# Module Javascript - Présentation du langage - Exercices d'application

February 15, 2011

# Contents

# Chapter 1

# Exercice

Pour l'ensemble des exercices de ce cours, nous utiliserons un éditeur de texte, et comme navigateur Internet, une version récente de Netscape ou de Mozilla. Comme vu dans le cours HTML, nous écrirons directement nos pages HTML, et nous les ouvrirons par la commande de Netscape "Fichiers -> ouvrir un fichier". Volontairement, nous épurerons au maximum le code HTML, de manière à nous concentrer sur le code Javascript.

## 1.1 Les premiers programmes Javascript

### 1.1.1 Insertion de codes dans une page HTML

Pour l'ensemble des exercices de ce cours, nous utiliserons comme étiteur de code HTML et Javascript XEmacs, et comme navigateur Internet, une version récente de Netscape. Comme vu dans le cours HTML, nous écrirons directement nos pages HTML dans XEmacs, et nous les ouvrirons par la commande de Netscape "Fichiers -> ouvrir un fichier". Volontairements, nous épurerons au maximum le code HTML, de manière à nous concentrer sur le code Javascript.

#### 1.1.1.1 Insertion de code Javascript à exécution directe

Sous XEmacs, écrire le code suivant, et enregistrer la page sous le nom "`Exercice 1.3.1`, dans le répertoire `Javascript/Exercices/`. Vous ouvrirez ensuite la page sous Netscape, en observant soigneusement l'interprétation du code et le comportement du navigateur.

```
<html>
<head>
<title>Exercice javascript 1.3.1</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF" text="#000000">
<script language = "javascript">
alert("Le message d'alerte s'affiche avant même le contenu de la page.\nCliquez ↩
    sur OK pour que la page s'affiche");
</script>
<h1>Exercice 1.3.1 : Code javascript a ex&eacute;cution directe</h1>
<p>Le script Javascript a &eacute;t&eacute; ex&eacute;cut&eacute; avant l' ↩
    affichage
  de la page.
</body>
</html>
```

Nous voyons ici une seconde instructions Javascript, l'instruction **alert**, qui permet d'ouvrir une boite de dialogue sur un navigateur. Nous aurons l'occasion de revenir sur cette instructions.

A retenir : quand le code Javascript est placé juste après la balise **<body>**, il est exécuté avant l'affichage de la page.

#### 1.1.1.2 Insertion de code Javascript à exécution différée

Selon le même principe que dans l'exercice précédent, écrire et affiché la page suivante, enregistrée sous le nom `Exercice 1.3.2`.

```
<html>
<head>
<title>Exercice javascript 1.3.2</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF" text="#000000">
<script language = "javascript">
function affiche()
{
   alert("Le message d'alerte s'affiche suite à un évènement.\nCliquez sur OK  ↩
       pour poursuivre");
}
</script>
<h1>Exercice 1.3.2 : Code javascript a ex&eacute;cution diff&eacute;r&eacute;e</ ↩
    h1>
<p>Le script Javascript est ex&eacute;cut&eacute; suite &agrave; l'arriv&eacute;e
  d'un &eacute;v&egrave;nement dans la page, ici quand l'utilisateur clique sur
  le bouton ci-dessous.
<form>
<center>
<input type = "button" name="evenement" value = "Cliquez ici pour générer un  ↩
    évènement" onclick= "javascript:affiche()"
</center>
</form>
</body>
</html>
```

Le code contient cette fois ci une fonction Javascript. Nous reviendrons sur ce principe de programmation, mais nous pouvons déjà remarquer que les instructions contenues dans cette fonction (l'instruction **alert** en l'ocurence) sont exécuté lors de l'appel à `affiche()`. Cet appel de la fonction est réalisé quand l'évènement **onclick** apparait sur le bouton du formulaire.

A retenir : pour une exécution de code Javascript différée, il est nécessaire d'englober le bloc d'instruction dans une fonction, qui est appelée depuis le code HTML.

#### 1.1.1.3 Insertion du code Javascript à l'intérieur d'une balise HTML

Toujours sur le même principe, écrire la page HTML `exercice1-3-3.html` contenant le code suivant :

```
<html>
<head>
<title>Exercice javascript 1.3.3</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<script language = "javascript">

</script>
</head>
<body bgcolor="#FFFFFF" text="#000000">

<h1>Exercice 1.3.3 : Code javascript ins&eacute;r&eacute; dans une balise HTML</ ↩
    h1>
<p>Le script Javascript est ex&eacute;cut&eacute; suite &agrave; l'arriv&eacute;e
  d'un &eacute;v&egrave;nement dans la page, ici quand l'utilisateur clique sur
  le bouton ci-dessous.
<form>
<center>
<input type = "button" name="evenement" value = "Cliquez ici pour générer un  ↩
    évènement" onclick= "javascript:alert('Ce est affiché suite à un évènement.\ ↩
    nCliquez sur OK pour poursuivre');"
```

```
</center>
</form>
</body>
</html>
```

Cette fois-ci, l'instruction **alert()** est directement insérée dans le code HTML, au niveau de l'attribut **onclick** de la balise button. Il est bien sûr possible de cumuler plusieurs instructions Javascript à la suite, en les séparant par des points - virgules. Les balises (au contenu vide) **<script></script>** placées en tête de la page permettent de préciser au névigateur qu'il aura du code à interpréter.

### 1.1.2 Insertion de code par appel de module externe

Dans les exercices précédents, nous avons inséré le code Javascript directement dans la page HTML. C'est la méthode la plus simple et la plus fréquemment utilisée dans la création de sites Internets.

Cependant, cette méthode trouve ses limites dans la réutilisation et la maintenance des fonctions. En effet, si vous utilisez souvent une même série d'intructions Javascript dans plusieurs pages HTML, et que vous souhaitez les modifier, vous devrez reprendre une par une l'ensemble des pages ... ce qui peut rapidement devenir fastidieux.

Une solution à ces problèmes est de placer des fonctions Javascript dans un module externe. Ce fichier contiendra le code source, et l'insertion dans la page HTML deviendra :

```
<script src = "URL du module externe">
..
..
</script>
```

Ces balises indiquent au navigateur non plus la nature du code qu'il doit utiliser, mais l'emplacement auquel il trouvera ce code. Ainsi, en modifiant le contenu du module externe, on modifiera le comportement de l'ensemble des pages qui s'y réfèrent.

#### 1.1.2.1 Rédaction du module externe

Le module externe doit être écrit au format texte simple, et portant l'extension .txt. Il doit être placé à son adresse d'appel. Il contient le code source Javascript, généralement écrit sous forme de fonctions.

Travail à faire : créer un module externe nommé **`mod_jvs1.txt`**, et le placer dans le répertoire `/javascript/exercices/modules/`. Y insérer le code de deux fonctions, similaires à la fonction **affiche()** vue dans les exercices précédents :

- **affiche1()** : ouvre une boite de dialogue affichant "Execution de la fonction 1"

  **affiche2()** : ouvre une boite de dialogue affichant "Execution de la fonction 2"

#### 1.1.2.2 Appel du module externe

Nous disposons donc désormais des fonctions affiche1() et affiche2(), dont les code sont décrits dans le module externes. Nous souhaitons utiliser ces fonctions dans une page HTML.

Travail à faire : selon le modèle vu dans le paragraphe 1.3.2, écrire la page HTML `exercice-1.4.2.html`, contenant deux boutons, exécutant chacun une des deux fonctions **affiche()** contenues dans le module externe `mod_jvs1.txt`.

```
<script src = "URL du module externe">
</script>
```

"URL du module externe" contiendra le chemin relatif du module externe par rapport à la page HTML, à savoir ici : `/modules/mod_jvs1.txt`. Par cette commande, le navigateur chargera en mémoire le contenu du module externe, et pourra exécuter les fonctions qui y sont contenues.