



Industrialisation Front-End

Romain Bohdanowicz

Twitter : @bioub - <https://github.com/bioub>

<http://formation.tech/>



Introduction



- Romain Bohdanowicz
Ingénieur EFREI 2008, spécialité en Ingénierie Logicielle
- Expérience
Formateur/Développeur Freelance depuis 2006
Plus de 9 500 heures de formation animées
- Langages
Expert : HTML / CSS / JavaScript / PHP / Java
Notions : C / C++ / Objective-C / C# / Python / Bash / Batch
- Certifications
PHP 5 / PHP 5.3 / PHP 5.5 / Zend Framework 1
- Particularités
Premier site web à 12 ans (HTML/JS/PHP), Triathlète à mes heures perdues
- Et vous ?
Langages ? Expérience ? Utilité de cette formation ?



Front-End IDEs



- ▶ Version orientée Web de IntelliJ IDEA de l'éditeur JetBrains

<https://www.jetbrains.com/webstorm/>

- ▶ Licence : Commercial

Licence entre 35 à 129 euros par an selon le profil et l'ancienneté.

Version d'essai 30 jours.

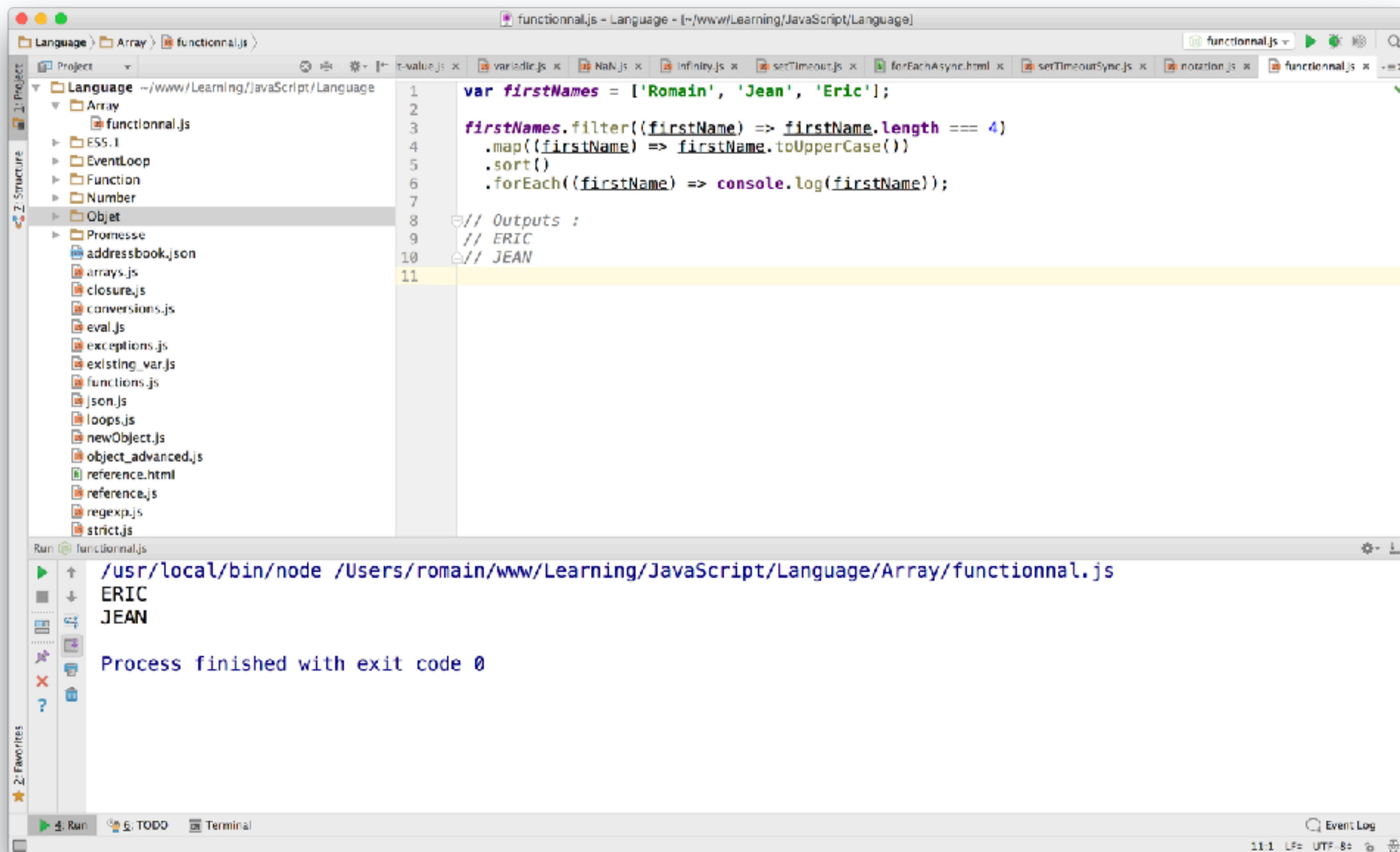
- ▶ Plugins :

Annuaire (642 en novembre 2016) : <https://plugins.jetbrains.com/webStorm>

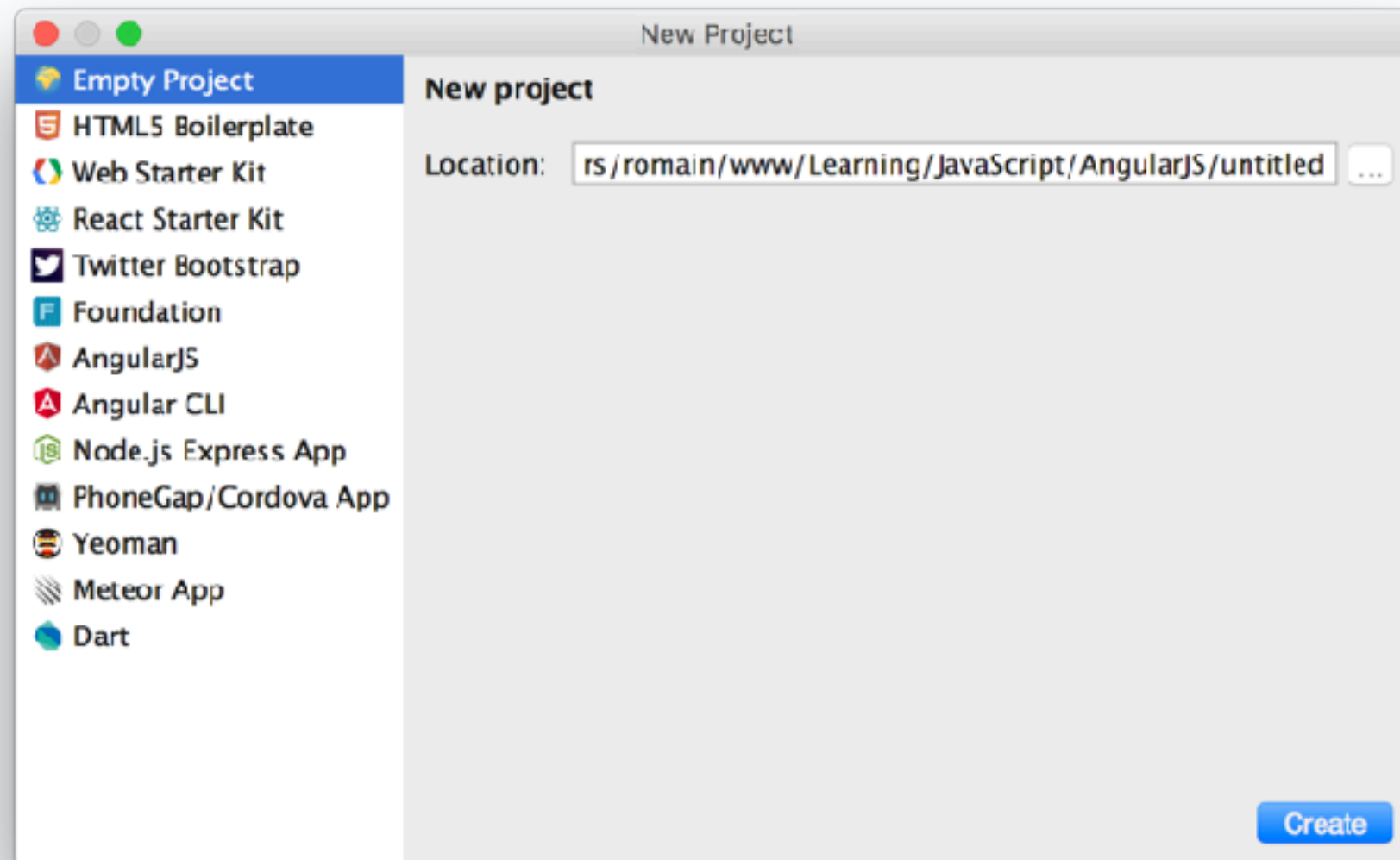
Langage de création : Java



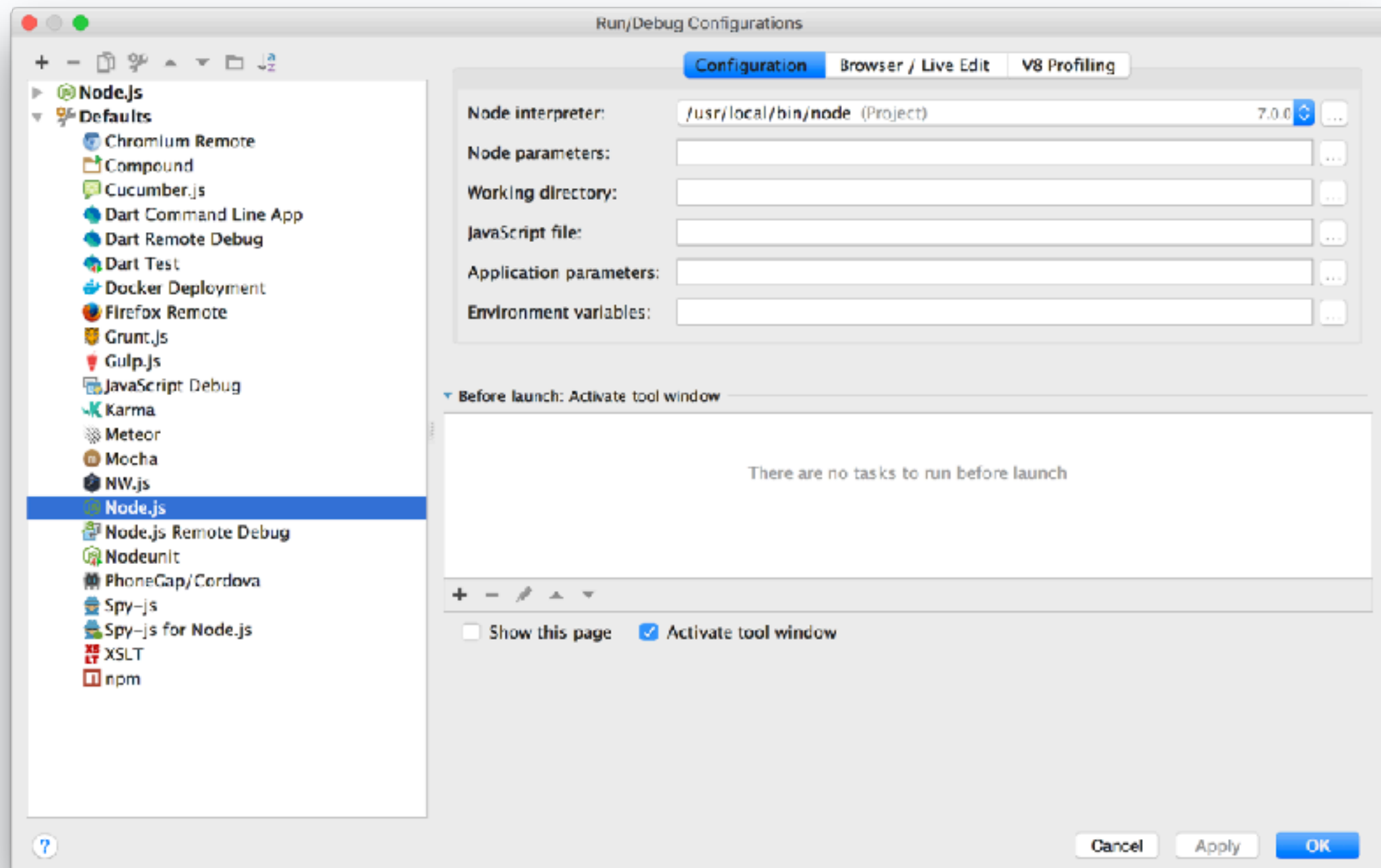
Front-End IDEs - Webstorm



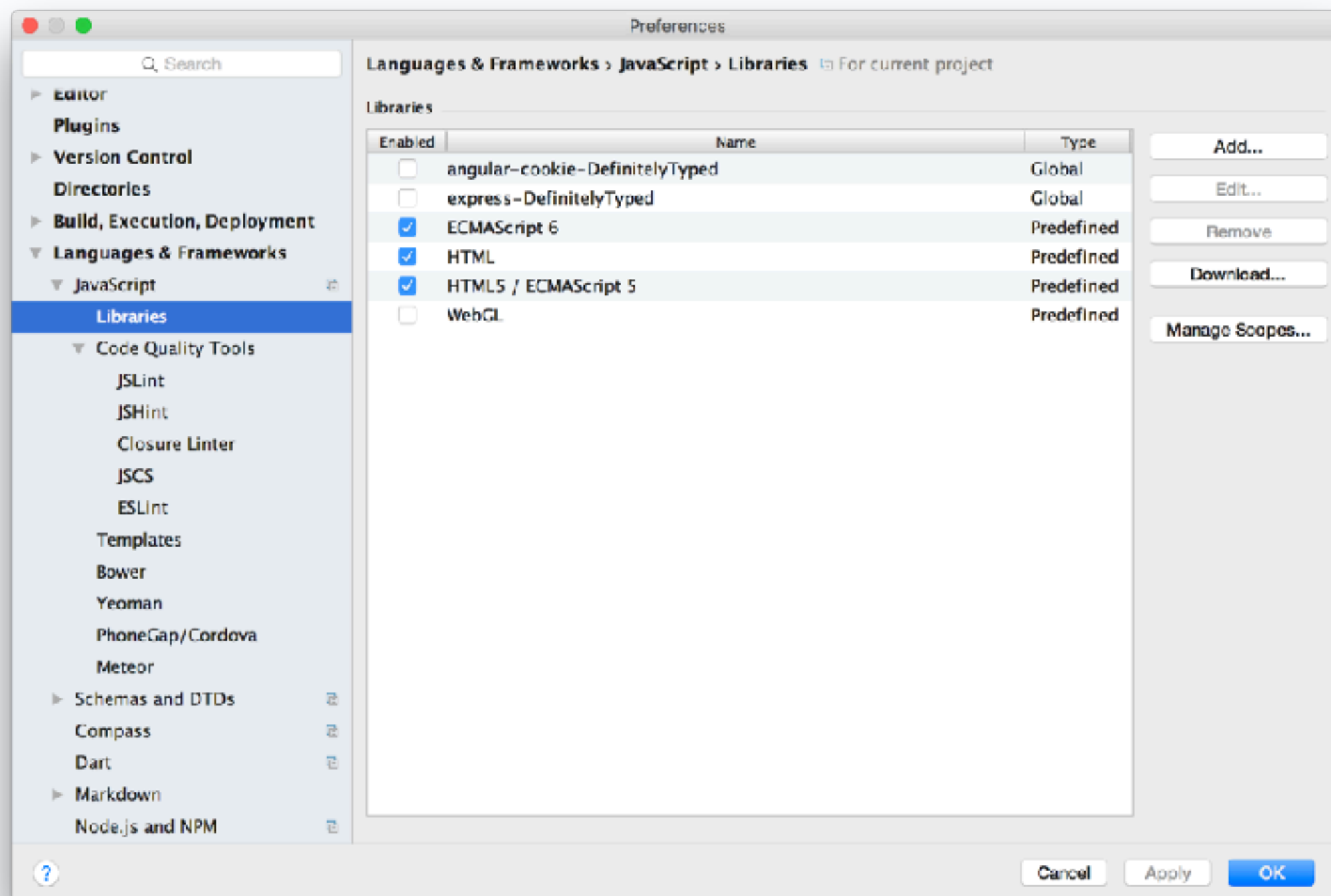
Front-End IDEs - Webstorm



Front-End IDEs - Webstorm



Front-End IDEs - Webstorm





- IDE créé par Github, tourne sous Electron (Chromium + Node.js)
<https://atom.io>
- Licence : MIT
La licence open-source la plus permissive
- Plugins :
Annuaire (5232 en novembre 2016) : <https://atom.io/packages>
Langage de création : JavaScript sous Node.js
Exemples : atom-ternjs, linter, JavaScript Snippets, autocomplete+, autoprefixer...)



Front-End IDEs - Atom



The screenshot shows the Atom IDE interface. On the left is a sidebar with a file explorer showing a project structure. The main editor area displays an HTML file named 'index.html' with the following content:

```
13 <!-- Application Metas End -->
14 <!-- Application Standalone emulation files -->
15 <link rel="stylesheet" href="../../c/UWA/assets/css/standalone.css" />
16 <script src="../../AmdLoader/AmdLoader.js"></script>
17 <script src="../../c/UWA/js/UWA_Standalone_Alonge.js"></script>
18
19 <!-- UIKIT files -->
20 <link rel="stylesheet" href="../../UIKIT/UIKIT.css">
21 <script src="../../UIKIT/UIKIT.js"></script>
22
23 <!-- Application JS Start -->
24 <script>
25   /* global widget, require */
26   require(['DS/ToDoDS/ToDoDS'], function(main) {
27     'use strict';
28
29     var myWidget = {
30
31       //The onLoad() function is the first one,
32       //it will be triggered by widget "onLoad" event.
33       onLoad: function() {
34
35         // Replaces body contents
36         //
37         //widget.body.innerHTML= "Hello World";
38         main(widget.body);
39       }
40     };
41
42     //The "onLoad" event is the very first event triggered when
43     // the widget is fully loaded.
44     widget.addEvent('onLoad', myWidget.onLoad);
45   });
46 </script>
47 <!-- Application JS End -->
48 </head>
49 <body>
```

The bottom status bar shows 'File 0', 'Project 0', 'No Issues', 'ToDoDS/index.html', '1:1', '1 deprecation', 'UTF-8', 'HTML', and '1 update'.



- IDE créé par Microsoft, tourne sous Electron (Chromium + Node.js)

<http://code.visualstudio.com/>

- Licence : MIT

La licence open-source la plus permissive

- Plugins :

Annuaire (1867 en novembre 2016) : <https://marketplace.visualstudio.com/VSCode>

Langage de création : JavaScript sous Node.js

- Documentation

<https://code.visualstudio.com/docs>



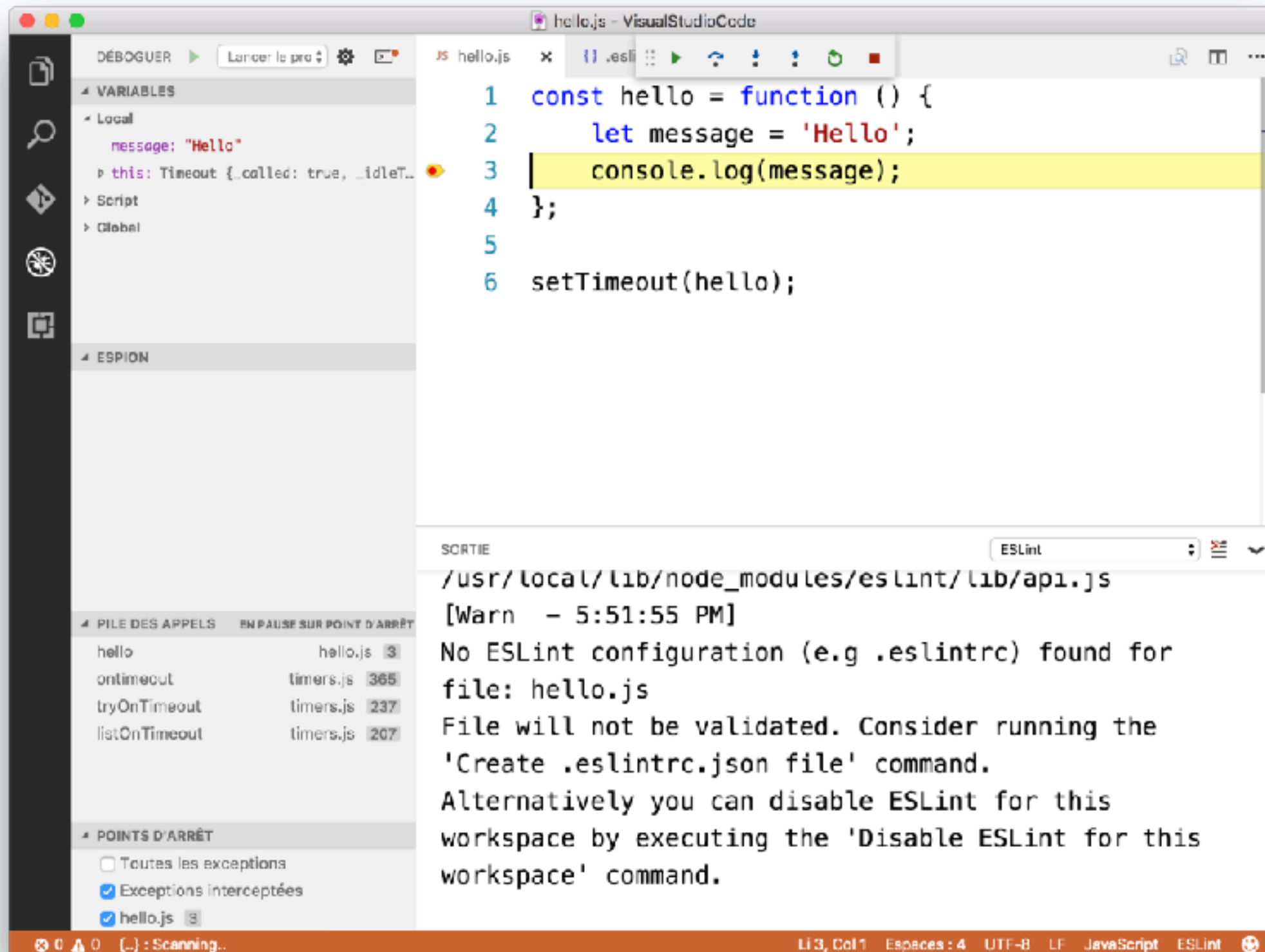
Front-End IDEs - Visual Studio Code

A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows a project structure with folders like 'src' and 'app'. The main editor area displays a TypeScript file named 'about.module.ts' with the following code:

```
1 import { Title } from '@angular/platform-browser';
2 import { NgModule } from '@angular/core';
3
4 import { AboutComponent } from './about.component';
5 import { AboutRoutingModule } from './about-routing.module';
6
7 @NgModule({
8   imports: [
9     AboutRoutingModule
10  ],
11   declarations: [
12     AboutComponent
13  ],
14   providers: [
15     Title
16  ],
17 })
18 export class AboutModule { }
```

The status bar at the bottom indicates the file is at line 18, column 1, using UTF-8 encoding and LF line endings.

Front-End IDEs - Visual Studio Code





- Permet de standardiser les configs des IDEs sur l'indentation et les retours à la ligne
<http://editorconfig.org>
- Supporté par la plupart des IDE
- Il suffit de créer un fichier .editorconfig à la racine d'un projet

```
# EditorConfig is awesome: http://EditorConfig.org

# top-most EditorConfig file
root = true

# Unix-style newlines with a newline ending every file
[*]
end_of_line = lf
insert_final_newline = true
charset = utf-8
indent_style = space
indent_size = 4

# HTML + JS files
[*.{html,js}]
indent_size = 2
```



Git

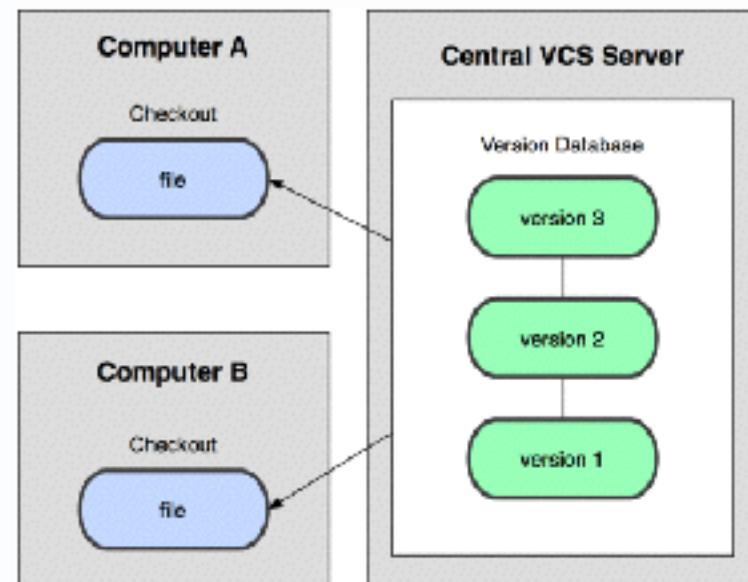


- Système de gestion de version distribué (DVCS)
- Créé par Linus Torvalds en 2005 pour gérer le code source du noyau Linux
- Permet de sauvegarder les différences entre plusieurs versions de fichiers (plutôt texte), principalement du code source
- Facilite la collaborations entre plusieurs développeurs

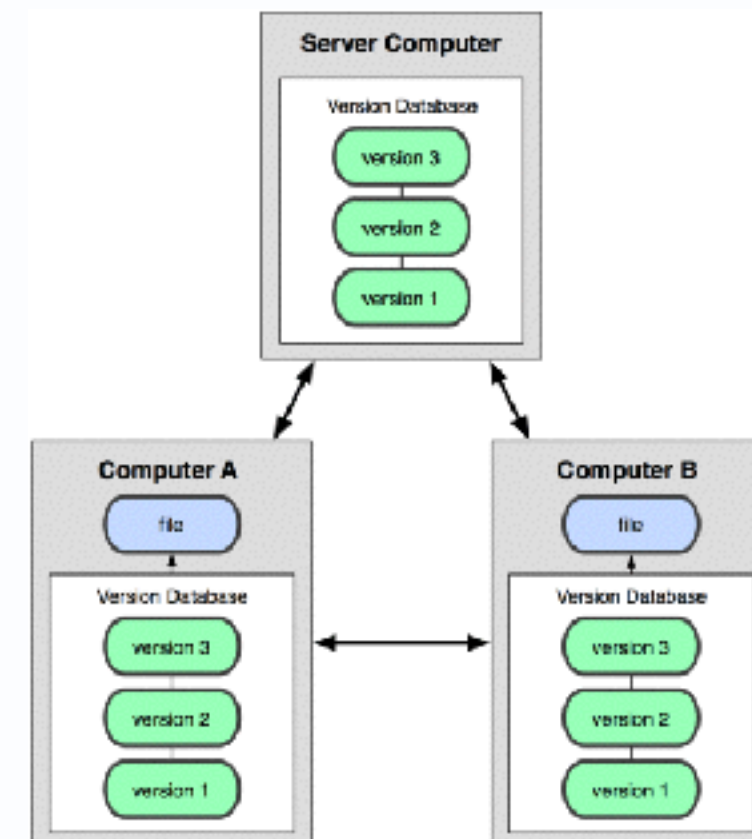
Git - CVCS vs DVCS



- Système de gestion de version centralisé (CVCS)
- Ex : CVS, Subversion



- Système de gestion de version distribué (DVCS)
- Ex : Git, Mercurial



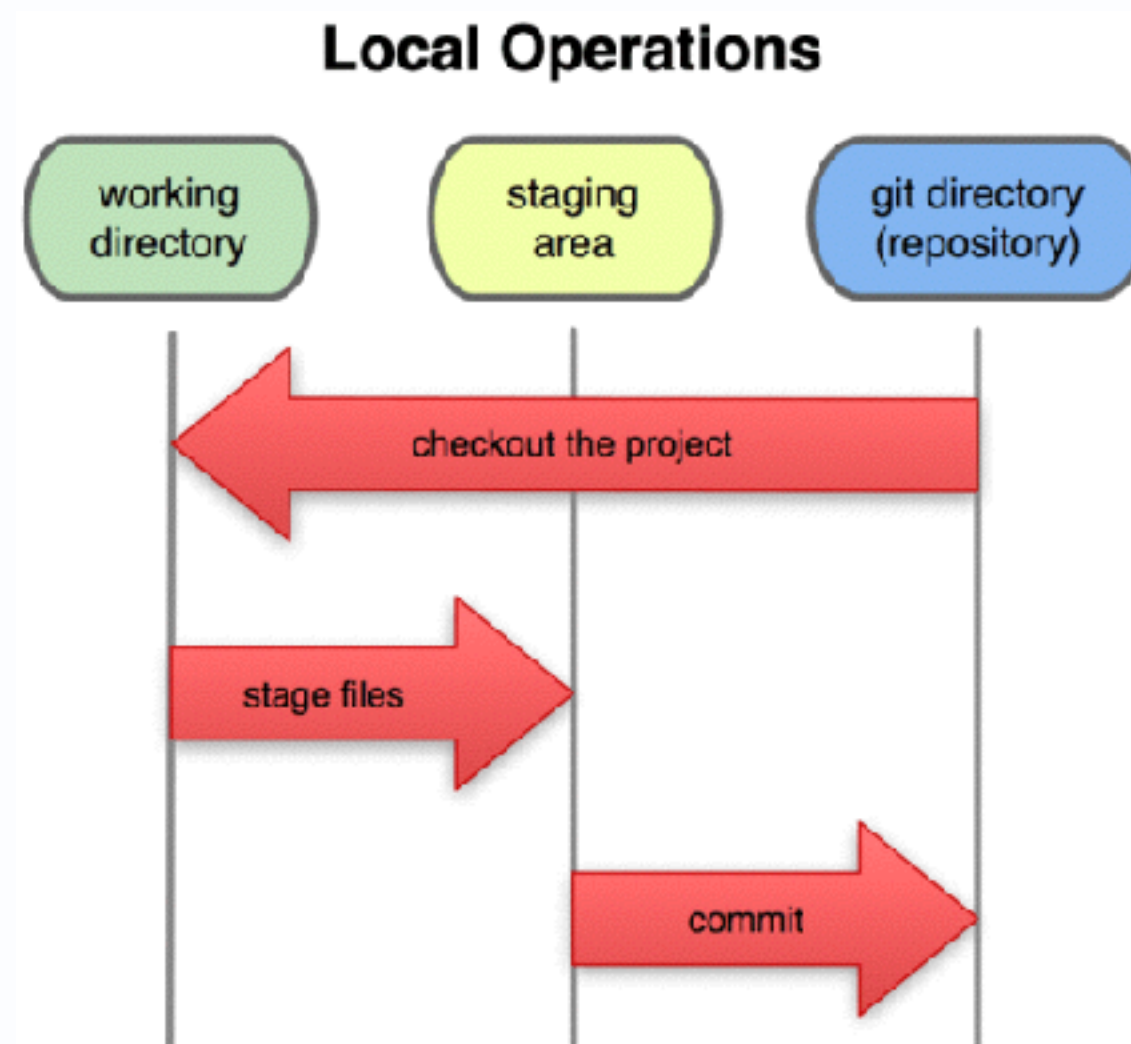


▸ Avantages du DVCS

- l'historique des sources est présent sur plusieurs machines (crash de disque...)
- ne pas avoir à être connecté au réseau pour versionner
- permet de collaborer à un projet et obtenir l'autorisation des mainteneurs à posteriori
- plus rapide (accès locaux)



- 3 états pour les fichiers
 - working directory (non-versionnés)
 - staging (indexés, à publier lors d'un prochain commit)
 - git repository (modifications enregistrées)



Git - Installation



▸ Linux

- `yum install git`
- `apt-get install git`

▸ Mac OS X

- <http://sourceforge.net/projects/git-osx-installer/>
- `brew install git`

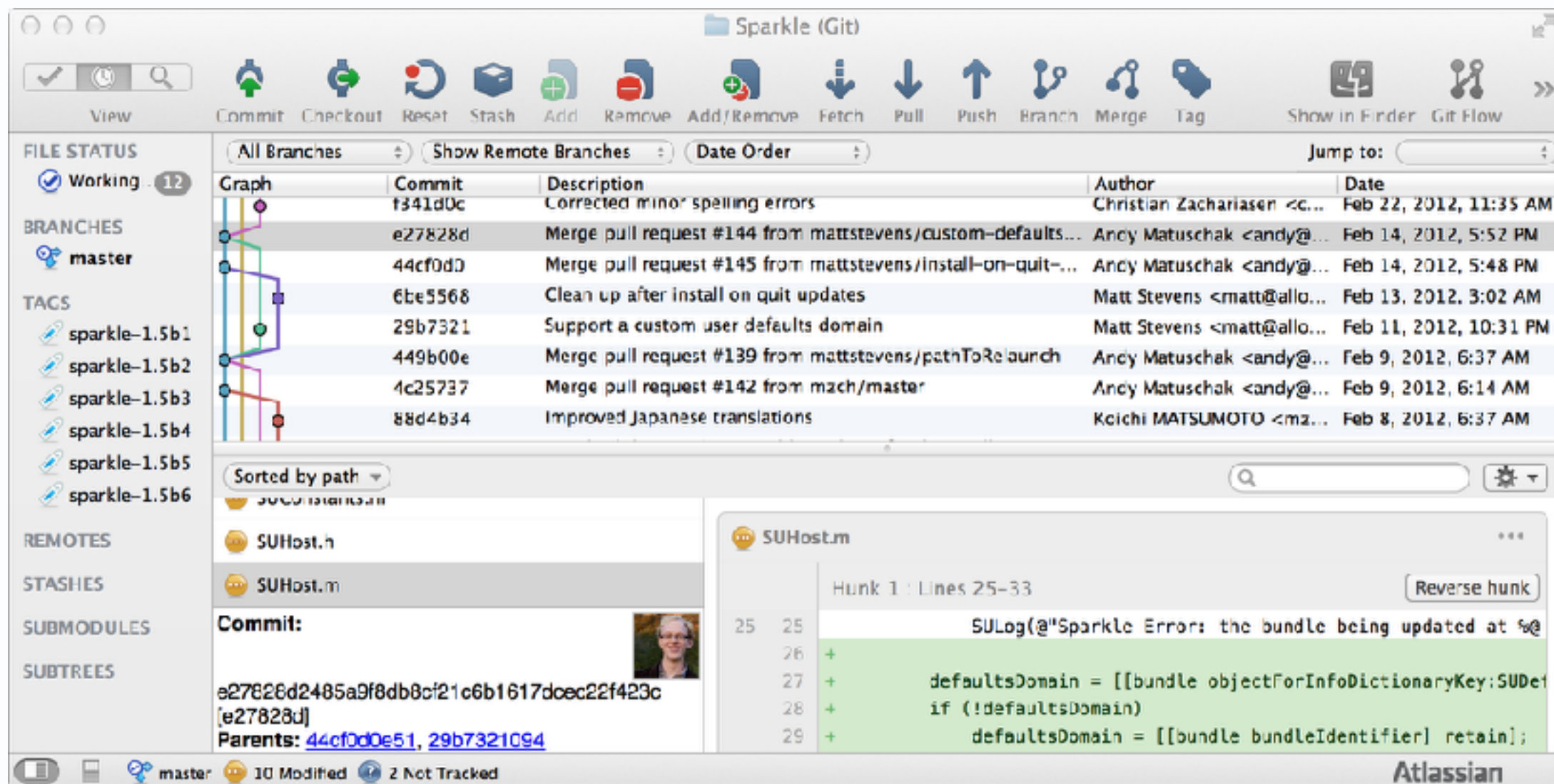
▸ Windows

- <http://msysgit.github.io>



► Mac OS X / Windows

- SourceTree (open-source) : <https://www.sourcetreeapp.com>





- Configurer l'utilisateur
 - `git config --global user.name "John Doe"`
 - `git config --global user.email johndoe@example.com`
- Créer un repository
 - `git init`
- Obtenir le status du projet
 - `git status`

A screenshot of a macOS terminal window titled "Tests-Git — bash — 78x15". The terminal shows the output of the `git status` command. It indicates an initial commit on the master branch with untracked files: README.md, index.html, scripts.js, and style.css. The prompt is `MacBook-Pro-de-Romain:Tests-Git romain$`.

```
MacBook-Pro-de-Romain:Tests-Git romain$ git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        README.md
        index.html
        scripts.js
        style.css

nothing added to commit but untracked files present (use "git add" to track)
MacBook-Pro-de-Romain:Tests-Git romain$
```



- Ajouter des sources (nouvelles ou modifiées) à l'index (add ou son alias stage)
 - `git add *.{css,js,html}`
 - `git stage README.md`
- Versionner
 - `git commit -m "Version initiale du projet"`

```
MacBook-Pro-de-Romain:Tests-Git romain$ git commit -m "Version initiale du projet"
[master (root-commit) f7bcc2b] Version initiale du projet
4 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.md
create mode 100644 index.html
create mode 100644 scripts.js
create mode 100644 style.css
MacBook-Pro-de-Romain:Tests-Git romain$
```




► Cloner un repository existant

Opération lente car il faut télécharger tout l'historique.

- `git clone https://github.com/twbs/bootstrap.git bootstrap-src`

A screenshot of a macOS terminal window titled "Bureau — git — 102x5". The terminal shows the execution of the command `git clone https://github.com/twbs/bootstrap.git bootstrap-src`. The output indicates the cloning process is in progress, with progress bars for counting and compressing objects, and a message for receiving objects at 16% completion.

```
MacBook-Pro-de-Romain:Desktop romain$ git clone https://github.com/twbs/bootstrap.git bootstrap-src
Cloning into 'bootstrap-src'...
remote: Counting objects: 73289, done.
remote: Compressing objects: 100% (10/10), done.
Receiving objects: 16% (12287/73289), 5.64 MiB | 357.00 KiB/s
```

► Ignorer des fichiers

Créer un fichier `.gitignore`

(peut également se faire dans `.git/info/exclude`)

A screenshot of a code editor showing a file named `.gitignore`. The file contains five lines of text, each representing a directory to be ignored: `.idea`, `node_modules`, `bower_components`, `dist`, and `maquette`. The lines are numbered 1 through 5 in the left margin.

```
.gitignore x
1 .idea
2 node_modules
3 bower_components
4 dist
5 maquette
6
```



- Soit le dossier dist, contenant 3 fichiers index.html, scripts.js et style.css
- On vérifie le status (ici non-indexé)
 - git status

```
MacBook-Pro-de-Romain:Tests-Git romain$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    dist/

nothing added to commit but untracked files present (use "git add" to track)
MacBook-Pro-de-Romain:Tests-Git romain$
```

- On l'ajoute à l'index puis commit
 - git add dist (ou git stage dist)
 - git commit -m "Ajout du dossier dist"



- Supprimer le dossier dist, 3 options :
 1. Le supprimer du répertoire de travail (depuis l'explorateur de fichier)
 2. Le supprimer de l'index en ligne de commande
 - `git rm -r dist`
 3. Le supprimer de l'index tout en le conservant dans le répertoire de travail (utile lorsqu'on oublie un fichier dans `.gitignore`)
 - `git rm -r --cached dist/`

```
MacBook-Pro-de-Romain:Tests-Git — bash — 75x12
MacBook-Pro-de-Romain:Tests-Git romain$ git status
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    dist/index.html
        deleted:    dist/scripts.js
        deleted:    dist/style.css

no changes added to commit (use "git add" and/or "git commit -a")
MacBook-Pro-de-Romain:Tests-Git romain$
```

Git - Commandes de bases



- Renommer un fichier
 - `git mv nom_origine nom_cible`
- Historique des modifications
 - `git log`
 - `git log --pretty=format:"%h - %an : %s" --graph`

```
MacBook-Pro-de-Romain:Tests-Git romain$ git log
commit bfbde8b79dab138dfe92b1adb7e9ca1807840d01d
Author: bioub <romain.bondanowicz@gmail.com>
Date:   Wed Aug 19 15:03:48 2015 +0200

    Suppression du dossier dist

commit dfef5509fc89517127941f33edb28a74dde6895a
Author: bioub <romain.bondanowicz@gmail.com>
Date:   Wed Aug 19 14:42:48 2015 +0200

    Ajout du dossier dist

commit f7bcc2b7d485db6011bb375e38a18d019a9bd17d
Author: bioub <romain.bondanowicz@gmail.com>
Date:   Wed Aug 19 14:19:17 2015 +0200

    Version initiale du projet
MacBook-Pro-de-Romain:Tests-Git romain$ git log --pretty=format:"%h - %an : %s" --graph
* bfbde8b - bioub : Suppression du dossier dist
* dfef550 - bioub : Ajout du dossier dist
* f7bcc2b - bioub : Version initiale du projet
MacBook-Pro-de-Romain:Tests-Git romain$
```



- ▶ Modifier le dernier commit
 - `git commit -m 'validation initiale'`
 - `git add fichier_oublie`
 - `git commit —amend`
- ▶ Désindexer un fichier déjà indexé
 - `git reset HEAD fichier_a_desindexer`
- ▶ Réinitialiser un fichier modifié
 - `git checkout fichier_a_reinitialiser`



- Ajouter un dépôt distant
 - `git remote add origin https://github.com/bioub/tests-git.git`
- Listes les dépôts distants
 - `git remote -v`
- Publier des sources sur un dépôt distant
 - `git push origin master`
(origin : nom du dépôt distant, master : branche locale)
- Récupérer les sources depuis un dépôt distant
 - `git pull origin master`



- Lister les tags
 - `git tag`
- Créer un nouveau tag
 - `git tag -a 0.9.0 -m "Version 0.9.0"`
- Tagger un précédent commit
 - `git tag -a 0.1.0 -m "Version 0.1.0" f7bcc2b2d`

A screenshot of a macOS terminal window titled "Tests-Git — bash — 87x6". The window shows the output of the command `git log --pretty=oneline`, displaying five commits with their hashes and messages. The final command shown is `git tag -a 0.1.0 -m "Version 0.1.0" f7bcc2b2d`.

```
MacBook-Pro-de-Romain:Tests-Git romain$ git log --pretty=oneline
0cefc8f807bb14adb362f55f3a85a396cb205b9a Ajout d'une ligne depuis github
bfbde8b99ab138dfe92b1adb7e9ca1007840d91d Suppression du dossier dist
dfef5500fc89517127944f33edb28a74dde6895a Ajout du dossier dist
f7bcc2b2d485db6011bb375e38a18d019a9bd17d Version initiale du projet
MacBook-Pro-de-Romain:Tests-Git romain$ git tag -a 0.1.0 -m "Version 0.1.0" f7bcc2b2d
```

- Partager les tags au serveur distant
 - `git push origin --tags`



- Créer une branche
 - `git branch fonc12`
(fonc12 : le nom de la nouvelle branche « fonctionnalité 12 »)
- Changer de branche
 - `git checkout fonc12`
- Créer et changer de branche
 - `git checkout -b fonc12`
- Récupérer les sources dans une nouvelle branch
 - `git fetch origin`



- ▶ Fusionner une branche dans la branche courante
 - `git merge func12`
- ▶ 3 cas possibles
 - Fast-forward (pas de commit sur la branche d'origine entre temps)

```
MacBook-Pro-de-Romain:Tests-Git$ git merge func12
Updating 0cefc8f..795dca3
Fast-forward
 scripts.js | 1 +
 1 file changed, 1 insertion(+)
```

```
MacBook-Pro-de-Romain:Tests-Git$ git log --pretty=format:"%h - %an : %s" --graph
* 4e6fac9 - bioub : Ajout d'un fond pour body
* 795dca3 - bioub : Ajout d'une ligne dans scripts.js
* 0cefc8f - Romain Bohdanowicz : Ajout d'une ligne depuis github
* bfbde8b - bioub : Suppression du dossier dist
* dfeF550 - bioub : Ajout du dossier dist
* f7bcc2b - bioub : Version initiale du projet
```

- Recursive

```
MacBook-Pro-de-Romain:Tests-Git$ git merge func12
Merge made by the 'recursive' strategy.
 scripts.js | 1 +
 1 file changed, 1 insertion(+)
```

```
MacBook-Pro-de-Romain:Tests-Git$ git log --pretty=format:"%h - %an : %s" --graph
* 32ab820 - bioub : Merge branch 'func12'
|
| * e8eaf16 - bioub : Ajout d'un log dans scripts.js
| | 4e6fac9 - bioub : Ajout d'un fond pour body
| |
| * 795dca3 - bioub : Ajout d'une ligne dans scripts.js
* 0cefc8f - Romain Bohdanowicz : Ajout d'une ligne depuis github
* bfbde8b - bioub : Suppression du dossier dist
* dfeF550 - bioub : Ajout du dossier dist
* f7bcc2b - bioub : Version initiale du projet
```

Git - Branches



- Conflit

```
MacBook-Pro-de-Romain:Tests-Git romain$ git merge func12
Auto-merging style.css
CONFLICT (content): Merge conflict in style.css
Automatic merge failed; fix conflicts and then commit the result.
MacBook-Pro-de-Romain:Tests-Git romain$
```

Corriger le fichier qui pose problème (penser

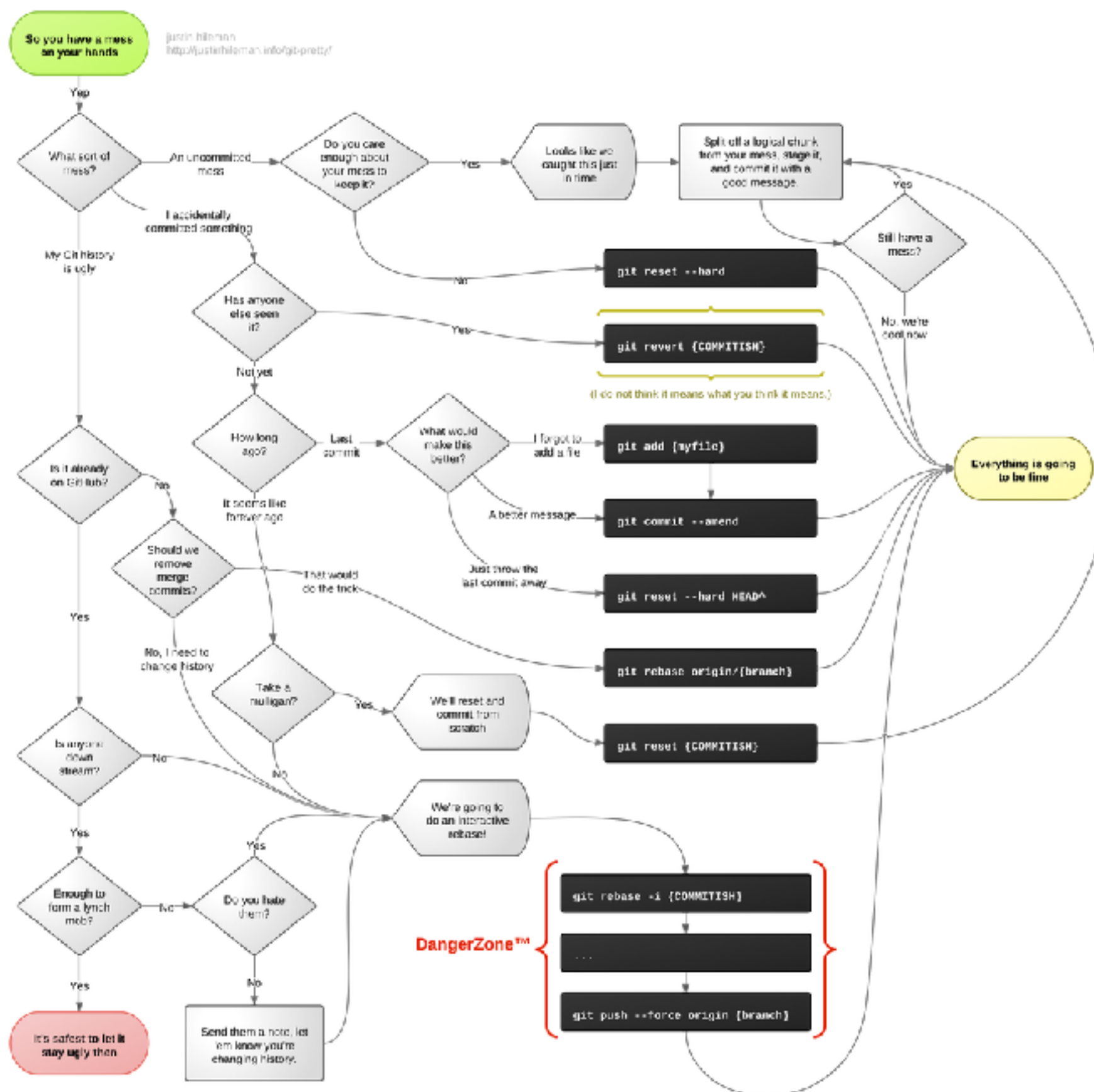
```
style.css
1 body {
2   background-color: #eee;
3   <<<<<<< HEAD
4   color: #222;
5   =====
6   color: #000;
7   >>>>>>> func12
8 }
9
```

<, ===== et >>>>)

Puis commit

```
MacBook-Pro-de-Romain:Tests-Git romain$ git log --pretty=format:"%h - %cn : %s" --graph
* 0ecc82e - bioub : Résolution des conflits (texte finalement en gris)
| \
| * b7c733a - bioub : Texte en noir
| * ef1500c - bioub : Texte en gris
| * 32ab820 - bioub : Merge branch 'func12'
| \
| \
| * e8eaf16 - bioub : Ajout d'un log dans scripts.js
| * 4e5fac9 - bioub : Ajout d'un fond pour body
| /
* 795dca3 - bioub : Ajout d'une ligne dans scripts.js
* 0cefc8f - Romain Bohdanowicz : Ajout d'une ligne depuis github
* bfbde8b - bioub : Suppression du dossier dist
* dfef550 - bioub : Ajout du dossier dist
* f7bcc2b - bioub : Version initiale du projet
```

Git - Branches





- ▶ Bonnes pratiques de gestion de branches

<http://nvie.com/posts/a-successful-git-branching-model/>

- ▶ Mise en place simplifiée de ces pratiques

http://danielkummer.github.io/git-flow-cheatsheet/index.fr_FR.html



npm



- Un gestionnaire de dépendances est un programme qui lance le téléchargement d'une bibliothèque dont dépend votre code, mais également de manière récursive toutes les bibliothèques dont dépendent la bibliothèque installée.
- Equivalent pour du code JavaScript à apt-get
- Principaux gestionnaires de dépendances :
 - Java : Maven, Gradle
 - Ruby : Bundler, gem
 - Python : pip
 - C# : nuget
 - PHP : composer, PEAR
 - Swift / Objective C : CocoaPods
 - JavaScript : npm, yarn, Bower, jspm, pnpm

npm - Gestion de dépendances



- Il existe plusieurs programme pour la gestion de dépendances en JavaScript
 - npm
 - <https://www.npmjs.com/>
 - La norme, s'installe en même temps de Node.js.
 - Yarn
 - <https://yarnpkg.com/>
 - Développé par Facebook, a une époque considéré comme plus sécurisé et moins exposé au bugs que npm. La version 5 de npm a comblé son retard. Yarn garde l'avantage de paralléliser les téléchargements.
 - pnpm
 - <https://pnpm.js.org/>
 - Ultra-rapide et moins gourmand en disque, n'installe les dépendances qu'une fois par machine, puis des liens symboliques dans les projets
- Benchmarks
 - <https://github.com/pnpm/node-package-manager-benchmark>



- jspm

<https://jspm.org/>

Permet historiquement d'utiliser facilement des modules ES6 avec SystemJS.
Pas recommandé pour de nouveaux projets.

- bower

<https://bower.io/>

Développé par Twitter, historiquement adapté aux dépendances front-end. Pas recommandé pour de nouveaux projets.

▸ Annuaire

- Il existe 2 annuaires de dépendances, npm et bower. Yarn, pnpm et jspm utilisent celui

npm - Où trouver des bibliothèques ?



▸ Sur le registre npm

- Moteur de recherche
<https://www.npmjs.com>
<https://npmsearch.com>
- Regarder les stats d'un paquet :
<https://www.npmjs.com/package/bootstrap>
- Paquets npm les plus utilisées (en nombre de dépendances)
<https://www.npmjs.com/browse/depended>

▸ Sur GitHub

- Recherche par nombre d'étoiles :
<https://github.com/search?q=stars%3A%3E0>
- Explorer les projets mis en avant par GitHub
<https://github.com/explore>
- Projets qui reçoivent en ce moment le plus d'étoiles
<https://github.com/trending>

npm - Où trouver des bibliothèques ?



▸ Awesome Lists

<https://github.com/sindresorhus/awesome>

- Awesome Node.js

<https://github.com/sindresorhus/awesome-nodejs>

- Awesome Frontend

<https://github.com/dypsilon/frontend-dev-bookmarks>

- Awesome React

<https://github.com/enaqx/awesome-react>

- Awesome Angular

<https://github.com/gdi2290/awesome-angular>

▸ Autres

- <https://bestof.js.org/>

- <https://npmcharts.com/>



- Gestionnaire de dépendance de Node.js (en général installé en même temps que Node.js)
- A l'origine, plutôt destiné à du code console ou serveur, bien que des bibliothèques comme jQuery ou Bootstrap y soient présentes depuis toujours
- Aujourd'hui le programme le plus complet mais pas le plus rapide
- Les programmes npm, yarn, pnpm sont incompatibles entre eux, il faut en choisir un dans un projet et s'y tenir



npm - Le fichier package.json



- Aujourd'hui le fichier package.json est le coeur de la configuration d'un projet JavaScript
- On y retrouve principalement
 - Nos dépendances de prod
 - Nos dépendances de dev
 - Les scripts permettant d'interagir avec le projet (build, serveur de dev, tests...)
 - Des noms, descriptions, auteurs, versions, dépôt git en cas de publication sur npm
 - De la config pour le projet où des outils du projets (linters, tests, builders...)
 - ...

npm - Le fichier package.json



- Création d'un fichier package.json minimal
- PAS DE COMMENTAIRES DANS UN FICHIER JSON !

```
{}
```

- Pour le créer en ligne de commande
 - Dans un projet (en mode interactif)
`npm init`
 - Dans un projet (en forçant yes à toutes les questions)
`npm init -fy`
 - Pour créer un projet depuis un paquet create-* (ici un paquet create-react-app)
`npm init react-app mon-app-react`
 - Pour créer un projet en lançant une commande
`npx create-react-app mon-app-react`

npm - Installation d'un nouveau paquet



- Installer un paquet
`npm install jquery --save`
- Installer un paquet (alias, --save par défaut depuis npm 5)
`npm i jquery`
- Installer un paquet de dev
`npm i jquery --save-dev`
- Installer un paquet de dev (alias)
`npm i jquery -D`
- Installer une version ancienne d'un paquet
`npm i jquery@1`
`npm i jquery@1.12`
`npm i jquery@1.12.0`
- Installer ou réinstaller la dernière version d'un paquet (dist-tag latest)
`npm i jquery@latest`
- D'autre dist-tags peuvent exister (cf la doc du paquet)
`npm i jquery@next`

npm - Lister les dépendances



- Lister les dépendances

`npm list`

```
├─┬ react@16.4.2
│   └─┬ fbjs@0.8.17
│       ├── core-js@1.2.7
│       ├── isomorphic-fetch@2.2.1
│       │   └─┬ node-fetch@1.7.3
│       │       └─┬ encoding@0.1.12
│       │           └─┬ iconv-lite@0.4.24 deduped
│       │               └─┬ is-stream@1.1.0
│       │                   └─┬ whatwg-fetch@2.0.4
│       └─┬ loose-envify@1.4.0 deduped
│           ├── object-assign@4.1.1 deduped
│           ├── promise@7.3.1
│           └─┬ asap@2.0.6 deduped
```

- deduped signifie que la dépendance est partagée par d'autre (se retrouve à la racine de node_modules)
- Lister que nos dépendances directes
`npm list --depth 0`

npm - Détecter des dépendances à mettre à jour



- Savoir quoi mettre à jour
npm outdated
- 3 cas possibles
 - rouge : mise à jour dispo
 - jaune : migration dispo
 - à jour : n'apparaît pas

```
MacBook-Pro:front-end-scs-angular romain$ npm outdated
Package      Current  Wanted  Latest  Location
@angular-devkit/build-angular 0.6.3    0.6.8    0.7.5  front-end-scs-angular
@angular/animations      6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/cli              6.0.3    6.0.3    6.1.5  front-end-scs-angular
@angular/common          6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/compiler        6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/compiler-cli     6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/core             6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/forms            6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/http             6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/language-service 6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/platform-browser 6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/platform-browser-dynamic 6.0.2    6.0.2    6.1.6  front-end-scs-angular
@angular/router           6.0.2    6.0.2    6.1.6  front-end-scs-angular
@ng-select/ng-select      2.0.3    2.6.0    2.6.0  front-end-scs-angular
@ngx-translate/core       10.0.1   10.0.2   10.0.2  front-end-scs-angular
```

- Versionnage sémantique
<https://semver.org/>
- MAJOR.MINOR.PATCH (ex : 1.2.3)
- New PATCH : pas de changement de comportement (API ou interface identiques)
- New MINOR : changements rétro-compatibles
- New PATCH : changements rétro-incompatibles (lire le guide de migration)



- Le fichier `package.json` permet de verrouiller
 - Une version majeure : `^1.2.3`
 - Une version mineure : `~1.2.3`
 - Une version patch : `1.2.3`
- Le fichier `package-lock.json` permet de verrouiller également les dépendances de ce paquet de telle sorte que tous les environnements (postes de dev, prod...) partagent les mêmes versions
- Le lockfile permet également de garder l'URL et le checksum de chaque dépendance du projet (perf + sécurité)
- npm maintient un cache sur le disque, les installations ultérieures seront plus rapides
- Les lockfiles de npm, yarn et pnpm sont incompatibles entre eux (parfois même entre 2 versions d'npm !)

npm - Mettre à jour



- Mettre à jour tous les paquets packages installés
`npm update`
- Mettre à jour un seul paquet
`npm update jquery`
- Migrer un paquet vers une version majeure
`npm i jquery@2`
- Migrer un paquet vers la dernière version majeure
`npm i jquery@latest`
- Désinstaller
`npm uninstall lodash`

npm - Lancer un script



- Lancer un script
`npm run-script nom-du-script`
- Lancer un script (alias)
`npm run nom-du-script`
- Certains script on des alias (start, test...)
`npm run-script test`
`npm run test`
`npm test`
`npm t`
- Certains script peuvent s'exécuter automatiquement
postinstall, prepublish, pretest...
- Lancer un script permet d'exécuter un programme local

npm - Configuration



- Utilisation d'un proxy
npm config set proxy http://host:8080
npm config set proxy http://user:pass@host:8080
- Supprimer une config
npm config rm proxy
- Lister les configs
npm config list



Frameworks HTML/CSS



- Popularisé lorsque Twitter a proposé sa bibliothèque UI en open source sous le nom de Bootstrap en 2011
- Unifie et accélère le développement, la majeure partie du CSS est déjà développée
- Inverse les responsabilités : le HTML fait la mise en forme en s'intégrant à un CSS existant

Frameworks HTML/CSS - Bootstrap



- Créé par Twitter
- Open Source depuis 2011
- Projet le plus populaire sur GitHub
Contributeurs : 658 - Watches : 5092 - Stars 83109 - Forks 33538 (juillet 2015)
- Ecrit avec jQuery, Less (Sass depuis la v4), QUnit, Grunt...
- Documentation
<http://getbootstrap.com>
- Support
 - (v3) : IE8 avec HTML5 shim et Respond.js
 - (v4) : IE10+

Frameworks HTML/CSS - Bootstrap



- Téléchargement :
<https://github.com/twbs/bootstrap/archive/v3.3.7.zip>
- CDN
<https://www.bootstrapcdn.com>
- Git
git clone <https://github.com/twbs/bootstrap.git>
- Bower
bower install bootstrap
- npm
npm install bootstrap
- Meteor
meteor add twbs:bootstrap
- Composer
composer require twbs/bootstrap



- Nécessite jQuery + HTML5 shim et Respond.js (IE8)

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <!-- The above 3 meta tags *must* come first in the head; any other head content must come
  *after* these tags -->
  <title>Bootstrap 101 Template</title>

  <!-- Bootstrap -->
  <link href="css/bootstrap.min.css" rel="stylesheet">

  <!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
  <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
  <!--[if lt IE 9]>
  <script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>
  <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
  <![endif]-->
</head>
<body>
<h1>Hello, world!</h1>

<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<!-- Include all compiled plugins (below), or include individual files as needed -->
<script src="js/bootstrap.min.js"></script>
</body>
</html>
```



- Mise en forme de balises existantes (bouton, formulaires...)
- Inclus Normalize.css
- Composants mis en forme :
 - Container
 - Grid system
 - Typography
 - Code
 - Tables
 - Forms
 - Buttons
 - Images
 - Helper classes
 - Responsive utilities

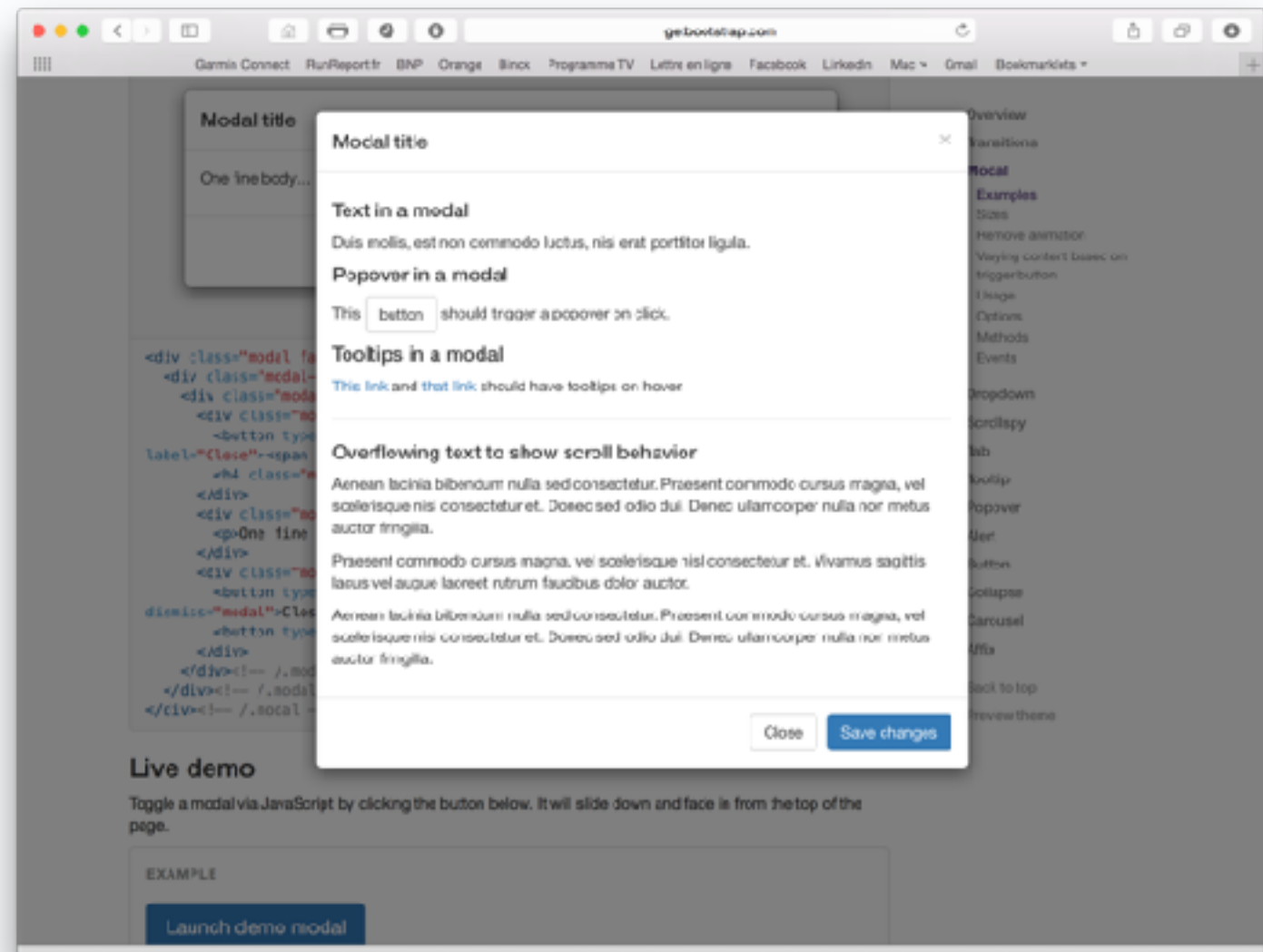


- HTML + CSS de composants plus haut niveau
- Composants :
 - Glyphicons
 - Dropdowns
 - Button groups
 - Button dropdowns
 - Input groups
 - Navs
 - Navbar
 - Breadcrumbs
 - Pagination
 - Labels
 - Badges
 - Jumbotron
 - Page header
 - Thumbnails
 - Alerts
 - Progress bars
 - Media object
 - List group
 - Panels
 - Responsive embed
 - Wells

Frameworks HTML/CSS - Bootstrap



- Plugins pour jQuery
- Composants :
 - Transitions
 - Modal
 - Dropdown
 - Scrollspy
 - Tab
 - Tooltip
 - Popover
 - Alert
 - Button
 - Collapse
 - Carousel
 - Affix





- 2e framework HTML/CSS/JS sur GitHub

- Documentation

<http://foundation.zurb.com>

- Stats Github :

Contributeurs : 705 - Watches : 1431 - Stars 20611 - Forks 4394 (juillet 2015)

- Ecrit avec jQuery, SASS, Jasmine, Grunt...

- Support : IE9+



Frameworks HTML/CSS - Semantic UI



- 3e framework HTML/CSS/JS sur GitHub
- Documentation
<http://semantic-ui.com>
- Stats Github :
Contributeurs : 111 - Watches : 994 - Stars 19191 - Forks 2129 (juillet 2015)
- Ecrit avec jQuery, LESS, Jasmine, Gulp...
- Support : Last 2 Versions FF, Chrome, IE 10+, Safari Mac





Modules/Loaders/Bundlers JavaScript



Modules JavaScript



▸ JavaScript à sa conception

- Objectif : créer des interactions côté client, après chargement de la page
- Exemples de l'époque :
 - Menu en rollover (image ou couleur de fond qui change au survol)
 - Validation de formulaire

▸ JavaScript aujourd'hui

- Applications front-end, back-end, en ligne de commande, de bureau, mobiles...
- Applications pouvant contenir plusieurs centaines de milliers de lignes de codes (Front-end de Facebook > 1 000 000 LOC)
- Il faut faciliter le travail collaboratif, en plusieurs fichiers et en limitant les risques de conflit

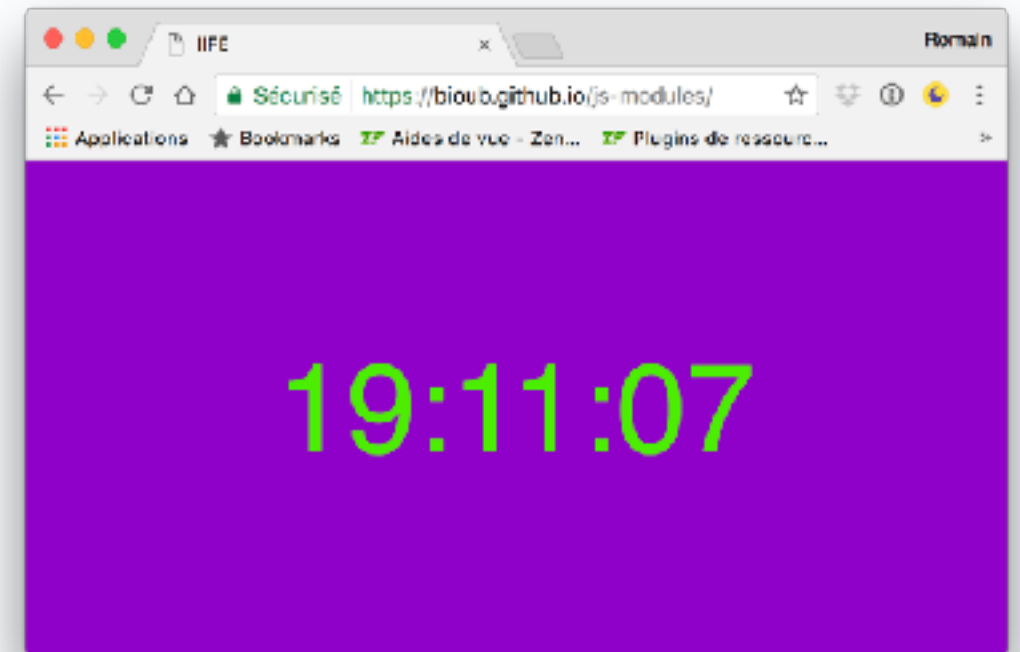
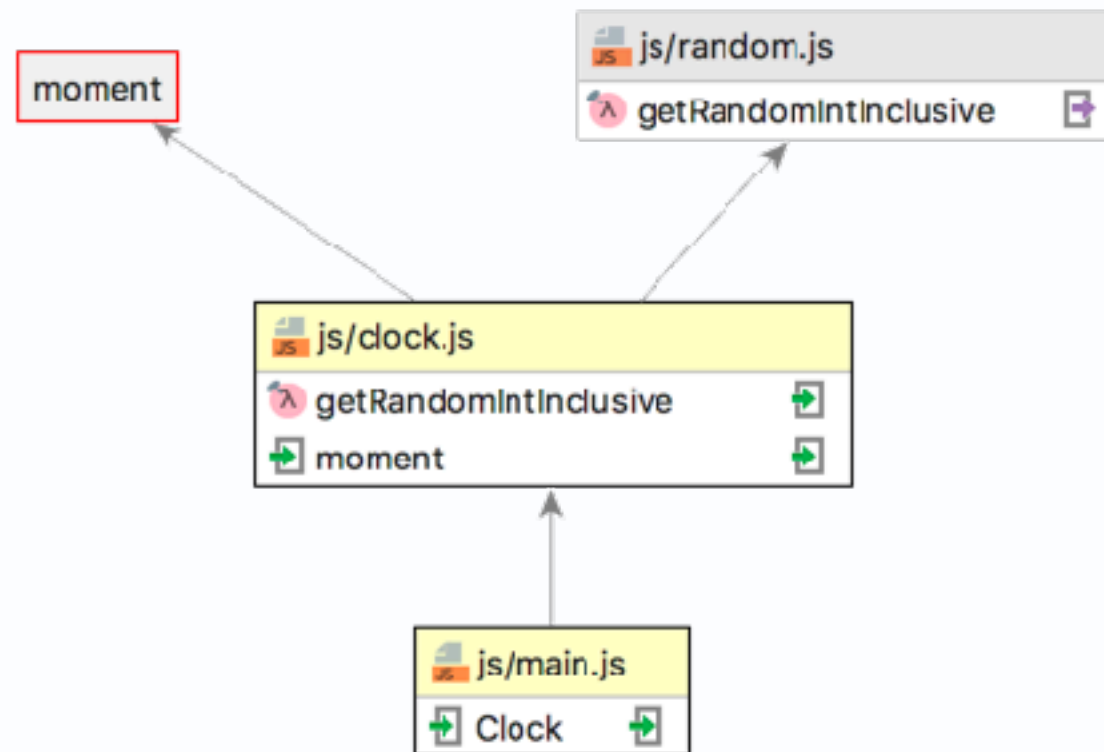


- Objectifs d'un module JavaScript
 - Créer une portée au niveau du fichier
 - Permettre l'export et l'import d'identifiants (variables, fonctions...) entre ces fichiers qui auront désormais leur propre portée
- Principaux systèmes existants
 - IIFE / Function Wrapper
 - CommonJS
 - AMD
 - UMD
 - SystemJS
 - ES6 (statiques mots clés import / export)
 - ESNext : import() (fonction asynchrone)

Modules JavaScript - Introduction



- ▶ Exemple utilisé pour la suite



- ▶ Le point d'entrée de l'application est le fichier `main.js`, qui dépend de `Clock` défini dans le fichier `clock.js`, qui dépend lui-même de `getRandomIntInclusive` du fichier `random.js` et `moment` défini dans le projet Open Source `moment.js`
- ▶ Exemples : <https://github.com/bioub/js-modules>
- ▶ Démo : <https://bioub.github.io/js-modules/>



► Immediately-invoked function expression (IIFE)

```
// random.js
(function (global) {
  'use strict';

  var getRandom = function() {
    return Math.random();
  };

  var getRandomIntInclusive = function(min, max) {
    min = Math.ceil(min);
    max = Math.floor(max);
    return Math.floor(getRandom() * (max - min + 1)) + min;
  };

  global.getRandomIntInclusive = getRandomIntInclusive;
})(this);
```

- Une function expression appelée immédiatement
 - Limite la portée des identifiants (getRandom, getRandomIntInclusive)
 - Permet de renommer localement des dépendances (this → global)



▸ Import / Export en IIFE

```
// clock.js
(function(global, moment, random) {
  'use strict';

  var Clock = function(parentElt) {
    this.parentElt = parentElt;
  };

  Clock.prototype.update = function() {
    var n = random(0, 255);
    document.body.style.backgroundColor = 'rgb('+n+', '+n+', '+n+')';
    this.parentElt.innerHTML = moment().format('HH:mm:ss');
  };

  Clock.prototype.start = function() {
    this.update();
    setInterval(this.update.bind(this), 1000);
  };

  global.Clock = Clock;
})(this, moment, getRandomIntInclusive));
```

- Pour importer on utilise des variables globales, éventuellement en paramètres d'entrée de la fonction pour pouvoir les renommer localement (getRandomIntInclusion → random)
- Pour exporter on crée des variables globales en étendant l'objet global



- ▶ Limiter les risques de conflits en IIFE

Exemple : API Maps de Google

```
<!DOCTYPE html>
<html>
<head></head>
<body>
<div id="map"></div>
<script>
  function initMap() {
    var uluru = {lat: -25.363, lng: 131.044};
    var map = new google.maps.Map(document.getElementById('map'), {
      zoom: 4,
      center: uluru
    });
    var marker = new google.maps.Marker({
      position: uluru,
      map: map
    });
  }
</script>
<script async defer src="https://maps.googleapis.com/maps/api/js?callback=initMap"></script>
</body>
</html>
```



► Limiter les risques de conflits en IIFE

```
(function(global, google) {  
  'use strict';  
  
  var GMap = function (parentElt, options) {  
    // ...  
  };  
  
  var GMarker = function (parentElt, options) {  
    // ...  
  };  
  
  global.google = google || {};  
  google.maps = google.maps || {};  
  google.maps.Map = GMap;  
  google.maps.Marker = GMarker;  
  
})(global, google);
```

- Pour limiter les conflits de nom on simule des namespaces.
- Il ne faut créer qu'une seule variable globale du nom de la société, nom du projet... (google dans l'exemple)



► Utilisation dans le Navigateur

```
<!DOCTYPE html>
<html lang="en">
<head></head>
<body>
  <div class="clock"></div>
  <script src="node_modules/moment/moment.js"></script>
  <script src="js/random.js"></script>
  <script src="js/clock.js"></script>
  <script src="js/main.js"></script>
</body>
</html>
```

- Avec les modules IIFE c'est au développeur de créer les balises script
- Les imports/exports se faisant via des variables globales, il faut maintenir ces balises dans l'ordre des dépendances

► Utilisation dans Node.js

- Node.js incluant son propre système de module, il n'est pas recommandé d'utiliser des modules IIFE



▸ CommonJS (parfois CJS)

Projet visant à créer des API communs pour du développement JavaScript hors navigateur (console, GUI...)

Exemple : standardiser l'accès aux fichiers

Le projet propose une norme pour le chargement de modules utilisé entre autre par Node.js

<http://wiki.commonjs.org/wiki/Modules/1.1.1>

▸ Création d'un module

```
// calcullette.js
exports.ajouter = function(nb1, nb2) {
  return Number(nb1) + Number(nb2);
};
```

▸ Utilisation

```
// main.js
var calc = require('./calcullette');

console.log(calc.ajouter(2, 3)); // 5
```



▸ Export

```
// (function (exports, require, module, __filename, __dirname) {  
'use strict';  
  
var getRandom = function() {  
    return Math.random();  
};  
  
exports.getRandomIntInclusive = function(min, max) {  
    min = Math.ceil(min);  
    max = Math.floor(max);  
    return Math.floor(getRandom() * (max - min + 1)) + min;  
};  
// });
```

- Le code s'exécute dans une fonction, pas besoin de la créer (ici en commentaire)
- Cette fonction contient un paramètre exports, qui est un objet que l'on pourra récupérer à l'import (il n'y a plus qu'à l'étendre)



▸ Import

```
// (function (exports, require, module, __filename, __dirname) {  
'use strict';  
  
var moment = require('moment');  
var getRandomIntInclusive = require('./random').getRandomIntInclusive;  
  
var Clock = function(parentElt) {  
  this.parentElt = parentElt;  
};  
  
// ...  
// });
```

- A l'import on utilise la fonction `require()`
 - Si le fichier est local on commencera toujours par `./` ou `../`
 - Sinon on fait référence à une installation via npm ou bien un fichier se trouvant dans le binaire de Node (`fs`, `http`, `readline...`)
 - La fonction `require` est synchrone et peut s'utiliser dans un `if` ou une boucle



▸ Export (autre chose qu'un objet)

```
// (function (exports, require, module, __filename, __dirname) {  
'use strict';  
  
var Clock = function(parentElt) {  
    this.parentElt = parentElt;  
};  
  
// ...  
  
module.exports = Clock;  
  
// });
```

- Lorsque l'on exporte autre chose qu'un objet, on peut écraser exports en écrivant module.exports (ici on exporte une fonction constructeur)
- Une bonne pratique pourrait être d'importer en début de fichier et d'exporter toujours en fin de fichier (en pratique c'est rarement le cas)



- ▶ Utilisation dans le Navigateur
 - Il faut passer par une bibliothèque externe :
 - Soit un bundler comme browserify (historique) ou webpack (moderne)
 - Soit un loader comme SystemJS
 - Les fichiers décrivant eux-même leurs dépendances, il suffit d'indiquer un point d'entrée dans l'application (ou plusieurs)
- ▶ Utilisation dans Node.js
 - Node.js supporte par défaut les modules CommonJS



- ▶ **Asynchronous Module Definition**

CommonJS ne permettant de charger des modules côté client sans transformation préalable. Des développeurs ont imaginé la syntaxe AMD.

- ▶ **Fonctionnement**

L'utilisation de modules AMD se fait via 2 fonctions globales : `require()` et `define()`. `define()` permet de définir un module et ses dépendances, `require` définit un point d'entrée dans l'application.



▸ Export

```
// random.js
define(function() {
  'use strict';

  return {
    getRandom: function () {
      return Math.random();
    },
    // ...
    getRandomIntInclusive: function (min, max) {
      min = Math.ceil(min);
      max = Math.floor(max);
      return Math.floor(this.getRandom() * (max - min + 1)) + min;
    }
  };
});
```

▸ define

- La définition d'un module se passe dans le callback de la fonction define
- Pour exporter on utilise la valeur de retour
- Si plusieurs valeurs à exporter il suffit de retourner un tableau ou un objet



► Import

```
// clock.js
define(['moment', './random'], function(moment, random) {
  'use strict';

  var getRandomIntInclusive = random.getRandomIntInclusive;

  var Clock = function(parentElt) {
    this.parentElt = parentElt;
  };

  // ...

  return Clock;
});
```

- Pour importer on passera un tableau en premier paramètre du define
- Le callback sera appelé avec les retours des modules importés, dans le même ordre



▸ Point d'entrée

```
// main.js
require(['./clock'], function(Clock) {
    'use strict';

    var clockElt = document.querySelector('.clock');
    var clock = new Clock(clockElt);
    clock.start();
});
```

▸ Require

- La fonction require permet de définir un point d'entrée dans l'application
- C'est elle qui va inclure les balises scripts sur la page
- Idéalement elle ne devrait être utilisée qu'un seul fois par page
- Les scripts chargent de manière asynchrone et s'exécute lorsqu'il n'ont pas de dépendances ou que toutes leurs dépendances ont été résolues



▸ Utilisation dans le Navigateur

- Il faut passer par une bibliothèque externe :
 - Soit un loader comme curl, Require.js ou plus récemment SystemJS
 - Soit un bundler comme webpack qui va faire une transformation

▸ Utilisation dans Node.js

- Non pertinent



- ▶ Universal Module Definition (UMD)
 - ▶ Un module universel, qui est à la fois AMD, CommonJS et parfois IIFE
<https://github.com/umdjs/umd>
 - ▶ Peut s'utiliser avec tous les loaders et les bundlers existants (et sans rien si IIFE)
 - ▶ Pertinent pour les projets où l'on ne peut pas prédire la techno qui sera utilisée pour charger le module
 - ▶ Exemple de module UMD :
 - ▶ jQuery
 - ▶ Lodash → « The Lodash library exported as a UMD module. »
<https://github.com/lodash/lodash>
 - ▶ Angular → @angular/core/bundles/core.umd.js



▸ Exemple de module UMD

```
(function (root, factory) {  
  if (typeof exports === 'object') {  
    // CommonJS  
    module.exports = factory(require('moment'), require('./random'));  
  } else if (typeof define === 'function' && define.amd) {  
    // AMD  
    define(['moment', './random'], function (moment, random) {  
      return factory(moment, random);  
    });  
  } else {  
    // IIFE (global var)  
    root.Clock = factory(root.moment, root.random);  
  }  
})(this, function (moment, random) {  
  'use strict';  
  
  var getRandomIntInclusive = random.getRandomIntInclusive;  
  
  var Clock = function(parentElt) {  
    this.parentElt = parentElt;  
  };  
  
  // ...  
  
  return Clock;  
}));
```



- Utilisation dans le Navigateur :
 - Sans rien si UMD inclus IIFE (pas toujours le cas)
 - Avec un bundler : browserify ou webpack
 - Avec un loader : curl, Require.js, SystemJS
- Utilisation dans Node.js
 - Avec la fonction require

Modules JavaScript - ECMAScript Modules



- ES6 introduit la notion de module au niveau du langage (ECMAScript Modules ou ESM)
- Le système est statique, les imports doivent être présent en début de fichier
- Ne permet donc pas de charger dynamiquement le contenu d'un tableau ou d'effectuer un import conditionnel comme avec CommonJS

Modules JavaScript - ECMAScript Modules



- Exporter

- Les modules ES permettent d'exporter des valeurs multiples (plutôt que de les regrouper dans un objet)

```
export function getRandom() {  
    return Math.random();  
}  
  
export function getRandomArbitrary(min, max) {  
    return Math.random() * (max - min) + min;  
}  
  
export function getRandomInt(min, max) {  
    min = Math.ceil(min);  
    max = Math.floor(max);  
    return Math.floor(Math.random() * (max - min)) + min;  
}  
  
export function getRandomIntInclusive(min, max) {  
    min = Math.ceil(min);  
    max = Math.floor(max);  
    return Math.floor(Math.random() * (max - min + 1)) + min;  
}
```

Modules JavaScript - ECMAScript Modules



- On peut également exporter en fin de fichier pour mieux lire les dépendances

```
function getRandom() {  
  return Math.random();  
}  
  
function getRandomArbitrary(min, max) {  
  return Math.random() * (max - min) + min;  
}  
  
function getRandomInt(min, max) {  
  min = Math.ceil(min);  
  max = Math.floor(max);  
  return Math.floor(Math.random() * (max - min)) + min;  
}  
  
function getRandomIntInclusive(min, max) {  
  min = Math.ceil(min);  
  max = Math.floor(max);  
  return Math.floor(Math.random() * (max - min + 1)) + min;  
}  
  
export {  
  getRandom,  
  getRandomArbitrary,  
  getRandomInt,  
  getRandomIntInclusive,  
}
```


Modules JavaScript - ECMAScript Modules



- Export par défaut
 - On peut également exporter une valeur par défaut (l'import est particulier)
 - Eviter d'exporter des fonctions multiples dans un objet par défaut

```
export default class Clock {  
  constructor(parentElt) {  
    this.parentElt = parentElt;  
  }  
  
  update() {  
    const r = getRandomIntInclusive(0, 255);  
    const g = getRandomIntInclusive(0, 255);  
    const b = getRandomIntInclusive(0, 255);  
    const now = new Date();  
  
    document.body.style.backgroundColor = `rgb(${r}, ${g}, ${b})`;  
    document.body.style.color = `rgb(${255 - r}, ${255 - g}, ${255 - b})`;  
    this.parentElt.innerHTML = now.toLocaleTimeString();  
  }  
  
  start() {  
    this.update();  
    setInterval(this.update.bind(this), 1000);  
  }  
}
```

Modules JavaScript - ECMAScript Modules



- On peut renommer les imports (en cas de conflit par exemple)

```
import { format } from 'date-fns/esm';
import { getRandomIntInclusive as rand } from './random';

export default class Clock {
  constructor(parentElt) {
    this.parentElt = parentElt;
  }

  update() {
    const r = rand(0, 255);
    const g = rand(0, 255);
    const b = rand(0, 255);
    const now = new Date();

    document.body.style.backgroundColor = `rgb(${r}, ${g}, ${b})`;
    document.body.style.color = `rgb(${255 - r}, ${255 - g}, ${255 - b})`;
    this.parentElt.innerHTML = now.toLocaleTimeString();
  }

  start() {
    this.update();
    setInterval(this.update.bind(this), 1000);
  }
}
```

Modules JavaScript - ECMAScript Modules



- Pour les imports par défaut il faut retirer les accolades
- L'import peut être renommé directement

```
import Horloge from './clock';  
  
let clockElt = document.querySelector('.clock');  
let clock = new Horloge(clockElt);  
clock.start();
```



▸ Utilisation dans le Navigateur :

- Sans rien dans une balise `<script type="module">`
- Avec un bundler : webpack ou Rollup
- Avec un loader : SystemJS

▸ Utilisation dans Node.js

- Avec le flag `--experimental-modules`
- Les fichiers doivent avoir l'extension `.jsm`

Modules JavaScript - Dynamic Imports



- ESNext prévoit un import dynamique (fonctions asynchrones retournants une promesse)
- webpack est déjà compatible (le code est mis dans un bundle différent, chargé au moment de l'import)

```
document.addEventListener('click', () => {  
  import('./calc').then(module) => {  
    console.log(module.add(1, 2));  
  });  
});
```

Modules JavaScript - Dynamic Imports



- Utilisation dans le Navigateur :
 - Avec webpack
- Utilisation dans Node.js
 - Avec webpack



Loaders JavaScript



- **Require.js**

Bibliothèque permettant le chargement de modules AMD.

- **Plugins**

Quelques plugins existent pour charger par exemple des fichiers CSS, des fichiers de traduction, etc...

- **r.js**

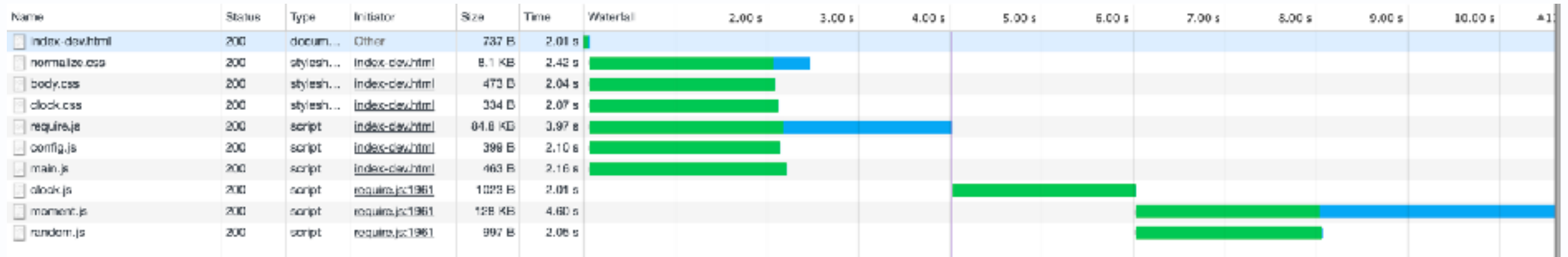
Il est possible d'utiliser un builder pour créer un seul fichier de production, le chargement se ferait sinon en « escalier ».



Loaders JavaScript - Require.js



► Chargement en escalier





- ▶ **SystemJS**

SystemJS est un loader universel qui sait charger des modules CommonJS, AMD, ES6 et IIFE dans les navigateurs et sous node.js

<https://github.com/systemjs/systemjs>

- ▶ **jspm**

Afin de faciliter le chargement de modules installés via des gestionnaires de dépendances, jspm permet le chargement de packages npm ou bien de

- ▶ **SystemJS Builder**

Afin de faciliter le chargement de modules installés via des gestionnaires de dépendance



Bundlers JavaScript



- ▶ Browserify

Permet de charger des modules CommonJS côté client.

- ▶ Installation :

`npm install -g browserify`

- ▶ Transormation en code client :

`browserify main.js > calculette-browser.js`

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title></title>
</head>
<body>
  <script src="calculette-browser.js"></script>
</body>
</html>
```



- **webpack**

webpack est un bundler universel, il sait charger n'importe quel type de modules, AMD, CommonJS, ES6

- Des loaders supplémentaires permettent de charger des fichiers CSS / LESS / SCSS / i18n / ...
- Depuis sa version 2, webpack supporte nativement les modules ES6, alors qu'il fallait utiliser un transpileur comme Babel dans la version 1.
- Le projet open-source le plus financé avec Vue.js (+ 300k\$ par an), son développeur contribue à webpack à temps plein.

Bundlers JavaScript - webpack



```
const HtmlWebpackPlugin = require('html-webpack-plugin');
const CleanWebpackPlugin = require('clean-webpack-plugin');

module.exports = {
  entry: './src/js/index.js',
  output: {
    path: __dirname + '/dist',
    filename: 'bundle.[hash].js'
  },
  module: {
    rules: [
      {
        test: /\.js$/,
        exclude: /(node_modules|bower_components)/,
        use: {
          loader: 'babel-loader',
          options: {
            presets: [["env", {
              "targets": {
                "browsers": ["> 1% in FR"]
              }
            }]]
          }
        }
      }
    ]
  },
  plugins: [
    new CleanWebpackPlugin(['dist']),
    new HtmlWebpackPlugin({
      template: './src/index.html'
    })
  ]
};
```



Préprocesseurs CSS



- Les préprocesseurs CSS sont des technologies qui permettent à des langages proches de CSS de transpirer en CSS en y ajoutant des fonctionnalités
- Comparateurs de préprocesseurs CSS
<http://csspre.com/compare/>



- Apparu en 2009, inspiré par SASS
<http://lesscss.org/>



- Variables

```
@link-color:      #428bca; // sea blue

a, .link {
  color: @link-color;
}

.widget {
  color: #fff;
  background: @link-color;
}
```

```
nav ul {
  &:extend(.inline);
  background: blue;
}
```



► Mixins (fonctions)

```
.border-radius(@radius) {  
  -webkit-border-radius: @radius;  
  -moz-border-radius: @radius;  
  border-radius: @radius;  
}  
  
#header {  
  .border-radius(4px);  
}  
  
.button {  
  .border-radius(6px);  
}
```

```
.foo {  
  background: #900;  
}  
@import "this-is-valid.less";
```



► Imbriquer

```
#header {  
  color: black;  
  .navigation {  
    font-size: 12px;  
  }  
  .logo {  
    width: 300px;  
  }  
}
```



► Transpiler

- Côté client :

```
<link rel="stylesheet/less" type="text/css" href="styles.less" />  
<script src="less.js" type="text/JavaScript"></script>
```

- En ligne de commande

- Installer

Transpiler

```
npm install -g less
```

```
lessc styles.less > styles.css
```

Préprocesseurs CSS - SASS



- Apparu en 2007
<http://sass-lang.com>
- A peu de choses près fonctionnalités égales à Less
- 2 syntaxes :
 - SASS

```
$primary-color: #333  
  
body  
  color: $primary-color
```

- SCSS (inspirée de Less)

```
$primary-color: #333;  
  
body {  
  color: $primary-color;  
}
```



- Apparu en 2010, inspiré par SASS et LESS
<http://learnboost.github.io/stylus/>
- Syntaxe encore plus concise

```
body
  font 12px Helvetica, Arial, sans-serif


a.button
  -webkit-border-radius 5px
  -moz-border-radius 5px
  border-radius 5px
```





- ▶ Less et Sass ont inspiré le W3C

Des normes sur les variables, opérateurs existe désormais nativement mais sont mal supportées

calc() as CSS unit value  - CR

Method of allowing calculated values for length units, i.e. width:
calc(100% - 3em)

Global: 76.09% + 5.16% = 81.25%
unprefixed: 75.55% + 5.16% = 80.71%

Current aligned Usage relative Show all

IE / Edge	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8							4.1	
² 9		31					4.3	
10		42					¹ 4.4	
11	38	43	7.1		7.1		¹ 4.4.4	
Edge	39	44	8	30	8.4	8	40	42
	40	45	9	31	9			
	41	46		32				
	42	47						



- Apparue fin 2013

<http://www.myth.io>

- Variables

```
:root {  
  --purple: #847AD1;  
  --large: 10px;  
}  
  
a {  
  color: var(--purple);  
}  
  
pre {  
  padding: var(--large);  
}
```

- Opérateurs

```
pre {  
  margin: calc(var(--large) * 2);  
}
```




- Apparue mi-2014

<http://cssnext.io>

- Supporte plus de nouveautés CSS que Myth

automatic vendor prefixes, custom properties & `var()`, reduced `calc()`, custom media queries, media queries ranges, custom selectors, `color()`, `hwb()`, `gray()`, `#rrggbbaa`, `rebeccapurple`, font-variant, filter, `rem` units, `:any-link` pseudo-class, `:matches` pseudo-class, `:not` pseudo-class, pseudo-elements, Alpha colors, Bonus features, `@import`, minification, @todo

cssnext {||||}



Supersets JavaScript

Supersets JavaScript - Introduction



- A l'instar des préprocesseur CSS, les Supersets JavaScript sont des surcouches qui transpilent en JavaScript
- Ils ajoutent des concepts inspirés d'autres langages (types, structures, sucre syntaxique...)
- La prochaine norme du langage JavaScript, ECMAScript 2015 (anciennement ECMAScript 6) peut également être utilisé comme un Superset, les navigateurs actuels (juillet 2015) ne le supportant pas encore.

Supersets JavaScript - CoffeeScript



- Apparu en 2009
<http://coffeescript.org/>
- Inspiré de Ruby (populaire chez ces développeurs, Ruby on Rails l'intègre depuis sa version 3.1)
- 42208 projets sur GitHub (juillet 2015)
- Offre principalement du sucre syntaxique



Supersets JavaScript - CoffeeScript



- ▶ Installation

`npm install -g coffeescript`

- ▶ Compilation

`coffee --compile --output dist/ src/`

Supersets JavaScript - CoffeeScript



▸ Variables et conditions

```
# CoffeeScript
aleatoire = 100 * Math.random()

console.log aleatoire # 46.67751151137054
console.log "Entre 25 et 50 " if 25 < aleatoire < 50 # Entre 25 et 50
```

```
// Generated by CoffeeScript 1.9.3
(function() {
  var aleatoire;

  aleatoire = 100 * Math.random();

  console.log(aleatoire);

  if ((25 < aleatoire && aleatoire < 50)) {
    console.log("Entre 25 et 50 ");
  }

}).call(this);
```

Supersets JavaScript - CoffeeScript



▸ Arrow functions

```
# CoffeeScript  
square = (x) -> x * x  
console.log square 3 # 9
```

```
// Generated by CoffeeScript 1.9.3  
(function() {  
  var square;  
  
  square = function(x) {  
    return x * x;  
  };  
  
  console.log(square(3));  
}).call(this);
```

Supersets JavaScript - CoffeeScript



► Classes

```
# CoffeeScript
class Personne
  constructor: (@prenom) ->
  hello: -> "Je m'appelle #{@prenom}";

romain = new Personne "Romain"
console.log romain.hello() # Je m'appelle Romain
```

```
// Generated by CoffeeScript 1.9.3
(function() {
  var Personne, romain;

  Personne = (function() {
    function Personne(prenom) {
      this.prenom = prenom;
    }

    Personne.prototype.hello = function() {
      return "Je m'appelle " + this.prenom;
    };

    return Personne;
  })();

  romain = new Personne("Romain");

  console.log(romain.hello());
}).call(this);
```


Supersets JavaScript - CoffeeScript



► Heritage

```
# CoffeeScript
class Personne
  constructor: (@prenom) ->
  hello: -> "Je m'appelle #{@prenom}"

class Formateur extends Personne
  constructor: (prenom, @specialite) -> super(prenom)
  hello: -> super() + ", ma spécialité est #{@specialite}"

romain = new Formateur "Romain", "CoffeeScript"
console.log romain.hello() # Je m'appelle Romain, ma spécialité est CoffeeScript
```

Supersets JavaScript - CoffeeScript



```
// Generated by CoffeeScript 1.9.3
(function() {
  var Formateur, Personne, romain,
    extend = function(child, parent) { for (var key in parent) { if (hasProp.call(parent, key)) child[key] =
parent[key]; } function ctor() { this.constructor = child; } ctor.prototype = parent.prototype; child.prototype = new
ctor(); child.__super__ = parent.prototype; return child; },
    hasProp = {}.hasOwnProperty;

  Personne = (function() {
    function Personne(prenom1) {
      this.prenom = prenom1;
    }

    Personne.prototype.hello = function() {
      return "Je m'appelle " + this.prenom;
    };

    return Personne;
  })();

  Formateur = (function(superClass) {
    extend(Formateur, superClass);

    function Formateur(prenom, specialite) {
      this.specialite = specialite;
      Formateur.__super__.constructor.call(this, prenom);
    }

    Formateur.prototype.hello = function() {
      return Formateur.__super__.hello.call(this) + (" , ma spécialité est " + this.specialite);
    };

    return Formateur;
  })(Personne);

  romain = new Formateur("Romain", "CoffeeScript");

  console.log(romain.hello());
}).call(this);
```



- ▶ **ECMAScript 2015 / ECMAScript 6**

La nouvelle version de JavaScript prévoit une syntaxe pour l'utilisation de module. A l'heure actuelle (juillet 2015), ni les navigateurs ni Node.js ou io.js ne supportent cette syntaxe.

- ▶ **Babel / Traceur**

Babel et Traceur sont des bibliothèques qui permettent de transpiler du code ES6 en ES5 et ainsi l'utiliser sur les moteurs actuels.

- ▶ **Installation :**

`npm install -g babel`

- ▶ **Utilisation (toutes les sources du répertoires src vers le répertoire dist) :**

`babel src --out-dir dist/`





► Nouveautés

<https://babeljs.io/docs/learn-es2015/>

<http://es6-features.org/>

- Arrows
- Classes
- Template Strings
- Default + Rest + Spread
- Let + Const
- Modules
- Module Loaders
- Map + Set + WeakMap + WeakSet
- ...

► Compatibilité

<https://kangax.github.io/compat-table/es6/>

Supersets JavaScript - Dart



- Créé par Google en 2013
- Permet de créer de programmes serveurs (VM), mobile (VM) et transpile en JavaScript pour les navigateurs
- Syntaxe identique à C/C++
- 3868 projets sur GitHub (juillet 2015)

```
void main()  
{  
  print("Hello !");  
}
```



Supersets JavaScript - Haxe



- Apparu en 2005
- Projet français permettant de transpiler dans différents language de programmation dont JavaScript
- 3178 projets sur GitHub (juillet 2015)

```
class HelloWorld {  
    static public function main() {  
        trace("Hello World");  
    }  
}
```





TypeScript

TypeScript - Introduction



- TypeScript : JavaScript + Typage statique
 - TypeScript est un langage créé par Microsoft, construit comme un sur-ensemble d'ECMAScript
 - Pour pouvoir exécuter le code il faut le transformer en JavaScript avec un compilateur
 - A quelques exceptions près et selon la configuration, le JavaScript est valide en TypeScript
 - Le principal intérêt de TypeScript est l'ajout d'un typage statique

TypeScript - Installation



- Installation
 - `npm install -g typescript`
- Création d'un fichier de configuration
 - `tsc --init`
- Compilation
 - `tsc`

TypeScript - Typage statique



- Le principal intérêt de TypeScript est l'introduction d'un typage statique

```
const lastName: string = 'Bohdanowicz';  
const age: number = 32;  
const isTrainer: boolean = true;
```

- Types basiques :

- *boolean*
- *number*
- *string*

TypeScript - Typage statique



▸ Avantages

- Complétion

```
const firstName: string = 'Romain';

function hello(firstName: string): string {
  return `Hello ${firstName.}`;
}
```

- `charAt(pos: number)`
- `charCodeAt(index: number)`
- `concat(... strings: string)`
- `indexOf(searchString: string,`

- Détection des erreurs

```
const firstName: string = 'Romain';

function hello(firstName: string): string {
  return `Hello ${firstName}`;
}

hello({
  firstName: 'Romain',
});
```

TypeScript - Typage statique



▸ Tableaux

```
const firstNames: string[] = ['Romain', 'Edouard'];  
const colors: Array<string> = ['blue', 'white', 'red'];
```

▸ Tuples

```
const email: [string, boolean] = ['romain.bohdanowicz@gmail.com', true];
```

▸ Enum

```
enum Choice {Yes, No, Maybe}  
  
const c1: Choice = Choice.Yes;  
const choiceName: string = Choice[1];
```

▸ Never

```
function error(message: string): never {  
    throw new Error(message);  
}
```

TypeScript - Typage statique



▸ Any

```
let anyType: any = 12;  
anyType = "now a string string";  
anyType = false;  
anyType = {  
  firstName: 'Romain'  
};
```

▸ Void

```
function withoutReturn(): void {  
  console.log('Do something')  
}
```

▸ Null et undefined

```
let u: undefined = undefined;  
let n: null = null;
```

TypeScript - Assertion de type



- Le compilateur ne peut pas toujours déterminer le type adéquat :

```
const formElt = document.querySelector('form.myForm');  
const url = formElt.action; // error TS2339: Property 'action' does not exist on  
type 'Element'.
```

- Il faut alors lui préciser, 3 syntaxes possibles

```
let formElt = <HTMLFormElement> document.querySelector('form.myForm');  
const url = formElt.action;
```

```
let formElt = document.querySelector<HTMLFormElement>('form.myForm');  
const url = formElt.action;
```

```
let formElt = document.querySelector('form.myForm') as HTMLFormElement;  
const url = formElt.action;
```

TypeScript - Inférence de type



- TypeScript peut parfois déterminer automatiquement le type :

```
const title = 'First Names';  
console.log(title.toUpperCase());  
  
const names = ['Romain', 'Edouard'];  
for (let n of names) {  
    console.log(n.toUpperCase());  
}
```

TypeScript - Interfaces



- Pour documenter un objet on utilise une interface
 - Anonyme

```
function helloInterface(contact: {firstName: string}) {  
    console.log(`Hello ${contact.firstName.toUpperCase()}`);  
}
```

- Nommée

```
interface ContactInterface {  
    firstName: string;  
}  
  
function helloNamedInterface(contact: ContactInterface) {  
    console.log(`Hello ${contact.firstName.toUpperCase()}`);  
}
```


TypeScript - Interfaces



- Les propriétés peuvent être :
 - optionnelles (ici *lastName*)
 - en lecture seule, après l'initialisation (ici *age*)
 - non déclarées (avec les crochets)

```
interface ContactInterface {  
    firstName: string;  
    lastName?: string;  
    readonly age: number;  
    [propName: string]: any;  
}  
  
function helloNamedInterface(contact: ContactInterface) {  
    console.log(`Hello ${contact.firstName.toUpperCase()}`);  
}
```



- Quelques différences avec JavaScript sur le mot clé class
 - On doit déclarer les propriétés
 - On peut définir une visibilité pour chaque membre : *public*, *private*, *protected*

```
class Contact {  
  private firstName: string;  
  
  constructor(firstName: string) {  
    this.firstName = firstName;  
  }  
  
  hello(): string {  
    return `Hello my name is ${this.firstName}`;  
  }  
}  
  
const romain = new Contact('Romain');  
console.log(romain.hello()); // Hello my name is Romain
```



- Une classe peut
 - Hériter d'une autre classe (comme en JS)
 - Implémenter une interface
 - Être utilisée comme type

```
interface Writable {  
  write(data: string): void;  
}  
  
class FileLogger implements Writable {  
  write(data: string): Writable {  
    console.log(`Write ${data}`);  
    return this;  
  }  
}
```

TypeScript - Génériques



- Permet de paramétrer le type de certaines méthodes

```
class Stack<T> {  
  private data: Array<T> = [];  
  push(val: T) {  
    this.data.push(val);  
  }  
  pop(): T {  
    return this.data.pop();  
  }  
  peek(): T {  
    return this.data[this.data.length - 1];  
  }  
}  
  
const strStack = new Stack<string>();  
strStack.push('html');  
strStack.push('body');  
strStack.push('h1');  
console.log(strStack.peek().toUpperCase()); // H1  
console.log(strStack.pop().toUpperCase()); // H1  
console.log(strStack.peek().toUpperCase()); // BODY
```

TypeScript - Décorateurs



- Permettent l'ajout de fonctionnalités aux classes ou membre d'une classe en annotant plutôt que via du code à l'utilisation
- Norme à l'étude en JavaScript par le TC39
<https://github.com/tc39/proposal-decorators>
- Supporté de manière expérimentale en TypeScript
- Pour activer leur support il faut éditer le tsconfig.json ou passer une option au compilateur

```
{  
  "compilerOptions": {  
    "target": "es5",  
    "experimentalDecorators": true  
  }  
}
```

TypeScript - Décorateurs



▸ Décorateur de classes

```
'use strict';

function Freeze(obj) {
  Object.freeze(obj);
}

@Freeze
class MyMaths {
  static sum(a, b) {
    return Number(a) + Number(b);
  }
}

try {
  MyMaths['subtract'] = function(a, b) {
    return a - b;
  };
}
catch(err) {
  // Cannot add property subtract, object is not extensible
  console.log(err.message);
}
```



▸ Décorateur de propriétés

```
import 'reflect-metadata';

const minLengthMetadataKey = Symbol("minLength");

function MinLength(length: number) {
  return Reflect.metadata(minLengthMetadataKey, length);
}

function validateMinLength(target: any, propertyKey: string): boolean {
  const length = Reflect.getMetadata(minLengthMetadataKey, target, propertyKey);
  return target[propertyKey].length >= length;
}

class Contact {
  @MinLength(7)
  protected firstName;

  constructor(firstName: string) {
    this.firstName = firstName;
  }

  isValid(): boolean {
    return validateMinLength(this, 'firstName');
  }
}

const romain = new Contact('Romain');
console.log(romain.isValid()); // false
```



Node.js



- ▶ Créé 2009 par Ryan Dahl
 - A l'origine, Ryan Dahl voulait simplifier la création d'une barre d'upload.
- ▶ Sponsorisé par la société Joyent.
- ▶ Un programme en ligne de commande combinant :
 - le moteur JavaScript V8 de Chrome
 - une boucle d'événement
 - une gestion bas niveau des entrées/sorties
- ▶ Un système en production :
 - Chez des startups à la pointe : Airbnb, ...
 - Dans des grands groupes : Microsoft, PayPal, Walmart, LinkedIn



- ▶ Windows

Exécutables : <https://nodejs.org/download/>

- ▶ OS X

Exécutables : <https://nodejs.org/download/>

Ou via homebrew : `brew install node`

- ▶ Debian / Ubuntu

`sudo apt-get update`

`sudo apt-get install nodejs npm`

- ▶ Pensez à ajouter le répertoire de Node au Path.

Node.js - Hello, word !



- ▶ Lancement du programme
node FILE_PATH.js
- ▶ Interruption
CTRL-C

A screenshot of a macOS terminal window titled "LearningJS - node - 78x16". The prompt is "MacBook-Pro-de-Romain:LearningJS romain\$". The command entered is "node Node.js/Slides/helloworld.js". The output shows the word "Helloworld" printed ten times, one per line. A cursor is visible at the end of the tenth line.

```
MacBook-Pro-de-Romain:LearningJS romain$ node Node.js/Slides/helloworld.js
Helloworld
Helloworld
Helloworld
Helloworld
Helloworld
Helloworld
Helloworld
Helloworld
Helloworld
Helloworld
```

```
/* Un simple helloworld */

/** @function helloworld */
function helloworld() {
  'use strict'; // bonne pratique
  console.log('Helloworld');
}

setInterval(helloworld, 1000);
```



- Parcourir un répertoire
 - Asynchrone

```
var fs = require('fs');  
  
fs.readdir('.', function (err, files) {  
  if (err) throw err;  
  console.log(files);  
});
```

- Synchrone

```
var fs = require('fs');  
  
console.log(fs.readdirSync('.'));
```



- Lire un fichier
 - Asynchrone

```
var fs = require('fs');  
  
fs.readFile('lorem.txt', {encoding: 'UTF-8'}, function (err, content) {  
  if (err) throw err;  
  console.log(content);  
});
```

- Synchrone

```
var fs = require('fs');  
  
var content = fs.readFileSync('lorem.txt', {encoding: 'UTF-8'});  
console.log(content);
```



- Ecrire dans un fichier
 - Asynchrone

```
var fs = require('fs');
var moment = require('moment');

var date = moment().format('DD/MM/YYYY à HH:mm:ss');
var message = 'Ligne loguée le ' + date;

fs.writeFile('log.txt', message + '\n', {flag: 'a'}, function (err) {
  if (err) throw err;
  console.log('Log enregistré ! ('+message+')');
});
```

- Synchrones

```
var fs = require('fs');
var moment = require('moment');

var date = moment().format('DD/MM/YYYY à HH:mm:ss');
var message = 'Ligne loguée le ' + date;

fs.writeFileSync('log.txt', message + '\n', {flag: 'a'});
```



formation.tech



GRUNT

Grunt



▸ Grunt JS

Permet l'automatisation de tâches de développement front-end.

▸ Exemples

- minifier ses fichiers JS
- compiler ses CSS
- compresser les images
- exécuter les tests
- vérifier les conventions de codage

Grunt - Installation



- Installation via npm :
npm install -g grunt-cli

Gruntfile.js

```
/*global module:false*/
module.exports = function(grunt) {

  grunt.initConfig({
    copy: {
      dist: {
        src: 'index.html',
        dest: 'dist/index.html'
      }
    },
    uglify: {
      dist: {
        src: 'script.js',
        dest: 'dist/script.js'
      }
    }
  });

  grunt.loadNpmTasks('grunt-contrib-copy');
  grunt.loadNpmTasks('grunt-contrib-uglify');

  // Default task.
  grunt.registerTask('default', ['copy',
  'uglify']);
};
```

package.json

```
{
  "engines": {
    "node": ">= 0.10.0"
  },
  "devDependencies": {
    "grunt": "^0.4.5",
    "grunt-contrib-copy": "^0.8.0",
    "grunt-contrib-uglify": "^0.9.1"
  }
}
```

Grunt - Hello, world !



src/index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title></title>
</head>
<body>
  <div>
    Prénom : <input type="text" id="prenom">
  </div>
  <p>
    Bonjour <span id="output"></span>
  </p>
  <script src="script.js"></script>
</body>
</html>
```

src/script.js

```
!function() {
  'use strict';

  var inputElt =
document.querySelector('#prenom');
  var outputElt =
document.querySelector('#output');

  inputElt.addEventListener('input', function()
{
  outputElt.innerHTML = inputElt.value;
});
})();
```

dist/index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title></title>
</head>
<body>
  <div>
    Prénom : <input type="text" id="prenom">
  </div>
  <p>
    Bonjour <span id="output"></span>
  </p>
  <script src="script.js"></script>
</body>
</html>
```

dist/script.js

```
!function(){"use strict";var
a=document.querySelector("#prenom"),b=document.qu
erySelector("#output");a.addEventListener("input"
,function(){b.innerHTML=a.value}})();
```

Grunt - Hello, world !



Gruntfile.js

```
/*global module:false*/
module.exports = function(grunt) {

  grunt.initConfig({
    copy: {
      dist: {
        src: 'index.html',
        dest: 'dist/index.html'
      }
    },
    uglify: {
      dist: {
        src: 'script.js',
        dest: 'dist/script.js'
      }
    }
  });

  grunt.loadNpmTasks('grunt-contrib-copy');
  grunt.loadNpmTasks('grunt-contrib-uglify');

  // Default task.
  grunt.registerTask('default', ['copy', 'uglify']);
};
```

Package.json

```
{
  "devDependencies": {
    "grunt": "^0.4.5",
    "grunt-contrib-copy": "^0.8.0",
    "grunt-contrib-uglify": "^0.9.1"
  }
}
```

- Package.json créé avec :
npm init
npm install grunt --save-dev
npm install grunt-contrib-copy --save-dev
npm install grunt-contrib-uglify --save-dev



- Liste des plugins pour grunt :
<http://gruntjs.com/plugins>
(4,403 plugins en juillet 2015)
- Les plugins contrib-* sont ceux des développeurs de grunt.



▸ jit-grunt :

Installation : `npm install jit-grunt --save-dev`

Simplifie le chargement de plugins

Avant

```
/*global module:false*/  
module.exports = function(grunt) {  
  
  grunt.loadNpmTasks('grunt-contrib-clean');  
  grunt.loadNpmTasks('grunt-contrib-concat');  
  grunt.loadNpmTasks('grunt-contrib-copy');  
  grunt.loadNpmTasks('grunt-contrib-cssmin');  
  grunt.loadNpmTasks('grunt-contrib-jshint');  
  grunt.loadNpmTasks('grunt-contrib-less');  
  grunt.loadNpmTasks('grunt-contrib-uglify');  
  grunt.loadNpmTasks('grunt-contrib-watch');  
  grunt.loadNpmTasks('grunt-google-cdn');  
  grunt.loadNpmTasks('grunt-rev');  
  grunt.loadNpmTasks('grunt-spritesmith');  
  grunt.loadNpmTasks('grunt-usemin');  
  
  grunt.initConfig({  
    // ...  
  });  
  
  // Default task.  
  grunt.registerTask('default', [  
    // ...  
  ]);  
};
```

Après

```
/*global module:false, require*/  
module.exports = function(grunt) {  
  'use strict';  
  
  require('jit-grunt')(grunt, {  
    useminPrepare: 'grunt-usemin',  
    cdnify: 'grunt-google-cdn',  
    sprite: 'grunt-spritesmith'  
  });  
  
  // Project configuration.  
  grunt.initConfig({  
    // ...  
  });  
  
  // Default task.  
  grunt.registerTask('default', [  
    // ...  
  ]);  
};
```



▸ grunt-contrib-less :

npm install grunt-contrib-less --save-dev

Compile des fichiers LESS en CSS

```
module.exports = function(grunt) {  
  
  // ...  
  
  grunt.initConfig({  
    less: {  
      dev: {  
        files: [{  
          expand: true,  
          cwd: 'less',  
          src: ['*.less'],  
          dest: 'css/',  
          ext: '.css'  
        }]  
      },  
    },  
  });  
  
  // Default task.  
  grunt.registerTask('default', [  
    // ...  
  ]);  
  
};
```



▸ grunt-autoprefixer :

npm install grunt-autoprefixer --save-dev

Rajoute automatiquement les préfixes -moz, -webkit, -o, -ms en fonction des versions minimales des navigateurs à supporter

```
module.exports = function(grunt) {  
  
    // ...  
  
    grunt.initConfig({  
        // ...  
        autoprefixer: {  
            options: {  
                browsers: ['last 2 versions', 'ie 8', 'ie 9']  
            },  
            dev: {  
                files: [{  
                    expand: true,  
                    cwd: 'css/',  
                    src: '{,*/}*.css',  
                    dest: 'css/'  
                }]  
            },  
        },  
    });  
  
    // Default task.  
    grunt.registerTask('default', [  
        // ...  
    ]);  
  
};
```



▸ grunt-contrib-watch :

npm install grunt-contrib-watch --save-dev

Surveille les modifications sur des fichiers, exécute des taches en cas de changement

```
module.exports = function(grunt) {  
  
    // ...  
  
    grunt.initConfig({  
        // ...  
        watch: {  
            less: {  
                files: ['less/**/*.less'],  
                tasks: ['less:dev', 'autoprefixer:dev']  
            }  
        },  
    });  
  
    // Default task.  
    grunt.registerTask('default', [  
        // ...  
    ]);  
};
```




- **grunt-contrib-concat :**
npm install grunt-contrib-concat --save-dev
Concatène plusieurs fichiers en un. Utile pour optimiser les temps de chargement CSS/JS
- **grunt-contrib-uglify :**
npm install grunt-contrib-uglify --save-dev
Compresse les fichiers JS
- **grunt-contrib-cssmin :**
npm install grunt-contrib-cssmin --save-dev
Compresse les fichiers CSS



- **grunt-contrib-copy :**
npm install grunt-contrib-copy --save-dev
Copie des fichiers
- **grunt-contrib-clean :**
npm install grunt-contrib-clean --save-dev
Supprime des fichiers



▸ grunt-usemin:

npm install grunt-usemin --save-dev

Génère une configuration pour concat, uglify, cssmin à partir d'un fichier HTML

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title></title>

  <!-- build:css css/app.css -->
  <link rel="stylesheet" href="css/body.css">
  <link rel="stylesheet" href="css/button.css">
  <!-- endbuild -->
</head>
<body>

  <!-- build:js js/app.js -->
  <script src="js/create-button.js"></script>
  <script src="js/button-listener.js"></script>
  <!-- endbuild -->
</body>
</html>
```

Gruntfile.js

```
/*global module, require*/
module.exports = function(grunt) {
  'use strict';

  // ...

  grunt.initConfig({
    // ...
    useminPrepare: {
      html: 'index.html'
    },

    usemin: {
      html: ['dist/{,*/}*.html'],
      css: ['dist/{,*/}*.css'],
      js: ['dist/{,*/}*.js'],
    },

  });

  // Default task.
  grunt.registerTask('default', [
    // ...
  ]);
};
```



config générée

```
{
  "concat": {
    "generated": {
      "files": [{
        "dest": ".tmp/concat/css/app.css",
        "src": ["css/body.css", "css/button.css"]
      }, {
        "dest": ".tmp/concat/js/app.js",
        "src": ["js/create-button.js", "js/button-
listener.js"]
      }]
    },
    "uglify": {
      "generated": {
        "files": [{
          "dest": "dist/js/app.js",
          "src": [".tmp/concat/js/app.js"]
        }]
      }
    },
    "cssmin": {
      "generated": {
        "files": [{
          "dest": "dist/css/app.css",
          "src": [".tmp/concat/css/app.css"]
        }]
      }
    }
  }
}
```

index.html généré

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title></title>

  <link rel="stylesheet" href="css/app.css">
</head>
<body>

<script src="js/app.js"></script>
</body>
</html>
```

app.css généré

```
body{background:beige}button{width:50px;height:50px}
```

app.js généré

```
!function(){"use strict";var
a=document.createElement("button");a.innerHTML=0,a.id="monBouton",document.body.appendChild(a)}(),!function()
{"use strict";var
a=document.querySelector("#monBouton");a.addEventListener("click",function(){this.innerHTML++}})();
```



- ▶ **contrib-connect :**
serveur web
- ▶ **karma :**
lancer des tests
- ▶ **concurrent :**
exécuter des tâches en parallèle
- ▶ **sass :**
compile des fichiers SASS en CSS
- ▶ **contrib-imagemin :**
compresser des images
- ▶ **contrib-htmlmin :**
minifier le HTML
- ▶ **newer :**
ne lancer les tâches que sur les nouveaux fichiers
- ▶ **rev :**
génère un nom de fichier avec hash pour le cache (avec usemin)
- ▶ **contrib-jshint, jscs :**
vérifie les conventions sur les fichiers JS
- ▶ **google-cdn :**
remplace les fichiers locaux par des CDN
- ▶ **spritesmith :**
génère des fichiers Sprite CSS



- ▶ Grunt Init

Assistant de création de projet grunt

- ▶ Installation

npm install -g grunt-init

- ▶ Création du projet

grunt-init gruntfile

- ▶ Créer son propre assistant

<https://github.com/gruntjs/grunt-init-gruntplugin>

```
Please answer the following:
[?] Is the DOM involved in ANY way? (Y/n) Y
[?] Will files be concatenated or minified? (Y/n) Y
[?] Will you have a package.json file? (Y/n) Y
[?] Do you need to make any changes to the above before continuing? (y/N) N

Writing Gruntfile.js...OK
Writing package.json...OK

Initialized from template "gruntfile".
```



► Gulp

Equivalent de grunt, repose sur les streams Node.js (utilise la RAM plutôt que les fichiers).

Devient très populaire, 1645 plugins contre 4403 pour grunt (juillet 2015)

► Broccoli

484 plugins

► Brunch

262 plugins

► Prepros / CodeKit

<https://prepros.io>

<https://incident57.com/codekit/>

gulpfile.js

```
var gulp = require('gulp');
var uglify = require('gulp-uglify');

gulp.task('scripts', function() {
  // Minify and copy all JavaScript (except vendor
  // scripts)
  gulp.src(['client/js/**/*.js', '!client/js/vendor/
  **'])
    .pipe(uglify())
    .pipe(gulp.dest('build/js'));

  // Copy vendor files
  gulp.src('client/js/vendor/**')
    .pipe(gulp.dest('build/js/vendor'));
});

// The default task (called when you run `gulp`)
gulp.task('default', function() {
  gulp.run('scripts');

  // Watch files and run tasks if they change
  gulp.watch('client/js/**', function(event) {
    gulp.run('scripts');
  });
});
```



Tests Automatisés



- Avec les tests automatisés, les scénarios de test sont codés et peuvent être rejoués régulièrement.
- 3 types de test :
 - Test unitaire
Permet de tester les briques d'une application (classes)
 - Test d'intégration
Teste que les briques fonctionnent correctement ensemble
 - Test fonctionnel
Vérifie l'application du point de vue du client

Tests Automatisés - Karma



- ▶ Lanceur de test

Permet de lancer vos tests simultanément dans Chrome, Firefox, Internet Explorer...

- ▶ Installation

`npm install -g karma-cli`

`npm install karma —save-dev`

- ▶ Configuration du projet

`karma init`

- ▶ Lancement des tests

`karma start`

```
Air-de-Romain:Jasmine romain$ karma init

Which testing framework do you want to use ?
Press tab to list possible options. Enter to move to the next question.
> jasmine

Do you want to use Require.js ?
This will add Require.js plugin.
Press tab to list possible options. Enter to move to the next question.
> no

Do you want to capture any browsers automatically ?
Press tab to list possible options. Enter empty string to move to the next question.
> Chrome
> Safari
>

What is the location of your source and test files ?
You can use glob patterns, eg. "js/*.js" or "test/**/*.js".
Enter empty string to move to the next question.
> 
```

```
Air-de-Romain:Jasmine romain$ karma start
02 09 2015 21:30:11.510:INFO [karma]: Karma v0.13.9 server started at http://localhost:9876/
02 09 2015 21:30:11.518:INFO [launcher]: Starting browser Chrome
02 09 2015 21:30:11.526:INFO [launcher]: Starting browser Safari
02 09 2015 21:30:12.723:INFO [Safari 8.0.7 (Mac OS X 10.10.4)]: Connected on socket HE38slHTBKXL5t5yAAAA with id 54715269
Safari 8.0.7 (Mac OS X 10.10.4): Executed 1 of 1 SUCCESS (0.038 secs / 0.003 secs)
Safari 8.0.7 (Mac OS X 10.10.4): Executed 1 of 1 SUCCESS (0.038 secs / 0.003 secs)
Chrome 45.0.2454 (Mac OS X 10.10.4): Executed 1 of 1 SUCCESS (0.04 secs / 0.008 secs)
TOTAL: 2 SUCCESS
```



- Créé en 2008 par les développeurs de jQuery
- Type xUnit (JUnit, PHPUnit...) : basés sur des assertions
- Plutôt destiné à du code client
- Installation
 - npm install --save-dev qunitjs
 - bower install --save-dev qunit
- Lancement des tests
 - Ouverture du fichier .html
 - grunt-contrib-qunit
 - karma-qunit

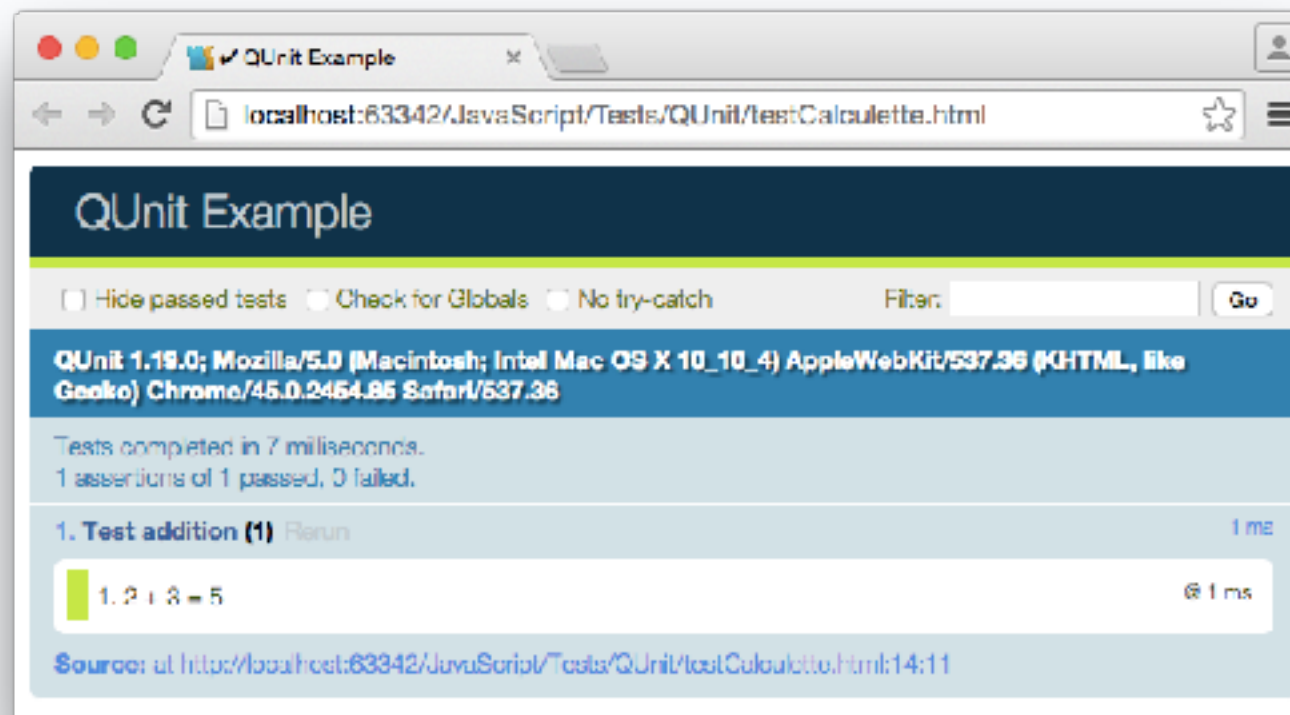


Tests Automatisés - QUnit



```
<!-- runner.html -->
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>QUnit Example</title>
  <link rel="stylesheet" href="node_modules/qunitjs/qunit/qunit.css">
</head>
<body>
<div id="qunit"></div>
<div id="qunit-fixture"></div>
<script src="calcullette.js"></script>
<script src="node_modules/qunitjs/qunit/qunit.js"></script>
<script src="calcullette-test.js"></script>
</body>
</html>
```

```
// calcullette-test.js
QUnit.test("Test addition", function(assert) {
  assert.equal(calcullette.ajouter(2, 3), 5, "2 + 3 = 5");
});
```





- Créé en 2010
- Type BDD (Behavior-Driven Development)
- Fonctionne pour le browser ou node.js
- Installation et lancement des tests (node)
npm install -g jasmine
jasmine init
jasmine
- Installation et lancement des tests (browser)
npm install --save-dev jasmine-core
SpecRunner.html
karma

Tests Automatisés - Jasmine



```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Jasmine Spec Runner v2.3.4</title>

  <link rel="shortcut icon" type="image/png" href="node_modules/jasmine-core/images/
jasmine_favicon.png">
  <link rel="stylesheet" href="node_modules/jasmine-core/lib/jasmine-core/jasmine.css">

  <script src="node_modules/jasmine-core/lib/jasmine-core/jasmine.js"></script>
  <script src="node_modules/jasmine-core/lib/jasmine-core/jasmine-html.js"></script>
  <script src="node_modules/jasmine-core/lib/jasmine-core/boot.js"></script>

  <!-- include source files here... -->
  <script src="calcullette.js"></script>

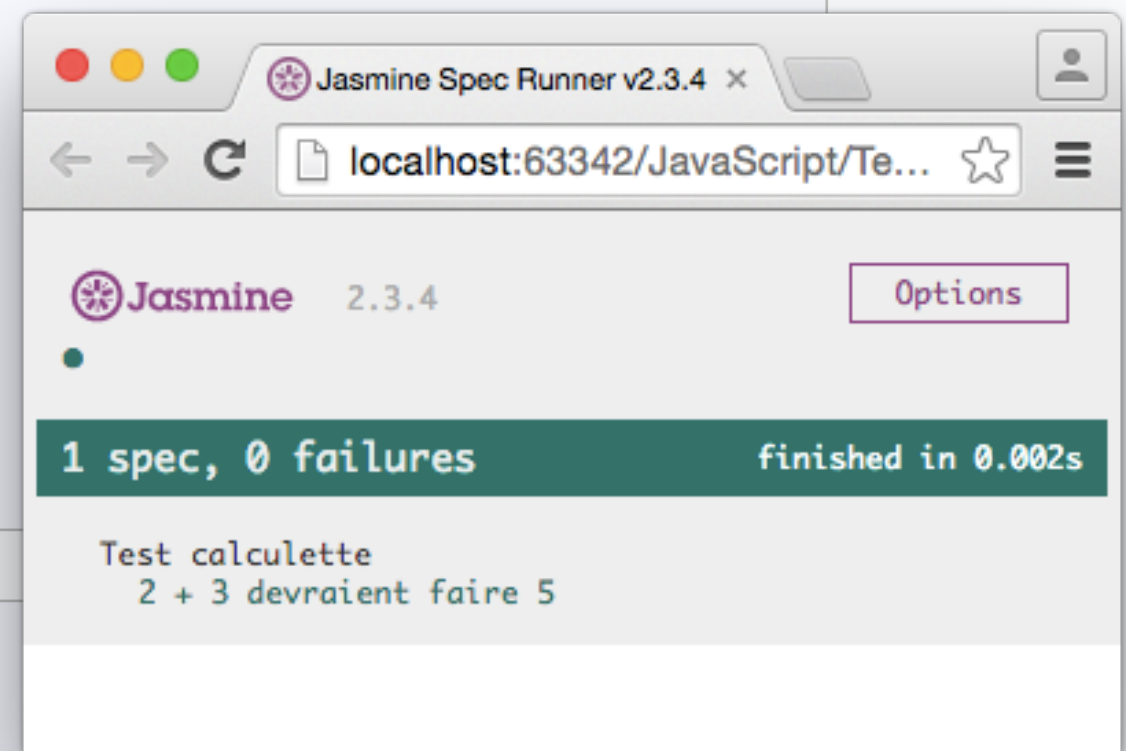
  <!-- include spec files here... -->
  <script src="spec/CalculletteSpec.js"></script>

</head>
<body>
</body>
</html>
```

```
describe("Test calcullette", function() {

  it("2 + 3 devraient faire 5", function() {
    expect(calcullette.ajouter(2, 3)).toEqual(5);
  });

});
```





- Créé en 2011
- Type assert ou BDD (le framework est flexible)
- Fonctionne pour le browser ou node.js
- Installation et lancement des tests (node)
npm install -g mocha
mocha
- Installation et lancement des tests (browser)
npm install -g mocha
mocha init
npm install chai
runner.html
karma

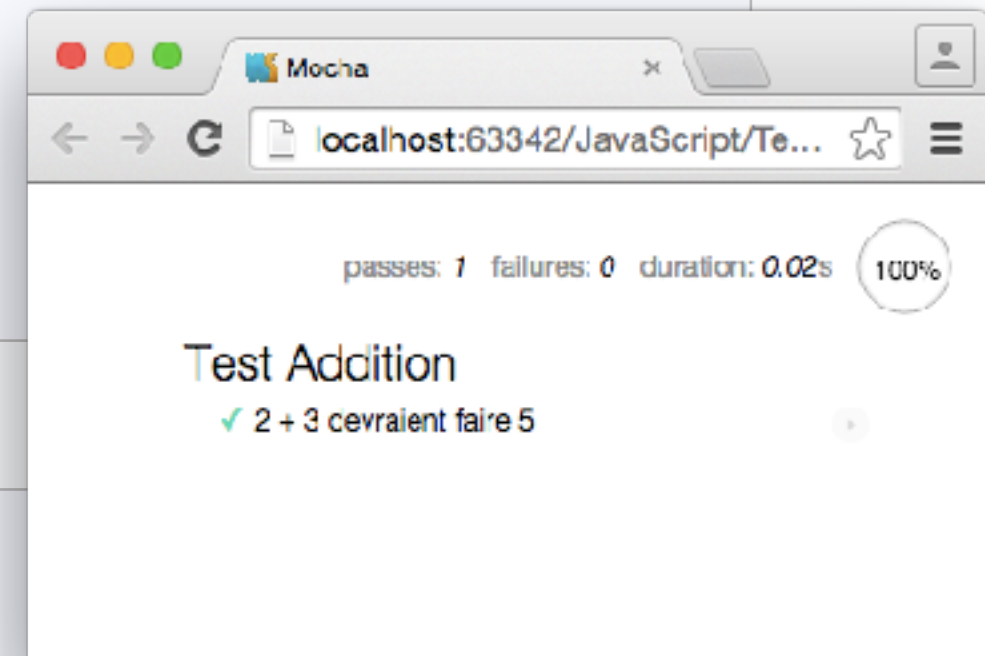
Tests Automatisés - Mocha



```
<!DOCTYPE html>
<html>
  <head>
    <title>Mocha</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="mocha.css" />
  </head>
  <body>
    <div id="mocha"></div>
    <script src="mocha.js"></script>
    <script src="node_modules/chai/chai.js"></script>
    <script>mocha.setup('bdd');</script>
    <script src="src/calcullette.js"></script>
    <script src="test/calcullette-test.js"></script>
    <script>
      mocha.run();
    </script>
  </body>
</html>
```

```
var assert = chai.assert;

describe('Test Addition', function() {
  it('2 + 3 devraient faire 5', function () {
    assert.equal(5, calcullette.ajouter(2, 3));
  });
});
```





Scaffolding



- Scaffolding
Permet d'initialiser des projets déjà structurés
- Yeoman
<http://yeoman.io>
- Installation
`npm install -g yo`
- Générateurs
2132 (juillet 2015)
<http://yeoman.io/generators/>
- Exemple : webapp
`npm install -g generator-webapp`



Scaffolding - Yeoman



► Création d'un projet

`mkdir mon-projet && cd mon-projet`

`yo webapp`

A screenshot of a macOS terminal window titled 'monprojet — node — 61x16'. The prompt is 'Air-de-Romain:monprojet remain\$'. The command 'yo webapp' has been executed. The output features a colorful ASCII art character on the left, a dashed box on the right containing the text: 'Allo 'allo! Out of the box I include HTML5 Boilerplate, jQuery, and a Gruntfile to build your app.', and a list of options at the bottom: '? What more would you like?' followed by '● Sass', '● Bootstrap', and '>● Modernizr'.

```
monprojet — node — 61x16
Air-de-Romain:monprojet remain$ yo webapp

  |  |  |
  |--(o)--
  |(_U_)|
  /__A__\
  |  ~  |
  |___|_|
  |  |  |  Y

'Allo 'allo! Out of the
  box I include HTML5
  Boilerplate, jQuery, and
  a Gruntfile to build
  your app.

? What more would you like?
● Sass
● Bootstrap
>● Modernizr
```