

ADOBE® ILLUSTRATOR® CS4



ADOBE ILLUSTRATOR CS4 GUIDE DES SCRIPTS



© 2008 Adobe Systems Incorporated. Tous droits réservés.

Guide des scripts pour Adobe Illustrator CS4

Si le présent guide est fourni avec un logiciel régi par un contrat d'utilisateur final, le guide, ainsi que le logiciel décrit, sont fournis sous licence et peuvent être utilisés ou copiés uniquement selon les clauses et conditions de la licence. A moins d'une autorisation expresse accordée par cette licence, aucune partie de ce guide ne peut être reproduite, stockée dans un système d'interrogation ou transmise, sous quelque forme ou par quelque moyen que ce soit (électronique, mécanique, enregistrement ou autre) sans l'autorisation écrite préalable d'Adobe Systems Incorporated. Veuillez noter que le contenu du présent guide est protégé par la loi sur les droits d'auteur, même s'il n'est pas distribué avec un logiciel régi par un contrat de licence utilisateur.

Les informations contenues dans ce guide sont fournies à titre purement informatif ; elles sont susceptibles d'être modifiées sans préavis et ne doivent pas être interprétées comme étant un engagement de la part d'Adobe Systems Incorporated. Adobe Systems Incorporated n'accepte aucune responsabilité quant aux erreurs ou inexactitudes pouvant être contenues dans le présent guide.

Veuillez noter que les illustrations existantes que vous souhaitez éventuellement inclure dans votre projet sont susceptibles d'être protégées par les lois sur les droits d'auteur. L'inclusion non autorisée de tels éléments dans vos nouveaux travaux peut constituer une violation des droits du propriétaire. Veuillez vous assurer de détenir toute autorisation nécessaire auprès du détenteur des droits.

Toute référence à des noms de sociétés dans des modèles ou images types n'est utilisée qu'à titre d'exemple et ne fait référence à aucune société réelle.

Adobe, le logo Adobe, Illustrator, Photoshop et InDesign sont des marques ou des marques déposées d'Adobe Systems Incorporated aux Etats-Unis et/ou dans d'autres pays.

Microsoft et Windows sont des marques ou des marques déposées de Microsoft Corporation aux Etats-Unis et/ou dans d'autres pays. Apple, Mac, Macintosh et Mac OS sont des marques déposées d'Apple Computer, Incorporated aux Etats-Unis et dans d'autres pays. JavaScript et toutes les marques relatives à Java sont des marques ou des marques déposées de Sun Microsystems, Incorporated aux Etats-Unis et dans d'autres pays. UNIX est une marque déposée de The Open Group.

Toutes les autres marques citées sont la propriété de leurs détenteurs respectifs.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, Californie 95110, Etats-Unis. Avis aux utilisateurs du gouvernement des Etats-Unis. Le logiciel et la documentation sont des « articles commerciaux », conformément à la définition du terme « Commercial Items » dans l'article 48 C.F.R. §2.101 du Code de la réglementation fédérale (Code of Federal Regulations), qui consistent en du « logiciel informatique commercial » et de la « documentation informatique commerciale », tel qu'il est mentionné dans les articles 48 C.F.R. §12.212 et 48 C.F.R. §227.7202. Conformément aux articles 48 C.F.R. §12.212 et 48 C.F.R. §§227.7202-1 à 227.7202-4, le logiciel commercial et la documentation logicielle commerciale sont fournis sous licence aux utilisateurs du gouvernement des Etats-Unis (a) uniquement à titre d'articles commerciaux (b) et leur confèrent seulement les droits octroyés à tous les autres utilisateurs selon les conditions mentionnées aux présentes. Droits non publiés réservés dans le cadre des lois sur les droits d'auteur en vigueur aux Etats-Unis. Adobe Systems Incorporated, 345 Park Avenue, San Jose, CA 95110-2704, Etats-Unis. Adobe s'engage à respecter la législation relative à l'égalité des chances y compris, le cas échéant, les dispositions du décret 11246, tel qu'amendé, à la section 402 de la loi sur l'assistance aux vétérans du Vietnam (Vietnam Era Veterans Readjustment Assistance Act) de 1974 (38 USC 4212), et à la section 503 de la loi sur la réadaptation (Rehabilitation Act) de 1973, telle qu'amendée, et la réglementation des articles 41 CFR, alinéas 60-1 à 60-60, 60-250 et 60-741. La clause d'action positive et la réglementation décrites dans la phrase précédente sont incluses par référence.

Sommaire

1	Introduction	6
	Qu'est-ce qu'un script ?	6
	Pourquoi utiliser les scripts ?	6
	Qu'est-ce qu'une action ?	6
	Prise en charge des scripts dans Adobe Illustrator CS4	7
	Fonctions ExtendScript	7
	Outils ExtendScript	7
	Extensions de scripts de fichiers	8
	Affichage des exemples de scripts	8
	Affichage du modèle d'objet	8
	Affichage du modèle d'objet JavaScript	8
	Affichage du modèle d'objet AppleScript	9
	Affichage du modèle d'objet VBScript	9
	Exécution de scripts	10
	Installation de scripts dans le menu Scripts	10
	Exécution de scripts à partir de l'élément de menu Autre script	10
	Scripts de démarrage (scripts .jsx uniquement)	11
	Changements survenus depuis la version CS3	12
	Problèmes connus	16
2	Modèle d'objet de script Illustrator	18
	Conventions d'attribution de nom pour les objets	19
	Objets de niveau supérieur (objets conteneurs)	19
	Application	19
	Document	20
	Calque	20
	Hiérarchie des illustrations	20
	Styles d'illustration	21
	Coloration d'objets	22
	Objets texte	22
	Blocs de texte	22
	Objets représentant le contenu de texte	23
	Styles de texte	24
	Objets dynamiques	25
	Symboles	25
	Transformations	25

3	Scripts Illustrator	26
	Lancement et fermeture d'Illustrator à partir d'un script	26
	Lancement et activation d'Illustrator	26
	Fermeture d'Illustrator	27
	Utilisation d'objets	27
	Obtention du document ou du calque situé au premier plan	27
	Création d'objets	27
	Objets de collection	28
	Objets sélectionnés	29
	Remarques sur le changement de nom d'objets stockés dans les panneaux de l'application	29
	Unités de mesure	30
	Unités em	30
	Positionnement et dimensions d'un élément de page	31
	Limites des éléments d'illustration	31
	Tracés et formes	32
	Niveaux d'interaction de l'utilisateur	33
	Impression de documents Illustrator	34
4	Création de scripts à l'aide d'AppleScript	35
	Pour plus d'informations	35
	Votre premier script Illustrator	35
	Ajout de caractéristiques à « Hello World »	36
	Référencement d'objets	36
	Obtention d'objets de documents et calques	37
	Création d'objets	37
	Utilisation de sélections	37
	Utilisation de blocs de texte	38
	Blocs liés	38
	Création de tracés et de formes	39
	Tracés	39
	Formes	40
5	Création de scripts à l'aide de JavaScript	42
	Pour plus d'informations	42
	Votre premier script Illustrator	42
	Ajout de caractéristiques à « Hello World »	43
	Utilisation de méthodes dans JavaScript	43
	Accès et référencement d'objets	44
	Référencement de l'objet application	44
	Accès aux objets de collection	44
	Création d'objets	45
	Utilisation de sélections	46
	Utilisation de blocs de texte	46
	Blocs liés	46

- Création de tracés et de formes 47
 - Tracés 47
 - Formes 48
- 6 Création de scripts à l'aide de VBScript 50**
 - Pour plus d'informations 50
 - Votre premier script Illustrator 50
 - Ajout de caractéristiques à « Hello World » 51
 - Accès et référencement d'objets 51
 - Obtention d'objets de collection 51
 - Création d'objets 52
 - Utilisation de sélections 52
 - Utilisation de blocs de texte 53
 - Blocs liés 53
 - Création de tracés et de formes 54
 - Tracés 54
 - Formes 55
 - Utilisation de valeurs d'énumération 56
- Index 58**

1 Introduction

Ce guide décrit l'interface de script pour Adobe® Illustrator® CS4.

Si vous êtes novice en matière de scripts ou que vous souhaitez obtenir des informations sur la manière d'utiliser les différents langages de script, reportez-vous au document *Introduction aux scripts Adobe*.

Qu'est-ce qu'un script ?

Un script est une série de commandes demandant à Illustrator d'exécuter une ou plusieurs opérations. Ces opérations peuvent être simples et n'affecter qu'un objet du document actuel, ou complexes et affecter des objets dans tous vos documents Illustrator. Ces opérations peuvent aussi impliquer d'autres applications, telles que des programmes de traitement de texte, des tableurs ou des logiciels de gestion de base de données.

La plupart des blocs de construction du script correspondent aux outils Illustrator, aux menus, aux panneaux et aux boîtes de dialogue pour lesquels vous possédez une expertise. Si vous savez ce que vous souhaitez exécuter par le biais d'Illustrator, vous pouvez écrire un script à cet effet.

Pourquoi utiliser les scripts ?

La conception graphique est un domaine où la créativité est reine, mais certains aspects de la conception ne sont pas du tout créatifs. Vous aurez sans doute remarqué que le positionnement ou le repositionnement d'images, la rectification d'erreurs dans le texte et la préparation des fichiers pour impression chez un fournisseur de services de réglage d'image empiètent largement sur le temps dont vous disposez pour le travail créatif à proprement parler.

Si vous investissez un peu de temps et d'efforts, vous pouvez apprendre à écrire des scripts courts et simples qui effectuent les opérations répétitives à votre place. Au fur et à mesure du développement de vos compétences, vous pourrez écrire des scripts de plus en plus complexes.

Les scripts peuvent également améliorer votre créativité en effectuant rapidement des opérations que vous pourriez ne pas avoir le temps de tester. Par exemple, vous pouvez écrire un script pour créer systématiquement une série d'objets, en modifiant les propriétés de position, contour et fond de nouveaux objets à un moment donné. Vous pouvez également écrire un script qui accède aux fonctions de matrice de transformation intégrée afin d'étirer, de mettre à l'échelle et de distordre une série d'objets. Sans script, vous perdriez certainement une partie du potentiel créatif de telles techniques qui requièrent une masse importante de travail.

Qu'est-ce qu'une action ?

Les actions et les scripts sont deux manières d'automatiser des opérations répétitives, mais elles fonctionnent de deux façons différentes :

- Les actions utilisent une interface utilisateur de programmes. Lors de l'exécution d'une action, les choix de menus sont exécutés, les objets sont sélectionnés et les tracés enregistrés sont créés. Les scripts n'utilisent pas d'interface utilisateur de programme pour effectuer les opérations et peuvent s'exécuter plus rapidement que les actions.

- Les actions possèdent des fonctionnalités très limitées pour obtenir des informations et y répondre. Vous ne pouvez pas ajouter de logique conditionnelle à une action. Par conséquent, les actions ne peuvent pas prendre de décisions reposant sur la situation actuelle, telles que le changement de type de contour des rectangles mais pas des ellipses. Les scripts sont capables d'obtenir des informations, de prendre des décisions et d'effectuer des calculs reposant sur les informations qu'ils reçoivent d'Illustrator.
- Un script peut exécuter une action, mais les actions ne peuvent pas exécuter de scripts.

Prise en charge des scripts dans Adobe Illustrator CS4

Les scripts d'Illustrator prennent en charge les scripts VBScript et JavaScript pour Windows, et les scripts AppleScript et JavaScript pour Mac OS.

REMARQUE : les applications Adobe adaptées aux scripts, y compris Illustrator, prennent aussi en charge ExtendScript, le système de mise en œuvre étendu de JavaScript ECMA Adobe. Les fichiers ExtendScript se distinguent par leur extension `.jsx`. Si vous attribuez une extension `.jsx` à vos fichiers JavaScript, vous pouvez tirer parti des outils et des fonctionnalités ExtendScript.

Fonctions ExtendScript

ExtendScript propose toutes les fonctions JavaScript standard, ainsi qu'un environnement de développement et de débogage, l'utilitaire d'écriture de scripts ExtendScript Toolkit (ESTK). ESTK est installé avec toutes les applications Adobe pouvant prendre en charge les scripts.

ESTK comprend un afficheur de modèle d'objet qui contient une documentation complète concernant les méthodes et propriétés des objets JavaScript.

Pour plus de détails sur l'accès à ESTK et à l'afficheur de modèle d'objet, reportez-vous à la section [« Affichage du modèle d'objet JavaScript », page 8](#).

Outils ExtendScript

ExtendScript offre également divers outils et utilitaires tels que :

- un utilitaire de localisation ;
- des outils qui vous permettent de combiner les scripts et de les diriger vers des applications spécifiques ;
- une représentation de fichiers et de dossiers indépendante des plates-formes ;
- des outils pour construire des interfaces utilisateur dans vos scripts ;
- un cadre de messagerie électronique qui vous permet d'envoyer et de recevoir des scripts et des données dans les applications Adobe adaptées aux scripts.

Pour plus de détails sur ces fonctions et sur d'autres fonctions, reportez-vous au *Guide des outils JavaScript*.

Extensions de scripts de fichiers

Pour qu'un fichier soit reconnu par Adobe Illustrator CS4 en tant que fichier de script valide, il doit posséder un nom d'extension de fichier correct :

Type de script	Type de fichier	Extension	Plates-formes
AppleScript	script compilé fichier OSAS	.sct (aucune)	Mac OS
JavaScript	texte	.js	Windows
ExtendScript		.jsx	Mac OS
VBScript	texte	.vbs	Windows

Affichage des exemples de scripts

Adobe fournit des exemples de scripts pour de nombreux objets, propriétés et méthodes dans le modèle d'objets de document (DOM) Illustrator CS4. Vous pouvez consulter des exemples de scripts à deux emplacements :

- Dans le dossier `/Scripts/Exemples de script` du répertoire d'installation d'Illustrator CS4
- Dans le document de référence de scripts d'Adobe Illustrator CS4 pour votre langage de script, situé dans le dossier `/Scripts/Documentation` de votre répertoire d'installation d'Illustrator CS4.

Affichage du modèle d'objet

Les langages de script pris en charge fournissent chacun une fonction permettant d'afficher les objets de script définis par Illustrator avec des détails de référence.

Affichage du modèle d'objet JavaScript

Pour afficher le modèle d'objet JavaScript dans Illustrator, procédez comme suit :

1. Démarrez l'utilitaire d'écriture de scripts ESTK.

Dans une installation Adobe par défaut, ESTK se trouve à l'emplacement suivant :

- Windows :
`disque système\Program Files\Adobe\Adobe Utilities\ExtendScript Toolkit CS4`
- Mac OS :
`disque système:Applications:Utilitaires:Utilitaires Adobe:ExtendScript Toolkit CS4`

2. Dans ESTK, choisissez la commande Aide > Outil de visualisation d'Object Model.
3. Dans la fenêtre Outil de visualisation d'Object Model, sélectionnez Adobe Illustrator CS4 Type Library dans la liste déroulante Navigateur.

Plusieurs exemples de scripts étendus sont disponibles dans le dossier `/Scripts/Exemples de script` du répertoire d'installation d'Illustrator CS4.

Vous pouvez également consulter des exemples de scripts et d'informations concernant les classes, les objets, les propriétés, les méthodes et les paramètres individuels dans le guide *Adobe Illustrator CS4 Scripting Reference: JavaScript* (Référence de script Adobe Illustrator CS4 : JavaScript) que vous pouvez trouver dans le dossier `/Scripts/Documentation` du répertoire d'installation d'Illustrator CS4.

Affichage du modèle d'objet AppleScript

Apple fournit l'application Editeur de scripts avec tous les systèmes Mac OS. Vous pouvez utiliser l'application Editeur de scripts pour consulter le dictionnaire AppleScript qui décrit les divers objets et commandes d'Illustrator.

Pour plus de détails sur l'utilisation de l'Editeur de scripts, reportez-vous à l'aide de l'Editeur de scripts.

1. Lancez l'application Editeur de scripts.

REMARQUE : dans une installation Mac OS par défaut, l'application Editeur de scripts se trouve dans `Applications:AppleScript:Script Editor`. Si vous ne trouvez pas l'application Editeur de scripts, vous devez la réinstaller à partir du CD du système Mac OS.

2. Choisissez la commande Fichier > Ouvrir un dictionnaire. Une boîte de dialogue Ouvrir un dictionnaire s'affiche dans l'application Editeur de scripts.
3. Dans cette boîte de dialogue, recherchez l'application Adobe Illustrator CS4, puis sélectionnez-la et cliquez sur le bouton Ouvrir.

La liste des objets et commandes Illustrator, qui inclut les propriétés et éléments associés à chacun des objets et les paramètres pour chaque commande, s'affiche dans l'application Editeur de scripts.

Plusieurs exemples de scripts étendus sont disponibles dans le dossier `:Scripts:Exemples de script` du répertoire d'installation d'Illustrator CS4.

Vous pouvez également consulter des exemples de scripts et d'informations concernant les classes, les objets, les propriétés, les méthodes et les paramètres individuels dans le guide *Adobe Illustrator CS4 Scripting Reference: AppleScript* (Référence de script Adobe Illustrator CS4 : AppleScript) que vous pouvez trouver dans le dossier `:Scripts:Documentation` du répertoire d'installation d'Illustrator CS4.

Affichage du modèle d'objet VBScript

VBScript fournit une bibliothèque type que vous pouvez utiliser pour consulter les propriétés et méthodes des objets Illustrator. Cette procédure explique comment consulter la bibliothèque type à partir de n'importe quel programme Microsoft Office. Votre éditeur VBScript possède certainement un accès à la bibliothèque. Pour plus d'informations, reportez-vous à l'aide de votre éditeur.

1. A partir de toute application Microsoft Office, choisissez la commande Outils > Macro > Visual Basic Editor.
2. Dans Visual Basic Editor, choisissez la commande Outils > Références.
3. Dans la boîte de dialogue qui s'affiche, cochez la case correspondant à la bibliothèque Adobe Illustrator CS4 Type Library, puis cliquez sur le bouton OK.
4. Choisissez la commande Affichage > Explorateur d'objets pour afficher la fenêtre Explorateur d'objets.
5. Choisissez l'option Illustrator dans la liste de bibliothèques ouvertes du menu déroulant situé en haut à gauche de la fenêtre Explorateur d'objets.

Plusieurs exemples de scripts étendus sont disponibles dans le dossier `/Scripts/Exemples de script` du répertoire d'installation d'Illustrator CS4.

Vous pouvez également consulter des exemples de scripts et d'informations concernant les classes, les objets, les propriétés, les méthodes et les paramètres individuels dans le guide *Adobe Illustrator CS4 Scripting Reference: VBScript* (Référence de script Adobe Illustrator CS4 : VBScript) que vous pouvez trouver dans le dossier `/Scripts/Documentation` du répertoire d'installation d'Illustrator CS4.

Exécution de scripts

L'interface Illustrator comprend un menu Scripts (Fichier > Scripts) qui fournit un accès facile et rapide à vos scripts.

Les scripts peuvent être directement répertoriés en tant qu'éléments de menu qui s'exécutent lorsque vous les sélectionnez. Reportez-vous à la section [« Installation de scripts dans le menu Scripts », page 10](#).

Vous pouvez également naviguer vers n'importe quel script de votre système de fichiers à partir du menu, puis exécuter le script. Reportez-vous à la section [« Exécution de scripts à partir de l'élément de menu Autre script », page 10](#).

Vous pouvez également choisir des scripts JavaScript avec une extension `.jsx` qui démarrent automatiquement lorsque vous lancez l'application. Pour plus de détails, reportez-vous à la section [« Scripts de démarrage \(scripts .jsx uniquement\) », page 11](#).

Installation de scripts dans le menu Scripts

Pour inclure un script dans le menu Scripts (Fichier > Scripts), enregistrez le script dans le dossier Scripts qui se situe dans le dossier `/Illustrator CS4/Paramètres prédéfinis` de votre répertoire d'installation d'Illustrator CS4. Le nom de fichier du script, sans l'extension du fichier, apparaît dans le menu Scripts.

Les scripts que vous ajoutez au dossier Scripts lorsqu'Illustrator est en cours d'exécution apparaîtront dans le menu Scripts au prochain lancement d'Illustrator.

Vous pouvez installer un nombre illimité de scripts dans le menu Scripts. Si vous possédez un grand nombre de scripts, utilisez des sous-dossiers dans le dossier Scripts pour mieux organiser les scripts du menu Scripts. Chaque sous-dossier apparaît en tant que sous-menu séparé comportant les scripts de ce sous-dossier.

Exécution de scripts à partir de l'élément de menu Autre script

L'élément Autre script figurant au bas du menu Scripts (Fichier > Scripts > Autre script) vous permet d'exécuter des scripts qui ne sont pas installés dans le dossier Scripts.

Lorsque vous sélectionnez l'option Autre script, la boîte de dialogue Ouvrir s'affiche : utilisez-la pour accéder à un fichier de script. Lorsque vous sélectionnez le fichier, le script s'exécute.

Seuls les fichiers dont les types sont pris en charge apparaissent dans cette boîte de dialogue. Pour plus de détails, reportez-vous à la section [« Prise en charge des scripts dans Adobe Illustrator CS4 », page 7](#).

Scripts de démarrage (scripts .jsx uniquement)

Les scripts JavaScript possédant une extension de fichier `.jsx` peuvent être installés dans un ou deux dossiers afin que les scripts s'exécutent automatiquement lorsque vous lancez Illustrator ou à chaque exécution de script. Ces dossiers sont les suivants :

- un dossier de scripts de démarrage spécifique à l'application, qui contient des scripts pour Illustrator CS4 ;
- un dossier de scripts de démarrage général, qui contient des scripts s'exécutant automatiquement lorsque vous lancez toute application Creative Suite 4.

Dossier de scripts de démarrage spécifique à l'application

Vous devez placer les scripts de démarrage spécifiques à l'application dans un fichier appelé `Startup Scripts`, que vous créez dans le répertoire d'installation d'Illustrator.

Par exemple, lorsqu'Illustrator CS4 est installé dans son emplacement par défaut, vous créez le dossier `Startup Scripts` aux emplacements suivants :

- Windows : `C:\Program Files\Adobe\Adobe Illustrator CS4\Startup Scripts\`
- Mac OS : `/Applications/Adobe Illustrator CS4/Startup Scripts/`

Les scripts JavaScript qui possèdent une extension `.jsx` et qui sont placés dans le dossier `Startup Scripts` s'exécutent automatiquement lorsque :

- l'application est lancée ;
- tout fichier JavaScript est sélectionné à partir du menu `Scripts (Fichier > Scripts)`.

Dossier de scripts de démarrage général

Le dossier de scripts de démarrage général comporte des scripts qui s'exécutent automatiquement lorsque vous lancez une application Creative Suite 4. Vous créez le dossier aux emplacements suivants :

- Windows : `Program Files/Common Files/Adobe/Startup Scripts CS4/Illustrator`
- Mac OS : `:Bibliothèque:Application Support:Adobe:Startup Scripts CS4:Illustrator`

Si un script du dossier de démarrage général doit uniquement être exécuté par Illustrator, le script doit inclure la directive `ExtendScript #target (#target illustrator)` ou le code suivant :

```
if( BridgeTalk.appName == "illustrator" ) {
    //continue executing script
}
```

Pour plus de détails, reportez-vous au *Guide des outils JavaScript*.

Changements survenus depuis la version CS3

Cette section énumère les changements apportés au modèle d'objet de script afin de prendre en charge les fonctions d'Illustrator CS4.

- Un nouveau type, `DocumentArtboardLayout`, définit la disposition des plans de travail dans un document. Il est utilisé par la classe `DocumentPreset` et la méthode `Add` dans la classe `Document`. Les dispositions possibles sont `GridByRow`, `GridByCol`, `Row`, `Column`, `RLGridByRow`, `RLGridByCol` et `RLRow`, par exemple :
 - AppleScript : `artboardLayout : grid by row`
 - JavaScript : `DocumentArtboardLayout.GridByRow`
 - VBScript : `AiDocumentArtboardLayout.aiGridByRow`
- Cinq nouvelles propriétés de la méthode `Add` de la collection `Documents` (`NumArtboards`, `ArtboardLayout`, `ArtboardSpacing`, `ArtboardRowsOrCols` et `MaxCanvasSizeEnabled`) prennent en charge la nouvelle fonction à plusieurs plans de travail.
- Cinq nouvelles propriétés de la classe `DocumentPreset` (`NumArtboards`, `ArtboardLayout`, `ArtboardSpacing`, `ArtboardRowsOrCols` et `MaxCanvasSizeEnabled`) prennent en charge la nouvelle fonction de plans de travail multiples.
- Deux nouvelles propriétés (`ArtboardRange` et `SaveMultipleArtboards`) prennent en charge la nouvelle fonction de plans de travail multiples. Elles s'appliquent à plusieurs classes : `ExportOptionsFlash`, `ExportOptionsPhotoshop`, `EPSSaveOptions`, `MXGSaveOptions` et `PDFSaveOptions` (`ArtboardRange` uniquement).
- Une nouvelle classe, `NonNativeItem`, représente un élément non natif dans un document Illustrator. Il s'agit d'une autre sous-classe de `PageItem`. Son parent peut être `Document`, `Group` ou `Layer`.
 - AppleScript : `non native item`
 - JavaScript : `nonNativeItem`
 - VBScript : `NonNativeItem`
- Une nouvelle collection, `NonNativeItems`, contient une collection d'objets `NonNativeItem`.
 - AppleScript : `non native items of document 1`
 - JavaScript : `app.activeDocument.nonNativeItems`
 - VBScript : `App.ActiveDocument.NonNativeItems`
- Un nouveau format de fichier FXG prend en charge un flux de travail créateurs-développeurs, en permettant l'enregistrement d'un document Illustrator en tant que fichier FXG, contenant des descriptions de style XML d'éléments graphiques d'Illustrator, qui peuvent ensuite être utilisés dans d'autres applications basées sur FXG/XML.
 - AppleScript : `fxg`
 - JavaScript : `DocumentType.FXG`
 - VBScript : `AiDocumentType.aiFXG`
- Deux nouvelles normes PDFX ont été ajoutées, `PDFX32002` et `PDFX42007`.

- Un nouveau type, `FXGVersion`, a été ajouté pour définir la version du fichier FXG. Une seule option est disponible : `FXG version 1.0`.
 - `AppleScript` : `version: version 1.0`
 - `JavaScript` : `FXGVersion.VERSION1.0`
 - `VBScript` : `AiFXGVersion.aiVersion1.0`
- Un nouveau type, `FiltersPreservePolicy`, définit s'il faut conserver l'apparence et le caractère modifiable des filtres lors de l'enregistrement au format FXG. Les options disponibles sont `ExpandFilters`, `RasterizeFilters` et `KeepFiltersEditable`.
 - `AppleScript` : `filters policy: keep filters editable`
 - `JavaScript` : `FiltersPreservePolicy.KEEPFILTERSEEDITABLE`
 - `VBScript` : `AiFiltersPreservePolicy.aiKeepFiltersEditable`
- Un nouveau type, `GradientsPreservePolicy`, définit s'il faut conserver l'apparence et le caractère modifiable des dégradés lors de l'enregistrement au format FXG. Les options disponibles sont `RasterizeGradients` et `KeepGradientsEditable`.
 - `AppleScript` : `gradients policy: keep gradients editable`
 - `JavaScript` : `GradientsPreservePolicy.KEEPGRADIENSTSEEDITABLE`
 - `VBScript` : `AiGradientsPreservePolicy.aiKeepGradientsEditable`
- Un nouveau type, `TextPreservePolicy`, définit s'il faut conserver l'apparence et le caractère modifiable du texte lors de l'enregistrement au format FXG. Les options disponibles sont `OutlineText`, `RasterizeText` et `KeepTextEditable`.
 - `AppleScript` : `text policy: keep text editable`
 - `JavaScript` : `TextPreservePolicy.KEEPTEXTEEDITABLE`
 - `VBScript` : `AiTextPreservePolicy.aiKeepTextEditable`
- Une nouvelle propriété `AutoKernType`, `MetricsRomanOnly`, indique un crénage pour les chiffres romains uniquement.
 - `AppleScript` : `character: MetricsRomanOnly, character style: MetricsRomanOnly, line: MetricsRomanOnly, paragraph: MetricsRomanOnly, paragraph style: MetricsRomanOnly, word: MetricsRomanOnly`
 - `JavaScript` : `AutoKernType.METRICSRomanOnly`
 - `VBScript` : `AiAutoKernType.aiMetricsRomanOnly`
- Une nouvelle propriété `FlashExportStyle`, `ArtBoardsToFiles`, permet d'exporter les plans de travail d'un document dans un fichier SWF distinct.
 - `AppleScript` : `export style: Artboards to Files`
 - `JavaScript` : `FlashExportStyle.ARTBOARDSTOFILES`
 - `VBScript` : `AiFlashExportStyle.aiArtboardsToFiles`

- La constante `CropBounds PrintingBounds` est devenue obsolète, maintenant que les plans de travail prennent en charge une nouvelle fonction de plans de travail multiples.
- Une prise en charge des formes/du codage de glyphes JIS90 et JIS04 a été ajoutée à `AlternateGlyphsForm`, par exemple :
 - **AppleScript** : `alternate glyphs: jis90`
 - **JavaScript** : `AlternateGlyphsForm.JIS90`
 - **VBScript** : `AiAlternateGlyphsForm.aiJIS90Form`
- De nouvelles propriétés ont été ajoutées à la classe `Application` : numéro de version et langue, par exemple :
 - **AppleScript** : `build number of application`
 - **JavaScript** : `app.buildNumber`
 - **VBScript** : `App.BuildNumber`
- Une nouvelle propriété de la classe `Document`, `RasterEffectSettings`, contient les paramètres d'effet pixellisé d'un document.
 - **AppleScript** : `raster effect settings of document 1`
 - **JavaScript** : `app.activeDocument.rasterEffectSettings`
 - **VBScript** : `App.ActiveDocument.RasterEffectSettings`
- Une nouvelle méthode a été ajoutée à la classe `Application`, `ConvertSampleColors`, pour convertir l'espace colorimétrique d'une couleur.
 - **AppleScript** : `set cmykColor to convert sample color source color space RGB source color rgbColor destination color space CMYK color conversion purpose default purpose`
 - **JavaScript** : `var cmykColor = app.convertSampleColor(ImageColorSpace.RGB, new Array(93, 32, 10), ImageColorSpace.CMYK, ColorConvertPurpose.defaultpurpose);`
 - **VBScript** : `cmykColor = appRef.ConvertSampleColor(2, Array(93, 32, 10), 3, 0)`
- Deux nouvelles méthodes ont été ajoutées à la classe `Document`. `WindowCapture` capture le document actif en image TIFF. `Rasterize` pixellise l'illustration source dans des limites spécifiées.
 - **AppleScript** : `capture document 1, rasterize document 1`
 - **JavaScript** : `app.activeDocument.windowCapture, app.activeDocument.rasterize`
 - **VBScript** : `App.ActiveDocument.WindowCapture, App.ActiveDocument.Rasterize`
- Une nouvelle méthode dans la collection `Swatches`, `GetSelected`, renvoie une liste `SwatchList` d'objets `Swatch` actuellement sélectionnés.
 - **AppleScript** : `get selected swatches of document 1`
 - **JavaScript** : `app.activeDocument.swatches.getSelected`
 - **VBScript** : `App.ActiveDocument.Swatches.GetSelected`

- Une nouvelle classe, `SwatchGroup`, contient un groupe d'objets `Swatch`. Elle contient deux méthodes : `GetAllSwatches` et `AddSwatch`.
 - AppleScript : `swatch group 1 of document 1`
 - JavaScript : `app.activeDocument.swatchGroups [0]`
 - VBScript : `App.ActiveDocument.SwatchGroups (1)`
- Une nouvelle collection, `SwatchGroups`, contient tous les objets `SwatchGroup` d'un document. Elle contient deux méthodes : `Remove` et `Add`.
 - AppleScript : `swatch groups in document 1`
 - JavaScript : `app.activeDocument.swatchGroups`
 - VBScript : `App.ActiveDocument.SwatchGroups`
- Un nouveau type, `SwatchList`, définit une collection d'objets `Swatch`. Il est utilisé par l'objet `SwatchGroup` et la collection `Swatches`.
- Une nouvelle propriété dans la classe `GradientStop`, `opacity`, permet à chaque point d'arrêt de définir son propre degré de transparence. La fusion alpha se produit entre les points d'arrêt.
 - AppleScript : `stop opacity of gradient stop`
 - JavaScript : `gradientStop.opacity`
 - VBScript : `GradientStop.Opacity`
- Une nouvelle propriété dans la classe `OpenOptionsAutoCAD`, `MergeLayers`, définit si les calques du fichier AutoCAD en cours d'ouverture doivent être fusionnés en un seul calque dans un document `Illustrator`.
 - AppleScript : `merge layers : true`
 - JavaScript : `mergeLayers = true`
 - VBScript : `MergeLayers = true`
- Une nouvelle classe, `Artboard`, définit un objet de plan de travail ou un plan de travail dans un document CS4. Elle comporte les propriétés `ArtboardRect`, `RulerPAR`, `ShowRulers`, `ShowCenter`, `ShowCrossHairs`, `ShowSafeAreas` et `ShowScreenEdge`.
 - JavaScript : `app.activeDocument.artboards [0]`
 - AppleScript : `artboard 1 of document 1`
 - VBScript : `App.ActiveDocument.Artboards (1)`
- Une nouvelle collection, `Artboards`, contient tous les objets `Artboard` d'un document. Elle comporte les méthodes `Remove`, `Add`, `GetActiveArtboardIndex` et `SetActiveArtboardIndex`.
 - AppleScript : `artboards of document 1`
 - JavaScript : `app.activeDocument.artboards`
 - VBScript : `App.ActiveDocument.Artboards`

- Deux nouvelles propriétés dans la classe `PrintJobOptions`, `PrintAllArtboards` et `ArtboardRange`, prennent en charge l'impression de plusieurs plans de travail.
- Un nouveau type, `RasterizationColorModel`, permet de spécifier le modèle de couleur sur `default`, `grayscale` ou `bitmap`, par exemple :
 - AppleScript : `color model : grayscale rasterization`
 - JavaScript : `RasterizationColorModel.GRAyscale`
 - VBScript : `AiRasterizationColorModel.aiGrayscale`
- Un nouveau type, `AntiAliasingMethod`, contrôle le type de méthode de pixellisation utilisée dans la pixellisation : `none`, `art optimized` et `type optimized`, par exemple :
 - AppleScript : `antialiasing method : art optimized`
 - JavaScript : `AntiAliasingMethod.ARTOPTIMIZED`
 - VBScript : `AiAntiAliasingMethod.aiArtOptimized`
- Une nouvelle classe, `RasterEffectOptions`, indique les paramètres d'effet de pixellisation d'un document. Ses propriétés sont `ColorModel`, `Resolution`, `Transparency`, `AntiAliasing`, `ClippingMask`, `ConvertSpotColors` et `Padding`.
 - AppleScript : `raster effect settings of document 1`
 - JavaScript : `app.activeDocument.rasterEffectSettings`
 - VBScript : `App.ActiveDocument.RasterEffectSettings`
- Une nouvelle classe, `RasterizeOptions`, indique les options pouvant être utilisées lors de la pixellisation d'illustrations. Ses propriétés sont `ColorModel`, `Resolution`, `Transparency`, `AntiAliasing`, `ClippingMask`, `ConvertSpotColors`, `ConvertTextToOutlines`, `IncludeLayers`, `BackgroundBlack` et `Padding`.
 - AppleScript : `rasterize document 1 uses rasterize options`
 - JavaScript : `app.activeDocument.rasterize uses rasterizeOptions`
 - VBScript : `App.ActiveDocument.Rasterize uses RasterizeOptions`

Problèmes connus

- Les scripts qui créent, enregistrent et ferment un grand nombre de fichiers Illustrator devraient régulièrement fermer Illustrator, puis le relancer. Le nombre maximal recommandé de fichiers à traiter avant de quitter Illustrator, puis de le relancer est de :
 - Sous Windows 500 fichiers
 - Sous Mac OS 1 000 fichiers

Pour plus d'informations sur la manière dont vous pouvez quitter et relancer Illustrator, reportez-vous aux sections « [Lancement et activation d'Illustrator](#) », page 26 et « [Fermeture d'Illustrator](#) », page 27.

- Le message d'avertissement « An Illustrator error occurred: 1346458189 ("PARM") » peut apparaître quand des scripts mal rédigés sont exécutés plusieurs fois dans Illustrator depuis ESTK.

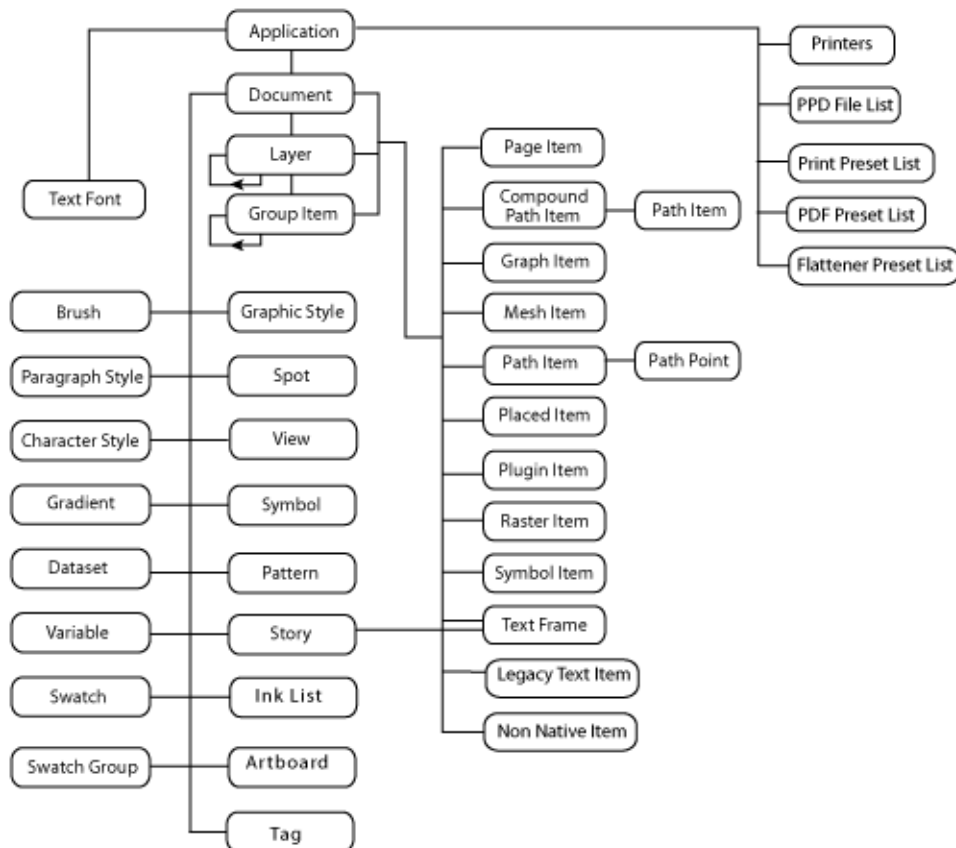
Les auteurs de scripts doivent faire très attention à l'initialisation des variables ainsi qu'aux conflits engendrés par les espaces de noms lorsqu'ils lancent l'exécution répétée d'un lot de scripts Illustrator dans Illustrator par le biais d'ESTK, et ce dans le cadre d'une session Illustrator unique. Chaque script lancé est exécuté au sein du même moteur ExtendScript persistant dans Illustrator.

Le débogueur ESTK utilise BridgeTalk pour communiquer avec Illustrator. Un moteur ExtendScript persistant, global et unique au sein d'Illustrator gère l'ensemble des communications BridgeTalk. Grâce à Internet, le moteur ExtendScript s'ajoute à tous les scripts exécutés précédemment. Les problèmes de code de script pouvant être à l'origine de cet incident sont les suivants :

- lecture de variables non initialisées ;
- conflits d'espaces de noms globaux, par exemple lorsque deux ensembles de scripts différents s'opposent.

2 Modèle d'objet de script Illustrator

Bien comprendre le modèle d'objet Illustrator vous permettra d'améliorer vos capacités d'écriture de script. Le schéma suivant présente la hiérarchie de contenance du modèle d'objet, qui commence par l'objet `application`. Les classes `layer` et `group item` peuvent comporter des objets imbriqués susceptibles, eux aussi, de renfermer des objets imbriqués supplémentaires.



En plus de ces modèles d'objets spécifiques aux applications, JavaScript fournit certains objets utilitaires, tels que les objets `File` et `Folder`, qui vous permettent un accès au système de fichiers indépendant du système d'exploitation. Pour plus de détails, reportez-vous au *Guide des outils JavaScript*.

Conventions d'attribution de nom pour les objets

Il existe un modèle d'objet pour l'interface de script Illustrator, mais les noms d'objets réels varient légèrement selon les langages de script :

- Sous AppleScript, les noms figurent tous en minuscules et les mots individuels sont séparés par un espace, par exemple :

```
graphic style
```

- Sous VBScript, les noms commencent par une majuscule et les autres mots contenus dans le nom se distinguent par une première lettre en majuscule, par exemple :

```
GraphicStyle
```

- Sous JavaScript, le nom commence par une minuscule et les autres mots contenus dans le nom se distinguent par une première lettre en majuscule, par exemple :

```
graphicStyle
```

Ce chapitre utilise des noms d'objets et de propriétés génériques, mais vous pouvez appliquer aisément ces conventions pour déterminer les noms correspondant à un langage spécifique.

Les noms de propriétés, noms de méthodes et objets cités dans ce document sont en police `monospaced`.

Objets de niveau supérieur (objets conteneurs)

Utilisez ces objets pour accéder à des informations globales sur l'application Illustrator ou à un document individuel.

Application

Les propriétés de l'objet `application` permettent à votre script d'accéder à des valeurs globales, telles que :

- la propriété `preferences utilisateur` qu'un utilisateur définit de manière interactive dans l'application Illustrator à l'aide de la boîte de dialogue Préférences (Edition > Préférences) ;
- les informations système telles que les polices installées (propriété `text fonts`) et les imprimantes (propriété `printer list`).

Il existe aussi des propriétés qui fournissent des informations spécifiques aux applications et des informations de niveau supérieur relatives à tout document ouvert :

- les informations relatives aux applications telles que le chemin d'installation (`path`), la `version` et le caractère `visible` d'Illustrator ;
- le document sélectionné (`current active`), c'est-à-dire la toile affichée et l'acceptation de la saisie de texte de l'utilisateur ;
- tous les `documents` ouverts.

Les commandes ou méthodes d'objets `application` permettent à votre script d'effectuer des actions sur les applications, par exemple :

- ouvrir (`open`) des fichiers ;
- annuler (`undo`) et rétablir (`redo`) des transactions ;
- fermer (`quit`) Illustrator.

Document

L'objet `document` que vos scripts peuvent créer ou auquel ils peuvent accéder à partir de l'objet `application` représente une toile ou un fichier Illustrator chargé. Les propriétés de l'objet `document` vous permettent d'accéder au contenu du document, par exemple :

- sélection (`selection`) en cours, ou objets graphiques sélectionnés par l'utilisateur dans le document ;
- tous les objets graphiques contenus, appelés `page items`, qui organisent la hiérarchie des illustrations ;
- objets graphiques particuliers, de type `symbols` et `text frames` ;
- tous les calques (`layers`) et le calque sélectionné (`active layer`).

Les propriétés du document vous renseignent également sur l'état du document en lui-même, par exemple :

- paramètres utilisateur pour le document tels que `ruler units` ;
- enregistrement (`saved`) éventuel du document depuis le dernier changement apporté au contenu ;
- chemin (`path`) du fichier associé.

Les méthodes de l'objet `document` permettent à vos scripts d'agir sur le document, par exemple :

- Enregistrer (`save`) un fichier au format Illustrator ou enregistrer sous (`save as`) les différents formats de fichiers pris en charge ;
- Activer (`activate`) ou fermer (`close`) un document ;
- Imprimer (`print`) le document. Vos scripts peuvent sélectionner une imprimante en référençant un objet `print options` ou les imprimantes disponibles via la propriété `printer list` de l'objet.

Calque

L'objet `layer` permet d'accéder au contenu, ou hiérarchie des illustrations, d'un calque spécifique. Vous pouvez accéder à l'objet `layer` via l'objet `document`. Les propriétés de l'objet `layer` permettent d'accéder à un calque ou d'obtenir des informations sur un calque, telles que :

- le caractère visible ou verrouillé (`locked`) du calque ;
- le caractère `opacity` du calque (transparence globale) et `z order position` (position dans la pile) ;
- les préférences en matière de création artistique pour le calque, telles que `artwork knockout` et `blending mode`.

Hiérarchie des illustrations

Le contenu d'un document Illustrator est appelé *hiérarchie des illustrations*. L'illustration est représentée par les objets suivants :

- `compound path item`
- `graph item`

- `group item`
- `legacy text item`
- `mesh item`
- `non native item`
- `path item`
- `placed item`
- `plugin item`
- `raster item`
- `symbol item` (voir la section [« Objets dynamiques », page 25](#))
- `text frame`

Vos scripts peuvent accéder à des objets graphiques et les manipuler via des collections à partir des objets `document` et `layer`. Il existe deux types de collections d'objets graphiques :

- Les objets de collection qui correspondent à chaque type d'objet d'illustration individuel, tel que l'objet `graph items` ou l'objet `mesh items`.
- L'objet `page items`, qui inclut tous les types d'objets graphiques.

Vous pouvez également utiliser l'objet `group item` pour référencer un ensemble groupé d'éléments d'illustration.

Vous pouvez créer des objets graphiques à l'aide de la commande `make` sous AppleScript ou de la méthode `add` de l'objet de collection d'un élément d'illustration. Par exemple, pour créer un objet `path item` :

AppleScript `set myPathItem to make new path item in current document`

JavaScript `var myPathItem = activeDocument.pathItems.add();`

VBScript `Set myPathItem = appRef.ActiveDocument.PathItems.Add()`

Les collections d'illustrations suivantes ne permettant pas la création d'objets par le biais de la commande `make` ou de la méthode `add` sont :

- l'objet `graph items` ;
- l'objet `mesh items` ;
- l'objet `plugin items` ;
- l'objet `legacy text items`.

Pour plus d'informations sur la création d'objets de ce type, reportez-vous au document de référence de scripts d'Adobe Illustrator CS4 pour votre langage.

Styles d'illustration

Votre script peut appliquer un style graphique à l'aide de l'objet `graphic style`. Pour appliquer un style graphique, utilisez la propriété `graphic styles` de l'objet `document` afin d'accéder à la méthode `apply to` de l'objet `graphic style`.

De la même manière, l'objet `brush` vous permet d'indiquer la forme à appliquer à l'illustration. Vous pouvez accéder à n'importe quelle forme à partir de l'objet de collection `brushes` qui est une propriété de l'objet `document`.

Coloration d'objets

Votre script peut appliquer une couleur, un motif ou un dégradé à un objet `path item` à l'aide des propriétés `fill color` ou `stroke color` :

- Les scripts peuvent définir de nouvelles nuances de couleur grâce à la commande `make` ou à la méthode `add` de l'objet `swatches`. Votre script peut également créer des tons de couleur à l'aide de la commande `make` ou de la propriété `add` de l'objet `spots`.
- Vous pouvez définir les attributs d'un objet `ink` grâce à l'objet `ink info` qui est une propriété de l'objet `ink`. Vous pouvez accéder aux objets `ink` par le biais de la propriété `ink list` de l'objet `document`.

Les objets ci-après vous permettent de créer des couleurs dans des espaces colorimétriques définis :

- l'objet `RGB color`, que vous pouvez définir à l'aide d'une plage de valeurs allant de 0 à 255 pour chacune des trois valeurs chromatiques individuelles ;
- l'objet `CMYK color`, que vous pouvez définir à l'aide de valeurs de pourcentage allant de 0 à 100 pour chacune des quatre valeurs chromatiques individuelles ;
- les objets `grayscale color` ou `LAB color`, que vous pouvez définir grâce à la même plage et au même nombre de valeurs utilisées dans l'application Illustrator.

Objets texte

Lorsque vous saisissez du contenu dans un document Illustrator, le type devient automatiquement un objet `text frame` et, en même temps, un objet `story`.

Pour observer cela, ouvrez un nouveau document dans Illustrator. Utilisez ensuite l'outil de texte horizontal pour saisir du texte, puis utilisez l'outil de texte vertical pour saisir du texte supplémentaire. Créez enfin un rectangle avant d'y saisir du texte. Exécutez maintenant le script JavaScript suivant :

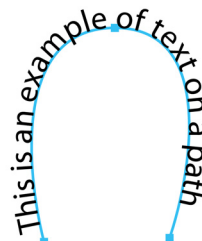
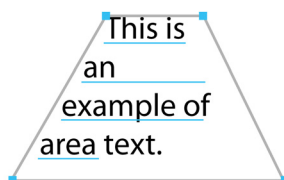
```
var myDoc = app.activeDocument
alert("There are " + myDoc.textFrames.length + " text frames.")
alert("There are " + myDoc.stories.length + " stories.")
```

Blocs de texte

Il existe trois types de bloc de texte :

- point (texte de point)
- area (texte captif)
- path (texte curviligne)

This is an
example of
point text.



Pour créer un type de bloc de texte spécifique, utilisez la propriété `kind` de l'objet `text frames` sous AppleScript. Les objets `text frames` sous JavaScript et VBScript contiennent des méthodes spécifiques de création de blocs de texte captif et curviligne.

Tout comme dans l'application Illustrator, vous pouvez lier des blocs de texte captifs ou curvilignes.

Pour lier des blocs de texte existants, vous devez utiliser la propriété `next frame` ou `previous frame` de l'objet `text frame`. Les blocs liés créent un seul objet `story`.

Pour plus de détails sur la création ou la liaison de blocs de texte, reportez-vous au chapitre de ce guide correspondant à votre langage de script.

Géométrie du texte

Même si les trois types de blocs de texte ont des caractéristiques communes de type `orientation`, chacun possède des qualités qui lui sont propres, qui se reflètent dans les propriétés de l'objet `text frame`. Par exemple :

- Un bloc de texte captif peut comporter des lignes et des colonnes auxquelles vous accédez par le biais des propriétés `row count` et `column count`.
- Le texte curviligne comporte des propriétés `start T value` et `end T value` qui indiquent où le texte commence et se termine sur le tracé.
- Les blocs de texte captifs et curvilignes sont associés à un objet `text path` qui est indiqué à l'aide de la propriété `text path` de l'objet `text frame`. Le tracé de texte définit la position du bloc de texte et son orientation (horizontale ou verticale) sur le plan de travail (la propriété `orientation` de l'objet `text frame` définit l'orientation du texte dans le bloc de texte).

La propriété `text path` n'est pas valide pour le texte de point, car la position et l'orientation de texte de point sont entièrement définies par les propriétés du bloc de texte lui-même.

REMARQUE : un tracé de texte n'est pas le même qu'un élément d'illustration de tracé. Les tracés de texte sont associés aux éléments d'illustration de tracé ; vous pouvez y accéder et les manipuler pour modifier l'apparence du bloc de texte associé.

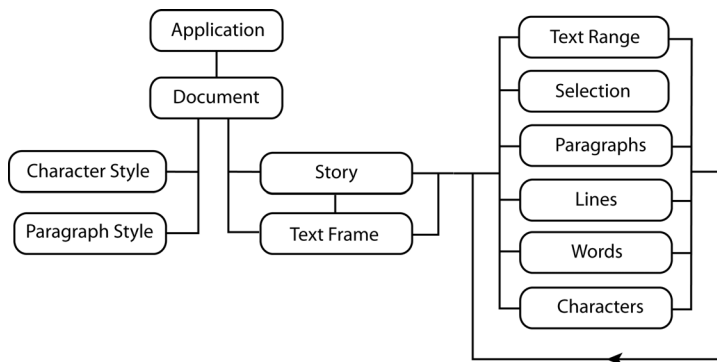
Objets représentant le contenu de texte

Il est possible d'accéder au contenu du texte réel d'un bloc de texte ou d'un article, de même qu'aux objets suivants :

- `characters`
- `words`
- `paragraphs`
- `lines`

Un objet `line` comporte tous les caractères d'une ligne contenus dans un objet `text frame` ou `story`. Tous les éléments d'illustration de texte comportent au moins une ligne de texte définie en tant qu'objet `line`. Les illustrations de texte peuvent comporter plusieurs lignes de texte si le texte contient des retours forcés à la ligne ou si ces caractères passent à la ligne suivante, car ils ne s'adaptent pas à la largeur de l'illustration de texte. Les collections accèdent aux objets de texte et les identifient dans les objets `text`

frame et story, par exemple, `textFrame("My Text Frame").paragraphs` ou `story("My Story").paragraphs`.



Les objets `text frame` et `story` comportent des propriétés `insertion point` et `text selection`. Les propriétés de l'objet `text frame` incluent également les fonctions qui définissent le bloc de texte, telles que :

- largeur (`width`), hauteur (`height`) et position du bloc ;
- caractère masqué (`hidden`) ou verrouillé (`locked`) du bloc ;
- caractère modifiable (`editable`) du texte.

REMARQUE : un objet `line` ne peut pas être créé dans un script. Votre script peut créer des objets `character`, `paragraph` et `word`.

Plages de texte

Les différents objets de texte d'un bloc de texte ou d'un article sont également représentés collectivement par l'objet `text range`. Par exemple, un caractère correspond à une plage de texte d'une longueur de 1 tandis qu'un mot correspond à une plage de texte qui comporte un espace à l'avant.

Vous pouvez définir le contenu d'un objet `text range` en transmettant une chaîne à l'aide de la propriété `contents`.

Styles de texte

Les éléments de style de texte, tels que `font`, `capitalization` et `justification`, sont représentés par les objets `paragraph attribute` et `character attribute`. Ces objets d'attribut sont des propriétés des objets `paragraph style` et `character style`. Les objets `paragraph style` et `character style` possèdent des méthodes `apply to` et `remove` qui permettent à votre script d'attribuer ou d'effacer les attributs d'un paragraphe, d'un caractère ou d'une plage de texte donnés.

Vous pouvez modifier l'affichage des propriétés d'une plage de texte en appliquant un style approprié ou en fournissant des priorités locales des attributs au niveau du texte ou du paragraphe :

- Les objets `character style` s'appliquent à des ensembles d'un ou plusieurs caractères. Ces objets contrôlent des fonctions des caractères telles que `font`, `alignment`, `leading`, `language` et `capitalization`, qui sont des propriétés de l'objet `character attribute`.
- Les objets `paragraph style` s'appliquent à des paragraphes. Ces objets contrôlent des fonctions de paragraphe telles que `first line indent`, `left indent` et `right indent`, qui sont des propriétés de l'objet `paragraph attribute`.

Objets dynamiques

Vous pouvez réaliser des graphiques adaptés aux données en créant des objets dynamiques. Dans l'application Illustrator, utilisez le panneau Variables pour créer ou éditer des variables telles que les données d'un graphe, un fichier lié, une chaîne de caractères et la visibilité ou des variables dont le type n'est pas spécifié. Lors de l'écriture d'un script, l'objet `variable` vous permet de représenter ce type de variable. La propriété `kind` de l'objet `variable` indique le type de données dynamiques que comporte un objet `variable`. Les objets `variable` sont des objets de niveau document : vous pouvez les créer à partir de l'objet `document`.

REMARQUE : ne confondez pas les objets `variable` avec les variables de script. Pour plus d'informations sur les variables Illustrator, les objets dynamiques et les graphiques adaptés aux données, reportez-vous à l'aide d'Illustrator.

Les ensembles de données, qui collectent des variables et leurs données dynamiques associées dans un objet, sont représentés par l'objet `dataset` dans le script. L'objet `dataset` fournit des méthodes permettant de mettre à jour et de supprimer les objets `dataset` dans vos scripts.

Symboles

Dans Illustrator, les symboles sont des éléments d'illustration stockés dans le panneau Symboles. Vos scripts peuvent créer, supprimer et dupliquer les objets `symbol`. Lorsque vous créez des objets `symbol` dans votre script, Illustrator les ajoute à votre panneau Symboles pour le document cible.

Un objet `symbol item` correspond à une instance de l'objet `symbol` dans un document. Chaque objet `symbol item` est lié à sa définition de symbole, afin que la modification de la définition d'un symbole mette à jour toutes les instances de ce symbole.

Votre script peut créer, supprimer et dupliquer les éléments de symbole. Les éléments de symbole sont des éléments d'illustration d'Illustrator et peuvent ainsi être traités de la même manière que les autres éléments d'illustration ou de page. Vous pouvez faire pivoter, redimensionner, sélectionner, verrouiller, masquer et effectuer d'autres opérations sur les éléments de symbole.

Transformations

L'objet `matrix` fournit un accès à la puissance des matrices des transformations géométriques. Les matrices de transformation d'Illustrator stockent les paramètres d'une opération qui met à l'échelle, fait pivoter ou déplace (translate) un objet sur une page. L'utilisation des matrices présente certains avantages :

- Grâce aux valeurs de transformation d'un objet `matrix`, vous pouvez utiliser les valeurs de manière répétée sur d'autres objets dans votre script.
- Grâce à la concaténation de la rotation, la translation et/ou la mise à l'échelle des matrices et à l'application de la matrice obtenue, vous pouvez effectuer de nombreuses transformations géométriques à l'aide d'une instruction de script unique.
- Vous pouvez inverser les valeurs de la matrice.
- Vous pouvez comparer les valeurs de deux matrices.

L'objet `application` offre des commandes ou méthodes qui permettent de créer, d'obtenir, d'inverser, de comparer ou de concaténer des matrices.

La commande ou la méthode utilisée pour appliquer une matrice correspond à la commande `transform` qui appartient à tout type d'objet sur lequel des transformations peuvent être effectuées.

3 Scripts Illustrator

Ce chapitre fournit une présentation sur l'utilisation des objets de script pour programmer Illustrator CS4. Des exemples spécifiques pour les langages de scripts pris en charge sont fournis dans les chapitres qui suivent.

Lancement et fermeture d'Illustrator à partir d'un script

Vos scripts peuvent vous permettre de lancer et de quitter Illustrator.

Lancement et activation d'Illustrator

AppleScript

Dans AppleScript, vous pouvez utiliser une instruction `tell` pour cibler Illustrator. La commande `activate` active Illustrator s'il n'est pas encore actif.

```
tell application "Adobe Illustrator"
    activate
end tell
```

JavaScript

Vous pouvez généralement exécuter des scripts JavaScript à partir du menu Scripts de l'application (Fichier > Scripts) ou du dossier de démarrage ; il n'est pas nécessaire de lancer Illustrator à partir de votre script.

Les informations sur le lancement d'Illustrator dans un script JavaScript ne figurent pas dans ce guide. Pour plus d'informations, faites une recherche sur la « messagerie interapplications » ou sur le « cadre de la messagerie JavaScript » dans le *Guide des outils JavaScript*.

VBScript

Il existe plusieurs façons de créer une instance d'Illustrator :

- `CreateObject` lance Illustrator en tant qu'application invisible si ce dernier n'est pas encore lancé. Si Illustrator est lancé en tant qu'application invisible, vous devez activer manuellement l'application pour la rendre visible :

```
Set appRef = CreateObject("Illustrator.Application")
```

Si plusieurs versions d'Illustrator sont installées sur la même machine et que vous utilisez la méthode `CreateObject` pour obtenir une référence d'application, l'utilisation de `"Illustrator.Application"` crée une référence à la dernière version d'Illustrator. Pour cibler spécifiquement une version précédente, indiquez l'identifiant de version à la fin de la chaîne :

Pour Illustrator 10, utilisez `"Illustrator.Application.1"`

Pour Illustrator CS, utilisez `"Illustrator.Application.2"`

Pour Illustrator CS2, utilisez "Illustrator.Application.3"
 Pour Illustrator CS3, utilisez "Illustrator.Application.4"
 Pour Illustrator CS4, utilisez "Illustrator.Application.CS4"

- Utilisez l'opérateur `New` si vous avez ajouté une référence à la bibliothèque type Illustrator du projet. Par exemple, la ligne suivante crée une référence à l'objet `Application` :

```
Set appRef = New Illustrator.Application
```

Fermeture d'Illustrator

AppleScript

Utilisez la commande `quit` suivante :

```
tell application "Adobe Illustrator"
    quit
end tell
```

JavaScript

Utilisez la méthode `app.quit()` suivante :

```
app.quit()
```

VBScript

Utilisez la méthode `Quit` de l'objet `Application` suivante :

```
Set appRef = CreateObject("Illustrator.Application")
appRef.Quit
```

Utilisation d'objets

Obtention du document ou du calque situé au premier plan

Pour faire référence au document sélectionné, utilisez la propriété `current document` de l'objet `application` dans AppleScript ou la propriété `active document` dans JavaScript ou VBScript. De la même manière, vous pouvez utiliser la propriété `current layer` ou `active layer` de l'objet `document` pour faire référence au calque sélectionné.

Il existe d'autres types de propriétés d'objet « actives » ou « sélectionnées », par exemple `active dataset` ou `active view`. Pour obtenir des détails, reportez-vous au document *Adobe Illustrator CS4 Scripting Reference* (Référence de script Adobe Illustrator CS4) correspondant à votre langage de script.

Création d'objets

Certains objets (outre l'objet `application` lui-même) ne peuvent être obtenus à partir de conteneurs ou d'objets parents. Votre script doit créer ces objets directement.

Les objets suivants doivent être créés spécifiquement :

CMYK color	ink info	print coordinate options
document preset	lab color	printer
EPS save options	matrix	printer info
export options AutoCAD	MXG save options	print flattener options
export options Flash	no color	print font options
export options GIF	open options	print job options
export options JPEG	open options AutoCAD	print options
export options Photoshop	open options FreeHand	print page marks options
export options PNG8	open options PDF	print paper options
export options PNG24	open options Photoshop	print postscript options
export options SVG	paper info	raster effect options
file	Pattern color	rasterize options
folder	PDF save options	screen
gradient color	PPD file	screen spot function
gray color	PPD file info	RGB color
Illustrator save options	print color management options	spot color
ink	print color separation options	tracing options

Les objets `file` et `folder` sont des périphériques Adobe ExtendScript conçus pour fournir un accès indépendant des plates-formes au système de fichier sous-jacent. Pour plus de détails sur l'utilisation de ces objets, reportez-vous au *Guide des outils JavaScript*.

Pour plus d'informations sur la création explicite d'un objet, reportez-vous au chapitre correspondant à votre langage de script.

Objets de collection

Vous pouvez obtenir la plupart des objets de collection à partir d'un conteneur. Par exemple, un objet de collection `path items` peut être contenu dans un objet `document` ou un objet `layer`. Pour obtenir un objet de collection `path items`, vous devez faire référence à un autre objet contenant. Pour obtenir un exemple, reportez-vous aux sections sur les langages ci-dessous.

AppleScript

Pour faire référence à un objet `path items` dans un document :

```
path item 1 in document 1
```

Pour faire référence à un objet `path items` dans un calque :

```
path item 1 in layer 1 in document 1
```

JavaScript

Pour faire référence à un objet `path items` dans un document :

```
documents[0].pathItems[1]
```

Pour faire référence à un objet `path items` dans un calque :

```
documents[0].layers[0].pathItems[0]
```

VBScript

Pour faire référence à un objet `path items` dans un document :

```
Documents (1) .PathItems (1)
```

Pour faire référence à un objet `path items` dans un calque :

```
Documents (1) .Layers (1) .PathItems (1)
```

Pour obtenir plus d'exemples sur les conteneurs d'éléments de collection, reportez-vous au tableau *Elements* de l'objet `document` dans le document *Adobe Illustrator CS4 Scripting Reference: AppleScript* (Référence de script Adobe Illustrator CS4 : AppleScript), ou le tableau *Properties* du document *Adobe Illustrator CS4 Scripting Reference: JavaScript* (Référence de script Adobe Illustrator CS4 : JavaScript) ou *Adobe Illustrator CS4 Scripting Reference: VBScript* (Référence de script Adobe Illustrator CS4 : VBScript). Un diagramme du modèle d'objet Illustrator CS4 est disponible à la section « [Modèle d'objet de script Illustrator](#) », page 18.

Objets sélectionnés

Vous souhaitez parfois écrire des scripts qui agissent sur les objets actuellement sélectionnés ou non. Par exemple, vous souhaitez appliquer une mise en forme donnée au texte sélectionné ou modifier la forme d'un tracé sélectionné.

Sélection de texte

Pour sélectionner du texte, utilisez la commande `select` ou la méthode de l'objet `text range`.

Sélection d'éléments d'illustration

Vous pouvez sélectionner un objet graphique (tel qu'un élément graphique, un élément de filet, un élément pixellisé, un élément de symbole, etc.) en définissant sa propriété `selected` sur `true`. (Dans AppleScript, `selected` est une propriété de l'objet `page items`.)

Référence à des éléments d'illustration sélectionnés

Pour faire référence à tous les objets actuellement sélectionnés d'un document, utilisez la propriété `selection` de l'objet `document`. Pour les objets du tableau de données de sélection, vous devez déterminer leur type, de manière à savoir quelles propriétés et méthodes ou commandes vous pouvez utiliser. Dans JavaScript et VBScript, chaque type d'objet d'illustration possède une propriété `typename` en lecture seule que vous pouvez utiliser pour déterminer le type d'objet. Dans AppleScript, utilisez la propriété `class`.

Remarques sur le changement de nom d'objets stockés dans les panneaux de l'application

Plusieurs objets peuvent être renommés, c'est-à-dire que vous pouvez modifier leur propriété `name`. Les types d'objets ci-après peuvent être classés par ordre alphabétique dans le panneau Illustrator correspondant. Si un script modifie le nom d'un tel objet, les références d'index à cet objet peuvent ne plus être valides.

Brush
 Gradient
 Graphic Style
 Pattern
 Swatch
 Symbol
 Variable

Unités de mesure

Illustrator utilise les points en tant qu'unité de mesure pour presque toutes les distances. 1 pouce correspond à 72 points. Il existe une exception pour les valeurs des propriétés telles que `kerning`, `tracking` et `aki` (utilisée pour les compositions de texte japonaises), pour lesquelles l'unité est l'« em » (voir la section « [Unités em](#) », page 30.)

Illustrator utilise des points lors de la communication avec vos scripts, *quelle que soit l'unité de graduation de la règle*. Si votre script repose sur l'addition, la soustraction, la multiplication ou la division de valeurs de mesure spécifiques pour des unités autres que le point, il doit effectuer toutes les conversions d'unités nécessaires afin que vos mesures soient en points. Par exemple, si vous souhaitez utiliser des pouces pour vos coordonnées ou vos unités de mesure, vous devez multiplier toutes les valeurs en pouces par 72 lorsque vous saisissez les valeurs dans votre script.

Le tableau suivant présente les formules de conversion pour les différentes unités de mesure :

Unité	Formule de conversion
centimètres	28,346 points = 1 centimètre
pouces	72 points = 1 pouce
millimètres	2,834645 points = 1 millimètre
picas	12 points = 1 pica
Q	0,709 point = 1 Q (1 Q équivaut à 0,23 millimètre)

JavaScript fournit le type d'objet `UnitValue`, qui propose des utilitaires de conversion d'unités. Pour plus de détails, reportez-vous au *Guide des outils JavaScript*.

Unités em

Les valeurs qui utilisent l'unité em au lieu de points sont mesurées en millièmes d'unité em.

Une unité em est proportionnelle à la taille du corps de police actuel. Par exemple, avec une police de 6 points, 1 em équivaut à 6 points ; avec une police de 10 points, 1 em équivaut à 10 points. Avec une police de 10 points, une valeur de crénage de 20 unités em correspond à :

$$(20 \text{ unités} \times 10 \text{ points}) / 1000 \text{ unités/em} = 0,2 \text{ points}$$

Positionnement et dimensions d'un élément de page

Illustrator utilise une géométrie simple à deux dimensions sous forme de points pour enregistrer la position des objets `page item` dans un document. Chaque objet `page item` d'un document a une propriété `position` qui définit un point fixe correspondant aux coordonnées de la page au format `[x, y]`. Le point fixe se trouve dans le coin supérieur gauche du cadre de sélection de l'objet.

Pour plus d'informations sur les types d'objets qui constituent la collection `page items`, reportez-vous à la section « [Hiérarchie des illustrations](#) », page 20.

Un point est désigné par des coordonnées :

- la position horizontale, `x` ;
- la position verticale, `y`.

Vous pouvez consulter ces coordonnées depuis le panneau Informations lorsque vous sélectionnez ou que vous créez un objet dans Illustrator.

Le point `ruler origin` (0,0) par défaut, servant de coordonnées numériques dans Illustrator, se trouve dans le coin inférieur gauche du document (`ruler origin` est une propriété de l'objet `document`). Sur l'axe horizontal, les coordonnées situées à la droite du point zéro de la règle sont des nombres positifs. Sur l'axe vertical, les coordonnées situées au-dessus du point zéro sont positives.

La propriété `page origin` d'un objet `document` définit le coin inférieur gauche de la zone imprimable du document en tant que point fixe.

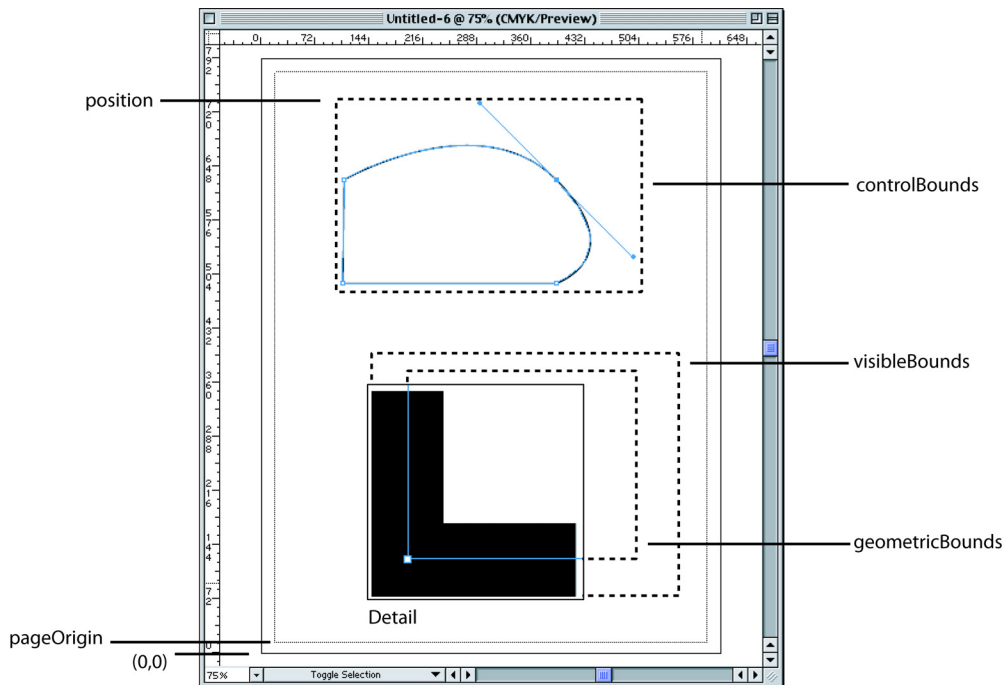
Chaque objet `page item` possède également des propriétés `width` et `height`. La valeur maximale autorisée pour la largeur ou la hauteur de l'élément de la page est de 16 348 points.

Limites des éléments d'illustration

Chaque objet `page item` possède trois propriétés qui utilisent les rectangles fixes pour décrire l'étendue globale de l'objet :

- La propriété `geometric bounds` d'un élément de page correspond aux dimensions rectangulaires du cadre de sélection de l'objet, à l'exclusion de l'épaisseur de contour.
- La propriété `visible bounds` d'un élément de page correspond aux dimensions de l'objet, y compris les épaisseurs de contour.
- La propriété `control bounds` correspond aux dimensions rectangulaires de l'objet, y compris les points de contrôle internes et externes.

L'image suivante illustre ces propriétés, à l'aide des conventions d'attribution de nom JavaScript.

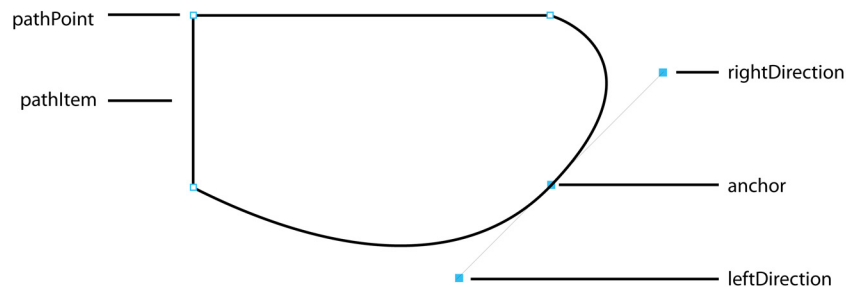


Tracés et formes

Les tracés sont représentés dans le modèle d'objets de document (DOM) Illustrator par l'objet `path item`. Les éléments de tracé comprennent toutes les illustrations qui contiennent des tracés, telles que les rectangles, les ellipses et les polygones, de même que les tracés libres.

Un tracé libre consiste en une série de points de tracé. Un point de tracé peut être défini de deux manières :

- en tant que série de coordonnées de page x et y ;
- en tant qu'objet `path point`, qui définit un point d'ancrage et deux points de direction ou poignées qui indiquent la courbe de segment du tracé.



Pour plus de détails, d'exemples et d'informations sur la création de formes, reportez-vous au chapitre correspondant à votre langage de script.

Niveaux d'interaction de l'utilisateur

Lorsque l'avis de l'utilisateur est requis, une boîte de dialogue apparaît généralement. C'est ce que l'on appelle l'*interaction de l'utilisateur*. Cette fonction très utile survient souvent en cas d'interaction directe avec une application. Cependant, lorsqu'un script interagit avec une application, la boîte de dialogue interrompt l'exécution du script jusqu'à sa fermeture. Cela peut constituer un problème sérieux dans le cas d'un environnement automatisé où personne ne se charge des boîtes de dialogue.

L'objet `application` comporte une propriété `user interaction level` qui vous permet de contrôler le niveau d'interaction autorisé pendant l'exécution du script. Vous pouvez supprimer l'interaction dans un environnement automatisé ou permettre quelques interactions lorsque les scripts sont utilisés d'une manière plus interactive.

AppleScript

Il est possible d'envoyer des commandes d'une machine à une autre à l'aide d'AppleScript afin d'autoriser d'autres interactions. Dans AppleScript, il existe quatre valeurs possibles pour la propriété `user interaction level` :

Valeur de la propriété	Résultat
<code>never interact</code>	Aucune interaction autorisée
<code>interact with self</code>	Interaction uniquement avec des scripts exécutés à partir du menu Scripts (Fichier > Scripts)
<code>interact with local</code>	Interaction uniquement avec des scripts exécutés sur la machine locale (y compris la machine elle-même)
<code>interact with all</code>	Interaction avec tous les scripts

Ces quatre valeurs vous permettent de contrôler l'interaction en fonction de la source des commandes de scripts. Par exemple, si l'application agit en tant que serveur pour les utilisateurs à distance, il serait difficile pour un utilisateur à distance de fermer une boîte de dialogue. En revanche, cela ne présente pas de problème pour quelqu'un qui se trouve devant la machine. Dans ce cas, un niveau d'interaction de type *interact with local* empêcherait les boîtes de dialogue d'interrompre les scripts à distance, mais permettrait de présenter les boîtes de dialogue associées aux scripts locaux.

JavaScript

Dans JavaScript, il existe deux valeurs possibles pour la propriété `app.userInteractionLevel` :

Valeur de la propriété	Résultat
<code>DISPLAYALERTS</code>	Interaction autorisée
<code>DONTDISPLAYALERTS</code>	Aucune interaction autorisée

VBScript

Dans VBScript, il existe deux valeurs possibles pour la propriété `UserInteractionLevel` de l'objet `Application` :

Valeur de la propriété	Résultat
<code>aiDisplayAlerts</code>	Interaction autorisée
<code>aiDontDisplayAlerts</code>	Aucune interaction autorisée

Impression de documents Illustrator

En utilisant la fonction de script `print options`, vous pouvez capturer et automatiser des parties de votre flux d'impression. Le script expose l'ensemble des possibilités d'impression d'Illustrator, dont certaines peuvent ne pas être accessibles via l'interface utilisateur de l'application.

Illustrator prend en charge une session d'impression au maximum à la fois, en raison de limitations de l'architecture d'impression actuelle.

La commande ou la méthode `print` de l'objet `document` prend en compte un seul paramètre en option, qui vous permet d'indiquer l'objet `print options`.

L'objet `print options` vous permet de définir les paramètres d'impression tels que PPD, les options PostScript, les options papier, les options de gestion de couleur, etc. L'objet `print options` comporte également une propriété `print preset` qui vous permet d'indiquer un paramètre prédéfini pour spécifier votre travail d'impression.

Lorsque vous définissez les propriétés d'un objet `print options`, vous pouvez prendre connaissance des imprimantes, PPD, paramètres prédéfinis d'impression et autres éléments disponibles grâce aux propriétés de la « liste » en lecture seule de l'objet `application`, comme les propriétés `printer list`, `PPD file list` et `print presets list`.

4 Création de scripts à l'aide d'AppleScript

Ce chapitre utilise des exemples de script, accompagnés d'explications pour vous aider à vous familiariser avec la création de scripts Illustrator à l'aide d'AppleScript.

Pour plus d'informations

Plusieurs exemples de scripts étendus sont disponibles dans le dossier `:Scripts:Exemples de script` du répertoire d'installation d'Illustrator CS4.

Pour plus de détails sur les classes, objets, propriétés, commandes et paramètres individuels, ainsi que pour obtenir des exemples de scripts illustrant leur application, reportez-vous au guide *Adobe Illustrator CS4 Scripting Reference: AppleScript* (Référence de script Adobe Illustrator CS4 : AppleScript), disponible dans le dossier `:Scripts:Documentation` du répertoire d'installation d'Illustrator CS4. Vous pouvez également consulter le dictionnaire d'Illustrator CS4 depuis l'application Editeur de scripts (voir la section [« Affichage du modèle d'objet AppleScript », page 9](#)).

Si vous ne comprenez pas les concepts et les termes utilisés dans ce chapitre, reportez-vous au document *Introduction aux scripts Adobe*.

Votre premier script Illustrator

Le premier projet classique abordé lors de l'apprentissage d'un nouveau langage de programmation consiste à afficher le message « Hello World! ». Dans cet exemple, vous allez créer un document Illustrator, puis ajouter un bloc de texte contenant ce message. Procédez comme suit :

1. Ouvrez l'application Editeur de scripts.

Dans une installation Mac OS par défaut, l'application Editeur de scripts se trouve dans `Applications:AppleScript:Script Editor`. Si vous ne trouvez pas l'application Editeur de scripts, vous devez la réinstaller à partir du CD du système Mac OS.

2. Entrez le script ci-dessous.

```
--Send the following commands to Illustrator
tell application "Adobe Illustrator"
--Create a new document
set docRef to make new document
--Create a new text frame with the string "Hello World"
set textRef to make new text frame in docRef
    with properties {contents: "Hello World!", position:{200, 200}}
end tell
```

3. Dans la barre d'outils de l'application Editeur de scripts, cliquez sur la commande Exécuter.

CONSEIL : pour ajouter le script au menu des scripts Illustrator (Fichier > Scripts), vous devez l'enregistrer dans le dossier Scripts. Le script s'affichera dans le menu au prochain démarrage d'Illustrator. Pour plus de détails, reportez-vous à la section [« Installation de scripts dans le menu Scripts », page 10](#).

Ajout de caractéristiques à « Hello World »

Nous allons maintenant créer un script qui modifiera le document Illustrator créé avec votre premier script. Notre deuxième script indique comment :

- obtenir le document actif ;
- obtenir la largeur du document actif ;
- redimensionner le bloc de texte pour l'adapter à la largeur du document.

Si vous avez fermé le document Illustrator, exécutez votre premier script une nouvelle fois pour créer un document.

Procédez comme suit :

1. Dans l'application Editeur de scripts, choisissez la commande Fichier > Nouveau pour créer un script.
2. Entrez le code suivant :

```
tell application "Adobe Illustrator"
-- current document is always the active document
  set docRef to the current document
  set docWidth to the width of docRef
-- resize the text frame to match the page width
  set width of text frame 1 of docRef to docWidth
-- alternatively, one can reference the item directly, as follows:
  set width of text frame 1 of current document to docWidth
end tell
```

3. Exécutez le script.

Référencement d'objets

Dans AppleScript, Illustrator renvoie des références à l'objet par index ou par nom. Voici, par exemple, une référence au premier tracé sur le calque 2 :

```
path item 1 of layer 2 of document 1
```

La position d'index de l'objet peut changer à mesure que vous créez ou supprimez d'autres objets. Par exemple, lorsque vous créez un élément de tracé sur `layer 2`, ce nouvel élément de tracé devient `path item 1 of layer 2 of document 1`. Le nouvel objet déplace l'élément de tracé d'origine en le forçant à la position 2. Ainsi, toute référence à `path item 1 of layer 2 of document 1` s'applique au nouvel objet. Cette méthode d'application de numéros d'index assure que le numéro le plus bas fait référence au dernier objet sur lequel vous avez travaillé.

Observez le script d'exemple suivant :

```
-- Make 2 new objects and try to select both
tell application "Adobe Illustrator"
  set newDocument to make new document
  set rectPath to make new rectangle in newDocument
  set starPath to make new star in newDocument
  set selection of newDocument to {rectPath, starPath}
end tell
```

Ce script ne sélectionne pas le rectangle et l'étoile, contrairement à ce qui était prévu. En fait, il ne sélectionne que l'étoile. Essayez d'exécuter le script avec la fenêtre Historique des événements ouverte, et observez les références renvoyées par Illustrator pour chacune des commandes `make` consécutives. (Choisissez l'option Hist. des événements au bas de la fenêtre de l'application Editeur de scripts.) Les deux commandes renvoient la même référence d'objet : `path item 1 of layer 1 of document 1`. Par conséquent, la dernière ligne se résout comme suit :

```
set selection of document 1 to {path item 1 of layer 1 of document 1,
    path item 1 of layer 1 of document 1}
```

Une meilleure approche consiste à référencer les objets par leur nom :

```
tell application "Adobe Illustrator"
    set newDocument to make new document
    make new rectangle in newDocument with properties {name:"rectangle"}
    make new star in newDocument with properties {name:"star"}
    set selection of newDocument to
        {path item "rectangle" of newDocument,
         path item "star" of newDocument}
end tell
```

Cet exemple illustre la nécessité d'identifier de manière unique les objets dans les scripts AppleScript. Nous vous recommandons d'affecter des noms ou des variables aux objets auxquels vous devrez accéder ultérieurement, car rien ne garantit que vous accédez aux bons objets lorsque vous utilisez la méthode d'accès par index.

Obtention d'objets de documents et calques

Ce script fait référence à un objet en tant que partie d'un document :

```
-- Get reference for first page item of document 1
tell application "Adobe Illustrator"
    set pageItemRef to page item 1 of document 1
end tell
```

Dans le script suivant, la variable `pageItemRef` ne fait pas nécessairement référence au même objet que dans le script précédent, car ce script inclut une référence à un calque :

```
-- Get reference for first page item of layer 1 of document 1
tell application "Adobe Illustrator"
    set pageItemRef to page item 1 of layer 1 of document 1
end tell
```

Création d'objets

Pour créer un objet dans AppleScript, vous devez utiliser la commande `make`.

Utilisation de sélections

Lorsqu'un utilisateur effectue une sélection dans un document, les objets sélectionnés sont stockés dans la propriété `selection` du document. Pour accéder à tous les objets sélectionnés dans le document actif :

```
tell application "Adobe Illustrator"
    set myDoc to current document
    set selectedObjects to selection of myDoc
end tell
```

En fonction des éléments sélectionnés, la valeur de la propriété `selection` peut être un tableau de données de tout type d'objets d'illustration. Pour obtenir ou manipuler les propriétés des éléments d'illustration sélectionnés, vous devez récupérer les éléments individuels dans le tableau de données. Pour rechercher un type d'objet, utilisez la propriété `class`.

L'exemple suivant permet d'obtenir le premier objet dans le tableau de données, puis d'afficher le type d'objet :

```
tell application "Adobe Illustrator"
    set myDoc to current document
    set selectedObjects to selection of myDoc
    set topObject to item 1 of selectedObjects
    display dialog (class of topObject)
end tell
```

Le premier objet d'un tableau de données de sélection correspond à l'objet sélectionné *ajouté* en dernier à la page, et non au dernier objet sélectionné.

Sélection d'objets d'illustration

Pour sélectionner un objet d'illustration, utilisez la propriété `selected` de l'objet.

Utilisation de blocs de texte

Pour créer un bloc de texte d'un type donné dans AppleScript, utilisez la propriété `kind` de l'objet `text frame`.

```
set myRect to make new rectangle in current document with properties
{position:{100, 700}, height:100, width:100}
set myAreaText to make new text frame in current document with properties
{kind:point text, contents:"Text Frame 1"}
```

Blocs liés

Tout comme dans l'application Illustrator, vous pouvez lier des blocs de texte captifs ou curvilignes.

Pour lier des blocs de texte existants, utilisez la propriété `next frame` ou `previous frame` de l'objet `text frame`.

Lors de la copie du script suivant dans votre éditeur de scripts, placez la valeur de la propriété `contents` sur une seule ligne. Le caractère de ligne longue (-) n'est pas valide dans une valeur de chaîne.

```
tell application "Adobe Illustrator"
    make new document
    make new rectangle in current document with properties
        {position:{100, 500}, height:100, width:100}
    make new text frame in current document with properties
        {kind:area text, text path:the result, name:"tf1",
        contents:"This is two text frames linked together as one story, with
        text flowing from the first to the last. First frame content. "}
    make new rectangle in current document with properties
        {position:{300, 700}, height:100, width:100}
    make new text frame in current document with properties
        {kind:area text, text path:the result, name:"tf2",
        contents:"Second frame content." }
    --use the next frame property to thread the frames
    set next frame of text frame "tf1" of current document to
        text frame "tf2" of current document
    redraw
end tell
```

Création d'un objet d'article unique à l'aide de blocs liés

Les blocs liés créent un seul objet `story`. Pour observer cette opération, exécutez le script AppleScript suivant après l'exécution du script à la section « [Blocs liés](#) », page 38.

```
display dialog ("There are " & (count(text frames of current document)) & " text frames.")
display dialog ("There are " & (count(stories of current document)) & " stories.")
```

Création de tracés et de formes

Cette section explique comment créer des éléments comportant des tracés.

Tracés

Pour créer un tracé linéaire ou libre, vous devez spécifier une série de points de tracé sous la forme d'une série de coordonnées `x, y` ou sous la forme d'objets `path point`.

Les coordonnées `x, y` limitent le tracé à des segments droits. Pour créer un tracé incurvé, vous devez créer des objets `path point`. Un tracé consiste en une combinaison de coordonnées de page et d'objets `path point`.

Spécification d'une série de coordonnées `x, y`

Pour spécifier un tracé à l'aide de paires de coordonnées de page, vous devez utiliser la méthode `entire path` de l'objet `path items`. Le script suivant spécifie trois paires de coordonnées `x, y` permettant de créer un tracé comportant trois points :

```
tell application "Adobe Illustrator"
set docRef to make new document
-- set stroked to true so we can see the path
set lineRef to make new path item in docRef with properties {stroked:true}
set entire path of lineRef to {{220, 475},{200, 300},{375, 300}}
end tell
```

Utilisation d'objets `PathPoint`

Pour créer un objet `path point`, vous devez définir trois valeurs relatives au point :

- un point `anchor` fixe, représentant le point sur le tracé ;
- une paire de points directeurs, `left direction` et `right direction`, qui vous permettent de contrôler la courbure du segment de tracé.

Vous définissez chaque propriété en tant que tableau de données de coordonnées de page au format `[x, y]` :

- Si les trois propriétés d'un objet `path point` présentent des coordonnées identiques et que les propriétés de l'objet `path point` suivant sur la ligne sont égales entre elles, vous obtenez un segment de ligne droite.
- Si plusieurs propriétés d'un objet `path point` présentent des valeurs différentes, le segment relié au point est incurvé.

Pour créer un tracé ou ajouter des points à un tracé existant, vous devez créer un objet `path item` à l'aide d'objets `path point`, puis ajouter les points de tracé en tant qu'objets enfants dans l'élément `path item` :

```
tell application "Adobe Illustrator"
set docRef to make new document
-- set stroked to true so we can see the path
set lineRef to make new path item in docRef with properties {stroked:true}
--giving the direction points the same value as the
--anchor point creates a straight line segment
set newPoint to make new path point of lineRef with properties
  {anchor:{220, 475},left direction:{220, 475},right direction:{220, 475},
  point type:corner}

set newPoint2 to make new path point of lineRef with properties
  {anchor:{375, 300},left direction:{375, 300},right direction:{375, 300},
  point type:corner}

--giving the direction points the different values
--creates a curve
set newPoint3 to make new path point of lineRef with properties
  {anchor:{220, 300},left direction:{180, 260},right direction:{240, 320},
  point type:corner}

end tell
```

Combinaison de types de point de tracé

Le script d'exemple suivant permet de créer un tracé avec trois points en combinant la propriété du tracé entier à un objet `path point` :

```
tell application "Adobe Illustrator"
set docRef to make new document
-- set stroked to true so we can see the path
set lineRef to make new path item in docRef with properties {stroked:true}
set entire path of lineRef to {{220, 475},{375, 300}}
set newPoint to make new path point of lineRef with properties
  {anchor:{220, 300},left direction:{180, 260},right direction:{240, 320},
  point type:corner}
end tell
```

Formes

Pour créer une forme, vous devez utiliser l'objet correspondant au nom de la forme (par exemple, `ellipse`, `rectangle` ou `polygon`), puis utiliser les propriétés de l'objet pour indiquer la position de la forme, la taille et d'autres informations, telles que le nombre de côtés dans un polygone.

Rappel :

- Toutes les mesures, ainsi que les coordonnées de page sont traitées par le moteur de création de scripts. Pour plus de détails, reportez-vous à la section [« Unités de mesure », page 30](#).
- Les coordonnées x et y sont mesurées à partir de l'angle inférieur gauche du document, comme indiqué dans le panneau Informations de l'application Illustrator. Pour plus de détails, reportez-vous à la section [« Positionnement et dimensions d'un élément de page », page 31](#).

Accès non réinscriptible

Les propriétés des formes d'éléments de tracés utilisent l'état d'accès non réinscriptible, ce qui indique que vous ne pouvez écrire la propriété qu'au moment de la création de l'objet. Pour les objets d'élément de tracé, les propriétés, non modifiables, sont en lecture seule.

Création d'un rectangle

Observez l'exemple suivant :

```
tell application "Adobe Illustrator"
set docRef to make new document
set rectRef to make new rectangle in docRef with properties
    {bounds:{288, 360, 72, 144}}
end tell
```

L'exemple permet de créer un rectangle avec les propriétés suivantes :

- L'angle supérieur droit du rectangle est placé à 4 pouces (288 points) du bas de la page et à 5 pouces (360 points) du bord gauche de la page.
- L'angle inférieur gauche du rectangle est placé à 1 pouce (72 points) du bord gauche de la page et à 2 pouces (144 points) du bas de la page.

Création d'un polygone

Observez l'exemple suivant :

```
tell application "Adobe Illustrator"
set docRef to make new document
set pathRef to make new polygon in docRef with properties
    {center point:{144, 288}, sides:7, radius:72, 0}
end tell
```

L'exemple permet de créer un polygone avec les propriétés suivantes :

- Le point central de l'objet est placé à 2 pouces (144 points) sur l'axe horizontal et à 4 pouces (288 points) sur l'axe vertical.
- Le polygone est composé de 7 côtés.
- La longueur du rayon du centre à chaque coin s'élève à 1 pouce (72 points).

5 Création de scripts à l'aide de JavaScript

Ce chapitre utilise des exemples de script, accompagnés d'explications pour vous aider à vous familiariser avec la création de scripts Illustrator à l'aide de JavaScript.

Pour plus d'informations

Plusieurs exemples de scripts étendus sont disponibles dans le dossier `/Scripts/Exemples de script` du répertoire d'installation d'Illustrator CS4.

Pour plus de détails sur les classes, objets, propriétés, méthodes et paramètres individuels, ainsi que pour obtenir des exemples de scripts illustrant leur application, reportez-vous au guide *Adobe Illustrator CS4 Scripting Reference: JavaScript* (Référence de script Adobe Illustrator CS4 : JavaScript), disponible dans le dossier `/Scripts/Documentation` du répertoire d'installation d'Illustrator CS4. Vous pouvez également consulter le dictionnaire d'Illustrator depuis l'afficheur de modèle d'objet d'ESTK. Pour plus de détails sur l'afficheur de modèle d'objet d'ExtendScript Toolkit, reportez-vous à la section [« Affichage du modèle d'objet JavaScript », page 8](#) ou consultez le *Guide des outils JavaScript*.

Si vous ne comprenez pas les concepts et les termes utilisés dans ce chapitre, reportez-vous au document *Introduction aux scripts Adobe*.

Votre premier script Illustrator

Le premier projet classique abordé lors de l'apprentissage d'un nouveau langage de programmation consiste à afficher le message « Hello World! ». Dans cet exemple, vous allez créer un document Illustrator, puis ajouter un bloc de texte contenant ce message. Procédez comme suit :

1. Utilisez n'importe quelle application de traitement de texte (comme Adobe InDesign® ou ESTK) pour saisir le texte ci-dessous :

```
//Hello World!  
var myDocument = app.documents.add();  
  
//Create a new text frame and assign it to the variable "myTextFrame"  
var myTextFrame = myDocument.textFrames.add();  
  
// Set the contents and position of the text frame  
myTextFrame.position = [200 200];  
myTextFrame.contents = "Hello World!"
```

Pour plus de détails sur l'emplacement d'ExtendScript Toolkit, reportez-vous à la section [« Affichage du modèle d'objet JavaScript », page 8](#).

2. Pour tester le script, utilisez l'une des méthodes suivantes :
 - Si vous utilisez ESTK, vous devez sélectionner Adobe Illustrator CS4 dans la liste déroulante située dans l'angle supérieur gauche, cliquer sur le bouton Oui pour lancer Illustrator, puis choisir la commande Déboguer > Exécuter dans ESTK afin d'exécuter le script.

- Si vous utilisez un autre éditeur de texte, enregistrez le fichier en texte seul dans un dossier de votre choix avec l'extension `.jsx`, puis lancez Illustrator. Dans Illustrator, choisissez la commande Fichier > Scripts > Autre script, puis recherchez et exécutez votre fichier de script.

CONSEIL : pour ajouter le script au menu des scripts Illustrator (Fichier > Scripts), vous devez l'enregistrer dans le dossier Scripts. Le script s'affichera dans le menu au prochain démarrage d'Illustrator. Pour plus de détails, reportez-vous à la section « [Installation de scripts dans le menu Scripts](#) », page 10.

Ajout de caractéristiques à « Hello World »

Nous allons maintenant créer un script qui modifiera le document Illustrator créé avec votre premier script. Notre deuxième script indique comment :

- obtenir le document actif ;
- obtenir la largeur du document actif ;
- redimensionner le bloc de texte pour l'adapter à la largeur du document.

Si vous avez fermé le document Illustrator, réexécutez votre premier script pour créer un document avant de continuer cet exercice.

Procédez comme suit :

1. Choisissez la commande Fichier > Nouveau dans l'éditeur de texte pour créer un script.
2. Entrez le code suivant :

```
var docRef = app.activeDocument;
var docWidth = docRef.width

var frameRef = docRef.textFrames[0]

frameRef.width = docWidth
```

3. Exécutez le script.

Utilisation de méthodes dans JavaScript

Lorsque les méthodes que vous utilisez présentent plusieurs paramètres, vous pouvez ignorer les paramètres facultatifs en fin de liste, mais pas ceux qui apparaissent au milieu. Si vous ne souhaitez pas spécifier un paramètre apparaissant en milieu de liste, vous devez insérer la valeur `undefined` pour appliquer la valeur par défaut du paramètre. Par exemple, la définition suivante décrit la méthode `rotate()` pour un objet d'illustration.

```
rotate
(angle
[, changePositions]
[, changeFillPatterns]
[, changeFillGradients]
[, changeStrokePattern]
[, rotateAbout])
```

Dans la définition, extraite du document *Adobe Illustrator CS4 Scripting Reference: JavaScript* (Référence de script Adobe Illustrator CS4 : JavaScript), les paramètres facultatifs apparaissent entourés de crochets (`[]`).

Pour appliquer une rotation de 30 degrés à l'objet et modifier le paramètre `fillGradients`, vous devez utiliser l'instruction suivante :

```
myObject.rotate(30, undefined, undefined, true);
```

Vous ne devez spécifier la valeur `undefined` que pour les paramètres `changePositions` et `changeFillPatterns`. Vous ne devez rien spécifier pour les deux paramètres facultatifs qui suivent le paramètre `changeFillGradients`, puisqu'ils sont à la fin de la liste de paramètres.

Accès et référencement d'objets

Lors de l'écriture d'un script, vous devez d'abord indiquer le fichier ou `document` sur lequel le script doit agir. A l'aide de l'objet `application`, le script peut créer un document, ouvrir un document existant ou agir sur un document déjà ouvert.

Le script peut créer des objets dans le document, effectuer des opérations sur des objets sélectionnés par l'utilisateur ou sur les objets d'une collection. Les sections suivantes illustrent plusieurs techniques d'accès, de référencement et de manipulation d'objets Illustrator.

Référencement de l'objet application

Pour obtenir une référence à un objet spécifique, vous devez explorer la hiérarchie du contenu. Tous les scripts JavaScript s'exécutent de l'intérieur de l'application Illustrator ; toutefois, aucune référence spécifique à l'objet `application` n'est nécessaire. Par exemple, pour affecter le document actif dans Illustrator à la variable `frontMostDocument`, vous pouvez référencer la propriété `activeDocument` de l'objet `application` comme suit :

```
var frontMostDocument = activeDocument;
```

Il est possible d'utiliser l'objet `application` dans une référence. Pour cela, vous devez utiliser la variable `app` globale. Les deux instructions qui suivent semblent identiques au moteur JavaScript :

```
var frontMostDocument = activeDocument;
```

```
var frontMostDocument = app.activeDocument;
```

Accès aux objets de collection

Tous les documents ouverts, ainsi que les objets qu'ils contiennent, sont rassemblés en objets de collection pour le type d'objet. Un objet de collection contient un tableau de données des objets auxquels vous pouvez accéder par index ou par nom. L'objet de collection prend la forme plurielle du nom de l'objet. Par exemple, l'objet de collection de l'objet `document` devient `documents`.

Le script d'exemple suivant permet d'obtenir tous les objets `graphic style` de la collection `graphic styles`, c'est-à-dire tous les styles graphiques disponibles pour le document actif :

```
var myStyles = app.activeDocument.graphicStyles;
```

Toutes les références numériques dans une collection commencent par zéro en JavaScript, c'est-à-dire que le premier objet dans la collection prend l'index [0].

En JavaScript, les numéros d'index ne changent pas lors de l'ajout d'un objet à une collection. Il existe toutefois une exception : l'objet `documents[0]` correspond toujours au document actif ou de premier plan.

Pour accéder au premier style dans une collection `graphic styles`, vous pouvez utiliser la variable déclarée dans le script d'exemple précédent ou bien utiliser la hiérarchie du contenu pour faire référence à la collection :

- Utilisation de la variable `myStyles` :

```
var firstStyle = myStyles[0];
```

- Utilisation de la hiérarchie du contenu :

```
var firstStyle = app.activeDocument.graphicStyles[0];
```

Les instructions suivantes affectent à une variable le nom du premier style graphique de la collection. Ces instructions sont interchangeable.

```
var styleName = myStyles[0].name
```

```
var styleName = firstStyle.name
```

```
var styleName = app.activeDocument.graphicStyles[0].name
```

Pour obtenir le nombre total d'objets dans une collection, utilisez la propriété `length` :

```
alert ( myStyles.length );
```

L'index du dernier style graphique dans la collection est `myStyles.length-1` (-1, car la collection commence à compter à partir de 0, et la propriété `length` à partir de 1) :

```
var lastStyle = myStyles[ myStyles.length - 1 ];
```

Une expression représentant la valeur de l'index apparaît entre crochets (`[]`) et entre guillemets.

Si vous connaissez le nom d'un objet, vous pouvez y accéder dans les collections en utilisant le nom encadré de crochets, par exemple :

```
var getStyle = myStyles[Ice Type];
```

Chaque élément de la collection est un objet du type recherché et vous pouvez accéder à ses propriétés dans la collection. Par exemple, pour rechercher un nom d'objet, utilisez la propriété `name` :

```
var styleName = app.activeDocument.graphicStyles[0].name;
```

Pour appliquer le style `lastStyle` au premier élément `pageItem` du document, utilisez sa méthode `applyTo()` :

```
lastStyle.applyTo( app.activeDocument.pageItems[0] );
```

Création d'objets

Pour créer des objets, vous pouvez utiliser un script. Pour créer des objets disponibles à partir d'objets de collection ou de *conteneurs*, vous devez utiliser la méthode `add()` de l'objet de conteneur :

```
var myDoc = app.documents.add()
var myLayer = myDoc.layers.add()
```

Certains types d'objet ne sont pas disponibles dans les conteneurs. Vous pouvez créer un objet de ce type en définissant une variable, puis en appliquant l'opérateur `new` avec un constructeur d'objet pour affecter un objet en tant que valeur. Par exemple, pour créer un objet `CMYKColor` à l'aide du nom de variable `myColor` :

```
var myColor = new CMYKColor()
```

Utilisation de sélections

Lorsqu'un utilisateur effectue une sélection dans un document, les objets sélectionnés sont stockés dans la propriété `selection` du document. Pour accéder à tous les objets sélectionnés dans le document actif :

```
var selectedObjects = app.activeDocument.selection;
```

En fonction des types d'éléments sélectionnés, la valeur de la propriété `selection` peut être un tableau de données de tout type d'objets d'illustration. Pour obtenir ou manipuler les propriétés des éléments d'illustration sélectionnés, vous devez récupérer les éléments individuels dans le tableau de données. Pour rechercher un type d'objet, utilisez la propriété `typename`.

L'exemple suivant permet d'obtenir le premier objet dans le tableau de données, puis d'afficher le type d'objet :

```
var topObject = app.activeDocument.selection[0];
alert(topObject.typename)
```

Le premier objet d'un tableau de données de sélection correspond à l'objet sélectionné *ajouté* en dernier à la page, et non au dernier objet sélectionné.

Sélection d'objets d'illustration

Pour sélectionner un objet d'illustration, utilisez la propriété `selected` de l'objet.

Utilisation de blocs de texte

Pour créer un bloc de texte d'un type donné en JavaScript, vous devez utiliser la méthode `textFrames` dont le nom correspond au type du bloc de texte, par exemple :

```
var rectRef = docRef.pathItems.rectangle(700, 50, 100, 100);
//use the areaText method to create the text frame
var areaTextRef = docRef.textFrames.areaText(rectRef);
```

Blocs liés

Tout comme dans l'application Illustrator, vous pouvez lier des blocs de texte captifs ou curvilignes.

Pour lier des blocs de texte existants, utilisez la propriété `nextFrame` ou `previousFrame` de l'objet `text frame`.

Lors de la copie du script suivant dans ESTK, placez la valeur de la propriété `contents` sur une seule ligne.

```
var myDoc = documents.add();
var myPathItem1 = myDoc.pathItems.rectangle(244, 64, 82, 76);
var myTextFrame1 = myDoc.textFrames.areaText(myPathItem1);
var myPathItem2 = myDoc.pathItems.rectangle(144, 144, 42, 116);
var myTextFrame2 = myDoc.textFrames.areaText(myPathItem2);

// use the nextFrame property to thread the text frames
myTextFrame1.nextFrame = myTextFrame2;
var sText = "This is two text frames linked together as one story, with text
flowing from the first to the last. This is two text frames linked together as one
story, with text flowing from the first to the last. This is two text frames linked
together as one story. ";
myTextFrame1.contents = sText;
redraw();
```

Création d'objet d'article unique à l'aide de blocs liés

Les blocs liés créent un seul objet `story`. Pour observer cette opération, exécutez le script JavaScript suivant après l'exécution du script à la section [« Blocs liés », page 46](#).

```
var myDoc = app.activeDocument
alert("There are " + myDoc.textFrames.length + " text frames.")
alert("There are " + myDoc.stories.length + " stories.")
```

Création de tracés et de formes

Cette section explique comment créer des éléments comportant des tracés.

Tracés

Pour créer un tracé libre, vous devez spécifier une série de points de tracé sous la forme d'une série de coordonnées `x, y` ou sous la forme d'objets `pathPoint`.

Les coordonnées `x, y` limitent le tracé à des segments droits. Pour créer un tracé incurvé, vous devez créer des objets `pathPoint`. Votre tracé consiste en une combinaison de coordonnées de page et d'objets `pathPoint`.

Spécification d'une série de coordonnées `x, y`

Pour spécifier un tracé à l'aide de paires de coordonnées de page, vous devez utiliser la méthode `setEntirePath()` de l'objet `pathItems`. Le script suivant spécifie trois paires de coordonnées `x, y` permettant de créer un tracé comportant trois points :

```
var myDoc = app.activeDocument;
var myLine = myDoc.pathItems.add();
//set stroked to true so we can see the path
myLine.stroked = true;
myLine.setEntirePath([[220, 475], [375, 300], [200, 300]]);
```

Utilisation d'objets `pathPoint`

Pour créer un objet `pathPoint`, vous devez définir trois valeurs relatives au point :

- un point `anchor` fixe, représentant le point sur le tracé ;
- une paire de points directeurs, `left direction` et `right direction`, qui vous permettent de contrôler la courbure du segment de tracé.

Vous définissez chaque propriété en tant que tableau de données de coordonnées de page au format `[x, y]`.

- Si les trois propriétés d'un objet `pathPoint` présentent des coordonnées identiques et que les propriétés de l'objet `pathPoint` suivant sur la ligne sont égales entre elles, vous obtenez un segment de ligne droite.
- Si plusieurs propriétés d'un objet `pathPoint` présentent des valeurs différentes, le segment relié au point est incurvé.

Pour créer un tracé ou ajouter des points à un tracé existant, vous devez créer un objet `pathItem` à l'aide d'objets `pathPoint`, puis ajouter les points de tracé en tant qu'objets enfants dans l'élément `pathItem` :

```
var myDoc = app.activeDocument;
var myLine = myDoc.pathItems.add();
//set stroked to true so we can see the path
myLine.stroked = true;

var newPoint = myLine.pathPoints.add();
newPoint.anchor = [220, 475];
//giving the direction points the same value as the
//anchor point creates a straight line segment
newPoint.leftDirection = newPoint.anchor;
newPoint.rightDirection = newPoint.anchor;
newPoint.pointType = PointType.CORNER;

var newPoint1 = myLine.pathPoints.add();
newPoint1.anchor = [375, 300];
newPoint1.leftDirection = newPoint1.anchor;
newPoint1.rightDirection = newPoint1.anchor;
newPoint1.pointType = PointType.CORNER;

var newPoint2 = myLine.pathPoints.add();
newPoint2.anchor = [220, 300];
//giving the direction points different values
//than the anchor point creates a curve
newPoint2.leftDirection = [180, 260];
newPoint2.rightDirection = [240, 320];
newPoint2.pointType = PointType.CORNER;
```

Combinaison de types de point de tracé

L'exemple suivant permet de créer un tracé avec trois points :

```
var myDoc = app.activeDocument;
var myLine = myDoc.pathItems.add();
myLine.stroked = true;
myLine.setEntirePath( [[220, 475], [375, 300]]);

// Append another point to the line
var newPoint = myDoc.myLine.pathPoints.add();
newPoint.anchor = [220, 300];
newPoint.leftDirection = newPoint.anchor;
newPoint.rightDirection = newPoint.anchor;
newPoint.pointType = PointType.CORNER;
```

Formes

Pour créer une forme, vous devez utiliser la méthode `pathItems` correspondant au nom de la forme (par exemple, `ellipse`, `rectangle` ou `polygon`), puis utiliser les paramètres pour indiquer la position de la forme, la taille et d'autres informations, telles que le nombre de côtés dans un polygone.

Rappel :

- Toutes les mesures, ainsi que les coordonnées de page sont traitées comme des points par le moteur de création de scripts. Pour plus de détails, reportez-vous à la section « [Unités de mesure](#) », page 30.

- Les coordonnées x et y sont mesurées à partir de l'angle inférieur gauche du document, comme indiqué dans le panneau Informations de l'application Illustrator. Pour plus de détails, reportez-vous à la section « [Positionnement et dimensions d'un élément de page](#) », page 31.

Création d'un rectangle

Observez l'exemple suivant :

```
var myDocument = app.documents.add()
var artLayer = myDocument.layers.add()
var rect = artLayer.pathItems.rectangle( 144, 144, 72, 216 );
```

L'exemple utilise la méthode `rectangle()` de l'objet `pathItems` pour créer un rectangle avec les propriétés suivantes :

- L'angle supérieur du rectangle est à 2 pouces (144 points) du bord inférieur de la page.
- Le bord gauche est à 2 pouces (144 points) du bord gauche de la page.
- Le rectangle mesure 1 pouce (72 points) de large sur 3 pouces (216 points) de long.

Création d'un polygone

Observez l'exemple suivant :

```
var myDocument = app.documents.add()
var artLayer = myDocument.layers.add()
var poly = artLayer.pathItems.polygon( 144, 288, 72.0, 7 );
```

L'exemple utilise la méthode `polygon()` pour créer un polygone avec les propriétés suivantes :

- Le point central de l'objet est placé à 2 pouces (144 points) sur l'axe horizontal et à 4 pouces (288 points) sur l'axe vertical.
- La longueur du rayon du centre à chaque coin s'élève à 1 pouce (72 points).
- Le polygone est composé de 7 côtés.

6 Création de scripts à l'aide de VBScript

Ce chapitre utilise des exemples de script, accompagnés d'explications pour vous aider à vous familiariser avec la création de scripts Illustrator à l'aide de VBScript.

Pour plus d'informations

Plusieurs exemples de scripts étendus sont disponibles dans le dossier `/Scripts/Exemples de script` du répertoire d'installation d'Illustrator CS4.

Pour plus de détails sur les classes, objets, propriétés, méthodes et paramètres individuels, ainsi que pour obtenir des exemples de scripts illustrant leur application, reportez-vous au guide *Adobe Illustrator CS Scripting Reference: VBScript* (Référence de script Adobe Illustrator CS : VBScript), disponible dans le dossier `/Scripts/Documentation` du répertoire d'installation d'Illustrator CS4. Vous pouvez également consulter la bibliothèque type d'Illustrator CS4 depuis la plupart des éditeurs VBScript ou de n'importe quelle application Microsoft Office, reportez-vous à la section « [Affichage du modèle d'objet VBScript](#) », page 9.

Si vous ne comprenez pas les concepts et les termes utilisés dans ce chapitre, reportez-vous au document *Introduction aux scripts Adobe*.

Votre premier script Illustrator

Le premier projet classique abordé lors de l'apprentissage d'un nouveau langage de programmation consiste à afficher le message « Hello World! ». Procédez comme suit :

1. Lancez une application de traitement de texte (par exemple, Bloc-notes).
2. Entrez le code suivant :

```
Rem Hello World
Set appRef = CreateObject("Illustrator.Application")

Rem Create a new document and assign it to a variable
Set documentRef = appRef.Documents.Add

Rem Create a new text frame item and assign it to a variable
Set sampleText = documentRef.TextFrames.Add

Rem Set the contents and position of the TextFrame
sampleText.Position = Array(200, 200)
sampleText.Contents = "Hello World!"
```

3. Enregistrez le fichier en texte seul dans un dossier de votre choix avec l'extension `.vbs`.
4. Pour tester le script, utilisez l'une des méthodes suivantes :
 - > Cliquez deux fois sur le fichier.
 - > Lancez l'application Illustrator, choisissez la commande Fichier > Scripts > Autre script, puis recherchez et exécutez votre fichier de script.

CONSEIL : pour ajouter le script au menu des scripts Illustrator (Fichier > Scripts), vous devez l'enregistrer dans le dossier Scripts. Le script s'affichera dans le menu au prochain démarrage d'Illustrator. Pour plus de détails, reportez-vous à la section « [Installation de scripts dans le menu Scripts](#) », page 10. En général, lorsque vous lancez un script VBScript à partir du menu Scripts, les boîtes de dialogue `msgBox` ne s'affichent pas correctement.

Ajout de caractéristiques à « Hello World »

Nous allons maintenant créer un script qui modifiera le document Illustrator créé avec votre premier script. Pour ce deuxième script, vous allez apprendre à :

- obtenir le document actif ;
- obtenir la largeur du document actif ;
- redimensionner le bloc de texte pour l'adapter à la largeur du document.

Si vous n'avez pas enregistré le document Illustrator avant de le fermer, exécutez votre premier script une nouvelle fois pour créer un document.

Procédez comme suit :

1. Copiez le script suivant dans votre éditeur de texte, puis enregistrez le fichier.

```
Set appRef = CreateObject("Illustrator.Application")
'Get the active document

Set documentRef = appRef.ActiveDocument

Set sampleText = documentRef.TextFrames(1)

' Resize the TextFrame item to match the document width

sampleText.Width = documentRef.Width

sampleText.Left = 0
```

2. Exécutez le script.

Accès et référencement d'objets

Lors de l'écriture d'un script, vous devez d'abord indiquer le fichier ou `Document` sur lequel le script doit agir. A l'aide de l'objet `Application`, le script peut créer un document, ouvrir un document existant ou agir sur un document déjà ouvert.

Le script peut créer des objets dans le document, effectuer des opérations sur des objets sélectionnés par l'utilisateur ou sur les objets d'une collection. Les sections suivantes illustrent des techniques d'accès, de référencement et de manipulation d'objets Illustrator.

Obtention d'objets de collection

En général, pour obtenir une référence à un objet spécifique, vous pouvez explorer la hiérarchie du contenu. Par exemple, pour utiliser la variable `myPath` afin de stocker une référence au premier élément `PathItem` dans le second calque du document actif :

```
Set myPath = appRef.ActiveDocument.Layers(2).PathItems(1)
```

Les scripts suivants montrent comment référencer un objet en tant que partie d'un document :

```
Set documentRef = appRef.ActiveDocument
```

```
Set pageItemRef = documentRef.PageItems(1)
```

Dans le script suivant, la variable `pageItemRef` ne fait pas nécessairement référence au même objet que le script précédent, car ce script inclut une référence à un calque :

```
Set documentRef = appRef.ActiveDocument
```

```
Set pageItemRef = documentRef.Layers(1).PageItems(1)
```

Les index VBScript commencent par 1 dans les collections d'objets. Toutefois, VBScript vous permet de spécifier si les index de tableaux de données commencent par 1 ou par 0. Pour plus de détails sur la spécification du numéro de départ des index de tableaux de données, reportez-vous aux documents ou didacticiels de VBScript.

Création d'objets

Pour créer des objets, vous pouvez utiliser un script. Pour créer des objets disponibles à partir d'objets de collection, vous devez utiliser la méthode `Add` de l'objet de collection.

```
Set myDoc = appRef.Documents.Add()
```

```
Set myLayer = myDoc.Layers.Add()
```

Certains objets de collection ne disposent pas d'une méthode `Add`. Pour créer un objet de ce type, définissez une variable et appliquez la méthode `CreateObject`. Par exemple, le code suivant crée un objet `CMYKColor` à l'aide du nom de variable `newColor` :

```
Set newColor = CreateObject ("Illustrator.CMYKColor")
```

Utilisation de sélections

Lorsqu'un utilisateur effectue une sélection dans un document, les objets sélectionnés sont stockés dans la propriété `selection` du document. Pour accéder à tous les objets sélectionnés dans le document actif :

```
Set appRef = CreateObject ("Illustrator.Application")
```

```
Set documentRef = appRef.ActiveDocument
```

```
selectedObjects = documentRef.Selection
```

En fonction des éléments sélectionnés, la valeur de la propriété `selection` peut être un tableau de données de tout type d'objets d'illustration. Pour obtenir ou manipuler les propriétés des éléments d'illustration sélectionnés, vous devez récupérer les éléments individuels dans le tableau de données. Pour rechercher un type d'objet, utilisez la propriété `typename`.

L'exemple suivant permet d'obtenir le premier objet dans le tableau de données, puis d'afficher le type d'objet :

```
Set appRef = CreateObject ("Illustrator.Application")
```

```
Set documentRef = appRef.ActiveDocument
```

```
selectedObjects = documentRef.Selection
```

```
Set topObject = selectedObjects(0)
```

```
MsgBox(topObject.TypeName)
```

La méthode `MsgBox` n'affiche pas de boîte de dialogue lors de l'exécution du script à partir du menu Scripts d'Illustrator (Fichier > Scripts).

Le premier objet d'un tableau de données de sélection correspond à l'objet sélectionné *ajouté* en dernier à la page, et non au dernier objet sélectionné.

Sélection d'objets d'illustration

Pour sélectionner un objet d'illustration, utilisez la propriété `Selected` de l'objet.

Utilisation de blocs de texte

Pour créer un bloc de texte d'un type donné en VBScript, vous devez utiliser la méthode `TextFrames` correspondant au type de bloc souhaité :

```
Set rectRef = docRef.PathItems.Rectangle(700, 50, 100, 100)

' Use the AreaText method to create the text frame
Set areaTextRef = docRef.TextFrames.AreaText(rectRef)
```

Blocs liés

Tout comme dans l'application Illustrator, vous pouvez lier des blocs de texte captifs ou curvilignes.

Pour lier des blocs de texte existants, utilisez la propriété `NextFrame` ou `PreviousFrame` de l'objet `TextFrames`.

Lors de la copie du script suivant dans votre éditeur de scripts ou de texte, placez la valeur de la propriété `Contents` sur une seule ligne. Le caractère de ligne longue (`\n`) n'est pas valide dans une valeur de chaîne.

```
Set appRef = CreateObject("Illustrator.Application")
Set myDoc = appRef.Documents.Add
Set myPathItem1 = myDoc.PathItems.Rectangle(244, 64, 82, 76)
Set myTextFrame1 = myDoc.TextFrames.AreaText(myPathItem1)
    myTextFrame1.Contents = "This is two text frames linked together as one story, with
text flowing from the first to the last."
Set myPathItem2 = myDoc.PathItems.Rectangle(144, 144, 42, 116)
Set myTextFrame2 = myDoc.TextFrames.AreaText(myPathItem2)

'Use the NextFrame property to thread the frames
myTextFrame1.NextFrame = myTextFrame2

appRef.Redraw()
```

Création d'objet d'article unique à l'aide de blocs liés

Les blocs liés créent un seul objet `story`. Pour observer cette opération, exécutez le script VBScript suivant après l'exécution du script à la section [« Blocs liés », page 53](#).

```
Set myDoc = appRef.ActiveDocument
myMsg = "alert("There are " & CStr(myDoc.TextFrames.Count) & " text frames. ") "
appRef.DoJavaScript myMsg
myMsg = "alert("There are " & CStr(myDoc.Stories.Count) & " stories. ") "
appRef.DoJavaScript myMsg
```

Création de tracés et de formes

Cette section explique comment créer des éléments comportant des tracés.

Tracés

Pour créer un tracé libre, vous devez spécifier une série de points de tracé sous la forme d'une série de coordonnées x, y ou sous la forme d'objets `PathPoint`.

Les coordonnées x, y limitent le tracé à des segments droits. Pour créer un tracé incurvé, vous devez créer des objets `PathPoint`. Votre tracé consiste en une combinaison de coordonnées de page et d'objets `PathPoint`.

Spécification d'une série de coordonnées x, y

Pour spécifier un tracé à l'aide de paires de coordonnées de page, vous devez utiliser la méthode `SetEntirePath()` de l'objet `PathItems`. Le script suivant spécifie trois paires de coordonnées x, y permettant de créer un tracé comportant trois points :

```
Set appRef = CreateObject ("Illustrator.Application")

Set firstPath = appRef.ActiveDocument.PathItems.Add
  firstPath.Stroked = True
firstPath.SetEntirePath(Array(Array(220, 475), Array(375, 300), Array(200, 300)))
```

Utilisation d'objets PathPoint

Pour créer un objet `PathPoint`, vous devez définir trois valeurs relatives au point :

- un point `anchor` fixe, représentant le point sur le tracé ;
- une paire de points directeurs, `left direction` et `right direction`, qui vous permettent de contrôler la courbure du segment de tracé.

Vous définissez chaque propriété en tant que tableau de données de coordonnées de page au format `(Array, (x, y))`.

- Si les trois propriétés d'un objet `PathPoint` présentent des coordonnées identiques et que les propriétés de l'objet `PathPoint` suivant sur la ligne sont égales entre elles, vous obtenez un segment de ligne droite.
- Si plusieurs propriétés d'un objet `PathPoint` présentent des valeurs différentes, le segment relié au point est incurvé.

Pour créer un tracé ou ajouter des points à un tracé existant, vous devez créer un objet `PathItem` à l'aide d'objets `PathPoint`, puis ajouter les points de tracé en tant qu'objets enfants dans l'élément `PathItem` :

```
Set appRef = CreateObject ("Illustrator.Application")

Set firstPath = appRef.ActiveDocument.PathItems.Add
  firstPath.Stroked = true
Set newPoint = firstPath.PathPoints.Add
'Using identical coordinates creates a straight segment
newPoint.Anchor = Array(75, 300)
newPoint.LeftDirection = Array(75, 300)
newPoint.RightDirection = Array(75, 300)
```

```

Set newPoint2 = firstPath.PathPoints.Add
newPoint2.Anchor = Array(175, 250)
newPoint2.LeftDirection = Array(175, 250)
newPoint2.RightDirection = Array(175, 250)

Set newPoint3 = firstPath.PathPoints.Add
'Using different coordinates creates a curve
newPoint3.Anchor = Array(275, 290)
newPoint3.LeftDirection = Array(135, 150)
newPoint3.RightDirection = Array(155, 150)

```

Combinaison de types de point de tracé

L'exemple suivant permet de créer un tracé avec trois points :

```

Set appRef = CreateObject("Illustrator.Application")
Set myDoc = appRef.ActiveDocument
Set myLine = myDoc.PathItems.Add
    myLine.Stroked = True
    myLine.SetEntirePath( Array( Array(320, 475), Array(375, 300)))

' Append another point to the line
Set newPoint = myLine.PathPoints.Add
'Using identical coordinates creates a straight segment
newPoint.Anchor = Array(220, 300)
newPoint.LeftDirection = Array(220, 300)
newPoint.RightDirection = Array(220, 300)

```

Formes

Pour créer une forme, vous devez utiliser la méthode `PathItems` correspondant au nom de la forme (par exemple, `ellipse`, `rectangle` ou `polygon`), puis utiliser les paramètres pour indiquer la position de la forme, la taille et d'autres caractéristiques, telles que le nombre de côtés dans un polygone.

Rappel :

- Toutes les mesures, ainsi que les coordonnées de page sont traitées par le moteur de création de scripts. Pour plus de détails, reportez-vous à la section [« Unités de mesure », page 30](#).
- Les coordonnées x et y sont mesurées à partir de l'angle inférieur gauche du document, comme indiqué dans le panneau Informations de l'application Illustrator. Pour plus de détails, reportez-vous à la section [« Positionnement et dimensions d'un élément de page », page 31](#).

Création d'un rectangle

Observez l'exemple suivant :

```

Set appRef = CreateObject("Illustrator.Application")
Set frontDocument = appRef.ActiveDocument
' Create a new rectangle with
' top = 144, left side = 144, width = 72, height = 144
Set newRectangle = frontDocument.PathItems.Rectangle(144,144,72,144)

```

L'exemple permet de créer un rectangle avec les propriétés suivantes :

- L'angle supérieur du rectangle est à 2 pouces (144 points) du bord inférieur de la page.
- Le bord gauche est à 2 pouces (144 points) du bord gauche de la page.
- Le rectangle mesure 1 pouce (72 points) de large sur 2 pouces (144 points) de long.

Création d'un polygone

Observez l'exemple suivant :

```
Set appRef = CreateObject("Illustrator.Application")
Set frontDocument = appRef.ActiveDocument
' Create a new polygon with
' top = 144, left side = 288, width = 72, height = 144
Set newPolygon = frontDocument.PathItems.Polygon(144,288,72,7)
```

L'exemple permet de créer un polygone avec les propriétés suivantes :

- Le point central de l'objet est placé à 2 pouces (144 points) sur l'axe horizontal et à 4 pouces (288 points) sur l'axe vertical.
- La longueur du rayon du centre à chaque coin s'élève à 1 pouce (72 points).
- Le polygone est composé de 7 côtés.

Utilisation de valeurs d'énumération

Les propriétés utilisant des valeurs d'énumération en VBScript font appel à une valeur sous forme numérique à la place d'une valeur sous forme de texte. Par exemple, la propriété `Orientation` de l'objet `TextFrame` spécifie si le contenu de texte dans un bloc de texte est horizontal ou vertical. La propriété utilise l'énumération `aiTextOrientation`, qui présente deux possibilités de valeur, `aiHorizontal` et `aiVertical` :

Pour rechercher les valeurs numériques des énumérations, utilisez l'une des méthodes suivantes :

- l'explorateur d'objets dans l'environnement de votre éditeur de scripts (voir la section « [Affichage du modèle d'objet VBScript](#) », page 9).
- le guide *Adobe Illustrator CS4 Scripting Reference: VBScript* (Référence de script Adobe Illustrator CS4 : VBScript), qui répertorie les valeurs numériques immédiatement à la suite des valeurs constantes dans le chapitre traitant des énumérations à la fin du manuel. L'exemple suivant est extrait de ce document :

Enumeration type	Values	What it means
<code>aiTextOrientation</code>	<code>aiHorizontal = 0</code> <code>aiVertical = 1</code>	The orientation of text in a text frame

L'exemple suivant spécifie une orientation verticale du texte :

```
Set appRef = CreateObject ("Illustrator.Application")
Set docRef = appRef.Documents.Add
Set textRef = docRef.TextFrames.Add
textRef.Contents = "This is some text content."
textRef.Left = 50
textRef.Top = 700
textRef.Orientation = 1
```

En général, une bonne méthode de travail consiste à placer la valeur de texte dans un commentaire à la suite de la valeur numérique, comme dans l'instruction d'exemple suivante :

```
textRef.Orientation = 1 ' aiVertical
```

Index

A

actions
à propos de, 6
Adobe Illustrator
Plug-in Software Development Kit Function
Reference, 32
AppleScript
conventions d'attribution de nom, 19
dictionnaire, 9
extensions de fichier, 8
application de styles, à propos de, 24
articles, à propos de, 22
attributs locaux, 24
attributs, à propos de, 24
axe horizontal (x), 31
axe vertical (y), 31

B

blocs de texte, 22
boîtes de dialogue
activation, 33
suppression, 33

C

centimètres, conversion, 30
classe de matrice, 25
coordonnées, à propos de, 31
CS2, modifications de la version, 16

D

dimensions, éléments de page, 31
documents
impression, 34
positionnement des éléments de page, 31

E

éléments de page
limites, 31
positionnement, 31
positionnement et dimensions, 31
ensembles de données, utilisation, 25
exécution de scripts, 10

exemples de scripts

création d'objets, 45, 52
création d'un polygone, 41, 49, 56
création d'un rectangle, 41, 49, 55
création d'un tracé, 39, 47, 54
création d'un tracé incurvé, 39, 48, 54
sélections, 52
ExtendScript, extension de fichier, 8
extensions de fichiers pour scripts valides, 7

F

fermeture d'Illustrator, 26

H

hauteur, valeur maximale autorisée, 31
« Hello World », script
amélioration, 36, 43, 51
création, 35, 42, 50

I

Illustrator
fermeture, 26
lancement, 26
spécification d'une version, 26
Illustrator. *Voir* Adobe Illustrator
impression
à propos de, 22
définition des options, 34
installation de scripts, 10

J

JavaScript
afficheur de modèle d'objet, 8
conventions d'attribution de nom, 19
extension de fichier, 8
modifications d'Illustrator CS2, 16
.jsx, extension, 8

K

kit de développement logiciel, 32

L

lancement d'Illustrator, 26
 largeur, valeur maximale autorisée, 31
 lignes, création, 23
 limites de contrôles, 31
 limites géométriques, 31
 limites visibles, 31

M

matrices de transformation, à propos de, 25
 matrices, à propos de, 25
 méthodes, utilisation, 43
 millimètres, conversion, 30
 modèle d'objet
 diagramme, 18
 texte, 22
 modèle d'objets
 modifications d'Illustrator CS2, 16

N

niveaux d'interaction de l'utilisateur, 33
 non réinscriptible, 41

O

objets
 création dans AppleScript, 37
 création dans JavaScript, 27
 création dans Visual Basic, 51
 création directe requise, 27
 création par script impossible, 28, 29
 dimensions, 31
 hiérarchie, 18
 sélection, 53
 objets, référencement. Voir référencement d'objets

P

paramètres, omission, 43
 picas, conversion, 30
 pages de texte
 contenu, 24
 utilisation d'illustrations de texte, 22
 point directeur droit, 32
 point directeur gauche, 32
 point zéro, 31
 points
 conversion, 30
 fixes, 31
 zéro, 31

points d'ancrage, 32
 points fixes, 31
 polices
 Voir aussi styles de caractère
 unités em, 30
 pouces, conversion de mesures, 30
 presse-papiers, vider avant de fermer, 26
 propriété aki, 30

Q

Q (unité), conversion, 30

R

rectangles
 création, 56
 fixes, 31
 rectangles fixes, 31
 référencement d'objets
 à propos de, 27
 AppleScript, 36

S

scripts
 à propos de, 6
 exécution, 10
 extensions de fichiers, 7
 installation, 10
 menu, 7
 prise en charge dans Illustrator, 7
 utilisation, 6
 SDK, 32
 sélection d'objets, 53
 sélections
 définition de contenu, 37, 46, 52
 utilisation, 37, 46, 52
 styles de caractère
 Voir aussi polices
 à propos de, 24
 symboles
 à propos de, 25
 éléments, 25

T

texte
 éléments d'illustration, 22
 pages. Voir pages de texte
 types de bloc, 22
 tracés
 à propos de, 32
 création, 54

U

unités de mesure, 30
unités em, 30

V

valeurs d'énumération, 56
valeurs de mesure, 30
variables
 suppression, 25
 utilisation, 25
VBScript
 bibliothèque de types, 9
 conventions d'attribution de nom, 19
 extension de fichier, 8
 valeurs d'énumération, 56
version d'application, 26
versions d'Illustrator, spécification, 26