



JS

JavaScript

Histoire

2

- ▶ Créé en 1995 par Brendan Eich
- ▶ Standard ECMAScript
- ▶ Actuellement sur ECMAScript 7 (depuis juin 2016)
- ▶ Popularisé notamment grâce au moteur V8 de Chrome qui sera utilisé par la suite par NodeJS



Fonctionnement

3

- ▶ Langage interprété
- ▶ Code intégré au HTML (sauf dans le cas de NodeJS)
- ▶ Langage faiblement typé
- ▶ Liaisons dynamiques: les références des objets sont vérifiées au chargement
- ▶ Accessibilité du code (sauf dans le cas de NodeJS)

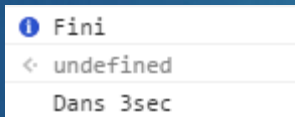
La notion de non-bloquant

4

JavaScript est un langage dit *non-bloquant* c'est-à-dire qu'il n'attendra pas la fin de l'exécution d'une fonction pour passer à la suite du script. Par exemple le code suivant :

```
setTimeout(function(){ console.log("Dans 3sec"); }, 3000);  
console.info('Fini');
```

Donnera dans la console :



```
❏ Fini  
⏪ undefined  
Dans 3sec
```

La console

5

- ▶ La console permet de faire ressortir des informations au développeur. Il existe par exemple plusieurs niveaux de logs :

➤ <code>console.log();</code>	<pre>> console.log('toto'); toto</pre>
➤ <code>console.info();</code>	<pre>> console.info('toto'); toto</pre>
➤ <code>console.warn();</code>	<pre>> console.warn('toto'); toto</pre>
➤ <code>console.error();</code>	<pre>> console.error('toto'); toto</pre>

- ▶ Il est aussi possible de faire des benchmarks directement via la console grâce à `console.time()` et `console.timeend()`
- ▶ Il est possible d'afficher toute la stacktrace grâce à `console.trace()`
- ▶ Il est aussi possible de faire des tests grâce à `console.assert()`

Les variables

6

- ▶ La déclaration d'une variable se fait via le mot clé *var* si le mot clé n'est pas précisé la variable sera globale
- ▶ Depuis ES6 il est possible de déclarer des variables via *let* et *const*, *let* permet de créer une variable qui ne sera valable que dans le contexte actuel et *const* sera une constante.
- ▶ Les variables sont sensibles à la casse.
- ▶ Une variable déclaré dans une fonction ne sera pas accessible depuis l'extérieur. Exemple :

```
var myVar = 2;

function myFunction () {
  var myVar = 3;
  console.info(myVar); // 3
}

myFunction();

console.info(myVar); // 2
```

Les conditions

7

► If/else if/else

```
if (i === 2) {  
  console.log(2);  
} else if (i === 3) {  
  console.log(3);  
} else {  
  console.log(0);  
}
```

► Switch

```
switch(myVar){  
  case 'un':  
    console.log('un');  
    break;  
  case 'deux':  
    console.log('deux');  
    break;  
  default:  
    console.log('none');  
}
```

► Ternaire

```
var value = (myVar)?'true':'false';
```

Les boucles

- ▶ while
- ▶ for
- ▶ do while
- ▶ foreach

Les fonctions

En JavaScript les fonctions peuvent être nommées ou non.

Exemple :

```
var myFunction = function () {  
    console.log('coucou');  
};  
  
function myFunction2 () {  
    console.log('coucou 2');  
}  
  
myFunction();  
myFunction2();
```

Les tableaux

10

- ▶ Pour déclarer un tableau il faut passer par `[]`, on ne passera pas la déclaration `new Array()` qui est dépréciée
- ▶ Pour ajouter des éléments dans le tableau on passera par la méthode `array.push()`
- ▶ Pour afficher un élément il faudra passer par une boucle ou par un accès direct *via* par exemple `array[0]`
- ▶ Il existe un grand nombre de méthodes pour interagir avec un tableau tel que `split()`, `shift()`, `pop()`, `join()`, `toString()`

Les objets

11

La création d'un objet peut être faite de trois façons :

- Via un JSON (JavaScript Object Notation)

```
var maVoiture = {  
  fabricant: "Ford",  
  modèle: "Mustang",  
  année: "1969"  
}
```

- Via la création d'un Object()

```
var maVoiture = new Object();  
maVoiture.fabricant = "Ford";  
maVoiture.modèle = "Mustang";  
maVoiture.année = 1969;
```

Les objets

12

► Via un constructeur :

```
function Voiture(fabricant, modèle, année) {  
    this.fabricant = fabricant;  
    this.modèle = modèle;  
    this.année = année;  
}  
  
var maVoiture = new Voiture('Ford', 'Mustang', '1969');
```

Pour accéder aux informations il est possible de faire ensuite
maVoiture.fabricant

- ▶ L'AJAX (Asynchronous JavaScript And XML) permet de faire des requêtes HTTP sans avoir besoin de rafraîchir la page
- ▶ AJAX utilise l'objet XMLHttpRequest() pour créer ses requêtes
- ▶ Il faut toujours vérifier que la page est prête (readyState à 4) et renvoie bien un code HTTP 200 (OK). Exemple :

```
var req = new XMLHttpRequest();
req.open('GET', 'http://www.google.com', true);
req.onreadystatechange = function() {
  if (req.readyState == 4 && req.status == 200) {
    console.log(req.responseText);
  } else {
    console.error("Erreur pendant le chargement de la page.\n");
  }
};
req.send(null);
```

- ▶ Si la requête est *cross-domain*, les CORS (Cross-Origin Resource Sharing) des headers doivent préciser qu'ils acceptent les requêtes de toutes les origines

Le prototypage

14

JavaScript est un langage objet à prototype ce qui permet de modifier un objet *via* un constructeur et non *via* une instance de classe.

Il est donc possible par exemple de rajouter une fonction qui sera utilisable par les chaînes de caractères. Exemple :

```
String.prototype.star = function () {  
    return "** " + this + " **";  
};  
  
console.log("Coucou".star()); // "** Coucou **"
```

Il existe plusieurs méthodes pour sélectionner un élément dans le DOM :

- ▶ `document.querySelector()`
- ▶ `document.querySelectorAll()`
- ▶ `document.getElementById()`
- ▶ `document.getElementsByName()`
- ▶ `document.getElementsByTagName()`
- ▶ `document.getElementsByClassName()`

Modification du HTML/CSS

16

Il existe plusieurs méthodes pour modifier un élément dans la page :

- ▶ `document.getElementById(id).style.propriété = 'nouveau style'` pour modifier le style d'un élément
- ▶ `document.getElementById(id).innerHTML = 'nouveau code HTML'` pour modifier le code HTML d'un élément

Évènements

17

En JavaScript il est possible d'écouter les évènements d'une page HTML. Il est par exemple possible d'écouter un changement dans un select ou la pression d'une touche de clavier sur un input. Exemple :

```
document.querySelector('select').onchange = function () {  
    console.info("Select modifié");  
};  
  
document.querySelector('input[type="text"]').onkeyup = function () {  
    console.info("Pression d'une touche du clavier");  
};
```

En plus

18

- ▶ API File
- ▶ Drag & Drop
- ▶ Audio
- ▶ Vidéo
- ▶ Canvas

Bibliothèques

19

- ▶ jQuery
- ▶ MooTools
- ▶ React

ECMAScript 6

20

- ▶ *let / const*
- ▶ Gabarit de chaînes de caractères
- ▶ Arrow functions
- ▶ `String.startsWith()`, `String.endsWith()`, `String.includes()`, `String.repeat()`
pour améliorer le `String.indexOf()`
- ▶ Classes
- ▶ Yield
- ▶ Unicode
- ▶ Proxy
- ▶ Promises !

ECMAScript 7

21

ECMAScript 7 est en cours de développement.

- ▶ Async / Await
- ▶ Opérateur d'exponentiation **
- ▶ Opérateur de binding ::
- ▶ Objets typés
- ▶ Décorateurs
- ▶ Object.observe() (O.o)
- ▶ Observable