Support de cours 2 JavaScript

FRÉDÉRIC NADARADJANE

LA GRANDE CLASSE

03/04/2023

7.L'objet Window

L'objet Window Présentation

L'objet de la fenêtre est pris en charge par tous les navigateurs. Il représente la fenêtre du navigateur.

- Tous les objets JavaScript globaux deviennent automatiquement membres de l'objet fenêtre.
  - Les variables globales sont des propriétés de l'objet fenêtre.
  - Les fonctions globales sont des méthodes de l'objet fenêtre.
- Même l'objet document (du DOM HTML) est une propriété de l'objet fenêtre.

L'objet Window
Dimensions

Deux propriétés peuvent être utilisés pour déterminer la taille de la fenêtre du navigateur.

Les deux propriétés renvoient les tailles en pixels

Les deux propriétés renvoient les tailles en pixels :

- Window.innerHeight : la hauteur interne de la fenêtre du navigateur (en pixels)
- Window.innerWidth : la largeur interne de la fenêtre du navigateur (en pixels)

L'objet Window
Méthodes

Quelques autres méthodes :

- window.open() – ouvre une nouvelle fenêtre

myWindow = window.open("", "", "width=100, height=100")

- window.close() – ferme la fenêtre actuelle
- window.moveTo() – mouvement de la fenêtre actuelle
- window.resizeTo() – resize de la fenetre en cours

L'objet Window History

L'objet window.history contient l'historique des navigateurs.

Pour protéger la confidentialité des utilisateurs, il existe des limites à la façon dont JavaScript peut accéder à cet objet.

Quelques méthodes :

- history.back () – identique la page consulter précédant la page actuelle
- history.forward () – identique la page consulter suivante la page actuelle

L'objet Window
Boîte contextuelles

Les boites contextuelles que nous avons vues précédemment sont des méthodes de l'objet window :

- window.alert()
- window.prompt()
- window.confirm()

L'objet Window
Exercice

Ouvrez une window de taille 200x200. on demande à l'utilisateur la taille du resize souhaité et on l'applique. Bougez là ensuite à 600,300.

L'objet Window
Exercice

Ouvrez une window de taille 100x600. on resize à 400x200. Bougez là ensuite à 200,600.

8. Les évènements

Les évènements Présentation

Le JavaScript peut être exécuté lorsqu'une événement se produit.

Exemples d'évènements :

- Lorsqu'un utilisateur clique sur la souris
- Lorsqu'une page web a été chargée
- Lorsqu'une image a été chargée
- Lorsque la souris se déplace sur un élément
- Lorsqu'un champ d'entrée est modifié
- Lorsqu'un formulaire HTML est soumis
- Lorsqu'un utilisateur appuie sur une touche

Les évènements
Intégration – inline

Pour exécuter du code lorsqu'un utilisateur clique sur un élément, ajoutez du code JavaScript à un attribut d'événement HTML:

onclick = JavaScript

Exemple :

```
<h1 onclick ="this.innerHTML = 'Oooops'">Click on this text!</h1>
```

Les évènements
Intégration – inline

Une deuxième possibilité est d'appeler directement une fonction JavaScript pour effectuer une action, ou une série d'actions :

onclick= fonctionJavaScript

Exemple :

HTML :

```
<h1 onclick="changeText(event)"> Click on this text !</h1>
```

JS :

```
function changeText(event){
        event.target.innerHTML = "Oooooops!"
}
```

Les évènements
Intégration – externe

Enfin, une troisième possibilité existe permettant d'attacher un évènement à une balise HTML, offrant la possibilité de séparer complètement le HTML du JavaScript :

#objet récupéré#.onclick = changeText()

Exemple :

HTML :

```
<h1 id= 'titlePrimary'>Click on this text </h1>
```

JS :

```
document.getElementById("titlePrimary").onclick = changeText;

function changeText(event){

        event.target.innerHTML="Oooops!";}
```

Les évènements
List des évènements

Liste des évènements communs :

Propriété	Description
onchange	Un élément HTML a été modifié
onclick	L'utilisateur clique sur un élément HTML
onmouseover	L'utilisateur déplace la souris sur un élément HTML
onmouseout	L'utilisateur déplace la souris hors d'un élément HTML
onkeydown	L'utilisateur appuie sur une touche du clavier
onload	Le navigateur a fini de charger la page

Liste des événements :https://developer.mozilla.org/fr/docs/Web/Events

Les évènements
Exemple

Evénement lié à l'appui et le relâchement de la souris

Exemple :

HMTL :

```
<div onmousedown="mDown(event)" onmouseup="mUp(event)"
    style="background-color:green;width:90px;height:20px;padding:40px;">
    Click Me</div>
```

JS:

```
function mDown(event) {
  var elem = event.target;
  elem.style.backgroundColor = "#1ec5e5"; elem.innerHTML = "Click down";
}
function mUp(event) {
  var elem = event.target;
  elem.style.backgroundColor="#D94A38"; elem.innerHTML="Click up";
}
```

Les évènements
Exemple

Evénement lié au survol de la souris

Exemple :

HMTL :

```
<div onmouseover="mOver(this)" onmouseout="mOut(this)"
style="background-color:#D94A38;width:120px;height:20px;padding:40px;">
Mouse Over Me</div>
```

JS:

```
function mOver(obj) {
        obj.innerHTML = "Thank You";
}

function mOut(obj) {
        obj.innerHTML = "Mouse Over Me";
}
```

Les évènements
Exemple

Evénement lié au chargement

Exemple :

HMTL :

```
<body onload ="mymessage()"> </body>
```

JS:

```
function mymessage(){
    alert("This message was triggered from the onload event");
}
```

Les évènements
Exercice

Dans une page html, mettez un titre h1 : Bonjour

Ensuite dans le javascript, demander le nom de l'utilisateur, via un prompt.

Lors du clique sur le titre « bonjour », le texte doit afficher à la place de celui-ci un titre en vert :

"Merci d’avoir cliqué [Nom de l'utilisateur]".

Les évènements
Exercice

Afficher une div avec ces propriétés :

- Un background orange
- Dimension : 400 x 400
- Des bords arrondi de 5px

On demandera le nom d’utilisateur via un prompt et lors du survole, mettez une alert "Merci [#PRENOM#], d'avoir survolé le carré".

Les évènements
Exercice

Créer un p avec un lorem,

Au clique sur ce texte on demandera à l'utilisateur son âge, s'il à plus de 60 ans la taille du texte sera de 70px, sinon si l'utilisateur à plus de 12 ans le texte aura pour taille 24px, et le texte aura 12px dans les autre cas.

Les évènements
Exercice

Compteur de clique :

Faire une div qui sera de couleur rouge et aura pour dimension :

600x600

A chaque clique sur la div un compteur sera incrémenté et on affichera ce compteur en dessous de la div

Les évènements
Ecouteur d'événements - Présentation

L'écouteur d'événements permet de gérer les différents événements attachés à un objet HTML

- La méthode addEventListener() attache un gestionnaire d'événements à l'élément spécifié.
- La méthode addEventListener() n'écrase pas les gestionnaires d'événements existants
- Vous pouvez ajouter de nombreux gestionnaires d'événements à un seul élémént.
- Vous pouvez ajour plusieurs gestionnaires d'événements du même type à un élément, c'est-à-dire deux événements "clic".

Les évènements
Ecouteur d'événements - Avantages

Les avantages d'un écouteur d'événements :

- La méthode addEventListener() facilite le contrôle de la réaction de l'événement

- Vous pouvez facilement supprimer un écouteur d'événements en utilisant la méthode removeEventListener()

Les évènements
Ecouteur d'événements – Ajout d'événements

Syntaxe pour réagir au clic :

#element récupéré#.addEventListener("click", function(){ alert("Hello world");});

#element récupéré#.addEventListener("click", myFunction);

Ajouter des événements de différents types au même élément :

#element récupéré#.addEventListener("mouseover", myFunction);

#element récupéré#.addEventListener("click", mySecondFunction);

#element récupéré#.addEventListener("mouseout", myThirdFunction);

Les évènements
Ecouteur d'événements – Arguments

Passer des arguments à un écouteurs d'événements :

Exemple :

```
document.getElementById("myBtn").addEventListener("click",function(){
  myFunction(p1, p2);
});
```

Les évènements
Ecouteur d'événements – Supprimer

Supprimer un écouteur d'évènement :

#élément récupéré#.removeEventListener("mousemove", myFunction);

Exemple :

Document.getElementById("myDiv").removeEventListener("mousemove", myFunction);

Les évènements
Les transitions

On peut également ajouter l’attribut transition vue en css

JS :

```
function changeText(event){
    event.target.innerHTML = "oops";
    event.target.style.color = 'blue';
    event.target.style.transition = "all 1s ease-out";
    document.getElementById('lorem').style.color = 'red';
}


document.getElementById("text").onclick = changeText;
```

Les évènements
Les transitions

On peut également déplacer des éléments html en fonction du clique

HTML :

```
<p>Click anywhere to move the ball</p>
  <div id=“ball"></div>
```

Les évènements
Les transitions

JS

```
var b = document.getElementById("ball");

document.addEventListener('click',
function(ev){

    b.style.transform = 'translateY(' + (ev.clientY-
25) + 'px)' +'translateX(' + (ev.clientX-25) + 'px)';

    b.style.transition = "transform 1s";

});
```

CSS

```
p{padding-left:60px;}

#foo{

border-radius:50px;

width:50px;

height:50px;

background:#c00;

position:absolute;

top:0;

left:0;

}
```

Les évènements
Exemple

Créons un h1 avec le texte : « survolez-moi », qui lors du survole de celui-ci change pour devenir Merci d'avoir survoler.

Un bouton permettra d'arrêter l'evenement.

Les évènements
Exercice

Crée un lorem dans un p au click il changera en h1, le changement se fera avec une transition d’1s en ease-out.

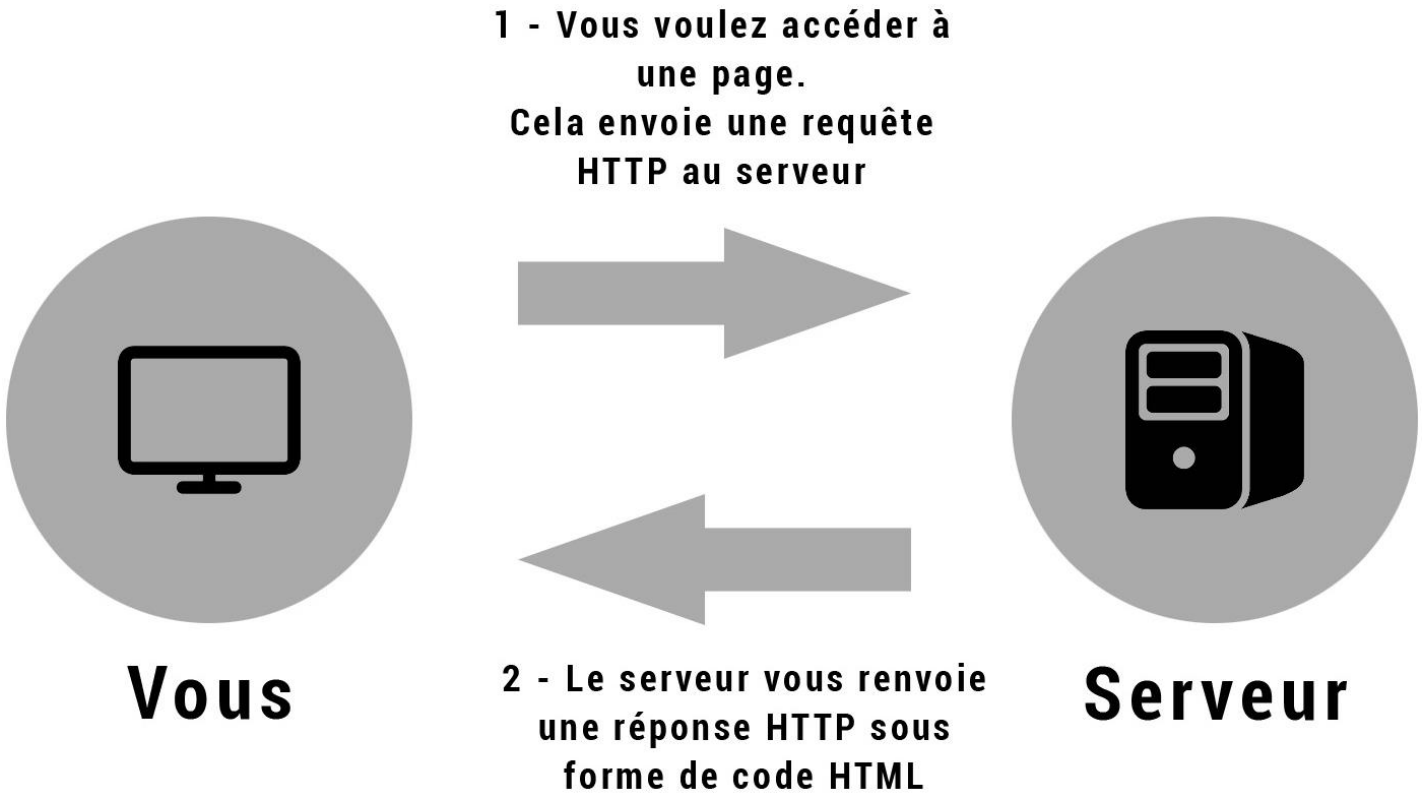
Le titre final sera en bleu.

9.AJAX

ASYNCHRONOUS JAVASCRIPT + XML

La notion de client - serveur

- Qu'avez-vous compris de la notion de client – serveur ?

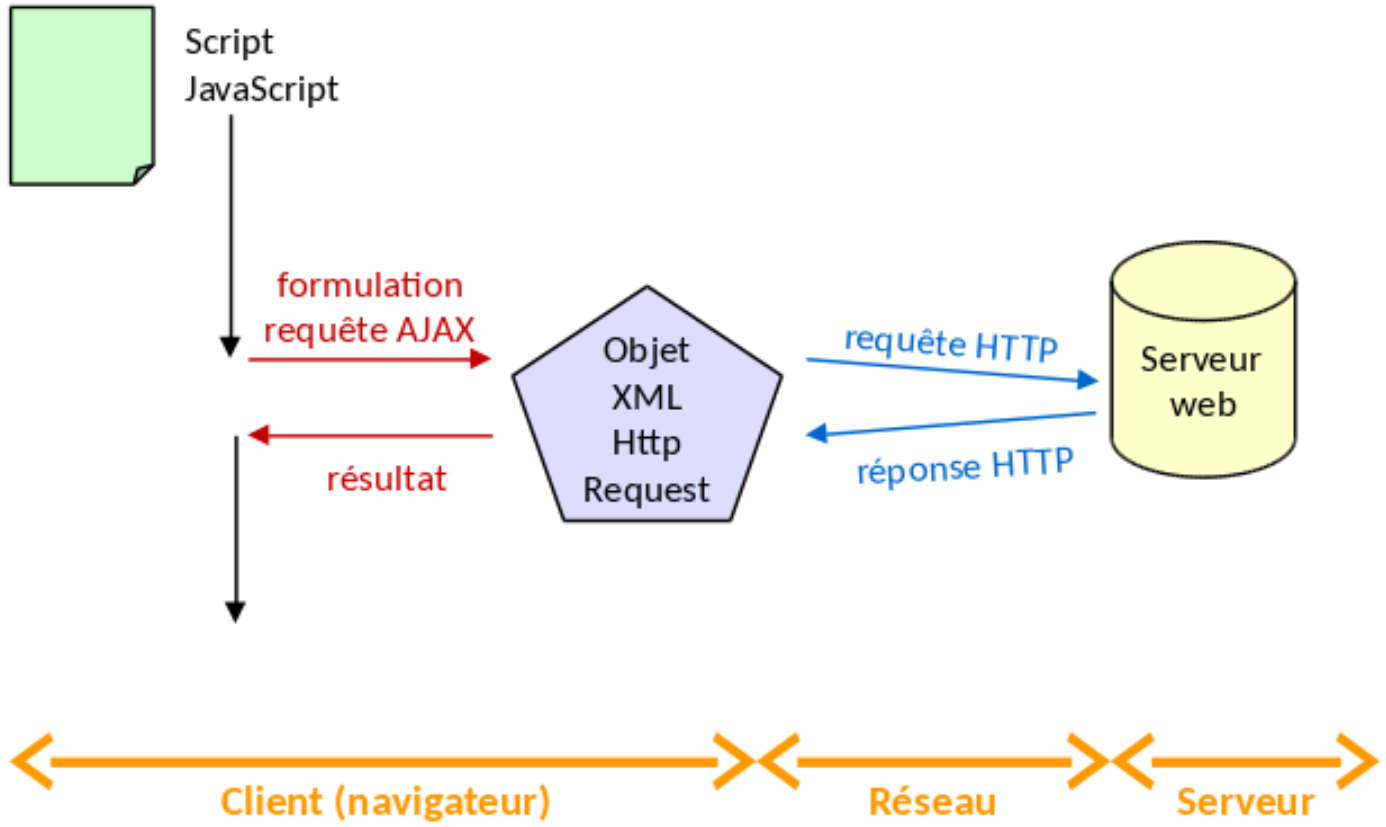


AJAX ? C'est quoi ?

AJAX (Asynchronous JavaScript + XML) est principalement utilisée pour apporter de l'interactivité au sein des pages d'un site web tout en économisant les ressources serveur.

En effet, AJAX permet de communiquer avec le serveur à l'aide de code Javascript en arrière-plan pendant que la page est affichée à l'écran. Ainsi le contenu de la page peut être modifié sans qu'il soit nécessaire de faire transiter et afficher la page en entier. Elle est particulièrement utilisée pour la mise à jour des formulaires et des paniers sur la plupart des sites web. C'est une technologie asynchrone : le code de la page continue de s'exécuter pendant que l'appel vers le serveur est effectué. Il faut garder à l'esprit cette information quand on utilise AJAX.

AJAX ? C'est quoi ?



AJAX
Comment construire une requête ajax

Pour recueillir des informations sur le serveur l'objet XHR dispose de deux méthodes:

- open: établit une connexion.
- send: envoie une requête au serveur.

Les données fournies par le serveur seront récupérées dans les champs de l'objet XMLHttpRequest:

- responseXml pour un fichier XML ou
- responseText pour un fichier de texte bru.

Il faut attendre la disponibilité des données, et l'état est donné par l'attribut **readyState** de XMLHttpRequest.

AJAX
L'objet XMLHttpRequest

Elle permet d'interagir avec le serveur, grâce à ses méthodes et ses attributs.

Attributs

readyState	le code d'état passe successivement de 0 à 4 qui signifie "prêt".
status	200 est ok 404 si la page n'est pas trouvée.
responseText	contient les données chargées dans une chaîne de caractères.
responseXml	contient les données chargées sous forme XML, les méthodes de DOM servent à les extraire.
onreadystatechange	propriété activée par un évènement de changement d'état. On lui assigne une fonction.

AJAX
L'objet XMLHttpRequest

Méthodes

open(mode, url, boolean)	mode: type de requête, GET ou POST url: l'endroit ou trouver les données, un fichier avec son chemin sur le disque. boolean: true (asynchrone) / false (synchrone). en option on peut ajouter un login et un mot de passe.
send("chaine")	null pour une commande GET.

AJAX
Construire une requête

Première étape: créer une instance

C'est juste une instance de classe classique mais deux options à essayer pour compatibilité avec les navigateurs.

```
1  if (window.XMLHttpRequest)    //  Objet standard
2  {
3      xhr = new XMLHttpRequest();     //  Firefox, Safari, ...
4  }
5  else  if (window.ActiveXObject)      //  Internet Explorer
6  {
7      xhr = new ActiveXObject("Microsoft.XMLHTTP");
8  }
```

AJAX
Construire une requête

Seconde étape: attendre la réponse

Le traitement de la réponse et les traitements qui suivent sont inclus dans une fonction, et la valeur de retour de cette fonction sera assignée à l'attribut **onreadystatechange** de l'objet précédemment créé.

```
xhr.onreadystatechange = function()
{
 // instructions de traitement de la réponse
};
```

```
if (xhr.readyState == 4)
{
// Reçu, OK
}
else
{
// Attendre...
}
```

AJAX
Construire une requête

Troisième étape: faire la requête elle-même

Deux méthodes de XMLHttpRequest sont utilisées:
- **open**: commande GET ou POST, URL du document, true pour asynchrone.
- **send**: avec POST seulement, données à envoyer au serveur.

La requête ci-dessous lit un document sur le serveur.

```
xhr.open('GET', 'https://jsonplaceholder.typicode.com/todos/1', true);
xhr.send(null);
```

AJAX
JSON.parse

Pour nous humain, xhr.responseText est un objet de type JSON car c'est ce que nous envoi l'API.

Pour le navigateur, xhr.responseText est juste une chaine de caractère.

Il faut donc lui faire comprendre que ce qu'il traite c'est bien un JSON.

La syntaxe :

```
JSON.parse(xhr.responseText).title
```

AJAX
Les films

Crée un compte gratuit chez http://www.omdbapi.com/apikey.aspx permet d'avoir la base de données de tout les films sorties au cinéma

Crée une requête AJAX (en reprenant la construction du cours)

1. A l'aide de la documentation (et de votre formateur), afficher toutes les informations sur le film titanic
2. Faites un prompt et affichez dans la console toutes les informations du film demandé par l'utilisateur.
3. Afficher le poster du film

AJAX
Les films

Crée un compte gratuit chez http://www.omdbapi.com/apikey.aspx permet d'avoir la base de données de tout les films sorties au cinéma

Crée une requête AJAX (en reprenant la construction du cours)

1. Créer une requête AJAX afin de récupérer le film irobot
2. Afficher le résultat de la requête dans une div qu'on appellera result

10.Les formulaires

Les formulaires
Présentation

Aujourd'hui vous connaissez que les prompts pour demander à l'utilisateur de rentrer des valeurs. Or il existe un autre moyen que vous connaissez surement qui sont les formulaires.

Pour rappel,

```
<form action="" method="get" class="form-example">
    <div class="form-example">
      <label for="name">Enter your name: </label>
      <input type="text" name="name" id="name" required>
    </div>
    <div class="form-example">
      <label for="email">Enter your email: </label>
      <input type="email" name="email" id="email" required>
    </div>
    <div class="form-example">
      <input type="submit" value="Subscribe!">
    </div>
  </form>
```

Les formulaires
Présentation

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 aller lire ou écrire une valeur,
- Noter un choix auquel on pourra associer un gestionnaire d'événement...
- Tous ces éléments renforceront grandement les capacités interactives de vos pages.

Les formulaires
Présentation

Mettons au point le vocabulaire que nous utiliserons.

Un formulaire est l'élément Html déclaré par les balises <form></form>.

Un formulaire contient un ou plusieurs éléments que nous appellerons des contrôles (widgets).

Ces contrôles sont notés par exemple par la balise <input type= ...>.

Les formulaires
Lire une valeur dans une zone de texte

HTML :

```
<form name="form1">
        <input type="text" name="input" value=""><br/>
        <input type="submit" name="bouton" value="Contrôler" onClick="controle(event)">
 </form>
```

JS :

```
function controle(event) {
var test = document.form1.input.value;
alert("Vous avez tapé : " + test);
}
```

Les formulaires
Lire une valeur dans une zone de texte

HTML :

```
 <form NAME="form2">
      <input type="text" name="input" value=""> Zone de texte d'entrée <BR>
      <input type="button" name="bouton" value="Afficher" onClick="afficher(form2)"><BR>
      <input type="text" name="output" value=""> Zone de texte de sortie
   <script src="js/formulaire.js"></script>
   </form>
```

JS :

```
function afficher(form2) {
var testin =document. form2.input.value;
document.form2.output.value=testin
}
```

Les formulaires
Les radios buttons

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

Propriété	Description
name	indique le nom du contrôle. Tous les boutons portent le même nom.
index	l'index ou le rang du bouton radio en commençant par 0.
checked	indique l'état en cours de l'élément radio
defaultchecked	indique l'état du bouton sélectionné par défaut.
value	indique la valeur de l'élément radio.

Les formulaires
Les radios buttons

HTML :

```
 <form name="form3">
      <input TYPE="radio" name="choix" value="1">Choix numéro 1<BR>
      <input TYPE="radio" name="choix" value="2">Choix numéro 2<BR>
      <input TYPE="radio" name="choix" value="3">Choix numéro 3<BR>
      <input TYPE="button"name="but" value="Quel et votre choix ?" onClick="choixprop(form3)">
   </form>
```

JS :

```
function choixprop(form3) {
if (form3.choix[0].checked) { alert("Vous avez choisi la proposition " + form3.choix[0].value) };
if (form3.choix[1].checked) { alert("Vous avez choisi la proposition " + form3.choix[1].value) };
if (form3.choix[2].checked) { alert("Vous avez choisi la proposition " + form3.choix[2].value) };
}
```

Les formulaires
Exercice : Newsletter

Faites un formulaire avec

- La civilité
- Un champ qui possède des bords arrondis de 5px (type email)
- un bouton envoyer de fond bleu clair (teinte au choix)

Une fois cliqué le formulaire renvoi une alert en disant "Merci [#Civilité#] d'avoir entrer votre e-mail, nous enverrons notre actualité à cette adresse : [#adresse de entrée#]"

Les formulaires
Les cases à cochés (checkbox)

Les boutons case à cocher sont utilisés pour noter un ou plusieurs choix (pour rappel avec les boutons radio un seul choix) parmi un ensemble de propositions. A part cela, sa syntaxe et son usage est tout à fait semblable aux boutons radio sauf en ce qui concerne l'attribut name.

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.
defaultchecked	indique l'état du bouton sélectionné par défaut.
value	indique la valeur de l'élément case à cocher.

Les formulaires
Les cases à cochés (checkbox)

HTML

```
<form name="form4">

    <input type="checkbox" name="check1"
value="1">choix numéro 1<br>

    <input type="checkbox" name="check2"
value="2">choix numéro 2<br>

    <input type="checkbox" name="check3"
value="3">choix numéro 3<br>

    <input type="button"name="but" value="corriger"
onclick="reponse(form4)">

 </form>
```

JS

```
function reponse(form4) {
if ( (form4.check1.checked) == true &&
(form4.check2.checked) == true &&
(form4.check3.checked) == false)
{

   alert("C'est la bonne réponse! ")

}
else
{

     alert("Désolé, continuez à chercher.")

}

}
```

Les formulaires
Les cases à cochés (checkbox)

HTML :

```
<form name="form4">
    <input type="checkbox" name="check1" value="1">choix numéro 1<br>
    <input type="checkbox" name="check2" value="2">choix numéro 2<br>
    <input type="checkbox" name="check3" value="3">choix numéro 3<br>
    <input type="button"name="but" value="corriger" onclick="reponse(form4)">
</form>
```

Les formulaires
Les cases à cochés (checkbox)

JS :

```
function reponse(form4) {
if ( (form4.check1.checked) == true && (form4.check2.checked) == true &&
(form4.check3.checked) == false)
{

   alert("C'est la bonne réponse! ")

}
else
{

    alert("Désolé, continuez à chercher.")}
}
```

Les formulaires
Liste de sélection (select)

Le contrôle liste de sélection vous permet de proposer diverses options sous la forme d'une liste déroulante dans laquelle l'utilisateur peut cliquer pour faire son choix. ce choix reste alors affiché.

La boite de la liste est crée par la balise <select> et les éléments de la liste par un ou plusieurs tags <option>. La balise </select> termine la liste.

Propriété	Description
name	indique le nom de la liste déroulante.
length	indique le nombre d'éléments de la liste. S'il est indiqué dans le tag <SELECT>, tous les éléments de la liste seront affichés. Si vous ne l'indiquez pas un seul apparaîtra dans la boite de la liste déroulante.
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. C'est lui qui apparaît alors dans la petite boite.

Les formulaires
Liste de sélection (select)

HTML :

Entrez votre choix : <form name="form5">

<select name="list">

<option value="1">elément 1

<option value="2">élément 2

<option value="3">elément 3

</select>

<input type="button"name="b" value="quel est l'élément retenu?" onclick="liste(form5)">
</form>

Les formulaires
Liste de sélection (select)

JS :

```
function liste() {
        alert("L'élément " + (form5.list.selectedIndex + 1));

}
```

Les formulaires
Les contrôles

Les contrôles basiques des éléments entrée dans un formulaire sont extrêmement courant.

Les fonctions les plus utilisées pour le contrôle sont (La liste est bien-sûr non exhaustive) :

string.**includes**([#valeurs que doit contenir la chaine de caractère#])

isNaN([#valeurs que l'on veut vérifié#])

typeof([#valeurs que l'on veut vérifié#])

Les formulaires string.includes

string.**includes**([#valeurs que doit contenir la chaine de caractère#])

- true si le string contient la valeur passé en paramètre (valeur mis entre parenthèse)

```
let sentence = "Hello world !";
let word = "Hello";
if(sentence.includes(word)){
    console.log("It contains");
}
else{
    console.log("It not contains");
}
```