

Les formulaires

Avec JavaScript, les formulaires HTML prennent une toute autre dimension. N'oublions pas qu'en JavaScript on peut accéder à chaque élément d'un formulaire pour, par exemple, y lire et/ou écrire une valeur. Tous ces éléments renforcent les capacités interactives des pages web.

1. La ligne de texte

La zone de texte est l'élément d'entrée/sortie par excellence de JavaScript.

La ligne de texte est créée par la balise `<input type="text">`.

La ligne de texte possède trois propriétés :

Propriété	Description
name	Indique le nom du contrôle.
defaultvalue	Indique la valeur par défaut qui sera affichée dans la zone de texte.
value	Indique la valeur en cours de la zone de texte. Soit celle saisie par l'utilisateur ou si celui-ci n'a pas rentré de valeur, la valeur par défaut.

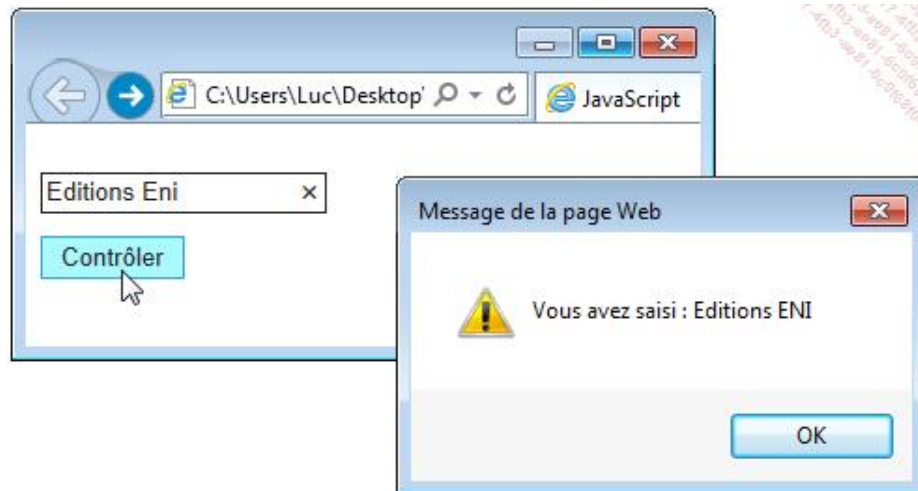
a. Lire une valeur

Saisir une valeur dans la zone de texte et la reproduire dans une boîte d'alerte.

Le script complet est :

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
<style>
#bouton { margin-top: 12px;}
</style>
<script>
function controle() {
var test = document.getElementById("input").value;
alert("Vous avez saisi : " + test);
}
</script>
</head>
<body>
<form>
<input type="text" id="input" name="input" value=""><br>
<input type="button" id="bouton" value="Contrôler"
onclick="controle()">
</form>
</body>
</html>
```

Lorsque le bouton **Contrôler** est cliqué, JavaScript appelle la fonction `controle()`. Cette fonction, définie entre les balises `<head>`, récupère, par l'intermédiaire de la variable `test`, la valeur de la zone de texte `var test = document.getElementById("input").value`. Cette variable est alors fournie comme argument d'une boîte d'alerte.



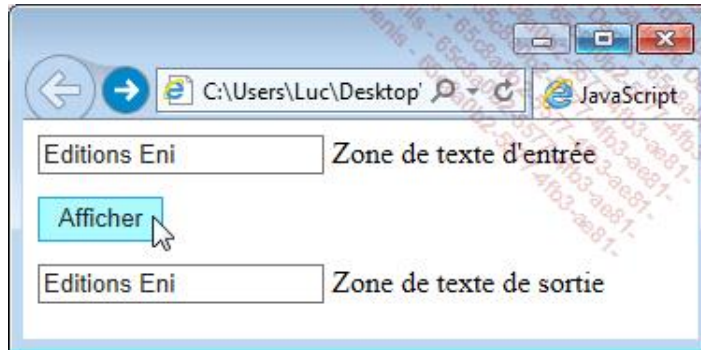
b. Reproduire une valeur

Entrer une valeur quelconque dans la zone de texte d'entrée. Afficher cette même valeur dans la zone de texte de sortie après avoir cliqué sur un bouton.

Voici le code :

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
<style>
#bouton { margin: 12px 0 12px 0}
</style>
<script>
function afficher() {
var testin = document.getElementById("input").value;
document.getElementById("output").value = testin;
}
</script>
</head>
<body>
<form>
<input type="text" id="input" name="input" value="">&nbsp;Zone de
texte d'entrée <br>
<input type="button" id="bouton" value="Afficher"
onclick="afficher()"><br>
<input type="text" id="output" name="output" value="">&nbsp;Zone
de texte de sortie
</form>
</body>
```

Lorsque le bouton **Afficher** est cliqué, JavaScript appelle la fonction `afficher()`. Cette fonction `afficher()` récupère, par l'intermédiaire de la variable `testin`, la valeur de la zone de texte d'entrée (`document.getElementById("input").value`). À l'instruction suivante, la variable `testin` est affectée à la valeur de la zone de texte de sortie en JavaScript : (`document.getElementById("output").value = testin`).



c. Tester un formulaire vide

Il est parfois impératif qu'un élément de formulaire soit rempli par le visiteur. Ce petit script teste si le visiteur a omis d'y introduire certaines données.

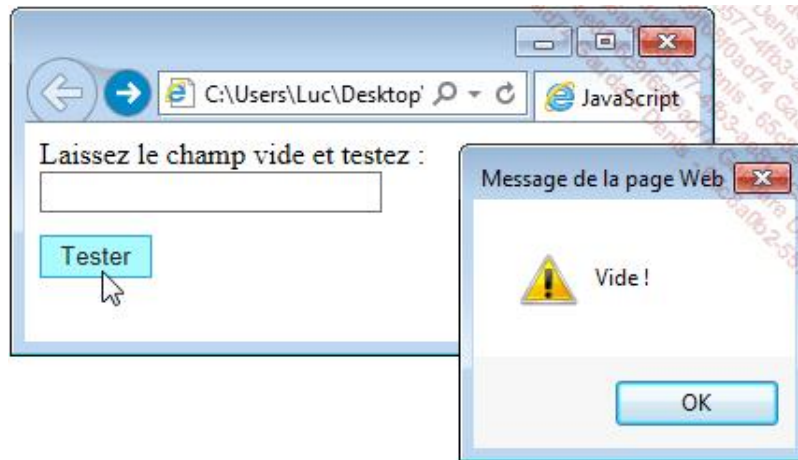
Nous avons déjà abordé cette problématique dans le chapitre précédent mais nous utilisons ici une autre façon de procéder.

Le script est :

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
<style>
#bouton { margin-top: 12px;}
</style>
<script>
function valider(){
input = document.getElementById("entree").value.length;
if (input == 0){
alert("Vide !");
}
}
</script>
</head>
<body>
<form>
Laissez le champ vide et testez :<br>
<input type="text" id="entree" name="entree" size="25"
value=""><br>
<input type="button" id="bouton" value="Tester"
```

```
onclick="valider() ">
</form>
</body>
</html>
```

La fonction `valider()` appelée par le clic sur le bouton (`onclick="valider()"`), récupère dans la variable `input` la longueur de ce qui a été saisi dans la zone de texte (`document.getElementById("entree").value.length`). Si cette longueur est égale à 0 (`input == 0`), l'utilisateur est averti par une boîte d'alerte que la zone est vide.

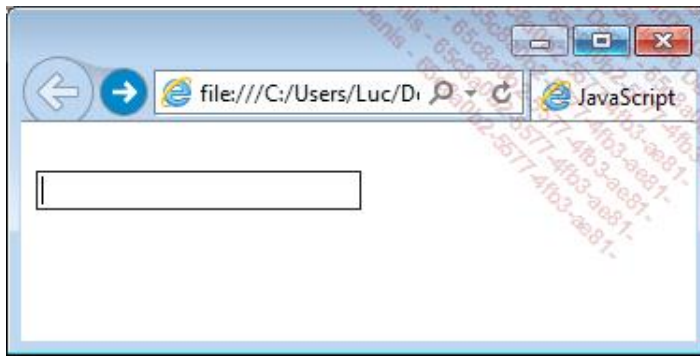


d. Donner le focus

Il est souvent convivial de placer directement le curseur dans la première zone de texte d'un formulaire.

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
</head>
<body onload="document.getElementById("entrée").focus();">
<br>
<form>
<input type="text" id="entree" name="entree" size="25" value="">
</form>
</body>
</html>
```

Au chargement de la page (événement `onload` du `body`), le focus est donné à la zone de texte dont le chemin a été déterminé par `document.getElementById("entrée")`.



Le curseur est bien positionné directement dans la zone de texte.

e. Saisie d'un nombre

Dans certains cas, le champ d'un formulaire ne devra accepter que des chiffres.

Le script suivant permet de le vérifier.

La fonction JavaScript `isNaN(argument)` est utilisée : elle évalue l'argument pour déterminer s'il ne s'agit pas d'un nombre (NaN pour *Not a Number*).

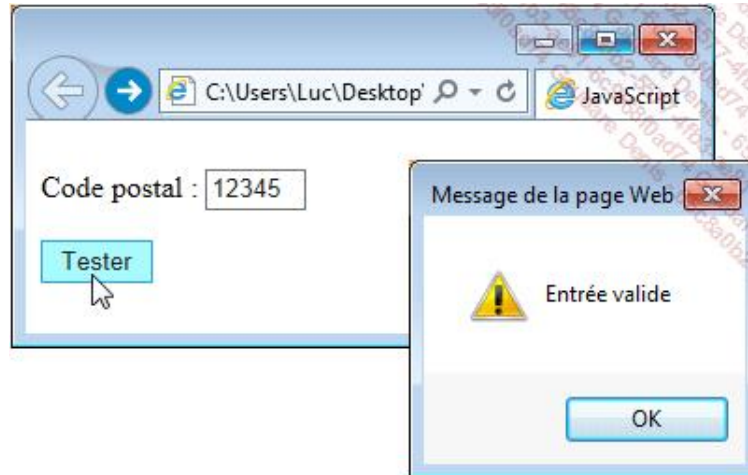
Le code devient :

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<script>
function verif() {
n=document.getElementById("nombre").value;
if (isNaN(n) || n == 0) {
alert("Entrer un nombre SVP !");
}
else {
alert("Entrée valide");
}
}
</script>
</head>
<body>
<br>
<form>
Code postal : <input type="text" id="nombre" name="nombre"
size="3" maxlength="5"><br>
<p><input type="button" name="bouton" value="Tester"
onclick="verif()"><p>
</form>
</body>
</html>
```

Une zone de texte de cinq caractères maximum (`maxlength="5"`) est prévue pour l'encodage du code postal.

Celui-ci doit comporter uniquement des chiffres et donc doit pouvoir être interprété comme un nombre. Le clic sur le bouton **Tester** appelle la fonction `verif()`.

Cette fonction initialise d'abord la variable `n` avec la valeur de la zone de texte dédiée au code postal (`document.getElementById("nombre").value`). Un test conditionnel (`if`) est effectué pour vérifier si la valeur de `n` n'est pas un nombre (`isNaN(n)`) ou est vide (`n == 0`). Dans ce cas, la valeur saisie dans le champ réservé au code postal est incorrecte et une boîte d'alerte invite le visiteur à entrer un nombre correct. Sinon, l'entrée est valide.

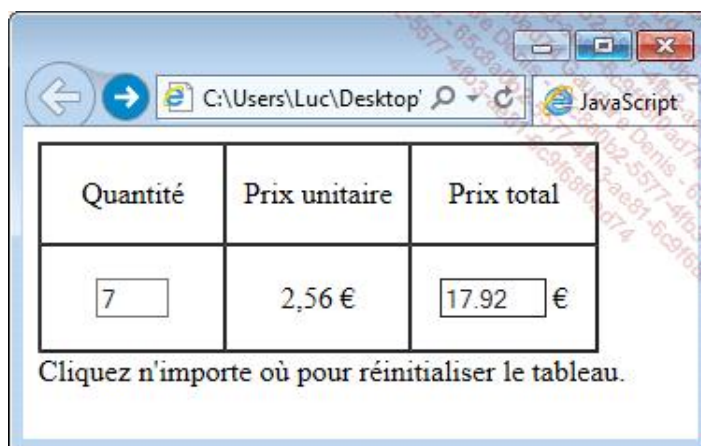


f. Calcul automatique

Les zones de texte peuvent également être utilisées pour recevoir des données ou contenir le résultat d'opérations effectuées par le script.

Dans l'exemple suivant, nous allons demander au visiteur la quantité de produits commandés. Au clic dans la zone de texte du prix total, celui-ci est automatiquement affiché.

Il est peut-être préférable, à ce stade, de déjà jeter un coup d'oeil à la capture d'écran suivante.



Le fichier devient :

```
<!DOCTYPE html>
<html lang="fr">
<head>
```

```

<title>JavaScript</title>
<meta charset="UTF-8">
<style>
table { width: 300px;
        border: 1px solid black;
        border-collapse: collapse;
        margin-bottom: 12px;}
p { text-align: center;}
td { width: 100px;
      border: 1px solid black;}
</style>
<script>
function calcul() {
a = document.getElementById("elem1").value;
b = document.getElementById("elem2").value;
document.getElementById("elem3").value = a * b;
}
function annul() {
document.getElementById("elem1").value="";
document.getElementById("elem3").value="";
}
</script>
</head>
<body>
<form>
<table>
<tr>
<td>
<p>Quantité</p>
</td>
<td>
<p>Prix unitaire</p>
</td>
<td>
<p>Prix total</p>
</td>
</tr>
<tr>
<td>
<p><input type="text" id="elem1" name="elem1" size="2"
value=""></p>
</td>
<td>
<p><input type="hidden" id="elem2" name="elem2" value="2.56">2,56
€</p>
</td>
<td>
<p><input type="text" id="elem3" name="elem3" size="5" value=""
onfocus="calcul()" onblur="annul()"> €</p>
</td>
</tr>
</table>
</form>
Cliquez n'importe où pour réinitialiser le tableau.

```

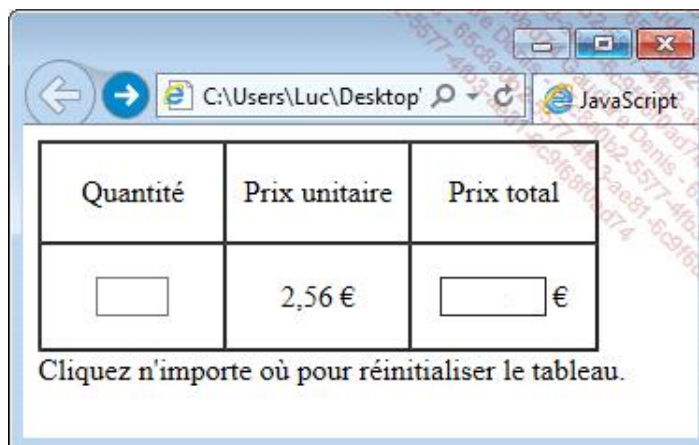
```
</body>
</html>
```

En cliquant dans la zone de texte du prix total, donc en donnant le focus, la fonction `calcul()` est appelée (`onfocus="calcul()"`).

Dans cette fonction `calcul()`, la valeur saisie dans la zone de texte relative aux quantités (`a = document.getElementById("elem1").value`) est sauvegardée dans la variable `a` et la valeur du prix unitaire entrée dans l'élément de formulaire caché `<input type="hidden">` (`b = document.getElementById("elem2").value`) est stockée dans la variable `b`. La multiplication de `a` et de `b` donne le prix total affiché dans la zone de texte prévue à cet effet (`document.getElementById("elem3").value = a * b;`).

Nous nous sommes servis de ce script pour illustrer l'événement `onBlur`. À la fin du script, la zone de texte du prix total a le focus. En cliquant n'importe où dans la page, cette zone perd le focus. C'est ce que désigne le terme de *Blur*. À cet événement est associée la fonction `annul(form) : onblur="annul(form)"`.

Cette fonction remet une chaîne de caractères vide dans les deux zones de texte (`document.getElementById("elem1").value=""` et `document.getElementById("elem3").value=""`). Ce qui réinitialise le script.



2. Les boutons de choix unique

Les boutons de choix unique, appelés aussi boutons radio, sont utilisés pour noter un choix, et seulement un seul, parmi un ensemble de propositions.

Les boutons de choix unique sont créés par les balises `<input type="radio">`.

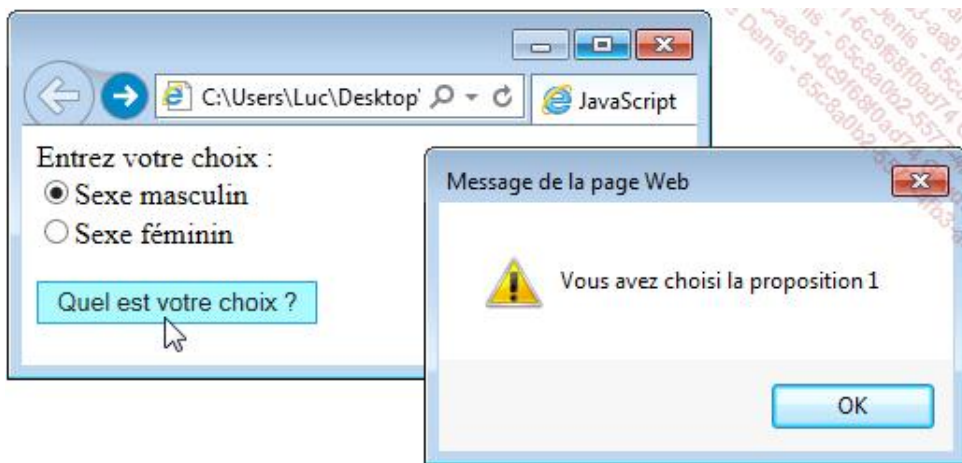
Propriété	Description
<code>name</code>	Indique le nom du contrôle. Tous les boutons portent le même nom.
<code>index</code>	L'index ou le rang du bouton radio démarrant à 0.
<code>checked</code>	Indique l'état en cours de l'élément radio (sélectionné ou non).
<code>defaultchecked</code>	Indique l'état du bouton sélectionné par défaut.
<code>value</code>	Indique la valeur de l'élément radio.

Exemple


```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
<script>
function choixsexe() {
if (document.getElementById("formulaire").choix[0].checked) {
alert("Vous avez choisi la proposition " +
document.getElementById("formulaire").choix[0].value);
}
if (document.getElementById("formulaire").choix[1].checked) {
alert("Vous avez choisi la proposition " +
document.getElementById("formulaire").choix[1].value);
}
}
}
</script>
</head>
<body>
Entrez votre choix :
<form id="formulaire">
<input type="radio" name="choix" value="1">Sexe masculin<br>
<input type="radio" name="choix" value="2">Sexe féminin<br>
<p><input type="button" value="Quel est votre choix ?"
onclick="choixsexe()"></p>
</form>
</body>
</html>
```

Dans le formulaire nommé `form`, deux boutons radio sont déclarés. Notez que le même nom est utilisé pour les deux boutons. Ensuite, un bouton déclenche par `onclick` la fonction `choixsexe()`.

Cette fonction vérifie quel bouton radio a été coché. Ces données des boutons sont disponibles grâce aux indices établis par rapport au nom des boutons radio soit `choix[0]`, `choix[1]`. La propriété `checked` du bouton est testée par `if (document.getElementById("formulaire").choix[x])`. Dans l'affirmative, une boîte d'alerte s'affiche. Ce message reprend la valeur (voir `value`) attachée à chaque bouton par le chemin `document.getElementById("formulaire").choix[x].value`.



3. Les boutons de choix multiples

Les boutons de choix multiples (aussi appelés *checkbox*) sont utilisés pour noter un ou plusieurs choix parmi un ensemble de propositions.

Les boutons de choix multiples sont créés par la balise `<input type="checkbox" />`.

Propriété	Description
name	Indique le nom du contrôle. Toutes les cases à cocher portent un nom différent.
checked	Indique l'état en cours de l'élément case à cocher (sélectionné ou non).
defaultchecked	Indique l'état du bouton sélectionné par défaut.
value	Indique la valeur de l'élément case à cocher.

Exemple

Soit la question à choix multiples suivante :

Sélectionner les balises Html :

- `<h1> ... </h1>`.
- `<a> ... `.
- `<op> ... </op>`.
- ` ... `.

La bonne réponse est la première, la deuxième et la quatrième proposition.

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
<script>
function reponse(form) {
if ((document.getElementById("check1").checked) == true &&
(document.getElementById("check2").checked) == true &&
```

```

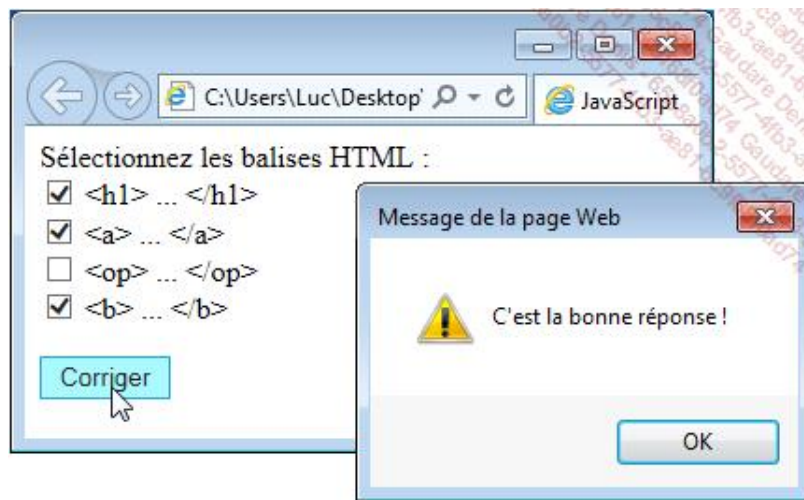
(document.getElementById("check3").checked) == false &&
(document.getElementById("check4").checked) == true)
{
alert("C'est la bonne réponse !");
}
else
{
alert("Désolé, continuez à chercher.");
}
}
}
</script>
</head>
<body>
Sélectionnez les balises HTML :
<form>
<input type="checkbox" id="check1" name="check1" value="1">
<h1> ... </h1><br>
<input type="checkbox" id="check2" name="check2" value="2">
<a> ... </a><br>
<input type="checkbox" id="check3" name="check3" value="3">
<op> ... </op><br>
<input type="checkbox" id="check4" name="check4" value="4">
<b> ... </b><br>
<p><input type="button" value="Corriger" onclick="reponse()"></p>
</form>
</body>
</html>

```

Dans le formulaire nommé `form`, quatre cases à cocher sont déclarées. Notez qu'un nom différent est utilisé pour les quatre boutons. Vient ensuite un bouton qui déclenche la fonction `reponse()` au moment du clic sur ce dernier.

Cette fonction teste les cases à cocher sélectionnées. Pour avoir la bonne réponse, il faut que les cases 1, 2 et 4 soient cochées. Le nom des cases à cocher permet de les identifier, notamment la propriété `checked` du bouton par `document.getElementById("checkx").checked`. Dans l'affirmative, la valeur renvoyée est `true`. Dans la négative, la valeur renvoyée est `false`.

Un test conditionnel vérifie par le "et logique" (`&&`) si les propositions 1, 2 et 4 sont vraies et la proposition 3 est fausse. Dans ce cas, une boîte d'alerte s'affiche pour annoncer la bonne réponse. Dans la négative, une autre boîte d'alerte invite à refaire le test.



4. Le menu déroulant

La liste de sélection permet de proposer diverses options sous la forme d'un menu déroulant, dans lequel l'utilisateur peut sélectionner une option. Une fois l'option sélectionnée, elle reste alors affichée.

Le menu déroulant est créé par la balise `<select> ... </select>` et les éléments de la liste par une ou plusieurs balises `<option> ... </option>`.

Propriété	Description
name	Indique le nom de la liste déroulante.
length	Indique le nombre d'éléments de la liste.
selectedIndex	Indique le rang à partir de 0 de l'élément de la liste qui a été sélectionné par l'utilisateur.
defaultselected	Indique l'élément de la liste sélectionné par défaut.

Un exemple habituel :

Une liste déroulante permet de choisir le navigateur. Les options proposées sont Internet Explorer, Firefox, Google Chrome et Autre.

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
<script>
function liste(form) {
alert("Le navigateur " +
(document.getElementById("list").selectedIndex + 1));
}
</script>
</head>
<body>
<p>Quel navigateur utilisez-vous ?</p>
<form>
```

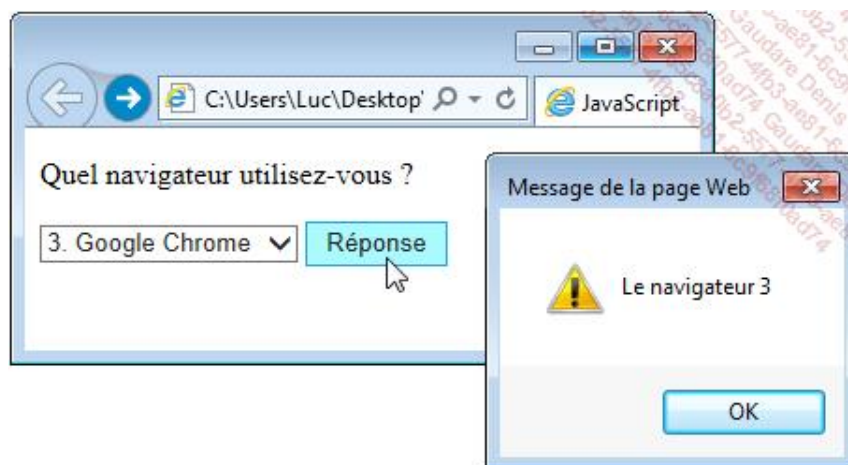
```

<select id="list" name="list">
<option value="1">1. Internet Explorer</option>
<option value="2">2. Firefox</option>
<option value="3">3. Google Chrome</option>
<option value="4">4. Autre</option>
</select>
<input type="button" value="Réponse" onclick="liste()">
</form>
</body>
</html>

```

Dans le formulaire nommé form, on déclare une liste de sélection par la balise `<select> ... </select>`. Entre ces balises, on déclare les différentes options de la liste par autant de balises `<option> ... </option>`. Ensuite un bouton déclenche la fonction `liste()`. Cette fonction fait ressortir l'option sélectionnée. Le chemin complet de l'élément sélectionné est `document.getElementById("list").selectedIndex`.

Comme l'index commence à 0, il faut ajouter 1 pour obtenir le rang exact de l'élément.



5. Le bouton d'envoi

Le bouton d'envoi, créé par la balise `<input type="submit">`, permet de transmettre les données du formulaire selon les spécifications de l'attribut `action="..."` de la balise `<form>`.

Propriété	Description
name	Indique le nom du contrôle.
value	Par défaut, le texte du bouton d'envoi est déterminé par votre navigateur, pour exemple Soumettre la requête sous Internet Explorer et Envoyer sous Firefox. Il est cependant possible de personnaliser le texte par défaut du bouton par l'attribut <code>value="valeur"</code> .
disabled	Permet de désactiver le bouton d'envoi.

6. Le bouton de réinitialisation

Il est utile de prévoir pour l'utilisateur la possibilité d'annuler toute saisie effectuée dans le formulaire et ainsi de réinitialiser (*reset*) un formulaire vierge.

Cette opération est réalisée par la balise `<input type="reset">`.

Tout comme pour le bouton de soumission, il est possible de modifier le texte par défaut du bouton par l'attribut `value="valeur"`.

Les autres attributs sont identiques à ceux du bouton d'envoi.

7. Le bouton de commande

Le bouton d'envoi et le bouton de réinitialisation, dans les deux sections précédentes, ont des fonctions bien définies par le langage HTML ou XHTML, ce sont respectivement les fonctions consistant à envoyer le formulaire selon les instructions définies par l'attribut `action` de la balise `<form>` et à réinitialiser un formulaire vide de toute donnée.

Il faut cependant prévoir des boutons dont la fonction pourrait être définie par le concepteur de la page, généralement grâce au JavaScript.

Pour ce faire la balise `<input type="button">` (qui prise isolément ne fait rien) est utilisée. L'action est alors définie par un gestionnaire d'événement, généralement du type `onclick`.

Propriété	Description
name	Avec l'attribut <code>name="nom"</code> , vous pouvez attribuer un nom au bouton.
value	Permet de définir le texte du bouton. L'attribut <code>value</code> est, dans le cas présent, obligatoire.

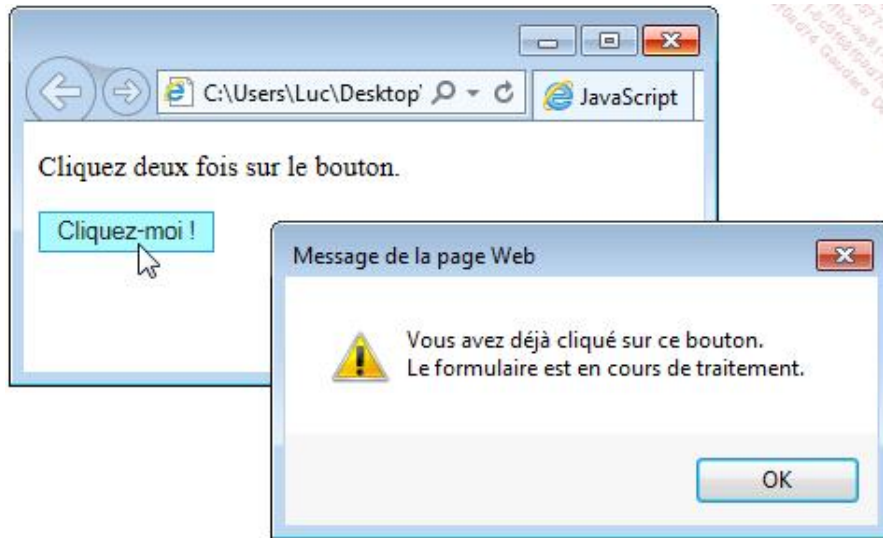
Exemple

Certains utilisateurs (impatients) ont la fâcheuse manie de cliquer plusieurs fois sur le bouton d'envoi. Ce script calmera leur ardeur.

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>JavaScript</title>
<meta charset="UTF-8">
<script>
var nombreclac = 0;
function compteclic() {
nombreclac++;
if (nombreclac>1) {
alert("Vous avez déjà cliqué sur ce bouton.\nLe formulaire est en
cours de traitement.");
}
}
}
</script>
</head>
<p>Cliquez deux fois sur le bouton.</p>
<form>
<input type="button" value="Cliquez-moi !" onclick="compteclic()">
</form>
</body>
```

Au clic sur le bouton d'envoi, la fonction `compteclique()` est appelée.

Cette fonction incrémente d'une unité (`nombreclique++`) la variable `nombreclique` qui a été préalablement initialisée à 0 (`var nombreclique=0`). Le script effectue alors un test (`if`) pour vérifier si la variable `nombreclique` est supérieure à 1 (`nombreclique>1`). Dans ce cas, une boîte d'alerte s'affiche (`alert("Vous avez déjà cliqué sur ce bouton")`).



8. L'instruction `this`

L'instruction `this` est un raccourci qui permet de référencer l'objet courant. Elle est souvent utilisée lorsque du code JavaScript est utilisé au sein d'une balise HTML, ce qui permet alors de faire référence à l'objet défini par cette balise.

Exemple

```
<form name="form">
<input type="button" value="Cliquez-moi !"
onclick="compteclique(this)">
</form>
```