

DÉVELOPPEMENT JAVASCRIPT

A large, stylized logo for JavaScript. The letters 'J' and 'S' are formed by thick black lines. The 'J' is a vertical bar with a horizontal stroke at the bottom. The 'S' is a thick, curved line forming a loop. The background is yellow.

Thomas Fritsch thomas@kumquats.fr

(<mailto:thomas@kumquats.fr>)

Enseignant & Formateur ~ Fondateur & Directeur technique de



- JS / HTML 5 / CSS 3 (React, Sass/Compass)
- PHP (Symfony, WordPress)
- Mobile (React Native, Cordova/PhoneGap)

<http://kumquats.fr> (<http://kumquats.fr>)  [@kumquatsfr](https://twitter.com/kumquatsfr) ([http://twitter.com/kumquatsfr](https://twitter.com/kumquatsfr))

VOUS

Qui a déjà :

- développé en JS ?
- entendu parler d'ES6 ?
- développé en ES6 ?
- utilisé jQuery ?
- appris ce qu'était AJAX ?
- développé avec
React/Angular/Vue ?
- subi un de mes cours ?
- envie de s'enfuir ?

PROGRAMME

1. Rappels
2. Premiers pas en JS

AVERTISSEMENT

Chasse au spoil !

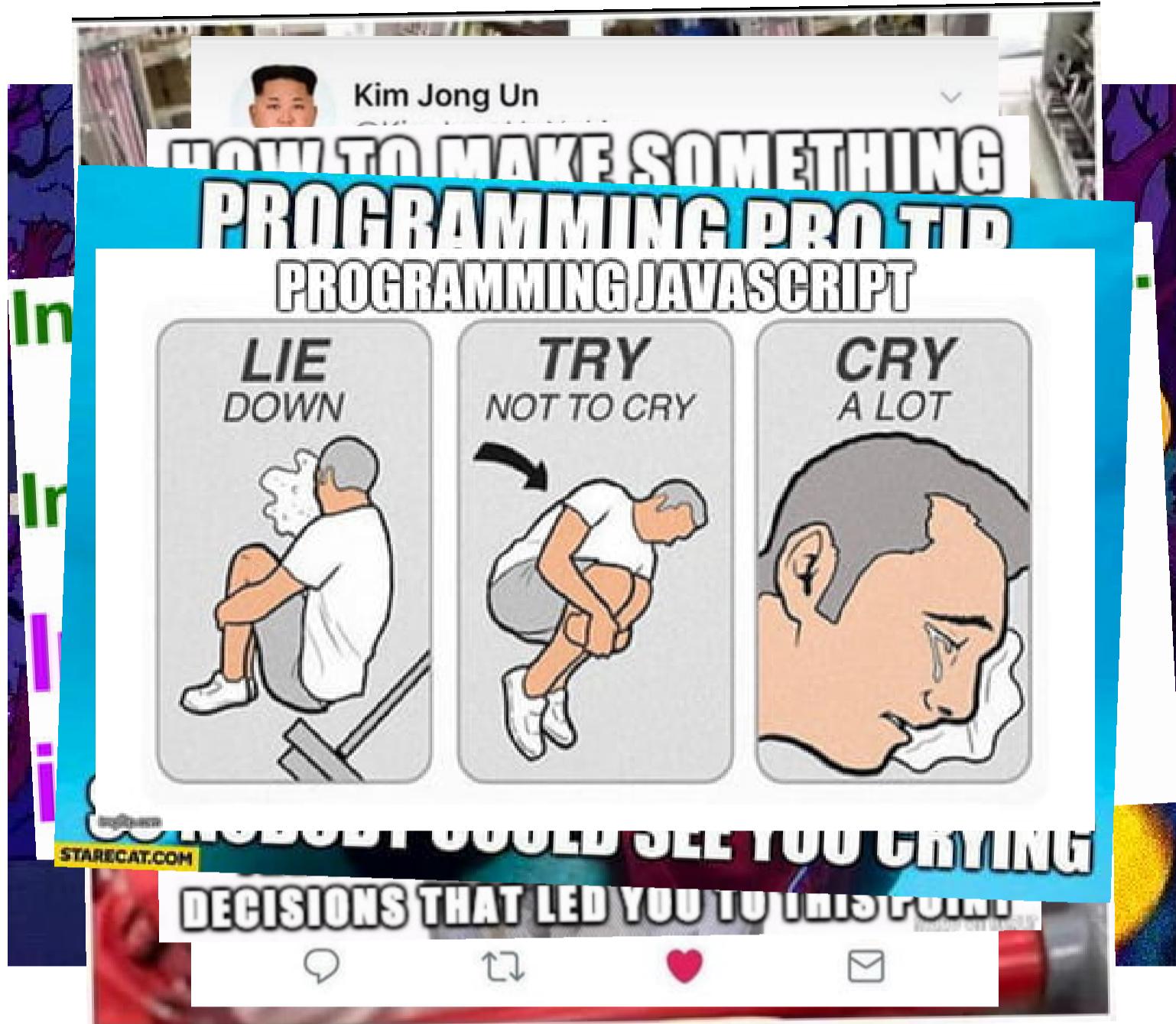


DES QUESTIONS ?

JAVASCRIPT,
POURQUOI
TANT
DE
Haine ?

“ *Franchement, JavaScript c'est un langage
merdique !* ”

le 14 mars 2018 - un prof de JAVA



POURQUOI ?

- pas de typage fort
- pas de POO
- scope incompréhensible
- "this" aléatoire
- pas d'outils de dev sérieux
- variables globales par défaut
 - pas de gestion des dépendances
 - ne marche pas dans tous les navigateurs
 - une sous-version pourrie de JAVA

ON VOUS **MENT !**

tout ça c'est du passé

JAVASCRIPT C'EST BIEN

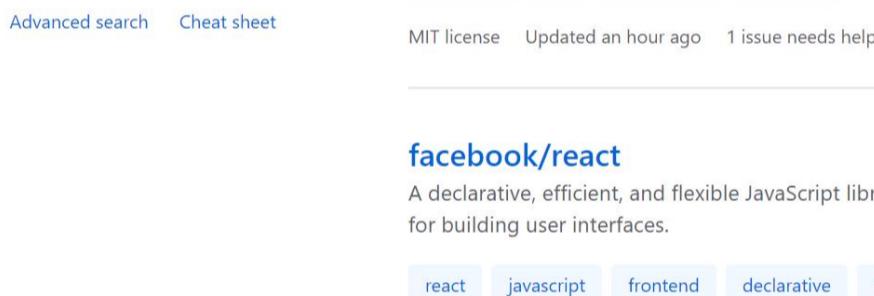
- universel / crossplatform
- 99% des terminaux compatibles

mais aussi ...



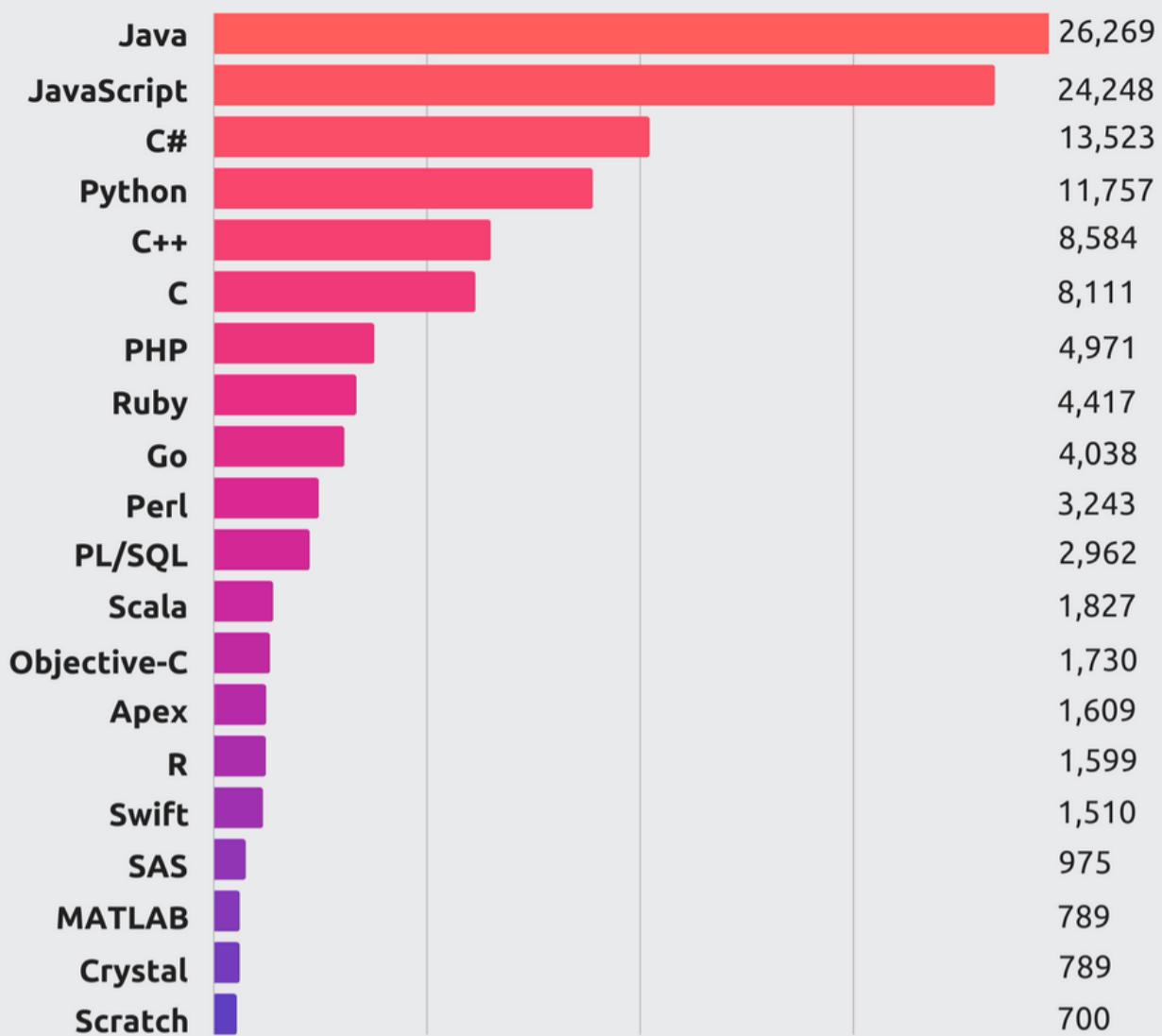
Notes :

Source : <https://github.com/search?o=desc&q=stars%3A%3E5000&s=stars&type=Repositories>
[\(https://github.com/search?o=desc&q=stars%3A%3E5000&s=stars&type=Repositories\)](https://github.com/search?o=desc&q=stars%3A%3E5000&s=stars&type=Repositories)



Most In-Demand Languages

Indeed Job Openings - Dec. 2017



JAVASCRIPT ALL THE THINGS^{2.9}

Permet de développer :

- des applis web front
- des applis serveur (Node.JS)
- des applis desktop (Electron)
- des applis mobiles (React Native, NativeScript, ...)
- ...

De la musique pour tous.

Des millions de titres. Aucune carte de crédit nécessaire.

TÉLÉCHARGER SPOTIFY FREE (/FR/DOWNLOAD/)

Notes :

Spotify utilise JS pour le développement de son application Desktop

Source : <https://www.quora.com/How-is-JavaScript-used-within-the-Spotify-desktop-application-Is-it-packaged-up-and-run-locally-only-retrieving-the-assets-as-and-when-needed-What-JavaScript-VM-is-used/answer/Mattias-Petter-Johansson> (<https://www.quora.com/How-is-JavaScript-used-within-the-Spotify-desktop-application-Is-it-packaged-up-and-run-locally-only-retrieving-the-assets-as-and-when-needed-What-JavaScript-VM-is-used/answer/Mattias-Petter-Johansson>)

"The Spotify desktop client UI is completely built with JavaScript, resting on top of the same C++ core that the iOS and Android clients uses."

Skype permet de rester en contact facilement

Parlez. Discutez. Collaborez.

[Obtenir Skype](#)

Certaines fonctionnalités de Skype peuvent différer avec la mise à jour anniversaire de Windows 10 ou une version ultérieure.

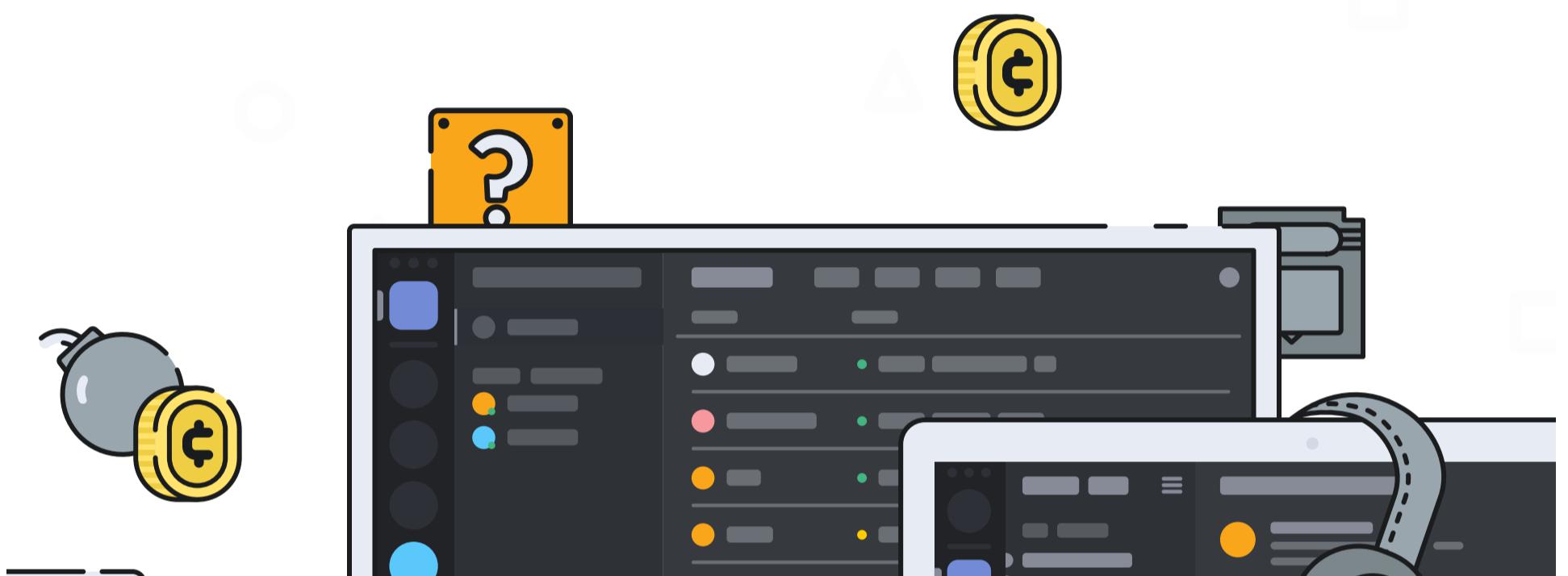
Notes :

Les applis Skype desktop et mobile sont codées en JS

Sources : <https://electronjs.org/apps/skype> (<https://electronjs.org/apps/skype>) & <https://facebook.github.io/react-native/showcase.html> (<https://facebook.github.io/react-native/showcase.html>)

Il est temps d'abandonner Skype et TeamSpeak.

Chat vocal et texte tout-en-un gratuits, sécurisés et qui fonctionnent sur PC et smartphone, tout ça pour les gamers. Arrêtez de payer pour des serveurs TeamSpeak et de galérer avec Skype. Simplifiez-vous la vie.

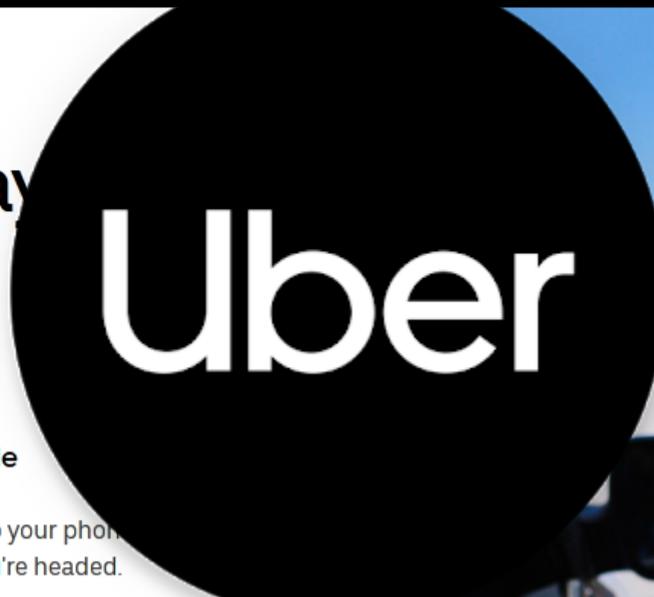
[Télécharger pour Windows](#)[Ouvrir Discord dans votre navigateur](#)

Notes :

Le client desktop de Discord ainsi que l'appli mobile sont aussi développés avec JavaScript (Electron)

Source : [\(https://en.wikipedia.org/wiki/Discord_\(software\)#cite_ref-15\)](https://en.wikipedia.org/wiki/Discord_(software)#cite_ref-15) & [\(https://facebook.github.io/react-native/showcase.html\)](https://facebook.github.io/react-native/showcase.html)

Move the way
you want

The Uber logo, which consists of the word "Uber" in white lowercase letters inside a large black circle.

Uber

Drive

Drive when you want. Find
opportunities around you.

[Learn more](#)

Ride

Tap your phone
when you're headed.

[Learn more](#)

[Sign up to drive →](#)

[Sign up to ride →](#)



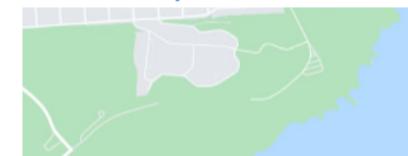
Notes :

Uber utilise JavaScript côté serveur avec Node.JS

Source : [\(https://foundation.nodejs.org/wp-content/uploads/sites/50/2017/09/Nodejs-at-Uber.pdf\)](https://foundation.nodejs.org/wp-content/uploads/sites/50/2017/09/Nodejs-at-Uber.pdf)

Where to?

Get a price estimate





See what's next.

OÙ QUE VOUS SOYEZ. SANS ENGAGEMENT.

PROFITEZ D'UN MOIS GRATUIT



Notes :

Les applis serveur de Netflix utilisent aussi Node.JS

Source : <https://medium.com/netflix-techblog/node-js-in-flames-ddd073803aa4> (<https://medium.com/netflix-techblog/node-js-in-flames-ddd073803aa4>)

ECMASCRIPT ?

- la Spec suivie par JavaScript
- beaucoup ne connaissent qu'ES5 (2009)
- ES6 sorti en 2015 ! nouvelle syntaxe, nouvelles possibilités
- depuis, une nouvelle version / an : ES7, ES8, ES9, ...
- POO, gestion des dépendances, block scope, Promises/Futures, ...
- code compilé en ES5 pour assurer rétro-compatibilité

GET BABEL HOLIDAY APPAREL 

Babel is a JavaScript compiler.

Use next generation JavaScript, today.

Babel 7 is out! Please read our [announcement](#) and [upgrade guide](#) for more information.

Put in next-gen JavaScript

```
var [a,,b] = [1,2]
```

Get browser-compatible JavaScript out

Notes :

<https://babeljs.io/> (<https://babeljs.io/>)

L'ES6 n'étant pas encore supporté de manière native par tous les navigateurs actuels, il est encore nécessaire d'utiliser un compilateur/préprocesseur.

Le principe est simple : on écrit son code en ES6 et le compilateur le convertit en JavaScript ES5 qui lui est reconnu par tous les navigateurs.

Il existe plusieurs préprocesseurs JS mais le plus connu est Babel (<https://babeljs.io/> (<https://babeljs.io/>))

[Docs](#)[Setup](#)[Try it out](#)[Videos](#)[Blog](#)[Donate](#)[Team](#)[GitHub](#)**SETTINGS**

- Evaluate
- Line Wrap
- Minify
- Prettify
- File Size
- Time Travel

Source Type

Module

PRESETS

- es2015
- es2015-loose
- es2016
- es2017
- stage-0
- stage-1
- stage-2
- stage-3
- react
- flow
- typescript

Notes :

Babel dispose d'un outil en ligne qui permet de tester la conversion de code ES5 en ES6. Pratique pour se rendre compte de l'intérêt de la nouvelle syntaxe notamment pour son côté beaucoup plus concis.

[Outil de test en ligne du compilateur babel \(repl\) \(https://babeljs.io/repl\)](https://babeljs.io/repl)

babel repl pré-rempli avec l'exemple de code ci-dessus ([https://babeljs.io/repl/build/master?babili=false&browsers=&build=&builtIns=false&spec=false&loose=false&code_lz=MYGwhgzhAECCB2BLAtmE0DeAoAAXPJcgEYCMATtALzQAMA3FmjRB0HPDDIwtAm0TwA5gKE5EEAClswAE1pNy5EJvhdoAMzQQWAnAGJgYeADEQAT137DZCWlQ4WUKeBIKYHxyNgAKcUlpfFkFABoyB2c3PQMjbwsWAEpsAID8AAAtVADpYlhMq_yLSirs01xN7J1c6gIQUNHKKanwAWwcHDZpyjZjBohK-bHcXFNXRcwRZURTaCiAQj3ypvaXPluij-Ky_dUNbRMEpQrJcPjhSmxyAB3aDwFjQgCibAh0Wgjyqg2gAHJUvB4OR8GYWnc7pi8tYcOMKVhxLhggRUSVNix4S4WFoAMJRql8MASIF2Mi3ms0Ox8GakZFypl0GApnIqUA&debug=false&forceAllTransforms=false&shippedProposals=false&ci3%2Cflow&prettier=false&targets=&version=7.2.2&envVersion="\)](https://babeljs.io/repl/build/master?babili=false&browsers=&build=&builtIns=false&spec=false&loose=false&code_lz=MYGwhgzhAECCB2BLAtmE0DeAoAAXPJcgEYCMATtALzQAMA3FmjRB0HPDDIwtAm0TwA5gKE5EEAClswAE1pNy5EJvhdoAMzQQWAnAGJgYeADEQAT137DZCWlQ4WUKeBIKYHxyNgAKcUlpfFkFABoyB2c3PQMjbwsWAEpsAID8AAAtVADpYlhMq_yLSirs01xN7J1c6gIQUNHKKanwAWwcHDZpyjZjBohK-bHcXFNXRcwRZURTaCiAQj3ypvaXPluij-Ky_dUNbRMEpQrJcPjhSmxyAB3aDwFjQgCibAh0Wgjyqg2gAHJUvB4OR8GYWnc7pi8tYcOMKVhxLhggRUSVNix4S4WFoAMJRql8MASIF2Mi3ms0Ox8GakZFypl0GApnIqUA&debug=false&forceAllTransforms=false&shippedProposals=false&ci3%2Cflow&prettier=false&targets=&version=7.2.2&envVersion=)

YOU LOVE



JAVASCRIPT

memegenerator.net

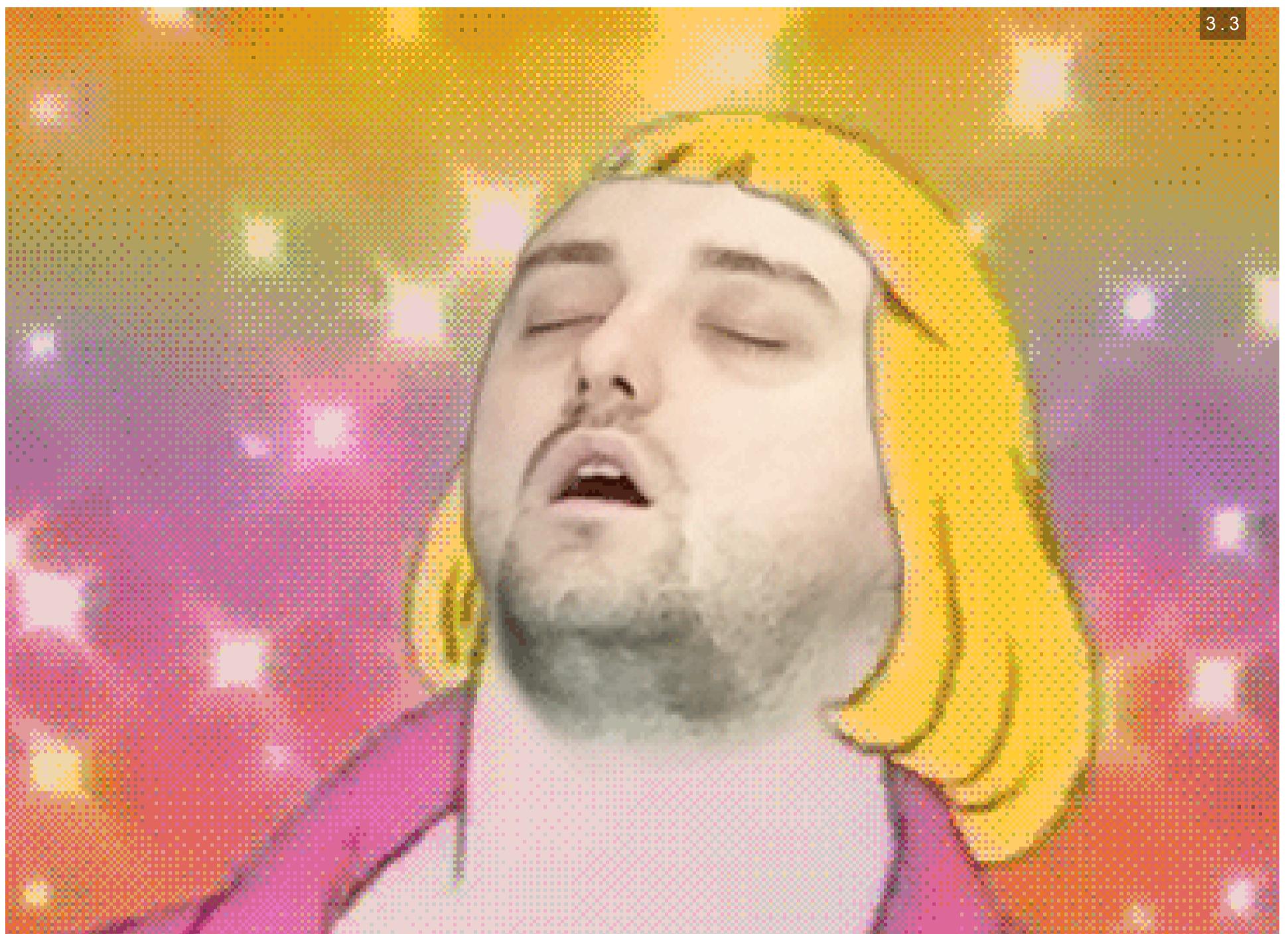
JS

1. Rappels

2. Premiers pas en JS
3. La POO en JS
4. API DOM
5. Ajax
6. jQuery

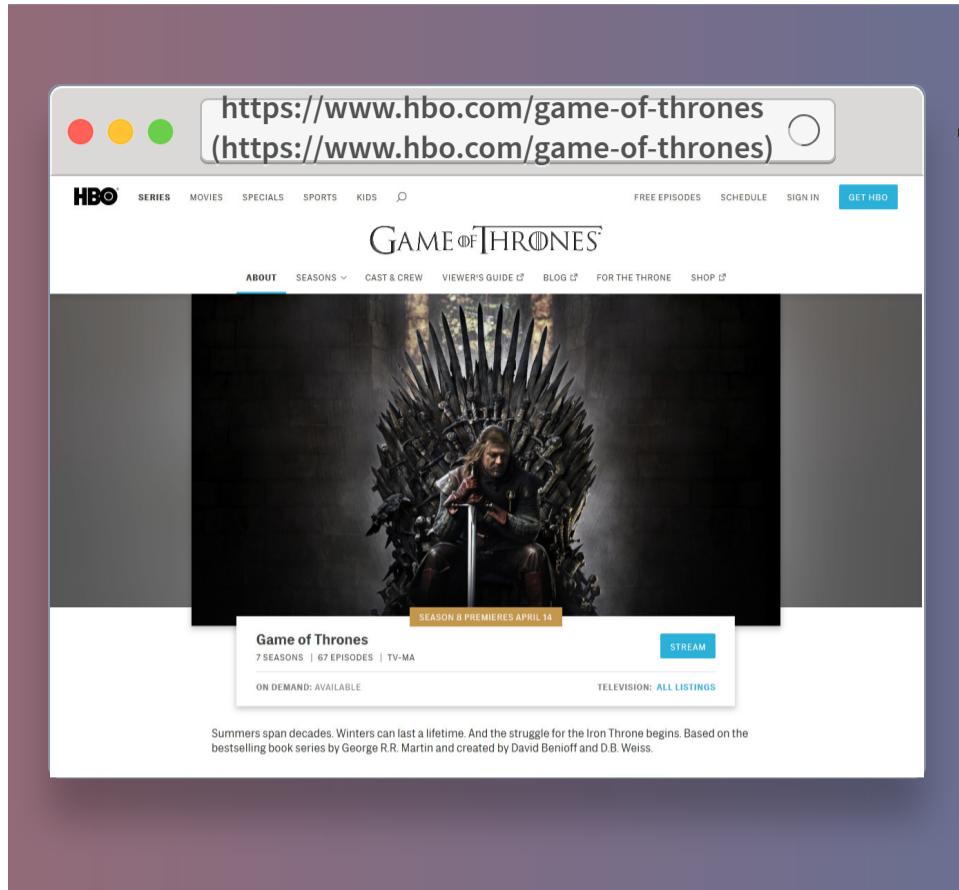
1. RAPPELS

- client/serveur & HTTP
- HTML ❤ JS



CLIENT / SERVEUR

CLIENT (NAVIGATEUR)



SERVEUR HTTP

→ GET /game-of-thrones HTTP/1.1 (Request)



HTML

← HTTP/1.1 200 OK (Response)
+ headers
+ body

Notes :

“ L'environnement client-serveur désigne un mode de communication à travers un réseau entre plusieurs programmes ou logiciels : l'un, qualifié de client, envoie des requêtes ; l'autre ou les autres, qualifiés de serveurs, attendent les requêtes des clients et y répondent. ”

Wikipedia

REQUÊTES HTTP

```
GET /game-of-thrones HTTP/1.1
Host: www.hbo.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2924.87
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q
=0.8
DNT: 1
Accept-Encoding: gzip, deflate, sdch
Accept-Language: fr-FR,fr;q=0.8,en-US;q=0.6,en;q=0.4
```

Notes :

Sous windows, on peut utiliser le programme telnet pour lancer des requêtes http (ou) : dans un terminal / command prompt, taper : copier coller :

```
telnet www.amc.com 80
```

puis taper :

```
GET /game-of-thrones HTTP/1.1
Host: www.hbo.com
```

Terminer par un double retour à la ligne à la fin qui indique au serveur la fin de la requête.

Vous verrez que le serveur retourne une Réponse "301 Moved permanently" car le http est automatiquement redirigé vers le https.

Pour effectuer des requêtes https, il faut utiliser [openssl \(https://www.openssl.org/\)](https://www.openssl.org/) :

```
openssl s_client -connect www.hbo.com:443
```

Puis taper :

```
GET /game-of-thrones HTTP/1.1
Host: www.hbo.com
```

RÉPONSE HTTP

```
HTTP/1.1 200 OK
Date: Fri, 18 Jan 2019 20:03:54 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 27968
Connection: keep-alive
Cache-Control: max-age=300, s-maxage=31536000
Content-Encoding: gzip
ETag: "2e734-57f82cf0aa99c-gzip"
Expires: Fri, 18 Jan 2019 20:08:54 GMT
Last-Modified: Tue, 15 Jan 2019 17:51:19 GMT
Server: Apache
Vary: Accept-Encoding,User-Agent
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: g
countryCode: BE
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Age: 180854
Accept-Ranges: bytes

<!doctype html>
<html lang="en">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8"/>

    <title>Game of Thrones - Official Website for the HBO Series</title>
    <meta name="viewport" content="width=device-width, initial-scale=1,
user-scalable=no, maximum-scale=1.0"/>

    <meta name="description" content="The official website for Game of
Thrones on HBO, featuring full episodes online, interviews, schedule
information and episode guides."/>
    <link rel="icon" href="/etc/designs/hboweb/favicon/favicon.ico"
type="image/x-icon"/>
    ...
  </head>
  <body>
    <h1>Game of Thrones</h1>
    <p>The official website for Game of Thrones on HBO, featuring full episodes online, interviews, schedule information and episode guides.</p>
    <ul>
      <li>Home</li>
      <li>Episodes</li>
      <li>Interviews</li>
      <li>Schedule</li>
      <li>Guides</li>
      <li>About</li>
      <li>Contact</li>
    </ul>
  </body>
</html>
```

GAME OF THRONES®

[ABOUT](#)[SEASONS ▾](#)[CAST & CREW](#)[VIEWER'S GUIDE ↗](#)[BLOG ↗](#)[FOR THE THRONE](#)[SHOP ↗](#)

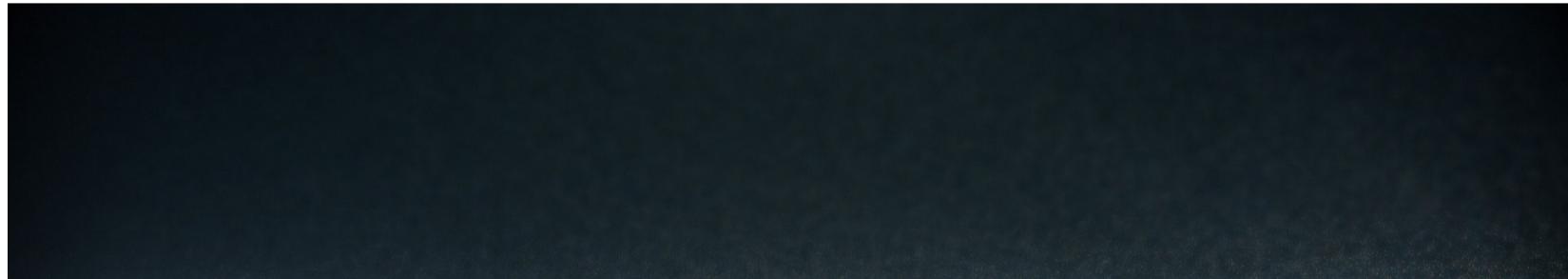
SEASON 8 PREMIERES APRIL 14

Game of Thrones

7 SEASONS | 67 EPISODES | TV-MA

[STREAM](#)[ON DEMAND: AVAILABLE](#)[TELEVISION: ALL LISTINGS](#)

Summers span decades. Winters can last a lifetime. And the struggle for the Iron Throne begins. Based on the bestselling book series by George R.R. Martin and created by David Benioff and D.B. Weiss.



SINGLE PAGE APP

Une seule page HTML + navigation en JS

1. l'utilisateur clique sur un lien de la page
2. le JS intercepte le clic et lance une requête HTTP (AJAX) vers le serveur
3. le serveur retourne les données au format brut (JSON/XML/...)
4. le JS parse les données reçues ...
5. ... et modifie le code HTML de la page (API DOM)

HTML



HTML

Le code HTML est composé d'un arbre de balises (XML) :

- converti en objets manipulables via l'API DOM
- affiché par le moteur de rendu du browser (webkit, gecko, blink, etc.) en attribuant un comportement différent à chaque balise selon son type

HTML + JS : INTÉGRATION

Inline

```
<a href="#" onclick="alert('Welcome to Westeros');return false;">  
    GOT  
</a>
```

Dans une balise script

```
<script>alert('Bienvenue à Westeros');</script>
```

Dans un fichier externe

```
<script src="westeros.js"></script>
```

HTML + JS : MAIS T'ES OÙ ?

```
<!doctype html>
<html>
<head>
  <title>L'histoire de Westeros</title>
  <meta charset="utf-8">
  <meta name="description" content="L'histoire de Westeros en 587 chapitres">
  <script src="build/main.js" defer></script> <!-- 🎻 -->
</head>
<body>
  <h1>Westeros history</h1>
  <p>The recorded history of Westeros extends back over
    12,500 years, according to tradition...</p>
</body>
</html>
```



Markup Validation Service

Check the markup (HTML, XHTML, ...) of Web documents

[Validate by URI](#)
[Validate by File Upload](#)
[Validate by Direct Input](#)

Validate by URI

Validate a document online:

Address:

[► More Options](#)

[Check](#)

This validator checks the [markup validity](#) of Web documents in HTML, XHTML, SMIL, MathML, etc. If you wish to validate specific content such as [RSS/Atom feeds](#) or [CSS stylesheets](#), [MobileOK content](#), or to [find broken links](#), there are [other validators and tools](#) available. As an alternative you can also try our [non-DTD-based validator](#).



Interested in understanding what new technologies are coming out of W3C? Follow [@w3cdevs](#) on [Twitter](#) to keep track of what the future looks like!

[Donate](#) and help us build better tools for a better web.

[Home](#) [About...](#) [News](#) [Docs](#) [Help & FAQ](#) [Feedback](#) [Contribute](#)



This service runs the W3C Markup Validator, [v1.3+hg](#).

COPYRIGHT © 1994-2013 W3C® (MIT, ERCIM, KEIO, BEIHANG), ALL RIGHTS RESERVED. W3C LIABILITY, TRADEMARK, DOCUMENT USE AND SOFTWARE LICENSING RULES APPLY. YOUR INTERACTIONS WITH THIS SITE ARE IN ACCORDANCE WITH OUR PUBLIC AND MEMBER PRIVACY STATEMENTS.



JS

1. Rappels

2. Premiers pas en JS

3. La POO en JS

4. API DOM

5. Ajax

6. jQuery

2. PREMIERS PAS EN JS

- Syntaxe de base
- Les types de données
- Structures de contrôle
- Fonctions

SYNTAXE DE BASE

- ";" à la fin de chaque instruction
- blocs délimités par des accolades "{}"
- lowerCamelCase & sensible à la casse
- commentaires avec // et /* */
- code exécuté par le navigateur de haut en bas du code
html

JavaScript reference

Jump to: [Global objects](#) [Statements](#) [Expressions and operators](#) [Functions](#) [Additional reference pages](#)

This part of the JavaScript section on MDN serves as a repository of facts about the JavaScript language. Read more about this reference.

Global objects

This chapter documents all the JavaScript standard built-in objects, along with their methods and properties.

[Value properties](#) 

Notes :

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference> (<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference>)

JavaScript Tutorial

[← Home](#)[Next →](#)

JavaScript is the programming language of HTML and the Web.

JavaScript is easy to learn.

This tutorial will teach you JavaScript from basic to advanced.

Examples in Each Chapter

With our "Try it Yourself" editor, you can change all examples and view the results.

[Example](#)

Notes :

<https://www.w3schools.com/js/> (<https://www.w3schools.com/js/>)

VARIABLES

3 façons de déclarer des variables :

- var
- var ES5 !
- let
- const

```
let ramseyIsDead; // déclaration de variable
let handOfTheKing = 'Ned Stark'; // déclaration et assignation de valeur
let king = 'Joffrey', // déclaration multiple
      city = 'King\'s Landing';

const hodor = 'hodor'; // déclaration de constante
                      // impossible de modifier l'affectation mémoire
```

CONST ?

```
const brother; // ⚡ Erreur : une constante doit avoir une  
  
const brothers = [ 'Robb', 'Bran' ]; // 🎉  
  
brothers.push('Rickon'); // 🎉  
  
brothers = [ 'Robb', 'Bran', 'Rickon' ]; // ⚡ Erreur : impossible de changer  
// l'affectation mémoire
```

LES TYPES DE DONNÉES

- Principaux types de données :
 - String
 - Number
 - Boolean
 - Array
 - Object
 - Date
 - Function
- Typage faible (enfin, pour l'instant...) et dynamique

LE TYPE STRING

déclaration

```
/* guillemets simples */
let s1 = 'je suis une chaîne avec des single quotes';

/* ou guillemets doubles */
let s2 = "je suis une chaîne avec des double quotes";

/* ou accent grave (template strings ES6) */
let s3 = `Les étudiants de ${getCurrentSchool()} sont les meilleurs`;
```

opérateur de concaténation : +

```
let episodeName = 'The door';
let title = '<h1>' + episodeName + '</h1>';
```

Notes :

On peut déclarer les chaînes de caractères de plusieurs manières, avec des guillemets simples, doubles ou avec l'accent grave (`). Cette dernière notation est issue de la spec ES6/ES2015.

En revanche on évite d'utiliser l'instruction `new String('ma chaine')` car elle consomme plus de ressources et provoque des résultats inattendus notamment lors de la comparaison de deux chaînes de caractères.

LE TYPE STRING

déclaration multiligne

```
s1 = 'on peut déclarer une chaine sur plusieurs lignes'  
    +'en utilisant '  
    +'la concaténation.';  
  
s2 = 'ou alors on utilise un antislash \  
      avant \  
      chaque saut de ligne.';  
  
s3 = `mais le plus simple  
c'est quand même  
d'utiliser les template strings (``` !`
```

LE TYPE STRING

Principales méthodes

```
const serie = 'Game Of Thrones';
console.log(
    serie.length, // 15 (longueur de la chaîne)

    serie.search( 'Of' ), // 5 (position de la chaîne recherchée)
    serie.search( 'LOL' ), // -1 (chaîne introuvable)

    serie.split( ' ' ), // ['Game', 'Of', 'Throne'] (découpe)

    serie.substring( 5, 7 ), // 'Of' (portion d'une chaîne)

    serie.toUpperCase() // 'GAME OF THRONES'
);
```

LE TYPE NUMBER

Déclaration

```
// Nombre entier  
let age = 9;  
  
// Nombre à virgule flottante  
let price = 12.5;
```

LE TYPE NUMBER

Attention à la conversion de type implicite

```
42 + 12          // 54
42 + "12"        // "4212"
42 + true         // 43
42 - "12"        // 30 !
```

LE TYPE NUMBER

Méthodes

```
let price = 12.5;
console.log(
  price.toFixed(2), // ?

  Math.min( 1337, 42 ), // ?

  Math.max( 1337, 42 ), // ?

  Math.pow( 8, 3 ), // ?

  Math.random(), // ?

);
```

```
let price = 12.5;
console.log(
  price.toFixed(2), // "12,50" (conversion en chaîne)

  Math.min( 1337, 42 ), // 42 (minimum de 2 valeurs)

  Math.max( 1337, 42 ), // 1337 (maximum de 2 valeurs)

  Math.pow( 8, 3 ), // 512 (puissance)

  Math.random(), // un nombre aléatoire entre 0 et 1

);
```

LE TYPE BOOLEAN

déclaration

```
var isThisSpoil = false;  
var isHodorDead = true;
```

Opérateurs logiques

- == === != !== : test de l'égalité (ou non) de valeurs
- < > >= <= : comparaison de valeurs inf. ou sup.
- ?: : Opérateur ternaire a ? b : c;

```
var willJonDieInNextEpisode = Math.random() > 0.5 ? 'yes' : 'no';
```

Notes :

Les opérateurs ternaires sont utilisés soit en remplacement d'instructions 'if' soit pour faciliter des affectations conditionnelles.

LE TYPE ARRAY

déclaration

```
const emptyArray = [];  
  
const brothers = [ 'Robb', 'Bran', 'Rickon' ];
```

accès à une valeur

```
let aliveBrother = brothers[ 1 ]; // "Bran"
```

Notes :

La notation `const a = new Array(b, c, d);` est déconseillée car elle peut conduire à des comportements non souhaités du fait de l'autre constructeur `new Array(length);`

LE TYPE ARRAY

Méthodes

LE TYPE OBJECT

- utilisé en POO (cf. chapitre associé)
- ou comme dictionnaire (tableau associatif)
- classe parente de toutes les autres

OBJET LITTÉRAL

```
const o = {}
```

```
const o = {  
    propriete: valeur,  
    ...  
}
```

```
const arya = {  
    age: 9,  
    name: 'No one',  
    friend: {  
        name: 'Mycah',  
        isDead: true  
    }  
}  
  
console.log( arya.name, arya['name'] ); // "No one" "No one"
```

LE TYPE OBJECT

constructeur Object

```
const o = new Object();  
  
o.propriete = valeur;  
  
o['propriete'] = valeur;
```

```
const arya = new Object();  
arya.age = 9;  
arya.name = 'No one';  
  
const mycah = new Object();  
mycah[ 'name' ] = 'Mycah';  
let propertyName = 'Dead';  
mycah[ 'is' + propertyName ] = true  
  
arya.friend = mycah;
```

LES TYPES VIDÉS

- null
- undefined

```
let hodor;
console.log(
  hodor,          // ?
  null == undefined, // ?
  null === undefined, // ?
  null == false, // ?
  null == 0, // ?
  false == 0, // ?
);
```

```
let hodor;
console.log(
  hodor,          // undefined
  null == undefined, // true
  null === undefined, // false
  null == false, // false
  null == 0, // false
  false == 0, // true
);
```

STRUCTURES DE CONTRÔLE

```
if ( condition ) {
    // traitement si la condition est vraie
} else {
    // traitement dans le cas contraire
}
```

```
switch ( variable ) {
    case value1 :
        // ...
        break;
    case value2 :
        // ...
        break;
    default :
        // ...
        break;
}
```

```
for ( let i = 0 ; i < 10 ; i++ ) {
    // ...
}
```

```
for ( let i = 0 ; i < myArray.length ; i++ ) {
    const value = myArray[i];
    console.log(value);
}
```

```
myArray.forEach( function( value ) {
    console.log(value);
} );
```

```
const arya = {
    age: 9,
    name: 'No one',
    friend: {
        name: 'Mycah',
        isDead: true
    }
}

for ( let key in arya ) {
    console.log( key, arya[key] ); // "age" 9
                                // "name" "No one"
```

```
// name NO one  
// "friend" "[object Object]"  
}
```

FONCTIONS

```
function makeNewEpisode( heroToKill, newCharacters = [] ) {  
    // fonction nommée ...  
    return episode;  
}  
  
const makeNewEpisode = function( heroToKill, newCharacters = [] ) {  
    // fonction anonyme ...  
    return episode;  
}  
  
const makeNewEpisode = ( heroToKill, newCharacters = [] ) => {  
    // arrow function (ES6) ...  
    return episode  
}  
  
const e = makeNewEpisode('Benjen Stark', ['Ed Sheeran', 'ice dragon']);
```

LA CHAÎNE DE PORTÉE

Référence à une variable :

1. Recherche de la variable dans le bloc de code courant
2. Recherche dans le bloc de code parent, et ainsi de suite
3. Recherche dans la portée globale
4. Exception de type ReferenceError

Notes :

La chaîne de portée représente la disponibilité d'une variable dans notre code.

En JavaScript les variables (let, const) ne sont de base accessibles qu'au sein des blocs de code dans lesquels elles sont définies.

LA CHAÎNE DE PORTÉE

```
function a() {  
    let i = 2;  
    function b( i ) {  
        let j = i;  
        if (j < 10) {  
            let k = 12;  
        }  
        return j - k;  
    }  
    const k = 42;  
    console.log( b( i*3 ) );  
    return j;  
}  
a();
```

Quelle valeur s'affiche dans la console ?



HTML

JS

```
1 ▼ function a() {  
2     let i = 2 ;  
3     ▼   function b( i ) {  
4         let j = i;  
5         if (j < 10) {  
6             let k = 12;  
7         }  
8         return j - k;  
9     }  
10    const k = 42;  
11    console.log( b( i*3 ) );  
12    return j;  
13 }  
14 a();
```

Console

Clear

>

[Console](#) [Assets](#) [Comments](#) [Shortcuts](#) Last saved 15 hours ago [Delete](#) [Collections](#) [Embed](#) [Export](#) [Share](#)

Notes :

<http://codepen.io/kumquats/pen/xGpLer?editors=0011> (<http://codepen.io/kumquats/pen/xGpLer?editors=0011>)

TRAVAUX PRATIQUES