

NODE.JS

BOOTSTRAP

Installation de Node.js

Installation

Installer Node.js sur <http://nodejs.org/>

Deux programmes sont installés :

NPM : le gestionnaire de package de Node.js

Node.js command prompt :

Une console de Windows configurée pour reconnaître Node.js

Lancer les programmes Node.js



Installation

Pour Windows, choisissez .exe ,
pour MAC OS .pkg et pour
Linux .tar.gz

Suivre ensuite l'installation



Download the Node.js source code or a pre-built installer for your platform, and start developing today.

Current version: v0.10.4



Windows Installer
node-v0.10.4-x86.msi

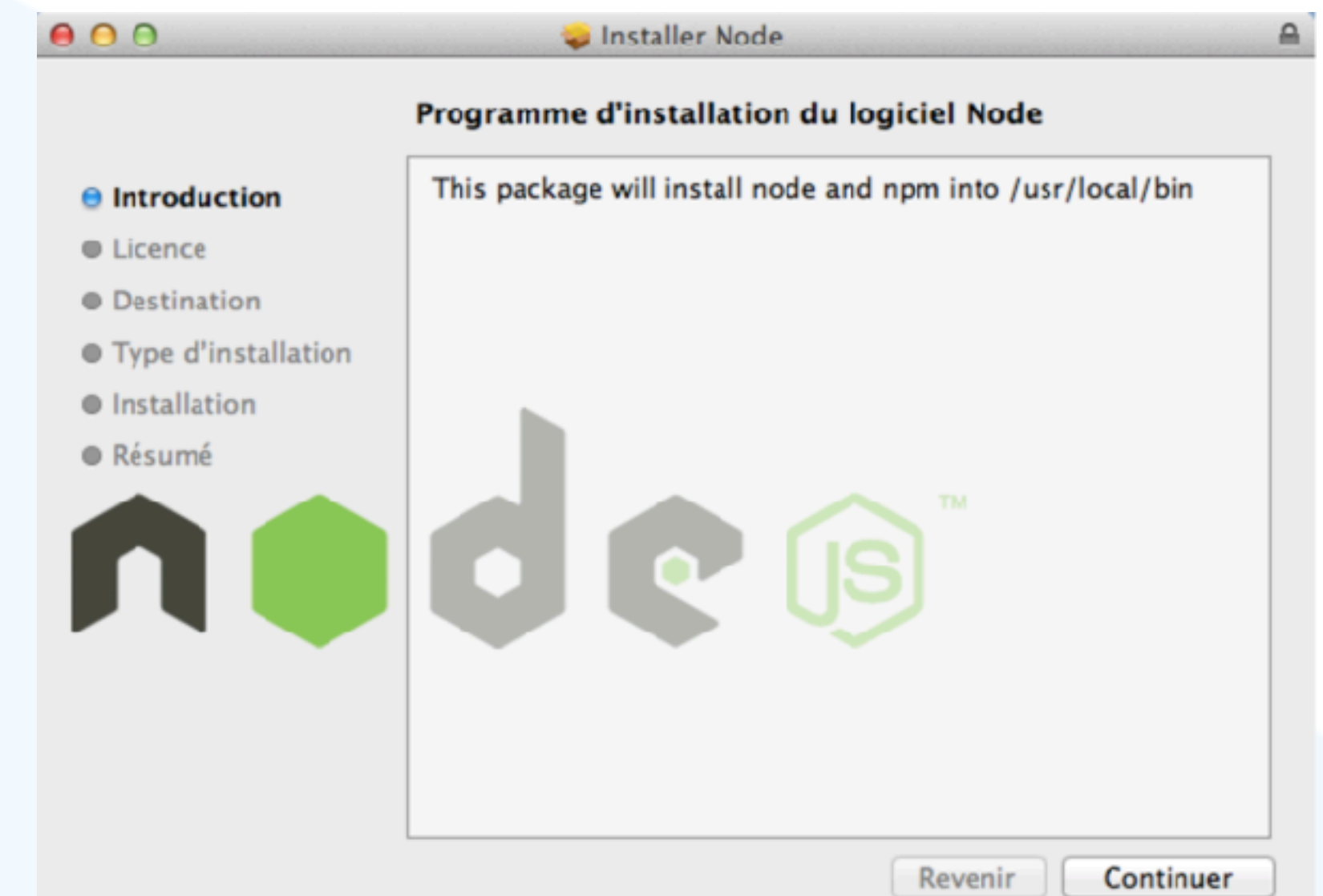


Macintosh Installer
node-v0.10.4.pkg



Source Code
node-v0.10.4.tar.gz

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.exe)	32-bit	64-bit
Mac OS X Installer (.pkg)	Universal	
Mac OS X Binaries (.tar.gz)	32-bit	64-bit
Linux Binaries (.tar.gz)	32-bit	64-bit
SunOS Binaries (.tar.gz)	32-bit	64-bit
Source Code	node-v0.10.4.tar.gz	



Tester si Node.js fonctionne

Tester node.js avec un code minimal :

1) Pour commencer, ouvrez votre éditeur de texte favori (Sublime Text, VSCode...) et rentrez le code JavaScript suivant :

```
1 console.log('Bienvenue dans Node.js !');
```

2) Enregistrez le fichier sous l'extension .js. Ici, test.js.

3) Ouvrez un terminal et placez vous dans le dossier du fichier .js et exécutez :

```
$ node test.js  
Bienvenue dans Node.js !
```

1) Histoire de Node.JS

Historique

Crée par Rayan Lienhart Dahl en 2009

Écrit en C/C++



Environnement d'exécution Javascript



Basé sur le moteur V8 Google Chrome



Qu'est ce que Node.JS?

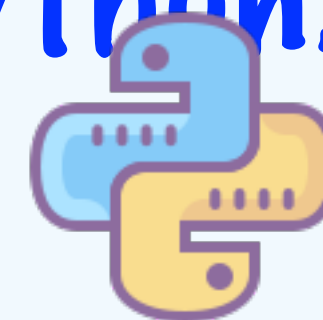


C'est un langage de programmation dit « back end »

Utilisation de Javascript V8 en dehors du navigateur

Utilisé pour écrire des services côté serveur appelés API (Application Programming Interface)

Alternative à des langages serveur comme PHP, Java ou Python.



Cible applicative

Utilisée pour des applications Web et Mobiles



Destiné pour Android , IOS et Windows

Pour les applications web et mobiles => Utilisation de Javascript V8

Caractéristiques de Node.js

❖ Utilisation de RTA et SPA

RTA =

Real Time Applications, ce sont les applications en temps réel, ce sont ces applications qui ont besoin de se mettre à jour plus fréquemment.

SPA =

Ce sont les initiales de Single Page Applications. Ce sont des applis dans lesquelles il n'y a qu'une page html et le contenu de cette page change en fonction des actions de l'utilisateur.

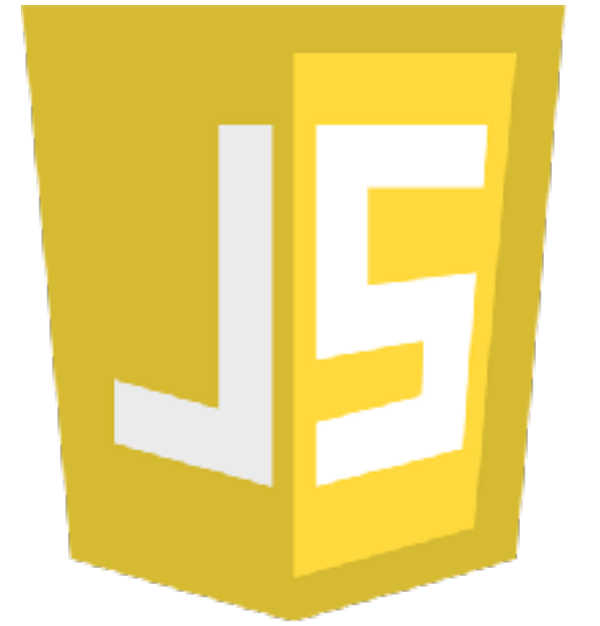
❖ Multithread et non bloquant

Le multithread est la capacité à effectuer plusieurs tâches en même temps.

Non bloquant signifie la capacité à lancer une tâche sans forcément attendre qu'elle se finisse pour passer à la suivante.

Avantages

S'appuie seulement sur du javascript côté serveur & côté client



Requêtes asynchrones => serveur non bloquant

Rapide et évolutif



Communauté très dynamique !



Inconvénients

API instable : Manque de cohérence



L'API de Node.js change fréquemment
Modifications parfois incompatibles avec les versions
antérieures.

Temps de développement plus important:



Node.js nécessite d'écrire tout son code à partir de zéro
Diminution de la productivité et ralentir le travail du développeur.

2) La tendance du marché

Les frameworks les + utilisés

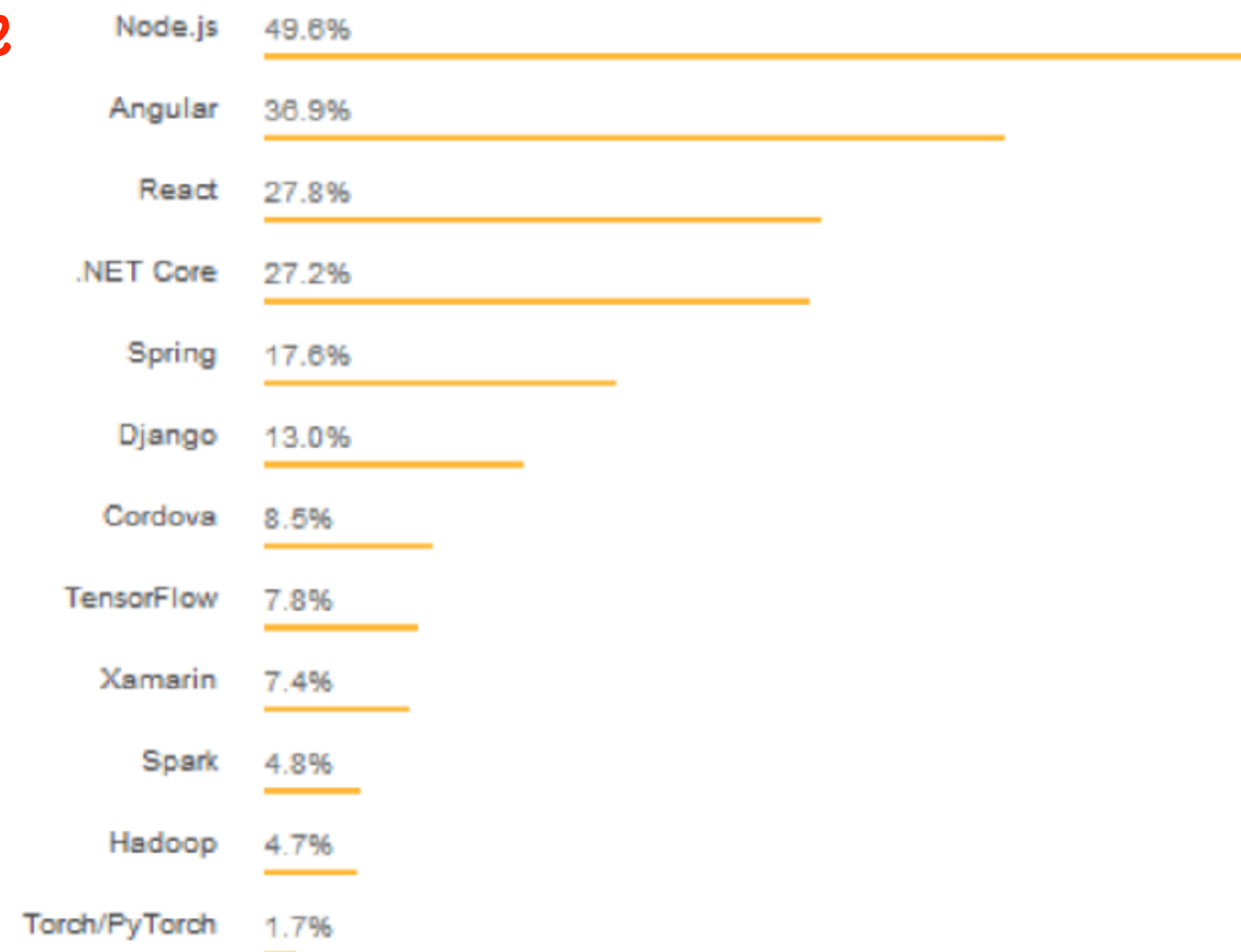
Stack Overflow dresse le classement des frameworks les plus utilisés

Node.js arrive en tête

Frameworks, Libraries, and Tools

All Respondents

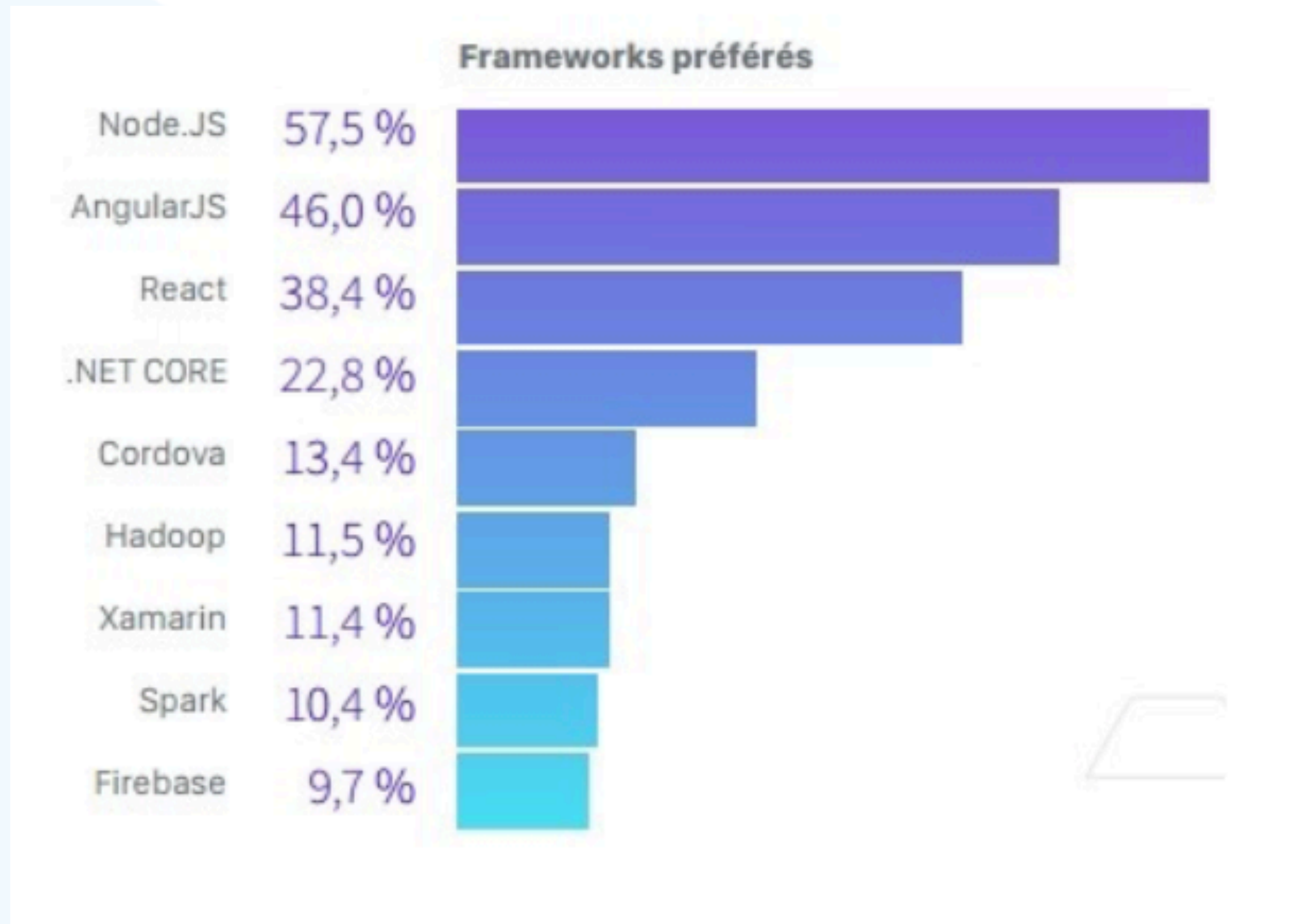
Professional Developers



51,620 responses; select all that apply

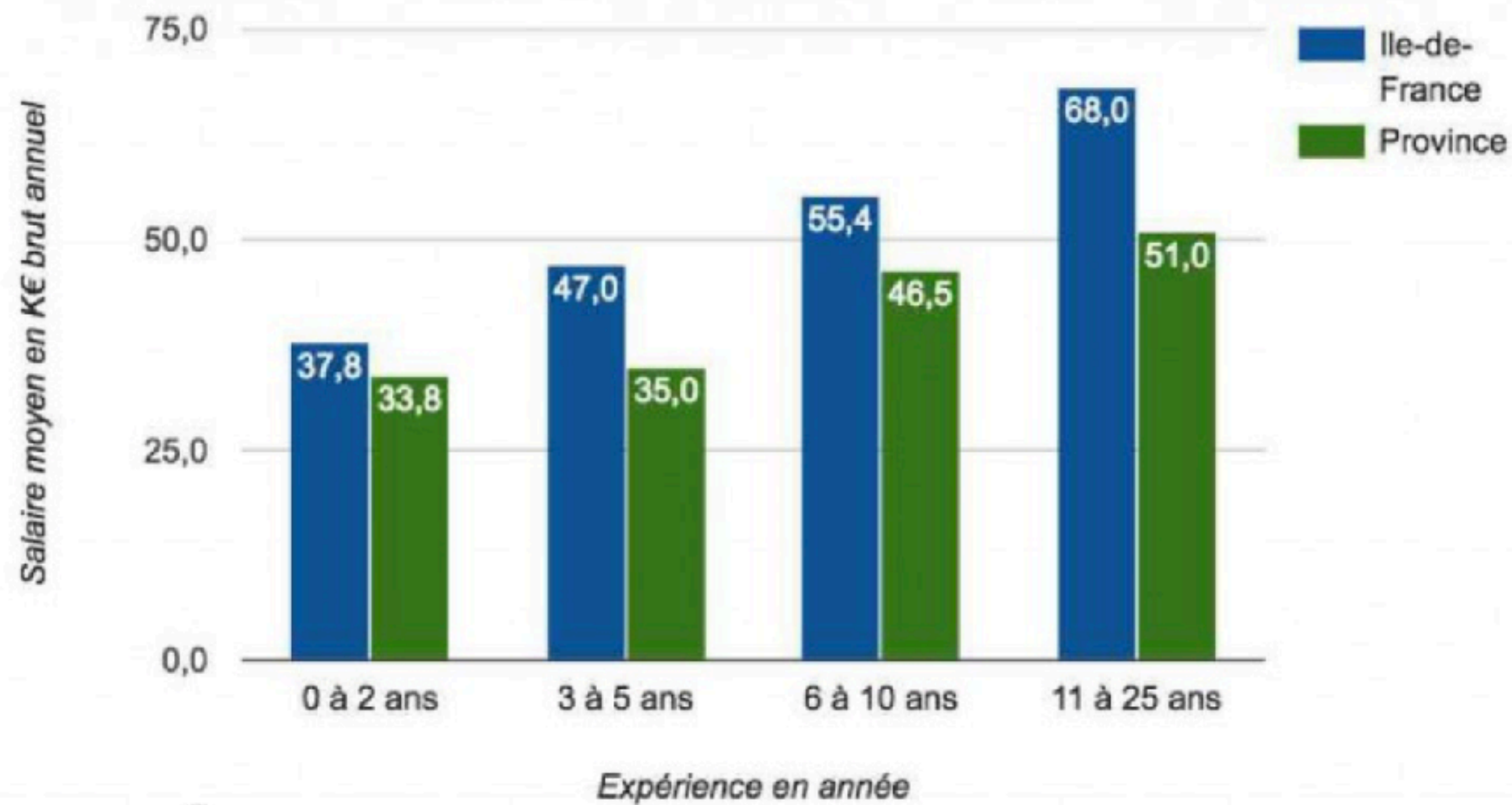
Les frameworks les + appréciés

Stack Overflow dresse le classement des frameworks les plus appréciés

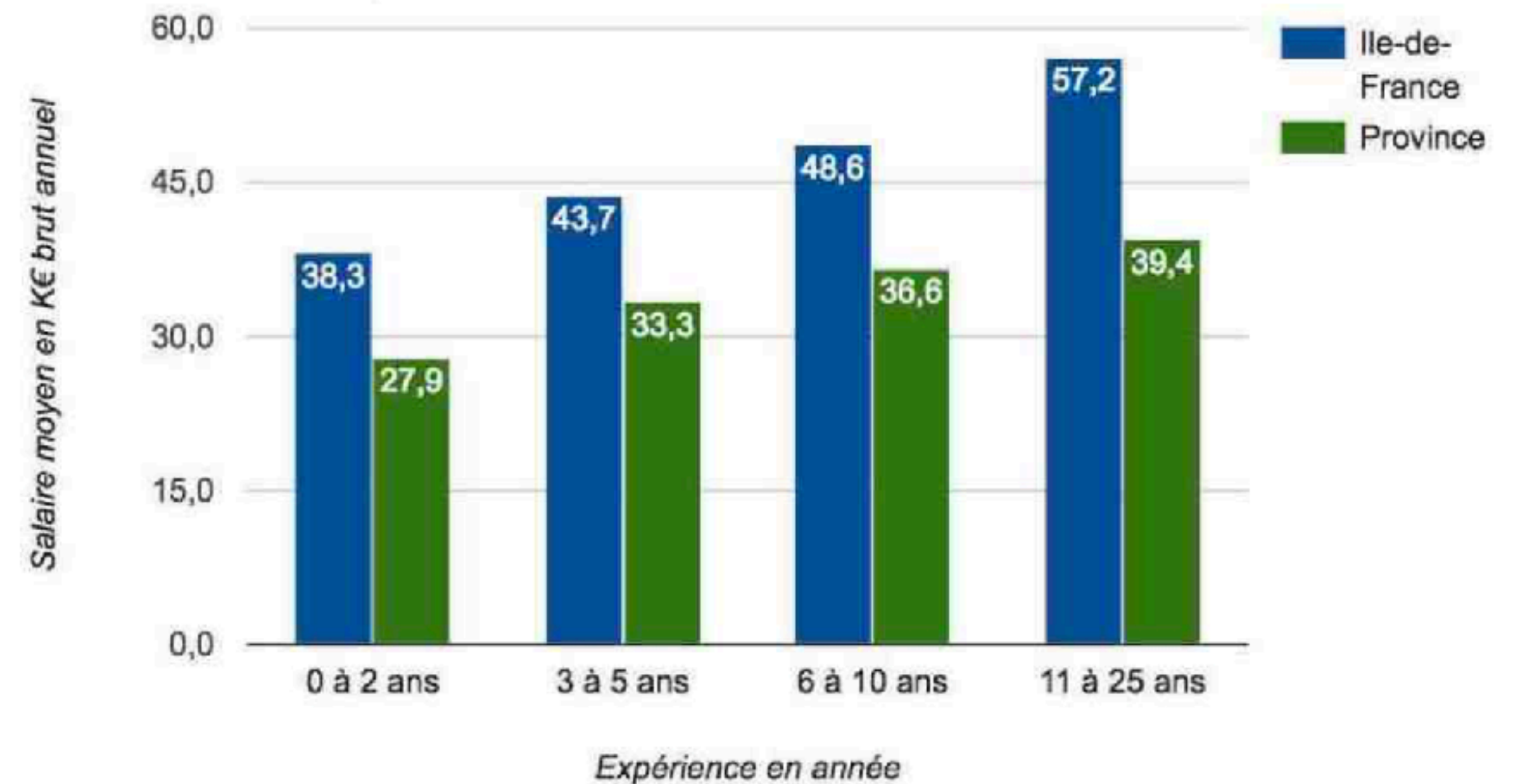


Salaire Node.JS vs salaire PHP

Salaire développeurs back-end JavaScript maîtrisant node.js



Salaire moyen des développeurs PHP maîtrisant un ou plusieurs frameworks



➡ Le salaire d'un dev node.js s'accroît davantage avec les années que dev PHP.

3) Utilisation de Node.JS

Utilisation

Application avec GUI (Graphical User Interface)



Réponds à des requêtes rapidement et efficacement en temps réel

Support de sockets



Exemple d'utilisation :
Linkedin, ebay, Yahoo, ..



Développer avec Node.JS

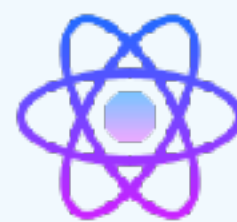


But : Construire une application Web avec des belles mises en page, des formulaires, des appels de serveur asynchrones à une base de données.



Nombreuses bibliothèques et outils disponibles

Front-End : Framework Angular , React, Bootstrap

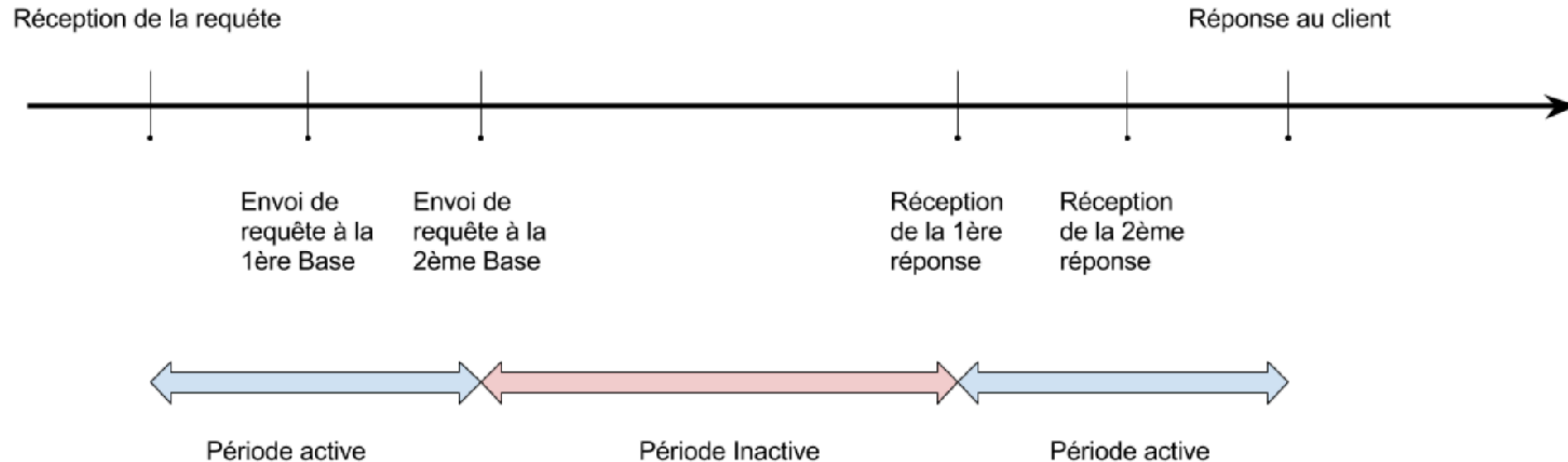


Exemple Monothread VS Multithread

J'envoie 1 requête à mon serveur et ce dernier doit interroger
2 bases de données pour recouper les informations et me répondre.

Exemple Monothread VS Multithread

Ce qu'il va se produire avec Node.js :



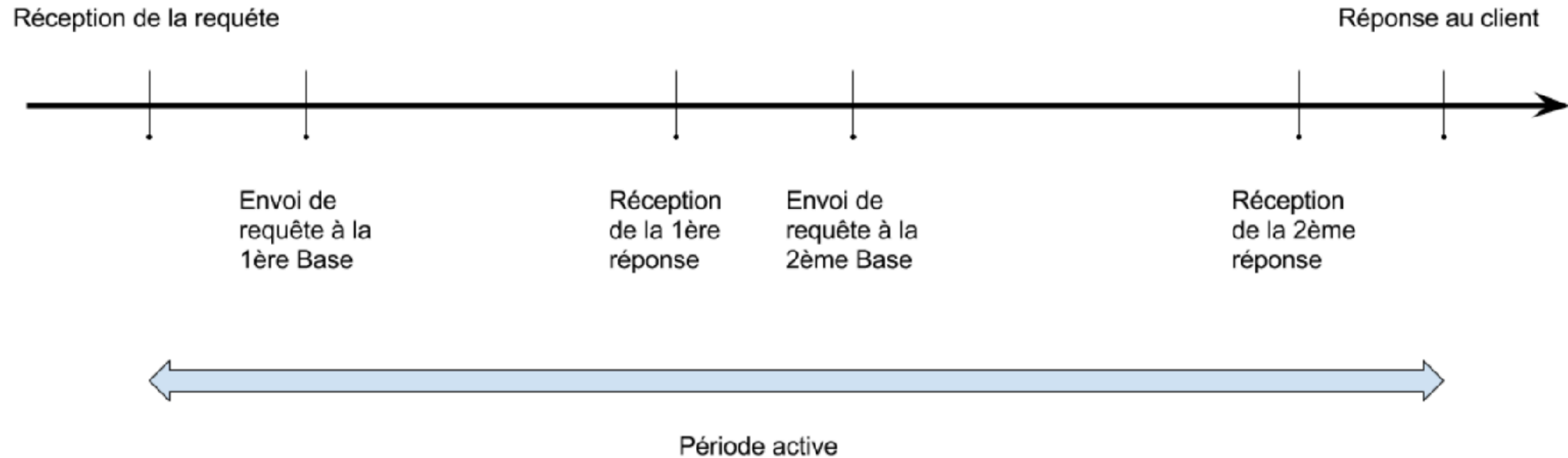
Exemple Monothread VS Multithread

Une fois que le NodeJS a envoyé ses 2 requêtes aux 2 bases, il va alors pouvoir s'occuper d'autres requêtes ou s'il n'a aucune opération à réaliser, se mettre en pause.

Quand les réponses vont arriver, NodeJS va donc terminer la tâche en cours (ou se réveiller) et va poursuivre le processus pour finalement répondre au client.

Exemple Monothread VS Multithread

Ce qu'il va se produire avec PHP :



Exemple Monothread VS Multithread

- 1) Envoi de la 1er requête
- 2) le processus qui exécute le code PHP va attendre la réponse
- 3) Une fois qu'il a reçu il va envoyer la 2ème requête
- 4) Le processus attend la réponse
- 5) Le processus envoie le retour au client

Pendant les temps d'attente, le processus est dit "figé" et ne peut exécuter aucune autre instruction.

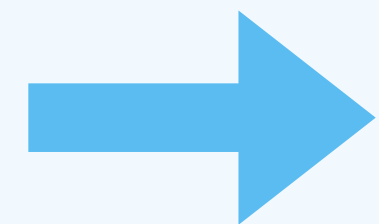
4) Structure de l'application

Module Node.JS

Environnement
modulaire

Différents modules inclus :

« fs » : système de fichier
« http » : serveur web
« Net » : réseau , TCP



Utilisation de NPM pour installer les modules



Qu'est ce qu'un module?

Librairie dédiée à Node.JS

On peut grâce aux modules :



Déclarer des objets



Utiliser leurs fonctions



Gérer une base de donnée

Comment utiliser un module?

Pour faire appel et créer une instance d'un module
On fait appel à « require »

Où ça ?

Dans le fichier.js de votre application



Exemple d'utilisation

Imaginons que nous utilisons le module Mysql pour accéder et manipuler une base de donnée

```
var mysql      = require('mysql');  
var connection = mysql.createConnection({  
  host      : 'localhost',  
  user      : 'me',  
  password  : 'secret',  
});  
  
connection.connect();  
  
connection.query('SELECT * FROM table', function(err, rows, fields) {  
  if (err) throw err;  
  
  console.log('The solution is: ', rows[0].solution);  
});  
  
connection.end();
```

On instance un nouvel objet mysql avec le 'require'

Création un objet connection grâce aux informations de connexion à la base de données

Un simple appel de la fonction connect() sur l'objet connection permettra de se connecter à la base de données.

Une fois la connexion établie, on peut effectuer une requête grâce à la fonction query() qui affichera ses résultats sur la console grâce à une fonction de callback.

Puis, on se déconnecte de la base de données.

Structure des fichiers

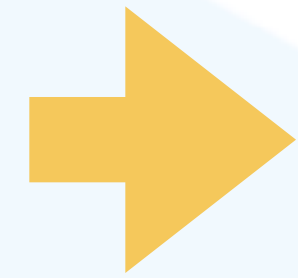
- **Modules : Scope locaux ou globaux**
- **Fichier JS**
- **Dossier client**
- **Package.json => Informations relatives au projet
(Nom, licence, dépendences,...)**

5) Les principaux packages

socket.io et request

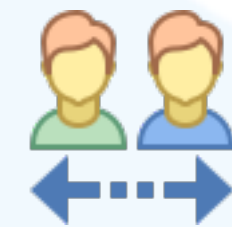
- Request : Client HTTP et utilisable avec tout type d'API
- socket.io : permet d'utiliser des sockets
Exemple : une application de Chat

Le module socket.io



Modules les plus utilisés et les plus utiles.

Permet de créer des WebSocket afin que le serveur puisse communiquer avec un ou plusieurs clients.



Echange de tous types de données via ces WebSockets (texte, binaire, JSON, tableaux ..



Le module socket.io

Exemple d'une socket côté serveur :

```
var io = require('socket.io').listen(8080);  
io.sockets.on('connection', function (socket) {  
  socket.emit('connect', "Bienvenue sur le serveur d'opérations mathématiques.");  
  socket.on('addition', function (data) {  
    sockets.emit('resultat', data['a']+data['b']);  
  });  
  socket.on('soustraction', function (data) {  
    sockets.emit('resultat', data['a']-data['b']);  
  });  
});
```

Instanciation d'un objet io qui écoute sur le port 8080

Sur chaque événement connection d'un socket à notre objet io
Nous envoyons un message de bienvenue à cette socket.

On lui demande d'écouter deux événements, une addition et l'autre soustraction

Si jamais on capte l'un de ces deux événements, on envoie à toutes les sockets clients l'événement "résultat" avec le résultat de l'opération des deux valeurs passées dans un JSON par un des clients.

Le module socket.io

Exemple d'une socket côté client :

Pour pouvoir utiliser les SocketIO, il faut mettre "/socket.io/socket.io.js" comme attribut src d'une balise script

```
var socket = io.connect('http://localhost:8080');  
    Se connecte au serveur en écoutant sur le port 8080  
socket.emit('addition', { a: 10, b: 20 });  
socket.emit('soustraction', { a: 2, b: 1 });
```

Il envoie deux événements, un soustraction et un autre addition, avec les deux opérandes passées en JSON

Exemple d'utilisation de Node.js

```
1 var http = require('http');
2
3 var server = http.createServer(function(req, res) {
4   res.writeHead(200);
5   res.end('Salut tout le monde !');
6 });
7 server.listen(8080);
```

- 1) Require effectue un appel à une bibliothèque de Node.js, ici la bibliothèque "http" qui nous permet de créer un serveur web
- 2) On appelle la fonction `createServer()` contenue dans l'objet `http` et on enregistre ce serveur dans la variable `server`.
- 3) Tout le code ci-dessus correspond à l'appel à `createServer()`. Il comprend en paramètre la fonction à exécuter quand un visiteur se connecte à notre site.

Le module socket.io

Exemple d'une socket côté client :

Pour pouvoir utiliser les SocketIO, il faut mettre "/socket.io/socket.io.js" comme attribut src d'une balise script

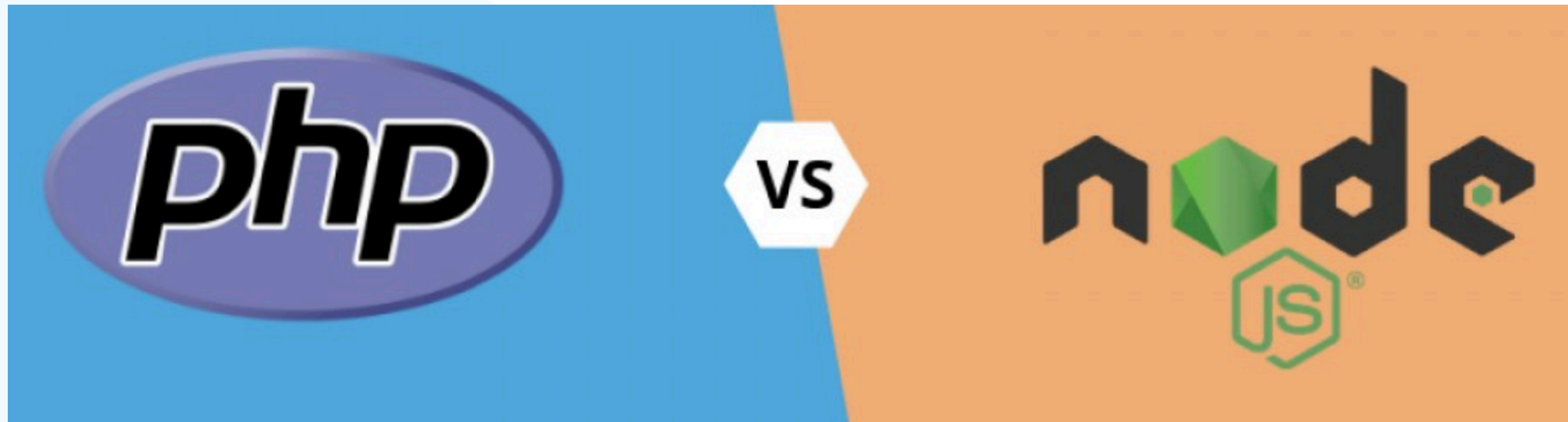
```
var socket = io.connect('http://localhost:8080');  
    Se connecte au serveur en écoutant sur le port 8080  
socket.emit('addition', { a: 10, b: 20 });  
socket.emit('soustraction', { a: 2, b: 1 });
```

Il envoie deux événements, un soustraction et un autre addition, avec les deux opérandes passées en JSON

6) Les concurrents

Les technologies concurrentes

Principal concurrent : PHP



PHP : Avantages et inconvénients

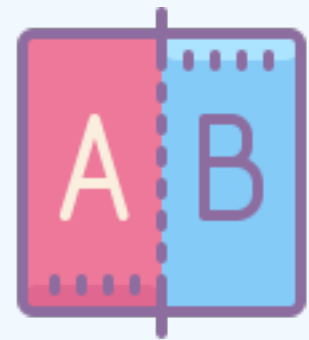
- Avantages

- * Peut être utilisé directement au sein des pages HTML grâce aux balises `<?php ?>`.
- * Utilise la programmation procédurale qui est plus facile à apprendre que la programmation orientée objet.
- * Il n'est pas typé et il n'y a pas besoin de déclarer les variables avant de s'en servir.

- Inconvénients

- * Il n'a pas de réel gestionnaire de paquets
- * PHP est sujet à de nombreuses vulnérabilités (Injections SQL, ...)
- * Il n'y a pas de réel API
- * Performance basse

Comparatif Node.js/PHP



Selon le rapport 2018 de JavaScript, Node.js est utilisé par 63% des développeurs contre 50% en PHP.

PHP est asynchrone : la requête reste bloquée en attendant une réponse, et ne peut pas passer à une autre requête, (contrairement à node.js)



Node.js : Même langage pour le développement front-end et back-end : le code se fait en JavaScript côté serveur et côté client, ce qui permet d'augmenter la productivité.

Node.js prend en charge le côté serveur contrairement à PHP.

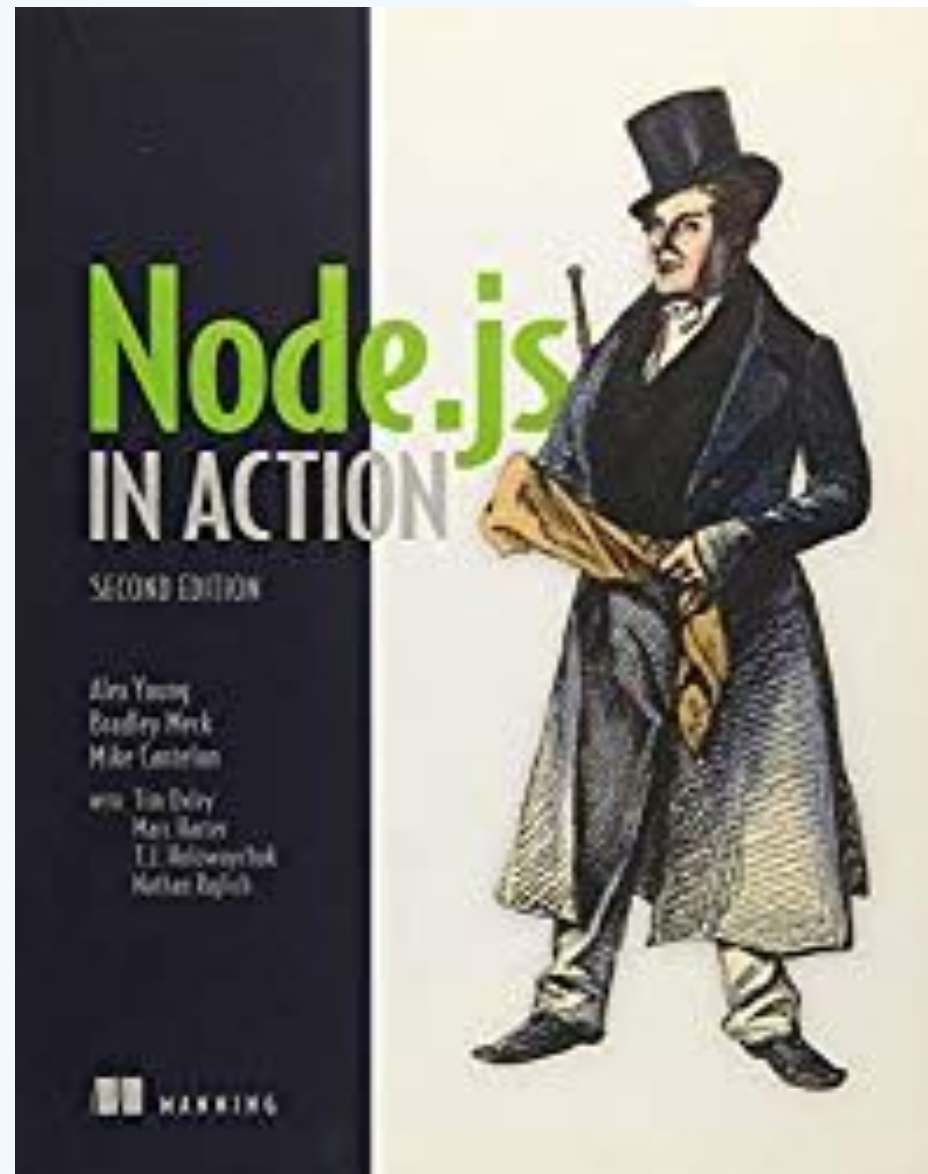


PHP est plus lent que Node.js

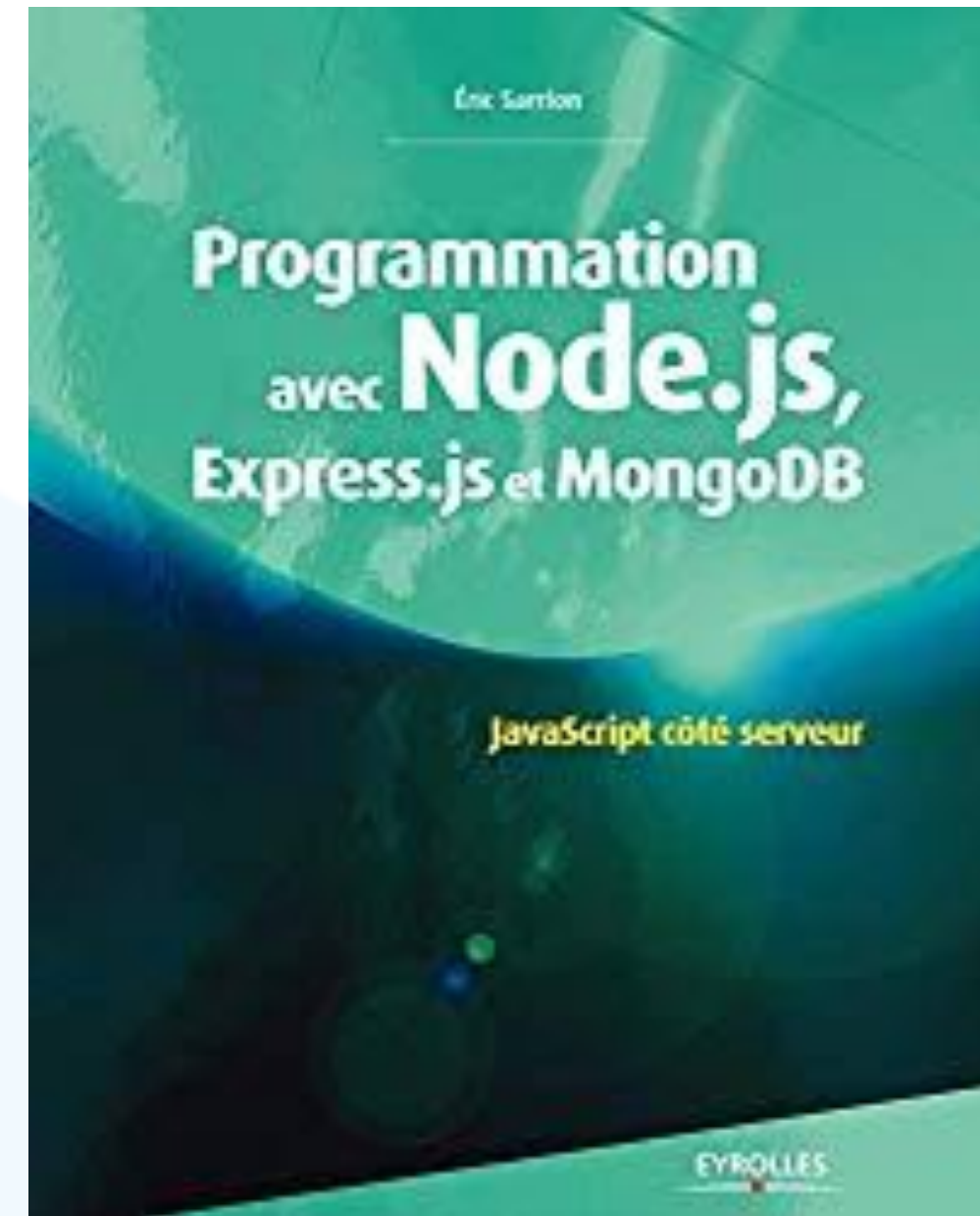


7) Référence bibliographiques

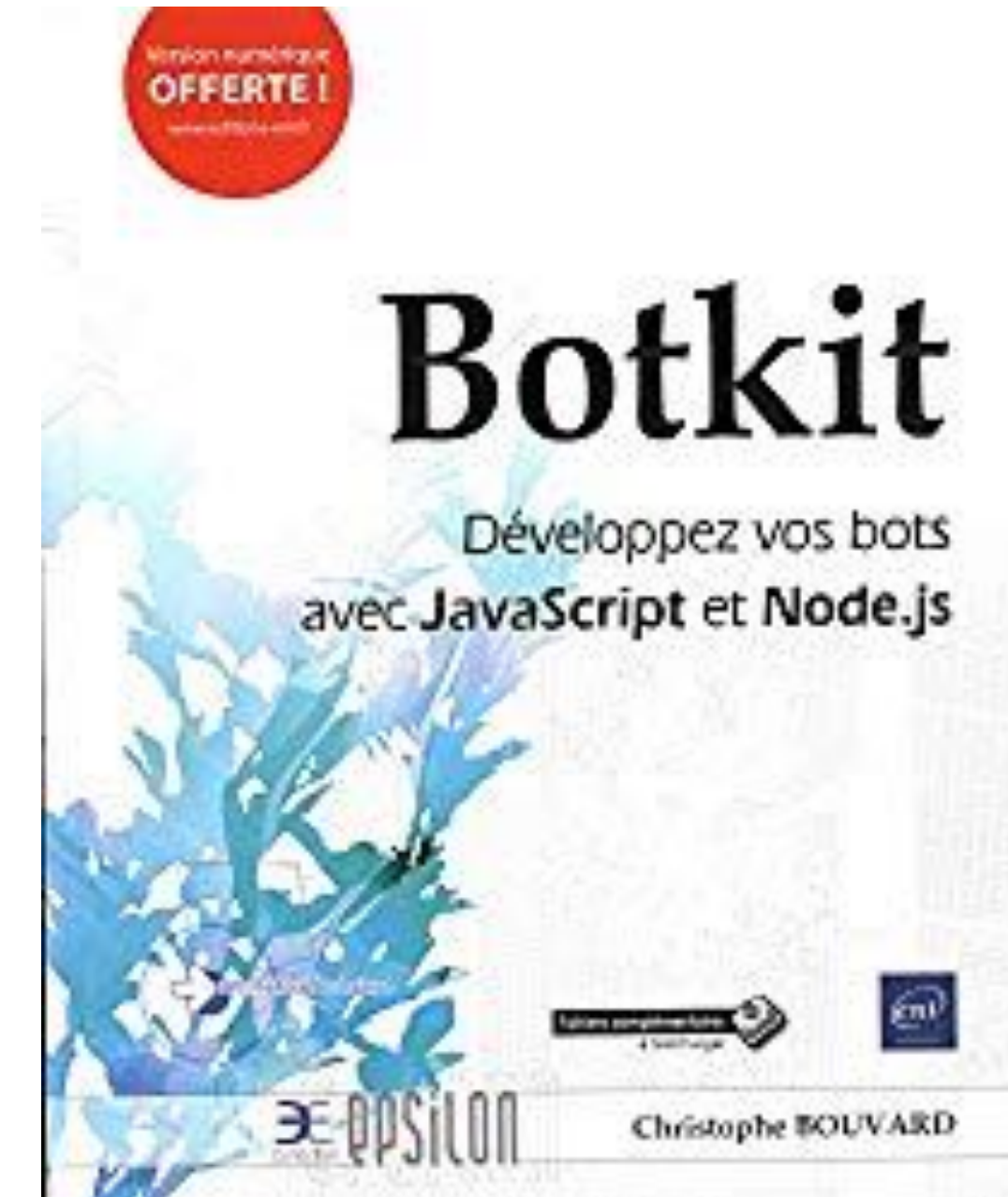
Références bibliographiques



Node.js In Action



Programmation avec Node.js, Express.js et MongoDB : JavaScript coté serveur
Livre écrit par Éric Sarrion



Botkit - Développez vos bots avec JavaScript et Node.js
Livre écrit par Christophe BOUVARD

II) BOOTSTRAP



KÉSAKO?



Crée en 2011 par les développeurs de Twitter



C'est un kit CSS, devenu le framework de référence CSS







Offre des Plugins JQuery de qualité pour enrichir les pages





Embarque des composants HTML et JavaScript

Les intérêts et les inconvénients

Avantages

-  Gain de temps de développement car ils proposent les fondations de la présentation.
-  Normalise la présentation en proposant un ensemble homogène de styles.
-  Propose une grille pour faciliter le positionnement des éléments.
-  Offre des éléments complémentaires : boutons esthétiques, barres de navigation, etc...
-  Prise en compte de tailles d'écran très variées

Inconvénients

-  Nécessite un temps d'apprentissage long pour bien connaître le framework.
-  La normalisation de la présentation peut devenir lassante en lissant les effets visuels.

Contenu du Kit Bootstrap



Mise en page basée sur une grille de 12 colonnes. Si on a besoin de plus de 12 colonnes, ou de moins, il est toujours possible de changer la configuration ;

Du code fondé sur HTML5 et CSS3 ;

Une bibliothèque totalement open source sous license MIT;

Du code qui tient compte du format d'affichage des principaux outils de navigation (responsive design) : smartphones, tablettes... ;

Des plugins jQuery de qualité ;

Une bonne documentation ;

La garantie d'une évolution permanente ;

Installation de Bootstrap

Très simple !

1) Allez sur : <https://getbootstrap.com/docs/4.3/getting-started/introduction/>

2) Il y a 3 boutons :

- "Download Bootstrap" : permet de récupérer juste les fichiers nécessaires au fonctionnement de Bootstrap.
- "Download source" : permet de récupérer en plus tous les fichiers sources.
- "Download Sass" : c'est un portage de Bootstrap en Sass pour les utilisateurs de projets qui utilisent Sass (Rails, Compass...).

3) Choisissez « Download Bootstrap »

4) Suivez ensuite les instructions à l'écran

Installation de Bootstrap

Si on utilise Node.js on effectuera ces commandes depuis son terminal:

```
npm install -g bower
```

```
bower install <package>
```

Utilisation de Bootstrap

Il reste plus qu'à placer les balises suivantes à l'intérieur de votre fichier entre les balises HEAD du document

```
<script src="https://code.jquery.com/jquery-3.1.1.min.js"></script>

<link crossorigin="anonymous"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
rel="stylesheet">

<link crossorigin="anonymous"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css"
integrity="sha384-rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp"
rel="stylesheet">

<script crossorigin="anonymous"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNlCPD7Txa"
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js">
```

Test de Bootstrap

Maintenant que Bootstrap est instancié, il faut le tester :

```
<!DOCTYPE>

<head>
  <script src="https://code.jquery.com/jquery-3.1.1.min.js"></script>
  <link rel="stylesheet" href="css/bootstrap.min.css">
  <link rel="stylesheet" href="css/bootstrap-theme.min.css">
  <script src="js/bootstrap.min.js"></script>
</head>

<body>
  <h1>Test avec Bootstrap</h1>
</body>
```

Comparaison

AVEC BOOTSTRAP

file:///C:/Users/javat/Desktop/Bootstrap/bootstrap.html

Test avec Bootstrap

```
<!DOCTYPE>
<head>
  <link rel="stylesheet" href="css/bootstrap.min.css">
  <link rel="stylesheet" href="css/bootstrap-theme.min.css">
  <script src="js/bootstrap.min.js"></script>
</head>
<body>
  <h1>Test avec Bootstrap</h1>
</body>
```

SANS BOOTSTRAP

file:///C:/Users/javat/Desktop/Bootstrap/bootstrap.html

Test Bootstrap

```
<!DOCTYPE>
<head>
</head>
<body>
  <h1>Test Bootstrap</h1>
</body>
```


Référence bibliographique



**Bootstrap 4 pour l'intégrateur web - CSS et Responsive
Web Design**
écrit par Christophe AUBRY



**Bootstrap 3
Le framework 100% Web Design**
écrit par Benoit Philibert

