Les gestionnaires d'événement

Les gestionnaires d'événement vont apporter l'élément d'interactivité qui caractérise le JavaScript. Par leur intermédiaire, les actions du visiteur mettent en œuvre les outils de programmation du JavaScript.

La gestion de ces événements (déclenchés généralement par l'internaute) fait le lien entre le langage HTML et le JavaScript.

1. La notion d'événement

Passons en revue différents événements implémentés en JavaScript. Cette liste n'est cependant pas limitative.

`onAbort`	À l'arrêt du chargement de la page ou d'une image par le bouton "Stop" du navigateur.
`onBlur`	À la perte du focus par un élément HTML (généralement lorsqu'on quitte un élément de formulaire).
`onClick`	Au clic de la souris sur un élément HTML.
`onDbClick`	Au double clic de la souris sur un élément HTML.
`onError`	À l'apparition d'une erreur dans la page.
`onFocus`	À la prise du focus d'un élément de formulaire.
`onKeyDown`	À la pression d'une touche du clavier.
`onKeyUp`	Au relâchement d'une touche du clavier qui a été enfoncée.
`onKeyPress`	À la saisie par une touche du clavier (combinaison de `onKeyDown` et `onKeyUp`).
`onLoad`	Au chargement de la page par le navigateur.
`onMouseDown`	À la pression du bouton de la souris.
`onMouseMove`	Au déplacement de la souris.
`onMouseOut`	À l'abandon d'un élément HTML par la souris.
`onMouseOver`	Au survol d'un élément HTML par la souris.
`onMouseUp`	Au relâchement du bouton de la souris (suite de `onMouseDown`).
`onMove`	Au déplacement de la fenêtre.
`onReset`	Au clic sur le bouton Annuler (*reset*) d'un formulaire.
`onResize`	Au redimensionnement de la fenêtre du navigateur.
`onScroll`	À l'utilisation de la barre de défilement.
`onSelect`	À la sélection d'un texte dans un élément HTML.
`onSubmit`	Au clic sur le bouton Envoyer (*submit*) d'un formulaire.
`onUnLoad`	Au moment où l'utilisateur quitte la page.

Ces événements peuvent être associés à une multitude de balises que nous ne détaillons pas ici.

La syntaxe est :

```
événement="function()"
```

Soit un événement (ils commencent tous par "on"), le signe égal et la fonction associée par le concepteur, entourée par des apostrophes.

Exemple

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

```
onclick="alert('Vous avez cliqué sur cet élément')"
```

De façon littérale, au clic de l'utilisateur, une boîte d'alerte s'ouvre avec le message indiqué dans le script.

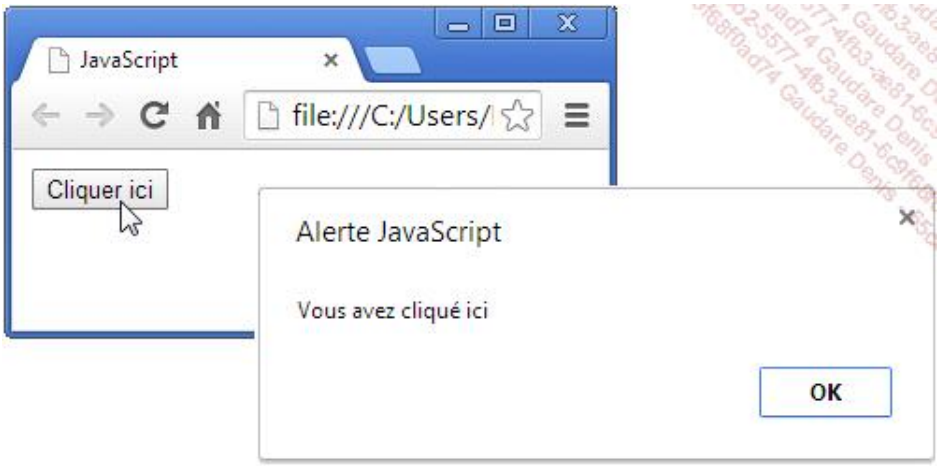
2. L'événement onClick

Le clic de la souris est un événement fréquemment utilisé. Il survient lorsque le visiteur clique, par exemple, sur un lien ou un élément de formulaire.

Exemple

Au clic sur le bouton, une boîte d'alerte surgit.

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
</head>
<body>
<form>
<input type="button" value="Cliquer ici" onclick="alert('Vous avez
cliqué ici')">
</form>
</body>
</html>
```



Le clic de la souris, dans le cas d'un lien par la balise <a> ... , est déjà utilisé par le HTML ou le XHTML pour atteindre la page spécifiée. Pour associer une action JavaScript au clic sur un lien, il a fallu introduire une notation particulière du genre :

```
<a href="javascript:alert('Bonjour !');">Cliquer ici !</a>
```

Ainsi dans l'attribut `href`, on indique au navigateur qu'il ne s'agit pas d'un lien mais d'une instruction JavaScript. D'où la notation `javascript`, double point et l'instruction retenue.

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

On rencontre aussi la notation :

```
<a href="javascript:void(0)" onclick="alert('Bonjour !'))"> Cliquer ici !</a>
```

Pour annuler la fonction de lien introduite par l'attribut `href`, l'expression `javascript:void(0)` est ici utilisée.

3. L'événement onFocus

Un événement est généré lorsqu'un élément est activé, par exemple une ligne de texte d'un formulaire.

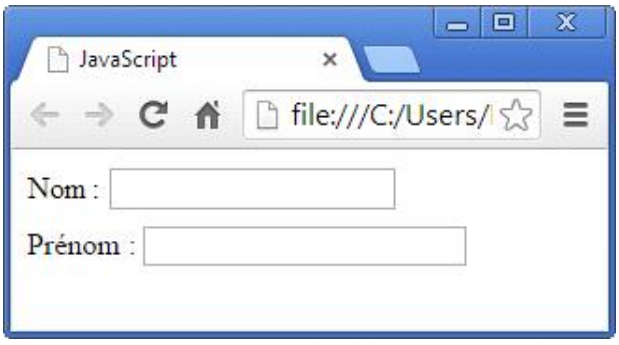
Exemple

Lorsque l'utilisateur clique dans une zone de texte, une inscription apparaît dans celle-ci.

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
</head>
<body>
<form>
Nom : <input id="t1" name="t1" onfocus="this.value='Entrez votre
nom ici.'"><br>
Prénom : <input id="t2" name="t2" onfocus="this.value='Entrez
votre prénom ici.'">
</form>
</body>
</html>
```

Après avoir cliqué dans la seconde ligne de texte, la capture d'écran suivante est obtenue.

Situation de départ :



Au focus d'un champ de formulaire :

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

4. L'événement onLoad et onUnLoad

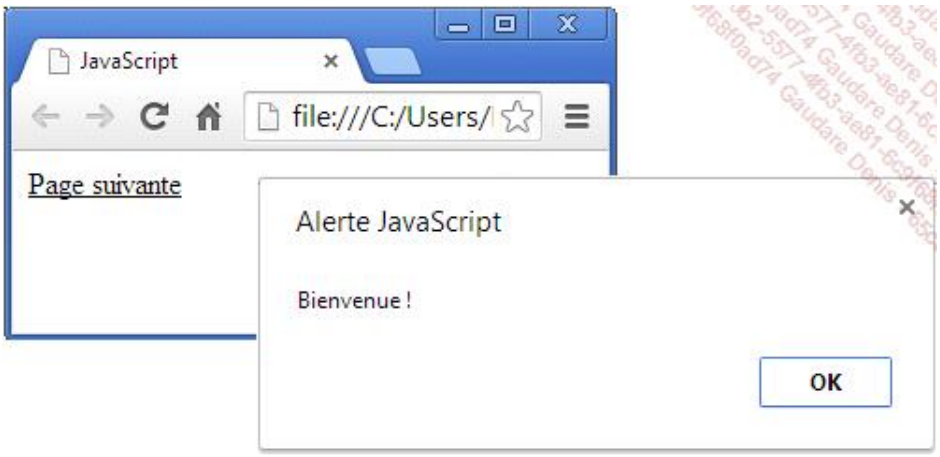
Au chargement (*load*) de la page ou à la sortie (*unload*) de la page, des événements sont générés.

Exemple

Un message d'accueil au chargement de la page et un autre message à la sortie de celle-ci.

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
</head>
<body onload="alert('Bienvenue !')" onunload="alert('A bientôt...')">
<a href="xxx.htm">Page suivante</a>
</body>
</html>
```

Ce script associe deux événements dans la même balise. Son écriture est simple. Il suffit d'encoder simplement les différents événements à la suite, car ils sont, depuis le HTML 4.0, considérés comme de simples attributs.



5. L'événement onMouseOver et onMouseOut

L'événement onMouseOver se produit lorsque le pointeur de la souris passe au-dessus d'un lien ou d'une image,

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

sans cliquer dessus.

L'événement `onMouseOut`, généralement associé à un événement `onMouseOver`, se produit lorsque le pointeur quitte la zone sensible (lien, image ou balise `<div>`).

Exemple

Ces deux événements sont souvent utilisés pour réaliser un effet, désormais classique, d'affichage d'une autre image au survol de l'image originale par le pointeur de la souris. Cet effet porte le nom de *rollover* dans la documentation anglo-saxonne.

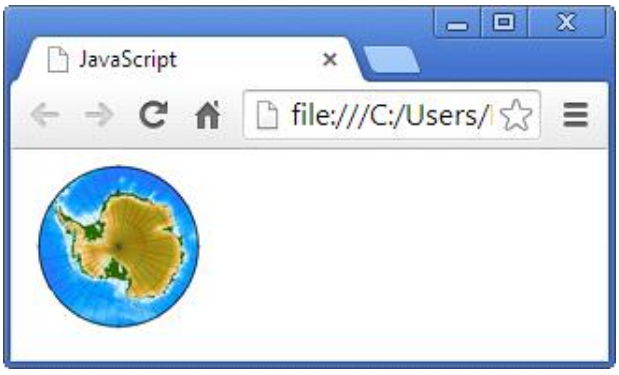
Il importe que les dimensions des images concernées par le script soient rigoureusement identiques.

Soit les images :

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
</head>
<body>
<img id="image" alt="mappemonde" src="monde1.png"
onmouseover="document.getElementById('image').src='monde2.png'
"onmouseout="document.getElementById('image').src='monde1.png'">
</body>
</html>
```

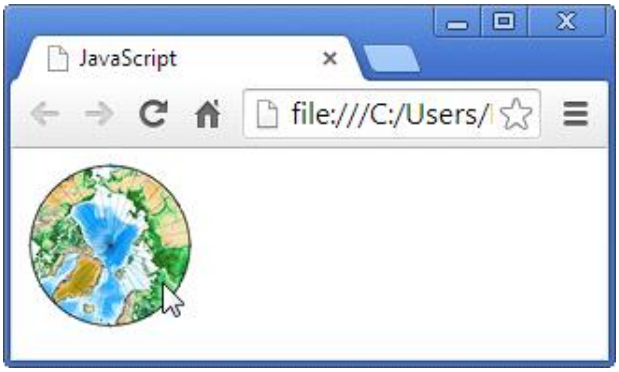
La balise image `<img>` affiche l'image initiale monde1.png (`src="monde1.png"`). On a pris soin de donner un identifiant à cette balise image (`name="image"`). Au survol de la souris (`onmouseover`), l'image monde2.png apparaît. Pour ce faire, on indique à JavaScript qu'il doit chercher dans le document l'objet "image" et l'attribut `src` qui fournit l'adresse de la nouvelle image. Soit le chemin complet, `document.getElementById("image")`. Lorsque le pointeur de la souris quitte l'image (`onmouseout`), l'image originale revient.

Situation initiale :



© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

Au survol de l'image par la souris :



6. L'événement onSubmit

L'événement créé par le clic sur le bouton d'envoi d'un formulaire peut judicieusement être mis à profit pour, par exemple, effectuer des vérifications concernant la saisie des champs de ce formulaire.

Exemple

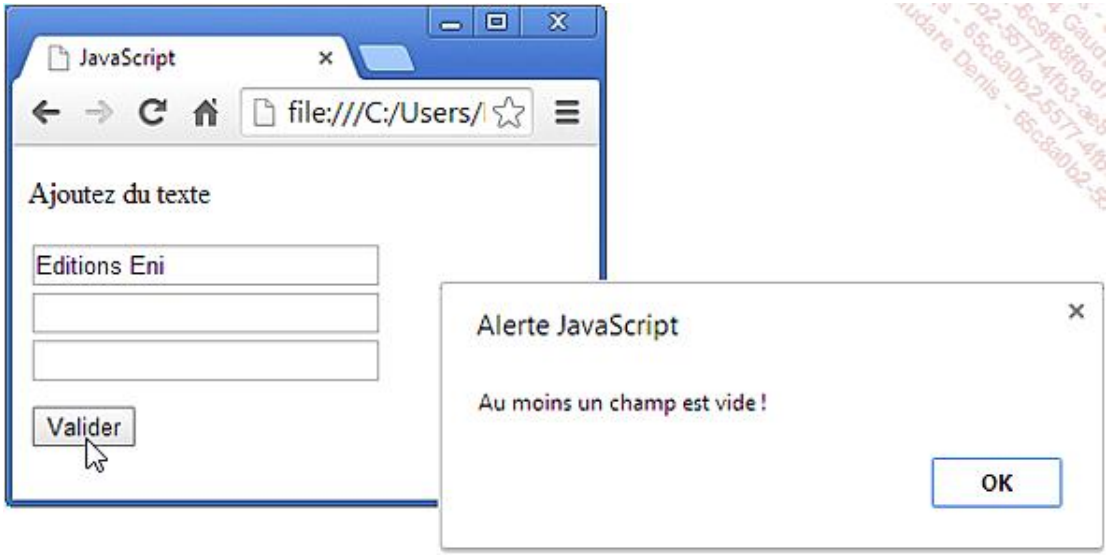
Ce script vérifie si les champs du formulaire ont bien été complétés.

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
<script>
function verif() {
a = document.getElementById("texte1").value;
b = document.getElementById("texte2").value;
c = document.getElementById("texte3").value;
if ((a && b && c) == "")
alert("Au moins un champ est vide !");
}
}
</script>
</head>
<body>
<p>Ajoutez du texte</p>
<form onsubmit="verif()">
<input id="texte1" name="texte1" size="26"><br>
<input id="texte2" name="texte2" size="26"><br>
<input id="texte3" name="texte3" size="26"><br>
<input type="submit">
</form>
</body>
</html>
```

Quand on clique sur le bouton d'envoi du formulaire, la fonction `verif()` est appelée `(onsubmit="verif()")`.

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

Pour la lisibilité du script, la valeur des différentes lignes de texte du formulaire est sauvegardée dans une variable. L'accès à cette valeur s'effectue par l'écriture du chemin désormais habituel document.formulaire.textex.value. Par un test conditionnel `if`, on vérifie si une des lignes de texte est vide, soit si a et b et c est vide (&& pour le et logique). Dans ce cas, une boîte d'alerte apparaît.



© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare