Programmation Web – Avancé

BINV2150 A : JavaScript (& JAVA SERVLETS)

Week 2

R. Baroni / J.L. Collinet / C. Damas

0

Table des matières

Tous les sujets traités pendant ce cours...

Table des matières

Table des matières

3

Introduction au langage JS côté client, à l'utilisation d'APIs du navigateur et de librairies JS

Découvrons le langage côté client...

Table des matières :

3. Introduction au langage JS côté client, à l'utilisation d'APIs du navigateur et de librairies JS

Table des matières :
3. Introduction au langage JS côté client, à l'utilisation d'APIs du navigateur et de librairies JS

Table des matières :
3. Introduction au langage JS côté client, à l'utilisation d'APIs du navigateur et de librairies JS

Table des matières :
3. Introduction au langage JS côté client, à l'utilisation d'APIs du navigateur et de librairies JS

Les opérateurs

- Comparable au Java
- Les opérateurs arithmétiques
- Les opérateurs d'assignements
- Les opérateurs de comparaisons
  - Opérateur d'égalité ou de non égalité avec conversion si les opérandes sont de type différents : == ou !=

Les opérateurs

- Les opérateurs de comparaisons

```
1   ==  1        // true
'1'  ==  1       // true
0    == false     // true
0    == null      // false
var object1 = {'key': 'value'}, object2 = {'key': 'value'};
object1 == object2 //false
```

Les opérateurs

- Les opérateurs de comparaisons
  - Opérateur d'égalité ou de non égalité stricte sans conversion de type : **===** ou **!==**
  - **Bonne pratique** : utiliser l'égalité stricte sauf quand vous êtes sûr de vouloir convertir le type

```
1   === 1       // true
'1' === 1       // false
```

Les opérateurs

- Les opérateurs logiques
- ...
- Détails : : https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Expressions_and_Operators

Les conditions

- Instructions conditionnelles :
  - **if ... else**
  - **switch**
- Détails : https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Control_flow_and_error_handling

Les fonctions personnalisées et anonymes

- Fonctions personnalisées

```
function welcomeMessage(message){
        alert('Messages :\n' + message);
        return 'Messages :\n' + message;
}
welcomeMessage(‘Hello à toutes et à tous !’);
```

Les fonctions personnalisées et anonymes

- Assigner une fonction comme valeur de variable

```
//Définition de la fonction welcomeMessage (voir slide précédent)
var x=welcomeMessage;
x('Yo');
```

Les fonctions personnalisées et anonymes

- Fonction anonyme (ou sans nom)

```
var welcome=function(message){
            alert('Messages :\n' + message);
            return 'Messages :\n' + message;
}
welcome('Hello à toutes et à tous !');
```

Les fonctions personnalisées et anonymes

- Arrow function

```
var welcome2=(message) =>{
            alert('Messages :\n' + message);
            return 'Messages :\n' + message;
}
welcome2('Hello à toutes et à tous !');

const higher= n => n+1;
```

- **Bonne pratique** : toujours mettre les () et les {}

Paramètres (ou arguments) de fonctions

- **undefined** est alloué aux valeurs manquantes (tous les paramètres sont optionnels)
- Portée locale au sein de la fonction
- Passage d'argument par valeur, sauf pour les objets
- Passage d'un objet par référence

Paramètres (ou arguments) de fonctions

- Paramètre optionnels avec valeur par défaut

```
var welcome3=function(message = 'Bienvenu à toutes et à tous'){
            alert('Messages :\n' + message);
            return 'Messages :\n' + message;
}
welcome3();
```

Les fonctions personnalisées et anonymes

- Retour de la valeur d'une fonction
- Détails : https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Functions

Interactions de base avec l'API DOM : Javascript dans le browser

- ECMAScript
  - Stadardisation du JS, pour permettre de multiples implémentations (ActionScript, JScript, JavaScript, CommonJS...)
  - Utilisation primaire pour des scripts côtés client, mais de plus en plus utilisé côté serveur avec Node.js
  - Version en cours approuvée : 9th Edition – ECMAScript 2018

Interactions de base avec l'API DOM : Javascript dans le browser

- ECMAScript
  - Example de nouveaux éléments ES6 (ou ES 2015) : arrow functions, mot-clé let et const, promises, classes...
  - Suivre le développement : https://tc39.es/ ou https://developer.mozilla.org/en-US/docs/Web/JavaScript/Language_Resources
  - Support pour tous les browsers modernes : ES5
  - Support de ES6 (ES 2015) par la majorité des browsers, mais pas IE !

Interactions de base avec l'API DOM : Javascript dans le browser

- Bonne pratique : appliquer ES6 et le « strict mode »
  - Plus de feedback d'erreurs sur le code
  - Déclaration : **"use strict";** au début d'un script ou d'une fonction
  - Détails : https://www.w3schools.com/js/js_strict.asp
- Accès à la console : **F12** ou **CTRL + SHIFT + i** (Chrome)

Interactions de base avec l'API DOM : Javascript dans le browser

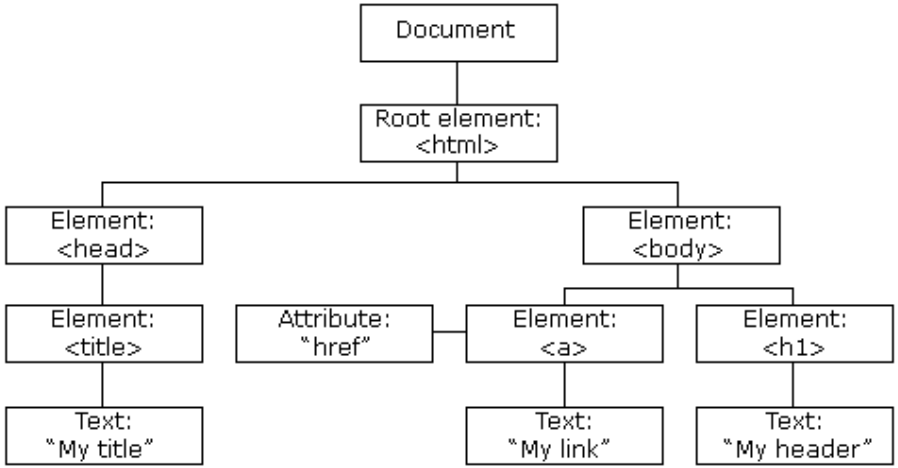
- **navigator** : objet contenant de l'information sur le browser
  - navigator.appName
  - navigator.appVersion
- **window** : objet manipulant la fenêtre du browser même : **alert()**, **prompt()**...

Interactions de base avec l'API DOM : Javascript dans le browser

- **document** : (**===window.document**) objet manipulant le corps de l'HTML de la page.
- **location** : (**===window.location**) objet représentant l'URL de la page.
  - **location.href** :cette URL.
  - **location.reload()** : recharge la page.
  - **location.replace('https://...')** : navigue à cette URL.
- Details sur ces objets : https://www.w3schools.com/jsref/

Interactions de base avec l'API DOM (Document Object Model)

- Représentation objet d'une page par le browser
- Mise à jour automatique d'une page du browser lors de manipulation JS sur un objet issu du DOM
- MàJ tant du HTML que du CSS : « CRUD operations" sur les éléments HTML et leur style (CSS)



The HTML DOM Tree of Objects [6]

Interactions de base avec l'API DOM (Document Object Model)

- Accéder à un élément HTML
  - **getElementById** : retourner l'élément dont l'attribut ID contient le paramètre donné
  - **getElementsByName** : retourner tous les éléments dont leur attribut « name » contient le paramètre donné
  - **querySelector** : retourner le 1er élément qui match un CSS selector (voir plus loin)
  - Autres méthodes : https://www.w3schools.com/jsref/dom_obj_document.asp

Interactions de base avec l'API DOM (Document Object Model)

- Accéder à une propriété ou à un événement d'un élément
  - **DEMO-02** : MàJ du texte associé à un bouton quand on clique dessus.
  - **DEMO-03A** : MàJ du texte d'une DIV quand on clique sur un bouton, en affichant le message associé à la fonction welcome().

JQuery

“write less, do more”

Utilisation de JQuery au sein de vos scripts

- Inclure à partir d'un CDN :
  `<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>`
- Télécharger et inclure la librairie :
  https://jquery.com/download/
  - PROD : `<script src="jquery-3.4.1.min.js"></script>`
  - DEV : `<script src="jquery-3.4.1.min.js"></script>`

Utilisation de JQuery au sein de vos scripts

- Utiliser la fonction jQuery via l'alias **$** : **$** === **jQuery**

```
$(“button").text("You clicked on me : )");
```

Utilisation principale de JQuery au sein de vos scripts

- **let elements=$(selector)**
- **selector** : chaîne de caractère qui décrit un chemin, quasi identique à un chemin CSS
- **elements** : tableau jQuery contenant tous les éléments du DOM atteint par le selector est des méthodes pour les manipuler

Utilisation principale de JQuery au sein de vos scripts

- Possède des fonctions permettant de manipuler ces éléments: **html(), text(), val(), .css(), attr(), prop(), on(), click(), append()...**
- Syntaxe pour appeler une méthode : **$(selector).action()**
- Tutorial intéressant pour JQuery : https://www.w3schools.com/jquery/default.asp

JQuerly selectors

Selector	$('Selector') renvoie sous forme d'objet(s) jQuery
$('#texteJQ')	la balise avec un attribut id valant 'texteJQ'
$('.bleu')	toutes les balises de classe bleu
$('h1')	toutes les balises <h1>
$('h1,h2')	toutes les balises <h1> et <h2>
$('ul')	
$('ul.bleu')	de classe bleu
$('div ul')	contenues dans une

JQuerly selectors

Selector	$(‘Selector') renvoie sous forme d'objet(s) jQuery
$('li[class]')	qui ont un attribut class
$('li[class="impair"]')	qui ont un attribut class de valeur impair
$(‘div img[width="40"]')	<img> qui ont un attribut width de valeur 40 contenues dans une

Comment s'assurer que le DOM est chargé avant utilisation du JS (ou JQuery) ?

- Appel du JS en fin de page Web
- Utilisation d'un événement « est chargé » :

```
$(document).ready(function(){
  // jQuery methods go here...
});
//ou
$(function(){
  // jQuery methods go here...
});
```

Introduction à Jquery en interaction avec le DOM

- **DEMO-03B :** Convertir en JQuery la DEMO-03A. Changer le code pour montrer l'appel du JS avant le chargement du DOM !

Introduction à la gestion d'événements

- Réagir aux actions des utlisateurs suite à la production d'un événement du browser : ajouter des gestionnaires d'événements
- Event handler properties (onclick, onfocus, on dblclick...)

```
document.body.onclick=()=> {console.log("click");}
```

Introduction à la gestion d'événements

- **addEventListener() & removeEventListener() :**

```
document.body.addEventListener("click", function {console.log("click")});
```

  - Ajout de plusieurs gestionnaires sur un élément : https://www.w3schools.com/js/js_htmldom_eventlistener.asp

- NB : Inline event handlers (à éviter!)

```
<body onclick="console.log('click')”>
```

Introduction à la gestion d'événements

- Event object : automatiquement passé au gestionnaire d'événements pour bénéficier d'infos et de fonctions supp.

```
var divs = document.querySelectorAll('div');
for (var i = 0; i < divs.length; i++) {
divs[i].onclick = function(e) {
e.target.style.backgroundColor = bgChange();
  }
}
```

Introduction à la gestion d'événements

- Intro à la gestion d'événements : https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Building_blocks/Events
- Liste des événements : https://developer.mozilla.org/en-US/docs/Web/Events

Introduction à la gestion d'événements

- **setTimeout(f,t)** : timer exécutant une fonction (f) quand celui-ci expire (après **t** millisecondes)
- **setInterval(f,t)** & **clearTimeout()** : exécuter une fonction f tous les t millisecondes jusqu'à l'appel de **clearTimeout()**
- Pourquoi les utiliser ?

Introduction à la gestion d'événements

- Plus de détails et exemple sur les timers :
  - https://www.w3schools.com/jsref/met_win_settimeout.asp
  - https://developer.mozilla.org/en-US/docs/Web/API/WindowOrWorkerGlobalScope/setTimeout

Exercices

A vous de jouer...

Interactions avec le DOM et gestion d'événements

- **EX-01A**: Réalisez un formulaire HTML (bien présenté) demandant un username, un email, et un password. Valider ce formulaire à l'envoi de celui-ci par le biais d'une gestion d'événement en JS. Utilisez JQuery pour accéder aux éléments.
  Pour la validation, vérifiez :
  - qu'il n'y ait pas de chiffre dans le username et qu'il ne soit pas vide;
  - que le format de l'email soit correct ;
  - que le password contienne au moins une Majuscule.

Interactions avec le DOM et gestion d'événements

- **EX-01A**:
  En cas d'erreurs, veuillez afficher celles-ci en dessous du formulaire, tout en mettant en rouge les inputs étant invalides. En cas de succès, accédez à une nouvelle URL de votre choix.
  - Notion intéressante - RegExp:
    - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions
    - https://www.w3schools.com/jsref/jsref_obj_regexp.asp

Interactions avec le DOM et gestion d'événements

- **EX-01B** :
  En cas de succès de la validation du formulaire, affichez un message visible pendant quelques secondes, en mettant en vert les inputs. Puis réalisez la redirection vers une nouvelle URL de votre choix.

Interactions avec le DOM et gestion d'événements

- **EX-01C (optionnel)** :
  Créez une page HTML contenant beaucoup de texte dans une police d'écriture difficilement lisible (petite taille). Faites en sorte que, lorsqu'un morceau de texte est sélectionné, celui-ci soit agrandi. En cas de double-clic, réinitialiser la taille de tout votre texte.
- **EX-01D (optionnel)** :
  Faites en sorte que le texte agrandi apparaisse au devant de la page, et disparaisse lorsque la souris le quitte.

Références

[1]	MDN web docs, Introduction to web APIs. Lien : https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Client-side_web_APIs/Introduction
[2]	MDN web docs, JavaScript Guide. Lien : https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide
[3]	w3schools.com, JavaScript Tutorial. Lien : https://www.w3schools.com/js/default.asp
[4]	tutorialspoints.com, Javascript Tutorial : Lien : https://www.tutorialspoint.com/javascript/index.htm

Références

[5]	Medium.com, Neal Burger, The end of life of IE11. Lien : https://medium.com/@burger.neal/the-end-of-life-of-internet-explorer-11-12736f9ff75f
[6]	w3schools.com, JS HTML DOM. Lien : http://www.w3schools.com/js/js_htmldom.asp