

## R1.02 développement d'interfaces WEB - JS

---

monnerat@u-pec.fr 

13 décembre 2024

IUT de Fontainebleau

Première partie I

# Bases de javascript

Introduction

Survol du langage

Javascript et html

- Intégration du code

- Gestion des événements

- Fonction réflexe

- Modification du css

# Introduction

Une application Web est une applications clients/serveur(s)



On peut la voir en trois couches ...

Front-End

Back-End

SGBD



PostgreSQL



ORACLE®



SGBD



Back-End





# Front-End

```
graph TD; FE[Front-End] --> S[Structure]; FE --> P[Présentation]; FE --> A[Applicatif];
```

The diagram illustrates the components of a Front-End. At the top, a large dashed orange box contains the text "Front-End". Below this box, three dashed orange boxes are arranged horizontally, each containing a component name. Three vertical dashed orange lines with downward-pointing arrowheads connect the bottom edge of the "Front-End" box to the top edge of each of the three component boxes.

Structure

Présentation

Applicatif

# Front-End



Présentation

Applicatif

# Front-End



Applicatif

# Front-End



Javascript est même utilisé coté serveur (Back-end) avec



- Les bases de JS reposent sur ECMAScript




spécifié et standardisé par ECMA ("European Computer Manufacturers Association").

- Dernière version : ECMA-262 Edition 6 (17 juin 2015)

<http://www.ecma-international.org/ecma-262/6.0/index.html> 

## Documentation

<https://developer.mozilla.org/fr/docs/Web/JavaScript> 

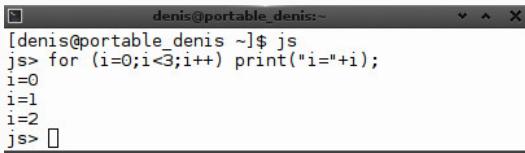
<http://www.w3schools.com/> 

Autant de moteurs javascript qu'il y a de navigateurs :

- SpiderMonkey pour firefox (écrit en C).
- JavaScriptCore pour safari (écrit en C++).
- V8 JavaScript engine pour chrome et chromium (écrit en C++).
- Carakan pour Opéra.
- Chakra pour internet explorer.



## Moteur spiderMonkey (mode console)



```
denis@portable_denis:~  
[denis@portable_denis ~]$ js  
js> for (i=0;i<3;i++) print("i="+i);  
i=0  
i=1  
i=2  
js> █
```



Domaine d'application : initialement le web ...

- Interface graphique des applications web.
- Validation des données.
- Modification dynamique de l'arbre du document. (DOM)
- googlemaps

Mais Javascript est aussi le langage de développement sur certains OS où les applications sont des applications HTML5 au sens large.



Beaucoup d'apis ....

- Ajax (XMLHttpRequest) : requête http asynchrone.
- WebWorkers : programmation parallèle.
- WebSockets : communication bidirectionnelle entre le client et le serveur.
- Canvas 2D Context : dessin avec la balise canvas.

... de bibliothèques et framework

- JQuery
- Mootools, Angular.js, Prototype.js, Dojo, React, Ember, Backbone, etc.
- Pixi, Phaser (jeux)

Il existe **deux façons** d'interagir avec une page html à partir de javascript :

## DHTML (historique)

- Nouveaux attributs pour la gestion événementielle (onClick, etc ....)
- Accès aux éléments par les objets prédéfinis (document.forms[0].elements[1])

## DOM

- La page est un arbre. On peut accéder aux noeuds, les modifier, etc ...
- Gestionnaire d'événements intégré à DOM.
- Modèle générique.

# Survol du langage

## Javascript

C'est un langage à objets utilisant le concept de prototype, disposant d'un typage faible et dynamique qui permet de programmer suivant plusieurs paradigmes de programmation : fonctionnelle, impérative et orientée objet.

Javascript permet, entre autres :

- Programmation événementielle. ex : changer une image au survol du pointeur.
- Effectuer des calculs. ex : lire la valeur saisie, la multiplier par 3.14 et afficher le résultat.
- Modifier la présentation et/ou le contenu du document html.
- Effectuer des requêtes HTTP.
- Accéder à des bases de données locales, etc.

Au niveau du langage, on distingue :

### **Le noyau, qui comporte**

- des objets prédéfinis (tableaux, etc...),
- des opérateurs,
- des structures algorithmiques.

### **La couche navigateur**

un ensemble d'objets associés au navigateur.

- fenêtres,
- formulaires,
- images, etc.

La suite n'est aucunement un cours complet sur Javascript, à peine un survol avec beaucoup d'oublis (volontaires).



- Opérateurs, boucles, conditionnelles similaires au C.
- **Mais**, variables typées dynamiquement
- **;** comme en C marque la fin d'une instruction. (optionnel)
- Mot clé **var**, **let**, **const** pour déclarer une variable. (optionnel)

```
for (let count=0;count<5,count++){
  print("valeur="+count);
}
let count=0;
const FINI=false;
while(!FINI){
  if (count>4){
    fini=true;
  }else{
    print("valeur="+count);
  }
  count++;
}

var count=0;
do{
  print("valeur="+count);
}while(count<5);
```



# Types simples

- Booléens (Boolean).
- Nombres (Number).
- Valeurs prédéfinies
  - true
  - false
  - undefined
  - Infinity
  - NaN

```
js> var x=2;
js> var y=3;
js> var z=x+y;
js> z;
5
js> z==(x+y);
true
js> (x==y);
false
```

```
js> z++;
5
js> ++z;
7
js> x/3;
0.6666666666666666
js> x>y? 0:1;
1
js> 128<<1;
256
js> 255|128;
255
js> 255&128
128
js> 0x23
35
js> 034
28
js> 5=="5"
true
js> 5==="5"
false
js>
```

# Arrays (les tableaux)

- Numérique ou associatif.
- Éléments de type hétérogène.
- Syntaxe : `a[1]=1` pour `[0,1,true]`.
- Propriété `length` pour la taille : `a.length` vaut 3.
- N'existe pas pour les tableaux associatifs.
- Beaucoup de méthodes (notamment `forEach`).
- Itération avec `for(let element of array){}`

```
js> var tab=[1,2,3];
js> var tab2=tab;
js> tab.push(5);
4
js> tab;
[1, 2, 3, 5]
js> tab.pop();
5
js> tab.unshift(5);
4
js> tab;
[5, 1, 2, 3]
js> tab.reverse();
[3, 2, 1, 5]
js> tab2;
[3, 2, 1, 5]
js> tab.sort();
[1, 2, 3, 5]
js>tab.splice(1,2,6,7);
[2, 3]
js>tab;
[1, 6, 7, 5]
js>tab.slice(1,3);
[6,7]
```

- Type String.
  - Beaucoup de méthodes.
  - Littéraux : 'val' ou "val".
- Expressions régulières
  - Support des expressions régulières classiques (grep unix).
  - littéraux : /pattern/flags.

```
js> 'wim web'.indexOf('i');
1
js> 'wim web'.lastIndexOf('w');
4
js> 'wim web'+ ' semestrel';
"wim web semestrel"
js> 'wim web'.split(" ");
["wim", "web"]
js> 'wim web'.substr(1,4);
"im w"
js> 'wim web'.slice(1,4);
"im "
js> 'wim web'.toLocaleUpperCase();
"WIM WEB"
js> 'wim web'[2];
"m"
js> /^w.+/.test("wim web");
true
js> "j'aime bien javascrit".match(/S+/g);
["j'aime", "bien", "javascrit"]
js>
```

- Comme les structures en C : regroupent des couples (clé,valeur).
- peuvent être vus comme des tableaux indicés.
- utilisés pour les tableaux associatifs.
- Syntaxe dédiée : JSON.

```
js> var obj={un:1,deux:2,trois:3,verite:true};
js> obj.deux;
2
js> obj['deux']
2
js> var obj2=obj;
js> obj.newprop="coucou";
"coucou"
js> obj2;
({un:1, deux:2, trois:3, verite:true,
  newprop:"coucou"})
js> obj2;
({un:1, deux:2, trois:3, verite:true,
  newprop:"coucou"})
js> obj['tab']=[true,1];
[true, 1]
js> obj
({un:1, deux:2, trois:3, verite:true,
  newprop:"coucou", tab:[true, 1]})
js> "trois" in obj;
true
js>obj.tab[0];
true
js>obj.tab[1];
```

- js fournit une "classe" prédéfinie Date.
- timestamp.
- Plusieurs constructeurs.
- Beaucoup de méthodes.

```
> var maintenant=new Date();
js> maintenant;
(new Date(1386414882694))
js> maintenant.toString()
"Sat Dec 07 2013 12:14:42 GMT+0100 (CET)"
js> var demain=new Date(2013,12,8,10,15,10)
(new Date(1389172510000))
js> demain.toString()
"Wed Jan 08 2014 10:15:10 GMT+0100 (CET)"
> demain.getDate();
8
js> demain.getDay();
0
js> demain.getMonth();
11
js> demain.getFullYear();
2013
js> demain.getMinutes();
15
js>demain.toString();
"Sun Dec 08 2013"
```

# L'objet Math

- L'objet global Math fournit beaucoup de fonctions et constantes mathématiques.
- `Math.abs`, `Math.ceil`, `Math.round`, `Math.log`, `Math.sin`, `Math.sqrt`, `Math.random`, etc.
- `Math.PI`, `Math.E`, `Math.SQRT2`, etc.

```
js> Math.PI;
3.141592653589793
js> Math.E;
2.718281828459045
js> Math.sin(Math.PI/2);
1
js> var rnd=Math.random();
js> rnd;
0.5382747752890101
js> Math.pow(Math.cos(rnd),2)+
  Math.pow(Math.sin(rnd),2);
0.9999999999999999
js> Math.max(10,2,3,-5,100,8)
100
js> Math.sqrt(64)
8
js> Math.log(2) == Math.LN2;
true
```

# Les fonctions

- `function` est un type du langage. (une "classe")
- On peut passer des fonctions en arguments d'autres fonctions.
- Des fonctions peuvent renvoyer des fonctions.
- Fonctions anonymes.
- Portée d'une variable définie uniquement par les blocks fonctionnelles.
- Des fonctions peuvent être des membres d'objets.

```
js>function f(a) {return 2 + a;};
js>typeof f;
"function"
js>f(3);
5
js>f("toto");
"2toto"
js>e=function somme(a,b){return a+b;};
js>e(3,6);
9
js>function fun(a,op,b)
  {
    return op(a,b);
  };
js>fun(10,somme,30);
40
js>fun(10,function(x,y){return x-y;},20);
js>-10
js>(function(x){return x+2;})(3)
5
```

# Javascript et html



# A quoi ça sert ?

Javascript est devenu le langage de script, interprété par tous les navigateurs, qui permet entre autres :

- Modifier dynamiquement la présentation (le css) de la page.
- Modifier dynamiquement la structure du document html.
- Réagir aux événements utilisateurs.
- etc

Un sur-ensemble du langage est disponible, avec des objets prédéfinis spécifiques au web :

## Attributs

- fenêtres,
- écran,
- formulaires,
- images, etc.

# Javascript et html

Intégration du code

# Exécution de code

On peut (faire) exécuter du code javascript de plusieurs manières.



1. Avec la balise `<script>`.
  - déclarations dans l'en-tête (`<head>...</head>`), souvent, mais possibilité d'insertion dans le body.
  - appel de fonction ou exécution de javascript dans le corps (`<body>...</body>`).
  - insertion de fichiers javascript "externes".
2. Clic sur un lien : `<a href="javascript:...">ici</a>`.
3. En réponse à un événement sur un élément html : attribut `onEvenement`. Le code javascript est exécuté lors de l'événement correspondant (DHTML).

# Inclusion : dans le code html

```
<html>
  <head>
    <script language="javascript">
      function fin(){
        alert('Bye');
      }
    </script>
  </head>
  <body>
    <script language="javascript">
      document.write('Pour se dire aurevoir');
      /* inutile de passer par js (pourquoi ?)*/
    </script>
    <br /><a href="javascript:fin();">cliquez ici</a>
    <br />ou passez la souris sur
    <a href="" onmouseover="fin();">ce lien</a>
  </body>
</html>
```

## Inclusion : depuis des fichiers externes

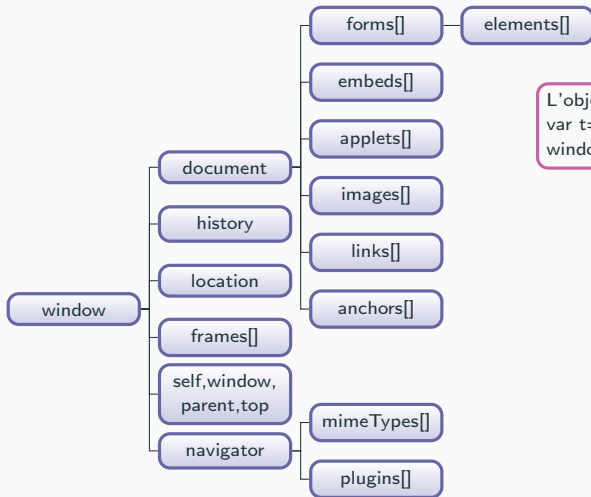
```
<html>
  <head>
    <script language="javascript" src="fin.js"></script>
  </head>
  <body>
    Pour se dire aurevoir
    <br /><a href="javascript:fin();">cliquez ici</a>
    <br />ou passez la souris sur
    <a href="" onMouseOver="fin();">ce lien</a>
  </body>
</html>
```

```
function fin(){ /* fichier fin.js */
  alert('Bye');
}
```

Objets instanciés au démarrage du navigateur. Ils permettent d'accéder à des informations concernant le navigateur, les documents html affichés, l'écran de la machine.

- **Navigator**
  - une seule instance, l'objet `navigator`.
  - infos sur le nom, version, plugins installés.
- **Window**
  - une instance par fenêtre et frame du document html.
  - accès à tous les objets créés par les balises html.
- **Screen**
  - une instance, l'objet `screen`
  - infos sur la largeur et hauteur en pixels, couleurs disponibles, etc...

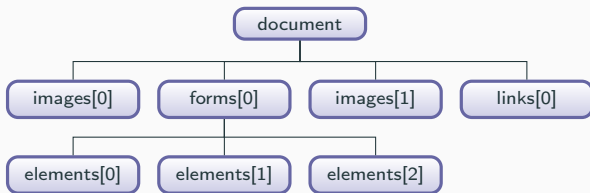
# Hierarchie des objets du navigateur



L'objet `window` est la racine.  
`var t=10 ; // variable nouvelle`  
`window.t=10 ;//idem`

# Accès aux éléments du document html (version dhtml)

Un champ de saisie est contenu dans un formulaire contenu dans le document



On peut accéder à un élément d'un de ces tableaux avec l'attribut name de la balise html.

```
<script>
document
  .forms.formulaire
  .elements.adresse.value="???";
document.forms[0].elements[0].value="???";
</script>
<form name="formulaire">
  <input type="texte" name="adresse">
</form>
```





Pour les plus hardis, il est possible de récupérer n'importe quel noeud (node) du document html. En particulier, si cet élément est identifié (attribut id). Cela renvoie à la représentation DOM du document.

```
<div id="container">  
  <h1>ceci est une division</h1>  
  <p>et cela un paragraphe</p>  
</div>
```

```
let mondiv = document.getElementById("container")  
mondiv.style.backgroundColor = "#fea360"  
let p = document.querySelector("p")
```

# **Javascript et html**

## **Gestion des événements**

onEvenement="Action\_Javascript\_ou\_Fonction() ;

ou en assignant la propriété correspondante de l'élément  
elm.onclick=function;

```
<!DOCTYPE html>
<html>
  <body>
    <button type="button" onclick="alert('Hello world!')">
      Click Me!
    </button>
  </body>
</html>
```

<code>onAbort</code>	<b>en cas d'interruption</b>
<code>onBlur</code>	<b>en quittant</b>
<code>onChange</code>	<b>après modification réussie</b>
<code>onClick</code>	<b>en cliquant</b>
<code>onDbClick</code>	<b>en double-cliquant</b>
<code>onError</code>	<b>en cas d'erreur</b>
<code>onFocus</code>	<b>en activant</b>
<code>onKeyDown</code>	<b>en appuyant sur une touche</b>
<code>onKeyPress</code>	<b>en maintenant une touche appuyée</b>
<code>onKeyUp</code>	<b>en relâchant la touche</b>
<code>onLoad</code>	<b>en chargeant le document</b>

<code>onMouseDown</code>	<b>en maintenant la touche de souris appuyée</b>
<code>onMouseMove</code>	<b>en bougeant la souris</b>
<code>onMouseout</code>	<b>en quittant l'élément avec la souris</b>
<code>onMouseover</code>	<b>en passant sur l'élément avec la souris</b>
<code>onMouseUp</code>	<b>en relâchant la touche de souris</b>
<code>onReset</code>	<b>en initialisant le formulaire</b>
<code>onSelect</code>	<b>en sélectionnant du texte</b>
<code>onSubmit</code>	<b>en envoyant le formulaire</b>
<code>onUnload</code>	<b>en quittant le fichier</b>
<code>javascript:</code>	<b>pour les liens</b>

- Consulter la doc pour la sensibilité des différentes balises.
- Pour la plupart des événements, le navigateur possède déjà un comportement par défaut.
- Quand un événement est intercepté, le navigateur exécute d'abord le traitement associé par le gestionnaire, puis celui par défaut.
- Pour empêcher le traitement par défaut, il faut retourner `false`.

```
<input type="text" value="" onKeyPress="return false;">
```



Pour être certain d'exécuter du javascript après que **tout** le document html est prêt :

- utiliser l'événement `onLoad` du `body`.
- placer le code dans une balise `script` à la fin du fichier `html`.
- utiliser l'attribut `defer` lors du chargement.

# Javascript et html

Fonction réflexe



# Sensibilisation dans un attribut html

Le code exécuté en réponse à un événement s'appelle un callback, où une fonction réflexe. Il est possible de transmettre des paramètres à cette fonction depuis l'html.

En particulier :

- l'élément html qui en est le siège avec le mot clé `this`
- l'objet `event` qui représente l'événement lui-même, et qui regroupe des renseignements contextuels : coordonnées de la souris, touche pressée, etc ...
- on peut mettre ce que l'on veut.

```
<!DOCTYPE html>
<html>
  <head>
    <script type="text/javascript">
      function ok(t,e,m){
        alert(t+": "+e+": "+m);
      }
    </script>
  </head>
  <body>
    <button type="button"
      onclick="ok(this,event,'ok')">
      Click Me!
    </button>
  </body>
</html>
```

Quelques propriétés de l'objet event (il y a différents types d'événements : clavier, souris, etc ...) :

- type : type.
- target ou srcElement : siège.
- which ou keyCode : code de la touche.
- which ou button : bouton souris.
- screenX|Y,clientX|Y,x|y,pageX|Y,layerX|Y,offsetX|Y : position souris.

```
function clavier(el,ev)
{
  alert("element "+el);
  alert("touche pressee "+String.fromCharCode(e.which));
}
```

```
<input type="text" value="" onKeyPress="clavier(this,event);">
```

# Sensibilisation avec le DOM HTML

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
    <button id='mybtn' type="button">
      Click Me!
    </button>
    <p id="date"></p>
  </body>
</html>
<script>
document.getElementById('toto').onclick=function(){
  document.getElementById('date').innerHTML=Date();
}
</script>
```

**Attention** la fonction réflexe ne peut recevoir qu'un argument : l'événement lui-même.

# Javascript et html

Modification du css

# Modification CSS

Chaque objet possède les attributs :

- `className`

```
document.forms.monformulaire.elements.nom.className="red";
```

- `classList`

```
el.classList.toggle("hidden");
```

- L'attribut `style`

```
el.style.backgroundColor = "#ff0";  
el.style.color = "#0f0";  
el.style.fontSize = "1.2em";  
el.style.textAlign = "center";
```

Remarque : lorsque la propriété css comporte dans son nom un tiret (background-color), celui-ci est supprimé en javascript, et la lettre d'après est en majuscule.