arguments

`scopeOfCheck`

- L'argument *`scopeOfCheck`* définit l'étendue de la vérification des liens. Il doit avoir pour valeur `"document"`, `"selection"` ou `"site"`.

Valeurs renvoyées

Aucune.

site.checkOut()

Disponibilité

Dreamweaver 3.

Description

Extrait les fichiers sélectionnés et traite les fichiers dépendants de l'une des façons suivantes :

- Si l'utilisateur a activé l'option Invite lors de Acquérir/Extraire dans les préférences, catégorie FTP du site, la boîte de dialogue Fichiers dépendants s'affiche.
- Si l'utilisateur a activé l'option Ne plus afficher ce message dans la boîte de dialogue Fichiers dépendants, puis qu'il a cliqué sur Oui, les fichiers dépendants sont téléchargés et aucune boîte de dialogue ne s'affiche.
- Si l'utilisateur a activé l'option Ne plus afficher ce message dans la boîte de dialogue Fichiers dépendants, puis qu'il a cliqué sur Non, les fichiers dépendants ne sont pas téléchargés et aucune boîte de dialogue ne s'affiche.

Arguments

`siteOrURL`

- L'argument *`siteOrURL`* doit être soit le mot-clé `"site"`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canCheckOut()*, page 606.

site.checkTargetBrowsers()

Disponibilité

Dreamweaver 3.

Description

Vérifie le navigateur cible des documents sélectionnés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.cloak()

Disponibilité

Dreamweaver MX.

Description

Voile la sélection en cours dans le panneau Site ou le dossier spécifié.

Arguments

siteOrURL

L'argument *siteOrURL* doit contenir l'une des deux valeurs suivantes :

- le mot-clé `"site"`, qui indique si `cloak()` doit envelopper la sélection dans le panneau Site ;
- l'URL d'un dossier particulier, qui indique si `cloak()` doit envelopper le dossier spécifié et tout son contenu.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canCloak()*, page 606.

site.compareFiles()

Disponibilité

Dreamweaver 8.

Description

Cette fonction lance l'application d'intégration d'outil Diff pour comparer deux fichiers.

Arguments

url

L'argument obligatoire *url* doit contenir l'une des deux valeurs suivantes :

- le mot-clé `"site"`, qui indique si `compare()` doit agir sur la sélection dans le panneau Site ;
- l'URL d'un fichier local à comparer avec sa version distante.

Valeurs renvoyées

Valeur booléenne : `true` si la comparaison réussit et `false` dans le cas contraire.

Activateur

Voir *site.canCompareFiles()*, page 607.

Exemple

L'exemple suivant compare les fichiers sélectionnés dans le panneau Site avec leurs versions distantes :

```
site.compareFiles("site");
```

site.defineSites()

Disponibilité

Dreamweaver 3.

Description

La fonction ouvre la boîte de dialogue Modifier les sites.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.deleteSelection()

Disponibilité

Dreamweaver 3.

Description

Supprime les fichiers sélectionnés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.deployFilesToTestingServerBin()

Disponibilité

Dreamweaver MX.

Description

Place le ou les fichiers spécifiés dans le dossier bin du serveur d'évaluation. Si aucun paramètre relatif au déploiement des fichiers de prise en charge n'est défini pour le site en cours, cette fonction appelle la boîte de dialogue Déployer les fichiers de prise en charge sur le serveur d'évaluation.

Arguments

`filesToDeploy`

- L'argument *`filesToDeploy`* correspond à un tableau de noms de fichiers que Dreamweaver déploiera.

Valeurs renvoyées

Valeur booléenne : `true` si le déploiement des fichiers réussit et `false` dans le cas contraire.

Exemple

Cet exemple déploie les fichiers image1.jpg et script1.js dans le dossier bin du serveur d'évaluation :

```
site.deployFilesToTestingServerBin("image1.jpg", "script1.js");
```

site.editColumns()

Description

Cette fonction affiche la boîte de dialogue Modifier les sites, qui contient la section Colonnes en mode Fichier.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.exportSite()

Disponibilité

Dreamweaver MX.

Description

Exporte un site Dreamweaver dans un fichier XML, qui peut être importé dans une autre instance de Dreamweaver pour permettre la duplication du site.

Toutes les informations contenues dans la boîte de dialogue Définition du site sont enregistrées dans un fichier XML qui comprend la liste des dossiers voilés et des informations sur le type de document par défaut. La seule exception est que l'utilisateur peut omettre le nom et le mot de passe de connexion lorsque l'accès FTP est défini. L'exemple suivant présente un fichier XML créé par Dreamweaver lors de l'exportation d'un site :

```
<?xml version="1.0" ?>
<site>
  <localinfo
    sitename="DW00"
    localroot="C:\Documents and Settings\jlondon\Desktop\DWServer\"
    imagefolder="C:\Documents and
  Settings\jlondon\Desktop\DWServer\Images\"
    spacerfilepath=""
    refreshlocal="TRUE"
    cache="FALSE"
    httpaddress="http://" curserver="webserver" />
  <remoteinfo
    accesstype="ftp"
    host="dreamweaver"
    remoteroot="kojak/"
    user="dream"
    checkoutname="Jay"
    emailaddress="jay@macromedia.com"
    usefirewall="FALSE"
    usepasv="TRUE"
    enablecheckin="TRUE"
    checkoutwhenopen="TRUE" />
  <designnotes
    usedesignnotes="TRUE"
    sharedesignnotes="TRUE" />
  <sitemap
    homepage="C:\Documents and Settings\jlondon\Desktop\DWServer\Untitled-
  2.htm"
    pagesperrow="200" columnwidth="125" showdependentfiles="TRUE"
    showpagetitles="FALSE" showhiddenfiles="TRUE" />
  <fileviewcolumns sharecolumns="TRUE">
    <column name="Local Folder"
      align="left" show="TRUE" share="FALSE" builtin="TRUE"
```

```
      localwidth="180" remotewidth="180" />
    <column name="Notes"
      align="center" show="TRUE" share="FALSE" builtin="TRUE"
      localwidth="36" remotewidth="36" />
    <column name="Size"
      align="right" show="TRUE" share="FALSE" builtin="TRUE"
      localwidth="-2" remotewidth="-2" />
    <column name="Type"
      align="left" show="TRUE" share="FALSE" builtin="TRUE"
      localwidth="60" remotewidth="60" />
    <column name="Modified"
      align="left" show="TRUE" share="FALSE" builtin="TRUE"
      localwidth="102" remotewidth="102" />
    <column name="Checked Out By"
      align="left" show="TRUE" share="FALSE" builtin="TRUE"
      localwidth="50" remotewidth="50" />
    <column name="Status" note="status"
      align="left" show="TRUE" share="FALSE" builtin="FALSE"
      localwidth="50" remotewidth="50" />
  </fileviewcolumns>
  <appserverinfo
    servermodel="ColdFusion"
    urlprefix="http://dreamweaver/kojak/"
    serverscripting="CFML"
    serverpageext=""
    connectionsmigrated="TRUE"
    useUD4andUD5pages="TRUE"
    defaultdoctype=""
    accesstype="ftp"
    host="dreamweaver"
    remoteroot="kojak/"
    user="dream"
    usefirewall="FALSE"
    usepasv="TRUE" />
  <cloaking enabled="TRUE" patterns="TRUE">
    <cloakedfolder folder="databases/" />
    <cloakedpattern pattern=".png" />
    <cloakedpattern pattern=".jpg" />
    <cloakedpattern pattern=".jpeg" />
  </cloaking>
</site>
```

Arguments

siteName

- L'argument *siteName* identifie le site à exporter. Si *siteName* est une chaîne vide, Dreamweaver exporte le site en cours.

Valeurs renvoyées

Valeur booléenne : `true` si le site nommé existe et si le fichier XML est correctement exporté, et `false` dans le cas contraire.

site.findLinkSource()

Disponibilité

Dreamweaver 3.

Description

Ouvre le fichier contenant le lien ou le fichier dépendant sélectionné et met en surbrillance le texte du lien ou la référence au fichier dépendant. Cette fonction agit seulement sur les fichiers de l'affichage Carte du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canFindLinkSource()*, page 608.

site.get()

Disponibilité

Dreamweaver 3.

Description

Obtient les fichiers spécifiés et traite les fichiers dépendants comme suit :

- Si l'utilisateur a activé l'option Invite lors de Acquérir/Extraire dans les préférences, catégorie FTP du site, la boîte de dialogue Fichiers dépendants s'affiche.
- Si l'utilisateur a activé l'option Ne plus afficher ce message dans la boîte de dialogue Fichiers dépendants, puis qu'il a cliqué sur Oui, les fichiers dépendants sont téléchargés et aucune boîte de dialogue ne s'affiche.
- Si l'utilisateur a activé l'option Ne plus afficher ce message dans la boîte de dialogue Fichiers dépendants, puis qu'il a cliqué sur Non, les fichiers dépendants ne sont pas téléchargés et aucune boîte de dialogue ne s'affiche.

Arguments

siteOrURL

- L'argument *siteOrURL* doit être soit le mot-clé `"site"`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canGet()*, page 608.

site.getAppServerAccessType()

Disponibilité

Dreamweaver MX.

Description

Renvoie la méthode d'accès utilisée pour tous les fichiers sur le serveur d'application du site en cours. Le site en cours est le site associé au document actuellement activé. Si aucun document n'est activé, le site ouvert dans Dreamweaver est utilisé.

REMARQUE	ColdFusion Component Explorer utilise cette fonction ; voir site.getAppServerPathToFiles() et site.getLocalPathToFiles().

Arguments

Aucun.

Valeurs renvoyées

L'une des chaînes suivantes :

- `"none"`
- `"local/network"`
- `"ftp"`
- `"source_control"`

site.getAppServerPathToFiles()

Disponibilité

Dreamweaver MX.

Description

Détermine le chemin d'accès aux fichiers distants sur le serveur d'application défini pour le site en cours. Le site en cours est le site associé au document actuellement activé. Si aucun document n'est activé, le site ouvert dans Dreamweaver est utilisé.

REMARQUE	ColdFusion Component Explorer utilise cette fonction ; voir site.getAppServerAccessType() et site.getLocalPathToFiles().

Arguments

Aucun.

Valeurs renvoyées

Si le type d'accès au serveur d'application est `local/network`, cette fonction renvoie un chemin ; dans le cas contraire, cette fonction renvoie une chaîne vide.

site.getAppURLPrefixForSite()

Disponibilité

Dreamweaver MX.

Description

Renvoie la valeur du préfixe de l'URL qui est extraite de l'adresse HTTP définie dans la section Infos locales de la boîte de dialogue de définition du site. Il s'agit du chemin figurant après http://*hostname*:*portnumber*/.

Arguments

{ siteName }

L'argument facultatif *siteName* est le nom du site dont vous souhaitez obtenir le préfixe d'URL. Si vous n'indiquez pas de lien, la fonction obtient le préfixe d'URL du site en cours.

Valeurs renvoyées

Une chaîne contenant le préfixe d'URL du site sélectionné.

Exemple

```
var sitePrefix = site.getAppURLPrefixForSite();
```

site.getCheckOutUser()

Disponibilité

Dreamweaver 3.

Description

Obtient les noms d'utilisateur et d'extraction associés au site en cours.

Arguments

Aucun.

Valeurs renvoyées

Chaîne contenant un nom d'utilisateur et un nom d'extraction, s'ils sont définis, ou une chaîne vide si les fonctions d'archivage et d'extraction sont désactivées.

Exemple

Un appel à la fonction `site.getCheckOutUser()` pourrait renvoyer ceci : `"denise (deniseLaptop)"`. Si aucun nom d'extraction n'est spécifié, seul le nom d'utilisateur est renvoyé (`"denise"`, par exemple).

site.getCheckOutUserForFile()

Disponibilité

Dreamweaver 3.

Description

Obtient le nom d'utilisateur et le nom d'extraction de l'utilisateur qui a extrait le fichier spécifié.

Arguments

`fileName`

- L'argument *`fileName`* correspond au chemin d'accès au fichier interrogé, exprimé sous la forme d'une URL de type file://.

Valeurs renvoyées

Chaîne contenant le nom d'utilisateur et le nom d'extraction de l'utilisateur qui a extrait le fichier spécifié, ou chaîne vide si le fichier n'a pas été extrait.

Exemple

Un appel à la fonction `site.getCheckOutUserForFile("file://C:/sites/avocado8/index.html")` pourrait renvoyer ceci : `"denise (deniseLaptop)"`. Si aucun nom d'extraction n'est spécifié, seul le nom d'utilisateur est renvoyé (`"denise"`, par exemple).

site.getCloakingEnabled()

Disponibilité

Dreamweaver MX.

Description

Détermine si le voilage est activé pour le site en cours.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le voilage est activé pour le site en cours et `false` dans le cas contraire.

site.getConnectionState()

Disponibilité

Dreamweaver 3.

Description

Obtient l'état de connexion en cours.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant si le site distant est connecté.

Activateur

Voir *site.canConnect()*, page 607.

site.getCurrentSite()

Disponibilité

Dreamweaver 3.

Description

Obtient le site en cours.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui contient le nom du site en cours.

Exemple

Si plusieurs sites sont définis, un appel à la fonction `site.getCurrentSite()` renvoie celui qui est actuellement affiché dans la liste des sites en cours du panneau Site.

site.getFocus()

Disponibilité

Dreamweaver 3.

Description

Détermine quel volet du panneau Site est activé.

Arguments

Aucun.

Valeurs renvoyées

L'une des chaînes suivantes :

- `"local"`
- `"remote"`
- `"site map"`

site.getLinkVisibility()

Disponibilité

Dreamweaver 3.

Description

Vérifie si tous les liens sélectionnés dans la carte du site sont visibles (c'est-à-dire s'ils ne sont pas identifiés comme masqués).

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si tous les liens sélectionnés sont visibles et `false` dans le cas contraire.

site.getLocalPathToFiles()

Disponibilité

Dreamweaver MX.

Description

Détermine le chemin d'accès aux fichiers locaux définis pour le site en cours. Le site en cours est le site associé au document actuellement activé. Si aucun document n'est activé, le site ouvert dans Dreamweaver est utilisé.

REMARQUE

ColdFusion Component Explorer utilise cette fonction ; voir site.getAppServerAccessType() et site.getAppServerPathToFiles().

Arguments

Aucun.

Valeurs renvoyées

Chemin d'accès aux fichiers qui résident sur l'ordinateur local pour le site en cours.

site.getSelection()

Disponibilité

Dreamweaver 3.

Description

Détermine quels sont les fichiers actuellement sélectionnés dans le panneau Site.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes représentant les chemins d'accès aux fichiers et dossiers sélectionnés, exprimé sous la forme d'une URL de type file://, ou d'un tableau vide si aucun fichier ni dossier n'est sélectionné.

site.getSiteForURL()

Disponibilité

Dreamweaver MX.

Description

Obtient le nom du site, s'il en a un, associé à un fichier donné.

Arguments

`fileURL`

- L'argument *`fileURL`* est l'URL complète (y compris la chaîne « `file://` ») d'un fichier nommé.

Valeurs renvoyées

Chaîne qui contient le nom du site, s'il en a un, dans lequel réside le fichier spécifié. La chaîne est vide lorsque le fichier spécifié n'existe dans aucun site défini.

site.getSites()

Disponibilité

Dreamweaver 3.

Description

Obtient la liste des sites définis.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes représentant les noms des sites définis, ou tableau vide si aucun site n'est défini.

site.getSiteURLPrefix()

Disponibilité

Dreamweaver 8.

Description

Obtient le préfixe de l'URL qui est extrait de l'adresse HTTP définie dans la section Infos locales.

Arguments

Aucun.

Valeurs renvoyées

Chaîne contenant le préfixe de l'URL du site.

Exemple

sitePrefix = getSiteURLPrefix();

site.importSite()

Disponibilité

Dreamweaver MX.

Description

Crée un site Dreamweaver à partir d'un fichier XML. Au cours de l'importation, si le dossier spécifié par l'attribut `localroot` de l'élément `<localinfo>` n'existe pas sur l'ordinateur local, Dreamweaver invite l'utilisateur à choisir un dossier racine local différent. Dreamweaver se comporte de la même manière lorsqu'il tente de localiser le dossier d'image par défaut spécifié par l'attribut `imagefolder` de l'élément `<localinfo>`.

Arguments

`fileURL`

- L'argument *`fileURL`* est une chaîne qui contient l'URL du fichier XML. Dreamweaver utilise ce fichier XML pour créer un nouveau site. Si `fileURL` est une chaîne vide, Dreamweaver invite l'utilisateur à choisir un fichier XML à importer.

Valeurs renvoyées

Valeur booléenne : `true` si le fichier XML nommé existe et si la création du site réussit et `false` dans le cas contraire.

site.invertSelection()

Disponibilité

Dreamweaver 3.

Description

Inverse la sélection dans la carte du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.isCloaked()

Disponibilité

Dreamweaver MX.

Description

Détermine si la sélection en cours dans le panneau Site ou le dossier spécifié est voilé.

Arguments

`siteOrURL`

- L'argument *`siteOrURL`* doit contenir l'une des deux valeurs suivantes :
  - le mot-clé `"site"`, qui indique que la fonction `isCloaked()` doit tester la sélection dans le panneau Site ;
  - l'URL de fichier d'un dossier particulier, qui indique si `isCloaked()` doit tester le dossier spécifié.

Valeurs renvoyées

Valeur booléenne : `true` si l'objet spécifié est voilé et `false` dans le cas contraire.

site.locateInSite()

Disponibilité

Dreamweaver 3.

Description

Recherche le ou les fichiers indiqués dans le volet spécifié du panneau Site et sélectionne les fichiers.

Arguments

localOrRemote, siteOrURL

- L'argument *localOrRemote* doit avoir la valeur `"local"` ou `"remote"`.
- L'argument *siteOrURL* doit être soit le mot-clé `"site"`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canLocateInSite()*, page 609.

site.makeEditable()

Disponibilité

Dreamweaver 3.

Description

Désactive le drapeau de lecture seule sur les fichiers sélectionnés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canMakeEditable()*, page 609.

site.makeNewDreamweaverFile()

Disponibilité

Dreamweaver 3.

Description

Crée un nouveau fichier Dreamweaver dans le panneau Site (dans le même dossier que le premier fichier ou dossier sélectionné).

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canMakeNewFileOrFolder()*, page 610.

site.makeNewFolder()

Disponibilité

Dreamweaver 3.

Description

Crée un nouveau dossier dans le panneau Site (dans le même dossier que le premier fichier ou dossier sélectionné).

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canMakeNewFileOrFolder()*, page 610.

site.newHomePage()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Nouvelle page d'accueil pour permettre à l'utilisateur de créer une nouvelle page d'accueil.

REMARQUE	Cette fonction agit seulement sur les fichiers de l'affichage Carte du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.newSite()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Définition du site pour définir un nouveau site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.open()

Disponibilité

Dreamweaver 3.

Description

Ouvre les fichiers actuellement sélectionnés dans le panneau Site. Si des dossiers sont sélectionnés, ils sont développés dans l'affichage Fichiers du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canOpen()*, page 610.

site.put()

Disponibilité

Dreamweaver 3.

Description

Place les fichiers spécifiés et traite les fichiers dépendants comme suit :

- Si l'utilisateur a activé l'option Invite lors de Placer/Archiver dans les préférences, catégorie FTP du site, la boîte de dialogue Fichiers dépendants s'affiche.
- Si l'utilisateur a activé l'option Ne plus afficher ce message dans la boîte de dialogue Fichiers dépendants, puis qu'il a cliqué sur Oui, les fichiers dépendants sont téléchargés et aucune boîte de dialogue ne s'affiche.
- Si l'utilisateur a activé l'option Ne plus afficher ce message dans la boîte de dialogue Fichiers dépendants, puis qu'il a cliqué sur Non, les fichiers dépendants ne sont pas téléchargés et aucune boîte de dialogue ne s'affiche.

Arguments

`siteOrURL`

- L'argument *`siteOrURL`* doit être soit le mot-clé `"site"`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canPut()*, page 611.

site.recreateCache()

Disponibilité

Dreamweaver 3.

Description

Recrée le cache du site en cours.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canRecreateCache()*, page 611.

site.refresh()

Disponibilité

Dreamweaver 3.

Description

Actualise la liste des fichiers sur le côté spécifié du panneau Site.

Arguments

whichSide

- L'argument *whichSide* doit avoir la valeur `"local"` ou `"remote"`. Si la carte du site est active et que *whichSide* a pour valeur `"local"`, la carte du site est actualisée.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canRefresh()*, page 612.

site.remoteIsValid()

Disponibilité

Dreamweaver 3.

Description

Détermine si le site distant est valide.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne qui indique si un site distant a été défini et, dans le cas où le type de serveur est Local/Réseau, si le lecteur est monté.

site.removeLink()

Disponibilité

Dreamweaver 3.

Description

Supprime le lien sélectionné du document situé au-dessus de lui dans la carte du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canRemoveLink()*, page 612.

site.renameSelection()

Disponibilité

Dreamweaver 3.

Description

Transforme le nom du fichier sélectionné en champ de texte et permet à l'utilisateur de renommer le fichier. Si plusieurs fichiers sont sélectionnés, cette fonction agit sur le dernier.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.runValidation()

Disponibilité

Dreamweaver MX.

Description

Exécute le validateur sur la totalité du site ou uniquement sur les éléments mis en surbrillance.

Arguments

selection

- L'argument *selection* est le paramètre qui spécifie que le validateur doit vérifier les éléments mis en surbrillance uniquement ; dans le cas contraire, le validateur vérifie l'ensemble du site en cours.

Valeurs renvoyées

Aucune.

site.saveAsImage()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Enregistrer sous pour permettre à l'utilisateur d'enregistrer la carte du site sous forme d'image.

Arguments

fileType

- L'argument *fileType* correspond au type d'image à enregistrer. Pour Windows, les valeurs autorisées sont `"bmp"` et `"png"`, et pour Macintosh `"pict"` et `"jpeg"`. Si cet argument n'est pas défini ou que sa valeur n'est pas valide sur la plate-forme en cours, il prend par défaut la valeur `"bmp"` pour Windows et `"pict"` pour Macintosh.

Valeurs renvoyées

Aucune.

site.selectAll()

Disponibilité

Dreamweaver 3.

Description

Sélectionne tous les fichiers de l'affichage actif (à savoir soit la carte du site, soit les fichiers de site).

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.selectHomePage()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue d'ouverture de fichier pour permettre à l'utilisateur de sélectionner une nouvelle page d'accueil.

REMARQUE	Cette fonction agit seulement sur les fichiers de l'affichage Carte du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.selectNewer()

Disponibilité

Dreamweaver 3.

Description

Sélectionne tous les fichiers les plus récents sur le côté spécifié du panneau Site.

Arguments

whichSide

- L'argument *whichSide* doit avoir la valeur `"local"` ou `"remote"`.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canSelectNewer()*, page 613.

site.serverActivity()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si Dreamweaver est en train d'interagir avec un serveur. Dreamweaver ne peut effectuer qu'une seule activité de serveur à la fois, cette fonction vous permet donc de déterminer s'il faut désactiver la fonctionnalité exigeant une interaction du serveur.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant si Dreamweaver est en train d'interagir avec un serveur.

Exemple

L'exemple suivant, extrait du fichier menus.xml, affiche un élément de menu s'il n'y a pas d'activité du serveur (et si un site est actuellement spécifié dans Dreamweaver) :

```
<menuitem name="Remove Connection Scripts" enabled="!site.serverActivity()
  && site.getCurrentSite() != ''"
  command="alert(MMDB.removeConnectionScripts())"
  id="SiteOptionsSiteMenu_RemoveConnectionScripts" />
```

site.setAsHomePage()

Disponibilité

Dreamweaver 3.

Description

Désigne le fichier sélectionné dans l'affichage Fichiers du site comme étant la page d'accueil du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

site.setCloakingEnabled()

Disponibilité

Dreamweaver MX.

Description

Détermine s'il est nécessaire d'activer le voilage pour le site en cours.

Arguments

enable

- L'argument *enable* est une valeur booléenne qui indique s'il est nécessaire d'activer le voilage. Une valeur `true` active le voilage pour le site en cours ; une valeur `false` désactive le voilage pour le site en cours.

Valeurs renvoyées

Aucun.

site.setConnectionState()

Disponibilité

Dreamweaver 3.

Description

Définit l'état de connexion du site en cours.

Arguments

bConnected

- L'argument *bConnected* est une valeur booléenne qui indique s'il existe une connexion (`true`) ou non (`false`) au site en cours.

Valeurs renvoyées

Aucune.

site.setCurrentSite()

Disponibilité

Dreamweaver 3.

Description

Ouvre le site spécifié dans le volet local du panneau Site.

Arguments

whichSite

- L'argument *whichSite* est le nom d'un site défini (tel qu'il apparaît dans la liste des sites en cours, dans le panneau Site ou dans la boîte de dialogue Modifier les sites).

Valeurs renvoyées

Aucune.

Exemple

Si trois sites sont définis (avocado8, dreamcentral et testsite, par exemple), un appel à la fonction `site.setCurrentSite("dreamcentral")` fait de dreamcentral le site en cours.

site.setFocus()

Disponibilité

Dreamweaver 3.

Description

Active le volet spécifié du panneau Site. Si le volet spécifié n'est pas affiché, la fonction l'affiche et l'active.

Arguments

whichPane

- L'argument *whichPane* doit correspondre à l'une des chaînes suivantes : `"local"`, `"remote"` ou `"site map"`.

Valeurs renvoyées

Aucune.

site.setLayout()

Disponibilité

Dreamweaver 3.

Description

Ouvre le volet Mise en forme de la carte du site dans la boîte de dialogue Définition du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canSetLayout()*, page 613.

site.setLinkVisibility()

Disponibilité

Dreamweaver 3.

Description

Affiche ou masque le lien en cours.

Arguments

bShow

- L'argument *bShow* est une valeur booléenne qui indique si le lien en cours ne doit plus être identifié comme masqué.

Valeurs renvoyées

Aucune.

site.setSelection()

Disponibilité

Dreamweaver 3.

Description

Sélectionne les fichiers ou les dossiers visibles dans le volet actif du panneau Site.

Arguments

arrayOfURLs

- L'argument *arrayOfURLs* est un tableau de chaînes correspondant aux chemins d'accès aux fichiers et aux dossiers du site sélectionné, exprimés sous la forme d'une URL de type file://.

REMARQUE	Pour les chemins de dossier, ne tapez pas la barre oblique (/) à la fin du chemin.

Valeurs renvoyées

Aucune.

site.siteRelativeToLocalPath()

Disponibilité

Dreamweaver 8.

Description

Convertit une référence d'URI relative de site en un chemin d'accès de fichier local.

Arguments

siteRelativeURI

- L'argument obligatoire `siteRelativeURI` est une chaîne contenant l'URI relative du site.

Valeurs renvoyées

Chaîne indiquant le chemin d'accès à un fichier local sur votre ordinateur local.

Exemple

L'exemple suivant

```
var filePath = site.siteRelativeToLocalPath("/myWebApp/myFile.xlml");
```

renvoie "C:\Inetpub\wwwroot\siteA\myFile.xml", en fonction de vos mappages de site et de l'adresse HTTP spécifiée dans la section Infos locales de la boîte de dialogue Définition du site.

site.synchronize()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Synchroniser les fichiers.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canSynchronize()*, page 614.

site.uncloak()

Disponibilité

Dreamweaver MX.

Description

Dévoile la sélection dans le panneau Site ou le dossier spécifié.

Arguments

siteOrURL

- L'argument *siteOrURL* doit contenir l'une des deux valeurs suivantes :
  - le mot-clé "site", qui indique que la fonction unCloak() doit agir sur la sélection dans le panneau Site ;
  - l'URL d'un dossier particulier, qui indique que la fonction unCloak() doit agir sur le dossier spécifié et tout son contenu.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canUncloak()*, page 615.

site.uncloakAll()

Disponibilité

Dreamweaver MX.

Description

Dévoile tous les dossiers dans le site sélectionné et désactive la case à cocher Voiler les fichiers se terminant avec : dans les Paramètres de voilage.

Arguments

Aucune.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canUncloak()*, page 615.

site.undoCheckOut()

Disponibilité

Dreamweaver 3.

Description

Retire des sites locaux et distants les fichiers verrouillés associés aux fichiers sélectionnés et remplace leur copie locale par la copie distante.

Arguments

siteOrURL

- L'argument *siteOrURL* doit être soit le mot-clé "site", indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canUndoCheckOut()*, page 615.

site.viewAsRoot()

Disponibilité

Dreamweaver 3.

Description

Place provisoirement le fichier sélectionné en première position sur la carte du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *site.canViewAsRoot()*, page 616.

CHAPITRE 13

Document

13

Les fonctions Document de Macromedia Dreamweaver 8 permettent d'effectuer des opérations dans le document modifié par l'utilisateur. Ces fonctions permettent notamment de convertir les tableaux en calques, d'exécuter une commande du dossier Configuration/Commands, de rechercher l'URL d'un fichier, de vérifier l'orthographe ou de définir les propriétés d'une page, de convertir une URL relative en URL absolue, d'obtenir le nœud parent sélectionné, d'effectuer l'encodage URL d'une chaîne ou d'exécuter un traducteur sur le document.

Fonctions relatives aux conversions

Ces fonctions permettent de convertir des tableaux en calques, des calques en tableaux et des feuilles de style en cascade (CSS) en balises HTML. Chacune de ces fonctions se comporte exactement comme une commande de conversion du menu Fichier ou Modifier.

dom.convertLayersToTable()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Convertir les calques en tableau.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canConvertLayersToTable()*, page 572.

dom.convertTablesToLayers()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Convertir les tableaux en calques.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canConvertTablesToLayers()*, page 572.

Fonctions relatives aux commandes

Ces fonctions permettent d'exploiter au mieux les fichiers figurant dans le dossier Configuration/Commands. Elles permettent de gérer le menu Commandes et d'appeler des commandes à partir d'autres types de fichiers d'extension.

dreamweaver.editCommandList()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Modifier la liste de commandes.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.popupCommand() (déconseillée)

Disponibilité

Dreamweaver 2, déconseillée depuis la version 3 : utiliser à la place dreamweaver.runCommand().

Description

Cette fonction exécute la commande spécifiée. Pour l'utilisateur, cela revient à choisir la commande dans un menu ; si une boîte de dialogue est associée à la commande, elle s'affiche. Cette fonction permet d'appeler une commande à partir d'un autre fichier d'extension. L'utilisateur ne peut effectuer aucune autre modification tant qu'il n'a pas fermé la boîte de dialogue.

REMARQUE

Cette fonction peut être appelée à partir de la fonction `objectTag()`, d'un script de fichier de commande ou du fichier d'inspecteur de propriétés.

Arguments

`commandFile`

- L'argument *`commandFile`* correspond au nom d'un fichier de commandes du dossier Configuration/Commands (`"Format Table.htm"`, par exemple).

Valeurs renvoyées

Aucune.

dreamweaver.runCommand()

Disponibilité

Dreamweaver 3.

Description

Exécute la commande spécifiée ; cette fonction équivaut à choisir la commande dans un menu. Si une boîte de dialogue est associée à la commande, elle apparaît et le script de la commande bloque les autres modifications jusqu'à ce que l'utilisateur ferme la boîte de dialogue. Cette fonction permet d'appeler une commande à partir d'un autre fichier d'extension.

REMARQUE	Cette fonction peut être appelée à partir de la fonction `objectTag()`, d'un script de fichier de commande ou du fichier d'inspecteur de propriétés.

Arguments

`commandFile, {commandArg1}, {commandArg2},...{commandArgN}`

- L'argument *`commandFile`* est un nom de fichier dans le dossier Configuration/Commands.
- Les autres arguments (*`commandArg1, commandArg2`*, etc.) sont facultatifs. Ils sont transmis à la fonction `receiveArguments()` dans l'argument *`commandFile`*.

Valeurs renvoyées

Aucune.

Exemple

Vous pouvez créer un inspecteur de propriétés personnalisé pour les tableaux. Il permettra à l'utilisateur d'accéder à la commande Formater le tableau à l'aide d'un bouton de l'inspecteur. Pour ce faire, appelez la fonction suivante à partir du gestionnaire d'événements `onClick` de ce bouton :

```
function callFormatTable(){
  dreamweaver.runCommand('Format Table.htm');
}
```

Fonctions relatives aux manipulations de fichiers

Ces fonctions permettent de créer, d'ouvrir et d'enregistrer des documents (y compris XML et XHTML), de convertir des documents HTML existants en XHTML et d'exporter des styles CSS vers des fichiers externes. Elles permettent également de rechercher des fichiers ou des dossiers, de créer des fichiers à partir de modèles, de fermer des documents et d'obtenir la liste des fichiers récemment ouverts.

dom.cleanupXHTML()

Disponibilité

Dreamweaver MX.

Description

Cette fonction est similaire à la fonction `convertToXHTML()`, à ceci près que celle-ci nettoie un document XHTML existant. Cette fonction peut être exécutée sur une sélection à l'intérieur du document. Vous pouvez utiliser la fonction `cleanupXHTML()` pour nettoyer la syntaxe dans l'ensemble d'un document XHTML ou dans la sélection en cours à l'intérieur du document.

Arguments

`bWholeDoc`

- L'argument *`bWholeDoc`* contient une valeur booléenne. Si la valeur est `true`, la fonction `cleanupXHTML()` nettoie l'ensemble du document ; dans le cas contraire, elle nettoie uniquement la sélection.

Valeurs renvoyées

Un tableau de six nombres entiers qui quantifient le nombre d'éléments suivants :

- erreurs XHTML réparées par Dreamweaver
- éléments `map` ne disposant pas d'un attribut `id` et ne pouvant pas être réparés
- éléments `script` ne disposant pas d'un attribut `type` et ne pouvant pas être réparés
- éléments `style` ne disposant pas d'un attribut `type` et ne pouvant pas être réparés
- éléments `img` ne disposant pas d'un attribut `alt` et ne pouvant pas être réparés
- éléments `area` ne disposant pas d'un attribut `alt` et ne pouvant pas être réparés

dom.convertToXHTML()

Disponibilité

Dreamweaver MX.

Description

Analyse le code HTML dans une arborescence DOM, insère les éléments manquants qui sont obligatoires pour XHTML, nettoie l'arborescence puis écrit celle-ci au format XHTML. Les directives, déclarations, éléments et attributs manquants que la fonction `convertToXHTML()` ajoute à l'arborescence DOM si nécessaire, incluent les éléments suivants :

- Une directive XML

- Une déclaration `doctype`
- L'attribut `xmlns` dans l'élément `html`
- Une section `head`
- Un élément `title`
- Une section `body`

Pendant la conversion, la fonction `dom.convertToXHTML()` convertit les balises et les attributs HTML purs en minuscules, écrit des balises et des attributs HTML à l'aide d'une syntaxe XHTML correcte et ajoute des attributs HTML manquants où elle le peut. Cette fonction traite les balises et les attributs tiers en fonction des paramètres de la boîte de dialogue Préférences.

Si le document est un modèle, la fonction `dom.convertToXHTML()` alerte l'utilisateur mais n'effectue pas la conversion.

Arguments

Aucun.

Valeurs renvoyées

Un tableau de six nombres entiers qui quantifient les éléments suivants :

- erreurs XHTML réparées par Dreamweaver
- éléments `map` ne disposant pas d'un attribut `id` et ne pouvant pas être réparés
- éléments `script` ne disposant pas d'un attribut `type` et ne pouvant pas être réparés
- éléments `style` ne disposant pas d'un attribut `type` et ne pouvant pas être réparés
- éléments `img` ne disposant pas d'un attribut `alt` et ne pouvant pas être réparés
- éléments `area` ne disposant pas d'un attribut `alt` et ne pouvant pas être réparés

Exemple

Dans le cadre d'une utilisation normale, une extension appelle d'abord la fonction `dreamweaver.openDocument()` ou `dreamweaver.getDocumentDOM()` pour obtenir une référence au document. L'extension appelle ensuite la fonction `dom.getIsXHTMLDocument()` pour déterminer si le document est déjà au format XHTML. Si ce n'est pas le cas, l'extension appelle la fonction `dom.convertToXHTML()` pour convertir le document au format XHTML. Ensuite, l'extension appelle la fonction `dreamweaver.saveDocument()` pour enregistrer le fichier converti sous un nouveau nom.

dom.getIsXHTMLDocument()

Disponibilité

Dreamweaver MX.

Description

Vérifie un document (en particulier la déclaration `<!DOCTYPE>`) pour savoir si celui-ci est au format XHTML.

Arguments

Aucun.

Valeurs renvoyées

`true` si le document est au format XHTML ; `false` dans le cas contraire

dreamweaver.browseForFileURL()

Disponibilité

Dreamweaver 1, améliorée dans les versions 2, 3 et 4.

Description

Ouvre le type de boîte de dialogue spécifié ayant le libellé spécifié dans la barre de titre.

Arguments

```
openSelectOrSave, {titleBarLabel}, {bShowPreviewPane}, ¬
  {bSupressSiteRootWarnings}, {arrayOfExtensions}
```

- L'argument *openSelectOrSave* est une chaîne qui indique le type de boîte de dialogue : `"open"`, `"select"` ou `"save"`.
- L'argument *titleBarLabel*, ajouté à Dreamweaver 2, est le libellé qui doit figurer dans la barre de titre de la boîte de dialogue. Si cet argument n'est pas défini, Dreamweaver utilise par défaut le libellé fourni par le système d'exploitation.
- L'argument *bShowPreviewPane*, ajouté à Dreamweaver 2, est une valeur booléenne indiquant si le volet d'aperçu de l'image doit être affiché dans la boîte de dialogue. Si l'argument a pour valeur `true`, la boîte de dialogue filtre les fichiers d'image ; s'il n'est pas défini, il prend par défaut la valeur `false`.
- L'argument *bSupressSiteRootWarnings*, ajouté à Dreamweaver 3, est une valeur booléenne indiquant s'il faut supprimer les avertissements signalant que le fichier sélectionné se trouve hors du dossier racine du site. Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

- L'argument *arrayOfExtensions*, ajouté à Dreamweaver 4, est un tableau de chaînes qui spécifie le contenu par défaut du menu déroulant Type au bas de la boîte de dialogue. La syntaxe correcte est `menuEntryText|.xxx[;.yyy;.zzz]|CCCC|` où *menuEntryText* est le nom du type de fichier qui s'affiche. Les extensions peuvent être spécifiées sous la forme *.xxx[;.yyy;.zzz]* ou *CCCC* où *.xxx* définit l'extension du type de fichier (*.yyy* et *.zzz* peuvent également spécifier des extensions de fichier multiples) et `CCCC` est la constante du type de fichier à quatre caractères utilisée sur Macintosh.

Valeurs renvoyées

Une chaîne contenant le nom du fichier, exprimé sous la forme d'une URL de type file://.

dreamweaver.browseForFolderURL()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Choisir un dossier ayant le libellé spécifié dans la barre de titre.

Arguments

{titleBarLabel}, {directoryToStartIn}

- L'argument *titleBarLabel* est le libellé qui doit s'afficher dans la barre de titre de la boîte de dialogue. S'il n'est pas spécifié, l'argument *titleBarLabel* prend par défaut la valeur « Choose Folder (Choisir un dossier) ».
- L'argument *directoryToStartIn* est le chemin indiquant où le dossier doit s'ouvrir, exprimé sous la forme d'une URL de type file://.

Valeurs renvoyées

Une chaîne contenant le nom du dossier, exprimé sous la forme d'une URL de type file://.

Exemple

Le code suivant renvoie l'URL d'un dossier :

```
return dreamweaver.browseForFolderURL('Select a Folder', ¬
dreamweaver.getSiteRoot());
```

dreamweaver.closeDocument()

Disponibilité

Dreamweaver 2.

Description

Ferme le document spécifié.

Arguments

documentObject

- L'argument *documentObject* correspond à l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`). Si l'argument *documentObject* fait référence au document actif, il se peut que la fenêtre du document ne se ferme pas tant que l'exécution du script qui appelle cette fonction n'est pas terminée.

Valeurs renvoyées

Aucune.

dreamweaver.createDocument()

Disponibilité

Dreamweaver 2, améliorée dans la version 4.

Description

En fonction de l'argument utilisé, cette fonction ouvre un nouveau document soit dans la même fenêtre, soit dans une nouvelle fenêtre. Le nouveau document devient le document actif.

REMARQUE	Cette fonction peut être appelée uniquement à partir du fichier menus.xml ou d'un fichier de commande ou d'inspecteur de propriétés. Si une action ou un objet tente d'appeler cette fonction, Dreamweaver affiche un message d'erreur.

Arguments

{bOpenInSameWindow}, {type}

- L'argument *bOpenInSameWindow* est une valeur booléenne indiquant si le nouveau document doit s'ouvrir dans la fenêtre en cours. Si l'argument *bOpenInSameWindow* a pour valeur `false`, qu'il n'est pas précisé ou que la fonction est appelée sur Macintosh, le nouveau document s'ouvre dans une nouvelle fenêtre.

- L'argument *type* indique le type de document à créer, conformément à ce qui a été déclaré dans le fichier Configuration/DocumentTypes/MMDocumentTypes.xml de Dreamweaver dans l'attribut `id` de la balise `documenttype`. L'argument *type* peut par exemple avoir la valeur `"HTML", "ASP-JS", "ASP-VB", "ColdFusion", "CFC", "JSP", "ASP.NET_VB"`, ou toute valeur similaire. Pour obtenir la liste complète des types disponibles, voir le fichier MMDocumentTypes.xml. Si vous ne définissez pas l'argument *type*, il adopte la valeur `"HTML"`.

REMARQUE	Vous pouvez étendre le fichier MMDocumentTypes en ajoutant de nouveaux types de documents. Pour plus d'informations sur l'extension des types de documents, voir ***Extension de Dreamweaver***.

Valeurs renvoyées

Objet document correspondant au nouveau document créé. Il s'agit de la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`.

dreamweaver.createXHTMLDocument()

Disponibilité

Dreamweaver MX.

Description

En fonction de l'argument utilisé, cette fonction ouvre un nouveau document XHTML soit dans la même fenêtre, soit dans une nouvelle fenêtre. Le nouveau document devient le document actif. Cette fonction est similaire à la fonction `dreamweaver.createDocument()`.

Lorsque Dreamweaver crée un nouveau document XHTML, il lit un fichier nommé default.xhtml, qui se trouve dans le dossier Configuration/Templates, et, en utilisant le contenu de ce fichier, il crée un fichier de sortie contenant les déclarations suivantes :

```
<?xml version="1.0">
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=" />
</head>

<body bgcolor="#FFFFFF" text="#000000">
```

```
</body>
</html>
```

La déclaration DTD par défaut est `XHTML 1.0 Transitional` et non pas `Strict`. Si l'utilisateur ajoute un jeu de cadres au document, Dreamweaver remplace la DTD par `XHTML 1.0 Frameset`. `Content-Type` est `text/html`, et `charset` est intentionnellement omis du fichier default.xhtml mais est rempli avant que l'utilisateur visualise le nouveau document. La directive `?xml` n'est pas obligatoire si le document utilise le codage de caractères UTF-8 ou UTF-16 ; si elle est présente, elle peut être restituée par d'autres navigateurs de version antérieure. Cependant, étant donné que cette directive doit se trouver dans un document XHTML, Dreamweaver l'utilise par défaut (à la fois pour les documents nouveaux et convertis). Les utilisateurs peuvent supprimer manuellement la directive. La directive `?xml` inclut l'attribut de codage qui correspond à `charset` dans l'attribut `Content-Type`.

Arguments

`{bOpenInSameWindow}`

- L'argument *`bOpenInSameWindow`* est une valeur booléenne indiquant si le nouveau document doit s'ouvrir dans la fenêtre en cours. Si cet argument a pour valeur `false`, qu'il n'est pas précisé ou que la fonction est appelée sur Macintosh, le nouveau document s'ouvre dans une nouvelle fenêtre.

Valeurs renvoyées

L'objet document correspondant au document nouvellement créé (même valeur que celle renvoyée par la fonction `dreamweaver.getDocumentDOM()`).

dreamweaver.createXMLDocument()

Disponibilité

Dreamweaver MX.

Description

Crée et ouvre un nouveau fichier XML contenant uniquement la directive XML.

Arguments

Aucun.

Valeurs renvoyées

Le DOM du nouveau fichier XML.

Exemple

L'exemple suivant crée un nouveau document, contenant uniquement la directive XML :

```
var theDOM = dreamweaver.createXMLDocument("document");
```

dreamweaver.exportCSS()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Exporter les styles dans un fichier CSS.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canExportCSS()*, page 584.

dreamweaver.exportEditableRegionsAsXML() (déconseillée)

Disponibilité

Dreamweaver 3, déconseillée depuis MX.

Description

Cette fonction ouvre la boîte de dialogue Exporter les régions modifiables sous XML.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.exportTemplateDataAsXML()

Disponibilité

Dreamweaver MX.

Description

Exporte le document actif dans le fichier indiqué, et ce au format XML. Cette fonction agit sur le document actif, qui doit être un modèle. Si vous ne spécifiez pas un argument de nom de fichier, Dreamweaver MX ouvre une boîte de dialogue pour demander la chaîne du fichier d'exportation.

Arguments

{filePath}

- L'argument facultatif *filePath* est une chaîne spécifiant le nom du fichier vers lequel Dreamweaver exporte le modèle. Exprimez l'argument *filePath* sous forme d'une URL de type `"file:///c|/temp/mydata.txt"`.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canExportTemplateDataAsXML()*, page 585.

Exemple

```
if(dreamweaver.canExportTemplateDataAsXML())
{
  dreamweaver.exportTemplateDataAsXML("file:///c|/dw_temps/
  mytemplate.txt")
}
```

dreamweaver.getDocumentDOM()

Disponibilité

Dreamweaver 2.

Description

Permet d'accéder à l'arborescence des objets du document spécifié. Une fois celle-ci renvoyée à l'appelant, ce dernier peut la modifier pour changer le contenu du document.

Arguments

{sourceDoc}

- L'argument *sourceDoc* doit être "document", "parent", "parent.frames[*number*]", "parent.frames['*frameName*']" ou une URL. L'argument sourceDoc prend par défaut la valeur "document" si vous ne fournissez pas de valeur. Ces valeurs sont expliquées ci-dessous :
  - La valeur document désigne le document actif contenant la sélection en cours.
  - La valeur parent désigne le jeu de cadres parent (si le document sélectionné se trouve dans un cadre).
  - Les valeurs parent.frames[*number*] et parent.frames['*frameName*'] désignent un document figurant dans un cadre spécifique du jeu de cadres contenant le document actif.
  - Si l'argument est une URL relative, celle-ci est relative au fichier de l'extension.

REMARQUE	Si l'argument a pour valeur "document", l'appelant doit être la fonction applyBehavior(), deleteBehavior(), objectTag() ou toute fonction d'un fichier de commande ou d'inspecteur de propriétés pouvant modifier le document.

Valeurs renvoyées

Objet document JavaScript à la racine de l'arborescence.

Exemples

L'exemple suivant utilise la fonction dreamweaver.getDocumentDOM() pour accéder au document actif :

```
var theDOM = dreamweaver.getDocumentDOM("document");
```

Dans l'exemple suivant, le document DOM actif identifie une sélection et la colle à la fin d'un autre document :

```
var currentDOM = dreamweaver.getDocumentDOM('document');
currentDOM.setSelection(100,200);
currentDOM.clipCopy();
var otherDOM = dreamweaver.openDocument(dreamweaver.¬
getSiteRoot() + "html/foo.htm");
otherDOM.endOfDocument();
otherDOM.clipPaste();
```

REMARQUE	L'argument openDocument() est utilisé car les méthodes de l'objet DOM agissent en principe uniquement sur les documents ouverts. L'exécution d'une fonction sur un document qui n'est pas ouvert entraîne une erreur de Dreamweaver. Lorsqu'une méthode de l'objet DOM ne peut être appliquée qu'au document actif ou à des documents fermés, cette caractéristique est indiquée dans sa description.

dreamweaver.getNewDocumentDOM()

Disponibilité

Dreamweaver MX ; argument `documentType` ajouté dans Dreamweaver 8.

Description

Permet d'accéder à l'arborescence modifiable d'un nouveau document vide. Cette fonction est similaire à la fonction `getDocumentDOM()`, à ceci près qu'elle pointe vers un nouveau document plutôt que vers un document existant et qu'elle n'ouvre pas le document.

Arguments

`{documentType}`

- L'argument *`documentType`* est une chaîne. Sa valeur doit être un type de document spécifié dans le fichier DocumentTypes.xml.

Valeurs renvoyées

Pointeur vers un nouveau document vide.

Exemple

Le code suivant renvoie le DOM d'un nouveau document vide :

```
var theDOM = dreamweaver.getNewDocumentDOM();
```

dreamweaver.getRecentFileList()

Disponibilité

Dreamweaver 3.

Description

Obtient la liste de tous les fichiers récemment ouverts et répertoriés au bas du menu Fichier.

Arguments

Aucun.

Valeurs renvoyées

Un tableau de chaînes représentant les chemins des derniers fichiers ouverts, exprimés sous la forme d'une URL de type file:///. Si aucun fichier n'a été ouvert récemment, la fonction ne renvoie aucune valeur.

dreamweaver.importXMLIntoTemplate()

Disponibilité

Dreamweaver 3.

Description

Importe un fichier de texte XML dans le modèle de document actif. Cette fonction agit sur le document actif, qui doit être un modèle. Si vous ne spécifiez pas un argument de nom de fichier, Dreamweaver ouvre une boîte de dialogue pour demander la chaîne du fichier d'importation.

Arguments

{filePath}

- L'argument facultatif *filePath* est une chaîne spécifiant le nom du fichier vers lequel Dreamweaver importe le modèle. Exprimez l'argument *filePath* sous forme d'une URL de type `"file:///c/temp/mydata.txt"`.

Valeurs renvoyées

Aucune.

dreamweaver.newDocument()

Disponibilité

Dreamweaver MX.

Description

Ouvre un document dans le site en cours et affiche la boîte de dialogue Nouveau document.

Arguments

{bopenWithCurSiteAndShowDialog}

- L'argument facultatif *bopenWithCurSiteAndShowDialog* peut avoir la valeur `true` ou `false`. Si vous souhaitez ouvrir un document dans le site en cours et afficher la boîte de dialogue Nouveau document, spécifiez `true` ; sinon, spécifiez `false`.

Valeurs renvoyées

Aucune.

dreamweaver.newFromTemplate()

Disponibilité

Dreamweaver 3.

Description

Crée un nouveau document à partir du modèle spécifié. Si vous ne spécifiez aucun argument, la boîte de dialogue Sélectionner le modèle s'affiche.

Arguments

{templateURL}, bMaintain

- L'argument *templateURL* est le chemin d'un modèle disponible sur le site en cours, exprimé sous la forme d'une URL de type file://.
- L'argument *bMaintain* est une valeur booléenne, pouvant être `true` ou `false`, indiquant si le lien vers le modèle d'origine doit être conservé ou pas.

Valeurs renvoyées

Aucune.

dreamweaver.openDocument()

Disponibilité

Dreamweaver 2.

Description

Ouvre un document à modifier dans une nouvelle fenêtre Dreamweaver et en fait le document actif. Pour l'utilisateur, cela revient à choisir Fichier > Ouvrir et à sélectionner un fichier. Si le fichier spécifié est déjà ouvert, la fenêtre correspondante s'affiche au premier plan. La fenêtre contenant le fichier spécifié devient la fenêtre active et le fichier sélectionné devient le document actif. Dans Dreamweaver 2, si la fonction d'archivage/extraction est activée, le fichier est extrait avant d'être ouvert. Dans Dreamweaver 3 et ses versions ultérieures, vous devez utiliser la fonction dreamweaver.openDocumentFromSite() pour obtenir ce comportement.

REMARQUE	Cette fonction provoque une erreur si elle est appelée à partir d'un fichier d'objet ou d'action de comportement.

Arguments

fileName

- L'argument *fileName* est le nom du fichier à ouvrir, exprimé sous forme d'une URL absolue. S'il s'agit d'une URL relative, elle est relative au fichier contenant le script ayant appelé cette fonction.

Valeurs renvoyées

L'objet document correspondant au fichier spécifié, soit la même valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`.

dreamweaver.openDocumentFromSite()

Disponibilité

Dreamweaver 3.

Description

Ouvre un document à modifier dans une nouvelle fenêtre Dreamweaver et en fait le document actif. Pour l'utilisateur, cela revient à double-cliquer sur un fichier dans le panneau Site. Si le fichier spécifié est déjà ouvert, la fenêtre correspondante s'affiche au premier plan. La fenêtre contenant le fichier spécifié devient la fenêtre active et le fichier sélectionné devient le document actif.

REMARQUE	Cette fonction ne peut pas être appelée à partir d'un fichier d'objet ou d'action de comportement sous peine de générer des erreurs.

Arguments

fileName

- L'argument *fileName* est le fichier à ouvrir, exprimé sous forme d'une URL absolue. S'il s'agit d'une URL relative, elle est relative au fichier contenant le script ayant appelé cette fonction.

Valeurs renvoyées

L'objet document correspondant au fichier spécifié, soit la même valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`.

dreamweaver.openInFrame()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Ouvrir dans un cadre. Lorsque l'utilisateur sélectionne un document, celui-ci s'ouvre dans le cadre actif.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canOpenInFrame()*, page 586.

dreamweaver.releaseDocument()

Disponibilité

Dreamweaver 2.

Description

Libère explicitement de la mémoire un document précédemment référencé.

Les documents référencés par les fonctions `dreamweaver.getObjectTags()`, `dreamweaver.getObjectRefs()`, `dreamweaver.getDocumentPath()` et `dreamweaver.getDocumentDOM()` sont automatiquement libérés au terme de l'exécution du script contenant l'appel. Si le script ouvre un nombre important de documents, vous devez utiliser cette fonction pour en libérer certains explicitement avant la fin de l'exécution du script, ceci afin d'éviter de saturer la mémoire.

REMARQUE	Cette fonction ne s'applique qu'aux documents référencés par une URL, qui ne sont pas ouverts dans un cadre ou dans une fenêtre de document et qui ne correspondent pas à des fichiers d'extension (les fichiers d'extensions externes sont chargés en mémoire au démarrage et n'en sont libérés que lorsque vous quittez Dreamweaver).

Arguments

`documentObject`

- L'argument *documentObject* est l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Aucune.

dreamweaver.revertDocument()

Disponibilité

Dreamweaver 3.

Description

Rétablit la version précédemment enregistrée du fichier spécifié.

Arguments

documentObject

- L'argument *documentObject* est l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canRevertDocument()*, page 588.

dreamweaver.saveAll()

Disponibilité

Dreamweaver 3.

Description

Enregistre tous les documents ouverts et ouvre la boîte de dialogue Enregistrer sous pour tous ceux qui n'ont pas encore été enregistrés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canSaveAll()*, page 588.

dreamweaver.saveDocument()

Disponibilité

Dreamweaver 2.

Description

Enregistre le fichier spécifié sur un ordinateur local.

REMARQUE	Dans Dreamweaver 2, si le fichier est accessible en lecture seule, Dreamweaver tente de l'extraire. Si le document reste en lecture seule après cette tentative ou qu'il ne peut pas être créé, un message d'erreur s'affiche.

Arguments

documentObject, {fileURL}

- L'argument *documentObject* est l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`).
- L'argument *fileURL* (facultatif) est une URL représentant un emplacement sur un ordinateur local. S'il s'agit d'une URL relative, elle est relative au fichier de l'extension. Dans Dreamweaver 2, cet argument est obligatoire. Dans Dreamweaver 4, si l'argument *fileURL* n'est pas défini et que le fichier a été enregistré précédemment, ce dernier est enregistré au même endroit ; sinon, une boîte de dialogue d'enregistrement s'affiche.

Valeurs renvoyées

Valeur booléenne indiquant le succès (`true`) ou l'échec (`false`) de l'opération.

Activateur

Voir *dreamweaver.canSaveDocument()*, page 589.

dreamweaver.saveDocumentAs()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Enregistrer sous.

Arguments

documentObject

- L'argument *documentObject* est l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Aucune.

dreamweaver.saveDocumentAsTemplate()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Enregistrer comme modèle.

Arguments

documentObject, [fileName]

- L'argument *documentObject* est l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`).
- L'argument *fileName* (facultatif) est le nom du fichier à ouvrir, exprimé sous forme d'une URL absolue.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canSaveDocumentAsTemplate()*, page 589.

dreamweaver.saveFrameset()

Disponibilité

Dreamweaver 3.

Description

Enregistre le jeu de cadres spécifié ou, si ce dernier n'a pas encore été enregistré, ouvre la boîte de dialogue Enregistrer sous.

Arguments

documentObject

- L'argument *documentObject* est l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canSaveFrameset()*, page 590.

dreamweaver.saveFramesetAs()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Enregistrer sous correspondant au fichier de jeu de cadres comprenant le DOM spécifié.

Arguments

documentObject

- L'argument *documentObject* est l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canSaveFramesetAs()*, page 590.

Fonctions globales relatives aux documents

Ces fonctions agissent sur l'ensemble d'un document. Elles permettent d'effectuer des vérifications orthographiques, de vérifier les navigateurs cibles, de définir les propriétés des pages et de déterminer les références d'objet correctes des éléments du document.

dom.checkSpelling()

Disponibilité

Dreamweaver 3.

Description

Vérifie l'orthographe sur l'ensemble du document (en ouvrant la boîte de dialogue Vérifier l'orthographe, si nécessaire) et prévient l'utilisateur lorsque la vérification est terminée.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.checkTargetBrowsers()

Disponibilité

Dreamweaver 3.

Description

Vérifie le navigateur cible du document. Pour vérifier le navigateur cible d'un dossier ou d'un groupe de fichiers, voir *site.checkTargetBrowsers()*, page 274.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.getParseMode()

Disponibilité

Dreamweaver MX 2004

Description

Permet d'obtenir le mode d'analyse du document. Ceci permet de contrôler la validation du document et de vérifier qu'il s'affiche au format HTML dans la fenêtre de document principale.

Arguments

Aucun.

Valeurs renvoyées

Une chaîne indiquant le mode d'analyse : `"html"`, `"xml"`, `"css"` ou `"text"`.

dom.hideInfoMessagePopup()

Disponibilité

Dreamweaver MX 2004.

Description

Masque le message qui peut s'afficher sous forme d'info-bulle dans la fenêtre de document.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Voir aussi

dom.showInfoMessagePopup(), page 333.

dom.runValidation()

Disponibilité

Dreamweaver MX, arguments facultatifs ajoutés à Dreamweaver MX 2004.

Description

Exécute le validateur sur un seul document spécifié (similaire à la fonction site.runValidation()). Le validateur vérifie que le langage du document est conforme à celui qui a été spécifié dans le doctype du document (comme, par exemple, HTML 4.0 ou HTML 3.2) et à celui qui a été spécifié par le modèle de serveur (comme, par exemple, ColdFusion ou ASP). Si aucun doctype n'a été défini pour le document, le validateur utilise le paramètre de langage spécifié dans la section Validateur de la boîte de dialogue Préférences.

Arguments

{controlString}, *{bOpenResultsWindow}*, *{bShowInfoMessage}*

- L'argument *controlString* (facultatif) peut avoir l'une des quatre valeurs suivantes : Chaîne vide, `"xml"`, `"auto-explicit"` ou `"auto-implicit"`.
  - Si la valeur de l'argument est une chaîne vide, le validateur effectue une validation par défaut. Si sa valeur est `"xml"`, il valide le document en tant que XML.
  - Si sa valeur est `"auto-explicit"` ou `"auto-implicit"`, Dreamweaver effectue une validation automatique (ou validation *en ligne*). Plutôt que d'être affichées dans la fenêtre de résultats de la validation (voir *dom.source.getValidationErrorsForOffset()*, page 552 et *dom.getAutoValidationCount()*, page 544), les erreurs sont soulignées dans le mode Code.
  - Si la valeur de l'argument *controlString* est `"auto-explicit"`, l'utilisateur est invité à enregistrer un document non enregistré avant la validation.
  - Si la valeur de l'argument *controlString* est `"auto-implicit"` et que la validation échoue, l'utilisateur n'est pas prévenu que le document en cours n'a pas été enregistré.

  REMARQUE La validation automatique (définie par la valeur `"auto-explicit"` ou `"auto-implicit"` de l'argument *controlString*) n'est disponible que pour la vérification du navigateur cible.

- L'argument *bOpenResultsWindow* (facultatif) doit avoir une valeur booléenne : pour ouvrir la fenêtre de résultats de la validation, sa valeur doit être `true` ; sinon, sa valeur est false. La valeur par défaut est `true`.

- L'argument *bShowInfoMessage* n'est utilisé que si la valeur de l'argument *controlString* est "auto-explicit" ou "auto-implicit". L'argument *bShowInfoMessage* doit avoir une valeur booléenne : si sa valeur est true, un message d'informations s'affiche sous l'élément de barre d'outils DW_ValidatorErrors, indiquant le nombre d'erreurs détectées ; si sa valeur est false, rien ne s'affiche. La valeur par défaut est false.

Valeurs renvoyées

Objet fenêtre des résultats de la validation.

Exemple

Dans l'exemple suivant, une validation régulière est effectuée lorsque l'utilisateur sélectionne Fichier > Vérifier la page > Valider le marqueur (ou Valider le document actuel dans le panneau Validation) :

```
dw.getDocumentDOM().runValidation('');
```

Dans l'exemple suivant, l'utilisateur est invité à enregistrer le document non enregistré, puis une validation automatique est effectuée. La fenêtre de résultats de la validation ne s'affiche pas et le nombre d'erreurs dans le document s'affiche sur la barre d'outils du document pour DW_ValidatorErrors :

```
dw.getDocumentDOM().runValidation('auto-explicit', false, true);
```

Dans l'exemple suivant, l'utilisateur n'est pas invité à enregistrer le document. Si le document n'a pas été enregistré, la validation ne peut pas démarrer. Si le document a été enregistré, Dreamweaver effectue une validation automatique. La fenêtre de résultats de la validation ne s'affiche pas et le nombre d'erreurs dans le document n'apparaît pas sur la barre d'outils du document :

```
dw.getDocumentDOM().runValidation('auto-implicit', false);
```

dom.showInfoMessagePopup()

Disponibilité

Dreamweaver MX 2004.

Description

Affiche un message sous forme d'info-bulle dans la fenêtre de document ou sous un élément de barre d'outils.

Arguments

location, message, timeout

- L'argument *location* peut être une chaîne indiquant un élément de barre d'outils, une chaîne vide ou l'un des mots-clés suivants : "top", "topright", "right", "bottomright", "bottom", "bottomleft", "left" ou "topleft". L'info-bulle s'affiche près du bord ou de l'angle spécifié et est centrée. Si la valeur de l'argument est une chaîne vide, l'info-bulle est centrée dans le document. Pour spécifier un élément de barre d'outils, indiquez "toolbar:*toolbarID*:*itemID*". L'ID de barre d'outils et l'ID d'élément de barre d'outils doivent correspondre aux ID du fichier toolbars.xml.
- L'argument *message* est une chaîne contenant le message.
- L'argument *timeout* est un nombre indiquant le temps d'affichage du message en millisecondes. Si sa valeur est 0 (valeur par défaut), le message reste affiché pendant une durée illimitée. Le message disparaît automatiquement lorsque l'utilisateur clique dessus, passe à un autre document ou lorsque la durée d'affichage spécifiée arrive à expiration.

Valeurs renvoyées

Aucune.

Exemple

Dans l'exemple ci-dessous, deux messages s'affichent sous forme d'info-bulles. La première ligne de code affiche le message "Ce message est au centre" au centre du document. Le second appel de la fonction showInfoMessagePopup() affiche le message "N'oubliez pas le titre de la fenêtre" pour la zone de texte Titre (ID : DW_SetTitle) sur la barre d'outils dont l'ID est DW_Toolbar_Main.

```
dw.getDocumentDOM.showInfoMessagePopup('', 'This message is in the center',
  5000);
dw.getDocumentDOM.showInfoMessagePopup('toolbar:DW_Toolbar_Main:DW_SetTitle
  ', 'Don't forget the title for the window', 5000);
```

Voir aussi

dom.hideInfoMessagePopup(), page 331.

dom.showPagePropertiesDialog()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Propriétés de la page.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.doURLDecoding()

Disponibilité

Dreamweaver MX.

Description

Utilise le mécanisme de décodage d'URL interne de Dreamweaver pour décoder des caractères spéciaux et des symboles dans les chaînes d'URL. Par exemple, cette fonction interprète `%20` comme un caractère d'espacement et le nom `&quot` comme des guillemets ".

Arguments

`inStr`

- L'argument `inStr` est la chaîne à décoder.

Valeurs renvoyées

Chaîne contenant l'URL décodée.

Exemple

L'exemple suivant appelle la fonction `dw.doURLDecoding()` afin de décoder les caractères spéciaux indiqués dans son argument et enregistre la chaîne obtenue dans `outStr` :

```
outStr = dreamweaver.doURLDecoding("http://maps.yahoo.com/py/
  ddResults.py?Pyt=Tmap&tarname=&tardesc=&newname=&newdesc=&newHash=&newTH
  ash=&newSts=&newTSts=&tlt=&tln=&slt=&sln=&newFL=Use+Address+Below&newadd
  r=2000+Shamrock+Rd&newcsz=Metroo+Park%2C+CA&newcountry=us&newTFL=Use+Add
  ress+Below&newtaddr=500+El+Camino&newtcsz=Santa+Clara%2C+CA&newtcountry=
  us&Submit=Get+Directions")
```

dreamweaver.getElementRef()

Disponibilité

Dreamweaver 2.

Description

Obtient la référence d'objet Netscape Navigator ou Internet Explorer pour un objet balise spécifique de l'arborescence DOM.

Arguments

NSorIE, tagObject

- L'argument *NSorIE* doit être "NS 4.0" ou "IE 4.0". En effet, le DOM et les règles relatives aux références imbriquées diffèrent dans Netscape Navigator 4.0 et Internet Explorer 4.0. Cet argument permet d'indiquer à quel navigateur doit correspondre la référence renvoyée.
- L'argument *tagObject* est un objet de balise dans l'arborescence DOM.

Valeurs renvoyées

Chaîne représentant une référence JavaScript valide à l'objet, telle que `document.layers['myLayer']`. La chaîne est soumise aux conditions suivantes :

- Dreamweaver renvoie des références Internet Explorer correctes pour les balises suivantes : A, AREA, APPLET, EMBED, DIV, SPAN, INPUT, SELECT, OPTION, TEXTAREA, OBJECT et IMG.
- Dreamweaver renvoie des références Netscape Navigator correctes pour les balises suivantes : A, AREA, APPLET, EMBED, LAYER, ILAYER, SELECT, OPTION, TEXTAREA, OBJECT et IMG et pour les balises DIV et SPAN à positionnement absolu. Pour les balises DIV et SPAN dont le positionnement n'est pas absolu, Dreamweaver renvoie "cannot reference <*tag*>".
- Dreamweaver ne renvoie pas de références pour les objets sans nom. Si un objet ne contient pas d'attribut NAME ou ID, Dreamweaver renvoie "unnamed <*tag*>". Si le navigateur ne prend pas en charge une référence par nom, Dreamweaver fait référence à l'objet à l'aide de son index (par exemple `document.myform.applets[3]`).
- Dreamweaver renvoie les références des objets nommés figurant dans des formulaires ou des calques sans nom (par exemple `document.forms[2].myCheckbox`).

dreamweaver.getObjectRefs() (déconseillée)

Disponibilité

Dreamweaver 1, déconseillée depuis Dreamweaver 3.

Description

Cette fonction recherche, dans les documents indiqués, les instances des balises spécifiées ou, si aucune balise n'est spécifiée, recherche toutes les balises du document et fournit des références aux balises en fonction du navigateur. Cette fonction revient à appeler la fonction `getElementsByTagName()`, puis la fonction `dreamweaver.getElementRef()` pour chaque balise de la `nodelist`.

Arguments

NSorIE, sourceDoc, {tag1}, {tag2},...{tagN}

- L'argument *NSorIE* doit être "NS 4.0" ou "IE 4.0". En effet, le DOM et les règles relatives aux références imbriquées diffèrent dans Netscape Navigator 4.0 et Internet Explorer 4.0. Cet argument permet d'indiquer à quel navigateur doit correspondre la référence renvoyée.
- L'argument *sourceDoc* doit être "document", "parent", "parent.frames[*number*]", "parent.frames['*frameName*']" ou une URL. document désigne le document actif contenant la sélection en cours. parent désigne le jeu de cadres parent (si le document sélectionné se trouve dans un cadre) ; parent.frames[*number*] et parent.frames['*frameName*'] désignent un document figurant dans un cadre spécifique du jeu de cadres contenant le document actif. Si l'argument est une URL relative, celle-ci est relative au fichier de l'extension.
- Le troisième argument et les arguments suivants, s'ils sont définis, correspondent aux noms de balises (par exemple "IMG", "FORM" ou "HR").

Valeurs renvoyées

Tableau de chaînes, chacun d'entre eux représentant une référence JavaScript valide à une instance nommée du type de balise demandé dans le document spécifié (par exemple "document.monCalque.document.monImage") pour le navigateur spécifié.

- Dreamweaver renvoie des références Internet Explorer correctes pour les balises suivantes : A, AREA, APPLET, EMBED, DIV, SPAN, INPUT, SELECT, OPTION, TEXTAREA, OBJECT et IMG.
- Dreamweaver renvoie des références Netscape Navigator correctes pour les balises suivantes : A, AREA, APPLET, EMBED, LAYER, ILAYER, SELECT, OPTION, TEXTAREA, OBJECT et IMG et pour les balises DIV et SPAN à positionnement absolu. Pour les balises DIV et SPAN dont le positionnement n'est pas absolu, Dreamweaver renvoie "cannot reference <*tag*>".
- Dreamweaver ne renvoie pas de références pour les objets sans nom. Si un objet ne contient pas d'attribut NAME ou ID, Dreamweaver renvoie "unnamed <*tag*>". Si le navigateur ne prend pas en charge une référence par nom, Dreamweaver fait référence à l'objet à l'aide de son index (par exemple document.myform.applets[3]).
- Dreamweaver ne renvoie pas de références pour les objets nommés figurant dans des formulaires ou des calques sans nom (par exemple document.forms[2].myCheckbox).

Lorsque la même liste d'arguments est transmise à la fonction getObjectTags(), les deux fonctions renvoient des tableaux de même longueur et de contenus identiques.

dreamweaver.getObjectTags() (déconseillée)

Disponibilité

Dreamweaver 1, déconseillée depuis Dreamweaver 3.

Description

Cette fonction recherche, dans le document indiqué, les instances des balises spécifiées ou, si aucune balise n'est spécifiée, toutes les balises du document. Cette fonction revient à appeler la fonction `getElementsByTagName()`, puis la fonction `outerHTML()` pour chaque balise de la `nodelist`.

Arguments

`sourceDoc, {tag1}, {tag2},...{tagN}`

- L'argument *`sourceDoc`* doit être `"document"`, `"parent"`, `"parent.frames[`*`number`*`]"`, `"parent.frames['`*`frameName`*`']"` ou une URL. `document` désigne le document actif contenant la sélection en cours. `parent` désigne le jeu de cadres parent (si le document sélectionné se trouve dans un cadre) ; `parent.frames[`*`number`*`]` et `parent.frames['`*`frameName`*`']` désignent un document figurant dans un cadre spécifique du jeu de cadres contenant le document actif. Si l'argument est une URL relative, celle-ci est relative au fichier de l'extension.
- Le second argument et les arguments suivants, s'il en existe, correspondent aux noms de balises (par exemple `"IMG"`, `"FORM"`, `"HR"`).

Valeurs renvoyées

Tableau de chaînes, chacun d'entre eux correspondant au code source d'une instance du type de balise demandé dans le document spécifié.

- Si l'un des arguments de *`tag`* est `LAYER`, la fonction renvoie toutes les balises `LAYER` et `ILAYER` et toutes les balises `DIV` et `SPAN` à positionnement absolu.
- Si l'un des arguments de *`tag`* est `INPUT`, la fonction renvoie tous les éléments du formulaire. Pour obtenir un type d'élément particulier du formulaire, définissez `INPUT/`*`TYPE`*, *`TYPE`* correspondant à `button`, `text`, `radio`, `checkbox`, `password`, `textarea`, `select`, `hidden`, `reset` ou `submit`.

Lorsque la même liste d'arguments est transmise à la fonction `getObjectRefs()`, les deux fonctions renvoient des tableaux de même longueur.

Exemple

Selon le contenu du document actif, `dreamweaver.getObjectTags("document", "IMG")` pourrait renvoyer un tableau contenant les éléments suivants :

- `"<IMG SRC="/images/dot.gif" WIDTH="10" HEIGHT="10" NAME="bullet">"`
- `"<IMG SRC="header.gif" WIDTH="400" HEIGHT="32" NAME="header">"`
- `"<IMG SRC="971208_nj.jpg" WIDTH="119" HEIGHT="119" NAME="headshot">"`

dreamweaver.getPreferenceInt()

Disponibilité

Dreamweaver MX.

Description

Vous permet d'extraire une préférence de nombre entier pour une extension donnée.

Arguments

section, key, default_value

- L'argument *section* est une chaîne qui spécifie la section des préférences contenant l'entrée.
- L'argument *key* est une chaîne qui spécifie l'entrée de la valeur à extraire.
- L'argument *default_value* est la valeur par défaut renvoyée par Dreamweaver si l'entrée est introuvable. Il doit s'agir d'un entier non signé compris entre 0 et 65 535 ou d'un entier signé compris entre -32 768 et 32 767.

Valeurs renvoyées

Valeur entière de l'entrée spécifiée dans la section ou valeur par défaut si la fonction ne trouve pas l'entrée. Envoie 0 si la valeur de l'entrée spécifiée n'est pas un entier.

Exemple

L'exemple suivant renvoie la valeur du paramètre Distance d'accrochage de la section Mon extension des Préférences. Si la section Mon extension n'existe pas ou si le paramètre Distance d'accrochage n'est pas défini, la fonction renvoie la valeur par défaut spécifiée (0).

```
var snapDist; //valeur par défaut si l'entrée est introuvable
snapDist = dreamweaver.getPreferenceInt("My Extension", "Snap Distance",
  0);
```

dreamweaver.getPreferenceString()

Disponibilité

Dreamweaver MX.

REMARQUE	Pour accéder aux préférences des sites, vous devez posséder la version 7.0.1. Vérifiez `dw.appVersion` pour connaître la version correcte avant d'accéder aux informations relatives au site.

Description

Permet d'extraire une préférence de chaîne que vous avez stockée pour une extension.

Arguments

`section, key, default_value`

- L'argument *`section`* est une chaîne qui spécifie la section des préférences contenant l'entrée.
- L'argument *`key`* est une chaîne qui spécifie la valeur à extraire.
- L'argument *`default_value`* est la chaîne par défaut renvoyée par Dreamweaver si l'entrée est introuvable.

Valeurs renvoyées

Chaîne de la préférence demandée ou valeur par défaut si la chaîne est introuvable.

Exemple

L'exemple suivant renvoie la valeur du paramètre Editeur de texte de la section Mon extension des Préférences. Si la section Mon extension n'existe pas ou si le paramètre Editeur de texte n'est pas défini, la fonction renvoie la valeur par défaut spécifiée par la variable `txtEditor`.

```
var txtEditor = getExternalTextEditor(); //pour définir la valeur de
  l'éditeur de texte par défaut
txtEditor = dreamweaver.getPreferenceString("My Extension", "Text Editor",
  txtEditor);
```

dreamweaver.setPreferenceInt()

Disponibilité

Dreamweaver MX.

Description

Vous permet de définir une préférence de nombre entier pour une extension donnée. Ce paramètre est enregistré avec les préférences de Dreamweaver lorsque Dreamweaver n'est pas actif.

Arguments

section, *key*, *new_value*

- L'argument *section* est une chaîne qui spécifie la catégorie de préférences définissant l'option. Si la catégorie n'existe pas, Dreamweaver la crée.
- L'argument *key* est une chaîne qui spécifie l'option de catégorie définie par la fonction. Si l'option n'existe pas, Dreamweaver la crée.
- L'argument *new_value* est un entier qui spécifie la valeur de l'option de catégorie.

Valeurs renvoyées

Valeur `true` en cas de succès et `false` dans le cas contraire.

Exemple

L'exemple suivant définit l'entrée Distance d'accrochage en fonction de la valeur de la variable `snapDist` de la catégorie Mon extension dans les Préférences :

```
var snapDist = getSnapDistance();
if(snapDist > 0)
{
  dreamweaver.setPreferenceInt("My Extension", "Snap Distance", snapDist);
}
```

dreamweaver.setPreferenceString()

Disponibilité

Dreamweaver MX.

REMARQUE

Pour accéder aux préférences des sites, vous devez posséder la version 7.0.1. Vérifiez `dw.appVersion` pour connaître la version correcte avant d'accéder aux informations relatives au site.

Description

Permet d'écrire une préférence de chaîne pour une extension. Ce paramètre est enregistré avec les préférences de Dreamweaver lorsque Dreamweaver n'est pas actif.

Arguments

section, *key*, *new_value*

- L'argument *section* est une chaîne qui spécifie la catégorie de préférences définissant l'option. Si la catégorie n'existe pas, Dreamweaver la crée.
- L'argument *key* est une chaîne qui spécifie l'option de catégorie définie par les fonctions. Si l'option de catégorie n'existe pas, Dreamweaver la crée.
- L'argument *new_value* est une chaîne qui spécifie la valeur de l'option de catégorie.

Valeurs renvoyées

Valeur `true` en cas de succès et `false` dans le cas contraire.

Exemple

```
var txtEditor = getExternalTextEditor();
dreamweaver.setPreferenceString("My Extension", "Text Editor", txtEditor);
```

dreamweaver.showTargetBrowsersDialog()

Disponibilité

Dreamweaver MX 2004.

Description

Ouvre la boîte de dialogue Navigateurs cibles. Cette boîte de dialogue permet à l'utilisateur de spécifier les versions des navigateurs utilisées par la fonction Vérification du navigateur cible pour contrôler les problèmes de compatibilité du navigateur sur la page actuelle.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Fonctions relatives aux chemins

Ces fonctions permettent d'obtenir et de manipuler les chemins d'accès aux différents fichiers et dossiers enregistrés sur le disque dur d'un utilisateur. Elles permettent, entre autres, de déterminer le chemin d'accès à la racine du site dans lequel réside le document actif et de convertir des chemins relatifs en URL absolues.

dreamweaver.getConfigurationPath()

Disponibilité

Dreamweaver 2.

Description

Obtient le chemin d'accès au dossier Configuration de Dreamweaver, exprimé sous la forme d'une URL de type file://.

Pour savoir comment Dreamweaver accède aux dossiers Configuration sur une plate-forme multiutilisateur, voir Extensions C dans l'aide Extension de Dreamweaver.

Arguments

Aucun.

Valeurs renvoyées

Chemin d'accès aux configurations de l'application.

Exemple

La fonction ci-dessous est utile pour faire référence à d'autres fichiers d'extension stockés dans le sous-dossier Configuration du dossier de l'application Dreamweaver :

```
var sortCmd = dreamweaver.getConfigurationPath() + ¬
"/Commands/Sort Table.htm"
var sortDOM = dreamweaver.getDocumentDOM(sortCmd);
```

dreamweaver.getDocumentPath()

Disponibilité

Dreamweaver 1.2.

Description

Obtient le chemin d'accès du document défini, exprimé sous la forme d'une URL de type file://. Cette fonction revient à appeler la fonction `dreamweaver.getDocumentDOM()` et à lire la propriété `URL` de la valeur renvoyée.

Arguments

`sourceDoc`

- La valeur de l'argument *sourceDoc* doit être "document", "parent", "parent.frames[*number*]" ou "parent.frames['*frameName*']". document désigne le document actif contenant la sélection en cours. parent désigne le jeu de cadres parent (si le document sélectionné se trouve dans un cadre) ; "parent.frames[*number*]" et "parent.frames['*frameName*']" désignent un document figurant dans un cadre spécifique du jeu de cadres contenant le document actif.

Valeurs renvoyées

Soit une chaîne contenant l'URL du document défini si le fichier a été enregistré, soit une chaîne vide si le fichier n'a pas été enregistré.

dreamweaver.getSiteRoot()

Disponibilité

Dreamweaver 1.2.

Description

Obtient le dossier racine local (tel que défini dans la boîte de dialogue Définition du site) du site associé au document sélectionné, exprimé sous la forme d'une URL de type file://.

Arguments

Aucun.

Valeurs renvoyées

Soit une chaîne contenant l'URL du dossier racine local du site où le fichier a été enregistré, soit une chaîne vide si le fichier n'est associé à aucun site.

dreamweaver.getTempFolderPath()

Disponibilité

Dreamweaver MX.

Description

Permet d'obtenir le chemin d'accès complet à un dossier temporaire dans lequel vous pouvez enregistrer les fichiers temporaires ou transitoires. Cette fonction recherche un dossier Temp dans le dossier Configuration de Dreamweaver. Si le système est multiutilisateur, la recherche se fait dans le dossier Configuration de l'utilisateur. Si aucun dossier Temp n'a été trouvé, la fonction en crée un. Les fichiers partagés qui ne sont pas transitoires doivent être enregistrés dans le dossier Configuration/Shared.

Arguments

Aucun.

Valeurs renvoyées

Chemin d'accès au dossier, exprimé sous la forme d'une URL de type file://.

Exemple

La ligne de code ci-dessous renvoie le chemin d'accès complet au fichier spécifié. Contrairement aux autres fonctions Dreamweaver (telles que `dreamweaver.getSiteRoot()`), la fonction `dw.getTempFolderPath()` ne renvoie pas de barre oblique (/) à la fin du chemin :

```
var myTempfile = dw.getTempFolderPath() + "/myTempFile.txt";
```

dreamweaver.relativeToAbsoluteURL()

Disponibilité

Dreamweaver 2.

Description

Si on lui fournit une URL relative et un point de référence donnés (chemin d'accès au document actif ou racine du site), cette fonction convertit l'URL relative en URL absolue (file://).

Arguments

`docPath, siteRoot, relURL`

- L'argument *`docPath`* correspond au chemin d'accès à un document sur l'ordinateur de l'utilisateur (le document actif, par exemple), exprimé sous la forme d'une URL de type file://, ou à une chaîne vide si *`relURL`* est une URL relative à la racine.
- L'argument *`siteRoot`* est le chemin d'accès à la racine du site, exprimé sous la forme d'une URL de type file://, ou une chaîne vide si *`relURL`* est une URL relative à un document.
- L'argument *`relURL`* est l'URL à convertir.

Valeurs renvoyées

URL absolue. La valeur renvoyée est générée conformément aux principes suivants :

- Si *`relURL`* est une URL absolue, aucune conversion n'a lieu et la valeur renvoyée est identique à *`relURL`*.

- Si *relURL* est une URL relative à un document, la valeur renvoyée est une combinaison de *docPath* + *relURL*.
- Si *relURL* est une URL relative à la racine, la valeur renvoyée est une combinaison de *siteRoot* + *relURL*.

Fonctions relatives à la sélection

Ces fonctions permettent d'obtenir et de définir la sélection dans les documents ouverts. Pour savoir comment obtenir et définir la sélection dans le panneau Site, voir *Fonctions relatives aux sites*, page 264.

dom.getSelectedNode()

Disponibilité

Dreamweaver 3.

Description

Obtient le nœud sélectionné. Revient à appeler la fonction `dom.getSelection()` et à transmettre la valeur renvoyée à la fonction `dom.offsetsToNode()`.

Arguments

Aucun.

Valeurs renvoyées

Objet balise, texte ou commentaire contenant la série de caractères spécifiée.

dom.getSelection()

Disponibilité

Dreamweaver 3.

Description

Obtient la sélection, exprimée en décalages de caractères, dans le code source du document.

Arguments

[bAllowMultiple]

- L'argument facultatif *bAllowMultiple* est une valeur booléenne qui indique si la fonction doit renvoyer plusieurs décalages lorsque plusieurs calques, cellules de tableau ou zones réactives de carte graphique sont sélectionnés.

  Si cet argument n'est pas défini, il prend par défaut la valeur `false`.

Valeurs renvoyées

Pour les sélections simples, un tableau contenant deux nombres entiers. Le premier entier correspond au décalage de caractères à l'ouverture de la sélection. Le second correspond au décalage de caractères à la fermeture de la sélection. Si les deux valeurs sont identiques, la sélection en cours correspond à un point d'insertion.

Pour les sélections complexes (lorsque la sélection se compose de plusieurs cellules de tableau, calques ou zones réactives de carte graphique), tableau contenant $2n$ nombres entiers, où *n* représente le nombre d'éléments sélectionnés. Le premier entier de chaque paire correspond au décalage de caractères de l'ouverture de la sélection (balise d'ouverture `TD`, `DIV`, `SPAN`, `LAYER`, `ILAYER` ou `MAP` comprise) ; le second entier correspond au décalage de caractères à la fermeture de la sélection (balise de fermeture `TD`, `DIV`, `SPAN`, `LAYER`, `ILAYER` ou `MAP` comprise). Si plusieurs rangées d'un tableau sont sélectionnées, le décalage de chaque cellule de chaque rangée est renvoyé. La sélection n'inclut jamais les balises `TR`.

dom.nodeToOffsets()

Disponibilité

Dreamweaver 3.

Description

Obtient la position d'un nœud donné dans l'arborescence DOM, exprimée en décalages de caractères dans le code source du document. Fonction valide pour n'importe quel document sur un lecteur local.

Arguments

node

- L'argument *node* doit être une balise, un commentaire ou un texte correspondant à un nœud de l'arborescence renvoyée par la fonction `dreamweaver.getDocumentDOM()`.

Valeurs renvoyées

Tableau qui contient deux nombres entiers. Le premier entier correspond au décalage du début de la balise, du texte ou du commentaire. Le second correspond au décalage de la fin du nœud, par rapport au début du document HTML.

Exemple

L'exemple de code suivant sélectionne le premier objet image du document actif :

```
var theDOM = dw.getDocumentDOM();
var theImg = theDOM.images[0];
var offsets = theDom.nodeToOffsets(theImg);
theDom.setSelection(offsets[0], offsets[1]);
```

dom.offsetsToNode()

Disponibilité

Dreamweaver 3.

Description

Obtient l'objet de l'arborescence DOM qui contient la série entière de caractères située entre l'ouverture et la fermeture définies. Fonction valide pour n'importe quel document sur un lecteur local.

Arguments

offsetBegin, offsetEnd

- L'argument *offsetBegin* indique le décalage depuis le début du document jusqu'au début d'une série de caractères correspondant à un objet dans l'arborescence DOM.
- L'argument *offsetEnd* indique le décalage depuis le début du document jusqu'à la fin d'une série de caractères correspondant à un objet dans l'arborescence DOM.

Valeurs renvoyées

Objet balise, texte ou commentaire contenant la série de caractères spécifiée.

Exemple

Le code suivant affiche une alerte si la sélection est une image.

```
var offsets = dom.getSelection();
var theSelection = dreamweaver.offsetsToNode(offsets[0], ¬
offsets[1]);
if (theSelection.nodeType == Node.ELEMENT_NODE && ¬
theSelection.tagName == 'IMG'){
  alert('The current selection is an image.');
}
```

dom.selectAll()

Disponibilité

Dreamweaver 3.

Description

Effectue une opération Sélectionner tout.

REMARQUE	Dans la plupart des cas, cette fonction sélectionne le contenu entier du document actif. Dans certains cas toutefois (lorsque le point d'insertion se trouve dans un tableau, par exemple), elle ne sélectionne qu'une partie du document actif. Pour définir le document entier comme sélection, utilisez la fonction `dom.setSelection()`.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.setSelectedNode()

Disponibilité

Dreamweaver 3.

Description

Définit le nœud sélectionné. Revient à appeler la fonction `dom.nodeToOffsets()` et à transmettre la valeur renvoyée à la fonction `dom.setSelection()`.

Arguments

`node, {bSelectInside}, {bJumpToNode}`

- L'argument *`node`* est un nœud de texte, de commentaire ou d'élément du document.
- L'argument *`bSelectInside`* (facultatif) est une valeur booléenne qui indique s'il faut sélectionner la propriété `innerHTML` du nœud. Cet argument n'est pertinent que si *`node`* est un nœud d'élément et qu'il prend par défaut la valeur `false` lorsqu'il n'est pas défini.
- L'argument *`bJumpToNode`* (facultatif) est une valeur booléenne qui indique s'il faut, le cas échéant, faire défiler la fenêtre de document pour rendre la sélection visible. S'il n'est pas défini, cet argument prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

dom.setSelection()

Disponibilité

Dreamweaver 3.

Description

Définit le début et la fin de la sélection dans le document.

Arguments

offsetBegin, offsetEnd

- Ces arguments correspondent respectivement à l'ouverture et à la fermeture de la nouvelle sélection, exprimées en décalages de caractères dans le code source du document. Si les deux valeurs sont identiques, la nouvelle sélection correspond à un point d'insertion. Si la nouvelle sélection n'est pas une sélection HTML valide, elle inclut les caractères de la première sélection HTML valide. Par exemple, si *offsetBegin* et *offsetEnd* définissent `SRC="myImage.gif"` comme sélection dans `<IMG SRC="myImage.gif">`, la sélection est étendue pour inclure également la balise `IMG`.

Valeurs renvoyées

Aucune.

dreamweaver.getSelection() (déconseillée)

Disponibilité

Dreamweaver 2 ; déconseillée depuis Dreamweaver 3. Voir *dom.getSelection()*, page 346.

Description

Obtient la sélection en cours, exprimée en décalages d'octets dans le code source du document.

Arguments

Aucun.

Valeurs renvoyées

Tableau qui contient deux nombres entiers. Le premier entier représente le décalage d'octets du début de la sélection et le second correspond au décalage d'octets de la fin de la sélection. Si les deux valeurs sont identiques, la sélection en cours correspond à un point d'insertion.

dreamweaver.nodeExists()

Disponibilité

Dreamweaver 3.

Description

Détermine si la référence au nœud indiqué est toujours valable. Il arrive souvent, lors de la rédaction d'extensions, que vous fassiez référence à un nœud, puis que vous effectuiez une opération qui le supprime (comme définir la propriété `innerHTML` ou `outerHTML` de son parent). Cette fonction vous permet de confirmer que le nœud n'a pas été supprimé avant de commencer à faire référence à l'une de ses propriétés ou méthodes. Le nœud référencé ne doit pas nécessairement se trouver dans le document actif.

Arguments

`node`

- L'argument *`node`* correspond au nœud à vérifier.

Valeurs renvoyées

Valeur booléenne : `true` si le nœud existe et `false` dans le cas contraire.

Exemple

L'exemple ci-dessous permet d'obtenir le nœud en cours, de localiser un tableau, puis d'appeler la fonction `dw.nodeExists()` afin de vérifier si le nœud original existe encore :

```
function applyFormatToSelectedTable(){

  // pour obtenir la sélection en cours
  var selObj = dw.getDocumentDOM().getSelectedNode();

  alternateRows(dwscripts.findDOMObject("presetNames").selectedIndex,
    findTable());

// pour restaurer la sélection d'origine, si elle existe encore ; dans le
  cas contraire, sélectionne simplement
// le tableau.

  var selArr;
```

```
  if (dw.nodeExists(selObj))
    selArr = dom.nodeToOffsets(selObj);
  else
    selArr = dom.nodeToOffsets(findTable());

  dom.setSelection(selArr[0],selArr[1]);
}
```

dreamweaver.nodeToOffsets() (déconseillée)

Disponibilité

Dreamweaver 2, déconseillée depuis Dreamweaver 3 : utiliser à la place *dom.nodeToOffsets()*, page 347.

Description

Obtient la position d'un nœud dans l'arborescence DOM, exprimée en décalages d'octets dans le code source du document.

Arguments

node

- L'argument *node* doit être une balise, un commentaire ou un texte correspondant à un nœud de l'arborescence renvoyée par la fonction `dreamweaver.getDocumentDOM()`.

Valeurs renvoyées

Tableau qui contient deux nombres entiers. Le premier entier représente le décalage d'octets de l'ouverture de la balise, du texte ou du commentaire et le second correspond au décalage d'octets de la fermeture du nœud.

dreamweaver.offsetsToNode() (déconseillée)

Disponibilité

Dreamweaver 2, déconseillée depuis Dreamweaver 3 : utiliser à la place *dom.offsetsToNode()*, page 348.

Description

Obtient l'objet de l'arborescence DOM qui contient la série entière de caractères située entre l'ouverture et la fermeture définies.

Arguments

offsetBegin, offsetEnd

- Ces arguments correspondent respectivement à l'ouverture et à la fermeture d'une série de caractères, exprimées en décalages d'octets dans le code source du document.

Valeurs renvoyées

Objet balise, texte ou commentaire contenant la série de caractères spécifiée.

dreamweaver.selectAll()

Disponibilité

Dreamweaver 3.

Description

Effectue une opération Sélectionner tout dans la fenêtre de document active, dans le panneau Site ou, sur Macintosh, dans le champ de texte actif d'une boîte de dialogue ou d'un panneau flottant.

REMARQUE	Si l'opération est effectuée dans le document actif, elle sélectionne dans la plupart des cas le contenu entier du document. Dans certains cas toutefois (lorsque le point d'insertion se trouve dans un tableau, par exemple), elle ne sélectionne qu'une partie du document. Pour définir le document entier comme sélection, utilisez la fonction `dom.setSelection()`.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canSelectAll()*, page 591.

dreamweaver.setSelection() (déconseillée)

Disponibilité

Dreamweaver 2, déconseillée depuis Dreamweaver 3 : utiliser à la place *`dom.setSelection()`*, page 350.

Description

Définit la sélection dans le document actif. Cette fonction ne peut déplacer la sélection qu'à l'intérieur du document actif. Elle ne peut pas rendre un autre document actif.

Arguments

offsetBegin, *offsetEnd*

- Ces arguments correspondent respectivement à l'ouverture et à la fermeture de la nouvelle sélection, exprimées en décalages d'octets dans le code source du document. Si les deux valeurs sont identiques, la nouvelle sélection correspond à un point d'insertion. Si la nouvelle sélection n'est pas une sélection HTML valide, elle inclut les caractères de la première sélection HTML valide. Par exemple, si *offsetBegin* et *offsetEnd* définissent `SRC="myImage.gif"` comme sélection dans `<IMG SRC="myImage.gif">`, la sélection est étendue pour inclure également la balise `IMG`.

Valeurs renvoyées

Aucune.

Fonctions de manipulation de chaînes

Ces fonctions vous permettent d'obtenir des informations sur une chaîne et de convertir une chaîne Latin1 en code national sur la plate-forme de l'utilisateur et vice-versa.

dreamweaver.doURLEncoding()

Disponibilité

Dreamweaver 1.

Description

Prend une chaîne et renvoie une chaîne convertie en URL en remplaçant tous les espaces et caractères spéciaux par les entités spécifiées.

Arguments

stringToConvert

- L'argument *stringToConvert* est une chaîne qui contient l'URL non codée qui est codée par la fonction.

Valeurs renvoyées

Une chaîne convertie en format d'URL.

Exemple

L'exemple suivant indique la `valeur URL` pour `"My URL-encoded string"` :

```
var URL = dw.doURLEncoding(theURL.value);
returns "My%20URL-encoded%20string"
```

dreamweaver.getTokens()

Disponibilité

Dreamweaver 1.

Description

Accepte une chaîne et la divise en expressions.

Arguments

searchString, separatorCharacters

- L'argument *searchString* est la chaîne à diviser en expressions.
- L'argument *separatorCharacters* est le ou les caractères indiquant la fin d'une expression. Les séparateurs figurant dans des chaînes entre guillemets sont ignorés. Tous les espaces éventuellement contenus dans `separatorCharacters` (les tabulations, par exemple) sont traités comme des séparateurs, comme si vous les aviez définis explicitement. Deux espaces consécutifs ou plus sont traités comme un seul séparateur.

Valeurs renvoyées

Tableau d'expressions.

Exemple

L'appel de la fonction `dw.getTokens()` ci-dessous renvoie les expressions indiquées à la suite :

```
dreamweaver.getTokens('foo("my arg1", 34)', '(),')
```

- `foo`
- `"my arg 1"`
- `34`

dreamweaver.latin1ToNative()

Disponibilité

Dreamweaver 2.

Description

Convertit une chaîne Latin 1 en code national sur la machine de l'utilisateur. Cette fonction permet d'afficher l'interface utilisateur d'un fichier d'extension dans une autre langue.

REMARQUE	Sous Windows, cette fonction n'a aucun effet car le code Windows repose sur Latin 1.

Arguments

stringToConvert

- L'argument *stringToConvert* est la chaîne à convertir de Latin 1 en code national sur la machine de l'utilisateur.

Valeurs renvoyées

Chaîne convertie.

dreamweaver.nativeToLatin1()

Disponibilité

Dreamweaver 2.

Description

Convertit une chaîne de code national en code Latin 1.

REMARQUE	Sous Windows, cette fonction n'a aucun effet car le code Windows repose sur Latin 1.

Arguments

stringToConvert

- L'argument *stringToConvert* est la chaîne à convertir du code national en code Latin 1.

Valeurs renvoyées

Chaîne convertie.

dreamweaver.scanSourceString()

Disponibilité

Dreamweaver UltraDev 1.

Description

Analyse une chaîne de code HTML et recherche les balises, les attributs, les directives et le texte. Pour chaque balise, attribut, directive et texte trouvé, la fonction `scanSourceString()` appelle une fonction de rappel fournie par l'appelant. Dreamweaver prend en charge les fonctions de rappel suivantes :

- `openTagBegin()`
- `openTagEnd()`
- `closeTagBegin()`
- `closeTagEnd()`
- `directive()`
- `attribute()`
- `text()`

Dreamweaver appelle les sept fonctions de rappel dans les cas suivants :

1. Dreamweaver appelle `openTagBegin()` pour chaque balise d'ouverture (par exemple `<font>`, et non `</font>`) et chaque balise vide (par exemple, `<img>` ou `<hr>`). La fonction `openTagBegin()` accepte deux arguments : le nom de la balise (par exemple, `font` ou `img`) et le décalage du document, qui est le nombre d'octets contenus dans le document avant le début de la balise. La fonction renvoie la valeur `true` si l'analyse doit continuer et `false` si elle doit s'arrêter.
2. Une fois `openTagBegin()` exécutée, Dreamweaver appelle `attribute()` pour chaque attribut HTML. La fonction `attribute()` accepte deux arguments : une chaîne contenant le nom de l'attribut (par exemple, `"color"` ou `"src"`) et une chaîne contenant la valeur de l'attribut (par exemple `"#000000"` ou `"foo.gif"`). La fonction `attribute()` renvoie une valeur booléenne indiquant si l'analyse doit continuer.
3. Une fois que tous les attributs de la balise ont été analysés, Dreamweaver appelle `openTagEnd()`. La fonction `openTagEnd()` accepte un argument : le décalage du document, qui est le nombre d'octets contenus dans le document avant la fin de la balise d'ouverture. Elle renvoie une valeur booléenne qui indique si l'analyse doit continuer.

4. Dreamweaver appelle `closeTagBegin()` pour chaque balise de fermeture (par exemple, `</font>`). Cette fonction accepte deux arguments : le nom de la balise à fermer (par exemple, `"font"`) et le décalage du document, qui correspond au nombre d'octets contenus dans le document avant le début de la balise de fermeture. La fonction renvoie une valeur booléenne qui indique si l'analyse doit continuer.
5. Une fois `closeTagBegin()` terminée, Dreamweaver appelle la fonction `closeTagEnd()`. La fonction `closeTagEnd()` accepte un argument : le décalage du document, qui correspond au nombre d'octets contenus dans le document avant la fin de la balise de fermeture. Elle renvoie une valeur booléenne qui indique si l'analyse doit continuer.
6. Dreamweaver appelle la fonction `directive()` pour chaque commentaire HTML, script ASP, script JSP ou script PHP. La fonction `directive()` accepte deux arguments : une chaîne contenant la directive et le décalage du document, qui correspond au nombre d'octets contenus dans le document avant la fin de la balise de fermeture. La fonction renvoie une valeur booléenne qui indique si l'analyse doit continuer.
7. Dreamweaver appelle la fonction `text()` pour chaque portion de texte du document, c'est-à-dire tout ce qui n'est pas une balise ni une directive. Les portions de texte incluent le texte qui n'est pas visible pour l'utilisateur, par exemple le texte contenu à l'intérieur d'une balise `<title>` ou `<option>`. La fonction `text()` accepte deux arguments : une chaîne contenant le texte et le décalage du document, qui correspond au nombre d'octets contenus dans le document avant la fermeture de la balise de fermeture. La fonction `text()` renvoie une valeur booléenne qui indique si l'analyse doit continuer.

Arguments

`HTMLstr, parserCallbackObj`

- L'argument *`HTMLstr`* est une chaîne qui contient un code.
- L'argument *`parserCallbackObj`* est un objet JavaScript qui a une ou plusieurs des méthodes suivantes : `openTagBegin()`, `openTagEnd()`, `closeTagBegin()`, `closeTagEnd()`, `directive()`, `attribute()` et `text()`. Pour de meilleurs résultats, *`parserCallbackObj`* doit être une bibliothèque partagée définie à l'aide de l'interface Extension C. Vous obtiendrez également de meilleures performances si la fonction *`parserCallbackObj`* définit uniquement les fonctions de rappel dont elle a besoin.

Valeurs renvoyées

Valeur booléenne : `true` si l'opération réussit et `false` dans le cas contraire.

Exemple

La séquence d'étapes suivante offre un exemple d'utilisation de la fonction `dreamweaver.scanSourceString()` :

1. Créez une implémentation pour une ou plusieurs des sept fonctions de rappel.
2. Ecrivez un script qui appelle la fonction `dreamweaver.scanSourceString()`.
3. La fonction `dreamweaver.scanSourceString()` transmet une chaîne contenant le code HTML et les pointeurs des fonctions de rappel que vous avez écrits. Par exemple, supposons que la chaîne HTML soit `"<font size=2>bonjour</font>"`.
4. Dreamweaver analyse la chaîne et détermine si elle contient une balise de police. Dreamweaver appelle ensuite les fonctions de rappel dans l'ordre suivant :
   - Fonction `openTagBegin()`
   - Fonction `attribute()` (pour l'attribut de taille)
   - Fonction `openTagEnd()`
   - Fonction `text()` (pour la chaîne « `bonjour` »)
   - Fonctions `closeTagBegin()` et `closeTagEnd()`

Fonctions relatives à la traduction

Ces fonctions permettent d'agir directement sur les traducteurs de données ou sur les résultats de la traduction. Elles permettent d'exécuter ou d'obtenir des informations sur un traducteur, de modifier le contenu d'une région verrouillée et de stipuler que le code traduit doit être utilisé lors de l'obtention et de la définition de décalages de sélection.

dom.runTranslator()

Disponibilité

Dreamweaver 3.

Description

Cette fonction exécute le traducteur spécifié sur le document. Cette fonction n'est valide que pour le document actif.

Arguments

`translatorName`

- L'argument *`translatorName`* est le nom d'un traducteur tel qu'il apparaît dans les préférences de traduction.

Valeurs renvoyées

Aucune.

dreamweaver.editLockedRegions()

Disponibilité

Dreamweaver 2.

Description

Selon la valeur de l'argument, autorise ou non la modification des régions verrouillées. Par défaut, les régions verrouillées ne peuvent pas être modifiées. Si vous tentez de modifier une région verrouillée avant de la rendre modifiable avec cette fonction, Dreamweaver émet un bip et interdit la modification.

REMARQUE	La modification de régions verrouillées peut avoir des conséquences inattendues sur les éléments de bibliothèque et les modèles. N'utilisez cette fonction qu'avec les traducteurs de données.

Arguments

`bAllowEdits`

- L'argument *`bAllowEdits`* doit avoir une valeur booléenne : `true` indique que les modifications sont autorisées ; `false` indique le contraire. Dreamweaver restaure automatiquement l'état par défaut (non modifiable) des régions verrouillées lorsque l'exécution du script qui appelle la fonction prend fin.

Valeurs renvoyées

Aucune.

dreamweaver.getTranslatorList()

Disponibilité

Dreamweaver 3.

Description

Cette fonction obtient la liste des traducteurs de données installés.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes, chacune représentant le nom d'un traducteur tel qu'il apparaît dans les préférences de traduction.

dreamweaver.useTranslatedSource()

Disponibilité

Dreamweaver 2.

Description

Cette fonction indique les valeurs renvoyées par `dom.nodeToOffsets()` et `dom.getSelection()`. Elles sont utilisées par `dom.offsetsToNode()` et `dom.setSelection()` et doivent être décalées dans le code source converti (code HTML contenu dans l'arborescence DOM après l'exécution du traducteur) et non dans le code source non converti.

REMARQUE	Cette fonction ne s'applique qu'aux fichiers de l'inspecteur de propriétés.

Arguments

`bUseTranslatedSource`

- L'argument *`bAllowEdits`* doit avoir une valeur booléenne : `true` si la fonction utilise des décalages dans le code source traduit ; `false` si la fonction utilise le code source non traduit.

  La valeur par défaut de l'argument est `false`. Lorsque le script appelant la fonction `dw.useTranslatedSource()` prend fin, Dreamweaver utilise automatiquement le code source non converti pour les appels suivants de `dw.getSelection()`, `dw.setSelection()`, `dw.nodeToOffsets()` et `dw.offsetsToNode()`, sauf si la fonction `dw.useTranslatedSource()` est appelée explicitement avec l'argument `false` avant la fin de l'exécution du script.

Valeurs renvoyées

Aucune.

Fonctions XSLT

Les fonctions XSLT permettent d'agir sur les fichiers XML. Elles permettent d'obtenir des informations sur les documents XML, y compris l'arborescence de schémas ou la référence à un document XML, et invitent l'utilisateur à indiquer le document XML associé au document XSLT actif.

MMXSLT.getXMLSchema()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie l'arborescence de schémas du fichier XML spécifié.

Arguments

schemaURI, {bRefresh}

- L'argument obligatoire *schemaURI* est une chaîne correspondant à une référence à un fichier XML local ou distant.
- L'argument facultatif *bRefresh* est une valeur booléenne : `true` actualise le schéma ; `false` renvoie la copie du schéma à partir du cache des schémas XML. La valeur par défaut est `false`.

Valeurs renvoyées

Chaîne contenant l'arborescence de schémas XML.

Exemple

L'exemple suivant obtient l'arborescence de schémas à partir du cache de schémas XML pour le fichier menus.xml :

```
var theSchema = MMXSLT.getXMLSchema("file:///c:/Program ¬
Files/Macromedia/dreamweaver/configuration/Menus/menus.xml"");
```

MMXSLT.getXMLSourceURI()

Disponibilité

Dreamweaver 8.

Description

Cette fonction obtient une référence au document source XML associé au document XSLT actif.

Arguments

xsltfileURI, {bUseTempForRemote}

- L'argument *xsltfileURI* est une chaîne correspondant à l'URI du fichier local qui pointe vers l'emplacement du fichier XSL.

- L'argument facultatif *bUseTempForRemote* est une valeur booléenne : `true` renvoie une référence au fichier temporaire XML (par exemple, `file:///C:/Documents and Settings/username/Local Settings/Temporary Internet Files/Content.IE5/GTSLQ9KZ/rss[1].xml`) qui est téléchargé lorsque le fichier XML original est distant (par exemple, `http://myHost/rssfeed.xml`) ; `false` renvoie une référence absolue.

Valeurs renvoyées

Une chaîne contenant la référence au document source XML associé au document XSLT actif. Si la référence à la source XML est une référence distante, la fonction renvoie le chemin d'accès à l'emplacement temporaire du fichier téléchargé.

Exemple

L'exemple suivant obtient la référence au document source XML associé à c:\myxslt\myxsltdocument.xsl :

```
var theXMLSource = MMXSLT.getXMLSourceURI("file:///c:/myxslt/
  myxsltdocument.xsl");
```

MMXSLT.launchXMLSourceDialog()

Disponibilité

Dreamweaver 8.

Description

Cette fonction invite l'utilisateur à indiquer le document source XML associé au document XSLT actif. L'utilisateur peut choisir une référence soit locale soit distante à un document XML.

Arguments

{xsltfileURI, bUseTempForRemote, bAddSchemaReference}

- L'argument *xsltfileURI* est facultatif. Il s'agit d'une chaîne correspondant à l'URI du fichier local qui pointe vers l'emplacement du fichier XSL. Si cet argument n'est pas défini, il prend comme valeur par défaut le document actuellement ouvert.
- L'argument facultatif *bUseTempForRemote* est une valeur booléenne : `true` renvoie une référence au fichier temporaire XML (par exemple, `file:///C:/Documents and Settings/username/Local Settings/Temporary Internet Files/Content.IE5/GTSLQ9KZ/rss[1].xml`) qui est téléchargé lorsque le fichier XML original est distant (par exemple, `http://myHost/rssfeed.xml`) ; `false` renvoie une référence absolue.

- L'argument *bAddSchemaReference* est facultatif. Il ajoute une référence dans le document actif qui pointe vers l'URI de la source XML spécifiée dans la boîte de dialogue de la source XML. Si cet argument n'est pas défini, il prend comme valeur par défaut le document actuellement ouvert.

Valeurs renvoyées

Une chaîne contenant la référence au document source XML associé au document XSLT actif. Si la référence à la source XML est une référence distante, la fonction renvoie le chemin d'accès à l'emplacement temporaire du fichier téléchargé.

Exemple

L'exemple suivant ouvre la boîte de dialogue Document source XML sans indiquer aucune valeur :

```
MMXSLT.launchXMLSourceDialog()
```

CHAPITRE 14

Contenu de page

14

Les fonctions relatives au contenu de page permettent d'effectuer des opérations qui affectent le contenu d'une page Web. Ces opérations comprennent la manipulation d'actifs dans le panneau Actifs, l'ajout de comportements, le coupage et le collage d'éléments provenant du Presse-papiers, l'application d'un modèle ou l'insertion d'un fragment de code.

Fonctions du panneau Actifs

Ces fonctions, programmées dans l'API comme panneau d'actifs, vous permettent de gérer et d'utiliser les éléments contenus dans le panneau Actifs (modèles, bibliothèques, images, contenu Shockwave et Flash de Macromedia, URL, couleurs et scripts).

dreamweaver.assetPalette.addToFavoritesFrom Document()

Disponibilité

Dreamweaver 4.

Description

Ajoute l'élément sélectionné dans la fenêtre de document à la liste des favoris. Cette fonction prend uniquement en charge les images, les fichiers Shockwave, les fichiers Flash, les couleurs de police et les URL.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.addToFavoritesFromSiteAssets()

Disponibilité

Dreamweaver 4.

Description

Ajoute les éléments sélectionnés dans la liste des sites à la liste des favoris et donne à chaque élément un surnom. Cette fonction ne supprime pas les éléments de la liste des sites.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.addToFavoritesFromSiteWindow()

Disponibilité

Dreamweaver 4.

Description

Ajoute les éléments sélectionnés dans le panneau Site ou dans la carte du site à la liste des favoris. Cette fonction prend uniquement en charge les images, les animations, les scripts, les fichiers Shockwave, les fichiers Flash et les URL (dans le cas de la carte du site). Si d'autres dossiers ou fichiers sont sélectionnés, ils sont ignorés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.copyToSite()

Disponibilité

Dreamweaver 4.

Description

Copie les éléments sélectionnés dans un autre site et les place dans la liste des favoris de ce site. Si ces éléments sont des fichiers (autres que des couleurs ou des URL), le fichier réel est copié dans ce site.

Arguments

`targetSite`

- L'argument *`targetSite`* est le nom du site cible renvoyé par l'appel `site.getSites()`.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.edit()

Disponibilité

Dreamweaver 4.

Description

Modifie les éléments sélectionnés à l'aide de l'éditeur externe principal ou de la commande d'édition personnalisée. Pour les couleurs, le sélecteur de couleur s'affiche. S'il s'agit d'URL, une boîte de dialogue invite l'utilisateur à entrer une URL et un surnom. Cette fonction n'est pas disponible pour la liste de couleurs et les URL du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.assetPalette.canEdit()*, page 581.

dreamweaver.assetPalette.getSelectedCategory()

Disponibilité

Dreamweaver 4.

Description

Renvoie la catégorie sélectionnée.

Arguments

Aucun.

Valeurs renvoyées

La catégorie sélectionnée, qui peut être l'une des catégories suivantes : `"templates"`, `"library"`, `"images"`, `"movies"`, `"shockwave"`, `"flash"`, `"scripts"`, `"colors"` ou `"urls"`.

dreamweaver.assetPalette.getSelectedItems()

Disponibilité

Dreamweaver 4.

Description

Renvoie un tableau des éléments sélectionnés dans le panneau Actifs, soit dans la liste des sites, soit dans celle des favoris.

Arguments

Aucun.

Valeurs renvoyées

Tableau de trois chaînes pour chaque élément sélectionné :

- La chaîne *`name`* est le nom, le nom de fichier ou le surnom qui s'affiche dans le panneau Actifs.
- La chaîne `value` est le chemin entier, l'URL complète ou la valeur chromatique, selon l'élément sélectionné.

- La chaîne *type* est soit `"folder"`, soit l'une des catégories suivantes : `"templates"`, `"library"`, `"images"`, `"movies"`, `"shockwave"`, `"flash"`, `"scripts"`, `"colors"` ou `"urls"`.

REMARQUE	Si aucun élément n'est sélectionné dans le panneau Actifs, cette fonction renvoie un tableau contenant une seule chaîne vide.

Exemple

Si « URL » est la catégorie et que le dossier « MyFolderName » et l'URL « MyFavoriteURL » sont sélectionnés dans la liste des favoris, la fonction renvoie :

```
items[0] = "MyFolderName"
items[1] = "//path/FolderName"
items[2] = "folder"
items[3] = "MyFavoriteURL"
items[4] = "http://www.MyFavoriteURL.com"
items[5] = "urls"
```

dreamweaver.assetPalette.getSelectedView()

Disponibilité

Dreamweaver 4.

Description

Indique quelle liste est affichée dans le panneau Actifs.

Arguments

Aucun.

Valeurs renvoyées

Renvoie une chaîne comportant la valeur `"site"` ou `"favorites"`.

dreamweaver.assetPalette.insertOrApply()

Disponibilité

Dreamweaver 4.

Description

Insère les éléments sélectionnés ou applique l'élément à la sélection en cours. Il applique des modèles, des couleurs et des URL à la sélection ; il insère également des URL et d'autres éléments au niveau du point d'insertion. Si aucun document n'est ouvert, cette fonction n'est pas disponible.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.assetPalette.canInsertOrApply()*, page 582.

dreamweaver.assetPalette.locateInSite()

Disponibilité

Dreamweaver 4.

Description

Sélectionne les fichiers associés aux éléments sélectionnés du côté local du panneau Site. Cette fonction n'est pas applicable aux couleurs ni aux URL. Elle est disponible dans la liste des sites et la liste des favoris. Si un dossier est sélectionné dans la liste des favoris, il est ignoré.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.newAsset()

Disponibilité

Dreamweaver 4.

Description

Crée un nouvel élément correspondant à la catégorie en cours dans la liste des favoris. Dans le cas des bibliothèques et des modèles, l'élément créé est un nouveau fichier de bibliothèque ou de modèle vierge auquel l'utilisateur peut immédiatement attribuer un nom. Pour les couleurs, le sélecteur de couleur s'affiche. S'il s'agit d'URL, une boîte de dialogue invite l'utilisateur à entrer une URL et un surnom. Cette fonction n'est pas disponible pour les images, les fichiers Shockwave, les fichiers Flash ou les scripts.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.newFolder()

Disponibilité

Dreamweaver 4.

Description

Crée un nouveau dossier ayant un nom par défaut (sans nom) dans la catégorie en cours et place une zone de texte autour du nom. Cette fonction est uniquement disponible dans la liste des favoris.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.recreateLibraryFrom Document()

Disponibilité

Dreamweaver 4.

Description

Remplace l'ancienne fonction `libraryPalette`, `recreateLibraryFromDocument()`. Il remplace un fichier d'élément de bibliothèque (Library item ou LBI) pour l'instance sélectionnée d'un élément de bibliothèque dans le document actif. Revient à cliquer sur Créer à nouveau dans l'inspecteur de propriétés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.refreshSiteAssets()

Disponibilité

Dreamweaver 4.

Description

Analyse le site, affiche la liste des sites et y ajoute des données.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.removeFromFavorites()

Disponibilité

Dreamweaver 4.

Description

Supprime les éléments sélectionnés de la liste des favoris. Cette fonction ne supprime pas les fichiers du disque, sauf dans le cas d'une bibliothèque ou d'un modèle où l'utilisateur est invité à confirmer l'opération avant que le fichier ne soit supprimé. Cette fonction est uniquement disponible dans la liste des favoris ou si la catégorie est Library ou Templates.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.renameNickname()

Disponibilité

Dreamweaver 4.

Description

Affiche une zone de texte autour du nom du dossier ou du surnom du fichier pour pouvoir le modifier. Cette fonction est uniquement disponible dans la liste des favoris ou pour la catégorie Library ou Template.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.setSelectedCategory()

Disponibilité

Dreamweaver 4.

Description

Affiche une catégorie différente.

Arguments

`categoryType`

- L'argument *`categoryType`* peut être l'une des catégories suivantes : `"templates"`, `"library"`, `"images"`, `"movies"`, `"shockwave"`, `"flash"`, `"scripts"`, `"colors"` ou `"urls"`.

Valeurs renvoyées

Aucune.

dreamweaver.assetPalette.setSelectedView()

Disponibilité

Dreamweaver 4.

Description

Passe de la liste des sites à la liste des favoris et vice versa.

Arguments

viewType

- L'argument *viewType* est une chaîne qui peut être `"site"` ou `"favorites"`.

Valeurs renvoyées

Aucune.

dreamweaver.libraryPalette.deleteSelectedItem() (déconseillée)

Disponibilité

Dreamweaver 3 ; déconseillée dans Dreamweaver 4 ; utiliser à la place dreamweaver.assetPalette.setSelectedCategory() et appeler `dreamweaver.assetPalette.removeFromFavorites()`.

Description

Cette fonction supprime l'élément de bibliothèque sélectionné du panneau Bibliothèque et supprime le fichier d'élément de la bibliothèque Dreamweaver (LBI) qui lui est associé du dossier Library à la racine du site actuel. Il peut rester des instances de l'élément supprimé sur certaines pages du site.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.libraryPalette.getSelectedItem() (déconseillée)

Disponibilité

Dreamweaver 3 ; version déconseillée dans Dreamweaver 4 ; utiliser à la place dreamweaver.assetPalette.getSelectedItems() .

Description

Cette fonction obtient le chemin de l'élément de bibliothèque sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui contient le chemin d'accès à l'élément de bibliothèque, exprimé sous la forme d'une URL de type file:///.

dreamweaver.libraryPalette.newFromDocument() (déconseillée)

Disponibilité

Dreamweaver 3 ; déconseillée dans Dreamweaver 4 ; utiliser à la place dreamweaver.assetPalette.setSelectedCategory() et appeler dreamweaver.assetPalette.newAsset() .

Description

Cette fonction crée un nouvel élément de bibliothèque basé sur l'élément sélectionné dans le document actif.

Arguments

bReplaceCurrent

- L'argument *bReplaceCurrent* est une valeur booléenne indiquant si la sélection doit être remplacée par une instance du nouvel élément de bibliothèque créé.

Valeurs renvoyées

Aucune.

dreamweaver.libraryPalette.recreateFromDocument() (déconseillée)

Disponibilité

Dreamweaver 3 ; version déconseillée dans Dreamweaver 4 ; utiliser à la place dreamweaver.assetPalette.recreateLibraryFrom Document() .

Description

Cette fonction crée dans le document actif un fichier LBI correspondant à l'instance sélectionnée d'un élément de bibliothèque. Revient à cliquer sur Créer à nouveau dans l'inspecteur de propriétés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.libraryPalette.renameSelectedItem() (déconseillée)

Disponibilité

Dreamweaver 3; déconseillée dans Dreamweaver 4 ; utiliser à la place dreamweaver.assetPalette.setSelectedCategory() avec l'argument `"library"` et appeler dreamweaver.assetPalette.renameNickname() .

Description

Cette fonction transforme le nom de l'élément de bibliothèque sélectionné en champ de texte et permet à l'utilisateur de renommer la sélection.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.referencePalette.getFontSize()

Disponibilité

Dreamweaver 4.

Description

Renvoie la taille de police actuelle de la zone d'affichage du panneau Référence.

Arguments

Aucun.

Valeurs renvoyées

Les tailles de police relatives exprimées par les valeurs `small`, `medium` ou `large`.

dreamweaver.referencePalette.setFontSize()

Disponibilité

Dreamweaver 4.

Description

Modifie la taille de police affichée dans le panneau Référence.

Arguments

`fontSize`

- L'argument *`fontSize`* correspond à l'une des tailles relatives suivantes : `small`, `medium` ou `large`.

Valeurs renvoyées

Aucune.

dreamweaver.templatePalette.deleteSelected Template() (déconseillée)

Disponibilité

Dreamweaver 3, déconseillée dans Dreamweaver 4 ; utiliser à la place dreamweaver.assetPalette.setSelectedCategory() avec `"templates"` comme valeur d'argument et appeler dreamweaver.assetPalette.removeFromFavorites() .

Description

Cette fonction supprime le modèle sélectionné du dossier des modèles.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.templatePalette.getSelected Template() (déconseillée)

Disponibilité

Dreamweaver 3 ; version déconseillée dans Dreamweaver 4 ; utiliser à la place dreamweaver.assetPalette.getSelectedItems() .

Description

Cette fonction obtient le chemin du modèle sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui contient le chemin d'accès du modèle, exprimé sous la forme d'une URL de type file://.

dreamweaver.templatePalette.renameSelected Template() (déconseillée)

Disponibilité

Dreamweaver 3, déconseillée dans Dreamweaver 4 ; utiliser à la place dreamweaver.assetPalette.setSelectedCategory() avec `"templates"` comme valeur d'argument et appeler dreamweaver.assetPalette.renameNickname() .

Description

Cette fonction transforme le nom du modèle sélectionné en champ de texte et permet à l'utilisateur de renommer la sélection.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Fonctionsrelatives aux comportements

Ces fonctions vous permettent d'associer des comportements à un objet ou de les en dissocier, d'identifier les comportements associés à un objet, d'obtenir des informations sur un objet auquel est associé un comportement, etc. Les méthodes de l'objet `dreamweaver.behaviorInspector` contrôlent ou agissent uniquement sur la sélection dans le panneau Comportements et non sur la sélection dans le document actif.

dom.addBehavior()

Disponibilité

Dreamweaver 3.

Description

Associe un nouveau couple événement/action (comportement) à l'élément sélectionné. Cette fonction n'est valide que pour le document actif.

Arguments

`event, action, {eventBasedIndex}`

- L'argument *`event`* est le gestionnaire d'événements JavaScript à utiliser pour associer le comportement à l'élément (par exemple, `onClick`, `onMouseOver` ou `onLoad`).
- L'argument *`action`* est l'appel de fonction qui serait renvoyé par `applyBehavior()` si l'action était ajoutée à l'aide du panneau Comportements (par exemple, `"MM_popupMsg('Hello World')"`).
- L'argument facultatif *`eventBasedIndex`* est la position à laquelle cette action doit être ajoutée. L'argument *`eventBasedIndex`* est un index de référence zéro ; par conséquent, s'il existe déjà deux actions associées à l'événement en question et que vous spécifiez *`eventBasedIndex`* comme étant l'action 1, cette dernière sera insérée et exécutée entre les deux autres. Si vous ne définissez pas cet argument, l'action est insérée à la suite des actions déjà associées à l'événement spécifié.

Valeurs renvoyées

Aucune.

dom.getBehavior()

Disponibilité

Dreamweaver 3.

Description

Obtient l'action qui se trouve à la position indiquée dans l'événement spécifié. Cette fonction agit sur la sélection en cours et n'est valide que pour le document actif.

Arguments

event, {eventBasedIndex}

- L'argument *event* est le gestionnaire d'événements JavaScript à utiliser pour associer l'action à l'élément (par exemple, `onClick`, `onMouseOver` ou `onLoad`).
- L'argument facultatif *eventBasedIndex* est la position de l'action à obtenir. Par exemple, si deux actions sont associées à l'événement spécifié, 0 est la première et 1 la seconde. Si cet argument n'est pas défini, la fonction renvoie toutes les actions associées à l'événement spécifié.

Valeurs renvoyées

Chaîne représentant l'appel de fonction (par exemple `"MM_swapImage('document.Image1','document.Image1','foo.gif','#933292969950')"`) ou tableau de chaînes, si vous n'avez pas défini l'argument *eventBasedIndex*.

dom.reapplyBehaviors()

Disponibilité

Dreamweaver 3.

Description

Vérifie si les fonctions associées aux appels de comportement sur le nœud spécifié sont présentes dans la section `HEAD` du document et, dans le cas contraire, les y insère.

Arguments

elementNode

- L'argument *elementNode* est un nœud d'élément du document actif. Si cet argument n'est pas défini, Dreamweaver recherche les appels de comportement orphelins sur tous les nœuds d'élément du document.

Valeurs renvoyées

Aucune.

dom.removeBehavior()

Disponibilité

Dreamweaver 3.

Description

Supprime l'action qui se trouve à la position indiquée dans l'événement spécifié. Cette fonction agit sur la sélection en cours et n'est valide que pour le document actif.

Arguments

event, {eventBasedIndex}

- L'argument *event* est le gestionnaire d'événements à utiliser pour associer l'action à l'élément (par exemple, `onClick`, `onMouseOver` ou `onLoad`). Si cet argument n'est pas défini, toutes les actions sont supprimées de l'élément.
- L'argument facultatif *eventBasedIndex* est la position de l'action à supprimer. Par exemple, si deux actions sont associées à l'événement spécifié, 0 est la première et 1 la seconde. Si cet argument n'est pas défini, toutes les actions associées à l'événement sélectionné sont supprimées.

Valeurs renvoyées

Aucune.

dreamweaver.getBehaviorElement()

Disponibilité

Dreamweaver 2.

Description

Obtient l'objet DOM correspondant à la balise à laquelle le comportement est appliqué. Cette fonction ne s'applique qu'aux fichiers d'action de comportement.

Arguments

Aucun.

Valeurs renvoyées

Un objet DOM ou une valeur `null`. La fonction renvoie la valeur `null` dans les cas suivants :

- lorsque le script en cours d'exécution n'est pas actif dans le contexte du panneau Comportements ;
- lorsque le panneau Comportements est utilisé pour modifier un comportement dans un scénario ;
- lorsque le script en cours d'exécution est appelé par la fonction `dreamweaver.popupAction()` ;
- lorsque le panneau Comportements associe un événement à un empaqueteur de lien et que ce dernier n'existe pas ;
- lorsque cette fonction se trouve en dehors d'un fichier d'action.

Exemple

La fonction `dreamweaver.getBehaviorElement()` peut être utilisée de la même manière que la fonction dreamweaver.getBehaviorTag() pour déterminer si l'action sélectionnée est adaptée à la balise HTML sélectionnée. Elle vous permet en outre d'accéder à des informations complémentaires sur la balise et ses attributs. Comme indiqué dans l'exemple suivant, si vous créez une action qui ne peut être appliquée qu'à un lien hypertexte (`A HREF`) ne renvoyant pas à un autre cadre ou une autre fenêtre, utilisez la fonction `getBehaviorElement()` dans la fonction qui initialise l'interface utilisateur de la boîte de dialogue des paramètres.

```
function initializeUI(){
  var theTag = dreamweaver.getBehaviorElement();
  var CANBEAPPLIED = (theTag.tagName == "A" && ¬
  theTag.getAttribute("HREF") != null && ¬
  theTag.getAttribute("TARGET") == null);
  if (CANBEAPPLIED) {
    // display the action UI
  } else{
    // display a helpful message that tells the user
    // that this action can only be applied to a
    // hyperlink without an explicit target]
  }
}
```

dreamweaver.getBehaviorEvent() (déconseillée)

Disponibilité

Dreamweaver 1.2, déconseillée dans Dreamweaver 2, car les actions sont désormais choisies avant les événements.

Description

Dans un fichier d'action de comportement, cette fonction obtient l'événement qui déclenche l'action.

Arguments

Aucun.

Valeurs renvoyées

Chaîne représentant l'événement. Il s'agit de la chaîne transmise à la fonction *`canAcceptBehavior()`* sous la forme d'un argument (`event`).

dreamweaver.getBehaviorTag()

Disponibilité

Dreamweaver 1.2.

Description

Obtient la source de la balise à laquelle le comportement est appliqué. Cette fonction ne s'applique qu'aux fichiers d'action.

Arguments

Aucun.

Valeurs renvoyées

Chaîne représentant la source de la balise. Il s'agit de la chaîne transmise à la fonction *`canAcceptBehavior()`* sous forme d'argument (`HTMLelement`). Si cette fonction apparaît en dehors d'un fichier d'action, la valeur renvoyée est une chaîne vide.

Exemple

Si vous créez une action qui ne peut être appliquée qu'à un lien hypertexte (`A HREF`), vous pouvez utiliser la fonction `getBehaviorTag()`, comme indiqué dans l'exemple suivant, dans la fonction qui initialise l'interface utilisateur de la boîte de dialogue des paramètres :

```
function initializeUI(){
  var theTag = dreamweaver.getBehaviorTag().toUpperCase();
```

```
  var CANBEAPPLIED = (theTag.indexOf('HREF') != -1));
  if (CANBEAPPLIED) {
    // display the action UI
  } else{
    // display a helpful message that tells the user
    // that this action can only be applied to a
    // hyperlink
  }
}
```

dreamweaver.popupAction()

Disponibilité

Dreamweaver 2.

Description

Invoque une boîte de dialogue de paramètres correspondant à l'action de comportement spécifiée. Pour l'utilisateur, cela revient à sélectionner l'action dans le menu déroulant des actions du panneau Comportements. Cette fonction permet aux fichiers d'extension autres que des actions d'associer des comportements aux objets dans le document de l'utilisateur. L'utilisateur ne peut effectuer aucune autre modification tant qu'il n'a pas fermé la boîte de dialogue.

REMARQUE	Cette fonction peut être appelée au sein de la fonction `objectTag()` ou dans tout script de fichier de commande ou dans le fichier de l'inspecteur de propriétés.

Arguments

`actionName, {funcCall}`

- L'argument *`actionName`* est une chaîne contenant le nom d'un fichier dans le dossier Configuration/Behaviors/Actions qui contient une action de comportement JavaScript (par exemple, `"Timeline/Play Timeline.htm"`).
- L'argument facultatif *`funcCall`* est une chaîne contenant un appel de fonction pour l'action définie dans `actionName` (par exemple, `"MM_playTimeline(...)"`). Cet argument, s'il est défini, est fourni par la fonction `applyBehavior()` du fichier d'action.

Valeurs renvoyées

Appel de fonction de l'action de comportement. Lorsque l'utilisateur clique sur OK dans la boîte de dialogue des paramètres, le comportement est ajouté au document actif (les fonctions appropriées sont ajoutées dans la section `HEAD` du document, le code HTML peut être ajouté au début de la section `BODY` et d'autres modifications peuvent être effectuées dans le document). L'appel de fonction (`"MM_playTimeline(...)"`, par exemple) n'est pas ajouté au document ; il devient la valeur renvoyée par cette fonction.

dreamweaver.behaviorInspector.getBehaviorAt()

Disponibilité

Dreamweaver 3.

Description

Obtient le couple événement/action (comportement) qui se trouve à la position indiquée dans le panneau Comportements.

Arguments

`positionIndex`

- L'argument *`positionIndex`* est la position de l'action dans le panneau Comportements. La première action de la liste est à la position 0.

Valeurs renvoyées

Tableau constitué de deux éléments :

- un gestionnaire d'événements ;
- un appel de fonction ou une instruction JavaScript.

Exemple

`positionIndex` étant un index de base zéro, si le panneau Comportements affiche la liste, un appel à la fonction `dreamweaver.behaviorInspector.getBehaviorAt(2)` renvoie un tableau constitué de deux chaînes : `"onMouseOver"` et `"MM_changeProp('document.moon','document.moon','src','sun.gif','MG')"`.

dreamweaver.behaviorInspector.getBehaviorCount()

Disponibilité

Dreamweaver 3.

Description

Compte le nombre d'actions associées à l'élément actuellement sélectionné à l'aide de gestionnaires d'événements.

Arguments

Aucun.

Valeurs renvoyées

Nombre entier qui représente le nombre d'actions attachées à l'élément. Ce nombre est équivalent au nombre d'actions visibles dans le panneau Comportements et comprend les actions de comportement Dreamweaver et le code JavaScript personnalisé.

Exemple

Un appel à la fonction `dreamweaver.behaviorInspector.getBehaviorCount()` pour le lien sélectionné `<A HREF="javascript:setCookie()" onClick="MM_popupMsg('A cookie has been set.');parent.rightframe.location.href='aftercookie.html'">` renvoie la valeur 2.

dreamweaver.behaviorInspector.getSelectedBehavior()

Disponibilité

Dreamweaver 3.

Description

Obtient la position de l'action sélectionnée dans le panneau Comportements.

Arguments

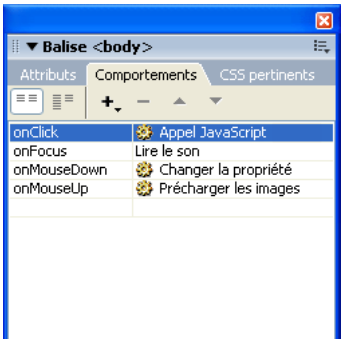
Aucun.

Valeurs renvoyées

Un nombre entier qui représente la position de l'action sélectionnée dans le panneau Comportements ou –1 si aucune action n'est sélectionnée.

Exemple

Si la première action du panneau Comportements est sélectionnée, comme le montre la figure ci-après, un appel à la fonction `dreamweaver.behaviorInspector.getSelectedBehavior()` renvoie le nombre 0 :



dreamweaver.behaviorInspector.moveBehaviorDown()

Disponibilité

Dreamweaver 3.

Description

Déplace une action de comportement vers le bas, à l'intérieur d'une séquence, en modifiant son ordre d'exécution au sein d'un événement.

Arguments

`positionIndex`

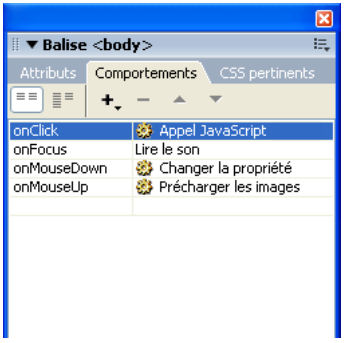
- L'argument *`positionIndex`* est la position de l'action dans le panneau Comportements. La première action de la liste est à la position 0.

Valeurs renvoyées

Aucune.

Exemple

Si le panneau Comportements est configuré comme dans l'illustration ci-après, l'appel de la fonction `dreamweaver.behaviorInspector.moveBehaviorDown(2)` aura pour effet d'intervertir les positions des actions Précharger les images et Changer la propriété dans l'événement `onMouseDown`. Pour toute autre position, la fonction `dreamweaver.behaviorInspector.moveBehaviorDown()` n'a aucun effet, car un seul comportement est associé aux événements `onClick` et `onFocus` et le comportement figurant à la position 3 se trouve déjà à la fin du groupe `onMouseDown`.



dreamweaver.behaviorInspector.moveBehaviorUp()

Disponibilité

Dreamweaver 3.

Description

Déplace un comportement vers le haut, à l'intérieur d'une séquence, en modifiant son ordre d'exécution au sein d'un événement.

Arguments

`positionIndex`

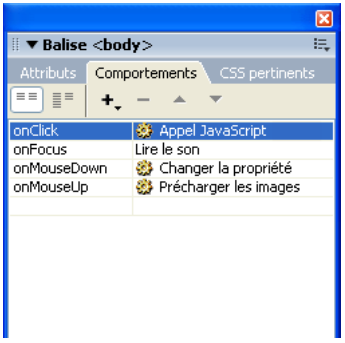
- L'argument *`positionIndex`* est la position de l'action dans le panneau Comportements. La première action de la liste est à la position 0.

Valeurs renvoyées

Aucune.

Exemple

Si le panneau Comportements est configuré comme dans l'illustration ci-après, l'appel de la fonction `dreamweaver.behaviorInspector.moveBehaviorUp(3)` aura pour effet d'intervertir les positions des actions Précharger les images et Changer la propriété dans l'événement `onMouseOver`. Pour toute autre position, la fonction `dreamweaver.behaviorInspector.moveBehaviorUp()` n'a aucun effet, car un seul comportement est associé aux événements `onClick` et `onFocus` et le comportement figurant à la position 2 se trouve déjà au début du groupe `onMouseDown`.



dreamweaver.behaviorInspector.setSelectedBehavior()

Disponibilité

Dreamweaver 3.

Description

Sélectionne l'action qui se trouve à la position indiquée dans le panneau Comportements.

Arguments

`positionIndex`

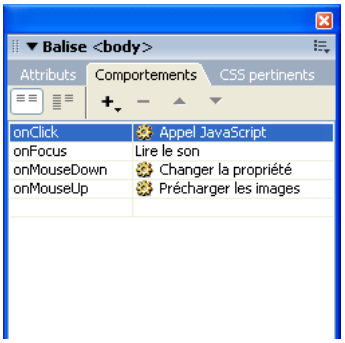
- L'argument *`positionIndex`* est la position de l'action dans le panneau Comportements. La première action de la liste est à la position 0. Pour désélectionner toutes les actions, spécifiez l'index de position (*`positionIndex`*) sur –1. Spécifier une position à laquelle ne correspond aucune action revient à spécifier –1.

Valeurs renvoyées

Aucune.

Exemple

Si le panneau Comportements est défini comme dans la figure suivante, l'appel de la fonction `dreamweaver.behaviorInspector.setSelection(2)` sélectionne l'action Changer la propriété associée à l'événement `onMouseDown` :



Fonctions relatives au Presse-papiers

Ces fonctions permettent de copier, de couper et de coller des informations à l'aide du Presse-papiers. Sur un ordinateur Macintosh, certaines fonctions du Presse-papiers peuvent également être appliquées aux champs de texte des boîtes de dialogue et des panneaux flottants. Dans ce cas, elles peuvent être implémentées en tant que méthodes de l'objet `dreamweaver` ou en tant que méthodes de l'objet DOM. La version `dreamweaver` de la fonction agit sur la sélection dans la fenêtre active, qu'il s'agisse d'une fenêtre de document, de l'inspecteur de code ou du panneau Site. Sur un ordinateur Macintosh, la fonction peut également être appliquée au contenu sélectionné d'une zone de texte s'il s'agit du champ en cours. La version DOM de la fonction agit toujours sur la sélection dans le document spécifié.

dom.clipCopy()

Disponibilité

Dreamweaver 3.

Description

Copie la sélection dans le Presse-papiers avec toutes les balises HTML correspondantes.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.clipCopyText()

Disponibilité

Dreamweaver 3.

Description

Copie le texte sélectionné dans le Presse-papiers sans les balises HTML correspondantes.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canClipCopyText()*, page 570.

dom.clipCut()

Disponibilité

Dreamweaver 3.

Description

Coupe la sélection et la place dans le Presse-papiers, avec toutes les balises HTML correspondantes.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.clipPaste()

Disponibilité

Dreamweaver 3.

Description

Colle le contenu du Presse-papiers dans le document actif, au niveau du point d'insertion ou par dessus la sélection en cours. Si le Presse-papiers contient des balises HTML, celles-ci sont interprétées comme telles.

Arguments

Aucun.

Valeurs renvoyées

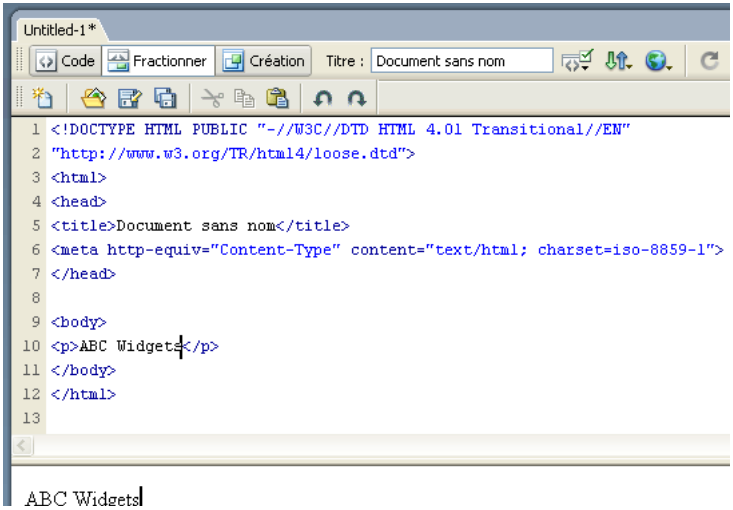
Aucune.

Activateur

Voir *dom.canClipPaste()*, page 571.

Exemple

Si le Presse-papiers contient `ABC Widgets`, un appel à la fonction `dw.getDocumentDOM().clipPaste()` produira le résultat suivant :



dom.clipPasteText() (déconseillée)

Disponibilité

Dreamweaver 3. Déconseillée dans Dreamweaver 8. Utilisez plutôt la fonction `dom.clipPaste("text")`.

Description

Colle le contenu du Presse-papiers dans le document actif, au niveau du point d'insertion en cours ou par dessus la sélection en cours. Cette fonction remplace les sauts de ligne éventuels par des balises `BR`. Si le Presse-papiers contient des balises HTML, celles-ci ne sont pas interprétées et les séparateurs de balises sont remplacés par `<` et `>` respectivement.

Arguments

Aucun.

Valeurs renvoyées

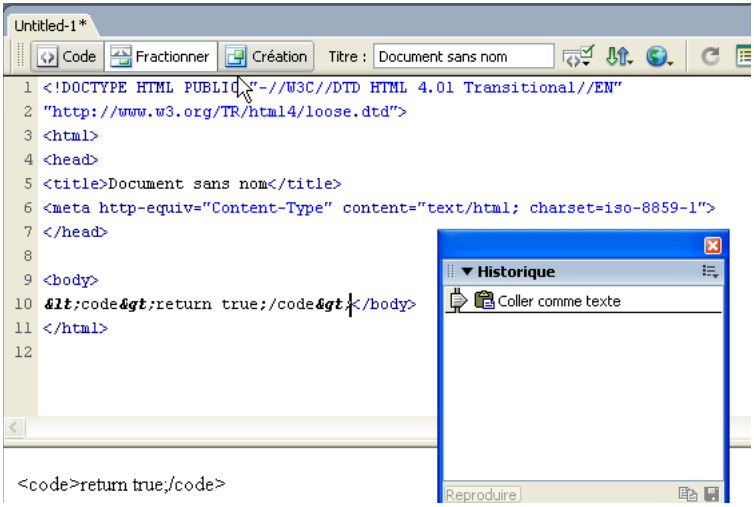
Aucune.

Activateur

Voir *dom.canClipPasteText()*, page 571.

Exemple

Si le Presse-papiers contient ce qui suit : `<code>return true;</code>`, un appel à la fonction `dw.getDocumentDOM().clipPasteText()` produira le résultat suivant :



dreamweaver.clipCopy()

Disponibilité

Dreamweaver 3.

Description

Copie la sélection à partir de la fenêtre de document, la boîte de dialogue, le panneau Site ou le panneau flottant en cours dans le Presse-papiers.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canClipCopy()*, page 582.

dreamweaver.clipCut()

Disponibilité

Dreamweaver 3.

Description

Supprime la sélection dans la fenêtre de document, la boîte de dialogue, le panneau Site ou le panneau flottant actif pour la placer dans le Presse-papiers.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canClipCut()*, page 583.

dreamweaver.clipPaste()

Disponibilité

Dreamweaver 3. Ajout de l'argument `strPasteOption` dans Dreamweaver 8.

Description

Colle le contenu du Presse-papiers dans la fenêtre de document, la boîte de dialogue, le panneau flottant ou le panneau Site actif.

Arguments

{strPasteOption}

- L'argument facultatif *strPasteOption* indique le type de collage à effectuer. Ces valeurs sont : `"text"`, `"structured"`, `"basicFormat"` et `"fullFormat"`.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canClipPaste()*, page 583.

Exemple

L'exemple suivant colle le contenu du Presse-papiers sous forme de texte :

```
dw.clipPaste("text");
```

dreamweaver.getClipboardText()

Disponibilité

Dreamweaver 3.

Description

Obtient tout le texte mémorisé dans le Presse-papiers.

Arguments

{bAsText}

- La valeur booléenne facultative *bAsText* spécifie si le contenu du Presse-papiers est extrait sous forme de texte. Si *bAsText* a la valeur `true`, le contenu du Presse-papiers est extrait sous forme de texte. Si *bAsText* a la valeur `false`, le contenu conserve sa mise en forme. Par défaut, cet argument renvoie la valeur `false`.

Valeurs renvoyées

Chaîne représentant le contenu du Presse-papiers (il peut s'agir de balises HTML) ou, si le Presse-papiers est vide, rien ne se passe.

Exemple

Si la fonction `dreamweaver.getClipboardText()` renvoie `"text <b>bold</b> text"`, alors `dreamweaver.getClipboardText(true)` renvoie `"text bold text"`.

Fonctions relatives aux éléments de bibliothèque et aux modèles

Ces fonctions permettent d'effectuer des opérations associées aux éléments de bibliothèque et aux modèles, telles que la création, la mise à jour et la rupture de liens entre un document et un élément de la bibliothèque ou un modèle. Les méthodes de l'objet `dreamweaver.libraryPalette` contrôlent ou agissent sur la sélection dans les éléments de bibliothèque du panneau Actifs, et non dans le document actif. De même, les méthodes de l'objet `dreamweaver.templatePalette` contrôlent ou agissent sur la sélection dans les objets de modèle du panneau Actifs.

dom.applyTemplate()

Disponibilité

Dreamweaver 3.

Description

Applique un modèle au document actif. Si vous ne spécifiez aucun argument, la boîte de dialogue Sélectionner le modèle s'affiche. Cette fonction n'est valide que pour le document actif.

Arguments

`{templateURL}, bMaintainLink`

- L'argument *`templateURL`* est le chemin d'un modèle disponible sur le site en cours, exprimé sous la forme d'une URL de type file://.
- L'argument *`bMaintainLink`* est une valeur booléenne qui indique si le lien au modèle d'origine doit être conservé (`true`) ou non (`false`).

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canApplyTemplate()*, page 570.

dom.detachFromLibrary()

Disponibilité

Dreamweaver 3.

Description

Rompt le lien entre l'instance actuellement sélectionnée d'un élément de bibliothèque et le fichier LBI qui lui est associé en supprimant les balises de verrouillage autour de la sélection. Revient à cliquer sur Détacher de l'original dans l'inspecteur de propriétés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.detachFromTemplate()

Disponibilité

Dreamweaver 3.

Description

Détache le document actif du modèle qui lui est associé.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.getAttachedTemplate()

Disponibilité

Dreamweaver 3.

Description

Obtient le chemin du modèle associé au document.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui contient le chemin d'accès du modèle, exprimé sous la forme d'une URL de type file://.

dom.getEditableRegionList()

Disponibilité

Dreamweaver 3.

Description

Obtient la liste des régions modifiables dans le corps du document.

Arguments

Aucun.

Valeurs renvoyées

Tableau de nœuds d'élément.

Exemple

dom.getSelectedEditableRegion(), page 399.

dom.getIsLibraryDocument()

Disponibilité

Dreamweaver 3.

Description

Détermine si le document est un élément de bibliothèque.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne qui indique si le document est un fichier LBI.

dom.getIsTemplateDocument()

Disponibilité

Dreamweaver 3.

Description

Détermine si le document est un modèle.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne qui indique si le document est un fichier DWT.

dom.getSelectedEditableRegion()

Disponibilité

Dreamweaver 3.

Description

Si la sélection ou le point d'insertion se trouve à l'intérieur d'une région modifiable, cette fonction obtient la position de cette dernière parmi toutes celles qui existent dans le corps du document.

Arguments

Aucun.

Valeurs renvoyées

Un index dans le tableau renvoyé par la fonction `dom.getEditbableRegionList()`. Pour plus d'informations, voir *dom.getEditableRegionList()*, page 398.

Exemple

Le code suivant affiche une boîte de dialogue présentant le contenu de la région modifiable sélectionnée :

```
var theDOM = dw.getDocumentDOM();
var edRegs = theDOM.getEditableRegionList();
var selReg = theDOM.getSelectedEditableRegion();
alert(edRegs[selReg].innerHTML);
```

dom.insertLibraryItem()

Disponibilité

Dreamweaver 3.

Description

Insère une instance d'un élément de bibliothèque dans le document.

Arguments

libraryItemURL

- L'argument *libraryItemURL* est le chemin d'accès à un fichier LBI, exprimé sous la forme d'une URL de type file://.

Valeurs renvoyées

Aucune.

dom.markSelectionAsEditable()

Disponibilité

Dreamweaver 3.

Description

Affiche la boîte de dialogue Nouvelle région modifiable. Lorsque l'utilisateur clique sur Nouvelle région, Dreamweaver marque la sélection comme étant modifiable et laisse le texte tel quel.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canMarkSelectionAsEditable()*, page 577.

dom.newEditableRegion()

Disponibilité

Dreamweaver 3.

Description

Affiche la boîte de dialogue Nouvelle région modifiable. Lorsque l'utilisateur clique sur Nouvelle région, Dreamweaver insère le nom de la région dans le document, entre accolades ({ }), à l'emplacement du point d'insertion.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canMakeNewEditableRegion()*, page 576.

dom.removeEditableRegion()

Disponibilité

Dreamweaver 3.

Description

Supprime une région modifiable du document. Si cette région possède un contenu, celui-ci est conservé ; seuls les marqueurs de région modifiable sont supprimés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canRemoveEditableRegion()*, page 578.

dom.updateCurrentPage()

Disponibilité

Dreamweaver 3.

Description

Met à jour les modèles ou les éléments de bibliothèque du document, ou les deux. Cette fonction n'est valide que pour le document actif.

Arguments

`{typeOfUpdate}`

- L'argument facultatif *`typeOfUpdate`* doit être `"library"`, `"template"` ou `"both"`. S'il n'est pas défini, il prend par défaut la valeur `"both"`.

Valeurs renvoyées

Aucune.

dreamweaver.updatePages()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Mettre à jour les pages et sélectionne les options spécifiées.

Arguments

{typeOfUpdate}

- L'argument facultatif *typeOfUpdate*, si vous le spécifiez, doit être `"library"`, `"template"` ou `"both"`. S'il n'est pas défini, il prend par défaut la valeur `"both"`.

Valeurs renvoyées

Aucune.

Fonctions du panneau Fragments de code

Avec Dreamweaver, les développeurs Web peuvent modifier et enregistrer des blocs de code réutilisables dans le panneau Fragments de code, puis les extraire si besoin est.

Le panneau Fragments de code stocke chaque fragment de code dans un fichier CSN enregistré dans le dossier Configuration\Snippets. Les fragments de code fournis avec Dreamweaver sont stockés dans les dossiers suivants :

- Accessible
- Comments
- Content_tables
- Filelist.txt
- Footers
- Form_elements
- Headers
- Javascript
- Meta

- Navigation
- texte

Les fichiers de fragment de code sont des documents XML. Par conséquent, vous pouvez spécifier le codage dans la directive XML de la manière suivante :

```
<?XML version="1.0" encoding="utf-8">
```

Voici un exemple de fichier de fragment de code :

```
<snippet name="Detect Flash" description="VBscript to check for Flash
  ActiveX control" preview="code" factory="true" type="wrap" >
     <insertText location="beforeSelection">
          <![CDATA[ ------- code --------- ]]>
     </insertText>
     <insertText location="afterSelection">
          <![CDATA[ ------- code --------- ]]>
     </insertText>
</snippet>
```

Les balises de fragment de code des fichiers CSN ont les attributs suivants :

Attribut	Description
`name`	Nom du fragment de code
`description`	Description du fragment de code
`preview`	Type d'aperçu : `"code"` pour afficher le fragment de code dans la zone d'aperçu ou `"design"` pour afficher le fragment de code rendu en code HTML dans la zone d'aperçu.
`type`	`"wrap"` si le fragment de code sert à envelopper la sélection de l'utilisateur, `"block"` si le fragment de code doit être inséré avant la sélection.

Vous pouvez faire appel aux méthodes suivantes pour ajouter les fonctions du panneau Fragments de code à vos extensions.

dreamweaver.snippetPalette.getCurrentSnippetPath()

Disponibilité

Dreamweaver MX 2004.

Description

Renvoie le chemin du fragment actuellement sélectionné dans le panneau Fragments de code.

Arguments

Aucun.

Valeurs renvoyées

Chemin, selon le dossier Snippets, du fragment actuellement sélectionné dans le panneau Fragments de code. Renvoie une chaîne vide si aucun fragment n'est sélectionné.

dreamweaver.snippetPalette.newFolder()

Disponibilité

Dreamweaver MX.

Description

Crée un nouveau dossier ayant un nom par défaut (*sans titre*) et place une zone de texte autour du nom.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.snippetPalette.newSnippet()

Disponibilité

Dreamweaver MX.

Description

Ouvre la boîte de dialogue Ajouter un fragment de code et l'affiche au premier plan.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.snippetPalette.editSnippet()

Disponibilité

Dreamweaver MX.

Description

Ouvre la boîte de dialogue Modifier le fragment de code et l'affiche au premier plan pour que vous puissiez modifier l'élément sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.snippetpalette.canEditSnippet()*, page 603.

dreamweaver.snippetPalette.insert()

Disponibilité

Dreamweaver MX.

Description

Applique le fragment sélectionné dans le panneau Fragment de code à la sélection en cours.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.snippetpalette.canInsert()*, page 603.

dreamweaver.snippetPalette.insertSnippet()

Disponibilité

Dreamweaver MX.

Description

Insère le fragment sélectionné dans la sélection en cours.

Arguments

- Chaîne spécifiant le chemin du fragment par rapport au dossier Snippets.

Valeurs renvoyées

Valeur booléenne.

Activateur

Voir *dreamweaver.snippetpalette.canInsert()*, page 603.

Exemple

L'appel de la fonction `dw.snippetPalette.insertSnippet()` insère le fragment de code à l'emplacement spécifié par l'argument dans le document actif, au point d'insertion :

```
dw.snippetPalette.insertSnippet('Text\\Different_Link_Color.csn');
```

dreamweaver.snippetPalette.rename()

Disponibilité

Dreamweaver MX.

Description

Affiche une zone de texte autour du nom du dossier ou du surnom du fichier sélectionné pour permettre de le modifier.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.snippetPalette.remove()

Disponibilité

Dreamweaver MX.

Description

Supprime l'élément ou le dossier sélectionné du panneau Fragment de code et supprime le fichier du disque.

Valeurs renvoyées

Aucune.

CHAPITRE 15

Documents dynamiques

15

Les fonctions de document dynamique de Macromedia Dreamweaver 8 effectuent des opérations liées aux pages de serveur web. Ces opérations peuvent, par exemple, être le renvoi d'une propriété pour le nœud sélectionné dans le panneau Composants, l'obtention d'une liste des sources de données dans le document utilisateur, l'affichage de contenu dynamique en mode Création, l'application d'un comportement de serveur à un document ou encore l'obtention des noms de tous les modèles de serveur actuellement définis.

Fonctions de composants de serveur

Ces fonctions permettent d'accéder au nœud sélectionné dans la commande d'arborescence Composants de serveur qui apparaît dans le panneau Composants. Vous pouvez également les utiliser pour actualiser l'affichage de l'arborescence de composants.

dreamweaver.serverComponents.getSelectedNode()

Disponibilité

Dreamweaver MX.

Description

Renvoie la propriété `ComponentRec` sélectionnée dans la commande d'arborescence Composants de serveur.

Arguments

Aucun.

Valeurs renvoyées

La propriété `ComponentRec`.

dreamweaver.serverComponents.refresh()

Disponibilité

Dreamweaver MX.

Description

Actualise l'affichage de l'arborescence de composants.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Fonctions relatives aux sources de données

Les fichiers de source de données sont stockés dans le dossier Configuration/DataSources. Tous les modèles de serveur ont leur propre dossier : ASP.Net/C#, ASP.Net/VisualBasic, ASP/JavaScript, ASP/VBScript, ColdFusion, JSP et PHP/MySQL. Le sous-dossier de chaque modèle contient des fichiers HTML et EDML associés aux sources de données de ce modèle de serveur.

Pour plus d'informations sur l'utilisation de sources de données dans Dreamweaver, voir Sources de données dans *Extension de Dreamweaver*.

dreamweaver.dbi.getDataSources

Disponibilité

Dreamweaver UltraDev 4.

Description

Appelle la fonction `findDynamicSources()` pour chaque fichier contenu dans le dossier Configuration/DataSources. Vous pouvez utiliser cette fonction pour générer une liste de toutes les sources de données du document de l'utilisateur. Cette fonction est répétée sur tous les fichiers du dossier Configuration/DataSources, appelle la fonction `findDynamicSources()` dans chaque fichier, concatène tous les tableaux renvoyés et renvoie le tableau de sources de données concaténé.

Arguments

Aucun.

Valeurs renvoyées

Un tableau contenant une liste concaténée de toutes les sources de données contenues dans le document de l'utilisateur. Chaque élément du tableau est un objet et chaque objet a les propriétés suivantes :

- La propriété `title` correspond au libellé qui apparaît à droite de l'icône de chaque nœud parent. La propriété `title` est toujours définie.
- La propriété `imageFile` est le chemin de fichier contenant l'icône (une image GIF) qui représente le nœud parent dans les boîtes de dialogue Données dynamiques et Texte dynamique ou dans le panneau Liaisons. La propriété `imageFile` est toujours définie.
- La propriété `allowDelete` est facultative. Si cette propriété est définie sur la valeur `false`, lorsque l'utilisateur clique sur ce nœud dans le panneau Liaisons, le bouton moins (-) apparaît désactivé. Si elle est définie sur la valeur `true`, le bouton moins () est activé. Si la propriété n'est pas définie, le bouton moins (-) est activé lorsque l'utilisateur clique sur l'élément (comme si la propriété était définie sur la valeur `true`).
- La propriété `dataSource` est le nom du fichier dans lequel la fonction `findDynamicSources()` est définie. Par exemple, la fonction `findDynamicSources()` dans le fichier Session.htm, situé dans le dossier Configuration/DataSources/ASP_Js, définit la propriété `dataSource` sur `session.htm`. Cette propriété est toujours définie.
- La propriété `name` est le nom du comportement de serveur associé à la source de données `dataSource`, s'il existe. La propriété `name` est toujours définie, mais il peut s'agir d'une chaîne vide (`""`) si aucun comportement de serveur n'est associé à la source de données (tel qu'une variable de session).

Fonctions de l'Extension Data Manager

Le gestionnaire de données d'extension (EDM) se compose des API de cette section. Vous pouvez programmer l'accès et la manipulation des données contenues dans les fichiers groupe et participant en appelant ces fonctions. L'EDM fonctionne de la manière suivante :

- L'EDM exécute toutes les entrées/sorties de fichier EDML pour les fichiers Groupe et Participant.
- En exécutant toutes les demandes de données pour le modèle de serveur en cours, l 'EDM agit comme un filtre de modèle de serveur.

dreamweaver.getExtDataValue()

Disponibilité

Dreamweaver UltraDev 4.

Description

Cette fonction extrait les valeurs de champ d'un fichier EDML pour les nœuds spécifiés.

Arguments

qualifier(s)

- L'argument *qualifier(s)* est une liste, de longueur variable (selon le niveau d'information demandé) de qualificateurs de nœuds séparés par des virgules comprenant le nom du groupe ou du participant, le sous-bloc (le cas échéant) et le nom du champ.

Valeurs renvoyées

Dreamweaver attend une valeur de champ. Si la valeur n'est pas spécifiée, Dreamweaver utilise la valeur par défaut.

Exemple

L'exemple suivant extrait la valeur de l'attribut d'emplacement pour la balise insertText du participant recordset_main participant :

```
dw.getExtDataValue("recordset_main", "insertText", "location");
```

dreamweaver.getExtDataArray()

Disponibilité

Dreamweaver UltraDev 4.

Description

Cette fonction extrait un tableau de valeurs d'un fichier EDML pour les nœuds spécifiés.

Arguments

qualifier(s)

- L'argument *qualifier(s)* est une liste, de longueur variable, de qualificateurs de nœuds séparés par des virgules, comprenant le nom du groupe ou du participant, le sous-bloc (le cas échéant) et le nom du champ.

Valeurs renvoyées

Dreamweaver attend un tableau de noms de nœuds enfants.

dreamweaver.getExtParticipants()

Disponibilité

Dreamweaver UltraDev 4.

Description

Cette fonction extrait la liste de participants d'un fichier de groupe EDML ou de fichiers participants.

Arguments

value, qualifier(s)

- L'argument *value* est une valeur de propriété. Il est ignoré s'il n'est pas renseigné. Par exemple, `dreamweaver.getExtParticipants("", "participant");`
- L'argument *qualifier(s)* est une liste, de longueur variable, de qualificateurs de nœuds ayant la propriété requise, séparés par des virgules.

Valeurs renvoyées

Dreamweaver attend un tableau de noms de participants ayant la propriété requise, si elle est spécifiée, et la propriété correspondant à la valeur requise, si elle est spécifiée.

dreamweaver.getExtGroups()

Disponibilité

Dreamweaver UltraDev 4.

Description

Extrait le nom du groupe, qui est l'équivalent du nom du comportement de serveur, d'un fichier groupe EDML.

Arguments

value, qualifier(s)

- L'argument *value* est une valeur de propriété. Il est ignoré s'il n'est pas renseigné.
- L'argument *qualifier(s)* est une liste, de longueur variable, de qualificateurs de nœuds ayant la propriété requise, séparés par des virgules.

Valeurs renvoyées

Dreamweaver attend un tableau de noms de groupes ayant la propriété requise, si elle est spécifiée, et la propriété correspondant à la valeur requise, si elle est spécifiée.

dreamweaver.refreshExtData()

Disponibilité

Dreamweaver UltraDev 4.

Description

Recharge tous les fichiers de données d'extension.

REMARQUE	Ceci est un nouveau tableau Note. Il est associé à la balise de paragraphe Table_anchor. Pour le créer, copiez et collez-le à partir de la page de référence.

CONSEIL	Vous pouvez créer une commande utile à partir de cette fonction, qui vous permettra de recharger les modifications apportées aux fichiers EDML de comportement de serveur sans avoir à redémarrer Dreamweaver.

Arguments

Aucun.

Valeurs renvoyées

Dreamweaver attend des données rechargées.

Fonctions Live data

Vous pouvez utiliser les fonctions Live Data suivantes pour reproduire les fonctionnalités de menu :

- La fonction `showLiveDataDialog()` s'utilise pour l'élément de menu Affichage > Paramètres Live Data.
- La fonction `setLiveDataMode()` s'utilise pour les éléments de menu Affichage > Live Data et Affichage > Actualiser les données dynamiques.
- La fonction `getLiveDataMode()` détermine si le mode Live Data est actif.

Vous pouvez utiliser les autres fonctions Live Data lorsque vous implémentez la fonction `liveDataTranslateMarkup()` pour l'API du traducteur.

dreamweaver.getLiveDataInitTags()

Disponibilité

Dreamweaver UltraDev 1.

Description

Renvoie les balises d'initialisation pour le document actif. Les balises d'initialisation sont les balises HTML que l'utilisateur fournit dans la boîte de dialogue Paramètres Live Data. Cette fonction est généralement appelée à partir de la fonction `liveDataTranslateMarkup()` d'un traducteur, de façon à ce que le traducteur puisse transmettre les balises à la fonction `liveDataTranslate()`.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui contient les balises d'initialisation.

dreamweaver.getLiveDataMode()

Disponibilité

Dreamweaver UltraDev 1.

Description

Détermine si la fenêtre Live Data est actuellement visible.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la fenêtre Live Data est visible ; false dans le cas contraire.

dreamweaver.getLiveDataParameters ()

Disponibilité

Dreamweaver MX.

Description

Obtient les paramètres d'URL définis comme paramètres Live Data.

Le mode Live Data vous permet d'afficher une page Web au stade de conception (comme si elle avait été traduite par le serveur d'application et renvoyée). En générant un contenu dynamique affichable en mode Création, vous pouvez visualiser la mise en forme d'une page avec ses données dynamiques et l'ajuster, le cas échéant.

Avant d'afficher les données dynamiques, vous devez saisir les paramètres Live Data de tout paramètre d'URL auquel le document fait référence. Cela évite que le serveur Web ne renvoie des erreurs pour des paramètres qui ne sont pas encore définis au stade de la simulation.

La saisie des paramètres d'URL s'effectue par paire nom/valeur. Ainsi, si les scripts de serveur d'un document font référence aux variables d'URL `ID` et `Name`, vous devez définir ces paramètres d'URL avant d'afficher les données dynamiques.

Vous pouvez saisir les paramètres Live Data depuis les endroits suivants dans Dreamweaver :

- Depuis la boîte de dialogue Paramètres Live Data, accessible à partir du menu Affichage.
- Dans le champ de texte URL qui apparaît dans la partie supérieure du document lorsque vous cliquez sur le bouton Affichage des données dynamiques, situé dans la barre d'outils.

Pour les paramètres `ID` et `Name` mentionnés ci-dessus, vous pouvez saisir les paires suivantes :

```
ID     22
Name   Samuel
```

Dans l'URL, ces paramètres apparaissent alors comme indiqué dans l'exemple ci-dessous :

```
http://someURL?ID=22&Name=Samuel
```

Cette fonction vous permet d'obtenir les paramètres Live Data par le biais de JavaScript.

Arguments

Aucun.

Valeurs renvoyées

Tableau qui contient les paramètres d'URL du document actif. Le tableau contient un nombre pair de chaînes de paramètres. Chaque association de deux éléments représente une paire nom/valeur de paramètre d'URL. L'élément pair correspond au nom du paramètre et l'élément impair à sa valeur. Dans l'exemple ci-dessus, `getLiveDataParameters()` renvoie donc le tableau suivant pour les paramètres `ID` et `Name` : `['ID','22','Name','Samuel']`.

Exemple

L'exemple suivant renvoie les paramètres spécifiés comme paramètres Live Data et les conserve dans `paramsArray` :

```
var paramsArray = dreamweaver.getLiveDataParameters();
```

dreamweaver.liveDataTranslate()

Disponibilité

Dreamweaver UltraDev 1.

Description

Envoie un document HTML entier à un serveur d'application, demande au serveur d'exécuter les scripts dans le document, puis renvoie le document HTML résultant. Cette fonction peut également être appelée à partir de la fonction `liveDataTranslateMarkup()` d'un traducteur ; si vous tentez de l'appeler à un autre moment, cela produit une erreur. La fonction `dreamweaver.liveDataTranslate()` effectue les opérations suivantes :

- Elle lit l'image animée (qui s'affiche à côté du bord droit de la fenêtre Live Data).
- Elle analyse les données saisies par l'utilisateur. Si vous cliquez sur l'icône Arrêter, la fonction est immédiatement terminée.
- Elle accepte un argument contenant une seule chaîne de l'appelant (cette chaîne représente généralement le code source entier du document de l'utilisateur. Il s'agit de la même chaîne que celle utilisée dans l'opération suivante).
- Elle enregistre la chaîne HTML du document de l'utilisateur dans un fichier temporaire stocké sur le serveur Live Data.
- Elle envoie une requête HTTP au serveur Live Data, en utilisant les paramètres définis dans la boîte de dialogue Paramètres Live Data.
- Elle reçoit la réponse HTML du serveur Live Data.
- Elle supprime le fichier temporaire du serveur Live Data.
- Elle arrête la lecture du fichier d'animation.
- Elle renvoie la réponse HTML à l'appelant.

Arguments

- Une seule chaîne, représentant généralement le code source entier du document actif de l'utilisateur.

Valeurs renvoyées

Objet `httpReply`. Cet objet est le même que la valeur renvoyée par la fonction `MMHttp.getText()`. Si l'utilisateur clique sur l'icône Arrêter, le code `httpReply.statusCode` de la valeur de retour est égal à 200 (état OK) et son `httpReply.data` est égal à une chaîne vide. Pour plus d'informations sur l'objet `httpReply`, voir Chapitre 2, “API HTTP,” on page 29.

dreamweaver.setLiveDataError()

Disponibilité

Dreamweaver UltraDev 1.

Description

Détermine le message d'erreur qui s'affiche si une erreur se produit lors de l'exécution de la fonction `liveDataTranslateMarkup()` dans un traducteur. Si le document transmis par Dreamweaver à `liveDataTranslate()` contient des erreurs, le serveur retransmet un message d'erreur au format HTML. Si le traducteur (code ayant appelé `liveDataTranslate()`) détermine que le serveur a renvoyé un message d'erreur, il appelle `setLiveDataError()` pour afficher le message d'erreur dans Dreamweaver. Ce message s'affiche une fois que la fonction `liveDataTranslateMarkup()` a été exécutée ; Dreamweaver affiche la description dans une boîte de dialogue d'erreurs. La fonction `setLiveDataError()` doit uniquement être appelée à partir de la fonction `liveDataTranslateMarkup()`.

Arguments

`source`

- L'argument *`source`* est une chaîne qui contient le code source, qui est analysé et affiché dans la boîte de dialogue du message d'erreur.

Valeurs renvoyées

Aucune.

dreamweaver.setLiveDataMode()

Disponibilité

Dreamweaver UltraDev 1.

Description

Active ou désactive l'affichage de la fenêtre Live Data.

Arguments

`bIsVisible`

- L'argument *`bIsVisible`* est une valeur booléenne qui indique si la fenêtre Live Data doit être visible. Si vous transmettez la valeur `true` à cette fonction et que Dreamweaver est défini pour afficher la fenêtre Live Data, l'effet est le même que si l'utilisateur cliquait sur le bouton Actualiser.

Valeurs renvoyées

Aucune.

dreamweaver.setLiveDataParameters()

Disponibilité

Dreamweaver MX.

Description

Définit les paramètres d'URL référencés dans le document pour une utilisation en mode Live Data.

Le mode Live Data vous permet d'afficher une page Web au stade de conception (comme si elle avait été traduite par le serveur d'application et renvoyée). En générant un contenu dynamique affichable en mode Création, vous pouvez visualiser la mise en forme d'une page avec ses données dynamiques et l'ajuster, le cas échéant.

Avant d'afficher les données dynamiques, vous devez saisir les paramètres Live Data de tout paramètre d'URL auquel le document fait référence. Cela évite que le serveur Web ne renvoie des erreurs pour des paramètres qui ne sont pas encore définis au stade de la simulation.

La saisie des paramètres d'URL s'effectue par paire nom/valeur. Ainsi, si les scripts de serveur d'un document font référence aux variables d'URL `ID` et `Name`, vous devez définir ces paramètres d'URL avant d'afficher les données dynamiques.

Cette fonction vous permet de définir les valeurs Live Data par le biais de JavaScript.

Arguments

`liveDataString`

- L'argument *`liveDataString`* est une chaîne qui contient les paramètres d'URL à définir, au sein de paires nom/valeur.

Valeurs renvoyées

Aucune.

Exemple

```
dreamweaver.setLiveDataParameters("ID=22&Name=Samuel")
```

dreamweaver.showLiveDataDialog()

Disponibilité

Dreamweaver UltraDev 1.

Description

Affiche la boîte de dialogue Paramètres Live Data.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Fonctions relatives aux comportements de serveur

Ces fonctions vous permettent de manipuler le panneau Comportements de serveur, accessible sous Fenêtre > Comportements de serveur. Vous pouvez utiliser ces fonctions pour rechercher tous les comportements de serveur sur une page. Vous pouvez aussi les utiliser pour appliquer un nouveau comportement à un document ou pour modifier un comportement existant, et ce par le biais d'un programme.

REMARQUE

Vous pouvez abréger `dw.serverBehaviorInspector` en `dw.sbi`.

dreamweaver.getParticipants()

Disponibilité

Dreamweaver UltraDev 4.

Description

La fonction JavaScript `dreamweaver.getParticipants()` répertorie les participants présents dans le document utilisateur. Une fois tous les participants du comportement trouvés, Dreamweaver mémorise ces listes de participants. En général, cette fonction est utilisée avec la fonction `findServerBehaviors()` (pour plus d'informations, voir Comportements de serveur dans *Extension de Dreamweaver*) pour repérer les instances d'un comportement dans le document utilisateur.

Arguments

`edmlFilename`

- L'argument *edmlFilename* est le nom du fichier groupe ou participant qui contient les noms des participants à localiser dans le document utilisateur. Cette chaîne est le nom de fichier, sans le suffixe d'extension .edml.

Valeurs renvoyées

Cette fonction renvoie un tableau contenant toutes les instances du participant spécifié (ou, dans le cas d'un fichier groupe, une instance d'un participant dans le groupe) apparaissant dans le document utilisateur. Le tableau contient des objets JavaScript, avec un élément pour chaque instance de participant trouvée dans le document utilisateur. Le tableau est trié selon l'ordre dans lequel les participants apparaissent dans le document. Chaque objet JavaScript présente les propriétés suivantes :

- La propriété *participantNode* est un pointeur relié au nœud participant dans le document utilisateur.
- La propriété *participantName* est le nom du fichier EDML du participant (sans le suffixe « .edml »).
- La propriété *parameters* est un objet JavaScript qui stocke toutes les paires paramètre/valeur.
- La propriété *matchRangeMin* définit le décalage de caractères à partir du nœud participant du document jusqu'au début du contenu du participant.
- La propriété *matchRangeMax* est un nombre entier du participant définissant le décalage depuis le début du nœud participant jusqu'au dernier caractère du contenu du participant.

dreamweaver.serverBehaviorInspector.getServerBehaviors()

Disponibilité

Dreamweaver UltraDev 1.

Description

Affiche une liste de tous les comportements sur une page. Si Dreamweaver détermine que la liste interne des comportements de serveur n'est peut-être pas à jour, il appelle la fonction `findServerBehaviors()` pour chaque comportement installé. Chaque fonction renvoie un tableau. Dreamweaver fusionne tous les tableaux en un seul tableau et le trie dans l'ordre où chaque objet `selectedNode` de comportement apparaît dans le document. Dreamweaver stocke le tableau fusionné localement. La fonction `getServerBehaviors()` renvoie un pointeur au tableau fusionné.

Arguments

Aucun.

Valeurs renvoyées

Tableau d'objets JavaScript. L'appel `findServerBehaviors()` renvoie les objets dans le tableau. Ils sont triés dans leur ordre d'apparition dans le panneau Comportements de serveur.

dreamweaver.popupServerBehavior()

Disponibilité

Dreamweaver UltraDev 1.

Description

Applique un nouveau comportement de serveur au document ou modifie un comportement existant. Si l'utilisateur doit définir les paramètres du comportement, une boîte de dialogue s'affiche.

Arguments

{behaviorName or *behaviorObject}*

- L'argument facultatif *behaviorName* est une chaîne qui représente le nom du comportement, la balise de titre d'un fichier ou un nom de fichier.
- L'argument facultatif *behaviorObject* est un objet comportement.

Si vous ne définissez pas cet argument, Dreamweaver exécute le comportement de serveur sélectionné. Si l'argument est le nom d'un comportement de serveur, Dreamweaver ajoute ce comportement à la page. Si l'argument est l'un des objets du tableau renvoyé par la fonction `getServerBehaviors()`, une boîte de dialogue s'affiche pour permettre à l'utilisateur de modifier les paramètres du comportement.

Valeurs renvoyées

Aucune.

Fonctions de modèle de serveur

Dans Macromedia Dreamweaver, tous les documents ont un type de document associé. Pour les types de documents dynamiques, Dreamweaver associe également un modèle de serveur (tel que ASP-JS, ColdFusion ou PHP-MySQL).

Les modèles de serveur sont utilisés pour regrouper une fonctionnalité spécifique à une technologie de serveur. Différents comportements de serveur, sources de données, etc., apparaissent en fonction du modèle de serveur associé au document.

Utilisez les fonctions de modèle de serveur pour déterminer l'ensemble des modèles de serveur actuellement définis, le nom, la langue et la version du modèle de serveur en cours, et si le modèle de serveur en cours prend en charge un ensemble nommé de caractères (tel que UTF-8).

REMARQUE Dreamweaver lit toutes les informations contenues dans le fichier HTML du modèle de serveur et stocke ces informations lors du premier chargement du modèle de serveur. Lorsqu'une extension appelle des fonctions telles que `dom.serverModel.getServerName()`, `dom.serverModel.getServerLanguage()` et `dom.serverModel.getServerVersion()`, ces fonctions renvoient les valeurs stockées.

dom.serverModel.getAppURLPrefix()

Disponibilité

Dreamweaver MX.

Description

Renvoie l'URL du dossier racine du site sur le serveur d'évaluation. Cette URL est la même que celle qui a été spécifiée pour le serveur d'évaluation dans l'onglet Avancé de la boîte de dialogue Définition du site.

Lorsque Dreamweaver communique avec votre serveur d'évaluation, il utilise HTTP (de la même manière qu'un navigateur). Il utilise alors cette URL pour accéder au dossier racine de votre site.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui maintient l'URL au serveur d'application utilisé pour les données dynamiques et le débogage.

Exemple

Si l'utilisateur crée un site et précise que le serveur d'évaluation est sur l'ordinateur local et que le dossier racine est intitulé « `employeeapp` », un appel à la fonction `dom.serverModel.getAppURLPrefix()` renvoie la chaîne suivante :

```
http://localhost/employeeapp/
```

dom.serverModel.getDelimiters()

Disponibilité

Dreamweaver MX.

Description

Permet au code JavaScript d'obtenir le délimiteur de script pour chaque modèle de serveur, afin de pouvoir séparer la gestion du modèle de serveur et celle du code utilisateur.

Arguments

Aucun.

Valeurs renvoyées

Tableau d'objets dans lequel tous les objets contiennent les trois propriétés suivantes :

- La propriété *`startPattern`* est une expression régulière qui correspond au délimiteur d'ouverture de script.
- La propriété *`endPattern`* est une expression régulière qui correspond au délimiteur de fermeture de script.
- Le modèle *`participateInMerge`* est une valeur booléenne qui indique si le contenu entre les délimiteurs utilisés doit (`true`) ou non (`false`) prendre part à la fusion de blocs.

dom.serverModel.getDisplayName()

Disponibilité

Dreamweaver MX.

Description

Obtient le nom du modèle de serveur qui apparaît dans l'interface utilisateur (UI).

Arguments

Aucun.

Valeurs renvoyées

Chaîne dont la valeur est le nom du modèle de serveur.

dom.serverModel.getFolderName()

Disponibilité

Dreamweaver MX.

Description

Obtient le nom du dossier utilisé pour ce modèle de serveur dans le dossier Configuration (tel que le sous-dossier ServerModels).

Arguments

Aucun.

Valeurs renvoyées

Chaîne dont la valeur est le nom du dossier.

dom.serverModel.getServerExtension() (déconseillée)

Disponibilité

Dreamweaver UltraDev 4 ; déconseillée dans Dreamweaver MX

Description

Renvoie l'extension par défaut des fichiers exploitant le modèle de serveur en cours (l'extension du fichier par défaut est la première dans la liste). Si aucun document utilisateur n'est sélectionné, l'objet `serverModel` est défini sur le modèle de serveur du site actuellement sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui représente les extensions de fichier prises en charge.

dom.serverModel.getServerIncludeUrlPatterns()

Disponibilité

Dreamweaver MX.

Description

Renvoie la liste de propriétés, qui vous permet d'accéder aux éléments suivants :

- Modèles URL du traducteur
- Références du fichier
- Type

Arguments

Aucun.

Valeurs renvoyées

Liste d'objets, un pour chaque `searchPattern`. Chaque objet présente les trois propriétés suivantes :

Propriété	Description
`pattern`	Expression JavaScript régulière spécifiée dans le champ `searchPattern` d'un fichier EDML (une expression régulière est délimitée par deux barres obliques (`//`)).
`fileRef`	L'index de base 1 de la sous-correspondance de l'expression régulière qui correspond à la référence du fichier inclus.
`type`	La portion de la valeur `paramName` qui reste après avoir ôté le suffixe _includeUrl. Ce type est affecté à l'attribut du type de balise `<MM:BeginLock>`. Pour avoir un exemple, voir Server Model SSI.htm dans le dossier Configuration/Translators.

Exemple

Le fragment de code suivant d'un fichier participant illustre une balise `searchPatterns` de traducteur :

```
<searchPatterns whereToSearch="comment">
   <searchPattern paramNames=",ssi_comment_includeUrl">
      <![CDATA[/<!--\s*#include\s+(file|virtual)\s*=\s*"([^"]*)"\s*-->/i]]>
   </searchPattern>
</searchPatterns>
```

Le modèle de recherche contient une expression JavaScript régulière qui spécifie deux sous-correspondances (toutes deux entre parenthèses). La première sous-correspondance est pour la chaîne de texte `file` ou `virtual`. La deuxième sous-correspondance est une référence de fichier.

Pour accéder au modèle URL du traducteur, votre code doit ressembler à l'exemple suivant :

```
var serverModel = dw.getDocumentDOM().serverModel;
var includeArray = new Array();
includeArray = serverModel.getServerIncludeUrlPatterns();
```

L'appel à la fonction `serverModel.getServerIncludeUrlPatterns()` renvoie les trois propriétés suivantes :

Propriété	Valeur renvoyée
`pattern`	`/<!--\s*#include\s+(file|virtual)\s*=\s*"([^"]*)"\s*-->/i`
`fileRef`	`2`
`type`	`ssi_comment`

dom.serverModel.getServerInfo()

Disponibilité

Dreamweaver MX.

Description

Renvoie des informations spécifiques au modèle de serveur en cours. Ces informations figurent dans le fichier de définition HTML correspondant au modèle de serveur, situé dans le dossier Configuration/ServerModels.

Vous pouvez modifier les informations contenues dans le fichier de définition HTML ou placer des valeurs ou fonctions de variable supplémentaires dans le fichier. Vous pouvez par exemple modifier les propriétés `serverName`, `serverLanguage` et `serverVersion`. La fonction `dom.serverModel.getServerInfo()` renvoie les informations ajoutées par l'auteur du modèle de serveur au fichier de définition.

REMARQUE Les autres valeurs définies dans les fichiers de modèles de serveur par défaut sont réservées à un usage interne.

Les propriétés `serverName`, `serverLanguage` et `serverVersion` sont spécifiques, accessibles directement au moyen des fonctions correspondantes suivantes :

- `dom.serverModel.getServerName()`
- `dom.serverModel.getServerLanguage()`
- `dom.serverModel.getServerVersion()`

Arguments

Aucun.

Valeurs renvoyées

Objet JavaScript qui contient diverses informations spécifiques au modèle de serveur en cours.

dom.serverModel.getServerLanguage() (déconseillée)

Disponibilité

Dreamweaver UltraDev 1 ; déconseillée dans Dreamweaver MX

Description

Détermine le modèle de serveur associé au document et renvoie cette valeur. Le langage serveur d'un site est la valeur choisie dans le paramètre Langage de script par défaut de l'onglet Infos du serveur d'application de la boîte de dialogue Définition du site. Pour obtenir la valeur de retour, cette fonction appelle la fonction `getServerLanguage()` dans l'API de modèle de serveur.

REMARQUE

La liste Langage de script par défaut existe uniquement dans Dreamweaver 4 et les versions antérieures. Pour Dreamweaver MX (ou version supérieure), la boîte de dialogue Définition du site ne répertorie pas les langages de script pris en charge. Toujours pour Dreamweaver MX (ou version supérieure), la fonction `dom.serverModel.getServerLanguage()` lit la propriété `serverLanguage` de l'objet renvoyé par un appel à la fonction `getServerInfo()` dans l'API de modèle de serveur.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui contient les langages de script pris en charge.

dom.serverModel.getServerName()

Disponibilité

Dreamweaver 1 ; amélioré dans la version Dreamweaver MX

Description

Extrait le nom du serveur associé au document et renvoie cette valeur. Le nom de serveur varie selon les technologies de serveur (comme ASP.NET et JSP) et non selon les langages d'une même technologie de serveur (comme ASP.NET VB et ASP.NET C#). Parmi les valeurs possibles, on retrouve `ASP`, `ASP.NET`, `Cold Fusion`, `JSP` et `PHP`.

Pour extraire le nom de modèle de serveur associé au document, voir *dom.serverModel.getDisplayName()*, page 422 ou *dom.serverModel.getFolderName()*, page 423.

REMARQUE

Pour Dreamweaver MX ou version ultérieure, la fonction `dom.serverModel.getServerName()` lit la propriété `serverName` de l'objet renvoyé par un appel à la fonction `getServerInfo()` dans l'API de modèle de serveur.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui contient le nom du serveur.

dom.serverModel.getServerSupportsCharset()

Disponibilité

Dreamweaver MX.

Description

Détermine si le modèle de serveur associé au document prend en charge le jeu de caractères nommé.

REMARQUE

Dreamweaver vous permet d'appeler cette fonction à partir du calque JavaScript mais aussi lorsque l'utilisateur modifie l'encodage dans la boîte de dialogue des propriétés. Si le modèle de serveur ne prend pas en charge le nouvel encodage des caractères, cette fonction renvoie la valeur `false` et Dreamweaver affiche une boîte de dialogue d'avertissement demandant à l'utilisateur s'il souhaite faire la conversion. Cela peut se produire par exemple lorsqu'un utilisateur essaie de convertir un document ColdFusion 4.5 en UTF-8, car ColdFusion ne prend pas en charge l'encodage UTF-8.

Arguments

`metaCharSetString`

- L'argument *`metaCharSetString`* est une valeur de chaîne qui dénomme un jeu de caractères donné. Cette valeur est la même que celle de l'attribut `"charset="` d'une balise `meta` associée à un document. Les valeurs prises en charge pour un modèle de serveur donné sont définies dans le fichier de définition HTML pour le modèle de serveur, situé dans le dossier Configuration/ServerModels.

Valeurs renvoyées

Valeur booléenne : `true` si le modèle de serveur prend en charge le jeu de caractères nommé ; `false` dans le cas contraire.

dom.serverModel.getServerVersion()

Disponibilité

Dreamweaver 1 ; amélioré dans la version Dreamweaver MX

Description

Détermine le modèle de serveur associé au document et renvoie cette valeur. Chaque modèle de serveur a une fonction `getVersionArray()` (dans l'API de modèle de serveur), qui renvoie un tableau de couples nom-version.

REMARQUE	Pour Dreamweaver, la fonction `dom.serverModel.getServerVersion()` lit d'abord la propriété `serverVersion` de l'objet renvoyé par un appel à la fonction `getServerInfo()` dans l'API de modèle de serveur. Si cette propriété n'existe pas, la fonction `dom.serverModel.getServerVersion()` la lit à partir de la fonction `getVersionArray()`.

Arguments

`name`

- L'argument *`name`* est une chaîne représentant le nom d'un modèle de serveur.

Valeurs renvoyées

Chaîne qui contient la version du modèle de serveur nommé.

dom.serverModel.testAppServer()

Disponibilité

Dreamweaver MX.

Description

Teste s'il est possible d'établir une connexion au serveur d'application.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne qui indique si la demande de connexion au serveur d'application a été acceptée.

dreamweaver.getServerModels()

Disponibilité

Dreamweaver MX.

Description

Obtient les noms de tous les modèles de serveur actuellement définis. Les noms sont les mêmes que ceux qui apparaissent dans la zone de texte Modèle de serveur de la boîte de dialogue Définition du site.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes. Tous les éléments de chaîne contiennent le nom d'un modèle de serveur actuellement défini.

CHAPITRE 16

Conception

16

Les fonctions de Conception de Macromedia Dreamweaver 8 permettent d'effectuer des opérations en rapport avec la personnalisation de l'apparence d'un document. Ces opérations incluent des fonctions permettant d'appliquer le style de feuille de style en cascade (CSS) spécifié, de fractionner le cadre sélectionné verticalement ou horizontalement, d'aligner les calques ou zones réactives sélectionnés, d'exécuter le plug-in sélectionné, de créer une cellule de Mise en forme ou de manipuler des lignes ou des colonnes d'un tableau.

fonctions CSS

Ces fonctions permettent d'appliquer, de retirer, de créer et de supprimer des styles CSS. Les méthodes de l'objet `dreamweaver.cssRuleTracker` contrôlent ou agissent sur l'élément sélectionné dans le panneau de suivi des règles CSS de l'inspecteur de sélections. Les méthodes de l'objet `dreamweaver.cssStylePalette` contrôlent ou agissent sur la sélection dans le panneau Styles, et non dans le document actif.

cssStylePalette.getInternetExplorerRendering()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si Dreamweaver affiche un rendu pour Internet Explorer.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si Dreamweaver affiche un rendu pour Internet Explorer ; `false` si Dreamweaver affiche un rendu selon la spécification CSS.

Exemple

L'exemple suivant désactive le rendu pour Internet Explorer :

```
if (cssStylePalette.getInternetExplorerRendering()){
  cssStylePalette.setInternetExplorerRendering(false);
  }
```

cssStylePalette.setInternetExplorerRendering()

Disponibilité

Dreamweaver 8.

Description

Cette fonction demande à Dreamweaver d'afficher un rendu pour Internet Explorer ou selon la spécification CSS.

Arguments

internetExplorerRendering

- L'argument obligatoire *internetExplorerRendering* indique s'il faut afficher un rendu pour Internet Explorer (`true`) ou selon la spécification CSS (`false`).

Valeurs renvoyées

Aucune.

Exemple

Voir *cssStylePalette.getInternetExplorerRendering()*, page 431.

dom.applyCSSStyle()

Disponibilité

Dreamweaver 4.

Description

Applique le style spécifié à l'élément spécifié. Cette fonction n'est valide que pour le document actif.

Arguments

elementNode, styleName, {classOrID}, {bForceNesting}

- L'argument *elementNode* désigne un nœud d'élément dans le DOM. Si l'argument *elementNode* a pour valeur NULL ou est exprimé sous forme d'une chaîne vide (""), la fonction agit sur la sélection en cours.
- L'argument *styleName* est le nom d'un style CSS.
- L'argument facultatif *classOrID* est l'attribut avec lequel le style doit être appliqué ("class" ou "id"). Si l'argument *elementNode* a pour valeur NULL ou s'il est exprimé sous forme d'une chaîne vide et qu'aucune balise n'entoure complètement la sélection, le style est appliqué avec des balises SPAN. Si la sélection est un point d'insertion, Dreamweaver utilise la méthode heuristique pour déterminer à quelle balise le style doit être appliqué.
- L'argument facultatif *bForceNesting* est une valeur booléenne qui indique si l'imbrication est autorisée ou non. Si le drapeau *bForceNesting* est défini, Dreamweaver insère une nouvelle balise SPAN au lieu de tenter de modifier les balises existantes dans le document. S'il n'est pas défini, cet argument a la valeur false par défaut.

Valeurs renvoyées

Aucune.

Exemple

Le code suivant applique le style red à la sélection, soit en entourant cette dernière de balises SPAN, soit en appliquant un attribut CLASS aux balises qui entourent la sélection :

```
var theDOM = dreamweaver.getDocumentDOM('document');
theDOM.applyCSSStyle('','red');
```

dom.getElementView()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie le mode Elément pour l'élément sélectionné dans le document. Si l'élément sélectionné est normal, la fonction getElementView() recherche le premier ancêtre de l'élément sélectionné qui est soit full soit hidden.

Arguments

Aucun.

Valeurs renvoyées

Une chaîne indiquant l'état de l'élément sélectionné. Ces valeurs sont :

- `"hidden"`, indique que l'élément possède des propriétés CSS pouvant entraîner un affichage masqué partiel ou total du contenu en mode Création. Les propriétés CSS prises en charge sont les suivantes :
  - débordement : hidden, scroll ou auto
  - affichage : aucune
- `"full"`, indique que par défaut l'élément est `"hidden"`, mais que la valeur actuelle est `"full"` comme définie par la fonction setElementView("full").
- `"normal"`, indique que l'élément n'a pour valeur ni `"hidden"` ni `"full"`.

Exemple

L'exemple suivant modifie l'état de l'élément sélectionné en `"full"` s'il a pour valeur `"hidden"`:

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM && getElementView() == "hidden"){
  currentDOM.setElementView("full");
  }
```

dom.getShowDivBackgrounds()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie l'état de l'assistance visuelle Arrière-plans des blocs de mise en forme.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne ; `true` si l'assistance visuelle Arrière-plans des blocs de mise en forme est activée ; `false` dans le cas contraire.

Exemple

L'exemple suivant vérifie si l'assistance visuelle Arrière-plans des blocs de mise en forme est activée et, si ce n'est pas le cas, l'active :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.getShowDivBackgrounds() == false){
  currentDOM.setShowDivBackgrounds(true);
  }
```

dom.getShowDivBoxModel()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie l'état de l'assistance visuelle Modèle de boîte des blocs de mise en forme.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne ; `true` si l'assistance visuelle Modèle de boîte des blocs de mise en forme est activée ; `false` dans le cas contraire.

Exemple

L'exemple suivant vérifie si l'assistance visuelle Modèle de boîte des blocs de mise en forme est activée et, si ce n'est pas le cas, l'active :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.getShowDivBoxModel() == false){
  currentDOM.setShowDivBoxModel(true);
  }
```

dom.getShowDivOutlines()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie l'état de l'assistance visuelle Contours des blocs de mise en forme.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne ; `true` si l'assistance visuelle Contours des blocs de mise en forme est activée ; `false` dans le cas contraire.

Exemple

L'exemple suivant vérifie si l'assistance visuelle Contours des blocs de mise en forme est activée et, si ce n'est pas le cas, l'active :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.getShowDivOutlines() == false){
  currentDOM.setShowDivOutlines(true);
  }
```

dom.removeCSSStyle()

Disponibilité

Dreamweaver 3.

Description

Supprime l'attribut `CLASS` ou `ID` de l'élément spécifié ou supprime les balises `SPAN` qui entourent complètement l'élément spécifié. Cette fonction n'est valide que pour le document actif.

Arguments

`elementNode, [classOrID]`

- L'argument *`elementNode`* désigne un nœud d'élément dans le DOM. Si l'argument *`elementNode`* est une chaîne vide (""), la fonction est appliquée à la sélection en cours.
- L'argument facultatif *`classOrID`* est l'attribut qui doit être supprimé (`"class"` ou `"id"`). Si l'argument *`classOrID`* n'est pas spécifié, il prend par défaut la valeur `"class"`. Si aucun attribut `CLASS` n'est défini pour l'argument *`elementNode`*, la balise `SPAN` entourant l'argument *`elementNode`* est supprimée.

Valeurs renvoyées

Aucune.

dom.resetAllElementViews()

Disponibilité

Dreamweaver 8.

Description

Cette fonction rétablit le mode Elément original de tous les éléments du document en supprimant tous les styles CSS générés en interne.

Arguments

forceRefresh

- L'argument facultatif *forceRefresh* est une valeur booléenne qui indique s'il faut actualiser le rendu de la totalité du document lorsqu'il n'y a pas de style CSS interne à supprimer. La valeur `true` provoque l'actualisation. La valeur par défaut est `false`.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant rétablit le mode Elément de tous les éléments du document sans avoir à actualiser le rendu :

```
var currentDOM = dw.getDocumentDOM();
currentDOM.resetAllElementViews(false);
```

dom.setElementView()

Disponibilité

Dreamweaver 8.

Description

Cette fonction définit le mode Elément pour l'élément sélectionné dans le document. Si l'élément sélectionné est `"normal"`, la fonction `setElementView()` recherche le premier ancêtre de l'élément sélectionné qui est `"full"` ou `"hidden"`.

Arguments

view

- L'argument obligatoire *view* est une chaîne qui définit l'élément sélectionné comme `"full"` ou `"hidden"`. Si l'élément sélectionné est `"normal"`, la fonction `setElementView()` recherche le premier ancêtre de l'élément sélectionné, qui correspond soit à `"full"` soit à `"hidden"`. Pour des informations supplémentaires, voir *dom.getElementView()*, page 433. Les valeurs possibles sont les suivantes :
  - `"full"` — Supprime le style CSS interne qui place l'élément en mode `"full"` pour permettre à l'élément de retrouver son état original.
  - `"hidden"` — Si l'élément sélectionné est en mode "hidden", Dreamweaver génère le style CSS permettant d'afficher tout le contenu puis l'applique comme une feuille de style interne au moment de la conception.

Valeurs renvoyées

Aucune.

Exemple

Voir *dom.getElementView()*, page 433.

dom.setShowDivBackgrounds()

Disponibilité

Dreamweaver 8.

Description

Cette fonction active ou désactive l'assistance visuelle Arrière-plans des blocs de mise en forme.

Arguments

show

- L'argument obligatoire *show* est une valeur booléenne qui indique s'il faut activer l'assistance visuelle Arrière-plans des blocs de mise en forme. Définir l'argument *show* sur `true` active l'assistance visuelle Arrière-plans des blocs de mise en forme.

Valeurs renvoyées

Aucune.

Exemple

Voir *dom.getShowDivBackgrounds()*, page 434.

dom.setShowDivBoxModel()

Disponibilité

Dreamweaver 8.

Description

Cette fonction active ou désactive l'assistance visuelle Modèle de boîte des blocs de mise en forme.

Arguments

show

- L'argument obligatoire *show* est une valeur booléenne qui indique s'il faut activer l'assistance visuelle Modèle de boîte des blocs de mise en forme. Définir l'argument *show* sur `true` active l'assistance visuelle Modèle de boîte des blocs de mise en forme.

Valeurs renvoyées

Aucune.

Exemple

Voir *dom.getShowDivBoxModel()*, page 435.

dom.setShowDivOutlines()

Disponibilité

Dreamweaver 8.

Description

Cette fonction active ou désactive l'assistance visuelle Contours des blocs de mise en forme.

Arguments

show

- L'argument obligatoire *show* est une valeur booléenne qui indique s'il faut activer l'assistance visuelle Contours des blocs de mise en forme. Définir l'argument *show* sur `true` active l'assistance visuelle Contours des blocs de mise en forme.

Valeurs renvoyées

Aucune.

Exemple

Voir *dom.getShowDivOutlines()*, page 435.

dreamweaver.cssRuleTracker.editSelectedRule()

Disponibilité

Dreamweaver MX 2004.

Description

Permet à l'utilisateur de modifier la règle sélectionnée dans l'outil de suivi des règles. Cette fonction affiche la règle sélectionnée dans la grille des propriétés CSS et, si cela s'avère nécessaire, affiche la grille des propriétés ainsi que la palette qui la contient.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.cssRuleTracker.canEditSelectedRule()*, page 592.

dreamweaver.cssRuleTracker.newRule()

Disponibilité

Dreamweaver MX 2004.

Description

Ouvre la boîte de dialogue Nouveau style CSS pour permettre à l'utilisateur de créer une nouvelle règle.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.cssStylePalette.applySelectedStyle()

Disponibilité

Dreamweaver MX.

Description

Applique le style sélectionné au document actif ou à la feuille de style associée, en fonction de l'élément sélectionné dans le panneau Styles CSS.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.cssStylePalette.canApplySelectedStyle()*, page 593.

dreamweaver.cssStylePalette.attachStyleSheet()

Disponibilité

Dreamweaver 4.

Description

Affiche une boîte de dialogue qui permet aux utilisateurs d'attacher une feuille de style au document actif ou à l'une des feuilles de style qui y sont associées en fonction de l'élément sélectionné dans le panneau Styles CSS.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.cssStylePalette.deleteSelectedStyle()

Disponibilité

Dreamweaver 3.

Description

Supprime du document le style actuellement sélectionné dans le panneau Styles CSS.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.cssStylePalette.canDeleteSelectedStyle()*, page 593.

dreamweaver.cssStylePalette.duplicateSelectedStyle()

Disponibilité

Dreamweaver 3.

Description

Duplique le style actuellement sélectionné dans le panneau Styles CSS et affiche la boîte de dialogue Dupliquer le style afin de permettre à l'utilisateur d'attribuer un nom ou un sélecteur au nouveau style.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.cssStylePalette.canDuplicateSelectedStyle()*, page 594.

dreamweaver.cssStylePalette.editSelectedStyle()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Définition du style correspondant au style sélectionné dans le panneau Styles CSS.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.cssStylePalette.canEditSelectedStyle()*, page 595.

dreamweaver.cssStylePalette.editSelectedStyleInCodeview()

Disponibilité

Dreamweaver 8.

Description

Cette fonction affiche le mode Code et déplace le curseur de la souris jusqu'au code du style actuellement sélectionné dans le panneau Styles.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.cssStylePalette.canEditSelectedStyleInCodeview()*, page 595.

dreamweaver.cssStylePalette.editStyleSheet()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Modifier feuille de style.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.cssStylePalette.canEditStyleSheet()*, page 596.

dreamweaver.cssStylePalette.getDisplayStyles()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si les styles CSS sont rendus. La valeur par défaut est `true`.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les styles CSS sont rendus ; `false` dans le cas contraire.

Exemple

```
var areStylesRendered = dw.cssStylePalette.getDisplayStyles();
```

dreamweaver.cssStylePalette.getMediaType()

Disponibilité

Dreamweaver MX 2004.

Description

Permet d'obtenir le type de média cible pour le rendu. Le type de média par défaut est `"screen"`.

Arguments

Aucun.

Valeurs renvoyées

Chaîne indiquant le type de média cible.

Exemple

```
var mediaType = dw.cssStylePalette.getMediaType();
```

dreamweaver.cssStylePalette.getSelectedStyle()

Disponibilité

Dreamweaver 3.0 ; `fullSelector` disponible dans Dreamweaver MX.

Description

Renvoie le nom du style actuellement sélectionné dans le panneau Styles CSS.

Arguments

`fullSelector`

- L'argument *fullSelector* est une valeur booléenne qui indique si le sélecteur complet ou la classe uniquement doivent être renvoyés. Si rien n'est précisé, seul le nom de classe est renvoyé. Par exemple, `p.class1` est un sélecteur qui signifie que le style est appliqué à n'importe quelle balise `p` de `class1`, mais il ne s'applique pas, par exemple, à une balise `div` de `class1`. Sans l'argument *fullSelector*, la fonction `dreamweaver.cssStylePalette.getSelectedStyle()` renvoie uniquement le nom de classe, `class1`, pour le sélecteur. L'argument *fullSelector* ordonne à la fonction de renvoyer `p.class1` au lieu de `class1`.

Valeurs renvoyées

Lorsque l'argument *fullSelector* a la valeur `true`, la fonction renvoie soit le sélecteur complet, soit une chaîne vide lorsque le nœud de la feuille de style est sélectionné.

Lorsque l'argument *fullSelector* a la valeur `false` ou est ignoré, une chaîne qui représente le nom de classe du style sélectionné est renvoyée. Si le style sélectionné n'a pas de classe ou si un nœud de feuille de style est sélectionné, une chaîne vide est renvoyée.

Exemple

Si le style `red` est sélectionné, un appel à la fonction `dw.cssStylePalette.getSelectedStyle()` renvoie `"red"`.

dreamweaver.cssStylePalette.getSelectedTarget() (déconseillée)

Disponibilité

Dreamweaver 3, déconseillée dans Dreamweaver MX, car le menu Appliquer à n'existe plus dans le panneau Styles.

Description

Cette fonction permet d'obtenir l'élément sélectionné dans le menu déroulant Appliquer à, en haut du panneau Styles.

Arguments

Aucun.

Valeurs renvoyées

Fonction ancienne ; renvoie toujours une valeur `null`.

dreamweaver.cssStylePalette.getStyles()

Disponibilité

Dreamweaver 3.

Description

Renvoie la liste de tous les styles de classe que contient le document actif. Sans arguments, il renvoie simplement les noms de sélecteurs de classes. Si l'argument *bGetIDs* est `true`, il renvoie simplement les noms des sélecteurs d'ID. Dans les deux cas, si l'argument *bGetFullSelector* est `true`, il renvoie le noms de sélecteur complet.

Prenons par exemple un fichier HTML avec le code suivant :

```
<style>
.test{ background:none };
p.foo{ background:none };
#bar {background:none };
div#hello p.world {background:none};
```

Les appels dans le tableau suivant renvoient les valeurs dans la colonne Résultat.

Appel de fonction	Résultat
dw.cssStylePalette.getStyles()	foo,test,world
dw.cssStylePalette.getStyles(true)	bar,hello
dw.cssStylePalette.getStyles(false, true)	p.foo,.test,div#hello p.world
dw.cssStylePalette.getStyles(true, true)	#bar,div#hello p.world

Arguments

{bGetIDs, bGetFullSelector}

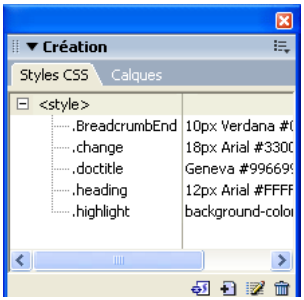
- L'argument *bGetIDs* est facultatif. Il s'agit d'une valeur booléenne qui, si défini sur `true`, oblige la fonction à renvoyer uniquement les noms des sélecteurs d'ID (la partie située après le "#"). La valeur par défaut est `false`.
- L'argument *bGetFullSelector* est facultatif. Il s'agit d'une valeur booléenne qui, si défini sur `true`, renvoie la chaîne du sélecteur complète, et non uniquement les noms. La valeur par défaut est `false`.

Valeurs renvoyées

Tableau de chaînes représentant les noms de tous les styles de classe contenus dans le document.

Exemple

Si la configuration du panneau Styles est calquée sur l'exemple suivant, un appel à la fonction `dreamweaver.cssStylePalette.getStyles()` renvoie un tableau qui contient ces chaînes : `"BreadcrumbEnd"`, `"change"`, `"doctitle"`, `"heading"` et `"highlight"` :



dreamweaver.cssStylePalette.newStyle()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Nouveau style.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.cssStylePalette.renameSelectedStyle()

Disponibilité

Dreamweaver 3.

Description

Modifie le nom de classe utilisé dans la règle sélectionnée dans le panneau Styles ainsi que toutes les instances du nom de classe figurant dans la règle sélectionnée.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.cssStylePalette.canRenameSelectedStyle()*, page 596.

dreamweaver.cssStylePalette.setDisplayStyles()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si les styles CSS doivent être rendus et actualise le rendu de tous les documents ouverts.

Arguments

display

- L'argument *display* doit avoir une valeur booléenne : `true` pour rendre les styles CSS ; false dans le cas contraire.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant désactive le rendu des styles CSS :

```
dw.cssStylePalette.setDisplayStyles(false);
```

dreamweaver.cssStylePalette.setMediaType()

Disponibilité

Dreamweaver MX 2004.

Description

Permet de définir le type de média cible pour le rendu. Actualise le rendu de tous les documents ouverts.

Arguments

mediaType

- L'argument *mediaType* indique le nouveau type de média cible.

Valeurs renvoyées

Aucune.

Exemple

```
dw.cssStylePalette.setMediaType("print");
```

dreamweaver.getBlockVisBoxModelColors()

Disponibilité

Dreamweaver 8.

Description

Cette fonction permet d'obtenir les couleurs utilisées pour rendre le modèle de boîte d'un bloc sélectionné lorsque l'assistance visuelle Modèle de boîte des blocs de mise en forme est activée.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes contenant deux chaînes :

- `marginColor`, qui représente la valeur hexadécimale de la couleur RVB sous la forme #RRVVBB ;`paddingColor`, qui représente la valeur hexadécimale de la couleur RVB sous la forme #RRVVBB

Exemple

L'exemple suivant vérifie la valeur de la couleur des marges et du remplissage ; si aucune n'est blanche, elles sont toutes deux définies sur blanc :

```
var boxColors = dreamweaver.getBlockVisBoxModelColors();
if ((boxColors[0] != "#FFFFFF") || (boxColors[1] != "#FFFFFF")){
     currentDOM.setBlockVisBoxModelColors("#FFFFFF", "#FFFFFF");
     }
```

dreamweaver.getBlockVisOutlineProperties()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie les propriétés de contour pour l'assistance visuelle de représentation des blocs.

Arguments

forWhat

- L'argument obligatoire *forWhat* est une chaîne. Les valeurs possibles sont les suivantes :`"divs"`, `"selectedDiv"` ou `"layers"` Si l'argument *forWhat* a pour valeur `"divs"`, la fonction renvoie les propriétés utilisées pour l'assistance visuelle qui met en surbrillance tous les blocs de mise en forme. Si l'argument *forWhat* a pour valeur `"selectedDiv"`, la fonction renvoie les propriétés utilisées pour l'assistance visuelle qui met en surbrillance les blocs de mise en forme sélectionnés. La valeur `layers` définit les calques.

Valeurs renvoyées

Tableau de chaînes contenant trois chaînes :

- `color`, qui correspond à la valeur hexadécimale de la couleur RVB, sous la forme #RRVVBB
- `width`, qui indique la largeur en pixels
- `style`, dont la valeur est soit `"SOLID"`, `"DOTTED"`, `"DASHED"` ou `"OUTSET"`

Exemple

L'exemple suivant renvoie les propriétés de contour de `"divs"` et définit le style de contour sur `"SOLID"` :

```
var outlineStyle = dw.getBlockVisOutlineProperties("divs");
if (outlineStyle[2] != "SOLID"){
     dw.setBlockVisOutlineProperties("divs", outlineStyle[0],
  outlineStyle[1], "SOLID");
    }
```

dreamweaver.getDivBackgroundColors()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie les couleurs utilisées par l'assistance visuelle Arrière-plans des blocs de mise en forme.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes contenant les 16 couleurs, chaque couleur étant représentée par la valeur hexadécimale de la couleur RVB, sous la forme #RRVVBB.

Exemple

L'exemple suivant renvoie les couleurs d'arrière-plans utilisées par l'assistance visuelle Arrière-plans des blocs de mise en forme :

```
var backgroundColors = dreamweaver.getDivBackgroundColors();
```

dreamweaver.setBlockVisOutlineProperties()

Disponibilité

Dreamweaver 8.

Description

Cette fonction définit les propriétés de contour pour l'assistance visuelle de représentation des blocs.

Arguments

forWhat, color, width, [style]

- L'argument obligatoire *forWhat* est une chaîne indiquant l'utilisation prévue pour la couleur et la largeur spécifiées. Les valeurs possibles sont les suivantes :`"divs"`, `"selectedDiv"` ou `"layers"` Si la valeur est `"layers"`, la couleur et la largeur spécifiées sont utilisées pour mettre en surbrillance tous les calques lorsque l'assistance visuelle Contours des blocs de mise en forme est activée. Si la valeur est `"divs"`, les arguments *color* et *width* permettent de mettre en surbrillance tous les divs et autres blocs de mise en forme. Si la valeur est `"selectedDiv"`, les arguments *color* et *width* permettent de mettre en surbrillance tout div ou bloc de mise en forme sélectionné.
- L'argument obligatoire *color* est une chaîne qui contient la valeur hexadécimale indiquant la couleur RVB sous la forme #RRVVBB.
- L'argument obligatoire *width* est un entier qui indique la largeur du contour, en pixels.

- L'argument facultatif *style* est une chaîne qui indique le style du contour. Les valeurs possibles sont les suivantes : "SOLID", "DOTTED", "DASHED" et "OUTSET". La valeur "OUTSET" est applicable aux calques uniquement. Cet argument est ignoré lorsque la valeur de l'argument *forWhat* est "selectedDiv".

Valeurs renvoyées

Aucune.

Exemple

Voir *dreamweaver.getBlockVisOutlineProperties()*, page 451.

dreamweaver.setDivBackgroundColors()

Disponibilité

Dreamweaver 8.

Description

Cette fonction définit les couleurs utilisées par l'assistance visuelle Arrière-plans des blocs de mise en forme.

Arguments

colors

- L'argument obligatoire *colors* est un tableau de chaînes contenant toutes les couleurs, représentées par des valeurs hexadécimales sous la forme #RRVVBB. Ce tableau doit contenir 16 couleurs.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant vérifie qu'il y ait 16 couleurs maximum spécifiées comme couleurs d'arrière-plan div et, le cas échéant, définit les couleurs utilisées comme couleurs d'arrière-plan en fonction des nuances de gris :

```
var currentDOM = dw.getDocumentDOM();
var divColors = currentDOM.getDivBackgroundColors("divs");
var shadesOfGray = new Array["#000000", "#111111", "#222222", "#333333",¬
  "#444444", "#555555", "#666666", "#777777", "#888888", "#999999", ¬
  "#AAAAAA", "#BBBBBB", "#CCCCCC", "#DDDDDD", "#EEEEEE", "#FFFFFF"]¬
var howManyColors = divColors.length;
if howManyColors <= 16{
  for (var i = 0; i < howManyColors; i++)
```

```
{
  currentDOM.setDivBackgroundColors("divs", shadeOfGray[i]);
}
```

Fonctions relatives aux cadres et aux jeux de cadres

Ces fonctions permettent d'effectuer les opérations suivantes : obtenir le nom des cadres constituant un jeu de cadres et diviser un cadre en deux.

dom.getFrameNames()

Disponibilité

Dreamweaver 3.

Description

Renvoie la liste de tous les cadres nommés du jeu de cadres.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes où chaque chaîne correspond au nom d'un cadre dans le jeu de cadres en cours. La fonction ignore tous les cadres sans nom. Si aucun des cadres du jeu n'est nommé, un tableau vide est renvoyé.

Exemple

Pour un document contenant quatre cadres, dont deux sont nommés, un appel à la fonction `dom.getFrameNames()` pourrait renvoyer un tableau contenant les chaînes suivantes :

- `"navframe"`
- `"main_content"`

dom.isDocumentInFrame()

Disponibilité

Dreamweaver 4.

Description

Indique si le document actif est affiché à l'intérieur d'un jeu de cadres.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le document se trouve dans un jeu de cadres et `false` dans le cas contraire.

dom.saveAllFrames()

Disponibilité

Dreamweaver 4.

Description

Si un document est un jeu de cadres ou se trouve à l'intérieur d'un jeu de cadres, cette fonction enregistre tous les cadres ou les jeux de la fenêtre de document. Si le document spécifié ne se trouve pas dans un jeu de cadres, cette fonction enregistre le document. Cette fonction ouvre la boîte de dialogue Enregistrer sous pour tous les documents qui n'ont pas été enregistrés précédemment.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.splitFrame()

Disponibilité

Dreamweaver 3.

Description

Fractionne le cadre sélectionné verticalement ou horizontalement.

Arguments

`splitDirection`

- L'argument *splitDirection* est une chaîne qui doit indiquer l'une des directions suivantes : "up", "down", "left" ou "right".

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canSplitFrame()*, page 580.

Fonctions relatives aux calques et aux cartes graphiques

Ces fonctions permettent d'aligner, de redimensionner et de déplacer des calques et des zones réactives de carte d'images. Il est indiqué, dans la description de chaque fonction, si cette dernière s'applique aux calques ou aux zones réactives.

dom.align()

Disponibilité

Dreamweaver 3.

Description

Aligne à gauche, à droite, en haut ou en bas les calques ou zones réactives sélectionnés.

Arguments

alignDirection

- L'argument *alignDirection* est une chaîne qui représente le bord par rapport auquel aligner les calques ou les zones réactives, à savoir "left", "right", "top" ou "bottom".

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canAlign()*, page 569.

dom.arrange()

Disponibilité

Dreamweaver 3.

Description

Déplace les zones réactives sélectionnées dans le sens indiqué.

Arguments

toBackOrFront

- L'argument *toBackOrFront* correspond au sens du déplacement des zones réactives, à savoir front ou back.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canArrange()*, page 570.

dom.makeSizesEqual()

Disponibilité

Dreamweaver 3.

Description

Affecte aux calques ou aux zones réactives sélectionnés la même hauteur ou la même largeur, ou les deux. Le dernier calque ou la dernière zone réactive sélectionnée sert de guide.

Arguments

bHoriz, bVert

- L'argument *bHoriz* est une valeur booléenne qui indique si les calques ou les zones réactives doivent être redimensionnés horizontalement.
- L'argument *bVert* est une valeur booléenne qui indique si les calques ou les zones réactives doivent être redimensionnés verticalement.

Valeurs renvoyées

Aucune.

dom.moveSelectionBy()

Disponibilité

Dreamweaver 3.

Description

Déplace les calques ou les zones réactives sélectionnés horizontalement et verticalement du nombre de pixels spécifiés.

Arguments

`x, y`

- L'argument *x* est le nombre de pixels dont la sélection doit être déplacée horizontalement.
- L'argument *y* est le nombre de pixels dont la sélection doit être déplacée verticalement.

Valeurs renvoyées

Aucune.

dom.resizeSelectionBy()

Disponibilité

Dreamweaver 3.

Description

Redimensionne le calque ou la zone réactive sélectionnée.

Arguments

left, top, bottom, right

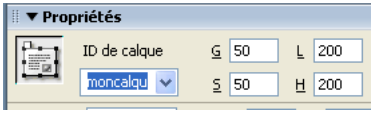
- L'argument *left* est la nouvelle position de la limite gauche du calque ou de la zone réactive.
- L'argument *top* est la nouvelle position de la limite supérieure du calque ou de la zone réactive.
- L'argument *bottom* est la nouvelle position de la limite inférieure du calque ou de la zone réactive.
- L'argument *right* est la nouvelle position de la limite droite du calque ou de la zone réactive.

Valeurs renvoyées

Aucune.

Exemple

Si le calque sélectionné possède les propriétés Gauche (left), Sommet (top), Largeur (width) et Hauteur (height) indiquées, un appel à la fonction `dw.getDocumentDOM().resizeSelectionBy(-10,-30,30,10)` revient à restaurer Gauche à 40, Sommet à 20, Largeur à 240 et Hauteur à 240.



dom.setLayerTag()

Disponibilité

Dreamweaver 3.

Description

Spécifie la balise HTML définissant le ou les calques sélectionnés.

Arguments

`tagName`

- L'argument *`tagName`* doit être `"layer"`, `"ilayer"`, `"div"` ou `"span"`.

Valeurs renvoyées

Aucune.

Fonctions d'environnement de mise en forme

Ces fonctions permettent d'effectuer des opérations associées aux paramètres d'utilisation d'un document. Elles permettent de modifier la source, la position et l'opacité du tracé de l'image, d'obtenir et de définir l'origine et les unités de mesure de la règle, d'activer et de désactiver la grille et d'en modifier les paramètres, de démarrer et d'arrêter l'exécution des plug-ins.

dom.getRulerOrigin()

Disponibilité

Dreamweaver 3.

Description

Renvoie l'origine de la règle.

Arguments

Aucun.

Valeurs renvoyées

Tableau contenant deux nombres entiers. Le premier entier correspond à la coordonnée *x* de l'origine, le second à la coordonnée *y*. Les deux valeurs sont exprimées en pixels.

dom.getRulerUnits()

Disponibilité

Dreamweaver 3.

Description

Renvoie les unités de mesure actuelles de la règle.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui contient l'une des valeurs suivantes :

- `"in"`
- `"cm"`
- `"px"`

dom.getTracingImageOpacity()

Disponibilité

Dreamweaver 3.

Description

Renvoie le paramètre d'opacité du tracé de l'image dans le document actif.

Arguments

Aucun.

Valeurs renvoyées

Valeur comprise entre 0 et 100, ou rien si l'opacité n'est pas définie.

Activateur

Voir *dom.hasTracingImage()*, page 581.

dom.loadTracingImage()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Sélectionner source de l'image. Si l'utilisateur sélectionne une image et clique sur OK, la boîte de dialogue Propriétés de la page s'ouvre et le champ Tracé de l'image contient une valeur.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.playAllPlugins()

Disponibilité

Dreamweaver 3.

Description

Exécute tous les plug-ins dans le document.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.playPlugin()

Disponibilité

Dreamweaver 3.

Description

Exécute le plug-in sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canPlayPlugin()*, page 577.

dom.setRulerOrigin()

Disponibilité

Dreamweaver 3.

Description

Définit l'origine de la règle.

Arguments

`xCoordinate, yCoordinate`

- L'argument *`xCoordinate`* est une valeur, exprimée en pixels, sur l'axe horizontal.
- L'argument *`yCoordinate`* est une valeur, exprimée en pixels, sur l'axe vertical.

Valeurs renvoyées

Aucune.

dom.setRulerUnits()

Disponibilité

Dreamweaver 3.

Description

Définit les unités de mesure de la règle.

Arguments

units

- L'argument *units* doit être "px", "in" ou "cm".

Valeurs renvoyées

Aucune.

dom.setTracingImagePosition()

Disponibilité

Dreamweaver 3.

Description

Déplace le coin supérieur gauche du tracé de l'image vers les coordonnées spécifiées. Si les arguments ne sont pas définis, la boîte de dialogue Ajuster la position du tracé de l'image s'affiche.

Arguments

x, y

- L'argument *x* est le nombre de pixels définissant la coordonnée horizontale.
- L'argument *y* est le nombre de pixels définissant la coordonnée verticale.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.hasTracingImage()*, page 581.

dom.setTracingImageOpacity()

Disponibilité

Dreamweaver 3.

Description

Définit le pourcentage d'opacité du tracé de l'image.

Arguments

opacityPercentage

- L'argument *opacityPercentage* doit être un nombre compris entre 0 et 100.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.hasTracingImage()*, page 581.

Exemple

L'exemple de code suivant règle l'opacité du tracé de l'image sur 30 %.

```
dw.getDocumentDOM().setTracingOpacity('30');
```

dom.snapTracingImageToSelection()

Disponibilité

Dreamweaver 3.

Description

Aligne le coin supérieur gauche du tracé de l'image avec le coin supérieur gauche de la sélection en cours.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.hasTracingImage()*, page 581.

dom.stopAllPlugins()

Disponibilité

Dreamweaver 3.

Description

Arrête l'exécution de tous les plug-ins en cours dans le document.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.stopPlugin()

Disponibilité

Dreamweaver 3.

Description

Arrête le plug-in sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne qui indique si la sélection est exécutée actuellement avec un plug-in.

Activateur

Voir *dom.canStopPlugin()*, page 580.

dreamweaver.arrangeFloatingPalettes()

Disponibilité

Dreamweaver 3.

Description

Déplace les panneaux flottants visibles vers leur position par défaut.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.showGridSettingsDialog()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Paramètres de la grille.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Fonctions relatives au mode de Mise en forme

Ces fonctions permettent de modifier les éléments de mise en forme d'un document. Elles s'appliquent aux paramètres de tableau, colonnes et cellules, y compris leur position, propriétés et aspect.

dom.addSpacerToColumn()

Disponibilité

Dreamweaver 4.

Description

Crée une image d'espacement transparente d'un pixel de hauteur au bas d'une colonne définie du tableau sélectionné. Cette fonction échoue si la sélection courante n'est pas un tableau.

Arguments

colNum

- L'argument *colNum* est la colonne au bas de laquelle l'image d'espacement est créée.

Valeurs renvoyées

Aucune.

dom.createLayoutCell()

Disponibilité

Dreamweaver 4.

Description

Crée une cellule de Mise en forme dans le document actif à la position et aux dimensions spécifiées, soit à l'intérieur d'un tableau de Mise en forme existant, soit dans une zone située sous le contenu existant sur la page. Si la cellule est créée dans un tableau de Mise en forme existant, elle ne doit ni chevaucher ni contenir d'autres cellules de Mise en forme ou des tableaux de Mise en forme imbriqués. Si le rectangle n'est pas à l'intérieur d'un tableau de Mise en forme existant, Dreamweaver tente de créer un tableau de Mise en forme devant contenir la nouvelle cellule. Cette fonction ne place pas le document en mode de Mise en forme. Elle échoue si la cellule ne peut pas être créée.

Arguments

left, top, width, height

- L'argument *left* est la position *x* de la bordure gauche de la cellule.
- L'argument *top* est la position *y* de la bordure supérieure de la cellule.
- L'argument *width* est la largeur de la cellule en pixels.
- L'argument *height* est la hauteur de la cellule en pixels.

Valeurs renvoyées

Aucune.

dom.createLayoutTable()

Disponibilité

Dreamweaver 4.

Description

Crée un tableau de Mise en forme dans le document actif à la position et aux dimensions spécifiées, soit à l'intérieur d'un tableau de mise en forme existant, soit dans une zone située sous le contenu existant sur la page. Si le tableau est créé dans un tableau de Mise en forme existant, il ne peut pas chevaucher d'autres cellules de Mise en forme ou des tableaux de Mise en forme imbriqués, mais il peut en contenir. Cette fonction ne place pas le document en mode de Mise en forme. Elle échoue si le tableau ne peut pas être créé.

Arguments

left, top, width, height

- L'argument *left* est la position *x* de la bordure gauche du tableau.
- L'argument *top* est la position *y* de la bordure supérieure du tableau.
- L'argument *width* est la largeur du tableau en pixels.
- L'argument *height* est la hauteur du tableau en pixels.

Valeurs renvoyées

Aucune.

dom.doesColumnHaveSpacer()

Disponibilité

Dreamweaver 4.

Description

Détermine si une colonne contient une image d'espacement générée par Dreamweaver. Cette fonction échoue si la sélection en cours n'est pas un tableau.

Arguments

colNum

- L'argument *colNum* est la colonne à vérifier pour une image d'espacement.

Valeurs renvoyées

Renvoie la valeur `true` si la colonne définie dans le tableau sélectionné contient une image d'espacement générée par Dreamweaver et `false` dans le cas contraire.

dom.doesGroupHaveSpacers()

Disponibilité

Dreamweaver 4.

Description

Détermine si le tableau sélectionné contient une ligne d'images d'espacement générées par Dreamweaver. Cette fonction échoue si la sélection en cours n'est pas un tableau.

Arguments

Aucun.

Valeurs renvoyées

Renvoie la valeur `true` si le tableau contient une ligne d'images d'espacement et `false` dans le cas contraire.

dom.getClickedHeaderColumn()

Disponibilité

Dreamweaver 4.

Description

Si l'utilisateur clique sur un bouton de menu dans l'en-tête d'un tableau du mode de Mise en forme, faisant ainsi apparaître le menu d'en-tête du tableau, cette fonction renvoie l'index de la colonne sur laquelle l'utilisateur a cliqué. Le résultat n'est pas défini si le menu d'en-tête du tableau n'est pas visible.

Arguments

Aucun.

Valeurs renvoyées

Nombre entier qui représente l'index de la colonne.

dom.getShowLayoutTableTabs()

Disponibilité

Dreamweaver 4.

Description

Détermine si le document actif affiche les tabulations des tableaux de Mise en forme dans le mode correspondant.

Arguments

Aucun.

Valeurs renvoyées

Renvoie la valeur `true` si le document actif affiche les tabulations des tableaux de Mise en forme dans le mode correspondant et `false` si tel n'est pas le cas.

dom.getShowLayoutView()

Disponibilité

Dreamweaver 4.

Description

Détermine le mode d'affichage du document actif, Mise en forme ou Standard.

Arguments

Aucun.

Valeurs renvoyées

Renvoie la valeur `true` si le document actif est en mode de Mise en forme et `false` s'il est en mode Standard.

dom.isColumnAutostretch()

Disponibilité

Dreamweaver 4.

Description

Détermine si une colonne doit être agrandie ou réduite automatiquement selon la taille du document. Cette fonction échoue si la sélection en cours n'est pas un tableau.

Arguments

`colNum`

- L'argument *`colNum`* est la colonne à redimensionner automatiquement ou dont la largeur est fixe.

Valeurs renvoyées

Renvoie la valeur `true` si la colonne située à l'index donné du tableau sélectionné doit s'agrandir automatiquement et `false` dans le cas contraire.

dom.makeCellWidthsConsistent()

Disponibilité

Dreamweaver 4.

Description

Dans le tableau sélectionné, cette fonction définit la largeur de chaque colonne de code HTML pour qu'elle corresponde à la largeur de rendu de la colonne. Cette fonction échoue si la sélection courante n'est pas un tableau.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.removeAllSpacers()

Disponibilité

Dreamweaver 4.

Description

Supprime toutes les images d'espacement générées par Dreamweaver à partir du tableau sélectionné. Cette fonction échoue si la sélection courante n'est pas un tableau.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.removeSpacerFromColumn()

Disponibilité

Dreamweaver 4.

Description

Supprime l'image d'espacement d'une colonne donnée et supprime la ligne d'espacement s'il n'y a plus d'images d'espacement générées par Dreamweaver. Cette fonction échoue si la sélection courante n'est pas un tableau.

Arguments

colNum

- L'argument *colNum* est la colonne dans laquelle l'image d'espacement doit être supprimée.

Valeurs renvoyées

Aucune.

dom.setColumnAutostretch()

Disponibilité

Dreamweaver 4.

Description

Passe d'une colonne automatiquement dimensionnée à une colonne d'une largeur fixe et vice versa. Si *bAutostretch* est défini sur `true`, la colonne située à l'index donné du tableau sélectionné est définie pour s'agrandir automatiquement ; dans le cas contraire, elle est définie sur une largeur fixe qui est celle du rendu en cours. Cette fonction échoue si la sélection courante n'est pas un tableau.

Arguments

colNum, bAutostretch

- L'argument *colNum* est la colonne à redimensionner automatiquement ou définie sur une largeur fixe.
- L'argument *bAutostretch* indique si la colonne doit être agrandie automatiquement (`true`) ou définie sur une largeur fixe (`false`).

Valeurs renvoyées

Aucune.

dom.getShowBlockBackgrounds()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie l'état de l'assistance visuelle qui active la coloration de l'arrière-plan de tous les blocs ou divs.

Arguments

allblocks

- L'argument obligatoire *allblocks* est une valeur booléenne. Définir la valeur sur `true` pour que l'argument ne soit appliqué qu'aux balises div. Définir la valeur sur `false` pour que l'argument soit appliqué à tous les éléments de bloc.

Valeurs renvoyées

Valeur booléenne. Si définie sur `true`, les arrière-plans sont imposés ; si définie sur `false`, les arrière-plans ne sont pas imposés.

Exemple

L'exemple suivant vérifie si la coloration d'arrière-plan de tous les blocs est imposée et, si ce n'est pas le cas, active la coloration d'arrière-plan de tous les blocs :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.getShowBlockBackgrounds(false) == false){
  currentDOM.setShowBlockBackgrounds(false);
  }
```

dom.getShowBlockBorders()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie l'état de l'assistance visuelle qui dessine les bordures de tous les blocs ou divs.

Arguments

`allblocks`

- L'argument obligatoire *`allblocks`* est une valeur booléenne. Définir la valeur sur `true` pour obtenir uniquement l'état des balises div. Définir la valeur sur `false` pour obtenir l'état de tous les éléments de bloc.

Valeurs renvoyées

Valeur booléenne ; si définie sur `true`, les bordures sont affichées ; si définie sur `false`, les bordures ne sont pas affichées.

Exemple

L'exemple suivant vérifie si l'assistance visuelle des bordures de blocs est activée et, si ce n'est pas le cas, l'active :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.getShowBlockBorders(false) == false){
  currentDOM.setShowBlockBorders(true);
  }
```

dom.getShowBlockIDs()

Disponibilité

Dreamweaver 8.

Description

Cette fonction renvoie l'état de l'assistance visuelle qui affiche les informations relatives à la classe et à l'ID de tous les blocs ou divs.

Arguments

allblocks

- L'argument obligatoire *allblocks* est une valeur booléenne. Définir la valeur sur `true` pour afficher uniquement la classe et l'ID des balises div. Définir la valeur sur `false` pour afficher la classe et l'ID de tous les éléments de bloc.

Valeurs renvoyées

Valeur booléenne : Si définie sur `true`, les ID sont affichés ; si définie sur `false`, les ID ne sont pas affichés.

Exemple

L'exemple suivant vérifie si les ID de blocs sont affichés et, si ce n'est pas le cas, les affiche :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.getShowBlockIDs(false) == false){
  currentDOM.setShowBlockIDs(true);
  }
```

dom.getShowBoxModel()

Disponibilité

Dreamweaver 8.

Description

Cette fonction active ou désactive l'assistance visuelle qui colore le modèle de boîte complet du bloc sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant vérifie si le modèle de boîte complet de la boîte sélectionnée est affiché en couleur et, si ce n'est pas le cas, le colore :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.getShowBoxModel() == false){
  currentDOM.setShowBoxModel(true);
  }
```

dom.setShowBlockBackgrounds()

Disponibilité

Dreamweaver 8.

Description

Cette fonction active ou désactive l'assistance visuelle qui commande la coloration de l'arrière-plan de tous les blocs ou divs.

Arguments

allblocks

- L'argument obligatoire *allblocks* est une valeur booléenne. Définir la valeur sur `true` pour appliquer la coloration d'arrière-plan aux balises div uniquement. Définir la valeur sur `false` pour appliquer la coloration d'arrière-plan à tous les éléments de bloc.

Valeurs renvoyées

Aucune.

Exemple

Voir *dom.getShowBlockBackgrounds()*, page 474.

dom.setShowBlockBorders()

Disponibilité

Dreamweaver 8.

Description

Cette fonction active ou désactive l'assistance visuelle qui dessine les bordures de tous les blocs ou divs.

Arguments

allblocks

- L'argument obligatoire *allblocks* est une valeur booléenne. Définir la valeur sur `true` pour appliquer les bordures aux balises div uniquement. Définir la valeur sur `false` pour appliquer les bordures à tous les éléments de bloc.

Valeurs renvoyées

Aucune.

Exemple

Voir *dom.getShowBlockBorders()*, page 474.

dom.setShowBlockIDs()

Disponibilité

Dreamweaver 8.

Description

Cette fonction active ou désactive l'assistance visuelle qui affiche l'ID et la classe de tous les blocs ou divs.

Arguments

allblocks

- L'argument obligatoire *allblocks* est une valeur booléenne. Définir la valeur sur `true` pour afficher la classe et l'ID des balises div uniquement. Définir la valeur sur `false` pour afficher la classe et l'ID de tous les éléments de bloc.

Valeurs renvoyées

Aucune.

Exemple

Voir *dom.getShowBlockIDs()*, page 475.

dom.setShowBoxModel()

Disponibilité

Dreamweaver 8.

Description

Cette fonction définit l'état de l'assistance visuelle qui colore le modèle de boîte complet du bloc sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le modèle de boîte est affiché ; `false` dans le cas contraire.

Exemple

Voir *dom.getShowBoxModel()*, page 476.

dom.setShowLayoutTableTabs()

Disponibilité

Dreamweaver 4.

Description

Définit le document actif pour qu'il affiche les tabulations des tableaux de Mise en forme chaque fois qu'il est en mode de Mise en forme. Cette fonction ne place pas le document en mode de Mise en forme.

Arguments

`bShow`

- L'argument *`bShow`* indique si les tabulations des tableaux de Mise en forme doivent s'afficher lorsque le document actif est en mode de Mise en forme. Si la valeur de l'argument *`bShow`* est `true`, Dreamweaver affiche les tabulations ; si la valeur de *`bShow`* est `false`, Dreamweaver ne les affiche pas.

Valeurs renvoyées

Aucune.

dom.setShowLayoutView()

Disponibilité

Dreamweaver 4.

Description

Met le document actif en mode de Mise en forme si *bShow* a pour valeur `true`.

Arguments

bShow

- L'argument *bShow* est une valeur booléenne qui fait passer le document actif du mode de Mise en forme au mode Standard, et vice versa. Si la valeur de l'argument *bShow* est `true`, le document actif est en mode de Mise en forme ; si la valeur de *bShow* est `false`, le document est en mode Standard.

Valeurs renvoyées

Aucune.

Fonctions relatives aux zooms

Fonctions de zoom avant et arrière en mode Création.

dreamweaver.activeViewScale()

Disponibilité

Dreamweaver 8.

Description

Cette fonction permet d'obtenir ou de définir une propriété de virgule flottante modifiable. Lorsque vous obtenez la valeur, Dreamweaver renvoie l'échelle de l'affichage actif tel qu'il apparaît dans la zone de liste modifiable Zoom, divisée par 100. Par exemple, 100 % correspond à 1.0 ; 50 % à 0.5, etc. Lorsque vous définissez la valeur, Dreamweaver définit la valeur dans la zone de liste modifiable Zoom. Cette valeur peut être comprise entre 0,06 et 64, ce qui correspond à 6 % et 6400 %.

Exemple

L'exemple suivant renvoie la valeur de l'échelle de l'affichage actif et effectue un zoom avant s'il cela est possible et si l'échelle est inférieure ou égale à 100 % :

```
if (canZoom() && dreamweaver.activeViewScale <= 1.0) {
  zoomIn();
  }
```

L'exemple suivant définit la valeur de l'échelle de l'affichage actif sur 50 % :

```
dreamweaver.activeViewScale = 0.50;
```

dreamweaver.fitAll()

Disponibilité

Dreamweaver 8.

Description

Cette fonction effectue un zoom avant ou arrière de manière à ce que la totalité du document s'adapte à la zone actuellement visible du mode Création.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canZoom()*, page 592.

Exemple

```
if (canZoom()){
  fitAll();
  }
```

dreamweaver.fitSelection()

Disponibilité

Dreamweaver 8.

Description

Cette fonction effectue un zoom avant ou arrière de manière à ce que la sélection en cours s'adapte à la zone actuellement visible du mode Création.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canFitSelection()*, page 586.

Exemple

```
if (canFitSeletion()){
  fitSelection();
  }
```

dreamweaver.fitWidth()

Disponibilité

Dreamweaver 8.

Description

Cette fonction effectue un zoom avant ou arrière de manière à ce que la largeur totale du document s'adapte à la zone actuellement visible du mode Création.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canZoom()*, page 592.

Exemple

```
if (canZoom()){
  fitWidth();
  }
```

dreamweaver.zoomIn()

Disponibilité

Dreamweaver 8.

Description

Cette fonction effectue un zoom avant sur la fenêtre de mode Création active. Le niveau de zoom correspond à la valeur prédéfinie dans le menu Zoom. Si aucune valeur n'est prédéfinie, cette fonction est inactive.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canZoom()*, page 592.

Exemple

```
if (canZoom()){
  zoomIn();
  }
```

dreamweaver.zoomOut()

Disponibilité

Dreamweaver 8.

Description

Cette fonction effectue un zoom arrière sur la fenêtre de mode Création active. Le niveau de zoom correspond à la valeur prédéfinie dans le menu Zoom. Si aucune valeur n'est prédéfinie, cette fonction est inactive.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canZoom()*, page 592.

Exemple

```
if (canZoom()){
  zoomOut();
  }
```

Propriétés et fonctions de guide

Les propriétés et fonctions de guide vous permettent d'afficher, de manipuler et de supprimer des guides utilisés par les utilisateurs pour mesurer et mettre en forme des éléments sur leurs pages HTML.

dom.clearGuides()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si tous les guides du document doivent être supprimés.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant supprime tous les guides contenus dans le document si ce dernier en contient au moins un :

```
var currentDOM = dw.getDocumentDOM();
  if (currentDOM.hasGuides() == true) {
  currentDOM.clearGuides();
}
```

dom.createHorizontalGuide()

Disponibilité

Dreamweaver 8.

Description

Cette fonction crée un guide horizontal à l'emplacement spécifié dans le document.

Arguments

`location`

- L'argument *`location`* représente l'emplacement du guide, qui correspond à une chaîne unique contenant à la fois la valeur et les unités de mesure sans espace entre les deux. Les unités possibles sont `"px"` pour les pixels et `"%"` pour les pourcentages. Par exemple, pour indiquer 10 pixels, `location = "10px"` ; pour indiquer 50 pourcent, `location = "50%"`.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant crée un guide horizontal à l'emplacement spécifié dans le document :

```
var currentDOM = dw.getDocumentDOM();
currentDOM.createHorizontalGuide("10px");
```

dom.createVerticalGuide()

Disponibilité

Dreamweaver 8.

Description

Cette fonction crée un guide vertical à l'emplacement spécifié dans le document.

Arguments

location

- L'argument *location* représente l'emplacement du guide, qui correspond à une chaîne unique contenant à la fois la valeur et les unités de mesure sans espace entre les deux. Les unités possibles sont `"px"` pour les pixels et `"%"` pour les pourcentages. Par exemple, pour indiquer 10 pixels, `location = "10px"` ; pour indiquer 50 pourcent, `location = "50%"`.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant crée un guide vertical à l'emplacement spécifié dans le document :

```
var currentDOM = dw.getDocumentDOM();
currentDOM.createVerticalGuide("10px");
```

dom.deleteHorizontalGuide()

Disponibilité

Dreamweaver 8.

Description

Cette fonction supprime le guide horizontal à l'emplacement spécifié.

Arguments

location

- L'argument *location* est une chaîne unique représentant l'emplacement à tester dans le document, contenant à la fois la valeur et les unités de mesure sans espace entre les deux. Les unités possibles sont `"px"` pour les pixels et `"%"` pour les pourcentages. Par exemple, pour indiquer 10 pixels, `location = "10px"` ; pour indiquer 50 pourcent, `location = "50%"`.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant supprime le guide horizontal à l'emplacement spécifié dans le document :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.hasHorizontalGuide("10px") == true) {
  currentDOM.deleteHorizonalGuide("10px");
}
```

dom.deleteVerticalGuide()

Disponibilité

Dreamweaver 8.

Description

Cette fonction supprime le guide vertical à l'emplacement spécifié.

Arguments

location

- L'argument *location* est une chaîne unique représentant l'emplacement à tester dans le document, contenant à la fois la valeur et les unités de mesure sans espace entre les deux. Les unités possibles sont `"px"` pour les pixels et `"%"` pour les pourcentages. Par exemple, pour indiquer 10 pixels, `location = "10px"` ; pour indiquer 50 pourcent, `location = "50%"`.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant supprime le guide vertical à l'emplacement spécifié dans le document :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.hasVerticalGuide("10px") == true) {
  currentDOM.deleteVerticalGuide("10px");
}
```

dom.guidesColor

Disponibilité

Dreamweaver 8.

Description

Cette propriété de couleur modifiable détermine la couleur des guides contenus dans le document. Vous pouvez définir et consulter cette propriété.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant affiche les guides en gris :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.guidesColor != "#444444"){
  currentDOM.guidesColor = "#444444";
  }
```

dom.guidesDistanceColor

Disponibilité

Dreamweaver 8.

Description

Cette propriété de couleur modifiable détermine la couleur des commentaires distants des guides contenus dans le document. Vous pouvez définir et consulter cette propriété.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant affiche la couleur des commentaires distants des guides en gris :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.guidesDistanceColor != "#CCCCCC"){
```

```
currentDOM.guidesDistanceColor = "#CCCCCC";
}
```

dom.guidesLocked

Disponibilité

Dreamweaver 8.

Description

Cette propriété booléenne modifiable détermine si les guides doivent être verrouillés dans le document. Vous pouvez définir et consulter cette propriété.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant verrouille les guides s'ils ne le sont pas :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.guidesLocked == false) {
  currentDOM.guidesLocked = true;
}
```

dom.guidesSnapToElements

Disponibilité

Dreamweaver 8.

Description

Cette propriété booléenne modifiable détermine s'il faut aligner les guides sur les éléments dans le document. Vous pouvez définir et consulter cette propriété.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant aligne les guides contenus dans le document sur les éléments :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.guidesSnapToElements == false) {
  currentDOM.guidesSnapToElements = true;
  }
```

dom.guidesVisible

Disponibilité

Dreamweaver 8.

Description

Cette propriété booléenne modifiable détermine si les guides sont visibles dans le document. Vous pouvez définir et consulter cette propriété.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant affiche les guides s'ils ne sont pas visibles :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.guidesVisible == false) {
  currentDOM.guidesVisible = true;
}
```

dom.hasGuides()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si le document contient au moins un guide. Vous pouvez définir et consulter cette propriété.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant supprime tous les guides contenus dans le document si ce dernier en contient au moins un :

```
var currentDOM = dw.getDocumentDOM();
  if (currentDOM.hasGuides() == true) {
  currentDOM.clearGuides();
}
```

dom.hasHorizontalGuide()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si le document contient un guide horizontal à l'emplacement spécifié.

Arguments

`location`

- L'argument *`location`* est une chaîne unique représentant l'emplacement à tester dans le document, contenant à la fois la valeur et les unités de mesure sans espace entre les deux. Les unités possibles sont `"px"` pour les pixels et `"%"` pour les pourcentages. Par exemple, pour indiquer 10 pixels, `location = "10px"` ; pour indiquer 50 pourcent, `location = "50%"`.

Valeurs renvoyées

Valeur booléenne : `true` s'il y a un guide horizontal à l'emplacement spécifié ; `false` dans le cas contraire.

Exemple

L'exemple suivant supprime tous les guides contenus dans le document si ce dernier contient un guide horizontal à l'emplacement spécifié :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.hasHorizontalGuide("10px") == true) {
  currentDOM.clearGuides();
}
```

dom.hasVerticalGuide()

Disponibilité

Dreamweaver 8.

Description

Cette fonction détermine si le document contient un guide vertical à l'emplacement spécifié.

Arguments

`location`

- L'argument *`location`* est une chaîne unique représentant l'emplacement à tester dans le document, contenant à la fois la valeur et les unités de mesure sans espace entre les deux. Les unités possibles sont `"px"` pour les pixels et `"%"` pour les pourcentages. Par exemple, pour indiquer 10 pixels, `location = "10px"` ; pour indiquer 50 pourcent, `location = "50%"`.

Valeurs renvoyées

Valeur booléenne : `true` s'il y a un guide vertical à l'emplacement spécifié ; `false` dans le cas contraire.

Exemple

L'exemple suivant supprime tous les guides contenus dans le document si ce dernier contient un guide vertical à l'emplacement spécifié :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.hasVerticalGuide("10px") == true) {
  currentDOM.clearGuides();
}
```

dom.snapToGuides

Disponibilité

Dreamweaver 8.

Description

Cette propriété booléenne modifiable détermine s'il faut aligner les éléments sur les guides dans le document. Vous pouvez définir et consulter cette propriété.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant aligne les éléments du document sur les guides :

```
var currentDOM = dw.getDocumentDOM();
if (currentDOM.snapToGuides == false) {
  currentDOM.snapToGuides = true;
}
```

Fonctions de modification des tableaux

Ces fonctions permettent d'insérer et de supprimer des lignes et des colonnes dans des tableaux, de modifier la largeur des colonnes et la hauteur des lignes, de convertir en pourcentage des mesures exprimées en pixels (et vice versa) et d'effectuer d'autres opérations standard de modification des tableaux.

dom.convertWidthsToPercent()

Disponibilité

Dreamweaver 3.

Description

Cette fonction convertit en pourcentage tous les attributs `WIDTH` exprimés en pixels du tableau en cours.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.convertWidthsToPixels()

Disponibilité

Dreamweaver 4.

Description

Cette fonction convertit en pixels tous les attributs `WIDTH` exprimés en pourcentage du tableau en cours.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.decreaseColspan()

Disponibilité

Dreamweaver 3.

Description

Cette fonction diminue l'étendue de colonnes de 1.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canDecreaseColspan()*, page 572.

dom.decreaseRowspan()

Disponibilité

Dreamweaver 3.

Description

Cette fonction diminue l'étendue de lignes de 1.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canDecreaseRowspan()*, page 573.

dom.deleteTableColumn()

Disponibilité

Dreamweaver 3.

Description

Cette fonction supprime du tableau la ou les colonnes sélectionnées.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canDeleteTableColumn()*, page 573.

dom.deleteTableRow()

Disponibilité

Dreamweaver 3.

Description

Cette fonction supprime du tableau la ou les lignes sélectionnées.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canDeleteTableRow()*, page 574.

dom.doDeferredTableUpdate()

Disponibilité

Dreamweaver 3.

Description

Si l'option Modification de tableau plus rapide est activée dans la catégorie Général des préférences, cette fonction oblige la mise en forme du tableau à adopter les modifications apportées récemment, et ce sans déplacer la sélection à l'extérieur du tableau. Cette fonction n'a aucun effet si l'option Modification de tableau plus rapide n'est pas activée.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.getShowTableWidths()

Disponibilité

Dreamweaver MX 2004.

Description

Indique si les largeurs d'un tableau s'affichent en mode Standard ou Tableaux développés (mode autre que Mise en forme). Pour plus d'informations sur l'affichage des tabulations de tableau en mode de Mise en forme dans Dreamweaver, voir *dom.getShowLayoutTableTabs()*, page 471.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si Dreamweaver affiche les largeurs du tableau en mode Standard ou Tableaux développés ; `false` dans le cas contraire.

dom.getTableExtent()

Disponibilité

Dreamweaver 3.

Description

Cette fonction renvoie le nombre de colonnes et de lignes du tableau sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Tableau contenant deux nombres entiers. Le premier entier correspond au nombre de colonnes, et le second au nombre de lignes. Si aucun tableau n'est sélectionné, aucune valeur n'est renvoyée.

dom.increaseColspan()

Disponibilité

Dreamweaver 3.

Description

Cette fonction augmente l'étendue de colonnes de 1.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canIncreaseColspan()*, page 575.

dom.increaseRowspan()

Disponibilité

Dreamweaver 3.

Description

Cette fonction augmente l'étendue de lignes de 1.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canDecreaseRowspan()*, page 573.

dom.insertTableColumns()

Disponibilité

Dreamweaver 3.

Description

Cette fonction insère le nombre de colonnes spécifié dans le tableau en cours.

Arguments

numberOfCols, bBeforeSelection

- L'argument *numberOfCols* est le nombre de colonnes à insérer.
- L'argument *bBeforeSelection* est une valeur booléenne : `true` indique que les colonnes doivent être insérées avant la colonne contenant la sélection ; `false` indique le contraire.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canInsertTableColumns()*, page 575.

dom.insertTableRows()

Disponibilité

Dreamweaver 3.

Description

Cette fonction insère le nombre de lignes spécifié dans le tableau en cours.

Arguments

numberOfRows, bBeforeSelection

- L'argument *numberOfRows* est le nombre de lignes à insérer.
- L'argument *bBeforeSelection* est une valeur booléenne : `true` indique que les lignes doivent être insérées au-dessus de la ligne contenant la sélection ; `false` indique le contraire.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canInsertTableRows()*, page 576.

dom.mergeTableCells()

Disponibilité

Dreamweaver 3.

Description

Cette fonction fusionne les cellules de tableau sélectionnées.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canMergeTableCells()*, page 577.

dom.removeAllTableHeights()

Disponibilité

Dreamweaver 3.

Description

Cette fonction supprime tous les attributs `HEIGHT` du tableau sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.removeAllTableWidths()

Disponibilité

Dreamweaver 3.

Description

Cette fonction supprime tous les attributs `WIDTH` du tableau sélectionné.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.removeColumnWidth()

Disponibilité

Dreamweaver MX 2004.

Description

Cette fonction supprime tous les attributs `WIDTH` d'une unique colonne sélectionnée.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.selectTable()

Disponibilité

Dreamweaver 3.

Description

Sélectionne un tableau entier.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canSelectTable()*, page 578.

dom.setShowTableWidths()

Disponibilité

Dreamweaver MX 2004.

Description

Permet d'activer ou de désactiver l'affichage des largeurs d'un tableau en mode Standard ou Tableaux développés (mode autre que Mise en forme). Cette fonction définit la valeur assignée au document en cours et à tous les documents *suivants*, sauf indication contraire. Pour plus d'informations sur la définition de l'affichage des tabulations de tableau en mode de Mise en forme dans Dreamweaver, voir *dom.setShowLayoutTableTabs()*, page 478.

Arguments

bShow

- *bShow* est un argument booléen qui indique si les largeurs de tableau doivent s'afficher lorsque le document en cours est en mode Standard ou Tableaux développés (mode autre que Mise en forme). Si la valeur de *bShow* est `true`, Dreamweaver affiche les largeurs. Si la valeur de *bShow* est `false`, Dreamweaver ne les affiche pas.

Valeurs renvoyées

Aucune.

dom.setTableCellTag()

Disponibilité

Dreamweaver 3.

Description

Cette fonction définit la balise de la cellule sélectionnée.

Arguments

tdOrTh

- L'argument *tdOrTh* doit être "td" ou "th".

Valeurs renvoyées

Aucune.

dom.setTableColumns()

Disponibilité

Dreamweaver 3.

Description

Cette fonction définit le nombre de colonnes du tableau sélectionné.

Arguments

numberOfCols

- L'argument *numberOfCols* indique le nombre de colonnes à définir dans le tableau.

Valeurs renvoyées

Aucune.

dom.setTableRows()

Disponibilité

Dreamweaver 3.

Description

Cette fonction définit le nombre de lignes du tableau sélectionné.

Arguments

numberOfCols

- L'argument *numberOfRows* indique le nombre de lignes à définir dans le tableau sélectionné.

Valeurs renvoyées

Aucune.

dom.showInsertTableRowsOrColumnsDialog()

Disponibilité

Dreamweaver 3.

Description

Cette fonction ouvre la boîte de dialogue Insérer des lignes ou des colonnes.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canInsertTableColumns()*, page 575 ou *dom.canInsertTableRows()*, page 576.

dom.splitTableCell()

Disponibilité

Dreamweaver 3.

Description

Cette fonction fractionne la cellule de tableau en cours en un nombre de lignes ou de colonnes donné. Si vous ne définissez pas l'un des deux arguments, ou les deux, la boîte de dialogue Fractionner la cellule s'affiche.

Arguments

{colsOrRows}, {numberToSplitInto}

- L'argument facultatif *colsOrRows* doit être "columns" ou "rows".
- L'argument facultatif *numberToSplitInto* indique en combien de lignes ou de colonnes la cellule doit être fractionnée.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canSplitTableCell()*, page 580.

CHAPITRE 17

Code

17

Les fonctions Code vous permettent d'effectuer des opérations sur un document affiché en mode Code. Ces opérations comprennent l'ajout de nouvelles balises de fonction ou de menu à un menu d'indicateurs de code, la recherche et le remplacement de modèles de chaînes, la suppression de la sélection en cours, l'impression de code complet ou sélectionné, la modification de balises ou l'application de formatage de syntaxe au code sélectionné.

Fonctions de code

Les indicateurs de code sont des menus contextuels qui s'affichent dans Macromedia Dreamweaver 8 lorsque vous tapez certains caractères en mode Code. Ils vous évitent de saisir tout le texte en proposant une liste de chaînes susceptibles de compléter la chaîne que vous tapez. Si la chaîne que vous tapez apparaît dans le menu, sélectionnez-la et appuyez sur Entrée ou Retour pour compléter votre saisie. Si vous tapez <, par exemple, un menu contextuel affiche une liste des noms de balises. Plutôt que de taper le reste du nom de la balise, vous pouvez la sélectionner dans le menu pour l'inclure à votre texte.

Vous pouvez ajouter des menus Indicateurs de code dans Dreamweaver en les définissant dans le fichier CodeHints.xml. Pour plus d'informations sur le fichier CodeHints.xml, voir *Extension de Dreamweaver.*

Vous pouvez également ajouter de nouveaux menus d'indicateurs de code de façon dynamique via JavaScript après chargement du contenu du fichier CodeHints.xml par Dreamweaver. Par exemple, le code JavaScript ajoute des données à la liste des variables de session dans le panneau Liaisons. Vous pouvez utiliser le même code pour ajouter un menu Indicateurs de code. Dans ce cas, Dreamweaver affiche un menu de variables de session lorsqu'un utilisateur tape **Session** en mode Code.

Le fichier CodeHints.xml et l'API JavaScript contiennent un sous-ensemble utile du moteur Indicateurs de code, mais certaines fonctionnalités Dreamweaver ne sont pas accessibles. Par exemple, comme il n'existe pas d'accroche JavaScript pouvant afficher un sélecteur de couleur, Dreamweaver ne peut pas exprimer le menu Valeurs des attributs à l'aide de JavaScript. Vous pouvez uniquement afficher un menu d'éléments de texte permettant d'insérer du texte.

La coloration du code vous permet de définir des styles de couleur de code et de modifier des modèles de coloration de code ou d'en créer de nouveaux. Vous pouvez spécifier ces styles et modèles en modifiant le fichier Colors.xml et celui de modèle de coloration. Pour plus d'informations sur ces fichiers, voir *Extension de Dreamweaver*.

L'API JavaScript pour les indicateurs et la coloration de code comprend les fonctions suivantes :

dreamweaver.codeHints.addMenu()

Disponibilité

Dreamweaver MX.

Description

Définit dynamiquement une nouvelle balise `menu` dans le fichier CodeHints.xml. Si une balise de menu est définie par le même modèle et le même type de document, cette fonction ajoute les éléments au menu existant.

Arguments

`menuGroupId, pattern, labelArray, {valueArray}, {iconArray}, {doctypes}, {casesensitive}`

- L'argument *`menuGroupId`* est l'attribut ID de l'une des balises `menugroup`.
- L'argument *`pattern`* est l'attribut de modèle de la nouvelle balise `menu`.
- L'argument *`labelArray`* est un tableau de chaînes. Chaque chaîne constitue le texte d'un élément de menu dans le menu contextuel.
- L'argument facultatif *`valueArray`* est un tableau de chaînes qui doit avoir la même longueur que l'argument *`labelArray`*. Lorsqu'un utilisateur sélectionne un élément dans le menu contextuel, la chaîne de ce tableau est insérée dans son document. Si la chaîne à insérer est toujours identique à celle de l'étiquette du menu, la valeur de cet argument peut être `null`.

- L'argument facultatif *iconArray* peut être une chaîne ou un tableau de chaînes. S'il s'agit d'une chaîne, elle spécifie l'URL d'un fichier d'image utilisé par Dreamweaver pour tous les éléments du menu. S'il s'agit d'un tableau de chaînes, il doit être de la même longueur que l'argument *labelArray*. Chaque chaîne est une URL (dossier Configuration de Dreamweaver) désignant un fichier d'image que Dreamweaver utilise comme une icône pour l'élément de menu correspondant. Si cet argument a la valeur `null`, Dreamweaver affiche le menu sans les icônes.
- L'argument facultatif *doctypes* indique que le menu est actif pour certains types de documents uniquement. Vous pouvez spécifier l'argument `doctypes` en tant que liste d'ID de types de documents séparés par des virgules. Pour obtenir une liste des types de documents Dreamweaver, voir le fichier Dreamweaver Configuration/Documenttypes/MMDocumentTypes.xml.
- L'argument facultatif *casesensitive* indique si le modèle fait la distinction entre les majuscules et les minuscules. Les valeurs possibles de l'argument *casesensitive* sont les valeurs booléennes `true` ou `false`. Par défaut, la valeur est `false` si vous omettez cet argument. Dans le cas où l'argument *casesensitive* aurait la valeur `true`, le menu d'indicateurs de code s'affiche uniquement si le texte entré par l'utilisateur correspond exactement au modèle spécifié par l'attribut de modèle. Dans le cas où l'argument *casesensitive* aurait la valeur `false`, le menu s'affiche, même si le modèle est en minuscules et que le texte est en majuscules.

Valeurs renvoyées

Aucune.

Exemple

Si l'utilisateur crée un jeu d'enregistrements appelé "`myRs`", le code suivant crée un menu pour `myRS`:

```
dw.codeHints.addMenu(
  "CodeHints_object_methods",  // menu is enabled if object methods are
  enabled
  "myRS.",                        // pop up menu if user types "myRS."
  new Array("firstName", "lastName"),  // items in drop-down menu for myRS
  new Array("firstName", "lastName"),  // text to actually insert in
  document
  null,              // no icons for this menu
  "ASP_VB, ASP_JS"); // specific to the ASP doc types
```

dreamweaver.codeHints.addFunction()

Disponibilité

Dreamweaver MX.

Description

Définit dynamiquement une nouvelle balise `function`. Si une balise `function` est définie par le même modèle et le même type de document, cette fonction remplace la balise `function` existante.

Arguments

`menuGroupId, pattern, {doctypes}, {casesensitive}`

- L'argument *`menuGroupId`* est l'attribut de chaîne d'ID d'une balise `menugroup`.
- L'argument *`pattern`* est une chaîne qui spécifie l'attribut de modèle de la nouvelle balise `function`.
- L'argument facultatif *`doctypes`* définit que cette fonction est active pour certains types de documents uniquement. Vous pouvez spécifier l'argument `doctypes` en tant que liste d'ID de types de documents séparés par des virgules. Pour obtenir une liste des types de documents Dreamweaver, voir le fichier Dreamweaver Configuration/Documenttypes/MMDocumentTypes.xml.
- L'argument facultatif *`casesensitive`* indique si le modèle fait la distinction entre les majuscules et les minuscules. Les valeurs possibles de l'argument `casesensitive` sont les valeurs booléennes `true` ou `false`. Par défaut, la valeur est `false` si vous omettez cet argument. Dans le cas où l'argument *`casesensitive`* aurait la valeur `true`, le menu d'indicateurs de code s'affiche uniquement si le texte entré par l'utilisateur correspond exactement au modèle spécifié par l'attribut de modèle. Si l'argument *`casesensitive`* est `false`, le menu s'affiche même si le modèle est en minuscules et le texte en majuscules.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant de la fonction `dw.codeHints.addFunction()` ajoute le modèle de nom de fonction `out.newLine()` au groupe de menu d'indicateurs de code `CodeHints_Object_Methods` et l'active uniquement pour les documents de type `JSP` :

```
dw.codeHints.addFunction(
  "CodeHints_Object_Methods",
  "out.newLine()",
  "JSP")
```

dreamweaver.codeHints.resetMenu()

Disponibilité

Dreamweaver MX.

Description

Réinitialise la balise de menu ou de fonction spécifiée à l'état dans lequel elle se trouvait juste après la lecture du fichier CodeHints.xml par Dreamweaver. En d'autres termes, l'appel de cette fonction annule l'effet des appels précédents des fonctions `addMenu()` et `addFunction()`.

Arguments

`menuGroupId, pattern, {doctypes}`

- L'argument *`menuGroupId`* est l'attribut de chaîne d'ID d'une balise `menugroup`.
- L'argument *`pattern`* est une chaîne qui spécifie l'attribut de modèle de la nouvelle balise `menu` ou `function` à réinitialiser.
- L'argument facultatif *`doctypes`* indique que le menu est actif pour certains types de documents uniquement. Vous pouvez spécifier l'argument `doctypes` en tant que liste d'ID de types de documents séparés par des virgules. Pour obtenir une liste des types de documents Dreamweaver, voir le fichier Dreamweaver Configuration/Documenttypes/MMDocumentTypes.xml.

Valeurs renvoyées

Aucune.

Exemple

Votre code JavaScript peut créer un menu Indicateurs de code qui contient toutes les variables de session définies par l'utilisateur. Dès que la liste des variables de session est modifiée, ce code met le menu à jour. Avant de charger la nouvelle liste de variables de session dans le menu, le code doit supprimer l'ancienne liste. L'appel de cette fonction supprime les anciennes variables de session.

dreamweaver.codeHints.showCodeHints()

Disponibilité

Dreamweaver MX.

Description

Dreamweaver appelle cette fonction lorsque l'utilisateur ouvre l'élément de menu Edition > Afficher les indicateurs de code. La fonction ouvre le menu Indicateurs de code à l'emplacement de la sélection en mode Code.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

L'exemple suivant ouvre le menu d'indicateurs de code au niveau du point d'insertion dans le document en mode Code.

```
dw.codeHints.showCodeHints()
```

dreamweaver.reloadCodeColoring()

Description

Recharge les fichiers de coloration de code dans le dossier Configuration/Code Coloring de Dreamweaver.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

```
dreamweaver.reloadCodeColoring()
```

Fonctions relatives à la recherche et au remplacement

Comme leur nom l'indique, ces fonctions permettent d'effectuer des recherches et des remplacements. Elles vont de la simple recherche de l'instance suivante d'une chaîne donnée à des opérations plus complexes de remplacement automatique.

dreamweaver.findNext()

Disponibilité

Dreamweaver 3 ; modifié dans Dreamweaver MX 2004.

Description

Recherche l'instance suivante de la chaîne de recherche précédemment définie par dreamweaver.setUpFind(), par dreamweaver.setUpComplexFind() ou par l'utilisateur (à l'aide de la boîte de dialogue Rechercher), puis la sélectionne dans le document.

Arguments

{bUseLastSetupSearch}

- L'argument facultatif *bUseLastSetupSearch* est une valeur booléenne. Si *bUseLastSetupSearch* a la valeur `true` (valeur par défaut si aucun argument n'est fourni), la fonction recherche l'occurrence suivante selon les paramètres définis lors de l'appel précédent à la fonction `dreamweaver.setupComplexFind()` ou à la fonction `dreamweaver.setupComplexFindReplace()`. Si vous définissez *bUseLastSetupSearch* sur la valeur `false`, la fonction ignore la recherche précédente et recherche l'instance suivante du texte sélectionné dans le document.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canFindNext()*, page 585.

dreamweaver.replace()

Disponibilité

Dreamweaver 3.

Description

Vérifie que la sélection actuelle correspond aux critères de recherche spécifiés par dreamweaver.setUpFindReplace(), par dreamweaver.setUpComplexFindReplace() ou par l'utilisateur dans la boîte de dialogue Remplacer ; la fonction remplace ensuite la sélection par le texte de substitution spécifié dans la demande de recherche.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.replaceAll()

Disponibilité

Dreamweaver 3.

Description

Remplace chaque section du document actif correspondant aux critères de recherche précédemment définis par la fonction dreamweaver.setUpFindReplace() ou dreamweaver.setUpComplexFindReplace() ou par l'utilisateur (dans la boîte de dialogue Remplacer) par le contenu de remplacement spécifié.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.setUpComplexFind()

Disponibilité

Dreamweaver 3.

Description

Prépare le terrain pour une recherche avancée de texte ou de balises en chargeant la requête XML spécifiée.

Arguments

`xmlQueryString`

- L'argument *`xmlQueryString`* est une chaîne de code XML commençant par `dwquery` et se terminant par `/dwquery`. Pour obtenir une chaîne formatée correctement, vous pouvez définir la requête à l'aide de la boîte de dialogue Rechercher, cliquer sur le bouton Enregistrer la requête, ouvrir ce fichier de requête dans un éditeur de texte et copier tout ce qui est compris entre le début de la balise `dwquery` et la fin de la balise `/dwquery`.

Valeurs renvoyées

Aucune.

Exemple

Dans l'exemple ci-après, la première ligne de code définit une recherche de balise et précise que la recherche doit porter sur le document actif ; la deuxième ligne exécute la recherche.

```
dreamweaver.setUpComplexFind('<dwquery><queryparams matchcase="false" ¬
ignorewhitespace="true" useregexp="false"/><find>¬
<qtag qname="a"><qattribute qname="href" qcompare="="
qvalue="#">¬
  </qattribute><qattribute qname="onMouseOut" qcompare="=" qvalue="" ¬
qnegate="true"></qattribute></qtag></find></dwquery>');
dw.findNext();
```

dreamweaver.setUpComplexFindReplace()

Disponibilité

Dreamweaver 3.

Description

Prépare le terrain pour une recherche avancée de texte ou de balises en chargeant la requête XML spécifiée.

Arguments

xmlQueryString

- L'argument *xmlQueryString* est une chaîne du code XML qui commence par la balise `dwquery` et se termine par la balise `/dwquery`. Pour obtenir une chaîne ayant formatée correctement, vous pouvez définir la requête à l'aide de la boîte de dialogue Rechercher, cliquer sur le bouton Enregistrer la requête, ouvrir ce fichier de requête dans un éditeur de texte et copier tout ce qui est compris entre le début de la balise `dwquery` et la fin de la balise `/dwquery`.

Valeurs renvoyées

Aucune.

Exemple

Dans l'exemple ci-après, la première ligne de code définit une recherche de balise et précise que la recherche doit porter sur quatre fichiers ; la deuxième ligne exécute la recherche et le remplacement.

```
dreamweaver.setUpComplexFindReplace('<dwquery><queryparams ¬
matchcase="false" ignorewhitespace="true" useregexp="false"/>¬
```

```
<find><qtag qname="a"><qattribute qname="href" qcompare="=" ¬
  qvalue="#"></qattribute><qattribute qname="onMouseOut" ¬
  qcompare="=" qvalue="" qnegate="true"></qattribute></qtag>¬
  </find><replace action="setAttribute" param1="onMouseOut" ¬
  param2="this.style.color='#000000';this.style.¬
fontWeight='normal'"/></dwquery>');
dw.replaceAll();
```

dreamweaver.setUpFind()

Disponibilité

Dreamweaver 3.

Description

Prépare le terrain pour l'exécution d'une recherche de texte ou de code source HTML en définissant les critères de recherche de l'opération `dreamweaver.findNext()` qui va suivre.

Arguments

`searchObject`

L'argument *`searchObject`* est un objet pour lequel les propriétés suivantes peuvent être définies :

- La propriété *`searchString`* est le texte à rechercher.
- La propriété *`searchSource`* est une valeur booléenne indiquant si la recherche doit également porter sur le code source HTML.
- La propriété facultative *`{matchCase}`* est une valeur booléenne indiquant si la recherche doit respecter les majuscules et les minuscules. Si cette propriété n'est pas définie, elle prend par défaut la valeur `false`.
- La propriété facultative *`{ignoreWhitespace}`* est une valeur booléenne indiquant si les différences entre les espaces blancs doivent être ignorées. La propriétés *`ignoreWhitespace`* prend sa valeur par défaut, `false`, si la valeur de la propriété *`useRegularExpressions`* est `true`, et `true` si la valeur de la propriété *`useRegularExpressions`* est `false`.
- La propriété *`{useRegularExpressions}`* est une valeur booléenne indiquant que la propriété `searchString` utilise des expressions régulières. Si cette propriété n'est pas définie, elle prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

Exemple

L'exemple de code suivant montre comment créer un objet à rechercher (*searchObject*) de trois façons différentes :

```
var searchParams;
searchParams.searchString = 'bgcolor="#FFCCFF"';
searchParams.searchSource = true;
dreamweaver.setUpFind(searchParams);

var searchParams = {searchString: 'bgcolor="#FFCCFF"', searchSource: true};
dreamweaver.setUpFind(searchParams);

dreamweaver.setUpFind({searchString: 'bgcolor="#FFCCFF"', searchSource: ¬
true});
```

dreamweaver.setUpFindReplace()

Disponibilité

Dreamweaver 3.

Description

Prépare une recherche de texte ou de code source HTML en définissant les critères de recherche et le cadre d'application de l'opération `dreamweaver.replace()` `dreamweaver.replaceAll()` qui va suivre.

Arguments

searchObject

L'argument *searchObject* est un objet pour lequel les propriétés suivantes peuvent être définies :

- La propriété *searchString* est le texte à rechercher.
- La propriété *replaceString* est le texte à substituer au texte recherché.
- La propriété *searchSource* est une valeur booléenne indiquant si la recherche doit également porter sur le code source HTML.
- La propriété facultative *{matchCase}* est une valeur booléenne indiquant si la recherche doit respecter les majuscules et les minuscules. Si cette propriété n'est pas définie, elle prend par défaut la valeur `false`.
- La propriété facultative *{ignoreWhitespace}* est une valeur booléenne indiquant si les différences entre les espaces blancs doivent être ignorées. La propriété *ignoreWhitespace* prend la valeur `false` si la propriété *useRegularExpressions* a la valeur `true` et prend la valeur `true` si l'expression *useRegularExpressions* a la valeur `false`.

- La propriété *{useRegularExpressions}* est une valeur booléenne indiquant que la propriété `searchString` utilise des expressions régulières. Si cette propriété n'est pas définie, elle prend par défaut la valeur `false`.

Valeurs renvoyées

Aucune.

Exemple

L'exemple de code suivant montre comment créer un objet à rechercher (*searchObject*) de trois façons différentes :

```
var searchParams;
searchParams.searchString = 'bgcolor="#FFCCFF"';
searchParams.replaceString = 'bgcolor="#CCFFCC"';
searchParams.searchSource = true;
dreamweaver.setUpFindReplace(searchParams);

var searchParams = {searchString: 'bgcolor="#FFCCFF"', replaceString:
  'bgcolor="#CCFFCC"', searchSource: true};
dreamweaver.setUpFindReplace(searchParams);

dreamweaver.setUpFindReplace({searchString: 'bgcolor="#FFCCFF"', ¬
replaceString: 'bgcolor="#CCFFCC"', searchSource: true});
```

dreamweaver.showFindDialog()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Rechercher.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canShowFindDialog()*, page 591.

dreamweaver.showFindReplaceDialog()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Remplacer.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canShowFindDialog()*, page 591.

Fonctions de modifications générales

Ces fonctions s'utilisent depuis la fenêtre de document. Elles permettent d'insérer du texte, du code HTML et des objets, d'appliquer, de modifier et de supprimer des marqueurs de police et de caractère, de modifier des balises et des attributs, etc.

dom.applyCharacterMarkup()

Disponibilité

Dreamweaver 3.

Description

Applique à la sélection le type de marqueur de caractère spécifié. Si la sélection est un point d'insertion, la fonction applique les marqueurs de caractère spécifiés au texte saisi après le point d'insertion.

Arguments

`tagName`

- L'argument *`tagName`* est le nom de la balise associé au marqueur de caractère. Il doit s'agir de l'une des chaînes suivantes : `"b"`, `"cite"`, `"code"`, `"dfn"`, `"em"`, `"i"`, `"kbd"`, `"samp"`, `"s"`, `"strong"`, `"tt"`, `"u"` ou `"var"`.

Valeurs renvoyées

Aucune.

dom.applyFontMarkup()

Disponibilité

Dreamweaver 3.

Description

Applique à la sélection en cours la balise `FONT`, ainsi que l'attribut spécifié et sa valeur.

Arguments

`attribute, value`

- L'argument *`attribute`* doit être `"face"`, `"size"` ou `"color"`.
- L'argument *`value`* est la valeur à affecter à l'attribut, comme `"Arial, Helvetica, sans-serif"`, `"5"` ou `"#FF0000"`.

Valeurs renvoyées

Aucune.

dom.deleteSelection()

Disponibilité

Dreamweaver 3.

Description

Supprime la sélection du document.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.editAttribute()

Disponibilité

Dreamweaver 3.

Description

Affiche l'interface permettant de modifier l'attribut de document spécifié. Dans la plupart des cas, il s'agit d'une boîte de dialogue. Cette fonction n'est valide que pour le document actif.

Arguments

attribute

- *attribute* est une chaîne qui spécifie l'attribut de balise à modifier.

Valeurs renvoyées

Aucune.

dom.exitBlock()

Disponibilité

Dreamweaver 3.

Description

Quitte le bloc de paragraphe ou d'en-tête en cours et place le point d'insertion à l'extérieur de tous les éléments de bloc.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.getCharSet()

Disponibilité

Dreamweaver 4.

Description

Renvoie l'attribut `charset` dans la balise Meta du document.

Arguments

Aucun.

Valeurs renvoyées

L'identité de codage du document. Par exemple, dans un document Latin1, cette fonction renvoie `iso-8859-1`.

dom.getFontMarkup()

Disponibilité

Dreamweaver 3.

Description

Obtient la valeur de l'attribut spécifié de la balise `FONT` pour la sélection en cours.

Arguments

`attribute`

- L'argument *`attribute`* doit être `"face"`, `"size"` ou `"color"`.

Valeurs renvoyées

Soit une chaîne contenant la valeur de l'attribut spécifié, soit une chaîne vide si l'attribut n'est pas défini.

dom.getLineFromOffset()

Disponibilité

Dreamweaver MX.

Description

Trouve le numéro de ligne d'un décalage de caractère précis dans le texte (le code HTML ou JavaScript) du fichier.

Arguments

`offset`

- L'argument *`offset`* est un nombre entier qui représente l'emplacement du caractère à partir du début du fichier.

Valeurs renvoyées

Nombre entier qui représente le numéro de la ligne dans le document.

dom.getLinkHref()

Disponibilité

Dreamweaver 3.

Description

Obtient le lien qui entoure la sélection en cours. Revient à effectuer une boucle sur les parents et les grands-parents du nœud en cours jusqu'à ce qu'un lien soit rencontré, puis à appeler la fonction `getAttribute('HREF')` sur ce lien.

Arguments

Aucun.

Valeurs renvoyées

Chaîne qui contient le nom du fichier lié, exprimé sous la forme d'une URL de type file://.

dom.getLinkTarget()

Disponibilité

Dreamweaver 3.

Description

Obtient la cible du lien qui entoure la sélection en cours. Revient à effectuer une boucle sur les parents et les grands-parents du nœud en cours jusqu'à ce qu'un lien soit rencontré, puis à appeler la fonction `getAttribute('TARGET')` sur ce lien.

Arguments

Aucun.

Valeurs renvoyées

Soit une chaîne contenant la valeur de l'attribut `TARGET` spécifié pour le lien, soit une chaîne vide si aucune cible n'est définie.

dom.getListTag()

Disponibilité

Dreamweaver 3.

Description

Obtient le style de la liste sélectionnée.

Arguments

Aucun.

Valeurs renvoyées

Soit une chaîne contenant la balise associée à la liste (`"ul"`, `"ol"` ou `"dl"`), soit une chaîne vide si aucune balise n'est associée à la liste. Cette valeur est toujours renvoyée en minuscules.

dom.getTextAlignment()

Disponibilité

Dreamweaver 3.

Description

Obtient l'alignement du bloc contenant la sélection.

Arguments

Aucun.

Valeurs renvoyées

Soit une chaîne contenant la valeur de l'attribut `ALIGN` de la balise associée au bloc, soit une chaîne vide si l'attribut `ALIGN` n'est pas défini. Cette valeur est toujours renvoyée en minuscules.

dom.getTextFormat()

Disponibilité

Dreamweaver 3.

Description

Obtient le format du bloc contenant la sélection.

Arguments

Aucun.

Valeurs renvoyées

Soit une chaîne contenant la balise de bloc associée au texte (comme `"p"`, `"h1"`, `"pre"`, etc.), soit une chaîne vide si aucune balise de bloc n'est associée à la sélection. Cette valeur est toujours renvoyée en minuscules.

dom.hasCharacterMarkup()

Disponibilité

Dreamweaver 3.

Description

Vérifie si le marqueur de caractère spécifié est déjà associé à la sélection.

Arguments

`markupTagName`

- L'argument *`markupTagName`* est le nom de la balise à vérifier. Il doit s'agir de l'une des chaînes suivantes : `"b"`, `"cite"`, `"code"`, `"dfn"`, `"em"`, `"i"`, `"kbd"`, `"samp"`, `"s"`, `"strong"`, `"tt"`, `"u"` ou `"var"`.

Valeurs renvoyées

Valeur booléenne qui indique si le marqueur de caractère spécifié est associé à la sélection entière. Cette fonction renvoie la valeur `false` si le marqueur spécifié n'est associé qu'à une partie de la sélection.

dom.indent()

Disponibilité

Dreamweaver 3.

Description

Applique un retrait à la sélection à l'aide des balises `BLOCKQUOTE`. Si la sélection est un élément de liste, cette fonction lui applique un retrait en convertissant l'élément sélectionné en liste imbriquée. La liste imbriquée est du même type que la liste externe et contient un seul élément (la sélection d'origine).

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.insertHTML()

Disponibilité

Dreamweaver 3.

Description

Insère un contenu HTML dans le document, au niveau du point d'insertion en cours.

Arguments

contentToInsert, {bReplaceCurrentSelection}

- L'argument *contentToInsert* est le contenu à insérer.
- L'argument facultatif *bReplaceCurrentSelection* est une valeur booléenne qui indique si le contenu spécifié doit remplacer la sélection en cours. Si la valeur de l'argument *bReplaceCurrentSelection* est `true`, le contenu remplace la sélection actuelle. Si sa valeur est `false`, le contenu est inséré après la sélection actuelle.

Valeurs renvoyées

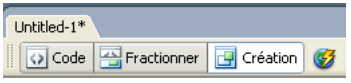
Aucune.

Exemple

Le code suivant insère la chaîne HTML `<b>130</b>` dans le document actuel :

```
var theDOM = dw.getDocumentDOM();
theDOM.insertHTML('<b>130</b>');
```

Le résultat apparaît dans la fenêtre de document, comme indiqué dans la figure suivante :



130

dom.insertObject()

Disponibilité

Dreamweaver 3.

Description

Insère l'objet spécifié et invite l'utilisateur à définir des paramètres, le cas échéant.

Arguments

objectName

- L'argument *objectName* est le nom d'un objet dans le dossier Configuration/Objects.

Valeurs renvoyées

Aucune.

Exemple

Un appel à la fonction `dom.insertObject('Button')` insère un bouton de formulaire dans le document actif, après la sélection en cours. Si aucun élément n'est sélectionné, cette fonction insère le bouton au niveau du point d'insertion en cours.

REMARQUE	Même si les fichiers d'objets peuvent être conservés dans des dossiers séparés, le nom de chaque fichier doit impérativement être unique. En effet, si le dossier Forms et le dossier MyObjects contiennent chacun un fichier nommé Button.htm, par exemple, Dreamweaver n'est pas capable de faire la différence entre les deux.

dom.insertText()

Disponibilité

Dreamweaver 3.

Description

Insère un contenu dans le document, au niveau du point d'insertion en cours.

Arguments

contentToInsert, {bReplaceCurrentSelection}

- L'argument *contentToInsert* est le contenu à insérer.

- L'argument facultatif *bReplaceCurrentSelection* est une valeur booléenne qui indique si le contenu spécifié doit remplacer la sélection en cours. Si la valeur de l'argument *bReplaceCurrentSelection* est `true`, le contenu remplace la sélection actuelle. Si sa valeur est `false`, le contenu est inséré après la sélection actuelle.

Valeurs renvoyées

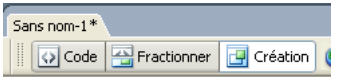
Aucune.

Exemple

Le code suivant insère le texte : `<b>130</b>` dans le document en cours :

```
var theDOM = dreamweaver.getDocumentDOM();
theDOM.insertText('<b>130</b>');
```

Les résultats apparaissent dans la fenêtre de document, comme indiqué dans la figure suivante :



130

dom.newBlock()

Disponibilité

Dreamweaver 3.

Description

Crée un bloc doté de la même balise et des mêmes attributs que le bloc contenant la sélection en cours ou, si le pointeur se trouve à l'extérieur de tous les blocs, crée un paragraphe.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Exemple

Si la sélection en cours se trouve à l'intérieur d'un paragraphe centré, un appel à la fonction `dom.newBlock()` insère `<p align="center">` après le paragraphe en cours.

dom.notifyFlashObjectChanged()

Disponibilité

Dreamweaver 4.

Description

Indique à Dreamweaver que le fichier Flash en cours a été modifié. Dreamweaver met à jour la fenêtre d'aperçu en la redimensionnant, le cas échéant, et en veillant à conserver le rapport hauteur/largeur d'origine. Par exemple, le texte Flash utilise cette fonction pour mettre à jour le texte en mode de Mise en forme à mesure que l'utilisateur en change les propriétés dans la boîte de dialogue Commande.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.outdent()

Disponibilité

Dreamweaver 3.

Description

Applique un retrait négatif à la sélection.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.removeCharacterMarkup()

Disponibilité

Dreamweaver 3.

Description

Supprime de la sélection le type de marqueur de caractère spécifié.

Arguments

tagName

- L'argument *tagName* est le nom de la balise associé au marqueur de caractère. Il doit s'agir de l'une des chaînes suivantes : "b", "cite", "code", "dfn", "em", "i", "kbd", "samp", "s", "strong", "tt", "u" ou "var".

Valeurs renvoyées

Aucune.

dom.removeFontMarkup()

Disponibilité

Dreamweaver 3.

Description

Supprime d'une balise FONT l'attribut spécifié, ainsi que sa valeur. Si, après suppression de l'attribut, il ne reste que la balise FONT, la balise FONT est également supprimée.

Arguments

attribute

- L'argument *attribute* doit être "face", "size" ou "color".

Valeurs renvoyées

Aucune.

dom.removeLink()

Disponibilité

Dreamweaver 3.

Description

Supprime le lien hypertexte de la sélection.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.resizeSelection()

Disponibilité

Dreamweaver 3.

Description

Redimensionne l'objet sélectionné et lui applique les nouvelles dimensions spécifiées.

Arguments

newWidth, newHeight

- L'argument *newWidth* spécifie la nouvelle largeur définie pour l'objet sélectionné par la fonction.
- L'argument *newHeight* spécifie la nouvelle hauteur définie pour l'objet sélectionné par la fonction.

Valeurs renvoyées

Aucune.

dom.setAttributeWithErrorChecking()

Disponibilité

Dreamweaver 3.

Description

Définit l'attribut spécifié sur la valeur indiquée pour la sélection en cours et affiche une invite utilisateur si le type de valeur est incorrect ou si la valeur n'est pas comprise dans la plage spécifiée. Cette fonction n'est valide que pour le document actif.

Arguments

attribute, value

- L'argument *attribute* spécifie l'attribut à définir pour la sélection actuelle.
- L'argument *value* spécifie la valeur à définir pour l'attribut.

Valeurs renvoyées

Aucune.

dom.setLinkHref()

Disponibilité

Dreamweaver 3.

Description

Transforme la sélection en lien hypertexte ou modifie la valeur de l'URL pour la balise HREF qui entoure la sélection en cours.

Arguments

`linkHREF`

- L'argument *`linkHREF`* est une URL (chemin relatif au document ou à la racine, ou URL absolue) contenant le lien. Si aucun argument n'est défini, la boîte de dialogue Sélectionner fichier HTML s'affiche.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canSetLinkHref()*, page 579.

dom.setLinkTarget()

Disponibilité

Dreamweaver 3.

Description

Définit la cible du lien qui entoure la sélection en cours. Revient à effectuer une boucle sur les parents et les grands-parents du nœud en cours jusqu'à ce qu'un lien soit rencontré, puis à appeler la fonction `setAttribute('TARGET')` sur ce lien.

Arguments

`[linkTarget]`

- L'argument facultatif *`linkTarget`* est une chaîne qui représente le nom d'un cadre ou d'une fenêtre, ou l'une des cibles réservées (`"_self"`, `"_parent"`, `"_top"` ou `"_blank"`). Si aucun argument n'est défini, la boîte de dialogue Définir la cible s'affiche.

Valeurs renvoyées

Aucune.

dom.setListBoxKind()

Disponibilité

Dreamweaver 3.

Description

Modifie le type du menu `SELECT` sélectionné.

Arguments

`kind`

- L'argument *`kind`* doit être `"menu"` ou `"list box"`.

Valeurs renvoyées

Aucune.

dom.showListPropertiesDialog()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Propriétés de la liste.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dom.canShowListPropertiesDialog()*, page 579.

dom.setListTag()

Disponibilité

Dreamweaver 3.

Description

Définit le style de la liste sélectionnée.

Arguments

listTag

- L'argument *listTag* est la balise associée à la liste. Il doit s'agir de "ol", "ul", "dl" ou d'une chaîne vide.

Valeurs renvoyées

Aucune.

dom.setTextAlignment()

Disponibilité

Dreamweaver 3.

Description

Affecte la valeur spécifiée à l'attribut ALIGN du bloc contenant la sélection.

Arguments

alignValue

- L'argument *alignValue* doit être "left", "center" ou "right".

Valeurs renvoyées

Aucune.

dom.setTextFieldKind()

Disponibilité

Dreamweaver 3.

Description

Définit le format du champ texte sélectionné.

Arguments

fieldType

- L'argument *fieldType* doit être "input", "textarea" ou "password".

Valeurs renvoyées

Aucune.

dom.setTextFormat()

Disponibilité

Dreamweaver 4.

Description

Définit le format de bloc du texte sélectionné.

Arguments

blockFormat

- L'argument *blockFormat* est une chaîne qui spécifie un des formats suivants : "" (aucun format), "p", "h1", "h2", "h3", "h4", "h5", "h6" ou "pre".

Valeurs renvoyées

Aucune.

dom.showFontColorDialog()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue du sélecteur de couleur.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.deleteSelection()

Disponibilité

Dreamweaver 3.

Description

Supprime la sélection du document actif, du panneau Site ou, sur Macintosh, il supprime la zone de texte active dans une boîte de dialogue ou un panneau flottant.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canDeleteSelection()*, page 584.

dreamweaver.editFontList()

Disponibilité

Dreamweaver 3.

Description

Ouvre la boîte de dialogue Modifier la liste des polices.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.getFontList()

Disponibilité

Dreamweaver 3.

Description

Obtient la liste de tous les groupes de polices apparaissant dans l'inspecteur de propriétés de texte et la boîte de dialogue Définition du style.

Arguments

Aucun.

Valeurs renvoyées

Tableau de chaînes dont chaque chaîne représente un élément de la liste des polices.

Exemple

Si vous avez effectué une installation par défaut de Dreamweaver, un appel à la fonction `dreamweaver.getFontList()` renvoie un tableau contenant les éléments suivants :

- `"Arial, Helvetica, sans-serif"`
- `"Times New Roman, Times, serif"`
- `"Courier New, Courier, mono"`
- `"Georgia, Times New Roman, Times, serif"`
- `"Verdana, Arial, Helvetica, sans-serif"`

dreamweaver.getFontStyles()

Disponibilité

Dreamweaver 4.

Description

Renvoie les styles pris en charge par la police TrueType spécifiée.

Arguments

`fontName`

- L'argument *`fontName`* est une chaîne contenant le nom de la police.

Valeurs renvoyées

Tableau de trois valeurs booléennes qui indique les styles pris en charge par la police. La première valeur indique si la police prend en charge le style *Gras*, la deuxième le style *Italique* et la troisième les styles *Gras* et *Italique*.

dreamweaver.getKeyState()

Disponibilité

Dreamweaver 3.

Description

Détermine si la touche de modification spécifiée est enfoncée.

Arguments

key

- L'argument *key* doit correspondre à l'une des valeurs suivantes : `"Cmd"`, `"Ctrl"`, `"Alt"` ou `"Shift"`. Sous Windows, `"Cmd"` et `"Ctrl"` désignent la touche Contrôle ; sur Macintosh, `"Alt"` désigne la touche Option.

Valeurs renvoyées

Valeur booléenne qui indique si la touche est enfoncée.

Exemple

Le code suivant vérifie si les touches Maj et Ctrl (Windows) ou Maj et Commande (Macintosh) sont enfoncées avant d'effectuer une opération.

```
if (dw.getKeyState("Shift") && dw.getKeyState("Cmd")){
  // execute code
}
```

dreamweaver.getNaturalSize()

Disponibilité

Dreamweaver 4.

Description

Renvoie la largeur et la hauteur d'un objet graphique.

Arguments

url

- L'argument *url* pointe vers un objet graphique dont les dimensions sont requises. Dreamweaver doit prendre en charge cet objet (GIF, JPEG, PNG, Flash ou Shockwave). L'URL fournie comme argument de la fonction `getNaturalSize()` doit correspondre à une URL absolue pointant vers un fichier local ; il ne peut pas s'agir d'une URL relative.

Valeurs renvoyées

Tableau contenant deux nombres entiers, le premier définissant la largeur de l'objet et le second sa hauteur.

dreamweaver.getSystemFontList()

Disponibilité

Dreamweaver 4.

Description

Renvoie une liste de polices à utiliser pour le système. Cette fonction peut appeler toutes les polices ou uniquement les polices TrueType. Ces polices sont requises pour l'objet Texte Flash.

Arguments

fontTypes

- L'argument *fontTypes* est une chaîne qui contient soit `"all"`, soit `"TrueType"`.

Valeurs renvoyées

Tableau de chaînes qui contient le nom de toutes les polices ; renvoie la valeur `null` si aucune police n'est trouvée.

Fonction relative à l'impression

Cette fonction permet à l'utilisateur d'imprimer le code depuis le mode Code.

dreamweaver.printCode()

Disponibilité

Dreamweaver MX.

Description

Sous Windows, cette fonction imprime toutes les portions de code ou seulement une sélection de code depuis le mode Code. Sur Macintosh, elle imprime toutes les portions de code ou seulement une plage de pages de code.

Arguments

showPrintDialog, document

- L'argument *showPrintDialog* est `true` ou `false`. Sous Windows, si cet argument a pour valeur `true`, la fonction `dreamweaver.PrintCode()` affiche la boîte de dialogue d'impression pour demander à l'utilisateur s'il souhaite imprimer tout le texte ou seulement une sélection de texte. Sur Macintosh, la fonction `dreamweaver.PrintCode()` affiche la boîte de dialogue d'impression pour demander à l'utilisateur s'il souhaite imprimer tout le texte ou seulement une plage de pages.

  Si l'argument a pour valeur `false`, `dreamweaver.PrintCode()` utilise la sélection précédente de l'utilisateur. La valeur par défaut est `true`.
- L'argument *document* est le DOM du document à imprimer. Pour plus d'informations sur la façon d'obtenir le DOM d'un document, voir *dreamweaver.getDocumentDOM()*, page 319.

Valeurs renvoyées

Valeur booléenne : `true` si le code peut s'imprimer ; `false` dans les autres cas.

Exemple

L'exemple suivant appelle `dw.PrintCode()` pour invoquer la boîte de dialogue d'impression pour le document utilisateur. Si la fonction renvoie la valeur `false`, le code affiche un message d'alerte informant l'utilisateur qu'il n'est pas possible d'exécuter la demande d'impression.

```
var theDOM = dreamweaver.getDocumentDOM("document");
if(!dreamweaver.PrintCode(true, theDOM))
{
  alert("Unable to execute your print request!");
}
```

Fonctions relatives à Quick Tag Editor

Ces fonctions permettent de se déplacer d'une balise à l'autre à l'intérieur et autour de la sélection en cours. Elles permettent de supprimer n'importe laquelle de ces balises, d'envelopper la sélection à l'intérieur d'une nouvelle balise et d'afficher Quick Tag Editor pour permettre à l'utilisateur de modifier certains attributs d'une balise.

dom.selectChild()

Disponibilité

Dreamweaver 3.

Description

Sélectionne un enfant de la sélection en cours. Revient à sélectionner la balise située immédiatement à droite dans le sélecteur de balises, en bas de la fenêtre de document.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.selectParent()

Disponibilité

Dreamweaver 3.

Description

Sélectionne le parent de la sélection en cours. Revient à sélectionner la balise située immédiatement à gauche dans le sélecteur de balises, en bas de la fenêtre de document.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.stripTag()

Disponibilité

Dreamweaver 3.

Description

Supprime les balises qui entourent la sélection actuelle, laissant intact leur contenu. Si la sélection contient plusieurs balises ou n'en contient aucune, Dreamweaver affiche un message d'erreur.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.wrapTag()

Disponibilité

Dreamweaver 3.

Description

Place la balise spécifiée autour de la sélection en cours. Si la balise n'est pas complète, Dreamweaver affiche un message d'erreur.

Arguments

startTag

- L'argument *startTag* est la source associée à la balise d'ouverture.

Valeurs renvoyées

Aucune.

Exemple

Le code suivant place un lien autour de la sélection en cours.

```
var theDOM = dw.getDocumentDOM();
var theSel = theDOM.getSelectedNode();
if (theSel.nodeType == Node.TEXT_NODE){
  theDOM.wrapTag('<a href="foo.html">');
}
```

dreamweaver.showQuickTagEditor()

Disponibilité

Dreamweaver 3.

Description

Affiche Quick Tag Editor pour la sélection en cours.

Arguments

{nearWhat}, {mode}

- L'argument facultatif *nearWhat* doit, s'il est spécifié, être `"selection"` ou `"tag selector"`. Si cet argument n'est pas défini, il prend par défaut la valeur `"selection"`.

- L'argument facultatif *mode* doit, s'il est spécifié, être "default", "wrap", "insert" ou "edit". Si l'argument *mode* a pour valeur "default" ou qu'il n'est pas défini, Dreamweaver utilise la méthode heuristique afin de déterminer le mode à utiliser pour la sélection en cours. Si l'argument *nearWhat* a pour valeur *"tag selector"*, l'argument mode est ignoré.

Valeurs renvoyées

Aucune.

Fonctions relatives au mode Code

Ces fonctions incluent les opérations associées à la modification du code source d'un document (et tous les changements ayant une incidence sur le mode Création). Les fonctions de cette section vous permettent d'ajouter des commandes de navigation aux modes Code au sein d'une fenêtre de document affichée dans deux volets ou dans la fenêtre de l'inspecteur de code.

dom.formatRange()

Disponibilité

Dreamweaver MX.

Description

Applique le formatage de syntaxe automatique de Dreamweaver à une plage définie de caractères en mode Code, et ce conformément aux paramètres de la boîte de dialogue Préférences > Format de code.

Arguments

startOffset, endOffset

- L'argument *startOffset* est un nombre entier qui représente le début de la plage définie, et ce sous la forme d'un décalage par rapport au début du document.
- L'argument *endOffset* est un nombre entier qui représente la fin de la plage définie, et ce sous la forme d'un décalage par rapport au début du document.

Valeurs renvoyées

Aucune.

dom.formatSelection()

Disponibilité

Dreamweaver MX.

Description

Applique le formatage de syntaxe automatique de Dreamweaver au contenu sélectionné (ce qui revient à utiliser l'option Commandes > Appliquer le format source à la sélection), conformément aux paramètres de la boîte de dialogue Préférences > Format de code.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.getShowNoscript()

Disponibilité

Dreamweaver MX.

Description

Obtient l'état actuel de l'option de contenu `noscript` (depuis l'option de menu Affichage > Contenu Noscript). Activée par défaut, la balise `noscript` identifie un script de page pouvant, au choix, être affiché ou non dans le navigateur.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu de la balise `noscript` est actuellement affiché ; `false` dans le cas contraire.

dom.getAutoValidationCount()

Disponibilité

Dreamweaver MX 2004.

Description

Obtient le nombre d'erreurs, d'avertissements et de messages d'information pour la dernière auto-validation (ou validation en ligne) du document. Actuellement seule une vérification du navigateur cible est effectuée durant l'auto-validation (voir *dom.runValidation()*, page 332).

REMARQUE

Cette fonction renvoie uniquement les résultats actuellement affichés dans la fenêtre de résultats du document. Pour vous assurer que les comptes sont à jour, vous pouvez appeler dom.runValidation() avant d'appeler cette fonction.

Arguments

Aucun.

Valeurs renvoyées

Un objet avec les propriétés suivantes :

- la propriété `numError`, qui est le nombre d'erreurs ;
- la propriété `numWarning`, qui est le nombre d'avertissements ;
- la propriété `numInfo`, qui est le nombre de messages d'information.

Exemple

```
theDom = dw.getDocumentDOM();
theDom.runValidation();
theDom.getAutoValidationCount();
```

dom.isDesignViewUpdated()

Disponibilité

Dreamweaver 4.

Description

Détermine si le contenu des modes Création et Texte est synchronisé pour les opérations Dreamweaver qui requièrent un état de document correct.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le mode Création (WYSIWIG) est synchronisé avec le texte du mode Texte et `false` dans le cas contraire.

dom.isSelectionValid()

Disponibilité

Dreamweaver 4.

Description

Détermine si une sélection est valide, ce qui signifie qu'elle est synchronisée avec le mode Création, ou s'il faut la déplacer avant qu'une opération n'ait lieu.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection en cours contient un segment de code correct ; `false` si le document n'a pas encore été synchronisé, étant donné que la sélection n'a pas été mise à jour.

dom.setShowNoscript()

Disponibilité

Dreamweaver MX.

Description

Active ou désactive l'option de contenu `noscript` (ce qui revient à utiliser l'option Affichage > Contenu Noscript). Activée par défaut, la balise `noscript` identifie un script de page pouvant, au choix, être affiché ou non dans le navigateur.

Arguments

`{bShowNoscript}`

- L'argument facultatif *`bShowNoscript`* est une valeur booléenne qui indique si le contenu de la balise `noscript` doit être affiché ; `true` si le contenu de la balise `noscript` doit être affiché ; `false` dans le cas contraire.

Valeurs renvoyées

Aucune.

dom.source.arrowDown()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion vers le bas du document affiché en mode Code, ligne par ligne. Si le contenu est déjà sélectionné, cette fonction étend la sélection ligne par ligne.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* est le nombre de lignes que le point d'insertion doit sauter. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.arrowLeft()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion vers la gauche de la ligne courante dans le mode Code. Si le contenu est déjà sélectionné, cette fonction étend la sélection vers la gauche.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* est le nombre de caractères que le point d'insertion doit sauter. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.arrowRight()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion vers la droite de la ligne courante dans le mode Code. Si le contenu est déjà sélectionné, cette fonction étend la sélection vers la droite.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* est le nombre de caractères que le point d'insertion doit sauter. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné ; dans le cas contraire, il ne l'est pas.

Valeurs renvoyées

Aucune.

dom.source.arrowUp()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion vers le haut du document affiché en mode Code, ligne par ligne. Si le contenu est déjà sélectionné, cette fonction étend la sélection ligne par ligne.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument *nTimes* est le nombre de lignes que le point d'insertion doit déplacer. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.
- L'argument *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.balanceBracesTextview()

Disponibilité

Dreamweaver 4.

Description

Cette fonction est une extension du mode Code qui permet d'équilibrer les parenthèses. Vous pouvez appeler la fonction `dom.source.balanceBracesTextView()` pour étendre la sélection mise en surbrillance ou le point d'insertion depuis le début de l'instruction entre parenthèses jusqu'à la fin de l'instruction afin d'équilibrer les caractères suivants : `[]`, `{}` et `()`. Tout nouvel appel étend la sélection à des niveaux supplémentaires de ponctuation imbriquée.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.source.endOfDocument()

Disponibilité

Dreamweaver 4.

Description

Place le point d'insertion à la fin du document actif affiché en mode Code. Si le contenu est déjà sélectionné, cette fonction étend la sélection à la fin du document.

Arguments

`bShiftIsDown`

- L'argument *`bShiftIsDown`* est une valeur booléenne qui indique si un contenu est sélectionné. Si *`bShiftIsDown`* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.endOfLine()

Disponibilité

Dreamweaver 4.

Description

Place le point d'insertion à la fin de la ligne courante. Si le contenu est déjà sélectionné, cette fonction étend la sélection jusqu'à la fin de la ligne courante.

Arguments

bShiftIsDown

- L'argument *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.endPage()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion à la fin de la page en cours ou à la fin de la page suivante (si le point d'insertion est déjà à la fin d'une page). Si le contenu est déjà sélectionné, cette fonction étend la sélection page par page.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* est le nombre de pages que le point d'insertion doit sauter. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si `bShiftIsDown` a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.getCurrentLines()

Disponibilité

Dreamweaver 4.

Description

Renvoie les numéros de ligne des décalages spécifiés à partir du début du document.

Arguments

Aucun.

Valeurs renvoyées

Numéros de ligne de la sélection en cours.

dom.source.getSelection()

Description

Obtient la sélection du document actif, exprimée en décalages de caractères dans le mode Code du document.

Arguments

Aucun.

Valeurs renvoyées

Paire de nombres entiers représentant les décalages à partir du début du document source. Le premier nombre entier correspond au début de la sélection et le second à la fin. Si les deux nombres sont égaux, la sélection est un point d'insertion. Si aucun élément n'est sélectionné dans la source, les deux nombres sont -1.

dom.source.getLineFromOffset()

Disponibilité

Dreamweaver MX.

Description

Place un décalage dans le document source.

Arguments

Aucun.

Valeurs renvoyées

Numéro de la ligne, ou -1 si le décalage est négatif ou se trouve après la fin du fichier.

dom.source.getText()

Disponibilité

Dreamweaver 4.

Description

Renvoie la chaîne de texte de la source comprise entre les décalages définis.

Arguments

startOffset, endOffset

- L'argument *startOffset* est un nombre entier qui représente le décalage à partir du début du document.
- L'argument *endOffset* est un nombre entier qui représente la fin du document.

Valeurs renvoyées

Chaîne qui représente le texte du code source compris entre les décalages *start* et *end*.

dom.source.getValidationErrorsForOffset()

Disponibilité

Dreamweaver MX 2004.

Description

Renvoie la liste des erreurs de validation au point de décalage spécifié ou recherche l'erreur suivante après le décalage. Si aucune n'est trouvée dans la fonction, la valeur `null` est renvoyée.

Arguments

offset, {searchDirection}

- L'argument *offset* est un nombre qui spécifie le décalage dans le code pour lequel la fonction renvoie des erreurs.
- L'argument facultatif *searchDirection* est une chaîne qui spécifie `"empty"`, `"forward"` ou `"back"`. Si spécifié, la fonction recherche vers la fin ou le début du document, en partant du décalage défini, les caractères comportant des erreurs et les renvoie. Si cela n'est pas spécifié, la fonction vérifie la présence d'erreur au point de décalage défini.

Valeurs renvoyées

Tableau d'objets ou valeur `null`. Chaque objet du tableau comporte les propriétés suivantes :

- L'objet `message` est une chaîne qui contient le message d'erreur.
- L'objet `floaterName` est une chaîne qui contient le nom de la fenêtre de résultats. Vous pouvez transmettre cette valeur aux fonctions `showResults()` ou `setFloaterVisibility()`.
- L'objet `floaterIndex` est un index d'éléments dans la liste de résultats de palette.

- L'objet `start` est l'index d'ouverture du code souligné.
- L'objet `end` est l'index de fermeture du code souligné.

REMARQUE	Les index de palette renvoyés ne doivent pas être conservés car ils varient fréquemment, par exemple, lors de l'ouverture ou de la fermeture de documents.

Exemple

L'exemple suivant appelle `getValidationErrorsForOffset()` pour vérifier qu'il n'y a pas d'erreur au décalage de la sélection actuelle. Si la fonction renvoie une erreur, le code appelle la fonction `alert()` pour afficher le message d'erreur à l'utilisateur.

```
var offset = dw.getDocumentDOM().source.getSelection()[0];
var errors =
  dw.getDocumentDOM().source.getValidationErrorsForOffset(offset);
if ( errors && errors.length > 0 )
  alert( errors[0].message );
```

dom.source.indentTextview()

Disponibilité

Dreamweaver 4.

Description

Déplace le texte sélectionné en mode Code d'une marque de tabulation vers la droite.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.source.insert()

Disponibilité

Dreamweaver 4.

Description

Insère la chaîne spécifiée dans le code source au niveau du point de décalage défini à partir du début du fichier source. Si le décalage n'est pas supérieur ou égal à zéro, l'insertion échoue et la fonction renvoie la valeur `false`.

Arguments

`offset, string`

- L'argument *`offset`* est le décalage à partir du début du fichier où la chaîne doit être insérée.
- L'argument *`string`* est la chaîne à insérer.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

dom.source.nextWord()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion au début du mot suivant (ou des mots suivants, si spécifié) en mode Code. Si le contenu est déjà sélectionné, cette fonction étend la sélection vers la droite.

Arguments

`{nTimes}, {bShiftIsDown}`

- L'argument facultatif *`nTimes`* est le nombre de mots que le point d'insertion doit sauter. Si *`nTimes`* n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *`bShiftIsDown`* est une valeur booléenne qui indique si un contenu est sélectionné. Si *`bShiftIsDown`* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.outdentTextview()

Disponibilité

Dreamweaver 4.

Description

Déplace le texte sélectionné en mode Code d'une marque de tabulation vers la gauche.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.source.pageDown()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion vers le bas du document affiché en mode Code, page par page. Si le contenu est déjà sélectionné, cette fonction étend la sélection page par page.

Arguments

`{nTimes}, {bShiftIsDown}`

- L'argument facultatif *`nTimes`* est le nombre de pages que le point d'insertion doit sauter. Si *`nTimes`* n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *`bShiftIsDown`* est une valeur booléenne qui indique si un contenu est sélectionné. Si *`bShiftIsDown`* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.pageUp()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion vers le haut du document affiché en mode Code, page par page. Si le contenu est déjà sélectionné, cette fonction étend la sélection page par page.

Arguments

`{nTimes}, {bShiftIsDown}`

- L'argument facultatif *nTimes* est le nombre de pages que le point d'insertion doit sauter. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.previousWord()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion au début du mot précédent (ou des mots précédents, si spécifié) en mode Code. Si le contenu est déjà sélectionné, cette fonction étend la sélection vers la gauche.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* est le nombre de mots que le point d'insertion doit sauter. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.
- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.replaceRange()

Disponibilité

Dreamweaver 4.

Description

Remplace la plage de texte source comprise entre *startOffset* et *endOffset* par *string*. Si *startOffset* est supérieur à *endOffset* ou si l'un des décalages n'est pas un nombre entier positif, cette fonction n'a aucun effet et renvoie la valeur `false`. Si *endOffset* est supérieur au nombre de caractères du fichier, cette fonction remplace la plage comprise entre *startOffset* et la fin du fichier. Si *startOffset* et *endOffset* sont supérieurs au nombre de caractères du fichier, cette fonction insère le texte à la fin du fichier.

Arguments

startOffset, endOffset, string

- L'argument *startOffset* est le décalage indiquant le début du bloc à remplacer.
- L'argument *endOffset* est le décalage indiquant la fin du bloc à remplacer.
- L'argument *string* est la chaîne à insérer.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

dom.source.scrollEndFile()

Disponibilité

Dreamweaver 4.

Description

Fait défiler le mode Code vers le bas du document sans déplacer le point d'insertion.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.source.scrollLineDown()

Disponibilité

Dreamweaver 4.

Description

Fait défiler le mode Code vers le bas ligne par ligne sans déplacer le point d'insertion.

Arguments

nTimes

- L'argument *nTimes* est le nombre de lignes à faire défiler. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.

Valeurs renvoyées

Aucune.

dom.source.scrollLineUp()

Disponibilité

Dreamweaver 4.

Description

Fait défiler le mode Code vers le haut ligne par ligne sans déplacer le point d'insertion.

Arguments

nTimes

- L'argument *nTimes* est le nombre de lignes à faire défiler. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.

Valeurs renvoyées

Aucune.

dom.source.scrollPageDown()

Disponibilité

Dreamweaver 4.

Description

Fait défiler le mode Code vers le bas page par page sans déplacer le point d'insertion.

Arguments

nTimes

- L'argument *nTimes* est le nombre de pages à faire défiler. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.

Valeurs renvoyées

Aucune.

dom.source.scrollPageUp()

Disponibilité

Dreamweaver 4.

Description

Fait défiler le mode Code vers le haut page par page sans déplacer le point d'insertion.

Arguments

nTimes

- L'argument *nTimes* est le nombre de pages à faire défiler. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.

Valeurs renvoyées

Aucune.

dom.source.scrollTopFile()

Disponibilité

Dreamweaver 4.

Description

Fait défiler le mode Code vers le haut du document sans déplacer le point d'insertion.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.source.selectParentTag()

Disponibilité

Dreamweaver 4.

Description

Cette fonction est une extension du mode Code qui permet d'équilibrer les balises. Vous pouvez appeler `dom.source.selectParentTag()` pour étendre la sélection ou le point d'insertion courant de la balise d'ouverture à la balise de fermeture. Les appels suivants étendent la sélection à des balises supplémentaires jusqu'à ce qu'il n'y ait plus de balises de fermeture.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dom.source.setCurrentLine()

Disponibilité

Dreamweaver 4.

Description

Place le point d'insertion au début de la ligne indiquée. Si l'argument *`lineNumber`* n'est pas un nombre entier positif, la fonction n'a aucun effet et renvoie la valeur `false`. Le point d'insertion est placé au début de la dernière ligne si *`lineNumber`* est supérieur au nombre de lignes de la source.

Arguments

`lineNumber`

- L'argument *`lineNumber`* est la ligne au début de laquelle le point d'insertion est placé.

Valeurs renvoyées

Valeur booléenne : `true` en cas de succès et `false` dans le cas contraire.

dom.source.startOfDocument()

Disponibilité

Dreamweaver 4.

Description

Place le point d'insertion au début du document affiché en mode Code. Si le contenu est déjà sélectionné, cette fonction étend la sélection au début du document.

Arguments

bShiftIsDown

- L'argument *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.startOfLine()

Disponibilité

Dreamweaver 4.

Description

Place le point d'insertion au début de la ligne courante. Si le contenu est déjà sélectionné, cette fonction étend la sélection au début de la ligne courante.

Arguments

bShiftIsDown

- L'argument *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.topPage()

Disponibilité

Dreamweaver 4.

Description

Déplace le point d'insertion en haut de la page courante ou de la page précédente (si le point d'insertion est déjà en haut d'une page). Si le contenu est déjà sélectionné, cette fonction étend la sélection page par page.

Arguments

{nTimes}, {bShiftIsDown}

- L'argument facultatif *nTimes* est le nombre de pages que le point d'insertion doit sauter. Si *nTimes* n'est pas défini, il prend par défaut la valeur 1.

- L'argument facultatif *bShiftIsDown* est une valeur booléenne qui indique si un contenu est sélectionné. Si *bShiftIsDown* a la valeur `true`, le contenu est sélectionné.

Valeurs renvoyées

Aucune.

dom.source.wrapSelection()

Disponibilité

Dreamweaver 4.

Description

Insère le texte de *startTag* avant la sélection en cours et le texte de *endTag* après la sélection en cours. La fonction sélectionne ensuite la plage entière entre les balises insérées, en incluant ces balises. Si la sélection en cours est un point d'insertion, la fonction place le point d'insertion entre *startTag* et *endTag* (*startTag* et *endTag* ne doivent pas nécessairement être des balises ; il peut s'agir de tout segment de texte de votre choix).

Arguments

startTag, *endTag*

- L'argument *startTag* est le texte à insérer au début de la sélection.
- L'argument *endTag* est le texte à insérer à la fin de la sélection.

Valeurs renvoyées

Aucune.

dom.synchronizeDocument()

Disponibilité

Dreamweaver 4.

Description

Synchronise les modes Code et Création.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Fonctions de l'éditeur de balises et de la bibliothèque de balises

Vous pouvez utiliser les éditeurs de balises pour insérer de nouvelles balises, pour modifier des balises existantes ou pour accéder à des informations de référence sur les balises. Le sélecteur de balises permet aux utilisateurs d'organiser leurs balises afin qu'ils puissent sélectionner les balises les plus fréquemment utilisées. Les bibliothèques de balises fournies avec Dreamweaver stockent des informations concernant les balises utilisées dans les langages de balisage standard et dans les langages de script et de balise fréquemment utilisés. Vous pouvez utiliser les fonctions de l'éditeur de balises JavaScript, du sélecteur de balises, de la bibliothèque de balises lorsque vous devez utiliser les éditeurs de balises et bibliothèques de balises dans vos extensions.

dom.getTagSelectorTag()

Disponibilité

Dreamweaver MX.

Description

Cette fonction obtient le nœud DOM de la balise sélectionnée dans la barre du sélecteur de balises se trouvant au bas de la fenêtre de document.

Arguments

Aucun.

Valeurs renvoyées

Nœud DOM de la balise sélectionnée ; `null` si aucune balise n'est sélectionnée.

dreamweaver.popupInsertTagDialog()

Disponibilité

Dreamweaver MX.

Description

Cette fonction vérifie les fichiers VTM pour s'assurer qu'un éditeur de balises a été utilisé pour la balise. Si c'est le cas, l'éditeur de cette balise apparaît et accepte la balise de début. Si ce n'est pas le cas, la balise de début est insérée telle quelle dans le document de l'utilisateur.

Arguments

Chaîne de balise de début qui comprend un des types suivants de valeurs initiales :

- une balise, comme `<input >`;
- une balise avec des attributs, comme `<input type='text'>` ;
- une directive, comme `<%= %>` .

Valeurs renvoyées

Valeur booléenne : `true` si un élément est inséré dans le document et `false` dans le cas contraire.

dreamweaver.popupEditTagDialog()

Disponibilité

Dreamweaver MX.

Description

Si une balise est sélectionnée, cette fonction ouvre l'éditeur de balises correspondant à cette balise pour vous permettre de modifier la balise.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

Activateur

Voir *dreamweaver.canPopupEditTagDialog()*, page 587.

dreamweaver.showTagChooser()

Disponibilité

Dreamweaver MX.

Description

Cette fonction affiche la boîte de dialogue Sélecteur de balises, la fait apparaître au premier plan et la rend active.

Arguments

Aucun.

Valeurs renvoyées

Aucune.

dreamweaver.showTagLibraryEditor()

Disponibilité

Dreamweaver MX.

Description

Cette fonction ouvre l'éditeur de la bibliothèque de balises.

Arguments

Aucun.

Valeurs renvoyées

Aucun.

dreamweaver.tagLibrary.getTagLibraryDOM()

Disponibilité

Dreamweaver MX.

Description

Associée à l'URL d'un fichier *filename*.vtm, cette fonction renvoie le DOM de ce fichier afin que son contenu soit modifié. Cette fonction ne doit être appelée que lorsque l'éditeur de la bibliothèque de balises est actif.

Arguments

fileURL

- L'argument *fileURL* est l'URL d'un fichier *filename*.vtm, relatif au dossier Configuration/Tag Libraries, comme indiqué dans l'exemple suivant :

```
"HTML/img.vtm"
```

Valeurs renvoyées

Un pointeur DOM désignant un fichier nouveau ou plus ancien du dossier TagLibraries.

dreamweaver.tagLibrary.getSelectedLibrary()

Disponibilité

Dreamweaver MX.

Description

Si un nœud de bibliothèque est sélectionné dans l'éditeur de la bibliothèque de balises, cette fonction obtient le nom de la bibliothèque.

Arguments

Aucun.

Valeurs renvoyées

Chaîne correspondant au nom de la bibliothèque sélectionnée dans l'éditeur de la bibliothèque de balises ; renvoie une chaîne vide si aucune bibliothèque n'est sélectionnée.

dreamweaver.tagLibrary.getSelectedTag()

Disponibilité

Dreamweaver MX.

Description

Si un nœud d'attribut est sélectionné, cette fonction obtient le nom de la balise qui contient l'attribut.

Arguments

Aucun.

Valeurs renvoyées

Chaîne correspondant au nom de la balise sélectionnée dans l'éditeur de la bibliothèque de balises ; renvoie une chaîne vide si aucune balise n'est sélectionnée.

dreamweaver.tagLibrary.importDTDOrSchema()

Disponibilité

Dreamweaver MX.

Description

Cette fonction importe un fichier DTD/Schéma à partir d'un serveur distant dans la bibliothèque de balises.

Arguments

fileURL, Prefix

- L'argument *fileURL* est le chemin du fichier DTD ou schéma, au format URL local.
- L'argument *Prefix* est la chaîne de préfixe qui doit être ajoutée à toutes les balises de cette bibliothèque.

Valeurs renvoyées

Nom de la bibliothèque de balises importées.

dreamweaver.tagLibrary.getImportedTagList()

Disponibilité

Dreamweaver MX.

Description

Cette fonction génère une liste d'objets `TagInfo` à partir d'une bibliothèque de balises importées.

Arguments

libname

- L'argument *libname* est le nom de la bibliothèque de balises importées.

Valeurs renvoyées

Tableau d'objets `tagInfo`.

Un objet `taginfo` contient des informations concernant une balise de la bibliothèque de balises. Les propriétés suivantes sont définies dans un objet `tagInfo` :

- la propriété `tagName`, qui est une chaîne ;
- la propriété `attributes`, qui est un tableau de chaînes. Chaque chaîne correspond au nom d'un attribut défini pour cette balise.

Exemple :

L'exemple suivant indique que l'utilisation de la fonction `dw.tagLibrary.getImportedTagList()` peut résulter en un tableau de balises de la bibliothèque `libName` :

```
// "fileURL" et "prefix" ont été saisis par l'utilisateur,
// indiquant à la bibliothèque de balise d'importer le DTD/schema.
var libName = dw.tagLibrary.importDTDOrSchema(fileURL, prefix);

// obtenir le tableau de balises pour cette bibliothèque
// Il s'agit de l'objet TagInfo.
var tagArray = dw.tagLibrary.getImportedTagList(libName);

// maintenant nous obtenons un objet sous forme de tableau contenant les
  informations de balise.
// Nous pouvons en extraire les infos. Ceci extrait les info du premier.
// Remarque : Cela suppose qu'il y a au moins un élément dans le tableau.
var firstTagName = tagArray[0].name;
var firstTagAttributes = tagArray[0].attributes;
// Remarquez que firstTagAttributes est un tableau d'attributs.
```

CHAPITRE 18

Activateurs

18

Les fonctions d'activateur de Macromedia Dreamweaver 8 déterminent si une autre fonction peut effectuer une opération donnée dans le contexte actuel. Les circonstances générales dans lesquelles chaque fonction renvoie la valeur `true` sont décrites dans la spécification de fonction correspondante. Toutefois, ces descriptions ne prétendent pas être exhaustives et ne couvrent pas nécessairement tous les cas où la fonction renverrait la valeur `false`.

Activateurs

Les fonctions d'activateur dans l'API JavaScript comprennent les fonctions suivantes.

dom.canAlign()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Aligner à gauche, Aligner à droite, Aligner en haut ou Aligner en bas.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant si deux calques ou zones réactives au minimum sont sélectionnés.

dom.canApplyTemplate()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Appliquer à la page. Cette fonction n'est valide que pour le document actif.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant si le document n'est pas un élément de bibliothèque ni un modèle, et si la sélection n'est pas encadrée de balises `NOFRAMES`.

dom.canArrange()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Mettre au premier plan ou Mettre en arrière-plan.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant si une zone réactive est sélectionnée.

dom.canClipCopyText()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Copier comme texte.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les décalages d'ouverture et de fermeture sont différents ; `false` dans le cas contraire, pour indiquer qu'aucune sélection n'a été effectuée.

dom.canClipPaste()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Coller.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le Presse-papiers contient des données pouvant être collées dans Dreamweaver ; `false` dans le cas contraire.

dom.canClipPasteText()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Coller comme texte.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le Presse-papiers contient un élément pouvant être collé dans Dreamweaver comme texte ; `false` dans le cas contraire.

dom.canConvertLayersToTable()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Convertir les calques en tableau.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu complet de la section `BODY` du document est compris dans des calques ; `false` dans le cas contraire.

dom.canConvertTablesToLayers()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Convertir les tableaux en calques.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si tout le contenu de la section `BODY` du document figure dans des tableaux et si le document n'est pas basé sur un modèle ; `false` dans le cas contraire.

dom.canDecreaseColspan()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Réduire l'étendue de colonnes.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la cellule active possède un attribut `COLSPAN` et si la valeur de cet attribut est supérieure ou égale à 2 ; `false` dans le cas contraire.

dom.canDecreaseRowspan()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Réduire l'étendue de lignes.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la cellule active possède un attribut `ROWSPAN` et si la valeur de cet attribut est supérieure ou égale à 2 ; `false` dans le cas contraire.

dom.canDeleteTableColumn()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Supprimer la colonne.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le point d'insertion se trouve dans une cellule ou si une cellule ou une colonne est sélectionnée ; `false` dans le cas contraire.

dom.canDeleteTableRow()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Supprimer la ligne.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le point d'insertion se trouve dans une cellule ou si une cellule ou une ligne est sélectionnée ; `false` dans le cas contraire.

site.canEditColumns()

Description

Vérifie si un site existe.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` s'il existe un site et `false` dans le cas contraire.

dom.canEditNoFramesContent()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Modifier le contenu sans cadres.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le document actif est un jeu de cadres ou s'il figure dans un jeu de cadres ; `false` dans le cas contraire.

dom.canIncreaseColspan()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Augmenter l'étendue de colonnes.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` s'il existe des cellules à droite de la cellule active ; `false` dans le cas contraire.

dom.canIncreaseRowspan()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Augmenter l'étendue de lignes.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` s'il existe des cellules au-dessous de la cellule active ; `false` dans le cas contraire.

dom.canInsertTableColumns()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Insérer une colonne.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection se trouve dans un tableau ; `false` si la sélection constitue un tableau entier ou si elle ne se trouve pas dans un tableau.

dom.canInsertTableRows()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Insérer une ligne.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection se trouve dans un tableau ; `false` si la sélection constitue un tableau entier ou si elle ne se trouve pas dans un tableau.

dom.canMakeNewEditableRegion()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Nouvelle région modifiable.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le document actif est un fichier de modèle (DWT).

dom.canMarkSelectionAsEditable()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Marquer la sélection comme modifiable.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` s'il y a une sélection et si le document actif est un fichier DWT ; `false` dans le cas contraire.

dom.canMergeTableCells()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Fusionner les cellules.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection est un regroupement adjacent de cellules de tableau ; `false` dans le cas contraire.

dom.canPlayPlugin()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Lire. Cette fonction n'est valide que pour le document actif.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection peut être exécutée avec un plug-in.

dom.canRedo()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Rétablir.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` s'il reste des opérations à rétablir ; `false` dans le cas contraire.

dom.canRemoveEditableRegion()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Rendre la région non modifiable.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le document actif est un modèle ; `false` dans le cas contraire.

dom.canSelectTable()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Sélectionner le tableau.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection ou le point d'insertion se trouve dans un tableau ; `false` dans le cas contraire.

dom.canSetLinkHref()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut modifier le lien qui entoure la sélection en cours ou en créer un si nécessaire.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection est une image, du texte ou si le point d'insertion se trouve dans un lien ; `false` dans les autres cas. Une sélection de texte se définit comme une sélection pour laquelle l'inspecteur de propriétés de texte s'ouvrirait.

dom.canShowListPropertiesDialog()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut afficher la boîte de dialogue Propriétés de la liste.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection est encadrée de balises `LI` ; `false` dans le cas contraire.

dom.canSplitFrame()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Fractionner le cadre [à gauche | à droite | vers le haut | vers le bas]

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection se trouve dans un cadre ; `false` dans le cas contraire.

dom.canSplitTableCell()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Fractionner la cellule.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le point d'insertion se trouve dans une cellule de tableau ou si la sélection est une cellule de tableau ; `false` dans les autres cas.

dom.canStopPlugin()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Arrêter.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection est exécutée actuellement avec un plug-in ; `false` dans les autres cas.

dom.canUndo()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Annuler.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` s'il reste des opérations à annuler ; `false` dans le cas contraire.

dom.hasTracingImage()

Disponibilité

Dreamweaver 3.

Description

Détermine si le document possède un tracé d'image.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le document possède un tracé d'image ; `false` dans le cas contraire.

dreamweaver.assetPalette.canEdit()

Disponibilité

Dreamweaver 4.

Description

Active les options de menu du panneau Actifs pour permettre leur modification.

Arguments

Aucun.

Valeurs renvoyées

Renvoie une valeur booléenne : `true` si l'actif peut être modifié ; `false` dans le cas contraire. Renvoie la valeur `false` pour les couleurs et les URL de la liste des sites et `false` pour une sélection de plusieurs couleurs et URL dans la liste des favoris.

dreamweaver.assetPalette.canInsertOrApply()

Disponibilité

Dreamweaver 4.

Description

Vérifie si les options sélectionnées peuvent être insérées ou appliquées. Renvoie les valeurs `true` ou `false` pour que l'insertion ou l'application des options de menu puissent être activées ou désactivées.

Arguments

Aucun.

Valeurs renvoyées

Renvoie une valeur booléenne : `true` si les éléments sélectionnés peuvent être insérés ou appliqués ; `false` si la page active est un modèle et que la catégorie en cours est Templates. La fonction renvoie également la valeur `false` si aucun document n'est ouvert ou si un élément de bibliothèque est sélectionné dans le document et que la catégorie active est Bibliothèque.

dreamweaver.canClipCopy()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Copier.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si un contenu sélectionné peut être copié dans le Presse-papiers ; `false` dans le cas contraire.

dreamweaver.canClipCut()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Couper.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si un contenu sélectionné peut être coupé dans le Presse-papiers ; `false` dans le cas contraire.

dreamweaver.canClipPaste()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Coller.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu du Presse-papiers, le cas échéant, peut être collé dans le document actif, dans la fenêtre active du panneau Site ou, sur Macintosh, dans un champ de texte d'un panneau flottant ou d'une boîte de dialogue ; `false` dans le cas contraire.

dreamweaver.canDeleteSelection()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut supprimer la sélection en cours. Celle-ci peut se trouver, selon le cas, dans la fenêtre de document, dans le panneau Site ou, sur Macintosh, dans un champ de texte d'un panneau flottant ou d'une boîte de dialogue.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si les décalages d'ouverture et de fermeture de la sélection sont différents, ce qui indique qu'une sélection est effectuée ; `false` s'ils sont identiques et qu'il n'y a donc qu'un point d'insertion.

dreamweaver.canExportCSS()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Exporter les styles CSS.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si un document contient des styles de classe définis dans la section `HEAD` ; `false` dans le cas contraire.

dreamweaver.canExportTemplateDataAsXML()

Disponibilité

Dreamweaver MX.

Description

Vérifie si Dreamweaver peut exporter le document actif au format XML.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le document actif peut être exporté et `false` dans le cas contraire.

Exemple

Dans l'exemple suivant, `dw.canExportTemplateDataAsXML()` est appelé pour déterminer si Dreamweaver peut exporter le document actif au format XML. S'il renvoie la valeur `true`, il appelle `dw.ExportTemplateDataAsXML()` pour l'exporter :

```
if(dreamweaver.canExportTemplateDataAsXML())
{
  dreamweaver.exportTemplateDataAsXML("file:///c|/dw_temps/
  mytemplate.txt")
}
```

dreamweaver.canFindNext()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Rechercher le suivant.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si un modèle de recherche a été défini ; `false` dans le cas contraire.

dreamweaver.canFitSelection()

Disponibilité

Dreamweaver 8.

Description

Vérifie si un élément est sélectionné dans un mode Création actif, ce qui signifie que `fitSelection()` peut être appelée.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si un élément est sélectionné dans un mode Création actif ; `false` dans le cas contraire.

dreamweaver.canOpenInFrame()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Ouvrir dans un cadre.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection ou le point d'insertion se trouve dans un cadre ; `false` dans le cas contraire.

dreamweaver.canPasteSpecial()

Disponibilité

Dreamweaver 8.

Description

Vérifie si Dreamweaver peut effectuer une opération Collage spécial.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le Presse-papiers contient du texte, du code HTML ou du code HTML pour Dreamweaver et que le mode Code, Création ou Inspecteur de code est actif ; `false` dans le cas contraire.

dreamweaver.canPlayRecordedCommand()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Reproduire la commande enregistrée.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` s'il existe un document actif et une commande enregistrée pouvant être exécutée ; `false` dans le cas contraire.

dreamweaver.canPopupEditTagDialog()

Disponibilité

Dreamweaver MX.

Description

Vérifie si la sélection en cours est une balise et si l'élément de menu Modifier la balise est actif.

Arguments

Aucun.

Valeurs renvoyées

Nom de la balise sélectionnée ou la valeur `null` si aucune balise n'est sélectionnée.

dreamweaver.canRedo()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Rétablir dans le contexte en cours.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant s'il existe des opérations pouvant être annulées.

dreamweaver.canRevertDocument()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Rétablir (revenir au dernier état enregistré).

Arguments

`documentObject`

- L'argument *`documentObject`* correspond à l'objet situé à la racine de l'arborescence DOM d'un document (c'est-à-dire la valeur renvoyée par la fonction `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Valeur booléenne indiquant si le document est à l'état non enregistré et s'il en existe une version enregistrée sur un lecteur local.

dreamweaver.canSaveAll()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Enregistrer tout.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant si deux documents non enregistrés, ou plus, sont ouverts.

dreamweaver.canSaveDocument()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Enregistrer sur le document spécifié.

Arguments

documentObject

- L'argument *documentObject* est la racine d'un DOM de document (valeur identique à celle renvoyée par `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Valeur booléenne indiquant si le document contient des modifications non enregistrées.

dreamweaver.canSaveDocumentAsTemplate()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Enregistrer comme modèle sur le document spécifié.

Arguments

documentObject

- L'argument *documentObject* est la racine d'un DOM de document (valeur identique à celle renvoyée par `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Valeur booléenne indiquant si le document peut être enregistré comme modèle.

dreamweaver.canSaveFrameset()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Enregistrer le jeu de cadres sur le document spécifié.

Arguments

documentObject

- L'argument *documentObject* est la racine d'un DOM de document (valeur identique à celle renvoyée par `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Valeur booléenne indiquant si le document est un jeu de cadres comportant des modifications non enregistrées.

dreamweaver.canSaveFramesetAs()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Enregistrer le jeu de cadres sous sur le document spécifié.

Arguments

documentObject

- L'argument *documentObject* est la racine d'un DOM de document (valeur identique à celle renvoyée par `dreamweaver.getDocumentDOM()`).

Valeurs renvoyées

Valeur booléenne qui indique si le document est un jeu de cadres.

dreamweaver.canSelectAll()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Sélectionner tout.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant s'il est possible d'effectuer une opération Sélectionner tout.

dreamweaver.canShowFindDialog()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Rechercher.

Arguments

Aucun.

Valeurs renvoyées

Une valeur booléenne qui est `true` si une fenêtre de document ou un panneau Site est ouvert. Cette fonction renvoie la valeur `false` lorsque la sélection se trouve dans la section `HEAD`.

dreamweaver.canUndo()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Annuler dans le contexte en cours.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant s'il existe des opérations pouvant être annulées.

dreamweaver.canZoom()

Disponibilité

Dreamweaver 8.

Description

Vérifie si un mode Création est actif, ce qui signifie que les commandes de base de zoom peuvent être appliquées.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si un mode Création est actif ; `false` dans le cas contraire.

dreamweaver.cssRuleTracker.canEditSelectedRule()

Disponibilité

Dreamweaver MX 2004.

Description

Vérifie si l'éditeur de grille de propriété peut s'appliquer à la règle sélectionnée. La grille de propriété pouvant afficher les règles dans les fichiers verrouillés, la valeur de renvoi `true` ne garantit pas que les règles peuvent être modifiées.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si l'éditeur de grille de propriété peut s'appliquer à la règle sélectionnée ; `false` dans le cas contraire.

Exemple

Le code suivant vérifie que la fonction de l'activateur a été définie sur la valeur `true` avant d'autoriser les modifications sur la règle sélectionnée :

```
if(dw.cssRuleTracker.canEditSelectedRule()){
  dw.cssRuleTracker.editSelectedRule();
}
```

dreamweaver.cssStylePalette.canApplySelectedStyle()

Disponibilité

Dreamweaver MX.

Description

Vérifie si le style sélectionné peut être appliqué au document actif.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Valeur booléenne : `true` si le style sélectionné comporte un sélecteur de classes ; `false` dans le cas contraire.

dreamweaver.cssStylePalette.canDeleteSelectedStyle()

Disponibilité

Dreamweaver MX.

Description

Vérifie si le style sélectionné peut être supprimé de la sélection en cours.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection peut être supprimée ; `false` dans le cas contraire.

dreamweaver.cssStylePalette.canDuplicateSelectedStyle()

Disponibilité

Dreamweaver MX.

Description

Vérifie si le style sélectionné peut être dupliqué dans le document actif.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Valeur booléenne : `true` si le style sélectionné peut être dupliqué ; `false` dans le cas contraire.

dreamweaver.cssStylePalette.canEditSelectedStyle()

Disponibilité

Dreamweaver MX.

Description

Vérifie si le style sélectionné peut être modifié dans le document actif.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Valeur booléenne : `true` si le style sélectionné peut être modifié ; `false` dans le cas contraire.

dreamweaver.cssStylePalette.canEditSelectedStyleInCodeview()

Disponibilité

Dreamweaver MX.

Description

Vérifie si le style sélectionné peut être modifié dans le document actif en mode Code.

Arguments

{ pane }

- L'argument facultatif *pane* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Valeur booléenne : `true` si le style sélectionné peut être modifié ; `false` dans le cas contraire.

dreamweaver.cssStylePalette.canEditStyleSheet()

Disponibilité

Dreamweaver MX.

Description

Vérifie si la sélection en cours contient des éléments de feuille de style modifiables.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la sélection est un nœud de feuille de style ou une définition de style n'est pas masquée et si elle est distincte de ce document ; `false` si la sélection est masquée ou si elle se trouve dans ce document.

dreamweaver.cssStylePalette.canRenameSelectedStyle()

Disponibilité

Dreamweaver MX.

Description

Vérifie si le style sélectionné peut être renommé dans le document actif.

Arguments

`{ pane }`

- L'argument facultatif *`pane`* est une chaîne spécifiant le volet du panneau Styles auquel cette fonction est appliquée. Les valeurs possibles sont les suivantes : `"stylelist"`, qui correspond à la liste des styles dans le mode “Tout” ; `"cascade"`, qui correspond à la liste des règles applicables concernées dans le mode “Courant” ; `"summary"`, qui correspond à la liste des propriétés de la sélection en cours dans le mode “Courant” et `"ruleInspector"`, qui correspond à la liste ou grille de propriétés modifiable dans le mode “Courant”. La valeur par défaut est `"stylelist"`.

Valeurs renvoyées

Valeur booléenne : `true` si le style sélectionné peut être renommé ; `false` dans le cas contraire.

dreamweaver.isRecording()

Disponibilité

Dreamweaver 3.

Description

Indique si Dreamweaver est en train de mémoriser une commande.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant si Dreamweaver est en train d'enregistrer une commande.

dreamweaver.htmlStylePalette.canEditSelection()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut modifier, supprimer ou dupliquer la sélection dans le panneau Styles HTML.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si Dreamweaver peut modifier, supprimer ou dupliquer la sélection dans le panneau Styles HTML ; `false` si aucun style n'est sélectionné ou si l'un des styles indéterminés est sélectionné.

dreamweaver.resultsPalette.canClear()

Disponibilité

Dreamweaver MX.

Description

Vérifie que vous pouvez effacer le contenu du panneau Résultats actuellement actif.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu peut être effacé ; `false` dans le cas contraire.

dreamweaver.resultsPalette.canCopy()

Disponibilité

Dreamweaver MX.

Description

Vérifie si la fenêtre de résultats peut afficher un message copié dans son contenu.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu peut être affiché ; `false` dans le cas contraire.

dreamweaver.resultsPalette.canCut()

Disponibilité

Dreamweaver MX.

Description

Vérifie si la fenêtre de résultats peut afficher un message coupé dans son contenu.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu peut être affiché ; `false` dans le cas contraire.

dreamweaver.resultsPalette.canPaste()

Disponibilité

Dreamweaver MX.

Description

Vérifie si la fenêtre de résultats peut afficher un message collé dans son contenu.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu peut être affiché ; `false` dans le cas contraire.

dreamweaver.resultsPalette.canOpenInBrowser()

Disponibilité

Dreamweaver MX.

Description

Vérifie si le rapport en cours peut être affiché dans un navigateur.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu peut être affiché ; `false` dans le cas contraire.

dreamweaver.resultsPalette.canOpenInEditor()

Disponibilité

Dreamweaver MX.

Description

Vérifie si le rapport en cours peut être affiché dans un éditeur.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le contenu peut être affiché ; `false` dans le cas contraire.

dreamweaver.resultsPalette.canSave()

Disponibilité

Dreamweaver MX.

Description

Vérifie si la boîte de dialogue d'enregistrement peut être lancée pour le panneau en cours. Actuellement, les panneaux Rapports du site, Vérification du navigateur cible, Validation et Vérificateur de lien prennent en charge la boîte de dialogue d'enregistrement.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true`, si la boîte de dialogue d'enregistrement apparaît ; `false` dans le cas contraire.

dreamweaver.resultsPalette.canSelectAll()

Disponibilité

Dreamweaver MX.

Description

Vérifie si un message Sélectionner tout peut être transmis à la fenêtre active.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true`, si le message Sélectionner tout peut être transmis ; `false` dans le cas contraire.

dreamweaver.siteSyncDialog.canCompare()

Disponibilité

Dreamweaver 8.

Description

Cette fonction vérifie si le menu contextuel Comparer peut être affiché dans la boîte de dialogue Synchroniser le site.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le menu contextuel Comparer peut être affiché dans la boîte de dialogue Synchroniser le site ; `false` dans le cas contraire..

dreamweaver.siteSyncDialog.canMarkDelete()

Disponibilité

Dreamweaver 8.

Description

Cette fonction vérifie si le menu contextuel Modifier l'action à supprimer peut être affiché dans la boîte de dialogue Synchroniser le site.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le menu contextuel Modifier l'action à supprimer peut être affiché ; `false` dans le cas contraire.

dreamweaver.siteSyncDialog.canMarkGet()

Disponibilité

Dreamweaver 8.

Description

Cette fonction vérifie si le menu contextuel Modifier l'action à obtenir peut être affiché dans la boîte de dialogue Synchroniser le site.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le menu contextuel Modifier l'action à obtenir peut être affiché ; `false` dans le cas contraire.

dreamweaver.siteSyncDialog.canMarkIgnore()

Disponibilité

Dreamweaver 8.

Description

Cette fonction vérifie si le menu contextuel Modifier l'action à ignorer peut être affiché dans la boîte de dialogue Synchroniser le site.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le menu contextuel Modifier l'action à ignorer peut être affiché ; `false` dans le cas contraire.

dreamweaver.siteSyncDialog.canMarkPut()

Disponibilité

Dreamweaver 8.

Description

Cette fonction vérifie si le menu contextuel Modifier l'action à placer peut être affiché dans la boîte de dialogue Synchroniser le site.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le menu contextuel Modifier l'action à placer peut être affiché ; `false` dans le cas contraire.

dreamweaver.siteSyncDialog.canMarkSynced()

Disponibilité

Dreamweaver 8.

Description

Cette fonction vérifie si le menu contextuel Modifier l'action à synchroniser peut être affiché dans la boîte de dialogue Synchroniser le site.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le menu contextuel Modifier l'action à synchroniser peut être affiché ; `false` dans le cas contraire.

dreamweaver.snippetpalette.canEditSnippet()

Disponibilité

Dreamweaver MX.

Description

Vérifie si vous pouvez modifier l'élément sélectionné et renvoie la valeur `true` ou la valeur `false` pour vous permettre d'activer ou de désactiver les éléments de menu destinés à l'édition.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si vous pouvez modifier l'élément sélectionné ; `false` dans le cas contraire.

dreamweaver.snippetpalette.canInsert()

Disponibilité

Dreamweaver MX.

Description

Vérifie si vous pouvez insérer ou appliquer l'élément sélectionné et renvoie la valeur `true` ou la valeur `false` pour vous permettre d'activer ou de désactiver les éléments de menu destinés à l'insertion ou à l'application.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si vous pouvez insérer ou appliquer l'élément sélectionné ; `false` dans le cas contraire.

site.browseDocument()

Disponibilité

Dreamweaver 4.

Description

Ouvre tous les documents sélectionnés dans une fenêtre de navigateur. Cela revient à utiliser la commande Aperçu dans le navigateur.

Arguments

`browserName`.

- L'argument *`browserName`* est le nom du navigateur tel qu'il est défini dans les préférences d'aperçu dans le navigateur. Si cet argument n'est pas défini, le navigateur principal de l'utilisateur est utilisé par défaut.

Valeurs renvoyées

Aucune.

site.canAddLink()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Lier au [fichier existant | nouveau fichier].

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le document sélectionné dans la carte du site est un fichier HTML ; `false` dans le cas contraire.

site.canChangeLink()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Modifier le lien.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si un fichier HTML ou Flash est lié au fichier sélectionné dans la carte du site ; `false` dans le cas contraire.

site.canCheckIn()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Archiver.

Arguments

`siteOrURL`.

- L'argument *`siteOrURL`* doit être soit le mot-clé `site`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Valeur booléenne : `true` si les conditions suivantes sont vraies ; `false` dans le cas contraire.

- Un site distant a été défini.

- Dans le cas où une fenêtre de document est active, le fichier a été enregistré sur un site local ou, dans le cas où le panneau Site est actif, un ou plusieurs fichiers ou dossiers sont sélectionnés.
- La fonction Check In/Check Out est activée pour le site.

site.canCheckOut()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Extraire sur le ou les fichiers spécifiés.

Arguments

`siteOrURL`

- L'argument *`siteOrURL`* doit être soit le mot-clé `"site"`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Valeur booléenne : `true` si toutes les conditions suivantes sont vraies ; `false` dans le cas contraire.

- Un site distant a été défini.
- Dans le cas où une fenêtre de document est active, le fichier appartient à un site local et il n'est pas déjà extrait ou, dans le cas où le panneau Site est actif, plusieurs fichiers ou dossiers sont sélectionnés et au moins un des fichiers sélectionnés n'a pas encore été extrait.
- La fonction Check In/Check Out est activée pour le site.

site.canCloak()

Disponibilité

Dreamweaver MX.

Description

Détermine si Dreamweaver peut effectuer une opération de voilage.

Arguments

siteOrURL

- L'argument *siteOrURL* doit être soit le mot-clé `site`, indiquant que la fonction `canCloak()` doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un dossier donné, indiquant que la fonction `canCloak()` doit agir sur le dossier spécifié et l'ensemble de son contenu.

Valeurs renvoyées

Valeur booléenne : `true` si Dreamweaver peut exécuter l'opération de voilage sur le site en cours ou le dossier spécifié ; `false` dans le cas contraire.

site.canCompareFiles()

Disponibilité

Dreamweaver 8.

Description

Cette fonction vérifie si Dreamweaver peut exécuter la fonction Comparer sur des fichiers sélectionnés.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si deux fichiers (un fichier local et un fichier distant, deux fichiers locaux ou deux fichiers distants) sont sélectionnés ; `false` dans le cas contraire.

site.canConnect()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut se connecter au site distant.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le site distant en cours est un site FTP ; `false` dans le cas contraire.

site.canFindLinkSource()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Rechercher la source du lien.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant que le lien sélectionné dans la carte du site n'est pas la page d'accueil.

site.canGet()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Acquérir.

Arguments

`siteOrURL`

- L'argument *`siteOrURL`* doit être soit le mot-clé `"site"`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Si l'argument est `site`, valeur booléenne indiquant si un ou plusieurs fichiers ou dossiers sont sélectionnés dans le panneau Site et si un site distant a été défini. Si l'argument est une URL, valeur booléenne indiquant si le document appartient à un site pour lequel un site distant a été défini.

site.canLocateInSite()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Retrouver sur le site local ou Retrouver sur le site distant (en fonction de l'argument)

Arguments

localOrRemote, siteOrURL

- L'argument *localOrRemote* doit avoir la valeur `"local"` ou `"remote"`.
- L'argument *siteOrURL* doit être soit le mot-clé `site`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

L'une des valeurs suivantes :

- si le premier argument est le mot-clé `local` et le second une URL, valeur booléenne indiquant si le document appartient à un site ;
- si le premier argument est le mot-clé `remote` et le second une URL, valeur booléenne indiquant si le document appartient à un site pour lequel un site local a été défini et, si le type de serveur est Local/Réseau, si le lecteur est monté ;
- si le second argument est le mot-clé `site`, valeur booléenne indiquant si les deux fenêtres contiennent des fichiers de site (et non la carte du site) et si la sélection se trouve dans le volet opposé à l'argument.

site.canMakeEditable()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Désactiver le mode Lecture seule.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si Dreamweaver peut effectuer une opération Désactiver le mode Lecture seule ; `false` si un ou plusieurs des fichiers sélectionnés sont verrouillés.

site.canMakeNewFileOrFolder()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Nouveau fichier ou Nouveau dossier dans le panneau Site.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si des fichiers sont visibles dans le volet sélectionné du panneau Site ; `false` dans le cas contraire.

site.canOpen()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut ouvrir les fichiers ou les dossiers actuellement sélectionnés dans le panneau Site.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si des fichiers ou des dossiers sont sélectionnés dans le panneau Site ; `false` dans les autres cas.

site.canPut()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Placer.

Arguments

siteOrURL

- L'argument *siteOrURL* doit être soit le mot-clé "site", indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

L'une des valeurs suivantes :

- si l'argument est le mot-clé site, renvoie la valeur true si des fichiers ou des dossiers sont sélectionnés dans le panneau Site et si un site distant a été défini ; renvoie la valeur false dans les autres cas ;
- si l'argument est une URL, renvoie la valeur true si le document appartient à un site pour lequel un site distant a été défini ; renvoie la valeur false dans le cas contraire.

site.canRecreateCache()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Recréer le cache du site.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : true si l'option Utiliser le cache pour accélérer les mises à jour des liens est activée pour le site en cours.

site.canRefresh()

Disponibilité

Dreamweaver 3.

Description

Vérifie si Dreamweaver peut effectuer une opération Actualiser [local | distant].

Arguments

localOrRemote

- L'argument *localOrRemote* doit avoir le mot-clé `"local"` ou `"remote"`.

Valeurs renvoyées

Valeur `true` si l'argument *localOrRemote* est le mot-clé `local` ; sinon, une valeur booléenne indiquant si un site distant a été défini.

site.canRemoveLink()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Supprimer le lien.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant qu'un fichier HTML ou Flash est lié au fichier sélectionné dans la carte du site.

site.canSetLayout()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Mise en forme.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la carte du site est visible ; `false` dans les autres cas.

site.canSelectAllCheckedOutFiles()

Disponibilité

Dreamweaver 4.

Description

Détermine si la fonction Archiver/Extraire est activée sur le site en cours.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le site autorise l'archivage et l'extraction ; `false` dans le cas contraire.

site.canSelectNewer()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Sélectionner [distants | locaux] plus récents.

Arguments

localOrRemote

- L'argument *localOrRemote* doit avoir le mot-clé `"local"` ou `"remote"`.

Valeurs renvoyées

Valeur booléenne indiquant si le document appartient à un site pour lequel un site distant a été défini.

site.canShowPageTitles()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Afficher les titres de page.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si la carte du site est visible; `false` dans le cas contraire.

site.canSynchronize()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Synchroniser.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne indiquant si un site distant a été défini.

site.canUncloak()

Disponibilité

Dreamweaver MX.

Description

Détermine si Dreamweaver peut effectuer une opération de suppression du voile.

Arguments

siteOrURL

- L'argument *siteOrURL* doit être soit le mot-clé `site`, indiquant que la fonction `canUncloak()` doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un dossier donné, indiquant que la fonction `canUncloak()` doit agir sur le dossier spécifié et l'ensemble de son contenu.

Valeurs renvoyées

Valeur booléenne : `true` si Dreamweaver peut exécuter l'opération de suppression de voilage sur le site en cours ou le dossier spécifié ; `false` dans le cas contraire.

site.canUndoCheckOut()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Annuler extraction.

Arguments

siteOrURL

- L'argument *siteOrURL* doit être soit le mot-clé `site`, indiquant que la fonction doit agir sur l'élément sélectionné dans le panneau Site, soit l'URL d'un fichier.

Valeurs renvoyées

Valeur booléenne : `true` si le fichier spécifié ou l'un des fichiers sélectionnés, au moins, a été extrait.

site.canViewAsRoot()

Disponibilité

Dreamweaver 3.

Description

Détermine si Dreamweaver peut effectuer une opération Afficher comme racine.

Arguments

Aucun.

Valeurs renvoyées

Valeur booléenne : `true` si le fichier spécifié est un fichier HTML ou Flash ; `false` dans le cas contraire.

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z