Présentation du Langage JavaScript (ES5)

par Stéphane MASCARON Architecte Logiciels Libres
Formation GRETA Sud Aquitaine.
FMS - Année 2018

dd

Le Langage JavaScript (ES5)

- **Histoire de JavaScript**
  - Le langage Javascript a été mis au point par Netscape en 1995. En décembre 1995, Sun et Netscape annoncent la sortie de JavaScript.
  - En mars 1996, Netscape met en œuvre le moteur JavaScript dans son navigateur Web Netscape Navigator 2.0.
  - Microsoft réagit alors en développant JScript, qu'il inclut ensuite dans Internet Explorer 3.0 en août 1996
  - Actuellement, la version du langage supporté par les navigateur est le Javascript 1.5.
  - La version 1.6 (EC6) n'est pas encore supportée.

Le Langage JavaScript (ES5)

- **Histoire (suite) :**
  - Le nom du programmeur de JavaScript est Brendan EICH.
  - Netscape soumet alors JavaScript à Ecma International 1 pour la standardisation. Les travaux débutent en novembre 1996, et se terminent en juin 1997 par l'adoption du nouveau standard ECMAScript.
  - Les spécifications sont rédigées dans le document Standard ECMA-262

Le Langage JavaScript (ES5)

- **Histoire (suite) :**
  - Le choix du nom JavaScript et un communiqué de presse commun de Netscape et Sun Microsystems, daté du 4 décembre 1995, qui le décrit comme un complément à Java, ont contribué à créer auprès du public une certaine confusion entre les deux langages.
  - Java != JavaScript

Le Langage JavaScript (ES5)

- **Vue d'ensemble du langage :**
  - JavaScript (ou plutôt ECMAScript, qui constitue le noyau commun de tous les langages qui se conforment au standard) est orienté objet.
    - Les fonctionnalités de base du langage et l'accès au système hôte sont fournis par des objets (window, navigator, etc.)
    - Un programme est un regroupement d'objets communicants.
    - Un objet est une collection de propriétés
    - Les propriétés sont des conteneurs qui contiennent des autres objets, des valeurs primitives, ou des fonctions.

Le Langage JavaScript (ES5)

- **Vue d'ensemble du langage (suite):**
  - Une valeur primitive est un élément de l'un des types intégrés : undefined, null, Boolean, Number et String.
  - Un objet est un élément du type intégré restant Object.
  - Une fonction est un objet appelable.
  - Une fonction qui est associée à un objet via une propriété est une méthode.

Le Langage JavaScript (ES5)

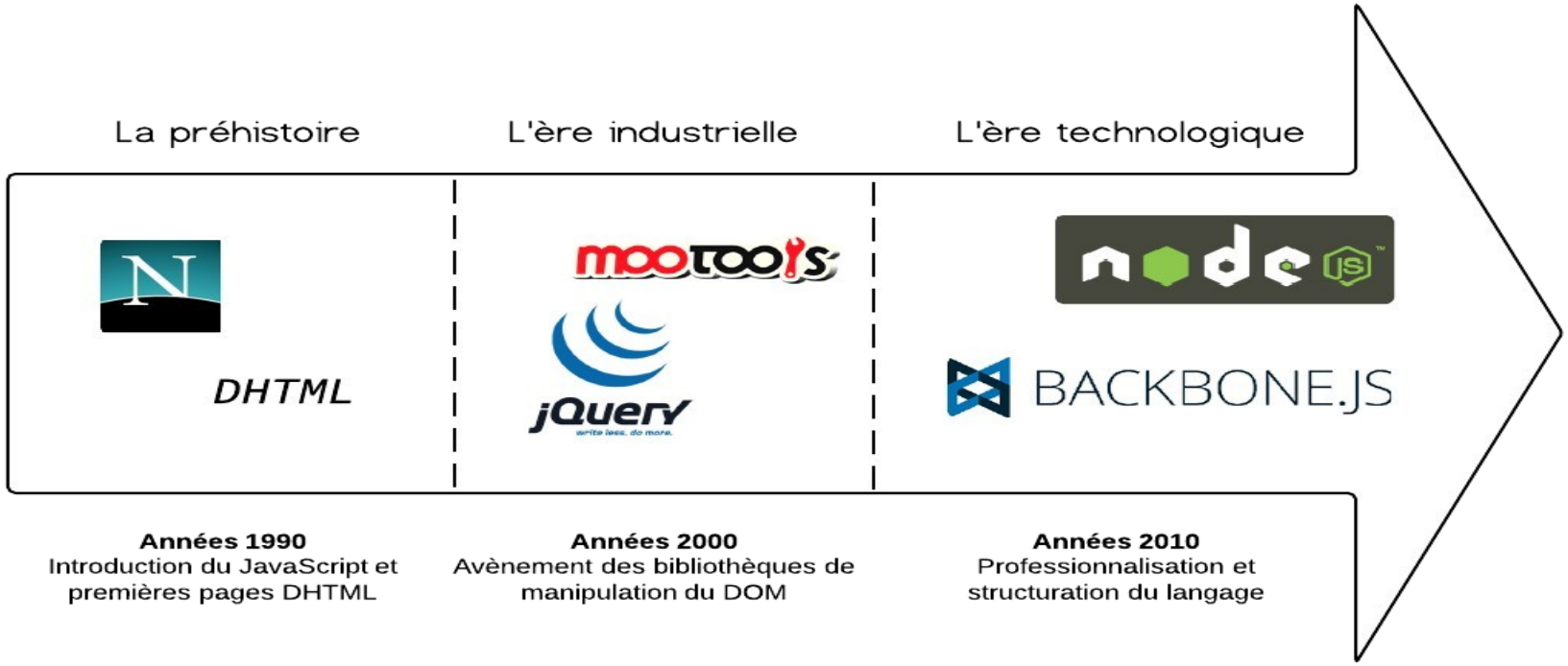
- **Vue d'ensemble du langage (suite) :**
  - ECMAScript définit une collection d'objets intégrés qui complètent la définition des entités du langage.
  - Ces objets intégrés comprennent l'objet global, et les objets suivants :
    - Object, Function, Array, String, Boolean, Number, Math, Date, RegExp, JSON
    - un certain nombre d'objets erreur : Erreur, EvalError, RangeError, ReferenceError, SyntaxError, TypeError et URIError.

Le Langage JavaScript (ES5)

- **Vue d'ensemble du langage (suite) :**
  - ECMAScript définit également un ensemble d'opérateurs, tels que les opérateurs arithmétiques, les opérateurs de décalage au niveau du bit, les opérateurs relationnels, les opérateurs logiques, etc.
  - La syntaxe du langage ressemble volontairement à la syntaxe du langage Java :
    - Cependant, elle est moins stricte, afin de rendre l'outil plus facile à utiliser.
    - Par exemple, il n'est pas obligatoire de déclarer le type d'une variable avant de l'utiliser

Le Langage JavaScript (ES5)

- **Frise historique** :
  - L'avènement de moteurs JavaScript comme V8 de google font de ce langage une solution viable et sûre pour les applications Web et le développement côté serveur.



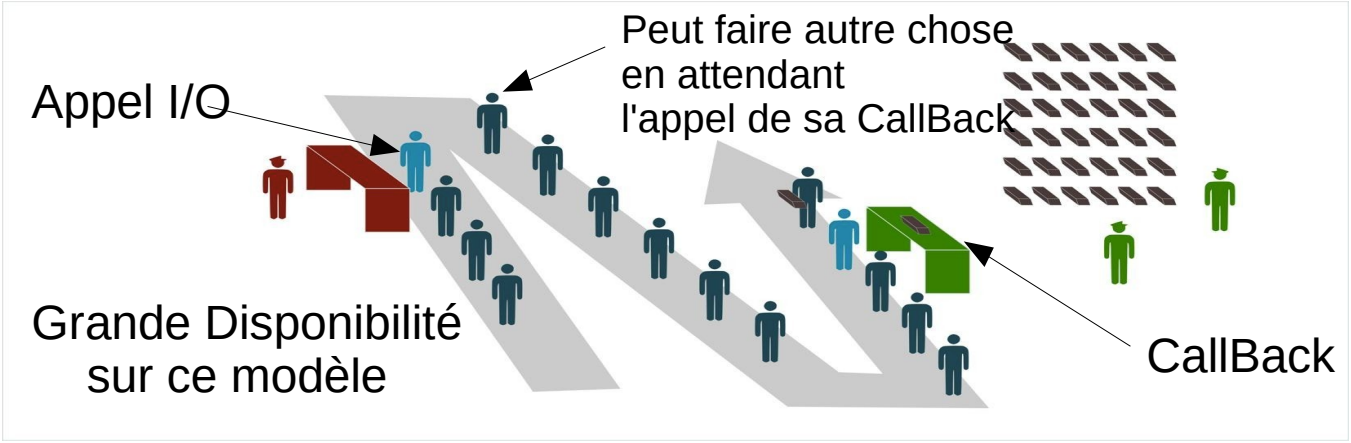
Le Langage JavaScript (ES5)

- **Où trouve-t-on du JavaScript ?**
  - Dans les applications Web, les sites Internet
  - Dans les téléphones portable avec les applications hybrides (Apache Cordova).
  - Das les applications côté serveur :
    - Node.js, IO.js (récemment Microsoft a demandé l'intégration de CHAKRA son moteur JS comme moteur de Node.js. La communauté a commencé le développement.(voir ici)
  - Dans des micro-contrôleurs comme « Espruino »

Le Langage JavaScript (ES5)

- **Particularité de JavaScript lors de l'exécution d'un programme**
  - Mono-thread d'exécution
    - Comment fonctionne Java ! (exemple Tomcat)

    - Fonctionnement de JavaScript :



Le Langage JavaScript (ES5)

- **Callback**
  - Vous allez entendre parler souvent de ce type de fonction. La meilleur façon de comprendre ce que c'est est de voir un exemple simple dans JavaScript.
    - Les fonction setTimeout() et setInterval() contiennent une fonction de callback

```
setInterval(function() {
    window.console.log("Hello from JavaScript");
}, 3000);
```

Le Langage JavaScript (ES5)

- **Test de setInterval(), code HTML :**

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <title>PREMIERS PAS AVEC BRACKETS</title>
        <script src="main.js"></script>
        <script>
            function init() {
                hello();
            }
        </script>
        <meta name="description" content="An interactive getting started
guide for Brackets.">
        <link rel="stylesheet" href="main.css">
    </head>
    <body onload="init();">
(…)
```

Le Langage JavaScript (ES5)

- **Test de setInterval(), code JavaScript :**

```
function hello() {
    setInterval(function () {
        console.log('toute les 3 secondes');
    }, 3000)
}
```

La déclaration de la fonction en rouge est une callback, comme JavaScript n'a qu'un Thread d'éxécution, pour exécuter le code toute les 3 secondes, on donne au moteur JS en paramètre l'adresse d'une fonction en paramètre à setInterval() et le moteur JS exécutera cette fonction chaque fois que 3 secondes seront passées. Nous reverrons plus en détail les subtilités des callbacks.

Le Langage JavaScript (ES5)

- **Généralité**
  - Propriétés :
    - langage de script interprété
    - orienté objet, faiblement typé
    - généralement exécuté dans un navigateur (maintenant côté serveur avec NODE.JS-V8)
  - Limitations :
    - pas d’entrées/sorties de fichiers (pour la sécurité)
    - développement difficile (erreurs silencieuses)
    - évolution entre les versions (pas de fonctions graphiques avant HTML5)

Le Langage JavaScript (ES5)

- **Syntaxe**
  - Instructions :
    - syntaxe similaire à C/C++/Java/C#
    - différences majuscules/minuscules (case sensitive)
    - noms de variables ou fonctions (identifiants) :
      - lettres, ’_’, ’$’ ou chiffre (sauf en début)
      - les mots-clés du langage sont réservés
    - instructions séparées par ’;’ (optionnel, mais à mettre! Pour pouvoir minifier le code)
    - commentaires de fin de ligne // ou multiligne /* ...*/

Le Langage JavaScript (ES5)

- **Variables et types**
  - Variables :
    - déclaration: var n=10;
    - non typé (en fait typage automatique et dynamique)
    - 2 niveaux:
      - global (déclaration optionnelle)

        ou
      - local à la fonction

Le Langage JavaScript (ES5)

- **Variables et types**
  - Types primitifs et littéraux :
    - **nombre**: décimal (ex. : 12.4) ou entier (ex. : 10)
    - **chaîne de caractères**: entre "guillements" ou ’apostrophes’
    - **booléen**: true et false
    - **null**: aucune valeur (différent de **undefined**!)

Le Langage JavaScript (ES5)

- **Objets**
  - Type de donées composites, mais sans déclaration (pas de classe)
  - Création d'un objet vide par l'opérateur 'new'

    ```
    var o = new Object();
    ```
  - Propriétés (= nom et valeur) accessibles par l'opérateur '.'

    ```
    o.x = 1; o.y = 2;
    ```
  - Vue alternative comme tableau associatif

    ```
    o["x"] = 1; o["y"] = 2;
    ```
  - déclaration en littéral

    ```
    var o = {x:1, y:2};
    ```
  - Classes d'objets définies dans le langage **String**, **Math**, **Array**, **Boolean**, **Date**, **Number**, **Function**. . .

Le Langage JavaScript (ES5)

- **Tableaux**
  - Les tableaux sont des objets, des objet aux éléments numérotés, accessibles par l'opérateur '**[]**'

    `a[0] = 1; a[1] = 2;`
  - Création par le constructeur **Array()**
    - Vide: **`var a = new Array();`**
    - Taille: **`var a = new Array(3);`**
    - Contenu: var a = **`new Array(1,"contenu", false);`**
  - indice initial: 0
  - Contenu composite (type primitifs ou objets, mélangés)
  - Taille du tableau : **`propriété a.length`**
  - déclaration d'un tableau littéral :

    `var a = [1, "contenu", false];`

Le Langage JavaScript (ES5)

- **Fonctions**
  - Déclaration par la mot-clef **function** :

```
function ajoute(x,y) {
   return x+y;
}
```

  - Renvoi le résultat par **return**
  - Appel de fonction

```
var s = ajoute(1,2);
```

  - Déclaration d'une fonction littérale :

```
var ajoute = function (x, y) {
   return x+y;
}
```

  - **Méthode**: fonction définie comme propriété d'un objet
  - Toujours déclarer les variables locales avec '**var**' (sinon on travaille avec des variables globales)

Le Langage JavaScript (ES5)

- **Opérateurs :**
  - Arithmétiques: **`+ - * / % (modulo)`**
  - Concaténation: **`+ (chaînes de caractères)`**
  - Affectation: **`= += *= -= . . .`**
  - Égalité: **`==`** **`!=`** (différent - y compris chaînes.)
  - Identité: **`===`** **`!===`** (égalité + type ident.)
  - Ordre: **`< <= >= >`** (nombre ou chaîne)
  - Logique: **`&&`** **`||`** **`!`**
  - Binaire: **`&`** (et) **`|`** (or) **`^`** (xor) **`~`** (not)

Le Langage JavaScript (ES5)

- **Conditions :**
  - Alternative (SI … ALORS … SINON)

```
if (choix == "oui") {
...
} else {
...
}
```

  - Branchement multiple

```
switch (nom) {
   case "Andreas":
      rep="Guten tag"; break;
   case "John":
      rep="Hello"; break;
   default:
      rep="Bonjour";
}
```

Le Langage JavaScript (ES5)

- **Boucles :**
  - Répétition avec incrément

```
for (var i=0; i<10; i++) {    // for(;;) = while(true)
   j += i;
}
```

  - Enumération des propriétés d'un objet

```
for (var i in window) {
   msg += i + " ";
}
```

  - Tant que ...

```
while (i<10) { ... }
```

  - Faire ... tant que

```
do { ... } while (a == "");
```

Le Langage JavaScript (ES5)

- **Ruptures de séquences**
  - Break

```
for (i=0; i<10; i++) {
   if (a[i] > 0) break;
}
```

  - Continue

```
for (i=0; i<10; i++) {
   if (a[i] == 0) continue;
...
```

  - Etiquettes (permettent des ruptures dans le cas de boucles imbriquées)

```
boucle1: for (i=0; i<10; i++) {
   for (j=0; j<10; j++)
      if (...) break boucle1;
   ...
```

Le Langage JavaScript (ES5)

- **Opérateurs spéciaux**
  - **`new`** : création d'un objet
  - **`delete`** : supprime une propriété d'un objet
  - **`typeof`** : renvoie le type de l'opérande ("**number**", "**string**", "**boolean**", "**object**", "**function**", "**undefined**") sous forme d'une chaîne de caractères.
  - **`in`** : vrai si la propriété appartient à l'objet
  - **`void`** : renvoi **`undefined`**
  - Exemple

```
if (typeof a == "object" && test in a) {
   delete a.test
}
```

Le Langage JavaScript (ES5)

- **Classe Standards**
  - Object :
    - Constructeur
      - **`new Object()`** : création d'un objet vide
    - Propriétés
      - **`constructor`** : fonction constructeur
    - Quelques méthodes d'Object
      - **`toString()`** : convertit en chaîne (à redéfinir)
      - **`valueOf()`** : rend l'objet ou la valeur de l'objet (si défini)
      - **`create()`** : Crée un nouvel objet avec un prototype donnée et des propriétés données.
      - **`getOwnPropertyNames()`** : Renvoie un tableau contenant les noms de toutes les propriétés propres à l'objet qui sont énumérables et non-énumérables.

Le Langage JavaScript (ES5)

- **Classe Standards**
  - Number, Boolean et String :

    Sont les **classes enveloppes des type primaires** correspondants avec **création et destruction automatique** des objets-enveloppe.
  - Constructeur :
    - **`new Number(valeur)`** ;
    - **`new Boolean(valeur)`** ; **`new String(valeur)`**: création explicite d'un objet.
  - Conversion :
    - **`Number(valeur)`**: convertit la valeur (ou chaîne) en nombre
    - **`Boolean(valeur)`**: toutes les valeurs converties à true à part

      0, NaN, null, undefined, ""
    - String(valeur): convertit la valeur en chaîne de caractères

      `s=String(3);  // '3'`

Le Langage JavaScript (ES5)

- **Objet Math**
  - **Constantes**
    - **`Math.PI`** : π
    - **`Math.E`** : base des logarithmes e
    - …
  - **Fonctions**
    - **`Math.abs(x)`** : valeur absolue
    - **`Math.sin(x)`,`cos(x)`,`tan(x)`**… fonctions trigonométriques
    - **`Math.ceil(x)`,`floor(x)`,`round(x)`** arrondi au-dessus, au-dessous, au plus proche
    - **`Math.sqrt(x)`,`log(x)`,`exp(x)`,`pow(x,y)`** racine, logarithme et puissance
    - **`Math.min(...)`,`max(...)`** plus petite ou plus grande valeur
    - **`Math.random()`** nombre aléatoire entre 0.0 et 1.0
    - …

Le Langage JavaScript (ES5)

- **Classe Array**
  - Constructeur
    - **`new Array(n)`** : création d'un tableau de taille n (facultatif: agrandissement automatique du tableau)
    - **`new Array(valeurs...)`**: création d'un tableau initialisé
  - Propriétés
    - **`length`**: taille du tableau
  - Méthodes
    - **`join(separateur)`**: concatène les éléments
    - **`slice(debut,fin)`**: rend un sous-tableau
    - **`sort(fonctionCmp)`**: trie le tableau
    - **`push(valeur...), pop()`:** gestion en pile
    - **`shift(),unshift(valeur...)`**: gestion en filevaleur...),
    - ...

Le Langage JavaScript (ES5)

- **Avec le HTML**
  - La balise '**script**'
    - Référence à un fichier JavaScript externe (préférable)

```
<script type="text/javascript" src="js/code.js"> </script>
```

    - Dans l'entête HTML pour un pré-chargement du JS
    - Il faut protéger le code JavaScript intégré à l'intérieur d'une section de données 'CDATA' (XHTML, XSLT)
  - Dans un gestionnaire d'événement

```
<input type="button" value="Oui" onclick="zoom()">
```

  - Comme pseudo- URL

```
<a href="javascript: void test()">
```

Le Langage JavaScript (ES5)

- **Exemple de JavaScript intégré :**

```
<html>
   <head>
   <script>
      function fait() {
         document.getElementById("ici").innerHTML =
         "Merci";
      }
   </script>
   </head>
   <body>
      <button id="ici" onclick="fait()">Cliquer
      ici</button>
   </body>
</html>
```

Le Langage JavaScript (ES5)

- **Accès aux balises**
  - Recherche par identifiant
    - HTML: **`<h1 id="titre1">La lune</h1>`**
    - JavaScript: **`document.getElementById("titre1")`**
    - Retourne un objet dont on peut lire ou modifier les propriétés (voir aussi **querySelector**(…)) et **querySelectorAll**(…) ;
  - Recherche par type de balise
    - **`document.getElementsByTagName("h1")`**
    - retourne un tableau de nœuds du DOM (node)

Le Langage JavaScript (ES5)

- **Modifier le document HTML**
  - Génération dynamique de code
    - dans le corps du document:

```
<script>
   document.write('<h1>'+titre+'</h1>');
</script>
```

    - exécuté au cours du chargement de la page
  - Modification du texte
    - HTML: `<div id="msg">Message</div>`
    - JavaScript:

```
document.getElementById("msg").innerHTML= "OK";
```

Le Langage JavaScript (ES5)

- **Le DOM**
  - Représentation du document en mémoire
    - Le document (X)HTML est structuré de manière arborescente.
    - Chaque noeud de cet arbre correspond aux balises de début et fin d'un élément.
    - JavaScript permet d'explorer et modifier la structure du document en mémoire par l'intermédiaire d'objets de type Node représentants les noeuds.
  - Exemple (déjà vu) d'accès à un nœud :

    HTML : **`<img id="dessin">`**

    JavaScript: `var node = document.getElementById("dessin");`

Le Langage JavaScript (ES5)

- **La classe ’Node’**
  - **Propriétés**
    - **`nodeType`**: document, élément, texte, commentaire...
    - **`nodeName`**: nom de l’élément (par ex. img)
    - **`nodeValue`**: valeur du contenu d’un élément texte
    - **`attributes[]`** tableau (en lecture) des attributs d’un élément
    - **`childNodes[]`** tableau (en lecture) des noeuds enfants
    - **`firstChild`**: premier enfant du noeud
    - **`lastChild`**: dernier enfant du noeud
    - **`parentNode`**: noeud-père
    - **`ownerDocument`**: document dont dépend le noeud

Le Langage JavaScript (ES5)

- **La classe ’Node’ (2)**
  - **Méthodes**
    - **`appendChild(node)`**: insère un noeud dans l’élément courant.
    - **`removeChild(node)`**: retire un noeud des fils de l’élément courant (le noeud n’est pas détruit)
    - **`cloneNode(bool)`**: clone le noeud courant (bool : de façon récursive ou pas)
    - Les noeuds qui représentent les éléments disposent en plus de propriétés du nom des attributs de la balise:

```
n = document.getElementById("dessin");
n.src = "new.png"; n.style.left = "10px";
```

Le Langage JavaScript (ES5)

- **La classe ’Document’**
  - La classe Document hérite de la classe Node et représente le nœud racine de l’arborescence d’un document
  - **Méthodes**
    - **`createElement(nom)`**: crée un élément avec le nom de balise
    - **`addEventListener(type,callback,bool)`**: ajouter un gestionnaire d’événement au document
    - **`getElementById(id)`**: noeud d’attribut id correspondant à la valeur ou null
    - **`getElementsByTagName(nom)`**: tableau d’éléments du nom spécifié

Le Langage JavaScript (ES5)

- **Le document HTML**
  - La classe **HTMLDocument** hérite de la classe... Document
  - Propriétés pour un document HTML
    - **`body`**: raccourci vers le noeud du corps du document HTML
    - **`cookie`**: associés au document
    - **`title`**: titre du document
    - **`URL`**: adresse du document

NB : Par commodité, la variable document représente la racine du document HTML courant en JavaScript:

```
document.body.appendChild(n);
```

Le Langage JavaScript (ES5)

- **Exemple de création d’un élément**

```
// Creation de l’objet image
var img = document.createElement("img");
// ou dans ce cas particulier
var img = new Image();
// Definition des attributs et style
img.src = "figure.png"
img.style.position = "absolute";
img.style.left = "10px";
// Affichage
document.body.appendChild(img);
```

Le Langage JavaScript (ES5)

- **La fenêtre courante**
  - La classe **Window** représente une fenêtre ou un onglet du navigateur
    - **L'objet window** de l'onglet courant **est le contexte d'exécution** JavaScript
    - Nos « **variables globales** » sont en fait **des attributs** de cet objet
  - Propriétés
    - **`document`** : document contenu dans la fenêtre
    - **`window/self`** : auto-référence
  - Méthode
    - **`alert()/prompt()`**: affiche ou lit une chaîne de caractères dans une boîte de dialogue
    - **`setInterval()/setTimeout()`**: programme l'exécution de code

Le Langage JavaScript (ES5)

- **Evénements**
  - Interactions avec l’utilisateur
    - Les actions de l’utilisateur déclenchent des événements
    - Le document web peut y réagir au moyen d’écouteurs qui s’ "abonnent" à certains types d’événements (c-à-d: des fonctions JavaScript qui vont être appelées automatiquement lorsque l’événement se produit)
  - Catégories d’événements
    - Touches clavier, boutons et mouvements de souris
    - Actions de formulaires
    - Vie du document ou des images

Le Langage JavaScript (ES5)

- **Evénements du document**
  - Document ou image
    - **`(on)load`**: après chargement de la page ou de l'image
    - **`(on)unload`**: en quittant la page
    - **`(on)abort`**: chargement de l'image interrompu
    - **`(on)error`**: erreur de chargement (image ou objet)
  - Intégration avec HTML
    - de nombreux éléments HTML ont des attributs du type **`onXXX="script"`** (onload, onerror, onclick, onkeypress...)
    - Ils associent le lancement d'un script-écouteur à un événement

Le Langage JavaScript (ES5)

- **Enregistrement d’une fonction d’initialisation**

```
<head>
<script>
   var e; // l’element "im1" n’existe pas encore
   function init() {
      e = document.getElementById("im1");
   }
</script>
</head>
<body onload="init();">
   <img id="im1" ...>
```

Le Langage JavaScript (ES5)

- **Enregistrement d'une fonction d'initialisation (2è méthode)**

```
<head>
<script>
   var e;
   window.onload = function(evt) {
      e = document.getElementById("im1") ;
   }
</script>
</head>
<body>
   <img id="im1" ...>
   (…)
</body>
```

Le Langage JavaScript (ES5)

- **Equivalent JavaScript de l'attribut HTML onload**
  - Lancer une (seule) fonction après chargement de la page

```
function init( ) { ... }
document.body.onload = init;
```

  - Gestionnaire moderne des événements
    - La méthode **addEventListener** est la plus générale
    - Elle peut s'appliquer au document ou à des éléments individuels (images...)

```
window.addEventListener('keypress', clavier, false);
window.addEventListener('load', init, false);
e.addEventListener('click', anime, false);
```

Le Langage JavaScript (ES5)

- **Evénements formulaires**
  - Evènements
    - **(on)submit**: envoi du formulaire (annulé si retourne ’false’)
    - **(on)reset**: réinitialisation
    - **(on)change**: modification d’un champ
    - **(on)select**: sélection dans un champ texte
  - Exemple de validation de formulaire :

```
function valider() {
   if (document.getElementById("nom").value == "") {
      alert(’Vous devez indiquer le nom’);
      return false; // empêche l’exécution par défaut de l’envoie du
         //  formulaire.
   }
}
```

HTML: `<form onsubmit="return valider()">`

Le Langage JavaScript (ES5)

- **Evénements souris – Principes**
  - Bouton de souris
    - **(on)mousedown**: pression du bouton de la souris
    - **(on)click**: appui et relâchement du bouton souris
    - **(on)dblclick**: double-clic de souris (interfère avec onclick)
    - **(on)mouseup**: relâchement du bouton de la souris
  - Déplacement de la souris
    - **(on)mouseover**: entrée du curseur dans l'élément
    - **(on)mouseout**: sortie du curseur de l'élément
    - **(on)mousemove**: déplacement de la souris (coûteux!)

Le Langage JavaScript (ES5)

- **Evénements souris - Position de la souris**
  - Spécificités JavaScript
    - HTML: ** **
    - DOM: propriétés ’clientX/clientY’ avec offset possible
  - Code JavaScript :

```
function follow(e) {
   alert("Clic en "+e.clientX+" " +e.clientY);
}
```

Le Langage JavaScript (ES5)

- **Evénements clavier – Principes**
  - Séquence des événements
    - **`(on)keydown`**: enfoncement de la touche
    - **`(on)keypress`**: envoi du caractère (à l'enfoncement puis à chaque répétition)
    - **`(on)keyup`**: relâchement de la touche
  - Propriétés de l'événement
    - **`type`**: type de l'événement ('keypress'...)
    - **`keyCode`**: code de la touche (pour keydown/up)
    - **`charCode`**: code du caractère (pour keypress)
    - **`altKey, ctrlKey, shiftKey`**: modificateurs associés

Le Langage JavaScript (ES5)

- **Evénements clavier - Caractères et touches**
  - Correspondance entre code et caractère
    - On peut avoir besoin de convertir le code ASCII en caractère
    - JavaScript: **`String.fromCharCode(charCode)`**
  - Quelques valeurs spéciales de keyCode

backspace	8	page haut	33
tabulation	9	page bas	34
entrée	13	fin	35
shift*	16	début	36
contrôle*	17	flèche gauche	37
alt*	18	flèche haut	38
bloc. maj.*	20	flèche droite	39
échappement	27	flèche bas	40

∗ ne déclenche pas d'événement keypress

Le Langage JavaScript (ES5)

- **Evénements clavier - Filtrage de formulaire**
  - Limitation de la saisie aux chiffres
    - JavaScript:

```
function filtre(e) {
   if (e.keyCode == 8)
      return; // autorise backspace
   if (e.charCode < 48 || e.charCode > 57)
      return false;
}
```

    - dans un formulaire HTML :

```
<input type="text" onkeypress="return filtre(event)">
```

Le Langage JavaScript (ES5)

- **Evénements clavier - Gestion dynamique**
  - Pas de fonction pour connaître l'état des touches du clavier, il faut enregistrer les changements lorsqu'ils se produisent...

    Savoir à tout moment si une touche est enfoncée

```
var touche_gauche = false;
function appuie(e) {
   if (e.keyCode == 37) touche_gauche = true;
}
window.addEventListener('keydown', appuie, false);
function relache(e) {
   if (e.keyCode == 37) touche_gauche = false;
}
window.addEventListener('keyup', relache, false);
```

Le Langage JavaScript (ES5)

- **Propriétés des événements**
  - Affichage des propriétés d'un événement

```
function afficherProps(e) {
   var msg = "";
   for (i in e) {
      msg += i + "=" + e[i] + ", ";
   }
   alert(msg);
}

window.addEventListener('XXX', afficheProps, false);
```

  - Permet de comparer le comportement de différents navigateurs

Le Langage JavaScript (ES5)

- **Propagation des événements**
  - Quel est l'ordre de traitement des événements dans le document ? ...ça dépend!
    - Modèle de base : les événements sont attachés à un seul élément
    - Modèle avancé : les événements se propagent dans la hiérarchie des éléments
      - en partant de la racine document (Netscape)
      - en remontant depuis l'élément (Microsoft)
      - dans les 2 sens (W3C)
    - Possibilité de bloquer la propagation
      - Microsoft: window.event.cancelBubble = true
      - W3C: e.stopPropagation()

Le Langage JavaScript (ES5)

- **Programmer une action**
  - Lancer l'exécution (répétitive ou non) d'un script ou d'une fonction
  - Déclenchement après un timer
    - **`setTimeout("script",delai)`** programme l'exécution du script après un délai en millisecondes
  - Action répétitive
    - **`var timer = setInterval("script",delai);`** répète le script avec un intervalle donné
    - **`clearInterval(timer);`** pour l'interrompre (il peut être nécessaire de définir timer en variable globale)
  - Pièges
    - La fonction connaîtra les variables globales au moment de l'exécution!
    - Limiter le nombre de timers actifs