

2020



express



# Programmation Web

Gildas Ménier gildas.menier@univ-ubs.fr

# Introduction

- Technologies et développement Web
- Base :
  - HTML & CSS, PHP
  - Mécanismes Client Serveur
  - **HTTP (à connaître)**
  - Sockets
  - Servlets
  - Bases de la programmation Java
  - Bases de la programmation C++
  - Fonctionnel (Java / Scala)
  - XML / XSLT
  - **GIT !! <https://git-scm.com/>**
- **Conditions cours : voir programmation multi paradigme**

# Introduction

## HTTP

### RFC 2616 - Hypertext Transfer Protocol -- HTTP/1.1 - IETF Tools

<https://tools.ietf.org/html/rfc2616> ▾ Traduire cette page

de PJ Leach - 1999 - Cité 21 fois - Autres articles

**RFC 2616** HTTP/1.1 June 1999 resource A network data object or service that can be identified by a URI, as defined in section 3.2. Resources may be available ...

#### RFC 7230

RFC 7230 HTTP/1.1 Message Syntax  
and Routing June 2014 ...

#### RFC 7232

RFC 7232 HTTP/1.1 Conditional  
Requests June 2014 Table of ...

Autres résultats sur ietf.org »

#### RFC 7233

RFC 7233 HTTP/1.1 Range Requests  
June 2014 Table of ...

#### RFC 2068

RFC 2068 HTTP/1.1 January 1997  
3.2 Uniform Resource Identifiers ...

**JS** Javascript





# Javascript : historique

- EcmaScript
- Mosaic → Netscape : liveScript
  - liveScript → javascript
- 1995 : interpréteur dans Netscape
- Microsoft propose JScript dans IE
- Netscape râle et demande standardisation

ECMA  
European association for  
standardizing information and  
communication systems



# Javascript



<http://www.ecma-international.org/>

Application	Dénomination	Correspondance ECMAScript
Navigateurs de type Gecko avec le moteur embarqué SpiderMonkey, dont Mozilla Firefox	JavaScript	ECMA-262, edition 3 <sup>1</sup>
Internet Explorer	JScript	ECMA-262, edition 3 <sup>6</sup>
Opera	ECMAScript, avec des extensions JavaScript et JScript	ECMA-262, edition 3
KHTML based browsers, including KDE's Konqueror	JavaScript	ECMA-262
Framework .NET de Microsoft	JScript .NET et Managed JScript	ECMA-262, edition 3 <sup>2</sup>
Adobe Flash	ActionScript	ECMA-262, edition 3 <sup>3</sup>
Adobe Acrobat	JavaScript	ECMA-262, edition 4 <sup>4</sup>
General purpose scripting language	DMDScript	ECMA-262, edition 3
OpenLaszlo Platform	JavaScript	ECMA-262, edition 3 <sup>5</sup>
iCab	InScript	ECMA-262, edition 3
Implémentation d'XML dans les navigateurs basés sur Gecko et les programmes embarqués comme SpiderMonkey	E4X	ECMA-357, edition 2



# Environnement

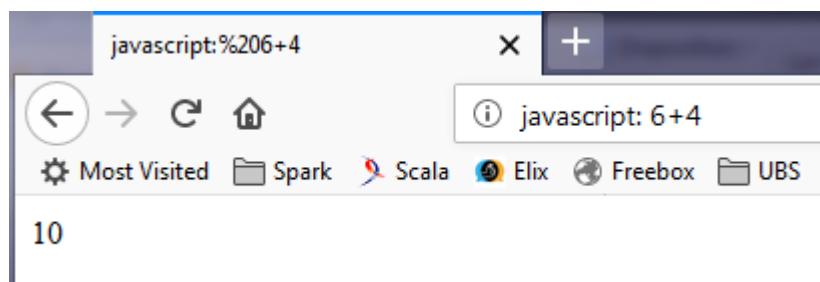
- Navigateur

- HTML

dans un fichier .html  
possible sans <html> etc

```
<script>  
document.write("hello")  
</script>
```

- javascript:



javascript:2+2  
javascript:for(i=0; i<10; i++) alert(i);

url

# Environnement



- **Java**

- Java 7 : Interpréteur Rhino
- Java 8 : Nashorn
- Lien avec Java
- Interpréteur en Java

- **Node.js**

- JIT

[voir introduction à node.js](#)



# Développement



- Eclipse + JSDT
- Komodo IDE
- Netbeans
- Sublime Text
- WebStorm
  - IntelliJ IDEA
- Notepad++
- TextMate
- Etc..



# WebStorm

The screenshot shows the WebStorm IDE interface. The code editor displays `main.js` with several functions: `querySelector`, `closeMenu`, and `toggleMenu`. A tooltip for `closeMenu` is visible, showing its definition and a warning about multiple definitions. The file browser on the left shows the project structure with files like `index.html`, `manifest.webapp`, and `styleguide.html`. The Gulp task list shows various build steps. The bottom panel features the debugger, showing the call stack and local variables for the current frame.

Free for students:  
Professional developer tools  
from JetBrains



Are you learning Java, PHP, Ruby, Python, JavaScript, Objective-C or .NET technologies?

Or maybe you just plan to? Do it right from the start, with award-winning professional developer tools from JetBrains. And the best part: it's free of charge.

**ReSharper**  
Visual Studio extension for .NET

**IntelliJ IDEA**  
The most intelligent IDE for Java

**PhpStorm**  
IDE for Web & PHP

**dotTrace**  
.NET performance profiler

**RubyMine**  
IDE for Ruby and Rails

**WebStorm**  
Smart JavaScript IDE

**dotCover**  
.NET code coverage tool

**PyCharm**  
Powerful Python & Django IDE

**AppCode**  
Smart iOS/XC IDE

**dotMemory**  
.NET memory profiler

**ReSharper C++**  
Visual Studio extension for C++

**CLion**  
Cross-platform C/C++ IDE

# Références

- Prérequis
  - CSS 3-4 (à connaître)
  - HTML 5 (à connaître)
  - HTTP 1.1 / RFC 2616 (à connaître)
- Livres & citations
  - **JavaScript: The Definitive Guide** David Flanagan
  - **Javascript Cookbook** Shelley Powers
  - **Secrets of the JavaScript Ninja** John Resig
  - **Javascript: The Good Parts** D.Crockford
  - **Functionnal programming in Javascript** Luis Atencio

# Introduction

- Industrie & formation :
  - **classiquement**
  - C, C++ en développement ..
  - Java ..
  - Objective-C et développements Android, MacOs etc..
  - Web : PHP avec un peu de Javascript ..
  - Langage coté serveur
  - Langage coté client
  - Plateformes hétérogènes : pleins d'outils, des langages différents
  - Compatibilités des matériels
  - Mises à jour (versions suivantes) et rétrocompatibilité
  - Passage au cloud difficile : hétérogénéité à gérer...
  - **10 ans d'études et 5-6 langages**

# Introduction

- Industrie & formation :
  - Full stack & fonctionnel
  - Développement d'applications web
  - Développements d'applications desktop
  - Toutes les applications sont compatibles et multi plateformes
  - Pas de problème pour les changements de version
  - Il est même possible d'utiliser la version suivante même si pas encore supportée
  - Ios, Android, Windows, micro contrôleurs
  - Cloud & Web
  - Un seul langage : Javascript
  - 2 ans de formation : un seul langage

# Introduction

- « Oui mais si on connaît un peu de Java, ou de C (ou de C#), alors Javascript ne pose pas de problème » (?!)
- Javascript est un langage orienté **fonctionnel**
  - Pas (ou peu) Java
  - Pas (ou peu) C
  - Formation fonctionnelle académique : Scheme (par exemple)

```
(define (factorial n)
  (let loop ((fact 1)
            (n n))
    (cond ((= n 0) fact)
          (else (loop (* n fact) (- n 1))))))
```

- Applications industrielles ?
- Java & C : procédural impératif
- Évènementiel

# Introduction

- Autre manière de penser et de développer
- Caractéristiques de Javascript (que n'ont pas Java, C etc..)
  - Les fonctions sont des objets (première classe)
    - On peut faire des variables qui référencent une fonction
    - On peut passer des fonctions comme paramètres d'autres fonctions
    - Une fonction peut renvoyer une autre fonction
  - Les clôtures de fonction et de contexte
    - Une fonction qui fait référence à une variable hors de son bloc, peut en maintenir l'état jusqu'à son utilisation
  - Javascript est un langage orienté objet mais
    - Pas orienté « classe » (comme java ou C++)
    - Orienté « prototypes » (comme self)
    - Programmer Javascript en utilisant les classes comme en java est un risque
      - Logique différente
      - Des objets de même classe peuvent avoir des attributs et des méthodes différentes (?!)

# Introduction

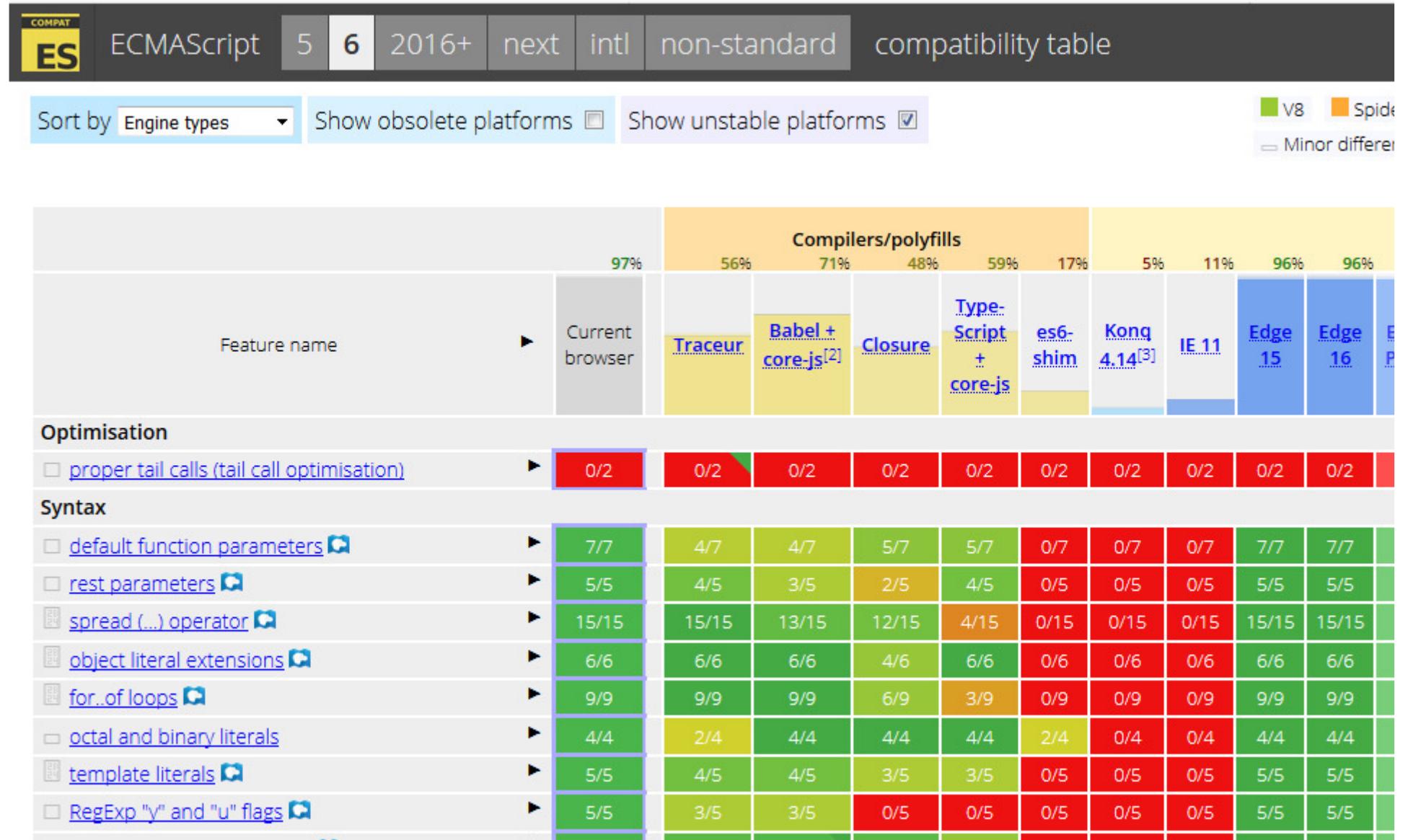
- Javascript permet de gérer l'asynchronisme
  - Générateurs
    - Des fonctions capables de générer des résultats successifs, de s'interrompre et de reprendre au besoin, leur évaluation
  - Promesses
    - Mécanisme qui permet d'utiliser des variables avant qu'elles n'aient de valeur définie. Contrairement à Java / C avec le passage par valeur (une valeur est évaluée avant de participer à un calcul)
  - Proxies, Map, Modules, RegExp etc..
- Erreur : Programmer Javascript comme un Java un peu différent syntaxiquement
- Javascript est **très** différent de Java

# Introduction

- Javascript : ECMAScript
- Comité de standardisation
- ECMA : European association for standardizing information and communication systems
- ES6, ES7 (maintenant ES8)
- Javascript est exécuté dans un interpréteur
  - Dans un navigateur Web
  - Ou bien Node.js (sorte de JVM pour Javascript)
- Compatibilités des navigateurs ?
- <https://kangax.github.io/compat-table/es6/>



# Introduction





# Introduction

# SpiderMonkey

V8

**JS**

Servers/runtimes											Mobile	
4%	66%	95%	59%	52%	97%	97%	2%	24%	Mobile		99%	99%
S	Echo JS	X6	JXA	Node 4 <sup>[5]</sup>	Node >=6.5 <7 <sup>[5]</sup>	Node >=8.7 <9 <sup>[5]</sup>	DUK 1.8	DUK 2.2	iOS >=10.3 <11	iOS 11	99%	99%
2	0/2	2/2	0/2	0/2	0/2	0/2	0/2	2/2	2/2	2/2	2/2	2/2
7	4/7	7/7	0/7	0/7	7/7	7/7	0/7	0/7	7/7	7/7	7/7	7/7
5	3/5	5/5	0/5	0/5	5/5	5/5	0/5	0/5	5/5	5/5	5/5	5/5
5	10/15	15/15	11/15	0/15	15/15	15/15	0/15	0/15	15/15	15/15	15/15	15/15
6	5/6	6/6	5/6	6/6	6/6	6/6	0/6	4/6	6/6	6/6	6/6	6/6
9	7/9	9/9	8/9	7/9	9/9	9/9	0/9	0/9	9/9	9/9	9/9	9/9
4	2/4	4/4	4/4	4/4	4/4	4/4	0/4	4/4	4/4	4/4	4/4	4/4
5	4/5	5/5	5/5	5/5	5/5	5/5	0/5	0/5	5/5	5/5	5/5	5/5
5	2/5	2/5	0/5	0/5	5/5	5/5	0/5	0/5	5/5	5/5	5/5	5/5
2	12/22	21/22	19/22	0/22	22/22	22/22	0/22	0/22	22/22	22/22	22/22	22/22
4	14/24	24/24	21/24	0/24	24/24	24/24	0/24	0/24	24/24	24/24	24/24	24/24
4	12/24	23/24	18/24	0/24	24/24	24/24	0/24	0/24	24/24	24/24	24/24	24/24
2	2/2	2/2	2/2	2/2	2/2	2/2	0/2	2/2	2/2	2/2	2/2	2/2
2	2/2	2/2	0/2	0/2	2/2	2/2	0/2	1/2	2/2	2/2	2/2	2/2
6	8/16	16/16	10/16	9/16	16/16	16/16	1/16	2/16	16/16	16/16	16/16	16/16

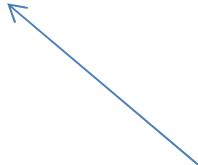
# Introduction

- Un des problèmes de approches classiques est la disponibilité / compatibilité des nouvelles versions du langage
- En Javascript, il est possible de travailler en version ES7 alors que seule la version ES6 est disponible
- Utilisation de **transpilateurs** :
  - Trans(late) + (com)piler = **transpiler**
  - Le code JS (ES7) est traduit à la volée en une version acceptable pour ES6
  - Pas de problème de compatibilité
  - <https://github.com/google/traceur-compiler>
  - <https://babeljs.io/>

# Exemple : Traceur

- Navigateur ECMAScript 5 (ECMA 6 Inconnu)
- Pas de classes
- Page Web :

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello World!</title>
  </head>
  <body>
    <h1 id="message"></h1>
  </body>
</html>
```



- On souhaite injecter un message là

# Exemple : Traceur

- Code ECMA 6 (impossible en ECM5)

```
class Salut {  
    constructor(message) {  
        this.message = message;  
    }  
  
    document.querySelector('#message')  
    bonjour() {  
        var element = document.getElementById('message');  
        element.innerHTML = this.message;  
    }  
};  
  
var salut= new Salut('Hello!');  
salut.bonjour();
```

# Exemple : Traceur

- Code à insérer :

```
<!DOCTYPE html>
<html>
...
<body>
<h1 id="message"></h1>

<script src="https://google.github.io/traceur-compiler/bin/traceur.js"></script>
<script src="https://google.github.io/traceur-compiler/bin/BrowserSystem.js"></script>
<script src="https://google.github.io/traceur-compiler/src/bootstrap.js"></script>

<script type="module" >
class Salut{
  constructor(message) {
    this.message = message;
  }

  bonjour() {
    var element = document.getElementById('message');
    element.innerHTML = this.message;
  }
};
```

# Exemple : Traceur

- Code à insérer :

```
var salut= new Salut('Hello!');  
salut.bonjour();  
</script>  
...  
</body>  
</html>
```

- Le code JS de traceur est déjà traduit en ECMA5
- Il contient le traducteur à la volée
- Node peut aussi le faire hors-ligne
- Interface Web

JS

# Exemple : Traceur

The screenshot shows a browser window with the URL [google.github.io/traceur-compiler/demo/repl.html#cls](http://google.github.io/traceur-compiler/demo/repl.html#cls). The title bar reads "Traceur Transcoding Demo". The left panel is labeled "Source" and contains the following JavaScript code:

```
1 class Salut {
2     constructor(message) {
3         this.message = message;
4     }
5
6     bonjour() {
7         var element = document.getElementById('message');
8         element.innerHTML = this.message;
9     }
10};
11
12 var salut= new Salut('Hello!');
13 salut.bonjour();
14|
```

The right panel shows the transcompiled JavaScript code:

```
1 $traceurRuntime.ModuleStore.getAnonymousModule(function() {
2     "use strict";
3     var Salut = function() {
4         function Salut(message) {
5             this.message = message;
6         }
7         return ($traceurRuntime.createClass)(Salut, {bonjour: function() {
8             var element = document.getElementById('message');
9             element.innerHTML = this.message;
10            }, {}});
11        }();
12        ;
13        var salut = new Salut('Hello!');
14        salut.bonjour();
15        return {};
16    });
17 //## sourceURL=traceured.js
18|
```

# Traceur Transcoding Demo

The screenshot shows a development environment for Traceur Transcoding. On the left, there's a code editor with some JavaScript code. In the center, there's a preview area showing the output of the transcoder. On the right, a context menu is open with various options for transcoding.

Code in the editor:

```
1 $traceurRuntime.ModuleStore
2   "use strict";
3   var Salut = function() {
4     function Salut(message)
5       this.message = message;
6     }
7     return ($traceurRuntime
8       var element = document.createElement('div');
9       element.innerHTML = this.message;
10      }, {});
11    }();
12    ;
13    var salut = new Salut('Hello');
14    salut.bonjour();
15    return {};
16  });
17 //# sourceURL=traceured.js
18
```

Options menu (context menu) items:

- Evaluate
- Show generated code
- Show all options
- arrowFunctions
- blockBinding
- classes
- computedPropertyNames
- defaultParameters
- destructuring
- forOf
- generators
- numericLiterals
- propertyMethods
- propertyNameShorthand
- restParameters
- spread
- symbols
- templateLiterals
- unicodeEscapeSequences
- unicodeExpressions
- properTailCalls

Idem pour babeljs

# Babel is a JavaScript compiler.

Use next generation JavaScript, today.

Put in next-gen JavaScript

```
[1, 2, 3].map(n => n ** 2);|
```

Get browser-compatible JavaScript out

```
[1, 2, 3].map(function (n) {  
  return Math.pow(n, 2);  
});
```

[Check out our REPL to experiment more with Babel!](#)

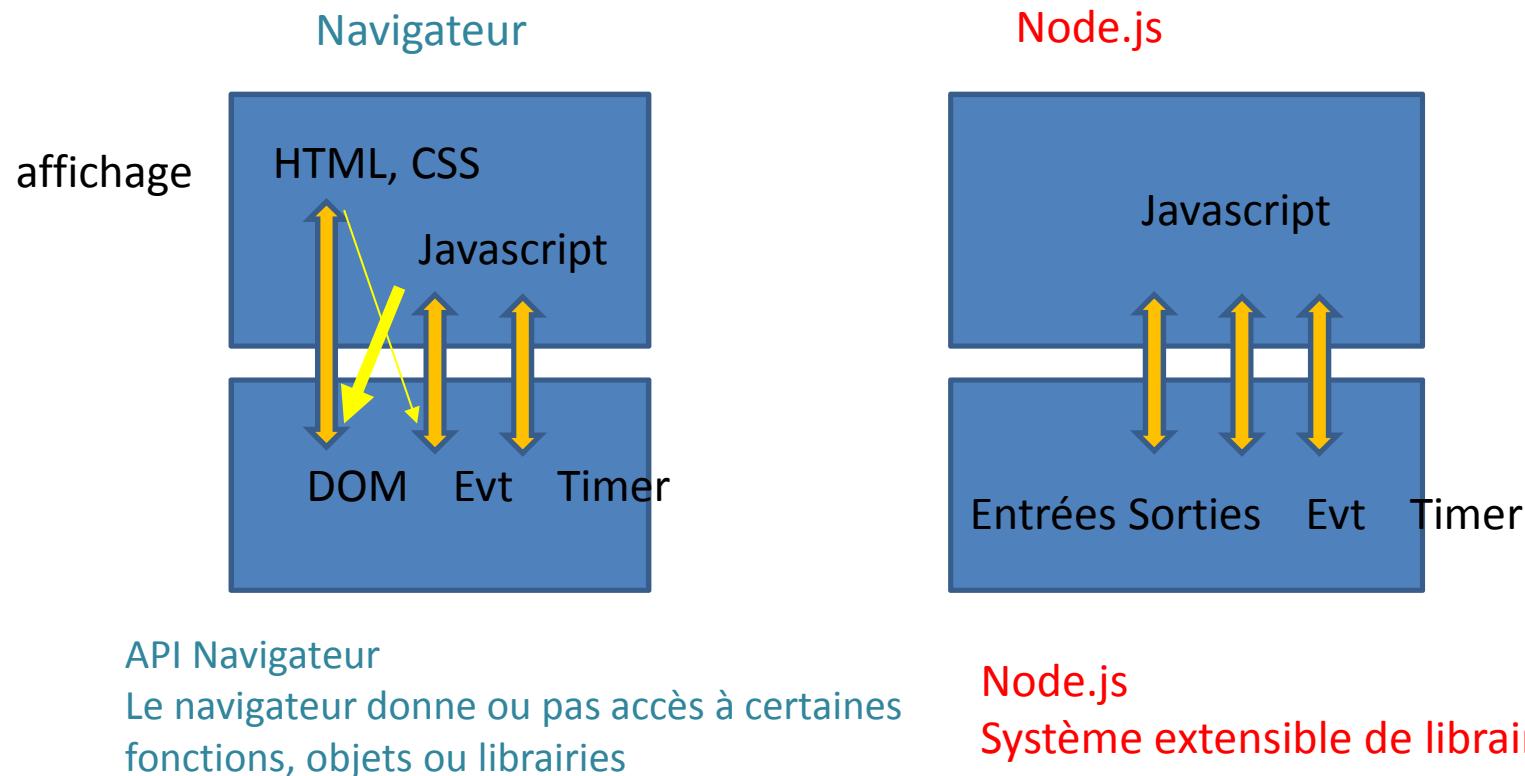
Latest From Our Blog: Nearing the 7.0 Release

# Introduction

- Deux environnements d'exécution pour Javascript
  - Le navigateur
    - L'interpréteur est intégré au navigateur (et donc figé)
    - Pas d'entrée/sortie fichier
    - Entrées sorties socket limitées
    - Performances dépendantes du navigateur
    - L'interface graphique dépend du navigateur
    - Généralement HTML, canvas, webgl ou librairies
  - Node.js
    - Pas de restriction
    - Mais pas (*a priori*) d'interface graphique (NW.js ou Electron)
    - Serveur

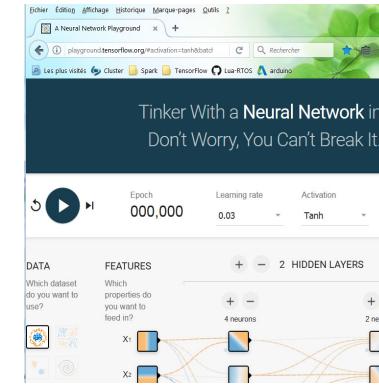
# Introduction

- Navigateur Web et Node.js



# Introduction

- DOM : Document Object Model
  - Hiérarchie d'objets qui représente le document HTML
  - Par exemple :



- *A priori*, pas de DOM côté Node.js
- Le javascript du navigateur modifie le DOM
- La modification du DOM change le document affiché
- L'affichage Node.js dépend des librairies disponibles
- Node.js ne s'exécute pas dans un navigateur
- Ligne de commande (comme la JVM)

```
C:\Users\menier>node programme.js
```

# Introduction

- Les évènements
  - Déclenchent des fonctions javascript
  - Ils peuvent survenir n'importe quand
  - Leur nature dépend de l'environnement d'exécution

Navigateur

Click souris

Changement d'un texte dans un formulaire

Réception d'information par le Web etc..

Node.js

Entrées Sorties

Communication avec Internet

En fonction des librairies (clavier etc)

- *Asynchrone* : le code Javascript s'exécute sans prévoir de tester explicitement ces événements
- Dès qu'un événement a lieu, une fonction spécifique interrompt le code en cours d'exécution
- Cette fonction gère l'évènement et le réponse du code Javascript
- Puis le code reprend

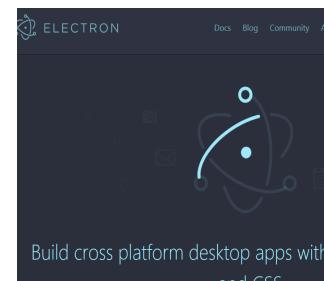
# Introduction

- Navigateur + node.js

- Il est possible d'associer node.js à un navigateur et à générer un code exécutable qui contient le meilleur des deux.
- Dans ce cas, on peut réaliser des applications ‘bureau’ ou desktop dans lesquelles les interfaces graphiques sont réalisées en HTML / CSS (même si on ne voit ni le navigateur, ni le code). Multi plateforme.



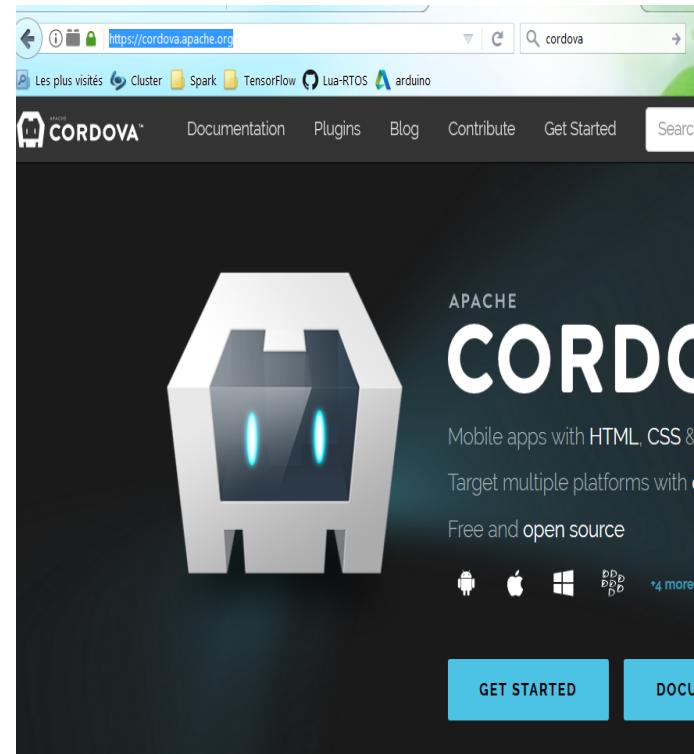
- Nws : <https://nwjs.io/>



- Electron : <https://electron.atom.io/>

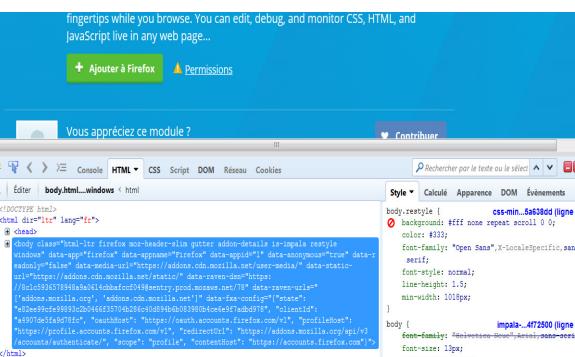
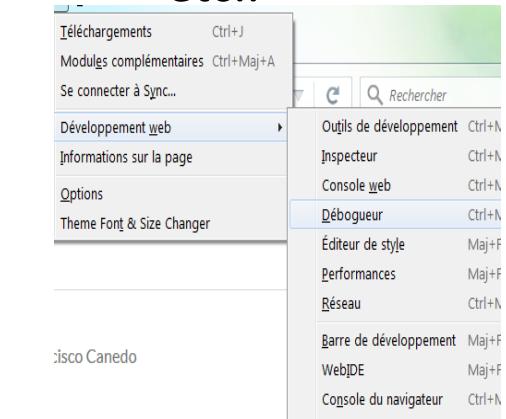
# Introduction

- Développement mobile
- <https://cordova.apache.org/>
- Cloud
  - Google cloud
  - Cloud9
  - Io.js
  - Etc..



# Introduction

- Outils de développement
  - Navigateur :  
etc..



Node.js : eclipse, IDE

A screenshot of the IntelliJ IDEA IDE. The interface shows a 'teensy' project structure with files like 'teensy.scala', 'GUI.scala', and 'Hc5.scala'. The code editor is open to 'PoolPoints.scala' with the following content:

```

var lw: Double = 0.0
def send(pcode: String, pca: String) {
  var code : String = "o"
  var canal : Int = 0
  var x: Double = 0.0
  var y : Double = 0.0
  var z: Double = 0.0
  var w : Double = 0.0
  try {
    ...
  }
}
  
```

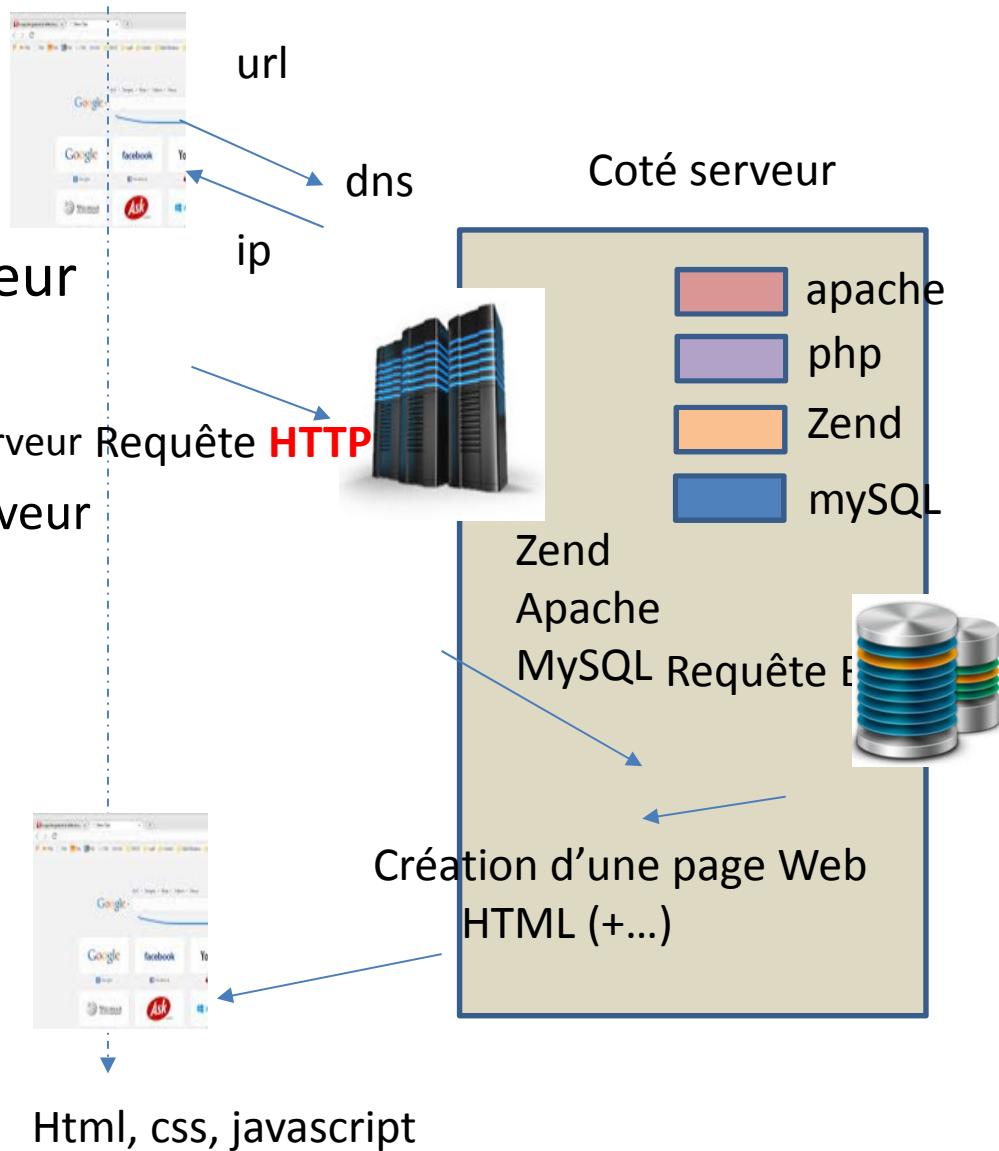
The code editor has syntax highlighting for Scala. The background shows a dark theme with some UI elements.

# Introduction

- Quelques outils de test
- E/S console
  - `console.log("hello")`
- `assert`
  - `assert(condition, message)`
  - `assert( a != 0, " attention, a == 0, division par zéro")`
  - Se déclenche si la condition **n'est pas vérifiée**
- performance (temps)
  - `console.time("temps pour la boucle")`
  - `for(var n=0; n<10000; n++) ....`
  - `console.timeEnd("temps pour la boucle")`
  - Affiche la durée sur la console

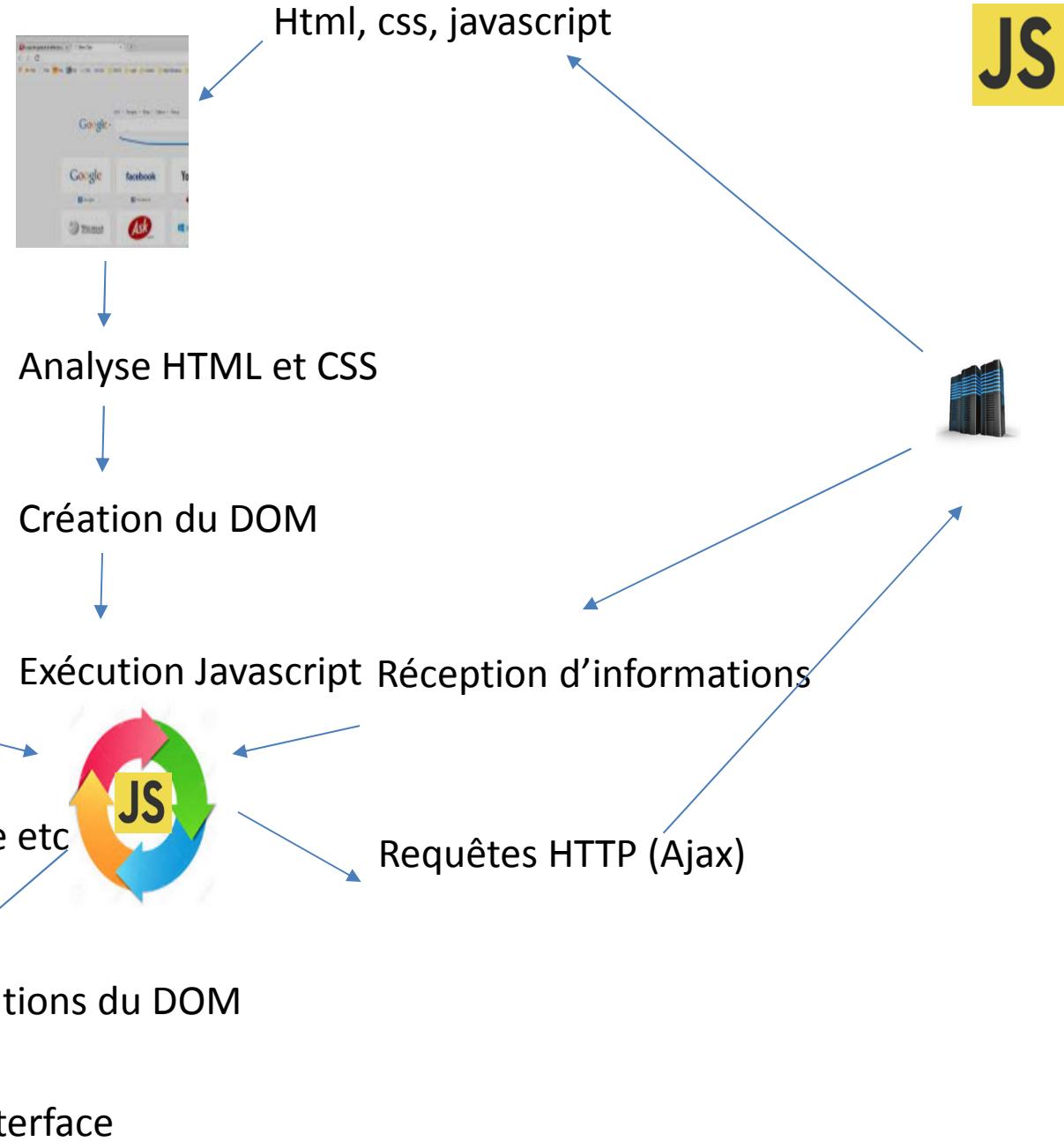
# Coté serveur

- PHP classique
- Traitements coté serveur
- Pages statiques
  - Les transformations coté serveur Requête HTTP
- Plusieurs technos coté serveur
  - Apache
  - PHP
  - MySQL
  - Etc..
- Administrations
  - Apache
  - PHP
  - PHPMyADMIN
  - Etc..



# Coté client

- HTML
  - + Javascript



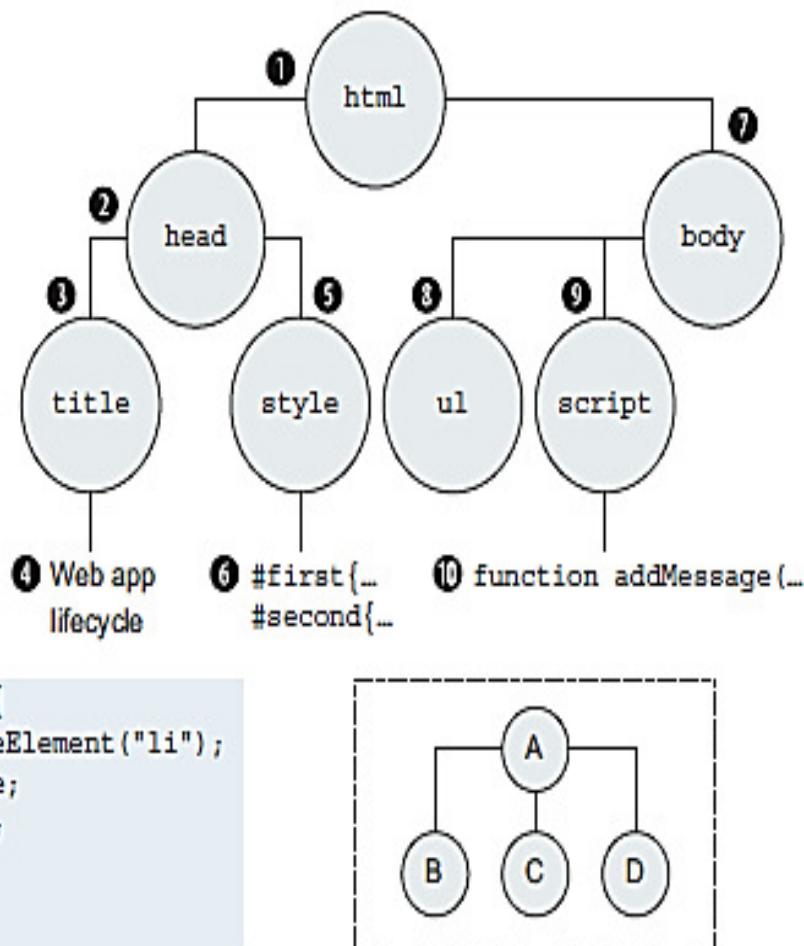
# DOM : Document Object Model

- Représentation objet du code HTML

```

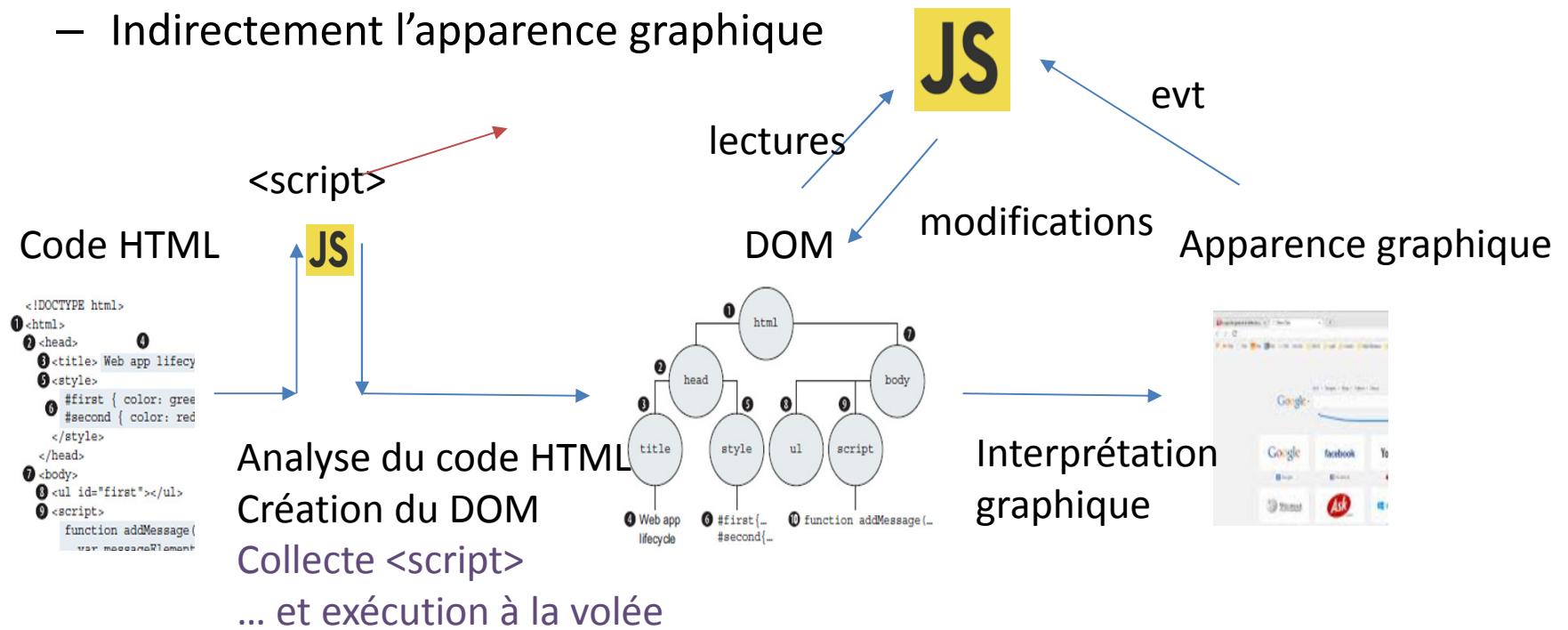
<!DOCTYPE html>
①<html>
②<head>          ④
    ③<title> Web app lifecycle </title>
    ⑤<style>
        ⑥ #first { color: green; }
        ⑦ #second { color: red; }
    </style>
</head>
⑦<body>
⑧<ul id="first"></ul>
⑨<script>
    function addMessage(element, message){
        var messageElement = document.createElement("li");
        messageElement.textContent = message;
        element.appendChild(messageElement);
    }
    ...

```



# DOM : Document Object Model

- Le DOM est stocké dans le navigateur
- L'affichage correspond à l'état du DOM
  - PAS A LA PAGE HTML
- Javascript modifie le DOM
  - Indirectement l'apparence graphique



# Construction du DOM

- Quand une balise <script> est rencontrée, la construction du DOM est suspendue
- Le code est exécuté (le code peut modifier le DOM)
- La construction du DOM reprend
- Techniquement, il est possible de modifier par programme Javascript la construction du DOM
  - Angular.js
- Attention à l'ordre code et des balises <script>
  - Dépend du navigateur
  - Effets de bord
- Les variables et fonctions déclarées sont accessibles globalement
- Objet commun ‘window’ qui contient l’état javascript

JS



Javascript

Eléments de base du langage



# Généralités

- Séparateur ;

- a = 5

- a = 5 ; b = 12



- a = 5 ; b = 12 ; c = 18

- a = 5 ; b = 12 ; c = 18 ;

- attention

return  
true ;

return ;  
true ;

return true ;



# Variables

- Définition (*locale* ?)

```
var maVariable ;
```

```
var maVariable = "texte" ;
```

```
// locale au bloc
```

```
{  
    let i = 5;  
}
```



# Types de base

- **String**

```
var nom= 'john' ;  
var nom= "john" ;  
var nom='john' + ' ' + 'Doe';
```

- **fonctions**

charAt(n), charCodeAt(n), concat(s1,s2,..),  
fromCharCode(), indexOf(S), lastIndexOf(S),  
match(regexp),  
replace(regexp, St), search(regexp), slice(start,end),  
split(separat,maxsplit), substr(start,lg),  
substring(de,a),  
toLowerCase(),toUpperCase(),valueOf()



# Types de base

- ECMA 6 : *Template litteral*

```
var nom = "John"  
var prenom = "Doe"
```

```
var bonjour = `Bonjour ${nom} ${prenom}!`  
// back-tick !!! Attention
```

// plusieurs lignes possibles

```
Var bonjour = `Bonjour  
Ca va ?  
Oui ?`
```



# Types de base

## ◦ String pratique

```
var heure = "12:30:32"  
var h_m_s = heure.replace(":", "_")
```

attention

```
var heure = "12:30:32"  
var hms = heure.split(":")  
for(i in hms) console.log(hms[i])
```

console.log  
(console browser ou node.js)

```
var heure = "12:30:32"  
if (heure.indexOf("12") != -1) console.log("midi");
```



# Types de base

- Number

```
var numero = 1 ;
```

```
var numero = 1.333 ;
```

```
// un seul type : number
```



# Types de base

- Boolean

```
var vrai = true ;
```

&&  
||  
!

```
var false = false ;
```



# Types de base

- **Array**

- Indices à partir de 0

```
var nb = ["zero", "un", "deux", "trois"; 4, 5, true ] ;
```

```
var debut = nb[0] ;
```



# Types de base

## ◦Objet

- var o = {}
  - Objet vide
- o.nom = "luc"
  - Rajout de propriété



# Types de base

## ◦Object.property

... propriété objet

```
var objet = { "nom":"Doe", "prenom":"john", "age":35 } ;
```

```
var leNom = objet.nom ;      ... hashtable
```

```
var leNom = objet["nom"];
```

## ◦null

```
var objet = null ;
```

Mieux que 0 ou "" : null indique *pas de valeur*

*Ce n'est pas un pointeur !!*

# Langage faiblement typé



- Attention

- changement de type

```
var a = "test" ;  
a = 5 ;
```

pb des langages faiblement typés : pas de contrainte, ni de vérification sur les types des valeurs renvoyés  
*(ici String) – JSX / TypeScript*

- sécurité (effet de bord)

```
var t = "test" ;  
var a = 5 ;  
var r = a + t ;  
var r = t + a ;
```

*Conversion implicite*

"5test"  
"test5"

# Langage faiblement typé



- Attention

- sécurité (effet de bord)

```
var a = 5 ;
```

```
var b = "3" ;
```

```
var r = a - b ; .... 2
```

```
var r = b - a ; .... -2     Attention !!!! +
```

# Comparaisons



## • Opérateurs booléens

```
var x = 10 ;
```

`==`      égalité (valeur)

<code>x == 8</code>	<code>false</code>
<code>x == 10</code>	<code>true</code>
<code>x == "10"</code>	<code>true</code>

`====`    égalité valeur type

<code>x === 10</code>	<code>true</code>
<code>x === "10"</code>	<code>false</code>

`!==`    type et valeur <>

<code>x !==="10"</code>	<code>true</code>
<code>x !== 10</code>	<code>false</code>

# Comparaisons



- **if**

```
if (x==5) {  
    ...  
}
```

```
if (x==5) {  
    ...  
} else {  
    ...  
}
```

```
if (x==5) {  
    ...  
} else if (x==6) {  
    ...  
} else if (x==7) {  
    ...  
} else {  
    ...  
}
```



# Comparaisons

- **switch**

```
switch (x) {  
    case 5 : ...  
    break ;  
  
    case 6 : ...  
    break ;  
  
    case 7 : ...  
    break ;  
  
    default : ...  
}
```



# Itérations

- `while( condition ) { ... }`
- `do { ... } while ( condition );`
- `for ( ;; ) { ... }`
- **break** : sortie du bloc
- **continue** : saute prochaine itération

déconseillés  
(à part *switch*)



# Itérations

- **for / in**

- **tableaux**

```
var jours = ["lundi","mardi","mercredi","jeudi","vendredi"];  
  
for( var jour in jours ) {  
    console.log( jours[jour] + "<br>" );  
}
```

- **objets**

```
var objet = { "nom":"Doe", "prenom":"john", "age":35 } ;
```

```
for( var prop in objet) {  
    console.log(prop) ;  
    // objet[prop]  
}
```

nom

prenom

age



# Fonctions

- Définition de fonction

- classique

```
function maFonction() {  
    console.log("Ca marche")  
}
```

- appel

```
maFonction();  
// appel immédiat
```



# Fonctions

- Paramètres

- ° sans type

```
function salutation(prenom, nom) {  
    console.log("Bonjour "+ prenom + " " + nom)  
}
```

- ° retours

```
function salutation(prenom, nom) {  
    var res = "Bonjour "+ prenom + " " + nom  
    return res ;  
    //  
}  
console.log( salutation("phil","collins") )
```

return termine la fct



# Fonctions

- Paramètres (es6)

- Par défaut

```
function salutation(prenom = "Jean", nom = "Dupond") {  
    console.log("Bonjour " + prenom + " " + nom)  
}
```

- Appels

```
salutation()  
salutation("bob")  
salutation("jerôme","Durand")
```

Remarque : attention, une grosse majorité des prédefinis JS ont des paramètres par défaut  
Cf callbacks etc..



# Fonctions

- **var fonction**

```
function plus(a,b) {  
    return a+b  
}
```

f est une variable de type *function*

```
var f = plus
```

```
var res = f(1,2)
```

```
console.log(res)
```



# Fonctions

- paramètre fonction

```
function plus(a,b) {  
    return a+b  
}
```

on peut passer des fonctions  
en paramètre

```
function appliquer(f, a, b) {  
    return f(a,b)  
}
```

```
console.log ( appliquer(plus, 1, 2) )
```



# Fonctions

- définition de fonction  
(anonyme)

```
var plus = function (a,b) {  
    return a+b  
}  
  
fonction anonyme
```

```
function appliquer(f, a, b) {  
    return f(a,b)  
}
```

```
console.log ( appliquer(plus, 1, 2) )
```



# Fonctions

- définition de fonction  
(anonyme)

```
function appliquer(f, a, b) {  
    return f(a,b)  
}
```

```
console.log ( appliquer( function(a,b) { return a+b }, 1, 2 ) )
```

très important : les fonctions sont (presque) des données comme les autres : on peut les passer en paramètre, les affecter à des variables etc..

*langage fonctionnel !*



# Fonctions

- Es6

```
var f = (a,b) => a+b
```

Cf Scala

**Une fonction est un objet comme les autres objets**

```
var somme = new Function('a', 'b', 'return a + b');  
console.log( somme(4,5))
```

// permet de construire des fonctions de manière dynamique  
// en cours de programme  
// Srialisation ?



# Types de base

## ◦Objet

- `var o = {}`
  - Objet vide
- `o.nom = "luc"`
  - Rajout de propriété
- `somme.comment = "trop Stylée"`
- `f.commentQuelleEst = "trop cheatée de ouf"`



```
> function compte() { }
undefined
> compte()
undefined
>
```



```
> function compte() {  
...   return 1  
... }  
undefined  
> compte()  
1  
>
```



```
> var a = 0
undefined
> function compte() {
...   a = a+1
...   return a
...
}
undefined
> compte()
1
> compte()
2
>
```



```
function compte() {  
    var a = 0  
  
    function compteTemp() {  
        a = a +1  
        return a  
    }  
  
    return compteTemp()  
}
```



```
> function compte() {
...   compte.a = compte.a+1
...   return compte.a
...
undefined
> compte.a = 0
0
> compte()
1
> compte()
2
>
```



```
> function compte() {
... if (typeof compte.a == 'undefined') compte.a = 0
... compte.a = compte.a+1
... return compte.a
...
undefined
> compte()
1
> compte()
2
> compte()
3
>
```

If ( compte.a === undefined )....



# Types de base

```
function estPremier(valeur) {  
  
    var premier = (valeur !== 1);  
    for (var i = 2; (i < valeur) && (premier) ; i++) {  
        if (valeur % i === 0) {  
            premier = false;  
        }  
    }  
    return premier;  
}
```



# Types de base

```
function estPremier(valeur) {  
  
    var premier = (valeur !== 1);  
    for (var i = 2; i < valeur; i++) {  
        if (valeur % i === 0) {  
            premier = false;  
            break;  
        }  
    }  
    return premier;  
}
```



# Types de base

tableau

```
function estPremier(valeur) {
```

valeur ? -> return vrai/faux sinon...

```
    var premier = (valeur !== 1);
    for (var i = 2; i < valeur; i++) {
        if (valeur % i === 0) {
            premier = false; <-- valeur ?
            break;
        }
    }
    return premier;
```

}

Solution ?



# Types de base

```
function estPremier(valeur) {
```

Tableau / hashtable

```
function estPremierUtil(valeur) {
```

valeur ? -> return vrai/faux sinon...

```
    var premier = (valeur !== 1);
        for (var i = 2; i < valeur; i++) {
            if (valeur % i === 0) {
                premier = false; <- valeur ?
                    break;
            }
        }
        return premier;

    }

return estPremierUtil(valeur)
}
```

Solution ? C ?



# Types de base

## Fonction avec un cache intégré

```
function estPremier(valeur) {  
    if (!estPremier.reponses){  
        estPremier.reponses = {};  
    }  
    if (estPremier.reponses[valeur] !== undefined) {  
        return estPremier.responses[valeur];  
    }  
    var premier = (valeur !== 1);  
    for (var i = 2; i < valeur; i++) {  
        if (valeur % i === 0) {  
            premier = false;  
            break;  
        }  
    }  
    return estPremier.reponses[valeur] = premier;  
}
```



# Fonctions

## • Langage objet et à *prototypes*

```
var obj = new Object() // var obj = {}
```

```
console.log(obj)  
// {}
```

```
obj.a = 45  
console.log(obj)  
// { a : 45 }
```

*création des attributs  
ou propriétés au fur et à mesure  
pas de définition de classe*

```
obj.b = "coucou"  
console.log(obj)  
// { a : 45, b : 'coucou' }
```



# Fonctions

## ... •Langage objet et à prototype

```
obj.f = function plus(x,y) { return x+y }
```

```
console.log(obj)  
// { a : 45, b : 'coucou', f : [Function : plus] }
```

```
console.log( obj.f(2,3) )  
// 5
```

*Création de méthodes*



# Fonctions

## • Accès

```
console.log(obj.a)  
// 45
```

*Double syntax :*  
*objet : obj.a*  
*hashtable : obj["a"]*

```
console.log(obj["a"])  
// 45
```

```
for(var i in obj) console.log( obj[i] )  
// 45  
// coucou  
// [Function]
```

*itérateur*



# Fonctions

Java : <http://wiki.fasterxml.com/JacksonHome>

## • JSON : JavaScript Object Notation

```
var obj = {  
    a : 45,  
    b : "coucou",  
    f : function(x,y) { return x+y } // ou bien f : (x,y) => x+y  
}
```

*Création d'un objet, attributs et méthodes*

```
var sObj = JSON.stringify( obj )  
// '{ a:45, b :"coucou"}'
```

*pas de méthodes en JSON  
(mais voir vm. node)*

```
var nObj = JSON.parse( sObj )  
// création de l'objet à partir de la chaîne de caractères
```



# Fonctions

- JSON : **JavaScript Object Notation**
  - Sérialisation (**stringify**)
  - Désérialisation (**parse**)
  - Transfert de données structurées
  - Plus léger que XML
  - Rapide à analyser
  - Traitement Javascript (objet)



# Fonctions

## • JSON : Gestion des méthodes

```
var obj = {  
  f: function() { // ou bien f : () => "hello"  
    return "Hello";  
  }  
};
```

*Fonction de transformation  
de valeur*



```
var sObj = JSON.stringify(obj, function(key, value) {  
  if (typeof value === 'function') {  
    return value.toString();  
  } else {  
    return value;  
  }  
});
```

*possibilité de tester le type  
d'un objet*

```
console.log(sObj);  
// {"f":"function () { return \"Hello\" }"}
```



# Fonctions

- JSON : Gestion des méthodes
  - eval
  - lent
  - portée
  - uneval ( non std : à éviter)
- node.js
  - librairie vm
  - Stable

- `new vm.Script(code, options)`
- `script.runInContext(contextifiedSandbox[, options])`
- `script.runInNewContext([sandbox[, options]])`
- `script.runInThisContext([options])`
- `vm.createContext([sandbox[, options]])`
- `vm.isContext(sandbox)`
- `vm.runInContext(code, contextifiedSandbox[, options])`
- `vm.runInDebugContext(code)` **deprecated**
- `vm.runInNewContext(code[, sandbox][, options])`
- `vm.runInThisContext(code[, options])`
- Example: Running an HTTP Server within a VM
- What does it mean to "contextify" an object?

Node.js

## VM (Executing JavaScript)

Stability: 2 - Stable

The `vm` module provides APIs for compiling and running code within V8 Virtual Machine contexts.

JavaScript code can be compiled and run immediately or compiled, saved, and run later.

A common use case is to run the code in a sandboxed environment. The sandboxed code uses a different V8 Context, meaning that it has a different global object than the rest of the code.

One can provide the context by "contextifying" a sandbox object. The sandboxed code treats any property on the `s` like a global variable. Any changes on global variables caused by the sandboxed code are reflected in the sandbox object.



# Fonctions

- Fonction de création d'objet

```
var obj = new Object()
```

```
var nouvelObjet = function () { return {} }  
// var nouvelObjet = () => {}
```

```
var t = new nouvelObjet()  
t.a = 45
```

```
console.log(t)  
// { a : 45 }
```

```
// la fonction nouvelObjet est un constructeur d'objet
```



# Fonctions

- Fonction de création d'objet

```
var Personne = function (prenom, quelNom, age) {  
    this.prenom = prenom  
    this.nom = quelNom  
    this.age = age  
}
```

```
var bob = new Personne("bob","dupond",34)
```

```
console.log(bob)  
// { prenom :'bob', nom :'dupond', age : 34 }
```



# Fonctions

• Donc...

```
function Personne(prenom, quelNom, age) {  
    this.prenom = prenom  
    this.nom = quelNom  
    this.age = age  
}
```

```
var bob = new Personne("bob", "dupond", 34)
```

```
var bob = { prenom : 'bob', nom : 'dupond', age:34  
 }
```

```
var Personne = function (prenom, quelNom, age) {  
    this.prenom = prenom  
    this.nom = quelNom  
    this.age = age  
}
```

```
var bob = new Object()  
bob.prenom = "bob"  
bob.nom= "dupond"  
bob.age = 34
```

```
var bob = new Personne("bob", "dupond", 34)
```



# Fonctions

## • Fonction de création d'objet

```
function Personne(prenom, quelNom, age) {  
    this.prenom = prenom  
    this.nom = quelNom  
    this.age = age  
    // méthodes ?  
}
```

```
var bob = new Personne("bob","dupond",34)  
  
console.log(bob)  
// { prenom :'bob', nom :'dupond', age : 34 }
```



# Fonctions

## • Constructeur

```
function Personne(prenom, quelNom, anneeN) {  
    this.prenom = prenom  
    this.nom = quelNom  
    this.anneeN = anneeN  
    this.age = function(anneeA) { return anneeA-anneeN }  
}
```

```
var bob = new Personne("bob","dupond",1980)  
  
console.log(bob.age(2015))  
// 35
```

*classe ?  
pas d'héritage ?  
(si... voir plus loin)*



# Fonctions

## ● Constructeur

```
function Personne(prenom, quelNom, anneeN) {  
    this.prenom = prenom  
    this.nom = quelNom  
    this.anneeN = anneeN  
    this.age = function(anneeA) { return anneeA-this.anneeN }  
}
```

```
var bob = new Personne("bob","dupond",1980)
```

```
console.log(bob.constructor)  
// [Function : Personne]
```



# Fonctions

- Constructeur et prototypes ?

```
var a = new bob.constructor("luc","courtrai",1934)
```



# Fonctions

- DeepClone

```
var objetCopie = JSON.parse(JSON.stringify(objetDeBase));  
// attention aux méthodes
```

```
var objetCopie = jQuery.extend(true, {}, objectDeBase)
```

```
// attention assign et create -> shallowCopy
```



# Fonctions

// pour les méthodes, utiliser

```
var sObj = JSON.stringify(obj, function(key, value) {  
    if (typeof value === 'function') {  
        return value.toString();  
    } else {  
        return value;  
    }  
});
```



# Fonctions

## • Prototype et classe

- C++, JAVA etc : langages objets à classes
- Une classe = ensemble de propriétés/méthodes
- Une instance est une réalisation de classe
- Une classe n'est pas un objet
- C'est le plan d'un objet
- Une instance est la réalisation du plan



# Fonctions

## • Javascript : Prototype

- LA NOTION DE CLASSE **N'EXISTE PAS !**
- Un objet peut se construire lui-même en rajoutant ses propriétés et ses méthodes
  - À la création (code)
  - À l'exécution (de manière dynamique)
- Un constructeur Javascript fabrique un objet en lui rajoutant des méthodes et des propriétés
  - C'est vraiment un constructeur
  - Il n'y a pas de différence entre la définition de la classe et le constructeur
- Un constructeur Java fabrique un objet comme une réalisation du plan + une initialisation
  - C'est plus un initialiseur qui complète la réalisation de la classe



# Fonctions

## • Javascript : *Prototype*

- Un objet peut servir de modèle pour la création d'un autre objet
- Il sert de modèle
- Il sert de *prototype*
- *Pas besoin de classe*
- *Toute fonction JS peut être utilisée comme un constructeur*
- *new permet en plus de fabriquer un objet*



# Fonctions

## ● Constructeur

```
function Personne(prenom, quelNom, age) {  
    this.prenom = prenom  
    this.nom = quelNom  
    this.anneeN = anneeN  
    this.age = function(anneeA) { return anneeA-this.anneeN }  
}
```

```
var bob = new Personne("bob","dupond",1980)
```

```
console.log(bob.constructor)  
// [Function : Personne]
```



# Fonctions

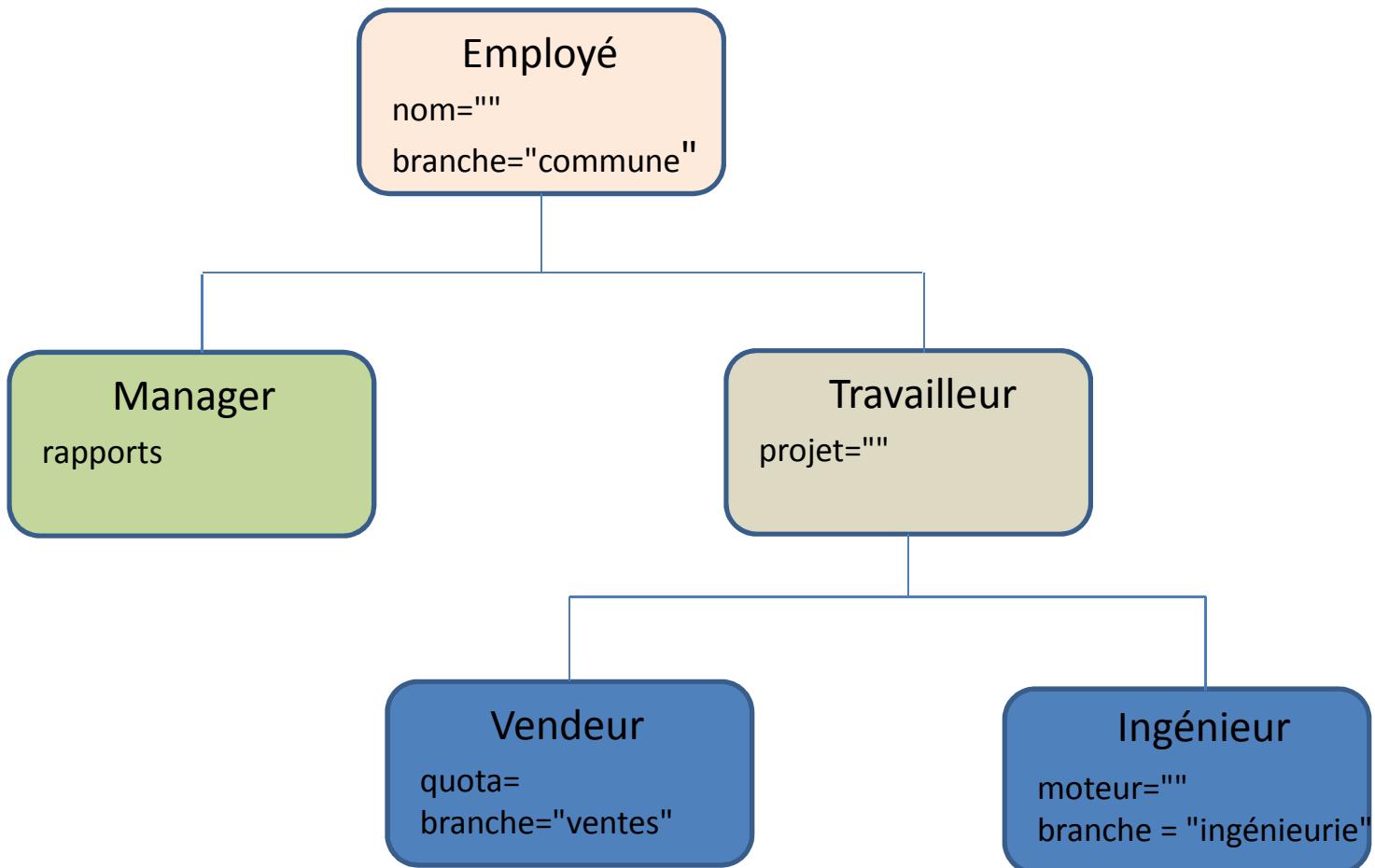
- **Langage à classes**

- Propriétés créées à la compilation
- Figées

- **Langage à Prototypes**

- Possibilité de rajouter / enlever des propriétés
- Compilation et exécution
- Si un objetX est utilisé comme prototype pour créer un autre objet
  - Rajouter une propriété ou une méthode à l'objetX rajoute automatiquement cette propriété à tous les objets de ce prototype

# Prototype et héritage



# Prototype et héritage



## • Prototype & classe

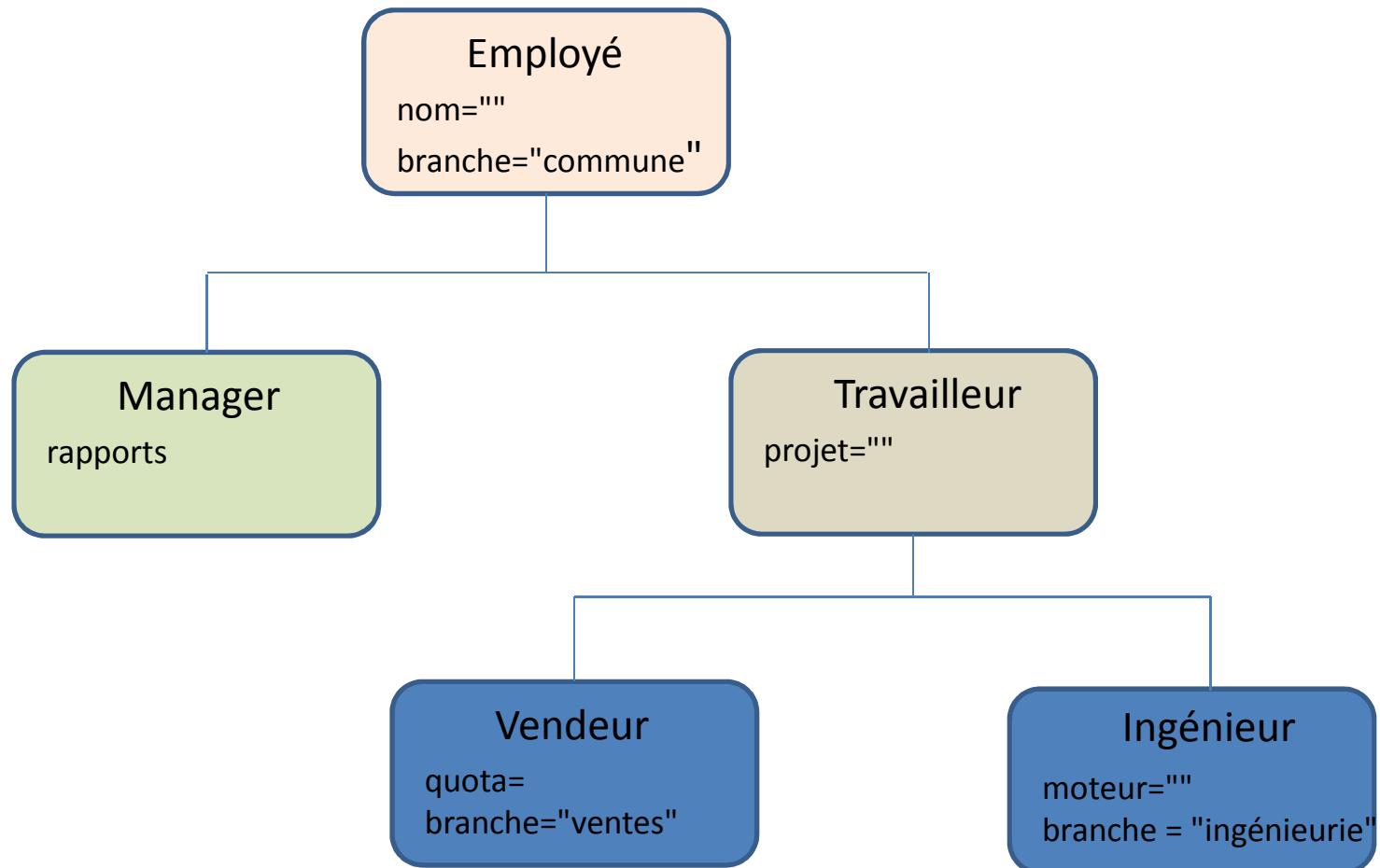
Javascript

```
function Employé () {  
    this.nom = "";  
    this.branche = "commun";  
}
```

Java

```
public class Employé {  
    public String nom;  
    public String branche;  
    public Employé () {  
        this.nom = "";  
        this.branche = "commun";  
    }  
}
```

# Prototype et héritage



# Prototype et héritage



## • Prototype & classe

Javascript

```
function Manager () {
    this.rapports = [];
}
Manager.prototype = new Employé;

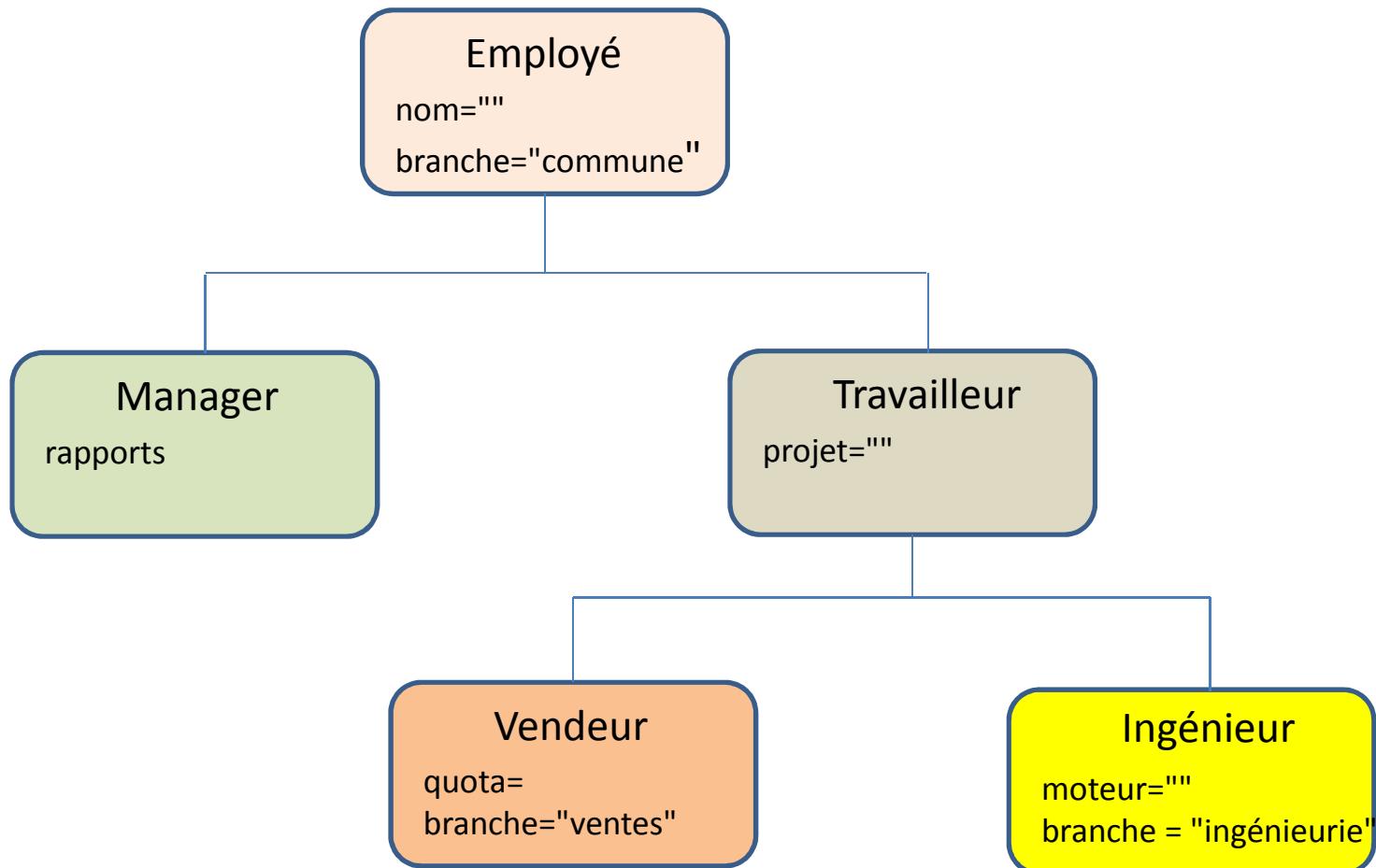
function Travailleur () {
    this.projets = [];
}
Travailleur.prototype = new Employé;
```

Java

```
public class Manager extends Employé {
    public Employé[] rapports;
    public Manager () {
        this.rapports = new Employé[0];
    }
}

public class Travailleur extends Employé {
    public String[] projets;
    public Travailleur () {
        this.projets = new String[0];
    }
}
```

# Prototype et héritage



# Prototype et héritage



## • Prototype & classe

Javascript

```
function Vendeur () {
    this.branche = "ventes";
    this.quota = 100;
}
Vendeur.prototype = new Travailleur;

function Ingénieur () {
    this.branche = "ingénierie";
    this.moteur = "";
}
Ingénieur.prototype = new Travailleur;
```

Java

```
public class Vendeur extends Travailleur {
    public double quota;
    public Vendeur () {
        this.branche = "ventes";
        this.quota = 100.0;
    }
}

public class Ingénieur extends Travailleur {
    public Ingénieur () {
        this.branche = "ingénierie";
        this.moteur = "";
    }
}
```



# Fonctions

## ● Prototype

```
function Personne(prenom, quelNom, age) {  
    this.prenom = prenom  
    this.nom = quelNom  
    this.anneeN = anneeN  
    this.age = function(anneeA) { return anneeA-anneeN }  
}  
  
bob = new Personne("bob", "Geldorf", 23)
```

```
Personne.prototype = {  
    donneNomComplet : function() {  
        return this.prenom+" "+this.nom  
    }  
}
```

```
console.log(bob.donneNomComplet())
```

# Prototype et héritage



## • Propriété des constructeurs

```
function DocumentImprimable (unDocument) {  
    this.document = unDocument;  
}
```

```
DocumentImprimable.prototype.print = function () {  
    console.log("impression de " + this.document);  
}
```

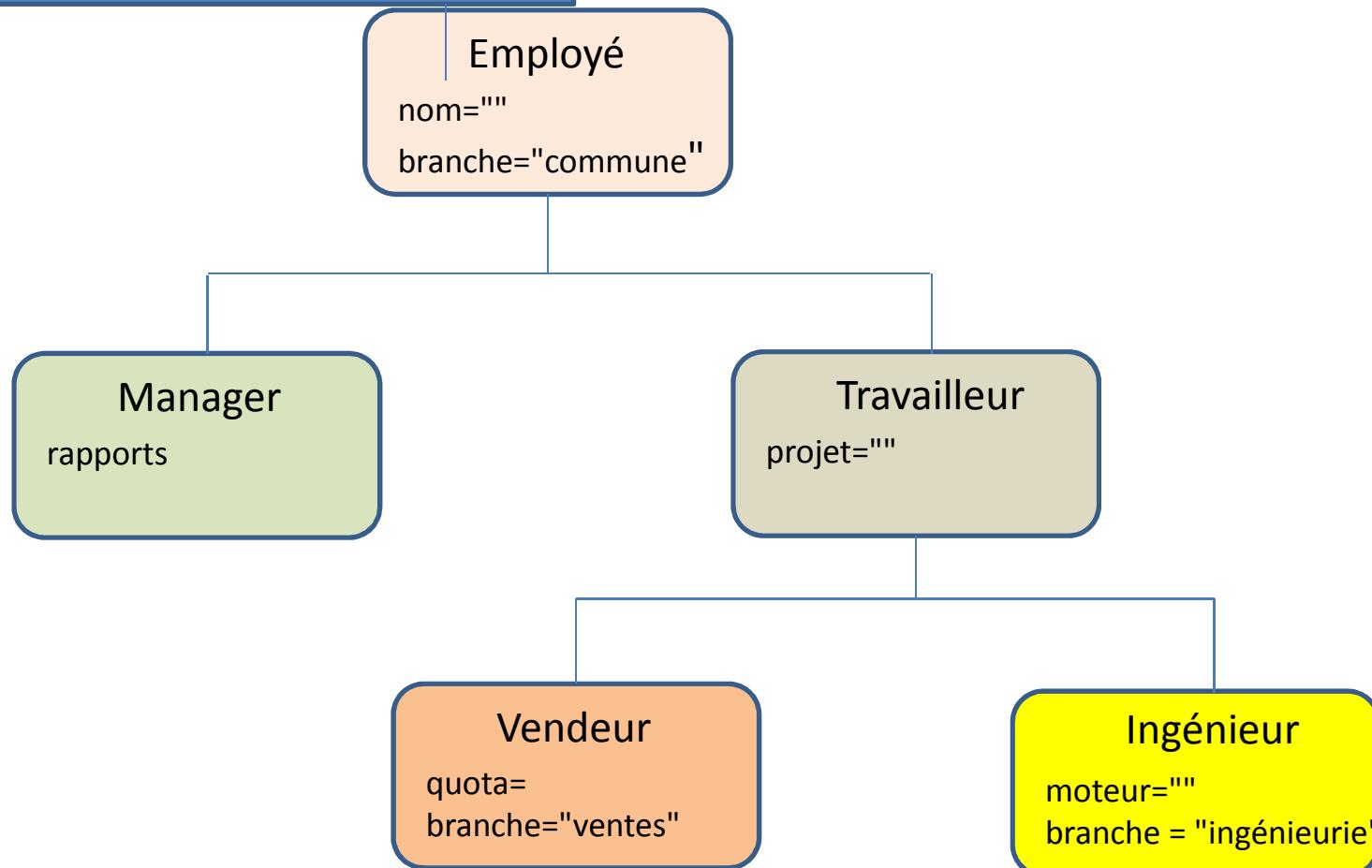
```
var doc = new DocumentImprimable ("un document particulier")
```

```
doc.print (); // doc a hérité de la méthode print
```

# Prototype et héritage



```
Employé.prototype.spécialité = "aucune";
```



# Prototype et héritage



```
function afficherProps(obj, nomObj) {  
    var result = "";  
    for (var i in obj) {  
        result += nomObj + "." + i + " = " + obj[i] + "\n";  
    }  
    result += "\n";  
    return result;  
}  
  
var v = new voiture("206", "peugeot")  
afficherProps(v, "voiture")  
  
// affichage :  
voiture.nom = 206  
voiture.fabricant = peugeot  
  
function voiture(nom_fabricant_) {  
    this.nom = nom_  
    this.fabricant = fabricant_  
}
```

# Prototype et héritage



```
function afficherProps(obj, nomObj) {  
    var result = "";  
    for (var i in obj) {  
        result += nomObj + "." + i + " = " + obj[i] + "\n";  
    }  
    result += "\n";  
    return result;  
}  
  
var v = new voiture("206", "peugeot")  
  
afficherProps(v, "voiture")  
  
// affichage :  
voiture.nom = 206  
voiture.fabricant = peugeot
```

# Prototype et héritage



```
let arr = [3, 5, 7];
arr.toto = "coucou";
```

→ propriétés

```
for (let i in arr) {
  console.log(i); // affiche 0, 1, 2, "toto" dans la console
}
```

→ valeurs

```
for (let i of arr) {
  console.log(i); // affiche 3, 5, 7 dans la console
}
```

# Prototype et héritage



## Itérateurs

```
function makeliterator(tableau){  
    var nextIndex = 0;  
  
    return {  
        next: function(){  
            return nextIndex < tableau.length ?  
                {value: tableau[nextIndex++], done: false} :  
                {done: true};  
        }  
    }  
}  
  
var it = makeliterator(['am', 'stram','gram']);  
console.log(it.next().value); // 'am'  
console.log(it.next().value); // 'stram'  
console.log(it.next().value); // 'gram'  
console.log(it.next().done); // true
```

# Prototype et héritage



## Fonction génératrices

```
function* test(indexe) {  
    while (indexe < 2) { yield indexe++; }  
}
```

```
const iterateur = test(0);
```

```
console.log(iterateur.next().value); // 0
```

```
console.log(iterateur.next().value); // 1
```

# Prototype et héritage



## Fonction génératrices

```
function* compteVentesPommes () {
    var listeVentes = [3, 7, 5];
    for (var i = 0; i < listeVentes.length; i++) {
        yield listeVentes[i];
    }
}

var magasinPommes = compteVentePomme();
console.log(magasinPommes.next()); // { value: 3, done: false }
console.log(magasinPommes.next()); // { value: 7, done: false }
console.log(magasinPommes.next()); // { value: 5, done: false }
console.log(magasinPommes.next()); // { value: undefined, done: true }
```

# Prototype et héritage



Fonction génératrices

```
function* range(indexe, maxindexe) {  
    while (indexe < maxindexe) {    yield indexe++; }  
}
```

```
console.log( [...range(0,10)] )  
// affiche [0,1,2,3,4,5,6,7,8,9]
```

# Prototype et héritage



... au passage ...

```
function* range(indexe, maxindexe) {  
    while (indexe < maxindexe) { yield indexe++; }  
}
```

```
var a = [...range(0,10)]
```

```
console.log( a ) // affiche [ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 ]
```

Mais

```
console.log( ...a ) // affiche 0 1 2 3 4 5 6 7 8 9  
// même chose que :  
console.log( 0,1,2,3,4,5,6,7,8,9 )
```

Math.max(1,2,3,4)

Math.max( ...a )

# Prototype et héritage



## Fonction génératrices

```
function* test(indexe, maxindexe) {  
    while (indexe < maxindexe) { yield indexe++; }  
}  
  
const iterator = test(0,10);  
  
// l'itérateur produit les valeurs de 0 à 9
```

En Scala **yield** produit des valeurs d'une collection  
Pas ici : itérateur  
Pour collecter les valeurs dans une collection :

[...iterator]  
Fabrique une liste des valeurs de l'itérateur



# Utilitaires

- **String**  
à connaître :

charAt, concat, indexOf, lastIndexOf, match, replace, search, slice, split, substr, substring, valueOf, toLowerCase, toUpperCase

## indexOf

```
var t = "my name is Bond"
if (t.toUpperCase().indexOf("BOND")!= -1) {
    console.log("c'est un espion
britannique !")
}
```



# Utilitaires

- **String**  
**replace**

```
var t = "my name is <name>"  
console.log( t.replace("<name>","Bond") )
```

- split**

```
var t = "la bombe explose dans : 00 : 34 : 23"  
var hms = t.split(" :")  
console.log( hms[1] )  
console.log( hms[2] )  
console.log( hms[3] )
```



# Utilitaires

## • Array à connaître

concat, indexOf, join, lastIndexOf, pop, push, reverse, shift,  
slice, sort, splice, toString, unshift, valueOf

## Itération

```
var jours =  
["lundi","mardi","mercredi","jeudi","vendredi"]  
for (var i=0 ; i< jours.length ; i++) {  
    console.log( jours[i] )  
}  
for (i in jours) {  
    console.log( jours[i] )  
}
```



# Utilitaires

## • Array

### join

```
var boom = [00,23,34]
console.log( boom.join(":") )
```

### indexOf

```
var employes =
["maheo","courtrai","bond","guidec"]
if (employes.indexOf("bond")!= -1) {
    console.log("nous avons un espion parmi nous")
}
```

let :

```
var employes = ["maheo","courtrai","Bond","guidec"]
if (employes.indexOf("bond")!= -1) {
    console.log("nous avons un espion parmi nous")
}
```



# Utilitaires

## • Array : map ? extension Ecma 262

```
if (!Array.prototype.map)
{
    Array.prototype.map = function(fun, /* thisp */)
    {
        var len = this.length;

        if (typeof fun != "function")
            throw new TypeError();

        var res = new Array(len);
        var thisp = arguments[1];

        for (var i = 0; i < len; i++)
        {
            if (i in this)
                res[i] = fun.call(thisp, this[i], i, this);
        }
        return res;
    };
}
```

**var nombres = [1, 4, 9];**  
**var racines = nombres.map(Math.sqrt);**  
**document.write("roots is : " + roots );**



# Utilitaires

## ● Array : map

```
var employes = ["maheo","courtrai","Bond","guidec"]
if (employes.map(function(s) {return s.toUpperCase()} ).indexOf("BOND") != -1)
{
    console.log("nous avons un espion parmi nous")
}
```

remarque : jQuery !  
\$.map(employes,  
function...  
\$.each(employes,  
function ...

## ● Cas particulier

```
Array.prototype.convertMajuscules=function()
{
    for (i=0;i<this.length;i++)
    {
        this[i]=this[i].toUpperCase();
    }
}
```

attention !

employes.convertMajuscules



# Utilitaires

```
var arr = [1,2,3,4,5]  
var x = 0
```

## ● Array : exemples

```
x = arr.unshift("zero") // x == 6 (length) et arr == [zero,1,2,3,4,5]
```

```
x = arr.push(6,7,8) // x == 9 et arr == [zero,1,2,3,4,5,6,7,8]
```

```
x = arr.shift() // x == zero et arr == [1,2,3,4,5,6,7,8]
```

```
x = arr.pop() // x == 8 et arr == [1,2,3,4,5,6,7]
```

```
x = arr.splice(3,3,"four","five","six") // x == [4,5,6] arr==[1,2,3,four,five,six,7]
```

```
x = arr.splice(3,1) // x == four et arr == [1,2,3,five,six,7]
```

```
x = arr.splice(3) // x == five, six, 7 et arr == [1,2,3]
```



# Utilitaires

## ● Array :

```
var agents = [
    { nom: 'John Bond' , age : 77, ville : 'Paris' },
    { nom: 'Harry Cover' , age : 24, ville : 'Toulouse' },
    { nom: 'Alfred Henry' , age : 56, ville : 'Munich' },
    { nom: 'Honoré Bond' , age : 28, ville : 'Londres' },
    { nom: 'Franck Pomme' , age : 18, ville : 'Rome' }
]
```

## ● Affichage : impératif

```
for(var i = 0; i< agents.length; i++) {
    console.log(agents[i])
}
```



# Utilitaires

- **Array :**

```
var agents = [
    { nom: 'John Bond' , age : 77, ville : 'Paris'},
    { nom: 'Harry Cover' , age : 24, ville : 'Toulouse'},
    { nom: 'Alfred Henry' , age : 56, ville : 'Munich'},
    { nom: 'Honoré Bond' , age : 28, ville : 'Londres'},
    { nom: 'Franck Pomme' , age : 18, ville : 'Rome'}
]
```

- **Affichage : fonctionnel**

```
agents.forEach(function(a) { console.log(a) })
```

```
agents.forEach( (a) => { console.log(a) })
```



# Utilitaires

## •Array :

```
var agents = [
    { nom: 'John Bond' , age : 77, ville : 'Paris' },
    { nom: 'Harry Cover' , age : 24, ville : 'Toulouse' },
    { nom: 'Alfred Henry' , age : 56, ville : 'Munich' },
    { nom: 'Honoré Bond' , age : 28, ville : 'Londres' },
    { nom: 'Franck Pomme' , age : 18, ville : 'Rome' }
]
```

## •Affichage

```
agents.forEach(function(element, index) { console.log(index+":"+agents[index]) })
```

0:[object Object]  
1:[object Object]  
2:[object Object]  
3:[object Object]  
4:[object Object]



# Utilitaires

## •Array :

```
var agents = [
    { nom: 'John Bond' , age : 77, ville : 'Paris' },
    { nom: 'Harry Cover' , age : 24, ville : 'Toulouse' },
    { nom: 'Alfred Henry' , age : 56, ville : 'Munich' },
    { nom: 'Honoré Bond' , age : 28, ville : 'Londres' },
    { nom: 'Franck Pomme' , age : 18, ville : 'Rome' }
]
```

## •Remplacer les éléments

```
agents.forEach(
    function(element, index, tableau) {
        tableau[index].ville = tableau[index].ville.toUpperCase() }
)
console.log(agents)
```

```
[ { nom: 'John Bond', age: 77, ville: 'PARIS' },
  { nom: 'Harry Cover', age: 24, ville: 'TOULOUSE' },
  { nom: 'Alfred Henry', age: 56, ville: 'MUNICH' },
  { nom: 'Honoré Bond', age: 28, ville: 'LONDRES' },
  { nom: 'Franck Pomme', age: 18, ville: 'ROME' } ]
```



# Utilitaires

- Transformation d'un tableau
  - Muable : Mutation du tableau
  - Pas toujours souhaitable

```
var tab= [23, 255, 122, 5, 16, 99]
var hextab = tab.foreach( function(elt, i, t) {t[i] = t[i].toString(16) } )
console.log(tab) // [17,FF,7A,5,10,63]
```

- Crédit à la communauté
  - map

```
var tab= [23, 255, 122, 5, 16, 99]
var hextab = tab.map( function(elt) { return elt.toString(16) } )
console.log(hextab) // [17,FF,7A,5,10,63]
```



# Utilitaires

- Array :

```
var agents = [
    { nom: 'John Bond' , age : 77, ville : 'Paris'},
    { nom: 'Harry Cover' , age : 24, ville : 'Toulouse'},
    { nom: 'Alfred Henry' , age : 56, ville : 'Munich'},
    { nom: 'Honoré Bond' , age : 28, ville : 'Londres'},
    { nom: 'Franck Pomme' , age : 18, ville : 'Rome'}
]
```

- Age moyen : impératif

```
var ageMoyen = 0

for(var i=0; i<agents.length; i++)
    ageMoyen = ageMoyen + agents[i].age

ageMoyen = ageMoyen/ agents.length
```



# Utilitaires

- Array :

```
var agents = [
    { nom: 'John Bond' , age : 77, ville : 'Paris'},
    { nom: 'Harry Cover' , age : 24, ville : 'Toulouse'},
    { nom: 'Alfred Henry' , age : 56, ville : 'Munich'},
    { nom: 'Honoré Bond' , age : 28, ville : 'Londres'},
    { nom: 'Franck Pomme' , age : 18, ville : 'Rome'}
]
```

- Age moyen : fonctionnel

```
var ageMoyen = agents.reduce(
    (a1, a2) => { age: (a1.age + a2.age) }
).age / agents.length
```



# Utilitaires

- Array :

```
var agents = [
    { nom: 'John Bond' , age : 77, ville : 'Paris'},
    { nom: 'Harry Cover' , age : 24, ville : 'Toulouse'},
    { nom: 'Alfred Henry' , age : 56, ville : 'Munich'},
    { nom: 'Honoré Bond' , age : 28, ville : 'Londre'},
    { nom: 'Franck Pomme' , age : 18, ville : 'Rome'}
]
```

- Tous majeurs :

```
var tousMajeurs = agents.every(
    function(a) {
        return (a.age >= 18)
    }
) // true
```



# Utilitaires

- Array :

```
var agents = [
    { nom: 'John Bond' , age : 77, ville : 'Paris' },
    { nom: 'Harry Cover' , age : 24, ville : 'Toulouse' },
    { nom: 'Alfred Henry' , age : 56, ville : 'Munich' },
    { nom: 'Honoré Bond' , age : 28, ville : 'Londres' },
    { nom: 'Franck Pomme' , age : 18, ville : 'Rome' }
]
```

- Espions retraités :

```
var retraite = agents.filter(
    (tropVieux) => tropVieux.age > 40
)
```

```
[ { nom: 'John Bond', age: 77, ville: 'Paris' },
  { nom: 'Alfred Henry', age: 56, ville: 'Munich' } ]
```



# Utilitaires

## • Array :

```
var agents = [
    { nom: 'John Bond' , age : 77, ville : 'Paris' },
    { nom: 'Harry Cover' , age : 24, ville : 'Toulouse' },
    { nom: 'Alfred Henry' , age : 56, ville : 'Munich' },
    { nom: 'Honoré Bond' , age : 28, ville : 'Londres' },
    { nom: 'Franck Pomme' , age : 18, ville : 'Rome' }
]
```

## • Quelles sont les villes des retraités :

```
var villes = agents.filter(
    function(tropVieux) { return tropVieux.age > 40 }
).map( function(a) { return a.ville })
```

[ 'Paris', 'Munich' ]



# Utilitaires

- Exceptions

- try / catch

```
try {  
    // code à tester  
} catch( erreur ) {  
    console.log(erreur.name+  
" :" +erreur.message)  
}
```

- throw

```
try {  
    if (...) throw { name :"oops", message :"ah ben"  
}  
} catch( erreur ) {  
    console.log(erreur.name+ " :" +erreur.message)  
} // finally
```

finally



# Clôtures

- Une *clôture* est une fonction dont le contenu a accès à l'extérieur de la fonction

... exemple

```
function montreNom (prenom, nom) {  
    var nameIntro = "Votre nom est ";  
  
    function nomComplet () {  
        return nameIntro + prenom + " " + nom;  
    }  
  
    return nomComplet ();  
}  
  
montreNom ("James", "Bond"); // Votre nom est James Bond
```



# Clôtures

- Les références sont conservées

```
function nomCelebre(prenom) {  
    var nameIntro = "La star c'est ";  
  
    function leNom(nom) {  
        return nameIntro + prenom + " " + nom  
    }  
    return leNom;  
}  
  
var sonNomComplet = nomCelebre("Justin");  
  
console.log(sonNomComplet("Bieber"))  
// La star c'est Justin Bieber
```

# Clôtures



- ... et peuvent changer

```
function celebreteID () {
    var identifiant = 999;

    return {
        getID: function () {
            return identifiant;
        },
        setID: function (nouvelID) {
            identifiant = nouvelID;
        }
    }
}

var id = celebreteID ()
    console.log(id.getID()) // 999
id.setID(567)
    console.log(id.getID()) // 567
```

jQuery  
node.js



# Clôtures

```
let ajouteur = nombre => valeur => valeur + nombre

let ajoute10 = ajouteur(10);
ajoute10(1); // affiche 11

let ajouteur = nombre =>
    valeur => valeur + nombre
let ajoute10 = ajouteur(10);
ajoute10(1); // affiche 11
```

```
function ajouteur(nombre) {
    function ajoute(valeur) {
        return valeur + nombre;
    }
    return ajoute;
}
```

# Clôtures / Scala



```
def ajouteur(n: Int)(x: Int) = (x + n)
```

```
def ajoute10 = ajouteur(10)_
```

```
println(ajoute10(1))
```



# Clôtures

- **Clôture fonctionnelle**
- Une clôture fonctionnelle est le mécanisme qui assure l'existence et l'accès du contexte d'une fonction
- Ce qui est nécessaire à l'exécution de la fonction est préservé hors contexte

## Fermeture (informatique)

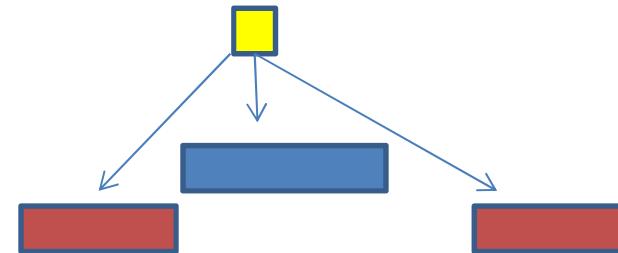
Pour les articles homonymes, voir [Closure](#) et [Fermeture](#).

Dans un [langage de programmation](#), une **fermeture** ou **clôture** (en anglais : [closure](#)) est une [fonction](#) accompagnée de son environnement lexical. L'environnement lexical d'une fonction est l'ensemble des variables *non locales* qu'elle a capturé, soit par *valeur* (c'est-à-dire par copie des valeurs des variables), soit par *référence* (c'est-à-dire par copie des adresses mémoires des variables)<sup>1</sup>. Une fermeture est donc créée, entre autres, lorsqu'une fonction est définie dans le corps d'une autre fonction et utilise des paramètres ou des variables locales de cette dernière.

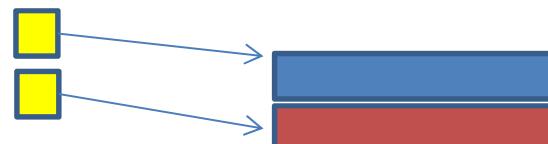
# Programmation Asynchrone



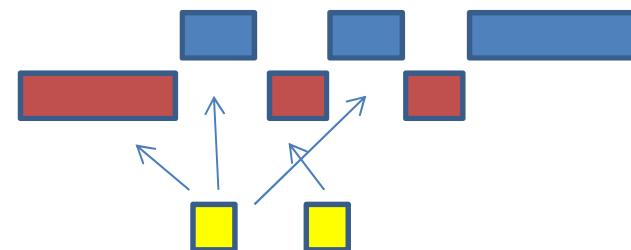
- Asynchrone



- Parallèle



- Concurrent

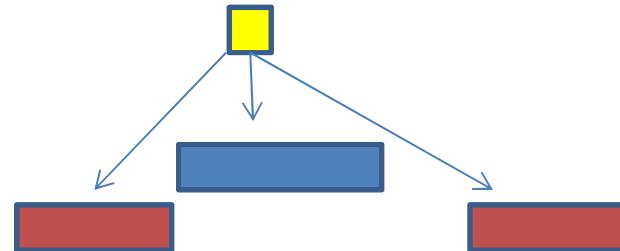


# Programmation **Asynchrone** et concurrente



- Javascript mono thread

- Asynchrone
- Voir Ajax
- Extensions ECMA 7



- setTimeout

- setInterval

- Ajax et IO (voir node.js)

# Programmation **Asynchrone** et concurrente



## • **setTimeout**

- Delais 1000 = 1s
- Attendre le délais puis appel de la fonction

```
setTimeout("alert('dring')", 1000)
```

À éviter

### • **timId**

- Descripteur de temporisateur
- Arrêt avant déclenchement

```
setTimeout(function() { alert('dring') }, 1000)
```

```
clearTimeout(timerId)
```

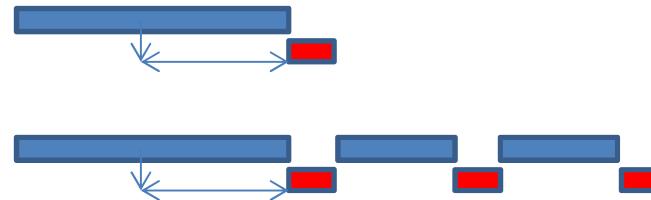
### • Par exemple

- Lancer une requête réseau ou autre et un timeout
- Si la requête se termine -> clearTimeOut
- Sinon, au bout d'un moment le code est exécuté
  - Traitement de l'échec de la requête

# Programmation Asynchrone et concurrente



- `setTimeout`
- `setInterval`



```
<div id="tictac">
  <span id="h">hh</span>:<span id="m">mm</span>:<span id="s">ss</span>
</div>
```

```
var timerId
```

```
function debutHorloge() {
  if (timerId) return
  timerId = setInterval(miseAJour, 1000)
  miseAJour()
}
```

```
function arretHorloge() {
  clearInterval(timerId)
  timerId = null
}
```

16:35:06

Start Stop

```
function miseAJour() {
  var date = new Date()

  var h= date.getHours()
  if (h < 10) h= '0'+hours
  document.getElementById('h').innerHTML = h

  var m= date.getMinutes()
  if (m < 10) m= '0'+m
  document.getElementById('m').innerHTML = m

  var s= date.getSeconds()
  if (s < 10) s = '0'+s
  document.getElementById('s').innerHTML = s
}
```





16:35:06

Start

Stop

```
<html>
<style>#h{color:red}#m{color:green}#s{color:blue}</style>
<body>
<div id="tictac">
<span id="h">hh</span>:<span id="m">mm</span>:<span id="s">ss</span>
</div>
<script type="text/javascript">

var timerId

function miseAJour() {
    var date = new Date()

    var h = date.getHours()
    if (h < 10) h = '0'+h
    document.getElementById('h').innerHTML = h

    var m = date.getMinutes()
    if (m < 10) m = '0'+m
    document.getElementById('m').innerHTML = m

    var s = date.getSeconds()
    if (s < 10) s = '0'+s
    document.getElementById('s').innerHTML = s
}

function debutHorloge() {
    if (timerId) return

    timerId = setInterval(miseAJour, 1000)
    miseAJour()
}

function arretHorloge() {
    clearInterval(timerId)
    timerId = null
}

</script>

<input type="button" onclick="clockStart()" value="Start">
<input type="button" onclick="clockStop()" value="Stop">
</body>
```



## ● Attention !!!

```
<input type="button"
      onclick="setTimeout(function() { this.value='OK' }, 100)"
      value="Clickez ici"
>
```

non

```
<input id="ok" type="button" value="Click me">
<script>
  document.getElementById('ok').onclick = function() {
    var that = this
    setTimeout(function() { that.value='OK' }, 100)
  }
</script>
```

ok

Capture de clôture

ok

```
<input type="button"
      onclick="var ici=this; setTimeout(function() { ici.value='OK' }, 100)"
      value="Clickez ici"
>
```



- Asynchrone dans le détail
  - En fait, liste d'évènements à traiter
    - Event List
    - Dès que possible
    - Possibilité de placer soi-même des evt
      - `setTimeout( ...., 0)`
      - Quelques fois, le navigateur déclenche l'action `onClick` puis l'action par défaut : si le script doit se faire après l'action par défaut...
        - ... utilisez `setTimeout(..., 0)`



- Web worker : HTML5
  - setTimeout
  - SetInterval
- Nouveau Thread Javascript
  - var w = new Worker('codep.js');
  - w.postMessage() ; // démarrage du worker
  - Execution du code en //
- Communication par messages
  - Cf AKKA



## • Exemple

```
// dans codep.js
var n = 1;
search: while (true) {
    n++;
    for (var i = 2; i <= Math.sqrt(n); i++)
        if (n % i == 0)
            continue search;
    // trouvé un nb premier
    postMessage(n);
}
```

```
<head>
<title>Exemple</title>
</head>
<body>
<p>Dernier nb premier trouvé <output id="resultat"></output></p>
<script>
var worker = new Worker('codep.js');
worker.onmessage = function (event) {
    resultat.textContent = event.data;
};
</script>
</body>
</html>
```



Aucun accès au DOM dans le WW  
Pas de console.log  
Pour quoi faire :

analyse d'image  
ajax bd infos importantes  
analyse texte



- Attention
  - Comportement local
  - About:config



## Agissez avec précaution

Modifier les préférences de configuration avancées peut affecter les performances et la sécurité de votre navigateur.



M'avertir lorsque j'essaie d'accéder à ces préférences

[Accepter le risque et poursuivre](#)



- Attention
  - Comportement local
  - `privacy.file_unique_origin`

privacy.file\_unique\_origin

**privacy.file\_unique\_origin**

**false**



```
<head>
  <title>Exemple</title>
</head>
<body>
  <p>Dernier nb premier trouvé <output id="resultat"></output></p>
  <script>
    var worker = new Worker('codep.js');
    worker.onmessage = function (event) {
      resultat.textContent = event.data.n+" "+event.data.nb;
    };
  </script>
</body>
</html>
```

● **Exemple**

```
// dans codep.js
var n = 1;
var combien = 0
search: while (true) {
  n++;
  for (var i = 2; i <= Math.sqrt(n); i++)
    if (n % i == 0) {
      continue search;
    }
  // trouvé un nb premier
  combien = combien + 1
  postMessage({n: n, nb:combien});
}
```

Dernier nb premier trouvé 268817 23547

Aucun accès au DOM dans le WW

Pas de console.log

Pour quoi faire :

analyse d'image

ajax bd infos importantes

analyse texte



# Navigateur & Javascript

## • Extensions

- Ecma6 - *harmony*, 7 etc
- **Destructuration / déconstructeur**

```
var dpt = [ 35, 56, 29]
```

```
var [Ivaine, Morbihan, Finistere] = dpt
```

```
console.log(Morbihan) // 56
```

## • *Pattern Matching (à la Scala)*

```
var dpt = [ 35, 56, 29]
```

```
var dpt = [ 35, 56, 29]
```

```
var [Ivaine, ...autres] = dpt
```

```
var [Ivaine, , Finistere] = dpt
```

```
console.log(autres) // [56, 29]
```

```
console.log(Finistere) //29
```



# Navigateur & Javascript

## • Extensions

- Problème :
  - pas toujours supportés par les interpréteurs
- node.js
  - --harmony
- Firefox
  - Moteur Js = SpiderMonkey
  - Support partiel
- Chrome & opera
  - Chrome://flags
- Babel (transpile)
- Google traceur-compiler
  - ES6 en entrée -> ES5



# Navigateur & Javascript

- Traceur-compiler

Source      **Traceur Transcoding Demo**      Options

```
1 var dpt = [29, 35, 56]
2 var [finistere, ...autres] = dpt
3 console.log(autres)
```

```
1 $traceurRuntime.ModuleStore.getAnonymousModule(fu
2   "use strict";
3   var $__2,
4     $__3;
5   var dpt = [29, 35, 56];
6   var $__1 = dpt,
7     finistere = ($__2 = $__1[$traceurRuntime.tol
8       autres = $traceurRuntime.iteratorToArray($__1
9       console.log(autres);
10      return {};
11    });
12 //# sourceURL=traceured.js
13
```

<http://google.github.io/traceur-compiler/demo/repl.html>

Possibilité d'inclure le traducteur dans la page (voir Angular)



# Navigateur & Javascript

## • Extensions

- Ecma6 - *harmony*, (7 en cours)
- Variables locales à un bloc : **let** / **const**

```
function test() {  
    var a = 5  
    if (a == 5) {  
        var o = 67  
    }  
    console.log(o)  
  
}  
  
test() // 67
```

```
function test() {  
    var a = 3  
    if (a == 5) {  
        var o = 67  
    }  
    console.log(o)  
  
}  
  
test() // undefined ?
```

```
function test() {  
    var a = 5  
    if (a == 5) {  
        let o = 67  
    }  
    console.log(o)  
  
}  
  
test() // undefined
```



# Navigateur & Javascript

## • Extensions

```
function test() {  
    var a = 5  
    if (a == 5) {  
        var o = 67  
    }  
    console.log(o)  
}
```

```
test() // 67  
console.log(o) // undefined
```

```
function test() {  
    var a = 5  
    if (a == 5) {  
        o = 67  
    }  
    console.log(o)  
}
```

```
test() // 67  
console.log(o) //67
```



# Navigateur & Javascript

## • Extensions

o = 5

var o = 5

let o = 5

```
fonction(a,b,c) {  
    if () {  
        if () {  
            }  
        }  
    }  
  
    if () {  
    }
```



# Navigateur & Javascript

## • Extensions Ecma6

### • Map

```
var wm = new Map();
```

```
var obj = {nom: "bond"}
```

```
wm.set(obj, "7");
console.log(wm.get(obj)); // 7
```

```
(function () {
  let o = {};
  wm.set(o, "bar");
})();
```

o n'existe plus

?!?

```
for(var v of wm) { console.log(v)}
// [ { nom: 'bond' }, 7 ]
// [ {}, 'bar' ]
```

mais en fait, si...



# Navigateur & Javascript

```
> var wm = new Map()
undefined
> var obj = { nom: "Bond" }
undefined
> wm.set(obj,"7")
Map { { nom: 'Bond' } => '7' }
> wm.get( { nom: "Bond" } )
undefined
> wm.get( obj )
'7'
> (function() { let o = {}; wm.set(o, "bar") ; })()
undefined
> wm
Map { { nom: 'Bond' } => '7', {} => 'bar' }
> -
```



# Navigateur & Javascript

## • Extensions Ecma6

- WeakMap
- objets comme clés
- (seulement !)

```
Thrown:  
TypeError: wm.entries(...) is not a function  
  
> console.log(wm.entries())  
Thrown:  
TypeError: wm.entries is not a function  
> for( var v in wm.entries() ) console.log(v)  
Thrown:  
TypeError: wm.entries is not a function  
> wm  
WeakMap { <items unknown> }  
> wm.foreach( (value, key) => console.log( ke  
Thrown:  
TypeError: wm.foreach is not a function  
> wm.forEach( (value, key) => console.log( ke  
Thrown:  
TypeError: wm.forEach is not a function
```

```
var wm = new WeakMap();
```

```
var obj = {nom: "bond"}
```

```
wm.set(obj, "7");  
console.log(wm.get(obj)); // 7
```

```
(function () {  
    let o = {};  
    wm.set(o, "bar");  
})();
```

o n'existe plus

```
// for(var v of wm) { console.log(v)}  
// Argh!
```

# Navigateur & Javascript



```
> wm.has( obj )
true
> wm.has(o)
Thrown:
ReferenceError: o is not defined
> wm.has({})
false
>
```



# Navigateur & Javascript

- **Extensions Ecma6**

- **Structures**

```
const Point2D = new StructType({ x: uint32, y: uint32 });
const Color = new StructType({ r: uint8, g: uint8, b: uint8 });
const Pixel = new StructType({ point: Point2D, color: Color });

var p = new Point2D(10,10)
```

case class Scala



# Navigateur & Javascript

## • Extensions Ecma6

### • Classes (!)

```
class Point extends Base {  
    constructor(x,y) {  
        super();  
        this[px] = x,  
        this[py] = y;  
        this.r = function() {  
            return Math.sqrt(x * x + y * y);  
        }  
    }  
  
    get x() { return this[px]; }  
    get y() { return this[py]; }  
}
```

PAS DE FUNCTION



# Navigateur & Javascript

## • Extensions Ecma6

- Variables privées, arguments par défaut
- Modules
- Générateurs

```
function* fibonacci() {  
    var i = 0, j = 1;  
    while (true) {  
        yield i; var t = i; i = j; j += t;  
    }  
}
```

Scala (lazy, for/yield etc)

```
var f = fibonacci();  
for (var i = 0; i < 10; i++) {  
    console.log(f.next()); // 0, 1, 1, 2, 3, 5, 8, 13, 21, 34  
}
```



# Navigateur & Javascript

## • Extensions Ecma7-8 (en cours)

- Macros
- Traitements parallèles
- Contrats
- DSL
- Promises & Futures
- ... voir MongoDB
- ...voir Scala



# Navigateur & Javascript

## • Extensions à Javascript

### • JSX & React

```
function formatName(user) {
  return user.firstName + ' ' + user.lastName;
}

const user = {
  firstName: 'Harper',
  lastName: 'Perez'
};

const element = (
  <h1>
    Hello, {formatName(user)}!
  </h1>
);

ReactDOM.render(
  element,
  document.getElementById('root')
);
```



# Navigateur & Javascript

- Extensions à Javascript

- TypeScript

```
function greeter(person: string) {  
    return "Hello, " + person;  
}  
  
let user = [0, 1, 2];  
  
document.body.innerHTML = greeter(user);
```



```
# Assignment:  
number = 42  
opposite = true  
  
# Conditions:  
number = -42 if opposite  
  
# Functions:  
square = (x) -> x * x  
  
# Arrays:  
list = [1, 2, 3, 4, 5]  
  
# Objects:  
math =  
  root: Math.sqrt  
  square: square  
  cube: (x) -> x * square x  
  
# Splats:  
race = (winner, runners...) ->  
  print winner, runners  
  
# Existence:  
alert "I knew it!" if elvis?  
  
# Array comprehensions:  
cubes = (math.cube num for num in list)
```



```
// Assignment:  
var cubes, list, math, num, number, opposite, race, square;  
  
number = 42;  
  
opposite = true;  
  
if (opposite) {  
    // Conditions:  
    number = -42;  
}  
  
// Functions:  
square = function(x) {  
    return x * x;  
};  
  
// Arrays:  
list = [1, 2, 3, 4, 5];  
  
// Objects:  
math = {  
    root: Math.sqrt,  
    square: square,  
    cube: function(x) {  
        return x * square(x);  
    }  
};  
  
// Splats:  
race = function(winner, ...runners) {  
    return print(winner, runners);  
};
```



# Navigateur & Javascript

## •Développement Javascript

- Navigateur + js (+css/HTML)
- Déploiement classique
- Création d'une application autonome

The screenshot shows the NW.js website homepage. At the top is a blue header with the NW.js logo (a compass icon) and the text "NW.js". Below the header is a dark grey navigation bar with links for "HOME | DOWNLOADS | BLOG | DOCUMENTATION". The main content area has a light blue background. It features a large "Download *Stable* for Windows (x64)" button with the subtext "Chromium 80 + Node 13.6.0". Below this are two blue buttons: "v0.44.0 NORMAL" and "v0.44.0 SDK". A "Release Notes" link is located between them. At the bottom of the main content area is a blue footer bar with the text "View on GitHub".

Node Web kit

Osx 32 64

Linux 32 64

Win 32 64

NW.js (previously known as node-webkit) lets you call all Node.js modules directly from DOM and enables a new way of writing applications with



# Navigateur & Javascript

## • Développement Javascript

- Navigateur + js (+css/HTML)
- Déploiement classique
- Création d'une application autonome





# HTML & navigateur

## •Principe

- HTML : codage de la structure de la page
  - Contenus statiques
  - Balises <H1 id = "id"> ... </H1>

- CSS : codage des différents styles (couleur, position etc)

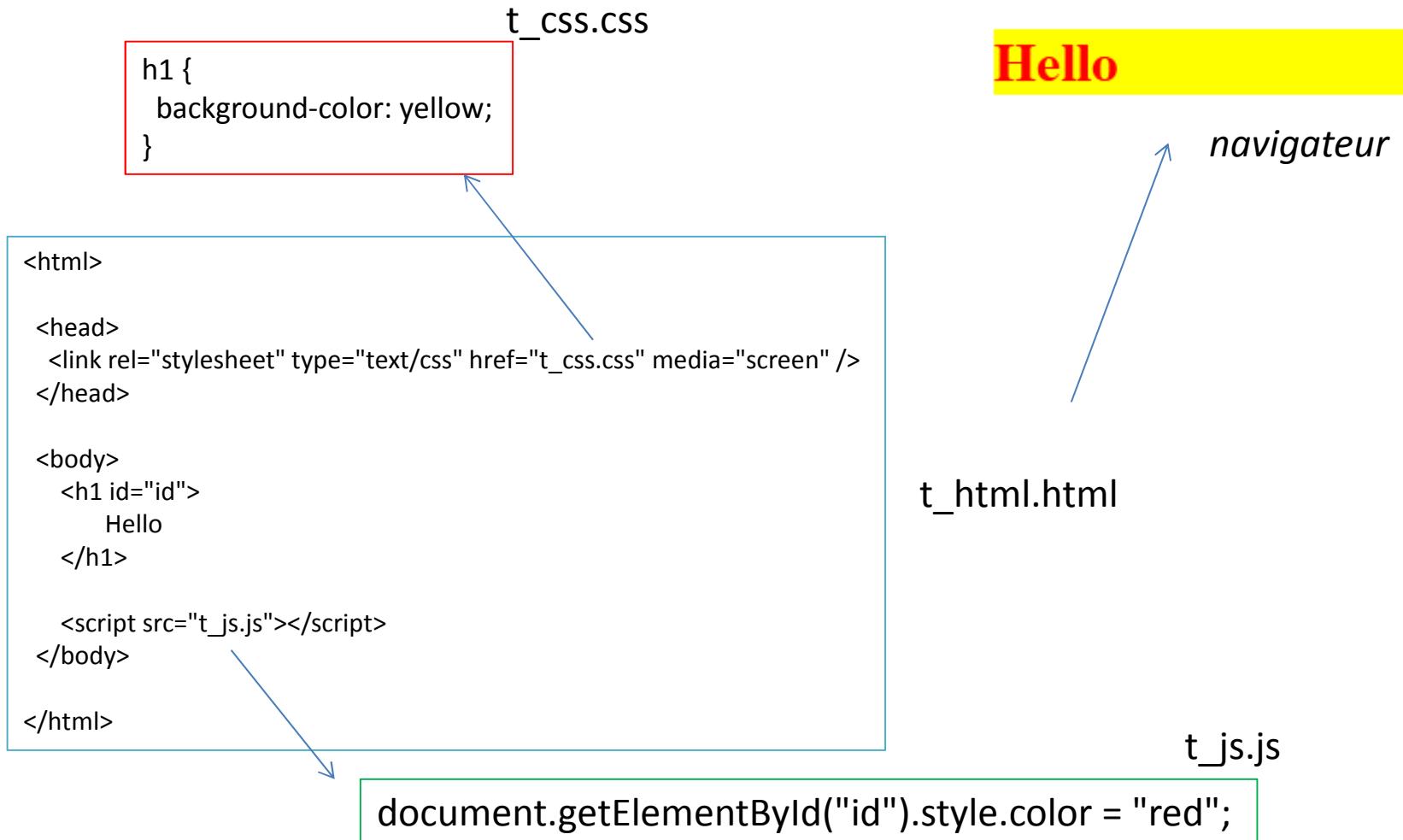
```
H1 {  
    background-color: yellow;  
}
```

- Javascript :

- Injection de contenu
- Gestion de formulaires
- Ajax
- Etc..

```
document.getElementById("id").style.color = "red";
```

# HTML & navigateur





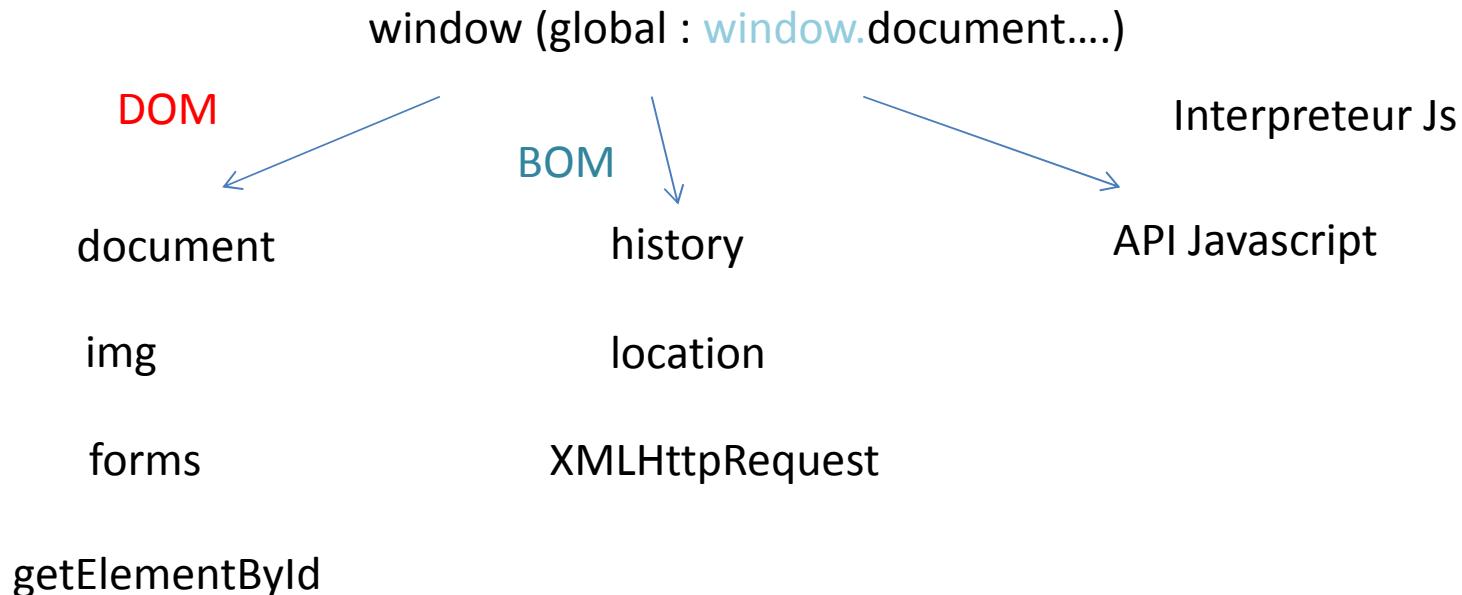
# HTML & navigateur

- Bien séparer
  - HTML
  - CSS
  - Javascript
- CSS dans <head>
- JS dans <head> ou fin <body>



- **CSS à revoir (ou à voir)**
- CSS 3 (CSS4 ?)

# HTML & navigateur



DOM : Document Object Model (DOM 0, 1, 2, 3, 4?) <http://www.w3.org/DOM/DOMTR>  
BOM : Browser Object Model



# HTML & navigateur

- DOM level 2 & 3
  - DOM 2 (actuel)
    - getElementById
    - addEventListeners, handleEvents
    - NodeIterator, TreeWalker
  - DOM 3 (en cours)
    - Load, Save
    - Lecture de documents DOM <-> XML
    - Xpath 1.0
- Mais... jQuery (voir plus loin)

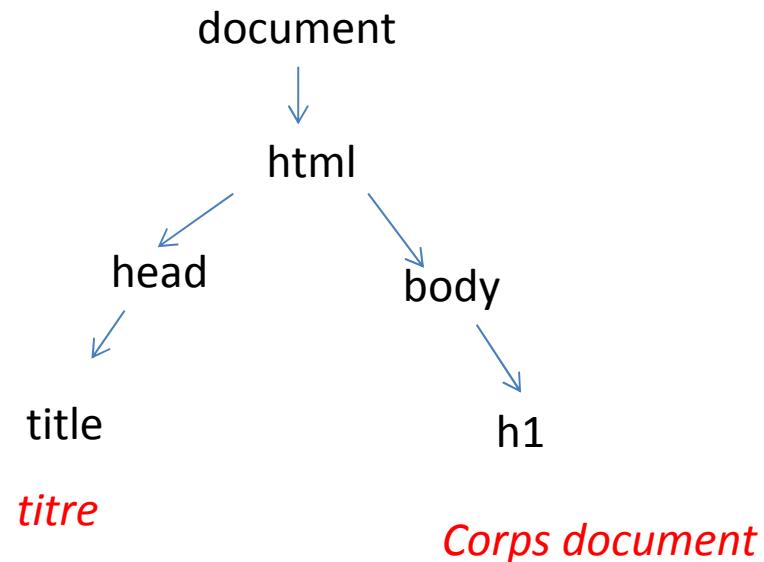


# HTML & navigateur

- Principes DOM

- Le document est représenté comme une arborescence d'objets

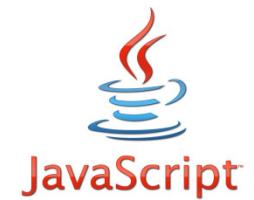
```
<html>
  <head> <title> titre </title> </head>
  <body style="background-color:lightgray">
    <h1 id = "test"> Corps document </h1>
  </body>
</html>
```



document.body.style = "background-color:red"

document.body.style.backgroundColor = 'red'

document.getElementById("test").style.color="red"



# HTML & navigateur

- FireFox : FI2
  - Permet de visualiser le DOM

The screenshot shows the Firefox Developer Tools (FI2) interface. The left sidebar displays "Android Studio" and "Workflow" sections, with "Tools Help" expanded to show "SDK Tools". The main content area is titled "Recommended Packages" with a note about SDK tools. The bottom half of the screen is the "Inspector" tab, which has tabs for "Inspector", "Console", "Debugger", "Style Editor", "Performance", "Network", and others. The "Inspector" tab is active, showing a DOM tree on the left and a "Rules" panel on the right. The DOM tree highlights the `<body>` element with the class "gc-documentation develop". The "Rules" panel shows the following CSS rule:

```
*::-moz-selection { background-color: #09C; color: #FFF; }
```

The "Computed" tab in the Rules panel shows the following styles for the `a` element:

```
element { }
```

The "Fonts" tab shows:

```
body { }
```



# HTML & navigateur

- FireFox : F12
- Les scripts de la page

The screenshot shows the Firefox Developer Tools interface with the 'Debugger' tab selected. The top bar has tabs for Inspector, Console, Debugger (selected), Style Editor, Performance, and Network. A search bar at the top right says 'Search scripts (Ctrl+P)'. The main area is divided into two panes: 'Sources' on the left and 'Call Stack' on the right. The 'Sources' pane lists several URLs and files:

- https://apis.google.com
- plusone.js
- cb=gapi.loaded\_0
- https://developer.android.com
- android\_3p-bundle.js
- docs.js
- jd\_tag\_helpers.js
- sdk-manager.html
- https://www.google-analytics....
- analytics.js
- https://www.google.com
- jsapi

The 'Call Stack' pane is currently empty. The code editor on the right displays the source code for the 'analytics.js' file, which is heavily obfuscated. The code starts with:

```
1 (function(){var n=Math,f>window,aa=encodeURIComponent;function P(a,b){return a.href=b}
2 var m="match",Ab="href",Ga="sendBeacon",A="split",H="join",ha="slice",jd="target",v="name
3 c,!!d):a.attachEvent&&a.attachEvent("on"+b,c)}catch(e){J(27)}},wa=function(a,b){if(a){var
4 a||":==a)return;return b}},za=function(a,b){if(1==b[y]&&null!=b[0]&&"object"==typeof b[
5 !1)return d},Cc=function(a){return K(a)[Qc]((\(/g,"%28")[Qc]((\(/g,"%29")),vc=/^(www\.)?g
6 a,!0);e.withCredentials=!0;e.setRequestHeader("Content-Type","text/plain");e.onreadystatechange=
7 function Pa(a){try{O[oa][Ga]}J(42):0.XMLHttpRequest&&"withCredentials"in new O.XMLHttpRequest
8 function Sa(a){var b=P(a,gd)||oc()"/collect",c=P(a,fa);!c&&a.get(Vd)&&(c=="beacon");if(c)
9 function vb(a){if(!a.get(Na))throw"abort";};var hd=function(){return n.round(2147483647*n
10 var ab=function(a,b,c,d){if(void 0!=c)switch(b){case Na:wb[pa](c)}var e=$a(b);e&&e.o?e.o(
11 a+"$"),b]},T=function(a,b,c){return S(a,b,c,void 0,db)},db=function(){}};var gb=q(a(f).Goog
12 var kb=S("location","dl","","lb=S("referrer","dr"),mb=S("page","dp","","S("hostname","dh"
13 S("campaignKeyword","ck");S("campaignContent","cc");var ub=S("eventCategory","ec"),xb=S("e
14 S("appVersion","av","","S("appId","aid","","S("appInstallerId","aiid","","S("exDescriptio
15 S("forceSSL",void 0,void 0,function(){return Ba},function(a,b,c){J(34);Ba=!!c});var ed=S(
16 var Qb=T("_oot"),dd=S("previewTask"),Rb=S("checkProtocolTask"),md=S("validationTask"),Sb=
17 Wc=T("legacyHistoryImport",void 0,!0),ac=T("storage",void 0,"cookie"),bc=T("allowLinker",
18 (a=b[m]/[\d]+/g))&&3<=a[y]&&(b=a[0]"+.+a[1]+" r"+a[2]);return b||void 0};var gc=functio
19 a[Fb]=b.responseEnd-b.responseStart;a[Hb]=b.fetchStart-c;a[Kb]=b.domInteractive-c;a[Lb]=b
20 Yc=function(a){if(a.get(Wc)){var b=P(a,W),c=P(a,$b)||xa(),d=Xc("__utma",c,b);d&&(J(19),a.
21 a),ic=function(a){return lc(a[A](".")[y]),kc=function(a){if(!a)return"/";1<a[y]&&a.lastI
22 Dc[z].ca=function(a,b){if(a.tagName){if("a"==a.tagName[I]()){a[Ab]&&Pc(a,qd(this,a[Ab],b)
23 var qd=function(a,b,c){var d=pd.exec(b);d&&3<=d[y]&&(b=d[1]+(d[3]?d[2]+d[3]:""));a=a[jd].
24 b.method[I]())&&(b.action=qd(a,b[kd]))}}};
```



# HTML & navigateur

- FireFox : F12
- Code CSS

The screenshot shows the Firefox Developer Tools interface, specifically the Style Editor tab. The left sidebar lists various CSS files and their rule counts: 'CSS' (1 rule), 'roboto' (7 rules), 'default.css' (1305 rules), and 'default+en.css' (350 rules). The main pane displays the following CSS code:

```
1 @font-face {  
2   font-family: 'Roboto Condensed';  
3   font-style: normal;  
4   font-weight: 400;  
5   src: local('Roboto Condensed'), local('RobotoCondensed-Regular');  
6 }  
7 }
```



# HTML & navigateur

- FireFox : F12
- Les accès Ajax / HTTP

A screenshot of the Firefox Developer Tools Network tab. The tab bar shows 'Console' highlighted with a blue oval. Below the tab bar is a message: 'Required. Your new SDK installation installs the latest version. Be sure to respond to the Android S... prompts to keep your SDK Tools up-to-date.' The main area displays a table of network requests:

Method	File	Domain	Type	Transferred	Size	0 ms	1,37
200	GET default.css?v=7	developer.android...	css	cached	0 KB		
200	GET default.css?v=7	developer.android...	css	cached	0 KB		
200	GET default.css?v=7	developer.android...	css	cached	0 KB		
200	GET default.css?v=7	developer.android...	css	cached	0 KB		
200	GET css?family=Roboto+Condensed	fonts.googleapis.c...	css	cached	0 KB		
200	GET css?family=Roboto:light,regular,...	fonts.googleapis.c...	css	cached	0 KB		
200	GET default+en.css	www.google.com	css	cached	0 KB		

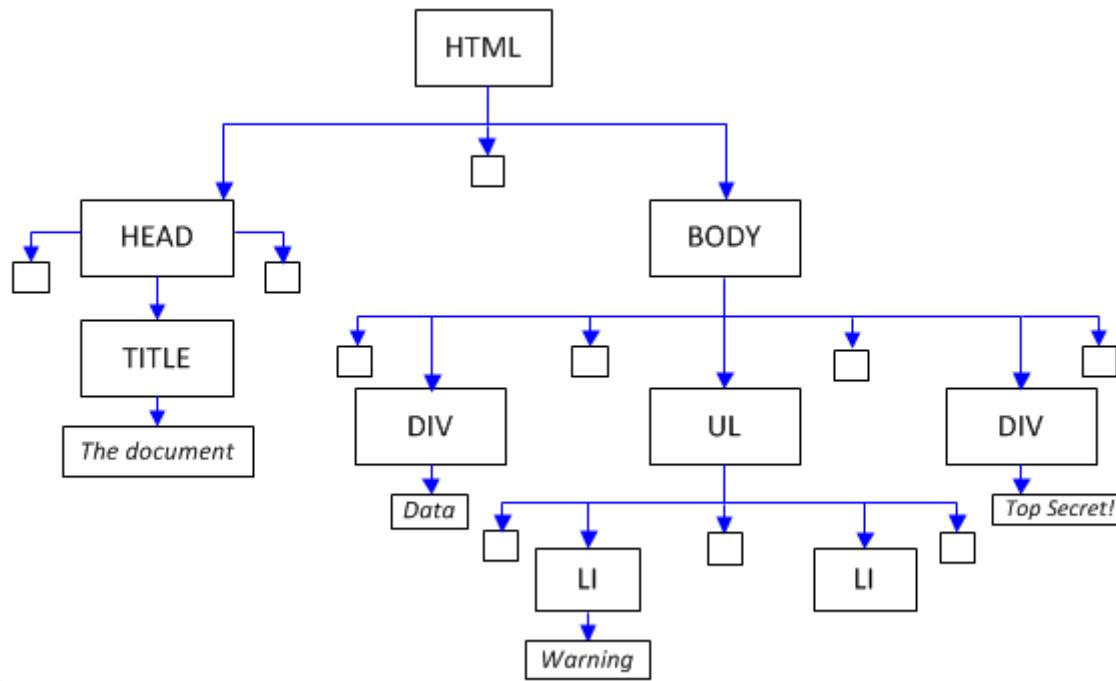
... sinon, alert("hello")

Remarque : console.log()

# HTML & navigateur

- Pièges

- WhiteSpaces (cf XML)
- Retours de ligne, avant et après balises etc.
- Attention !



Privilégier `document.getElementById` !!



# HTML & navigateur

- DOM 1

- Fonctions et méthodes pour
- Rajouter un nœud, enlever un nœud
- Se déplacer dans l'arbre DOM
- Tester des valeurs d'attributs
- Les chemins dépendent des navigateurs
- Effort de normalisation

- DOM 2

- Recherche de balises
- Méthodes de recherches
  - getElementById* etc..

- DOM3

- Xpath (mais jQuery)



# HTML & navigateur

- DOM 2

- `document.getElementById`
- Moins connu : référence automatique

```
<h1 id="idt">  
    Hello  
</h1>
```

- Fabrique une variable Javascript `idt`

```
alert(idt.style.color);
```

- `document.getElementsByTagName("div")`

- Attention au 's'
- Renvoie un tableau
- Peut aussi être appelé sur un nœud
  - Sous-hiérarchie



# HTML & navigateur

- DOM 2

- `document.getElementsByClassName()`
- Class CSS

```
<h1 class="a b c">
    Hello
</h1>
<script>
    alert( document.getElementsByClassName('a')[0].innerHTML )
</script>
```

- `innerHTML`
- `text`



# HTML & navigateur

- DOM 2

- `document.querySelectorAll`
- Requête CSS3

```
<ul>
  <li> am </li>
  <li> stram </li>
  <li> gram </li>
</ul>
<ul>
  <li> pic </li>
  <li> et </li>
  <li> pic </li>
</ul>
<script>
  var elt = document.querySelectorAll("UL > LI:last-child")
  for(var i=0; i<elt.length; i++) {
    alert(elt[i].innerHTML)
  }
</script>
```

`querySelector == querySelectorAll(...)[0]`

`querySelector` (le premier qui est ok)  
`querySelectorAll` (tous)



# HTML & navigateur

- DOM 2-3 (FireFox – Mozilla)
  - Xpath

```
var xpathResult = document.evaluate(  
    xpathExpression,          // expression XPath  
    contextNode,              // nœud racine (document)  
    namespaceResolver,        // espace de nom (uri)  
    resultType,               // XPathResult.ANY_TYPE  
    result );                 // null ou objet XPathResult
```

- exemple

```
var paragraphCount = document.evaluate(  
    'count(//p)',  
    document,  
    null,  
    XPathResult.ANY_TYPE,  
    null );  
alert( 'Ce document contient ' + paragraphCount.numberValue + ' éléments de paragraphe' );
```

Conversion en nombre

# HTML & navigateur



```
var headings = document.evaluate('//h2', document, null, XPathResult.ANY_TYPE, null );  
  
var thisHeading = headings.iterateNext();  
  
var alertText = 'Les en-têtes de niveau 2 présents dans ce document sont :\n'  
  
while (thisHeading) {  
    alertText += thisHeading.textContent + '\n';  
    thisHeading = headings.iterateNext();  
}  
}
```



# HTML & navigateur

- querySelector / querySelectorAll etc..

- DOM Classique

```
var table = document.getElementById("matable");
var tbodies = table.tBodies;
var rangees = null;
var cellules = [];

for (var i = 0; i < tbodies.length; i++) {
    rangees = tbodies[i].rows;
    for (var j = 0; j < rangees.length; j++) {
        cellules.push(rangees[j].cells[1]);
    }
}
```

```
<table id="matable">
    <td> ... </td>
    ...
</table>
```

[https://www.w3schools.com/jsref/coll\\_table\\_tbodies.asp](https://www.w3schools.com/jsref/coll_table_tbodies.asp)

# HTML & navigateur



- querySelector / querySelectorAll etc..

- JQuery

```
var cellules = $('#matable td');
```

```
<table id="matable">  
  <td> ... </td>  
  ...  
</table>
```

- MAIS !...

- On récupère une collection d'objets Jquery
  - .get
  - Librairie à charger (lourd)



# HTML & navigateur

- querySelector / querySelectorAll etc..

- querySelectorAll

```
var cellules = document.querySelectorAll("#matable td");
for (var i = 0; i < cellules.length; i++) {
    // Faire quelque chose avec cellules.item(i) ou cellules[i]
}
```

- Pas de librairie
- Très rapide

```
<table id="matable">
    <td> ... </td>
    ...
</table>
```



# HTML & navigateur

- querySelector / querySelectorAll exemples.

```
var cellules = document.querySelectorAll("#matable>tbody>tr>td:nth-of-type(2)");
var premier = document.querySelector("p:first-child");
var paragraphes = document.querySelectorAll("p.intro, p.suite");
var paragraphe = document.querySelector("p.intro, p.suite");

var importants = document.querySelectorAll('.important');
for (var i = 0; i < importants.length; i++) {
    importants[i].style.backgroundColor = 'yellow';
}

var monbouton = document.querySelector('#afficherMessage');
monbouton.onclick = function() {
    document.querySelector('#message').style.display = 'block';
}
```

[https://developer.mozilla.org/fr/docs/Web/CSS/S%C3%A9lecteurs\\_CSS](https://developer.mozilla.org/fr/docs/Web/CSS/S%C3%A9lecteurs_CSS)



# HTML & navigateur

La pseudo-classe `:nth-of-type(an+b)` permet de cibler un élément qui possède  $an+b-1$  frères étant les mêmes éléments dans l'arbre du document et partageant le même élément parent et avec `n` un entier incrémenté à partir de 0.

```
/* Cible les éléments <div> si ceux-ci sont les */
/* 4e, 8e, 16e, 20e, etc. d'un élément parent */
div:nth-of-type(4n) {
    background-color: lime;
}
```



# HTML & navigateur

[https://developer.mozilla.org/fr/docs/Web/CSS/S%C3%A9lecteurs\\_CSS](https://developer.mozilla.org/fr/docs/Web/CSS/S%C3%A9lecteurs_CSS)

MDN web docs Technologies ▾ Guides et références ▾ Votre avis ▾

## Sélecteurs CSS

Aller à : [Les sélecteurs simples](#) [Les combinateurs](#) [Les pseudo-classes](#) [Les pseudo-éléments](#) [Spécificités](#)

Technologies web pour développeurs > [CSS](#) > Sélecteurs CSS

---

Sujets associés

- CSS
- Référence CSS
- CSS Selectors
  - ▼ Guides
    - Utiliser la pseudo-classe `:target` dans un sélecteur
  - ▼ Sélecteurs simples
    - Sélecteurs de type
    - Sélecteurs de classe
    - Sélecteurs d'ID
    - Sélecteurs universels
    - Sélecteurs d'attribut
  - ▼ Combinateurs

Les sélecteurs définissent les éléments sur lesquelles s'applique un ensemble de styles CSS.

---

## Les sélecteurs simples

**Les sélecteurs de type**  
Ce sélecteur simple permet de cibler les éléments qui correspondent au nom de l'élément.  
**Syntaxe :** `nomlement`  
**Exemple :** `input` permettra de cibler n'importe quel élément `<input>`.

**Les sélecteurs de classe**  
Ce sélecteur simple permet de cibler les éléments en fonction de la valeur attribut `class`.  
**Syntaxe :** `.nomclasse`  
**Exemple :** `.index` permettra de cibler n'importe quel élément qui possède l'attribut `class="index"` (vraisemblablement définie avec un attribut `class="index"`).



# HTML & navigateur

- Accès aux éléments

- DOM (chemins) : dépend du navigateur
- getElementById
- Variable implicite
- getElementsByTagName
  - getElementsByTagName (id = "" -> name= "")
- getElementsByClassName
- querySelector / querySelectorAll
- Xpath
- JQuery

(voir formulaires)



# HTML & navigateur

## •Formulaires

```
document.forms.f1
```

```
documents.forms[0]
```

```
document.forms["f1"]
```

```
<form name="f1">  
  <input name="un" value="1">  
  <input name="deux" value="2">  
</form>
```

name à éviter  
id ici...

```
document.forms.f1.elements[0]
```

```
<form>  
  <input type="radio" name="age" value="10">  
  <input type="radio" name="age" value="20">  
</form>
```

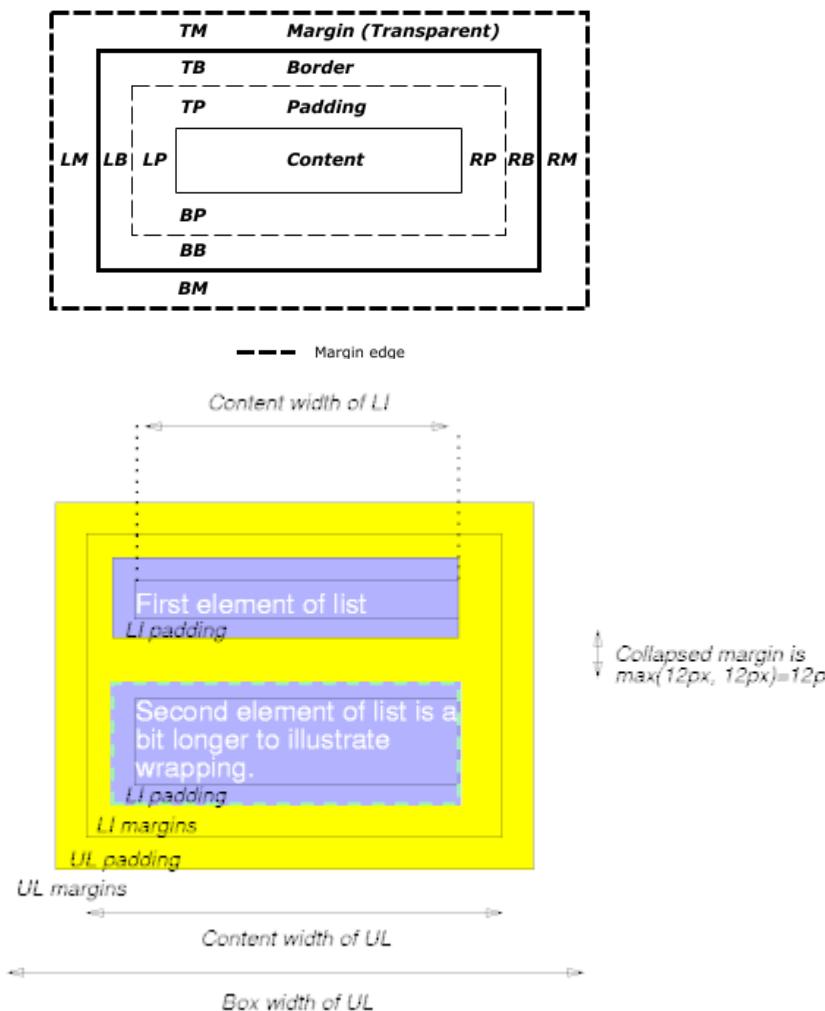


```
var form = document.forms[0]  
var elems = form.elements.age // tableau !!  
var e = elems[0]
```

# HTML & navigateur



## • Styles et CSS : rappels



```
<HTML>
<HEAD>
<TITLE>Examples of margins, padding, and borders</TITLE>
<STYLE type="text/css">
  UL {
    background: yellow;
    margin: 12px 12px 12px 12px;
    padding: 3px 3px 3px 3px;
    /* No borders set */
  }
  LI {
    color: white;          /* text color is white */
    background: blue;       /* Content, padding will be blue */
    margin: 12px 12px 12px 12px;
    padding: 12px 0px 12px 12px; /* Note 0px padding right */
    list-style: none        /* no glyphs before a list item */
    /* No borders set */
  }
  LI.withborder {
    border-style: dashed;
    border-width: medium;   /* sets border width on all sides */
    border-color: lime;
  }
</STYLE>
</HEAD>
<BODY>
<UL>
  <LI>First element of list
  <LI class="withborder">Second element of list is
    a bit longer to illustrate wrapping.
</UL>
</BODY>
</HTML>
```



# HTML & navigateur

- Styles et CSS : rappels

A voir : px, pt, em etc..

<http://www.w3.org/Style/Examples/007/units.fr.html>

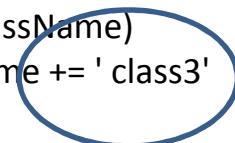
- class ou className

- Nom d'une catégorie (classe)
  - Catégorie de propriétés (style)

- Correspondance CSS -> JavaScript

- background-color -> backgroundColor
  - border-right-width -> borderRightWidth

```
<body class="class1 class2">  
    <script>  
        alert(document.body.className)  
        document.body.className += ' class3'  
    </script>  
</body>
```



Ajouter ou enlever une classe

element.style.width='100px'  
style="width:100px"

# HTML & navigateur



- Attention :

- Feuille de style CSS est appliquée
- .style rajoute (ou masque) la feuille de style
- .style ne permet pas de lire la feuille CSS
- Remettre .style à vide pour retrouver les réglages de la feuille de style

```
<style>
  body { margin: 10px }
</style>
<body>
  <script>
    alert(document.body.style.marginTop)
  </script>
</body>
```

Ne donne pas de bon résultat !  
Différences entre CSS et .style



# HTML & navigateur

- Attention :

```
<style> body { margin: 10px } </style>
<body>
<script>
  document.body.style.margin = '20px'
  alert(document.body.style.marginTop) // 20px
</script>
</body>
```

Ici style.margin ok

Pour remettre la valeur de css

document.body.style.marginTop=""



# HTML & navigateur

- Calculer le style après css + application de .style
- getComputedStyle
  - DOM2

```
<style> body { margin: 10px } </style>
<body>
  <script>
    var computedStyle = getComputedStyle(document.body, null)
    alert(computedStyle.marginTop) // ok (même avec .style)
  </script>
</body>
```

- Attention à IE !
  - currentStyle
  - Problème de conversion



# HTML & navigateur

- Event & déclenchement d'actions

- Interaction entre la page HTML et l'interpréteur JS
- 3 sortes d'evt
  - Déplacement souris, click bouton etc
  - Fenêtre déplacée ou changement de taille
  - Ajax (chargement de page etc)
- Lien entre l'evt et une fonction js

```
<input id="bouton1" value="cliquer ici" onclick="alert('ok');" type="button"/>
```

- Généralement attribut onNOMEVT

- onclick
- onload
- Etc

Manière de faire à éviter

Mélange code HTML et JS

Maintenance difficile

Ok pour evt et situations simples

A éviter dev sérieux...

- Remarque

- Javascript : maj et min important
- HTML : aucune importance
- HTML minuscules préférées

```
<button onclick="alert(this.innerHTML)">le soleil brille </button>
```



# HTML & navigateur

- Création d'une méthode javascript pour l'objet DOM
- Javascript : simplement rajouter une méthode
- Nommage important

```
<input id="myElement" type="button" value="click ici"/>
```

```
<script>
document.getElementById('myElement').onclick = function() {
    alert('ok')
}
</script>
```

- En minuscule (attention)
- onevent
- Possibilité de mettre une fonction déjà déclarée
  - document.getElementById('id').onclick = maFonction
  - document.getElementById('id').onclick = maFonction()
  - Différences ?



# Navigateur

- Cookies

- ↳ 4k maximum
- ↳ tracer activité
- ↳ chargement lent (chargé à chaque page)
- ↳ envoyé par les requêtes (post)

- Web Storage

- ↳ plus **simple** que les cookies pour conserver de l'information entre sessions
- ↳ accès si et quand nécessaire
- ↳ dans HTML5 ?
- ↳ > 5Mo

# Navigateur

- Cookies & Javascript

- Cookie : trace que peut laisser un Site Web sur le navigateur
- Réside sur le client (cookies.txt - netscape -/ rep cookies - ie -)
- Contient :
  - identification du serveur
  - durée de validité
  - infos spécifiques du serveur
- ‘Mémoire’ d ’une connexion

# Navigateur

- Cookies & Javascript

- exemple : Ecriture

```
<FORM NAME="nomForm">  
    Entrer votre nom :  
    <INPUT TYPE=TEXT NAME="nom"  
          onBlur = "ecrireCookie()" >  
</FORM>
```

Entrez votre nom :

# Navigateur

- Cookies & Javascript

```
dateExpiration = new Date();  
dateExpiration.setMonth(dateExpiration.getMonth()+6)
```

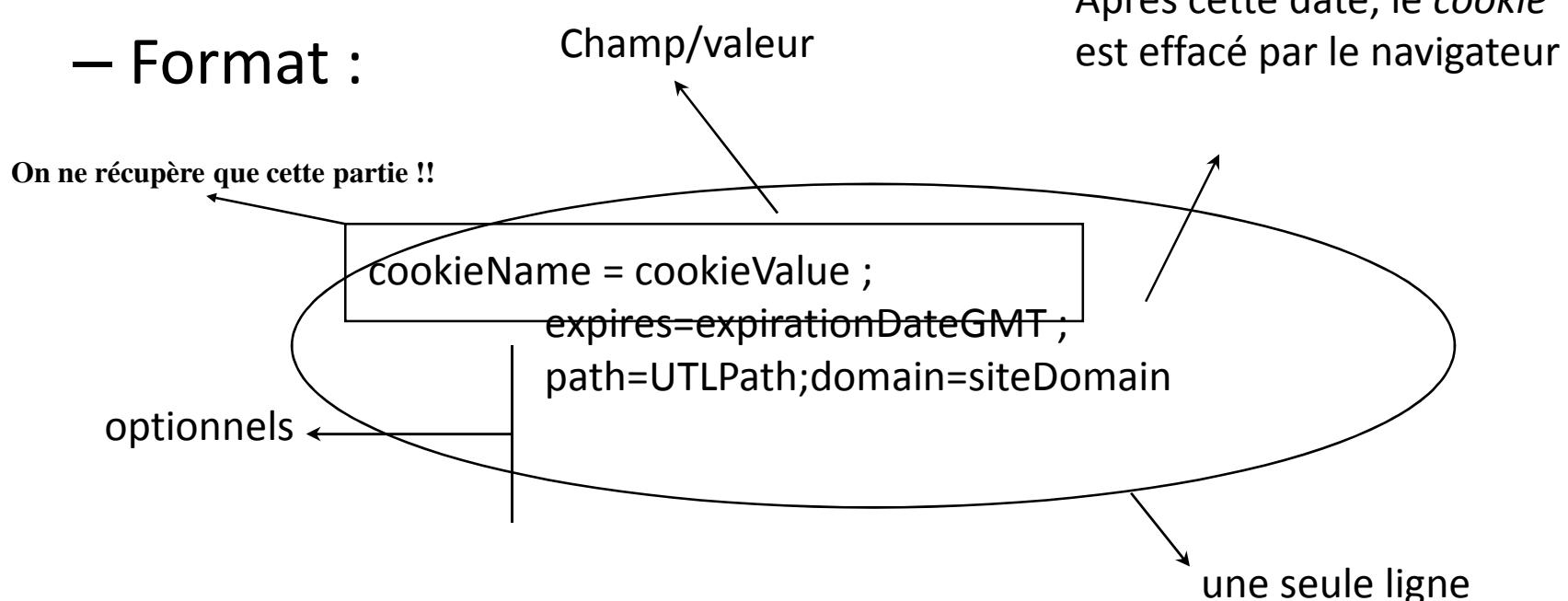
expire dans 6 mois

```
function ecrireCookie() {  
    nomUtilisateur = document.monForm.nom.value  
    document.cookie = "userName=" + nomUtilisateur  
        + ";expires=" + dateExpiration.toGMTString();  
}
```

sur une seule ligne

# Navigateur

- Cookies & Javascript



# Navigateur

- Cookies & Javascript
  - exemple : Lecture

```
<SCRIPT ...>
```

```
dateExpiration = new Date()  
dateExpiration.setMonth(dateExpiration.getMonth()+6)
```

```
nomUtilisateur = "";  
if (document.cookie != "") {  
    nomUtilisateur = document.cookie.split("= ")[1]  
}  
function ecrireCookie(...  
...}
```

```
</SCRIPT>
```



Découpage d'une chaîne  
en tableau ( séparateur '=' )

# Navigateur

- Cookies & Javascript

```
<BODY onLoad=' document.monForm.nom.value = nomUtilisateur ' >  
  
<FORM NAME=' nom '>  
    ...  
</FORM>
```

Entrez votre nom :

Le champ est initialisé au nom du cookie, sinon à vide  
(pendant 6 mois)

# Navigateur

- Cookies & Javascript
  - Multiples cookies pour une page :

```
"cookieName1=cookieValue1;expires1=expirationDateGMT1;path1=sitePath1;  
domain1=siteDomain1" ;_  
"cookieName2=cookieValue2;expires2=expirationDateGMT2;path2=sitePath2;  
domain2=siteDomain2" ;_  
"cookieName3=cookieValue3;expires3=expirationDateGMT3;path3=sitePath3;  
domain3=siteDomain3"
```
  - ajouter / modifier par `document.cookie = ...`

# Navigateur

- Cookies & Javascript

- Gestion de cookies multiples :

```
function cookieVal(nomCookie) {  
    tabCookies = document.cookie.split(";_")  
    for(i=0; i < tabCookies.length; i++) {  
        if (nomCookie == tabCookies[i].split("=")[0]) {  
            return tabCookies[i].split("=")[1]  
        }  
    }  
    return 0  
}
```

‘ ; ’ + un espace

Utilisation :

cookieVal(' nomUtilisateur ')

# Navigateur

- Cookies & Javascript
  - Effacer un cookie :
    - changer sa date d'expiration
    - date actuelle - 1 jour par exemple
  - A faire en exercice :
    - un script qui affiche dans une page HTML les nom, valeurs et date d'expiration des cookies pour la page courante.

# Navigateur

- Cookies & Javascript
  - Applications : (à savoir faire)
    - Compteur de visite de page pour un utilisateur
      - ‘vous êtes passé 13 fois nous voir’
    - Indiquer à un visiteur ce qui est nouveau sur un site depuis sa dernière visite
      - création dynamique d ’une page ‘nouveau - pour vous’
    - Commerce électronique

Développeur Javascript [f](#) [X](#) [+](#)

[←](#) [→](#) [C](#) [H](#) <https://www.lemondeinformatique.fr/actualites/lire-developpeur-javascript-full-stack> [...](#) [...](#) [...](#) [...](#) [...](#)

Les plus visités [ipcny](#) [qc](#) [Cluster](#) [Spark](#) [Lua](#) [Inv](#) [Ent](#) [NoteBook](#) [Gogs](#) [Sam](#) [Nin](#) [M](#)

# Le développement full stack en Ile-de-France

Véronique Arène , publié le 19 Février 2020



Avec un salaire de démarrage compris entre 39 et 45 000 euros brut par an en Ile-de-France, le développeur JavaScript full stack coiffe au poteau les autres spécialistes du code, révèle une étude réalisée par Urban Linker. Parmi les autres technologies prisées, on trouve l'incontournable Python, ainsi que PHP majoritairement sur les frameworks Symfony et Laravel.



## **globale du client**

- 1** Vers une vision globale en temps réel du client
- 2** Adobe, son cœur de métier : le marketing automation
- 3** ...

[LIRE LE DOSSIER >](#)

startups à Paris ne déroge pas à cette tendance générale, avec énormément de recherches de professionnels de JavaScript, qu'ils soient orientés front-end, back-end (avec Node.js donc) ou fullstack. Du fait d'une demande très forte de la part des startups, les profils fullstack JS (soit 39 à 45 K€ de rémunération annuelle pour un junior et jusqu'à 70 K€ voire plus pour un Lead fullstack en Ile-de-France) sont en effet parmi les plus pénuriques du marché des nouvelles technologies, souligne Urban Linker. Le cabinet note tout de même qu'une expertise purement front à un niveau senior ou Lead développeur, est valorisée de la même manière que celle des profils fullstack. Autre fait notable: la différence de salaires entre les développeurs front et ceux qui codent uniquement en Node.js semble très légère, mais elle n'est pas insignifiante en particulier pour les profils juniors. L'étude souligne que lorsque les profils Node.js se seniorisent, ils ne se cantonnent pas à la partie serveur d'une application web et deviennent fullstack. Assez logiquement, la maîtrise de Node.js en plus d'un framework front favorise des salaires plus élevés que la moyenne.

# Navigateur



- Local storage

localStorage

Conservé d'une session à l'autre

```
localStorage.setItem('ma couleur','bleu');
```

```
var couleur = localStorage.getItem('ma couleur'); // bleu
```

```
localStorage.removeItem('ma couleur');
```

```
var taste = localStorage.getItem('ma couleur'); // null
```



# Navigateur

sessionStorage

- Session storage

non Conservé d'une session à l'autre

```
sessionStorage.setItem('ma couleur','bleu');
```

```
var couleur = sessionStorage.getItem('ma couleur'); // bleu
```

```
sessionStorage.removeItem('ma couleur');
```

```
var taste = sessionStorage.getItem('ma couleur'); // null
```

Effacé à la fermeture Navigateur /  
onglets site

# Navigateur



localStorage

## • Remarques

```
localStorage["ma couleur"] = "bleu"
```

```
// localStorage.maCouleur
```

```
localStorage.clear()
```

```
if (window.localStorage)
{
...
if (localStorage) { ... }
```

Seulement des chaînes de caractère



# Navigateur

localStorage

## • Utilisez JSON

```
var voiture = {};  
voiture.roues = 4;  
voiture.portes = 2;  
voiture.son = 'vroom';  
voiture.nom = 'Flash McQueen';
```

```
console.log( voiture );
```

```
localStorage.setItem( 'voiture', JSON.stringify(voiture) );
```

```
console.log( JSON.parse( localStorage.getItem( 'voiture' ) )  
);
```



# Navigateur

localStorage

- Utilisations

- Cache local

- Requête (par exemple Ajax)

- Stockage local du résultat (cache)

- Rechargement de la page

- Cache rechargeé ou TimeOut page

- Contenu d'un formulaire

- retours en arrière

- Jeux

- Partagés entre plusieurs onglets !

# Navigateur



sessionStorage

## •Formulaire

```
<textarea id="nom" name="nom"
    onchange="sessionStorage.nom=this.value">
</textarea>

<script>
// est-ce que sessionStorage est supporté ?
if(sessionStorage) { // typeof sessionStorage!= 'undefined'
    if( 'nom' in sessionStorage) {
        document.getElementById('nom').value = sessionStorage.nom;
    }
} else {
    // problème à gérer (ou pas : cookies ?)
}
</script>
```

# Navigateur



localStorage

```
<script>  
if(localStorage) {
```

## • Compteur de visites

```
    var nbvisites = localStorage.getItem('visites');
```

```
    if(nbvisites!=null) {
```

```
        nbvisites = parseInt(nbvisites, 10);  
    } else {  
        nbvisites = 1;  
    }
```

```
    nbvisites++;
```

```
    localStorage.setItem('visites',nbvisites);
```

```
    document.getElementById('visites').innerHTML = nbvisites;  
} else {  
    // alert("localStorage n'est pas supporté");  
}
```

```
</script>
```

<p>Vous avez vu cette page  
<span id="visites"></span> fois</p>

diff cookies : info non  
envoyée  
au serveur (stats)



# Navigateur

<html>

```
<head>
  <title>List of Plug-Ins</title>
</head>
```

<body>
 <table border="1">

```
    <tr>
      <th>Plug-in Name</th>
      <th>Filename</th>
      <th>Description</th>
    </tr>
```

```
    <script language="JavaScript" type="text/javascript">
      for (i=0; i<navigator.plugins.length; i++) {
        document.write("<tr><td>");
        document.write(navigator.plugins[i].name);
        document.write("</td><td>");
        document.write(navigator.plugins[i].filename);
        document.write("</td><td>");
        document.write(navigator.plugins[i].description);
        document.write("</td></tr>");
      }
    </script>
```

</table>

</body>
</html>

## • plugins

Plug-in Name	Filename	Description
Adobe Acrobat	appdf32.dll	Adobe PDF Plug-In For Firefox and Netscape 11.0.12
Adobe Acrobat	nppdf32.dll	Adobe PDF Plug-In For Firefox and Netscape 11.0.12
Google Earth Plugin	npgeplugin.dll	GEPlugin
Google Update	npGoogleUpdate3.dll	Google Update
Java Deployment Toolkit 8.0.600.27	npdeployJava1.dll	NPRuntime Script Plug-in Library for Java(TM) Deploy
Java(TM) Platform SE 8 U60	npjp2.dll	Next Generation Java Plug-in 11.60.2 for Mozilla browsers
Microsoft Office 2013	NPSPWRAPI.DLL	The plugin allows you to have a better experience with Microsoft SharePoint
QuickTime Plug-in 6.5.1	npqtplugin.dll	The QuickTime Plugin allows you to view a wide variety of multimedia content in Web pages. For more information, visit the <a href="#">QuickTime</a> Web site.
QuickTime Plug-in 6.5.1	npqtplugin7.dll	The QuickTime Plugin allows you to view a wide variety of multimedia content in Web pages. For more information, visit the <a href="#">QuickTime</a> Web site.
QuickTime Plug-in 6.5.1	npqtplugin5.dll	The QuickTime Plugin allows you to view a wide variety of multimedia content in Web pages. For more information, visit the <a href="#">QuickTime</a> Web site.
QuickTime Plug-in 6.5.1	npqtplugin2.dll	The QuickTime Plugin allows you to view a wide variety of multimedia content in Web pages. For more information, visit the <a href="#">QuickTime</a> Web site.
QuickTime Plug-in 6.5.1	npqtplugin3.dll	The QuickTime Plugin allows you to view a wide variety of multimedia content in Web pages. For more information, visit the <a href="#">QuickTime</a> Web site.
QuickTime Plug-in 6.5.1	npqtplugin6.dll	The QuickTime Plugin allows you to view a wide variety of multimedia content in Web pages. For more information, visit the <a href="#">QuickTime</a> Web site.
QuickTime Plug-in 6.5.1	npqtplugin4.dll	The QuickTime Plugin allows you to view a wide variety of multimedia content in Web pages. For more information, visit the <a href="#">QuickTime</a> Web site.
Shockwave Flash	NPSWF32_18_0_0_232.dll	Shockwave Flash 18.0 r0
VLC Web Plugin	npvlc.dll	VLC media player Web Plugin 2.1.3
VirtualGeoGP	npQtAPI3DPlugin.dll	VirtualGeoGP Plugin v3.1.0.2040
WacomTabletPlugin	npWacomTabletPlugin.dll	Plugins for Wacom tablets

# Canvas



## • Contexte graphique <canvas>

- Accès via getContext('2d')
  - Webgl
  - Webgl2

```
<canvas id="monCanvas" width="500" height="200"></canvas>
```

```
<script>
```

```
var canvas = document.getElementById('monCanvas');
```

```
var context = canvas.getContext('2d');
```

```
context.font = '40pt Calibri';
```

```
context.fillStyle = 'blue';
```

```
context.fillText('Hey Salut', 150, 100);
```

```
</script>
```

```
context.beginPath();
```

```
context.moveTo(100, 150);
```

```
context.lineTo(450, 50);
```

```
context.stroke();
```

- Voir la documentation *canvas*



## HTML Reference

- [HTML by Alphabet](#)
- [HTML by Category](#)
- [HTML Global Attributes](#)
- [HTML Events](#)
- [HTML Canvas](#)
- [HTML Audio/Video](#)
- [HTML Doctypes](#)
- [HTML Colornames](#)
- [HTML Colorgroups](#)
- [HTML Colorpicker](#)
- [HTML Colormixer](#)
- [HTML Character Sets](#)
- [HTML URL Encode](#)
- [HTML Language Codes](#)
- [HTML Country Codes](#)
- [HTTP Messages](#)
- [HTTP Methods](#)
- [PX to EM Converter](#)
- [Keyboard Shortcuts](#)

2:14  
October  
<!-->  
<!DOCTYPE

## HTML canvas rotate() Method

[HTML Canvas Reference](#)

### Example

Rotate the rectangle 20 degrees:



JavaScript:

```
var c=document.getElementById("myCanvas");
var ctx=c.getContext("2d");
ctx.rotate(20*Math.PI/180);
ctx.fillRect(50,20,100,50);
```

[Try it Yourself >](#)

Edit This Code:

[See Result »](#)

```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

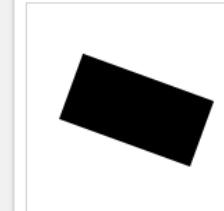
<script>

var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.rotate(20 * Math.PI / 180);
ctx.fillRect(50, 20, 100, 50);

</script>

</body>
</html>
```

Result:



canvas rotate

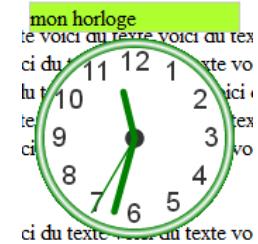
Web Shopping Vidéos Images Actualités Plus ▾ OI

Environ 965 000 résultats (0,50 secondes)

### HTML canvas rotate() Method - W3Schools

[www.w3schools.com/tags/canvas\\_rotate.asp](#) ▾ Traduire cette page  
Internet Explorer 9, Firefox, Opera, Chrome, and Safari support the rotate() method.  
Note: Internet Explorer 8 and earlier versions, do not support the <canvas> ...

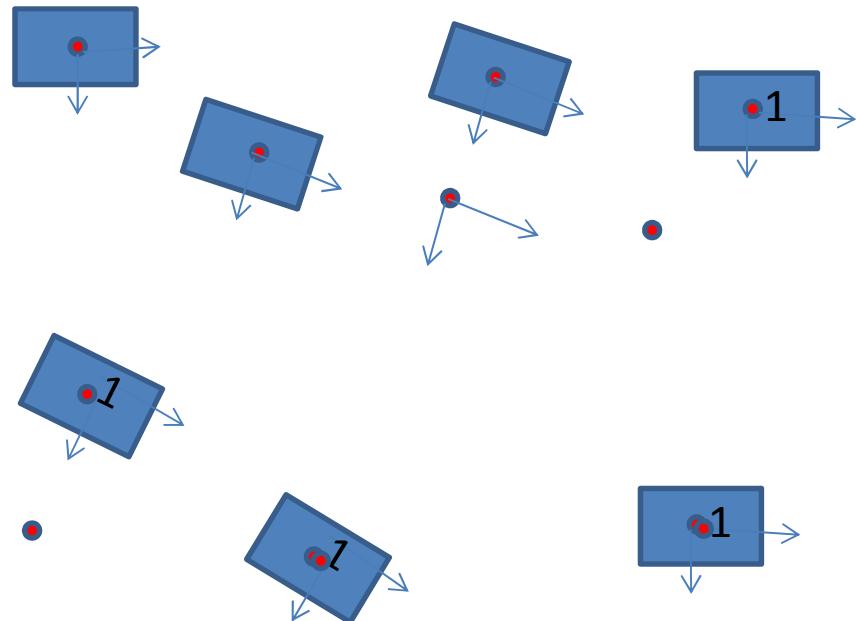
# Canvas



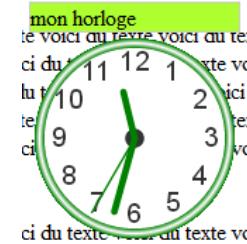
Voir le TD Candy Crush

- rotate et translate : matrices de transformation affine

```
var ang;  
var num;  
ctx.font = radius*0.28 + "px arial";  
ctx.textBaseline="middle";  
ctx.textAlign="center";  
for(num = 1; num < 13; num++){  
    ang = num * Math.PI / 6;  
    ctx.rotate(ang);  
    ctx.translate(0, -radius*0.80);  
    ctx.rotate(-ang);  
    ctx.fillText(num.toString(), 0, 0);  
    ctx.rotate(ang);  
    ctx.translate(0, radius*0.80);  
    ctx.rotate(-ang);  
}
```



# Canvas



CF cours Opengl avec les matrices de transformation **push** et **pop**

```
context.save();
// save state 1
context.translate(canvas.width);

context.save();
// save state 2
context.rotate(Math.PI / 4);

context.save();
// save state 3
context.scale(2, 2);

context.fillStyle = 'blue';
context.fillRect(rectWidth / - rectWidth, rectHeight);

context.restore();
// restore state 3
context.fillStyle = 'red';
context.fillRect(rectWidth / - rectWidth, rectHeight);

context.restore();
// restore state 2
context.fillStyle = 'yellow';
context.fillRect(rectWidth / - rectWidth, rectHeight);

context.restore();
// restore state 1
context.fillStyle = 'green';
context.fillRect(rectWidth / - rectWidth, rectHeight);
```

# Javascript

```
var x = 1;
```

```
if(x === "1") {
    console.log("Egalité, 1 partout");
}
```

```
var tab = [0]
console.log( tab == tab) // true
console.log( tab == !tab) // true
```

```
tab = [null, undefined, []]
console.log( tab == ',' ) // true
```

```
Console.log(Math.min() < Math.max()) //
false
```

```
> Math.min()
Infinity
> Math.max()
-Infinity
> Math.min()<Math.max()
false
>
```

## HORROR SHOW



```
var x = 1;
```

```
if(x === "1") {
    console.log("Egalité, 1 partout");
}
```

```
16 == [16]      // true
16 == [1,6]     // false
"1,6" == [1,6]  // true
```

```
var a = "1"
var b = 2
var c = a + b // c = "12"
```

# Javascript HORROR SHOW



```
alert(([![]+[]][+[]]+([![]+[]][+!+[]])+([![]]+[])[+!+[]+[+[]]]+([![]+[]][!+[]+!+[]]));
```



```
console.log(([![]+[]][!+[]+!+[]])) // 1  
console.log([!+[]+!+[]]) // [2]  
console.log([!+[]]) // true
```

# Javascript

HORROR  
SHOW



```
function maFonction() {  
    var bar = 10;  
    //  
    // [...code...]  
    //  
    baz = 20; // "bar" ("baz") est (définie comme) une variable globale  
}
```

variables globales :  
**Interdiction**

```
var calculateTopTenCosts = function()  
{  
    var sum = 0;  
  
    for(top = 0; top < 10; top++) {  
        sum += getRankCosts(top);  
    }  
  
    return sum;  
}
```

renvoie toujours 0 dans  
un  
navigateur, marche dans  
node :

Pourquoi ?

# Javascript

HORROR  
SHOW



```
function test() {  
    return  
    {  
        bar : "test"  
    };  
}
```

```
function test(a)  
{  
    if(a === 0) {  
        var c = 20;  
    }  
    return c;  
}
```

```
[5, 12, 9, 2, 18, 1, 25].sort(); // [1, 12, 18, 2, 25, 5, 9]
```

```
[5, 12, 9, 2, 18, 1, 25].sort(function(a, b){  
    return a - b;  
});
```

# Javascript

HORROR  
SHOW



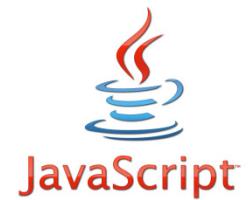
```
> var a = "test"
undefined
> typeof a
'string'
> a instanceof String
false
> a = new String("test")
[String: 'test']
> typeof a
'object'
> a instanceof String
true
>
```

```
var a = "test"
var b = typeof a           // 'String'
var c = a instanceof String // false
```

```
var a = new String("test")
var b = typeof a           // 'object'
var c = a instanceof String // true
```

# JSLint

- [Read the instructions.](#)
- [Enjoy \*The Good Parts\*.](#)



jslint.com

## Source

```
function test() {
    return
    {
        bar : "test"
    };
}
```

[JSLint](#)

[clear](#)

## Warnings

**JSLint was unable to finish.**

Expected 'use strict' before 'return'.

*line 1 column 4*

return

Unexpected trailing space.

*line 1 column 10*

return

Expected ';' and instead saw '{'.

*line 2 column 8*

{

## Function Report

global test

test()

*line 0*

embre

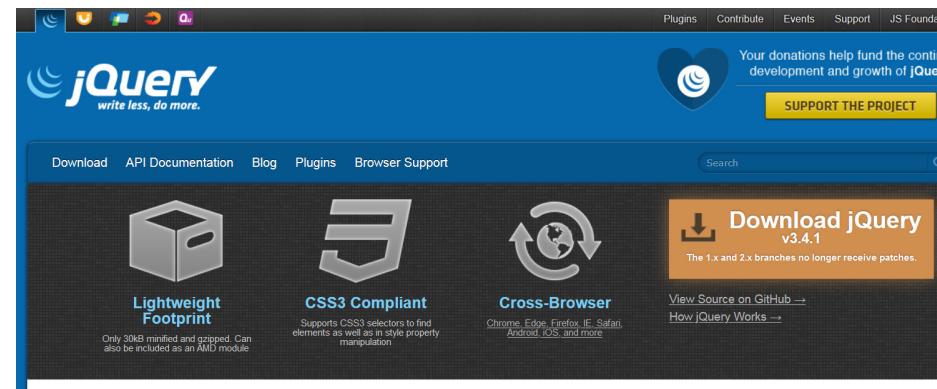
[JSLint edition 2015-08-22](#)



Gildas Ménier  
[gildas.menier@univ-ubs.fr](mailto:gildas.menier@univ-ubs.fr)

# JQuery : librairie Javascript (DomS)

- Librairie openSource
  - John Resig
  - 14Ko / 90 Ko
- DomScript
  - IE6, FF2+, Safari 3, Opera 9, Chrome
  - Google, Dell etc..
- Widgets
- Effets
- Javascript
- Dom
- CSS (selecteurs)
- Ajax
- <> Mootools
- <> Dojo, Kendo, prototype etc..



# JQuery : principes

- IBM, Amazon, Dell, Twitter, Bank of America, Netflix, Nokia (Web runtime) etc...
- Ce n'est pas une librairie Javascript (ou pas que...)
- Accès CSS
  - Cf XPath
  - XHTML
- <script type="text/javascript"  
src="https://code.jquery.com/jquery-**3.4.1**.min.js" ></script>

32ko

```
/*! jQuery v@3.2.1 jquery.com | jquery.org/license */
(function(a,b){function G(a){var b=F[a]={};return p.each(a.split(s),function(a,c){b[c]=!0}),b}function
J(a,c,d){if(d==b&&a.nodeType==1){var e="da.....
```

# JQuery : principes

- <script type="text/javascript" src="https://code.jquery.com/jquery-3.2.1.js" > </script>

```
/*
 * jQuery JavaScript Library v3.2.1
 * https://jquery.com/
 *
 * Includes Sizzle.js
 * https://sizzlejs.com/
 *
 * Copyright JS Foundation and other contributors
 * Released under the MIT license
 * https://jquery.org/license
 *
 * Date: 2017-03-20T18:59Z
 */
(function( global, factory ) {

    "use strict";

    if ( typeof module === "object" && typeof modul
        // For CommonJS and CommonJS-like envir
        // is present, execute the factory and
        // For environments that do not have a
        // (such as Node.js), expose a factory
        // for the global object
        .factory(factory, !1)
    );
})()
```

32ko

# JQuery : principes

- <script type="text/javascript" src="https://code.jquery.com/jquery-3.2.1.min.js" ></script>

```
/*! jQuery v3.2.1 | (c) JS Foundation and other contributors | jquery.org/license */
!function(a,b){"use strict";"object"==typeof module&&"object"==typeof module.exports?i
window with a document");return b(a):b(a)>("undefined"!=typeof window?window:this,fu
c=[],d=a.document,e=Object.getPrototypeOf,f=c.slice,g=c.concat,h=c.push,i=c.indexOf,j=
var c=b.createElement("script");c.text=a,b.head.appendChild(c).parentNode.removeChild
\xA0]+$/g,t=/^-ms-/,u=/-([a-z])/g,v=function(a,b){return b.toUpperCase()};r.fn=r.proto
{return null==a?f.call(this):a<0?this[a+this.length]:this[a]},pushStack:function(a){v
r.each(this,a)},map:function(a){return this.pushStack(r.map(this,function(b,c){return
this.pushStack(f.apply(this,arguments))}),first:function(){return this.eq(0)},last:f
this.pushStack(c>=0&&c<b?[this[c]]:[]),end:function(){return this.prevObject||this.co
a,b,c,d,e,f,g=arguments[0]||{},h=1,i=arguments.length,j=!1;for("boolean"==typeof g&&(
h++)if(null!=(a=arguments[h]))for(b in a)c=a[b].d=a[b].o!==d&&(i&&d&&(r.isPlainObject
```

Sans Ajax (26 ko)

# JQuery : principes

- <script type="text/javascript"  
src="https://code.jquery.com/jquery-3.2.1.slim.js" ></script>

```
/*
 * jQuery JavaScript Library v3.2.1 -ajax,-ajax/jsonp,-ajax/load,-ajax/parseXML,-ajax/script,-
ajax/var/location,-ajax/var/nonce,-ajax/var/rquery,-ajax/xhr,-manipulation/_evalUrl,-
event/ajax,-effects,-effects/Tween,-effects/animatedSelector
 * https://jquery.com/
 *
 * Includes Sizzle.js
 * https://sizzlejs.com/
 *
 * Copyright JS Foundation and other contributors
 * Released under the MIT license
 * https://jquery.org/license
 *
 * Date: 2017-03-20T19:00Z
 */
(function( global, factory ) {

    "use strict";
```

# JQuery : principes

- Utilisation d'un CDN **Content Delivery Network**
  - Réseau de serveurs qui délivrent le même contenu
  - Accès *rapide* et toujours disponible
  - Jquery : librairie utilisée par ENORMEMENT de pages
  - Stockage local mais références (mises à jour)
  - <https://code.jquery.com/>
- 

## jQuery CDN - Latest Stable Version

Powered by  StackPath

### jQuery Core

Showing the latest stable release in each major branch. [See all versions of jQuery Core.](#)

click

### jQuery 3.x

- jQuery Core 3.3.1 - [uncompressed](#), [minified](#), [slim](#), [slim minified](#)

### jQuery 2.x

- jQuery Core 2.2.4 - [uncompressed](#), [minified](#)

### jQuery 1.x

# JQuery : principes

Code Integration X

```
<script
  src="https://code.jquery.com/jquery-3.4.1.js"
  integrity="sha256-WpOohJOqMqqyKL9FccASB900KwACQJpFTUBLTYOVvVU="
  crossorigin="anonymous"></script>
```

⊕

The `integrity` and `crossorigin` attributes are used for [Subresource Integrity \(SRI\) checking](#). This allows browsers to ensure that resources hosted on third-party servers have not been tampered with. Use of SRI is recommended as a best-practice, whenever libraries are loaded from a third-party source. Read more at [srihash.org](#)



À copier /coller dans le code js pour avoir accès à jquery  
UTILISEZ LE CDN Jquery  
<https://code.jquery.com/>

# JQuery : principles

## jQuery Core

Showing the latest stable release in each major branch. [See all versions of jQuery Core.](#)

### Code Integration

```
<script
  src="https://code.jquery.com/jquery-3.3.1.js"
  integrity="sha256-2Kok7MhOyXpgUVvAk/HJ2jigOSYS2auK4Pfzbm7uH60="
  crossorigin="anonymous"></script>
```



The `integrity` and `crossorigin` attributes are used for [Subresource Integrity \(SRI\) checking](#). This allows browsers to ensure that resources hosted on third-party servers have not been tampered with. Use of SRI is recommended as a best-practice, whenever libraries are loaded from a third-party source. Read more at [srihash.org](#)

[jQuery Versions](#)

### TABLE OF CONTENTS

- 1. **Introduction**
  - 1.1 Goals
  - 1.2 Use Cases/Examples
    - 1.2.1 Resource Integrity
- 2. **Conformance**
  - 2.1 Key Concepts and Terminology
  - 2.2 Grammatical Concepts
- 3. **Framework**

Attention !

W3C Recommendation

## Subresource Integrity

W3C Recommendation 23 June 2016

This version:

<http://www.w3.org/TR/2016/REC-SRI-20160623/>

Latest published version:

<http://www.w3.org/TR/SRI/>

Latest editor's draft:

<https://w3c.github.io/webappsec-subresource-integrity/>

Implementation report:

# JQuery : principles

## Using Subresource Integrity

You use the Subresource Integrity feature by specifying a base64-encoded cryptographic hash of a resource (file) you're telling the browser to fetch, in the value of the **integrity** attribute of any `<script>` or `<link>` element.

An **integrity** value begins with at least one string, with each string including a prefix indicating a particular hash algorithm (currently the allowed prefixes are `sha256`, `sha384`, and `sha512`), followed by a dash, and ending with the actual base64-encoded hash.

- An **integrity** value may contain multiple hashes separated by whitespace. A resource will be loaded if it matches one of those hashes.

Example **integrity** string with base64-encoded sha384 hash:

```
1 | sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/uxy9rx7HNQ1GYl1kPzQho1wx4JwY8wC
```

- An **integrity** value's "hash" part is, strictly speaking, a **cryptographic digest** formed by applying a particular hash function to some input (for example, a script or stylesheet file). But it's common to use the shorthand **hash** to mean

# JQuery : principes

Autres CDN :

<https://developers.google.com/speed/libraries/#jquery>

jQuery

**3.x snippet:**

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
```

[https://docs.microsoft.com/en-us/aspnet/ajax/cdn/overview#jQuery\\_Releases\\_on\\_the\\_CDN\\_0](https://docs.microsoft.com/en-us/aspnet/ajax/cdn/overview#jQuery_Releases_on_the_CDN_0)

## jQuery Releases on the CDN

The following releases of jQuery are hosted on the CDN:

### jQuery version 3.3.1

- <http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.js>
- <http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.min.js>
- <http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.min.map>
- <http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.slim.js>
- <http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.slim.min.js>
- <http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.slim.min.map>

# JQuery : principes

Plugins : <https://plugins.jquery.com/tag/ui/>

 Popular Tags

- ui (542)
- jquery (482)
- form (285)
- animation (273)
- input (252)
- image (210)
- responsive (184)
- slider (172)
- ajax (154)
- scroll (140)

## Tagged: ui

**Chosen**

Chosen is a JavaScript plugin that makes long, unwieldy select boxes much more user-friendly. It is currently available in both jQuery and Prototype flavors.

**jQuery File Upload – jQuery UI version**

File Upload widget with multiple file selection, drag&drop support, progress bars and preview images for jQuery. Supports cross-domain, chunked and resumable file uploads and client-side image resizing. Works with any server-side platform (PHP, Python, Ruby on Rails, Java, Node.js, Go etc.) that supports standard HTML form file uploads.

**jQuery UI Slide Effect**

Slides an element in and out of the viewport.

**jQuery UI Puff Effect**

# JQuery : principes

- Javascript :

```
var checkedvalue ;
var elements = document.getElementsByTagName('input');
for( var n=0; n <elements.length; n++) {
    if (      elements[n].type=='radio' &&
        elements[n].name=='groupe' &&
        elements[n].checked ) {
            checkedValue = elements[n].value
        }
    }
```

# JQuery : principes

- Concision / simplicité :

```
var checkedValue= $('[name="groupe"]:checked').val() ;
```

# JQuery : principes

- Appels fonctionnels (chainage)

```
$("div.test").add("p.quote").addClass("blue").slideDown("slow");
```

```
$.each([1,2,3], function() { document.write(this + 1); });
```

## Code typique JQuery

```
$(function() {  
    $(".load_page_on_click").click(function() {  
        var email = $("input#email").val();  
        $.ajax({  
            async: "true",  
            type: "GET",  
            url: "mapage.php",  
            data: "email=" + encodeURIComponent(email) + "&action=get_email",  
            error: function(errorData) {  
                $("#error").html(errorData);  
            },  
            success: function(data) {  
                $("#container").html(data); $("#error").append("Contenu chargé");  
            }  
        }); // $.ajax  
    }); // $(".load_page_on_click").click  
}); // $(function
```

The screenshot shows a Mozilla Firefox browser window with the following details:

- Title Bar:** jquery - Mozilla Firefox
- Menu Bar:** Fichier Historique Marque-pages Outils ?
- Toolbar:** Back Home Stop Address bar: http://jquery.developpeur-web2.com/documentation.php
- Sidebar:** Infos Cherche Fac Sites Prog
- Page Content:**
  - Header:** JQuery Framework Javascript, AJAX
  - Navigation:** ACCUEIL ACTUALITÉS DÉMONSTRATION DOCUMENTATION TÉLÉCHARGEMENT LIENS UTILES
  - Breadcrumbs:** Accueil > Documentation
  - Left Sidebar (Sections du manuel):**
    - FONCTIONS DE BASE
    - SELECTEURS**
    - ATTRIBUTS
    - FONCTIONS DE PARCOURS
    - MANIPULATION
    - CSS
    - ÉVÈNEMENTS
    - EFFETS
    - AJAX
    - FONCTIONS UTILES
  - Main Content Area (DOCUMENTATION DU FRAMEWORK JQUERY):**

Ces pages de documentation sont une traduction libre de la documentation officielle : <http://docs.jquery.com>. Il est fort probable que cette traduction contienne des lacunes, n'hésitez pas à m'en faire part: [me citer](#).

Cette traduction prends en compte les dernières modifications de la version 1.3.1 de JQuery, hormis cours de mise à jour.

# JQuery : principes

- Principes :
  - Utilisation des sélecteurs CSS
    - XPath
- Exemple : p a
  - Les balises <a> *filles* des balises <p>
  - Pour identifier ces balises
    - `jQuery("p a")`
    - ...ou bien `$( "p a" )`
  - Renvoie un objet qui contient l'ensemble des balises DOM (seulement les 'a' filles de 'p')
  - Propriétés spéciales (itération etc..)

# JQuery : principes

- Principes :
  - *Wrapper* jQuery
    - Contient une collection d'éléments
    - Méthodes de transformation
    - Méthodes de filtrage de ces éléments
    - Itérations, tests etc...

```
var checkedValue= $('[name="groupe"]:checked').val() ;
```

---

## Sélection d'un ensemble de balises

# JQuery : principes

- Cascade

```
$("div.steackhache").hide();
```

On cache chaque div de classe steackhache

- À chaque application, l'objet est retourné
- Appel de fonctions en cascade :

```
$("div.steackhache").hide().addClass("removed");
```

On cache chaque élément et on leur rajoute la classe ‘removed’

- ‘itérateur caché’
- Map / list compréhension [ | | ]

# JQuery

- Tableau ?

*sélecteur ID*

```
$("#mdiv").html("texte à mettre dans la div");
```

- Accès ‘tableau’ possible (éléments DOM)

```
$("#mdiv")[0].innerHTML="texte à mettre dans la div";
```

- Ici, un seul élément (ID)

# JQuery : principes

- Tableau ?

```
$('div.aremplir').html("texte à mettre dans les div");
```

- Accès ‘tableau’ possible (éléments DOM)

```
var elements = $('div.aremplir');
for(var i=0; i<elements.length; i++)
    elements[i].innerHTML = "texte à mettre dans les div" ;
```

# JQuery : principes

- Fonctions utilitaires

**`$.trim(uneChaineDeCaractères);`**

- \$ ou bien JQuery
  - *Méthode de classe*

**`jQuery.trim(uneChaineDeCaractères)`**

# JQuery : principes

- Gestionnaires d'evt et *callbacks*

- En entête :

```
<body onLoad= ..... >
```

- *En théorie* lancement quand la page est chargée

- DOM ok

- Images etc..

```
window.onLoad = function() {
```

```
.....
```

```
};
```

- **<script> à la fin (avant </BODY>)**

# JQuery : principes

- Gestionnaires d'evt et *callbacks* (ready handler)

```
jQuery(document).ready(function() {
```

....

```
});
```

```
$(function() {
```

....

```
});
```

*Plusieurs possibles*

*exécution dans l'ordre de def  
(pas possible avec onLoad)*

***Seulement le DOM***

# JQuery : principes

- Création d'objets DOM

```
<html>
  <head>
    <title> test </title>
    <link rel="stylesheet" type="text/css" href="styles/core.css" />
    <script type="text/javascript" src="scripts/jquery-1.4.js"> </script>
    <script>
      $(function() {
        $("<p> un paragraphe </p>").insertAfter("#test");
      });
    </script>
  </head>
  <body>
    <p id="test" > et voici : </p>
  </body>
</html>
```

et voici :  
un paragraphe

# JQuery : principes

- Extension de jQuery

N'existe pas en JQuery

```
$('form#myForm input.special').disable();
```

- Très simple de rajouter des prédéfinis à jQuery

Fonctions pour le wrapper : on rajoute une méthode

```
$fn.disable = function() {           Pour chaque élément
    return this.each(function() {
        if (this.disable ==null) this.disable = true;
    });
}
```

JQuery UI

## Remarque : existence d'un objet en JS

```
if (objet) {  
    // l'objet existe  
} else {  
    // l'objet n'existe pas  
}
```

- Si l'objet n'existe pas, if (objet) / false
- Préférer if (objet == null) ...

# JQuery

- Principes
  - \$, sélecteur, itérateurs, extensions
- Sélecteurs

# JQuery : sélecteurs CSS

- Syntaxe :
  - `$(sélecteur)`
  - CSS3
    - Indépendant du navigateur
      - Même si le nav n'est pas CSS3
    - CSS de base
      - `a` (toutes les balises `<a>`)
      - `#ID` (toutes les balises avec attribut `id=ID`)
      - `.classe` (toutes les balises de `classe` css)
      - `a p` (toutes les balises `<p>` descendant de `<a>`)

# JQuery : sélecteurs CSS

- Toutes les balises de fin (filles)
  - div p (donc **ok** pour <div> <table> <p> )
  - div#recette p#ingredient
  - div.important h1.probleme p.nonresolu
  - p#avatar.scenario p.personnage
- Toutes les balises de fin **directement** filles
  - p > a (donc **pas ok** pour <div> <table> <p> )
  - ul.maListe > li > a

# JQuery : sélecteurs CSS

- Attributs :
  - a[ href ] (présence de l'attribut)
  - a[ href^="[http://](#)" ]
  - input[type="text"]
  - \$('form#entre input[type="text"]').disable ;
  - \$('p#lienexternes a[ href\$=".pdf"]').attr("target", "\_blank");
  - \$('a[ href\* ="google"] img:not([src]))).attr("src", "google.jpg");
- attribut<sup>^</sup> =
  - La valeur de l'attribut **commence par**
- attribut<sup>\$</sup> =
  - La valeur de l'attribut **se termine par**
- attribut<sup>\*</sup>=
  - La valeur de l'attribut **contient la sous chaîne**

**:not**  
**négation d'une propriété**

# JQuery : sélecteurs CSS

- \* n'importe quel élément
- E tous les éléments <E>
- E F tous les éléments <F> fils de <E>
- E>F tous les éléments <F> **directement** fils de <E>
- E+F tous les éléments <F> **immédiatement** précédés de <E>
- E~F tous les éléments <F> précédés par E
- E.C les éléments <F> de classe C
  - .C même chose que \*.C
- E#I tous les éléments <E id="I">
  - #I même chose que \*#I
- E[A] tous les éléments <E> avec un attribut A
- E[A=], E[A^=], E[A\$=], E[A!=], E[A\*=]

# JQuery : sélecteurs CSS

- Filtres: sélection par **position**
  - :first (premier élément) :last (dernier) exemple **a:first**
  - :even :odd
  - :first-child exemple **ul li:first-child**
  - :last-child
  - :only-child (fils sans frères)
  - :nth-child(n) (attention commence à 1)
  
  - :eq(n) equals (attention commence à 0) nième elt
  - :gt(n) greater than (après le nième)
  - :lt(n) less than (avant le nième)
  
- Remarque : ,

# JQuery : sélecteurs CSS

- Filtres/Sélecteur jQuery : attention

**filtre ≠ sélecteur**

- `$('input[type=checkbox][checked]')`
  - État initial (présence de l'attribut checked)
  - Correspond au code HTML
  - Ne correspond pas à l'état courant
- Filtre jQuery :
  - `$('input[type=checkbox]:checked')`
  - `($('input:checked')`
- Extension des filtres : `$("input:checkbox:checked")`

# JQuery : sélecteurs CSS

- Filtres jQuery
  - :animated (cf controles animés)
  - :button (submit, reset, button)
  - :checkbox
  - :checked (checkbox ou radio)
  - :contains(text)
  - :disabled
  - :enabled
  - :file (input[type=file])
  - :has(selector) (voir + loin)
  - :header (H1...H6)
  - :hidden
  - :image (input[type=image])

# JQuery : sélecteurs CSS

- Filtres jQuery
  - `:input` (`input; select; textarea; button`)
  - `:not(selecteur)` (voir + loin)
  - `:parent` (seulement les éléments qui ont des enfants non vides)
  - `:password`
  - `:radio`
  - `:reset`
  - `:selected` (`<option>` élément sélectionnés)
  - `:submit`
  - `:text` `input[type=text]`
  - `:visible`

# JQuery : sélecteurs CSS

- filtres jQuery : remarques
  - Combiner les filtres :
    - :checkbox:checked:enabled:visible
    - :checkbox:not(:checked:enable)
  - :not()
    - On veut sélectionner toutes les images sauf celles qui contiennent « chien »
    - `\$('not(img[src*="chien"])')`

# JQuery : sélecteurs CSS

- Filtres jQuery : remarques
  - :not()
    - On veut sélectionner toutes les images sauf celles qui contiennent « chien »
    - `$('.:not(img[src*="chien"])')`
    - `$('.img:not([src*="chien"])')`
    - **Attention !**

# JQuery : sélecteurs CSS

- Filtres jQuery : remarques
  - :has
    - div span
      - Sélectionne **seulement les spans** filles de *div*
      - Et si on veut **sélectionner les div** qui ont des *span* filles ?
    - div:has(span)
  - `$(‘tr:has(img[src$=.gif]’))`
    - Sélection de la liste des balises <tr> qui contiennent des images (n’importe où dans la hiérarchie) qui ont un attribut src qui se termine par ‘.gif’.

# JQuery : sélecteurs CSS

- Remarques
  - Appliquer un sélecteur à une sous hiérarchie DOM
    - `$(selecteur)`
      - Application à toute la hiérarchie courante
    - `$(selecteur, 'div#ici')`
      - Application à la div d'ID 'ici'  
(changement de la racine d'application)
  - `($('div span:has(p.test > img:first[src$=".pdf"])),  
span:not(:has(form)),  
span.test')`

intersection

# JQuery : Génération de code HTML

- Sélecteurs + filtres -> localisation / action = attributs + HTML

- Génération de code html

- `$(‘<div>hello</div>’)`
    - `$(‘<div>’)`
      - `$(‘<div></div>’)` ou `$(‘<div/>’)`

- Création d'attributs

- `$(‘<img>’).attr("src", "im.jpg")`
    - `$(‘<img>’, { src: ‘im.jpg’ })`
    - `$(‘<img>’, {`  
  
`src: ‘im.jpg’,`  
`title: ‘vous m’avez cliqué dessus !’,`  
`click: function() {`  
`alert($(this).title));`  
`}`  
`})`

# JQuery : Génération de code HTML

- Sélecteurs + filtres -> localisation / action = attributs + HTML
  - .attr(nomAttribut, valeur)
    - ou \$( , {})
  - Idem pour la création de propriétés css
  - Exemple :

```
$(<img>,
{
    src: 'images/nounours.jpg',
    alt: 'nounours',
    title: 'bonne nuit les petits',
    click: function() {
        alert($(this).attr('title'));
    }
}).css( { cursor: 'pointer',
    border: '1px solid black',
    padding: '12px 12px 20px 12px',
    backgroundColor: 'red'
}).appendTo('body');
```

autre solution avec  
css en attribut ?

# JQuery : Génération de code HTML

- Remarques :
  - .attr(nomAttribut, valeur)
    - Également utilisé en **modification / accès attribut**
    - `$('img#monimage').attr('src')` (accès)
    - `$('img#monimage').attr('src','p.gif');` (modification)
    - `($('img#monimage').attr( { src:'p.gif' } ));` (modification)
  - removeAttr(nomAttribut)
    - Permet d'enlever un attribut

# JQuery : Génération de code HTML

- Accès aux propriétés du wrapper
  - `$('#laDiv').html('il y a ' + $('a').size() + ' hyperliens dans cette page');`
    - Génération de texte (cf la taille de l'ensemble de liens)
    - Insertion du texte comme code HTML (.html)
    - .html() (!... contenu)
  - Cf exemple précédent
    - `$( quoi ).insertion( à quel endroit )`
    - `$( à quel endroit ).inserer( quoi )`
  - `$('a')[4]`
  - `$('a').get(4)`
  - Voir plus loin pour la gestion des wrappers

# JQuery : Génération de code HTML

- Fonctions d'accès
  - **.html()**
    - Renvoie le contenu HTML du premier élément
    - Équivalent de innerHTML
  - **.html(contenu)**
    - Contenu peut être du texte
    - Contenu peut être une fonction

```
$('p').html( function(idx, content) {  
    return "idx="+idx+" "+content;  
});
```

# JQuery : Génération de code HTML

- Fonctions d'accès

- **.text()**

- Renvoie le contenu text de tous les éléments

- **.text(contenu)**

- Remplace le contenu par le texte
    - <, > et & remplacés par leur équivalent textuel
    - Cf innerText

# JQuery : Génération de code HTML

- Fonctions d'accès
  - **.append(contenu)**
    - Modifie le contenu des éléments
    - `$(‘p’).append(‘<b> du text </b>’);`
    - `$(‘p’).append($(‘ul’));`
      - Déplacement et copie des balises `<ul>`
      - Attention ! (ici, le **append** est un **move**)
      - Copies autant que nécessaire

# JQuery : Génération de code HTML

- Fonctions d'accès
  - .append(contenu)
    - Contenu peut également être une fonction
    - Argument de la fonction : élément traité
    - Valeur de retours = contenu à modifier

# JQuery : Génération de code HTML

- Fonctions d'accès
  - .prepend(contenu)
    - .append mais au début de la balise
    - \$('p').prepend('<b> du text </b>');
    - \$('p').prepend(\$('ul'));
      - Déplacement et copie des balises <ul>
      - Attention ! (ici, le **prepend** est un **move**)
      - Copies autant que nécessaire

# JQuery : Génération de code HTML

- Fonctions d'accès
  - .before()
  - .after()
- \$('p').append()
  - Où.ajouter.quoi
- \$('<b> du text </b>').appendTo(\$('p'))
  - Quoi.ajouter.où
    - .prependTo
    - .insertBefore
    - .insertAfter

# JQuery : Génération de code HTML

- ‘emballage’ de balises

```
$('p').wrap('<h1></h1>')
```

```
<p> lkjnl kjnl </p>
<p> lkjljk zd zoidj z </p>
<p> kjdz zdlkj zdlkj zd </p>
```



```
<h1> <p> lkjnl kjnl </p> </h1>
<h1> <p> lkjljk zd zoidj z </p> </h1>
<h1> <p> kjdz zdlkj zdlkj zd </p> </h1>
```

```
$('p').wrapAll('<h1></h1>')
```

```
<p> lkjnl kjnl </p>
<p> lkjljk zd zoidj z </p>
<p> kjdz zdlkj zdlkj zd </p>
```



```
<h1> <p> lkjnl kjnl </p>
<p> lkjljk zd zoidj z </p>
<p> kjdz zdlkj zdlkj zd </p> </h1>
```

# JQuery : Génération de code HTML

- ‘déballage’ de balises

`$('p').unwrap()`

```
<h1> <p> lkjnl kjnl </p> </h1>
<h1> <p> lkjljk zd zoidj z </p> </h1>
<h1> <p> kjdz zdlkj zdlkj zd </p> </h1>
```



```
<p> lkjnl kjnl </p>
<p> lkjljk zd zoidj z </p>
<p> kjdz zdlkj zdlkj zd </p>
```

(enlève les balises parent / 1 niveau)

# JQuery : Génération de code HTML

- Enlever des élément
  - \$('p').remove()
- Remplacer des élément
  - replaceWith

**`$('img').replaceWith('<span>une image </span>')`**

- exercice : remplacer chaque image (avec un attribut alt) par un span

# JQuery : Génération de code HTML

**Replacer chaque image avec un alt par un span qui contient le ‘alt’...**

```
$(‘img[alt]’).each( function() {  
    $(this).replaceWith(‘<span>’+$this.attr(‘alt’)+’<span>’)  
});
```

**Utilisation de l’itérateur obligatoire pour avoir accès à chaque valeur des attributs, élément par élément**

# JQuery : Gestion des formulaires

- Voir également  
<http://jquery.malsup.com/form/>
- `val()`
  - Accès : `$('#monNom').val()`
  - Modification : `$('#monNom').val('joe')`
  - Test / modification:
    - `($('input,select').val(['un','deux','trois']));`
    - Recherche tous les éléments de la page de type input ou select qui ont comme valeur ‘un’, ou ‘deux’ ou ‘trois’ : tous ceux qui seront trouvés seront sélectionnés (et auront une des valeur choisies ou cochée)

# JQuery : Gestion des événements

- Forme générale (événements DOM 1)

```
<script type="text/javascript">
    $(function() {
        $('#exemple')[0].onmouseover= function(event) {
            alert('passage souris');
        };
    });
</script>
```

...

```
<body>
    
</body>
```

# JQuery : Gestion des événements

- Forme générale (évenements DOM 1)

- Problèmes
  - Un seul gestionnaire d'événement / événement
    - Element.onclick = premièrefonction;
    - Element.onclick = secondefonction;
  - Un gestionnaire appelle plusieurs gestionnaires ?

```
$('#element')[0].onclick = function(event) {  
    premièrefonction(event);  
    secondefonction(event);  
}
```

- Réutilisation du code délicate

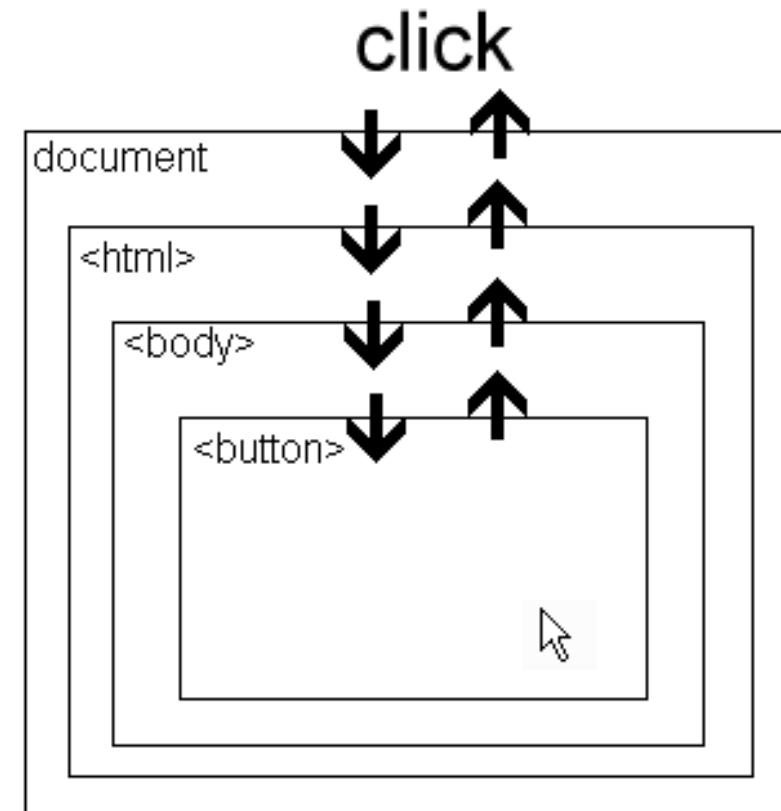
# JQuery : Gestion des événements

- Forme générale (évenements DOM 2)
  - Gestionnaires d'évenements et listeners
  - addEventListener(eventType,listener,useCapture)

```
<script>
    $(function() {
        var element = $('#exemple')[0];
        element.addEventListener('click', function(event) { alert('am');}, false);
        element.addEventListener('click', function(event) { alert('stram');}, false);
        element.addEventListener('click', function(event) { alert('gram');}, false);
    });
</script>
...
<body>
    <img id='exemple' src='test.jpg'/>
</body>
```

# JQuery : Gestion des événements

- Capture / Bubble



# JQuery : Gestion des événements

- Capture / Bubble
  - addEventListener(eventType,listener,useCapture)
    - useCapture à true : capture
    - useCapture à false : bubble
  - Cf <a href="#" onclick="...; return false"> ...

# JQuery : Gestion des événements

- Cf <a href="#" onclick="...; return false"> ...

```
$(function() { var element = $('#exemple')[0];
    element.addEventListener('click', function(event) { alert('am'); return false;}, false);
    element.addEventListener('click', function(event) { alert('stram'); return false;}, false);
    element.addEventListener('click', function(event) { alert('gram'); return false;}, false);
});
```

- Arrêt de la propagation de l'événement
- Cf dessin bubble

# JQuery : Gestion des événements

- JQuery : Unification du modèle d'événement
  - Pour tous les navigateurs (ouf)
  - bind(eventType, data, handler)
  - bind(eventMap) (> 3.0 deprecated, utiliser **on**)

```
$('#exemple')  
    .bind('click',function(event) { alert('am'); })  
    .bind('click',function(event) { alert('stram'); })  
    .bind('mouseover',function(event) { alert('gram'); }) ;
```

jQuery 3.x.y : utiliser 'on' au lieu de 'bind'

# JQuery : Gestion des événements

- bind : espace de noms
  - Catégorie d'événements

```
$('#exemple')  
    .bind('click.comptine',function(event) { alert('am'); })  
    .bind('click.comptine',function(event) { alert('stram'); })  
    .bind('click.comptine',function(event) { alert('gram'); }) ;
```

- Désactiver les événements
  - \$('\*').unbind("click.comptine");
- Exemple : mode édition / utilisation

jQuery 3.x.y : utiliser 'off' au lieu de 'unbind'

# JQuery : Gestion des événements

- bind : autre syntaxe

```
$('.bouton').bind({  
    click: function(event) {...},  
    mouseenter: function(event) {...},  
    mouseleave: function(event) {...}  
});
```

# JQuery : Gestion des événements

- Événements

<b>blur</b>	<b>change</b>	<b>click</b>	<b>dblclick</b>	<b>error</b>
<b>focus</b>	<b>focusin</b>	<b>focusout</b>	<b>keydown</b>	<b>keypress</b>
<b>keyup</b>	<b>load</b>	<b>mousedown</b>	<b>mouseenter</b>	<b>mouseleave</b>
<b>mousemove</b>	<b>mouseout</b>	<b>mouseover</b>	<b>mouseup</b>	<b>ready</b>
<b>resize</b>	<b>scroll</b>	<b>select</b>	<b>submit</b>	<b>unload</b>

# JQuery : Gestion des événements

- One-shot bind
  - `one(eventType, data, listener)`
  - Permet de fabriquer un gestionnaire 'jetable'
    - Une fois l'action déclenchée, un `unbind` est effectuée sur cette action
  - Ne pas confondre avec `on`

# JQuery : Gestion des événements

- `function(event)`
  - Mozilla : `event`
  - IE : `window.event`
  - jQuery : copie en fonction du navigateur (ouf)
  - Propriétés de `event` (par exemple `event.altKey`)
    - `altKey` : true si alt pressée (ou option pour le mac)
    - `ctrlKey`
    - `currentTarget`
    - `data` : cf appel bind
    - `metaKey` (ctrl pour le pc ou Command pour le mac)
    - `relatedTarget` (entrée ou sortie au déclenchement)

# JQuery : Gestion des événements

- Propriétés de event :
  - pageX
  - pageY : position de l'événement / page d'origine
  - screenX
  - screenY : position événement / écran
  - shiftKey
  - timestamp
  - type
  - which
    - Clavier : code numérique de la touche
      - <- 37 (attention Safari donne n'importe quoi)
    - Souris : boutons (1 – 2 – 3)

# JQuery : Gestion des événements

- Toggle
  - Toggle(listener1, listener2, ....)
  - Liste circulaire d'événements click
  - 1-2-3-1-2-3-1-2-3-1...

```
$('img[src*=small]').toggle(  
    function() { $(this).attr('src', $(this).attr('src').replace(/small/,'medium')); },  
    function() { $(this).attr('src', $(this).attr('src').replace(/medium/,'large')); },  
    function() { $(this).attr('src', $(this).attr('src').replace(/large/,'small')); },  
);
```

Cf exemple hibiscus

# JQuery : Gestion des événements

- live / die < 1.7 enlevé à partir 1.9
  - Gestionnaires d'evt accrochés aux éléments
  - Ne peuvent exister que si les éléments existent (déjà)
  - Ajax + création de div + récupération d'éléments
  - Idée : fabriquer des classes d'événement qui s'appliquent **automatiquement** à des objets (s'ils existent)
    - Dès qu'un objet est créé ou importé, il reçoit des gestionnaires **automatiquement**
  - Approche proactive
  - Associer des classes .CSS à des comportements
    - *Quand un objet à une classe particulière, il a automatiquement un comportement particulier (sans avoir à lui ajouter de listeners)*

# JQuery : Gestion des événements

- live / die < 1.7 enlevé 1.9 : utiliser **on**
  - `$('.reactifclick').live('click', function(event) { alert("ouille");});`
  - Dès qu'un objet de la classe **.reactifclick** est créé, il devient actif
  - Très important pour l'exploitation d'Ajax
  - Equivalent de bind (mais sans objet ou avant même que l'objet n'existe)
  - Equivalent de unbind : **die**
  - `$('.reactifclick').die('click');`

# JQuery : Gestion des événements

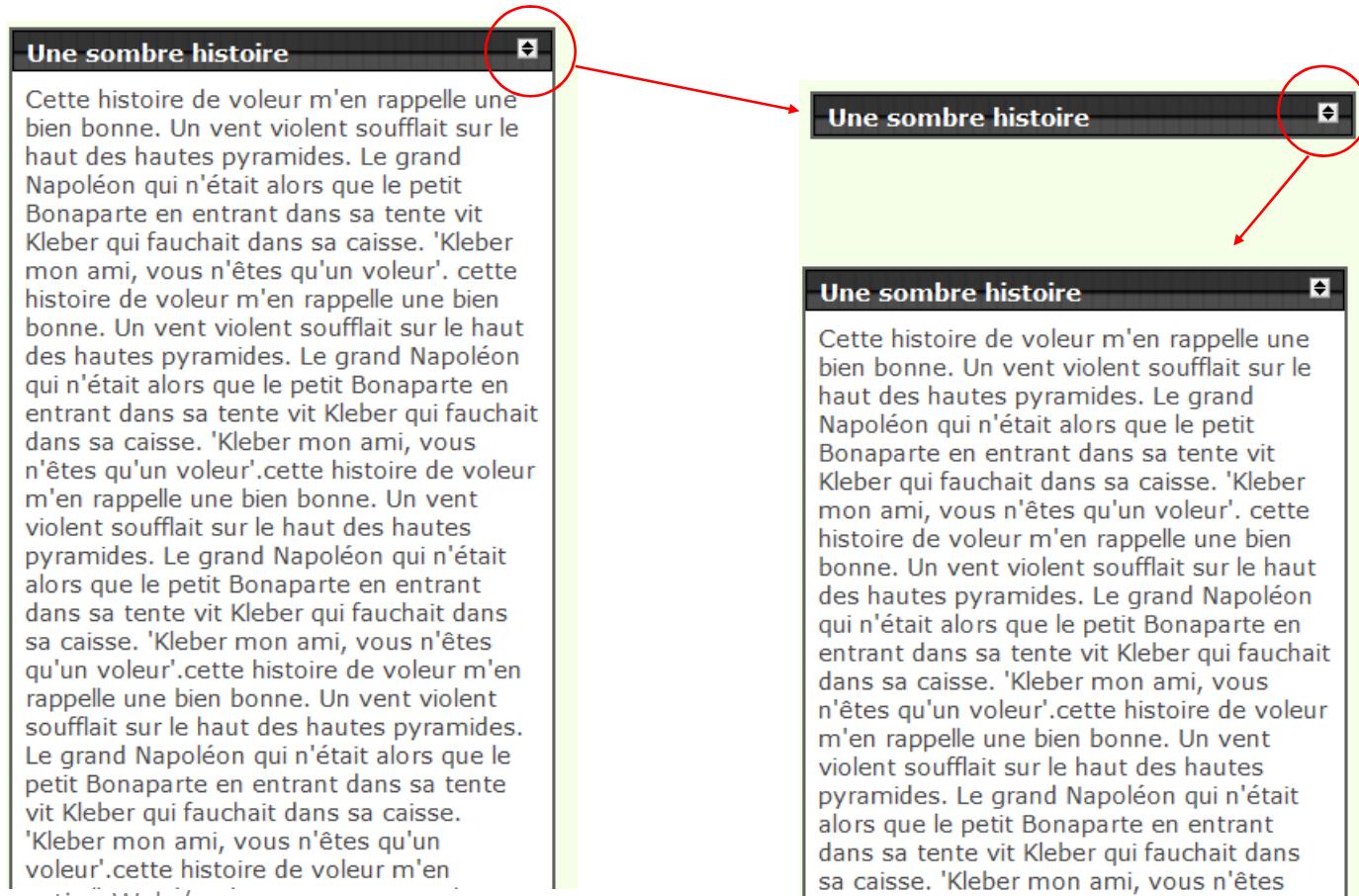
- jQuery >= 1.7
    - **on** et **off** remplacent (*avantageusement bind, unbind, live ...*)
    - live est accroché au document
      - Performance
    - Accrocher **on** au parent (le plus proche – même dynamique)
- ```
$('#parent').on('click', '.reactifclick', function() {  
    // Code  
});
```
- Filtrage des évènements qui remontent des objets de classe **.reactifclick** (donc, pas nécessairement existants au moment du **on**)

# JQuery : Interfaces graphiques

- JQuery
  - Indépendant du navigateur
  - Manipulation du DOM
  - Création de code HTML
  - Gestion d'événements
  - Gestion de l'affichage graphique

# JQuery : Interfaces graphiques

- Cadre refermable



# JQuery : Interfaces graphiques

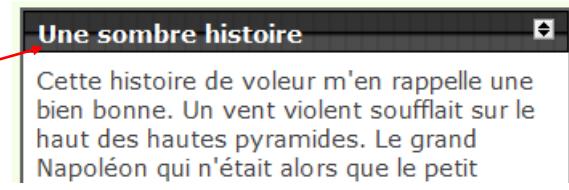
- Principe

```

<div class="module">
    <div class="caption">
        <span>Une sombre histoire</span>
        
    </div>
    <div class="body">
        Cette histoire de voleur m'en rappelle une bien bonne. Un vent violent soufflait sur le haut des hautes pyramides. Le grand Napoléon qui n'était alors que le petit
    </div>
</div>

</body>

```

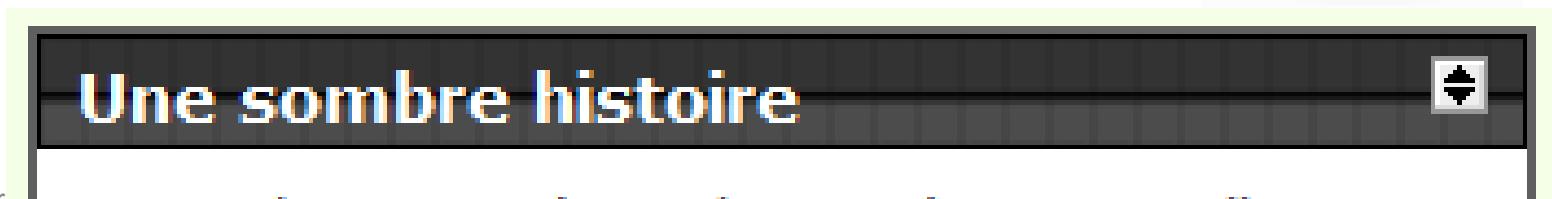
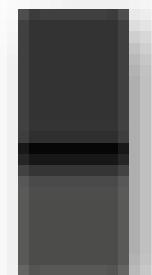


# JQuery : Interfaces graphiques

- Principe

```
div.module {  
    width: 340px;  
    border: 2px #5f5f5f solid;  
    background-color: white;  
}  
  
div.module div.caption {  
    background: black url('module.caption.backg.png');  
    border: 1px solid black;  
    padding: 0 8px;  
    height: 24px;  
    color: white;  
    font-weight: bold;  
}
```

*div.caption fille de div.module*



# JQuery : Interfaces graphiques

```
<html>
  <head>
    <title>Kleber</title>
    <link rel="stylesheet" type="text/css" href="../styles/core.css">
    <link rel="stylesheet" type="text/css" href="module.css">
    <script type="text/javascript" src="../scripts/jquery-1.4.js"></script>
    <script type="text/javascript">
      $(function() {
        $('div.caption img').click(function() {
          var body$ = $(this).closest('div.module').find('div.body');
          if (body$.is(':hidden')) {
            body$.show();
          }
          else {
            body$.hide();
          }
        });
      });
    </script>
  </head>
```

# JQuery : Interfaces graphiques

```
$('div.caption img').click(function() {
```

On change la fonction 'click' pour toutes les images filles des div de classe caption

```
<div class="module">
  <div class="caption">
    <span>Une sombre histoire</span>
    
  <div class="body">
```

# JQuery : Interfaces graphiques

```
var body$ = $(this).closest('div.module').find('div.body');
```

Pour chaque image de l'ensemble sélectionné, on cherche la div de classe body de la div de class module la + proche.

```
<div class="module">
  <div class="caption">
    <span>Une sombre histoire</span>
    
  <div class="body">
    Cette histoire de voleur m'en rappelle
    | entrant dans sa tente vit Kleber qui faucha
    | entrant dans sa tente vit Kleber qui faucha
    | entrant dans sa tente vit Kleber qui faucha
    | entrant dans sa tente vit Kleber qui faucha
    | entrant dans sa tente vit Kleber qui faucha
    | entrant dans sa tente vit Kleber qui faucha
  </div>
```

# JQuery : Interfaces graphiques

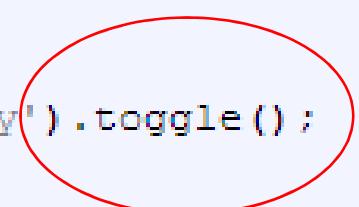
```
if ($('body').is(':hidden')) {  
    $('body').show();  
}  
  
else {  
    $('body').hide();  
}
```

**Si l'objet est caché, on l'affiche, sinon on le cache**

# JQuery : Interfaces graphiques

## Utilisation classique : méthode toggle()

```
<script type="text/javascript">
    $(function() {
        ...
        $('div.caption img').click(function() {
            $(this).closest('div.module').find('div.body').toggle();
        });
    });
</script>
```



# JQuery : Interfaces graphiques

## Aspect psycho des interfaces graphiques (in/out)

**hide(vitesse, callback)**

**show(vitesse, callback)**

**toggle(vitesse, callback)**

**vitesse = nombre de millisecondes**

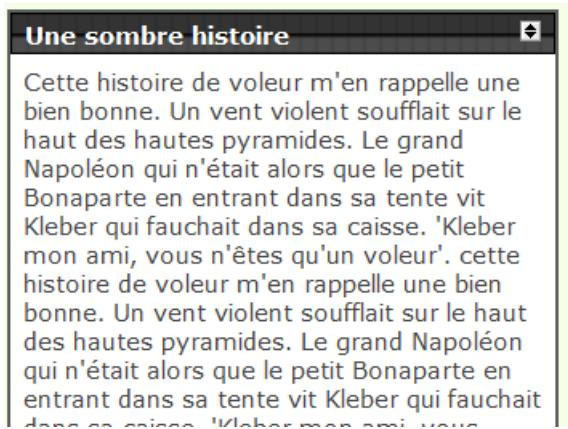
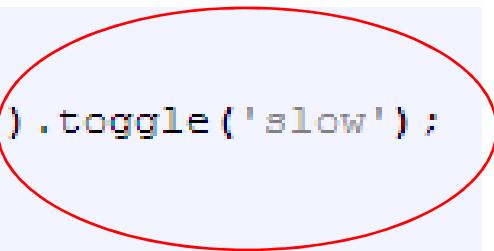
**= ‘slow’, ‘normal’, ‘fast’**

*callback : facultatif : est appelée à la fin de l'action*

# JQuery : Interfaces graphiques

```
...     $('div.caption img').click(function() {
...       $(this).closest('div.module').find('div.body').toggle('slow');
...     });

```



# JQuery : Interfaces graphiques

*callback*

```

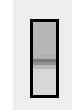
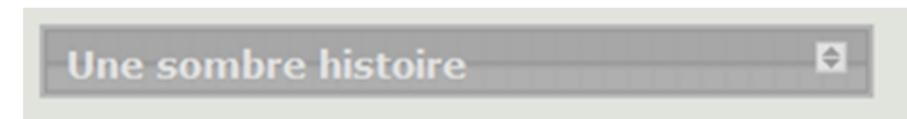
... $('div.caption img').click(function() {
...   $(this).closest('div.module').find('div.body')
...     .toggle('slow', function() {
...       $(this).closest('div.module').toggleClass('rolledup');
...     });
... });

```

```

]div.module.rolledup div.caption{
  background: black url('module.caption.backg.rolledup.png');
}

```



# JQuery : Interfaces graphiques

- **toggleClass(nom)**
  - ◆ Ajoute la classe nom (si elle ne fait pas partie des classes des objets) ou enlève la classe (si elle fait déjà partie des classes)
  - ◆ Combinaison de :
    - **removeClass(nom)**
    - **addClass(nom)**
    - Une variable d'état

# JQuery : Interfaces graphiques

- Show /hide
  - ◆ Animation de la taille
  - ◆ + animation de l'opacité
  - ◆ `fadeIn(vitesse, callback)`
  - ◆ `fadeOut(vitesse, callback)`
  - ◆ `fadeTo(vitesse, opacité, callback)`
  - ◆ Opacité : 0 -> 1.0

# JQuery : Interfaces graphiques

- Slide
  - ◆ `slideDown(vitesse, callback)`
  - ◆ `slideUp(vitesse, callback)`
  - ◆ `slideToggle(vitesse, callback)`
- Asynchrone
  - ◆ Non bloquant pour le code
  - ◆ Thread
  - ◆ `.stop`
  - ◆ `fadeIn/fadeOut et fadeToggle ?`

# JQuery : Interfaces graphiques

- **jQuery**
  - ◆ Base de prédéfinis d'animation minimale
  - ◆ Fonction **animate**
  - ◆ **animate(prop, durée, easing, callback)**
  - ◆ **Animate(prop, options)**
    - **prop** : une table hash avec les propriétés CSS à animer
    - **durée** ('slow' etc..)
    - **easing** ('linear' ou 'swing' ou callback)
    - **options** : hash avec comme clés :
      - ◆ **duration**
      - ◆ **easing**
      - ◆ **queue**
      - ◆ **step (callback)**

# JQuery : Interfaces graphiques

```
$('#ladiv').animate( {opacity:0}, 'slow')
```

```
$('#ladiv').animate( {opacity:1}, 'slow')
```

- fadeToggle ? :

```
$('#ladiv').animate( {opacity:'toggle'}, 'slow')
```

- À rajouter à jQuery

```
$.fn.fadeToggle = function(speed) {  
    return this.animate( {opacity:'toggle'}, speed);  
};
```

# JQuery : Interfaces graphiques

- **animate** permet d'animer simultanément plusieurs propriétés **css**
  - ◆ opacity : 'hide'
  - ◆ opacity : 'show'
  - ◆ opacity : 'toggle'
- **Attention** : { opacity:'hide', top:400}
- **Attention** : position: relative
  - ◆ Pour décrocher l'objet de la page (ou hiérarchie)
  - ◆ Positions négatives
  - ◆ Relatif à sa position d'origine
  - ◆ Exemple animation de **top:** et **left:**
  - ◆ Animations simultanées
  - ◆ jQuery UI : animation des couleurs

# JQuery : Interfaces graphiques

```
$('.animer').each(function(){
    $(this)
        .css('position','relative')
        .animate(
            {
                opacity: 0,
                top: $(window).height() -
                    $(this).height() -
                    $(this).position().top
            },
            'slow',
            function() { $(this).hide(); }
        );
});
```

Si 'hide', callback inutile

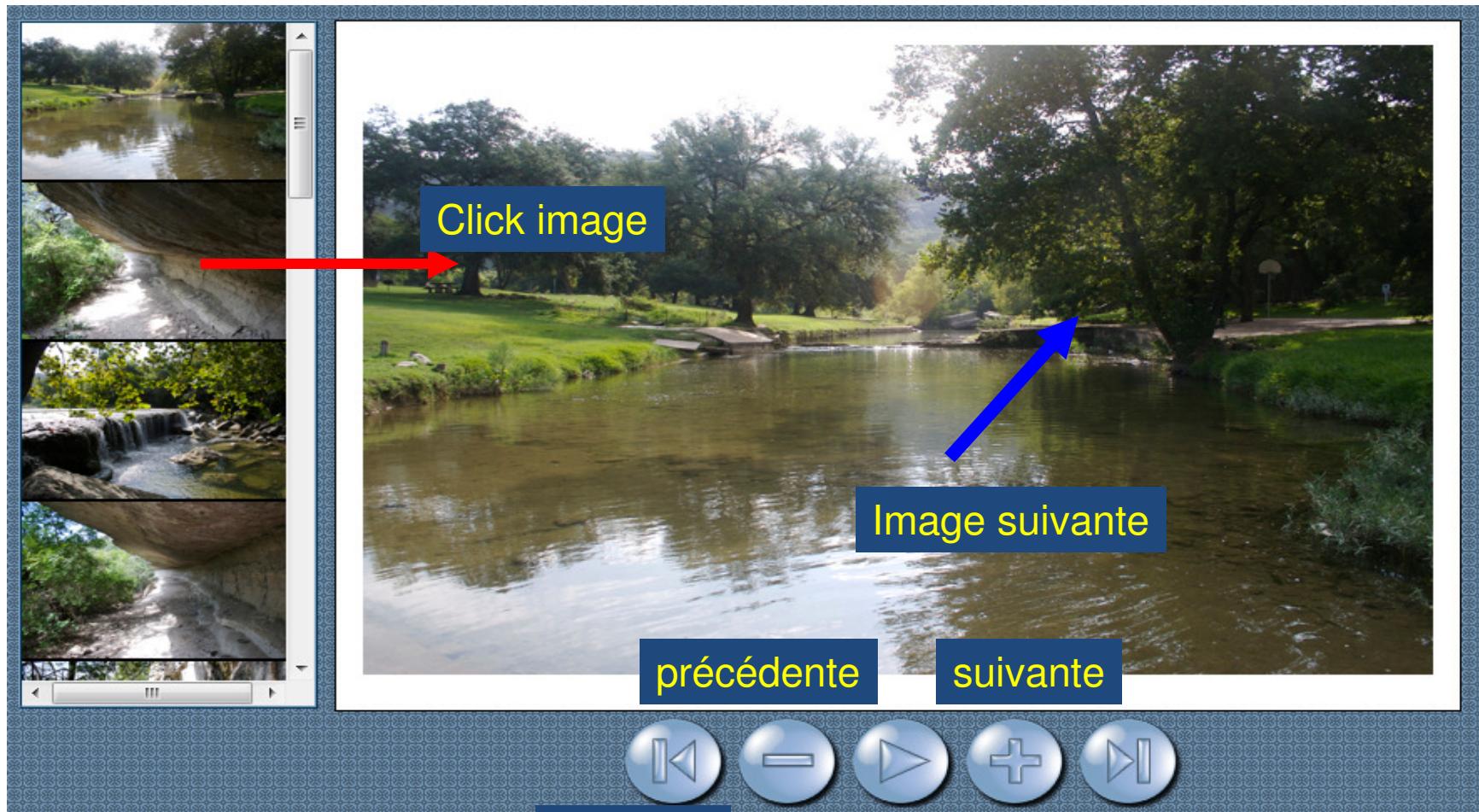
# JQuery : Interfaces graphiques

```
$('#testSubject').each(function() {
    var position = $(this).position();
    $(this)
        .css({position:'absolute',top:position.top,
               left:position.left})
        .animate(
            {
                opacity: 'hide',
                width: $(this).width() * 5,
                height: $(this).height() * 5,
                top: position.top - ($(this).height() * 5 / 2),
                left: position.left - ($(this).width() * 5 / 2)
            },
            'normal');
});
```

# jQuery : étendre jQuery

- Projet :
  - ◆ Fabriquer un *slideshow*
  - ◆ Réutilisable
  - ◆ Assigner des boutons de contrôle
  - ◆ Module jQuery

# jQuery : extension

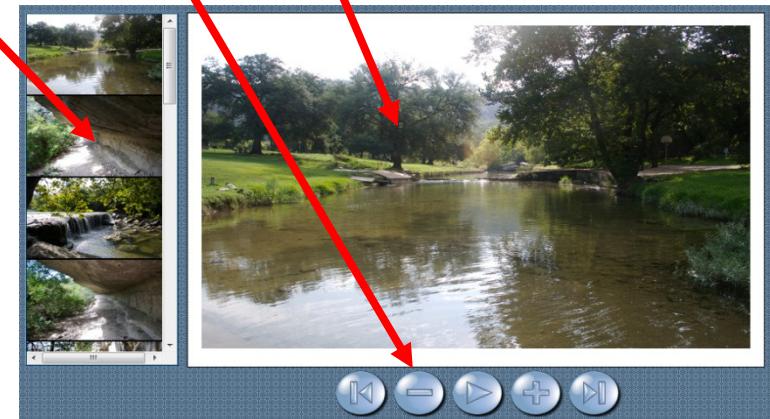


# jQuery : extension

- `$(..).photomatic(options)`
  - Permet de définir un projecteur pour les images du multi ensemble
  - Options est une *hashtable* qui contient les options de projection et de contrôle
    - Par exemple, quel bouton permet d'avancer
    - Quel délais entre chaque image ?

# jQuery : extension

```
<script type="text/javascript">
$(function() {
  $('#thumbnailsPane img').photomatic({
    photoElement: '#photoDisplay',
    previousControl: '#previousButton',
    nextControl: '#nextButton',
    firstControl: '#firstButton',
    lastControl: '#lastButton',
    playControl: '#playButton',
    delay: 1000
  });
})
</script>
```

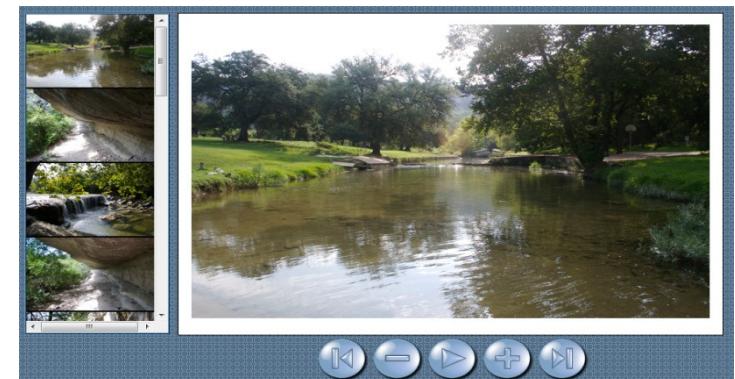


# jQuery : extension

- Réutilisation
  - L'utilisateur peut associer n'importe quel objet à n'importe quel contrôle...
  - ... à n'importe quel endroit
  - Non dépendant de la liste d'image
  - Eventuellement la liste des images peut changer (*chat*)
  - Thèmes d'interfaces
  - *Skinning*

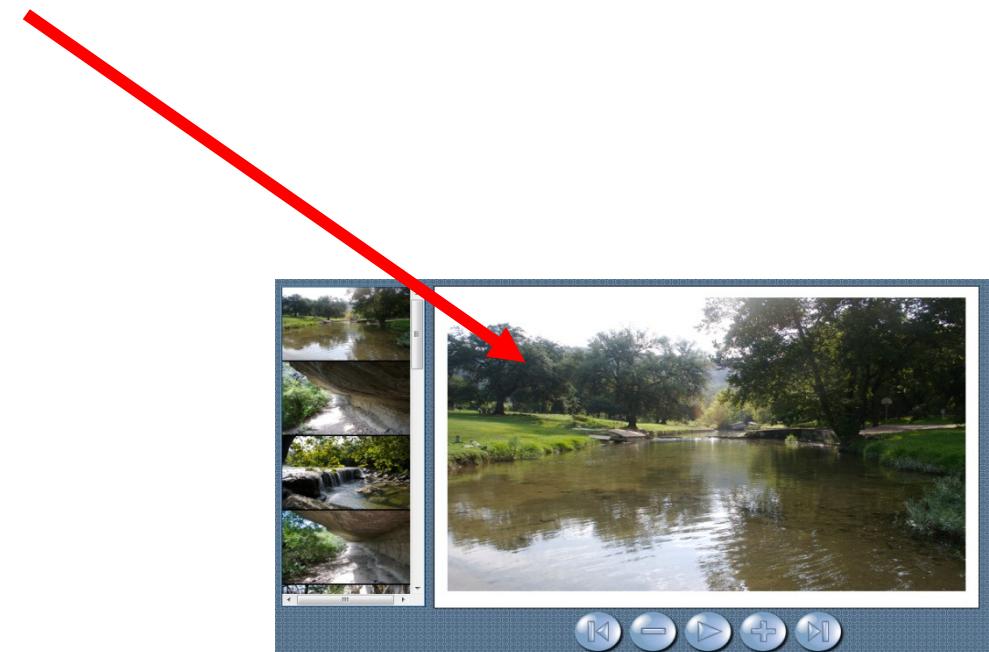
# jQuery : extension

```
<div id="thumbnailsPane">  
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
    
</div>
```



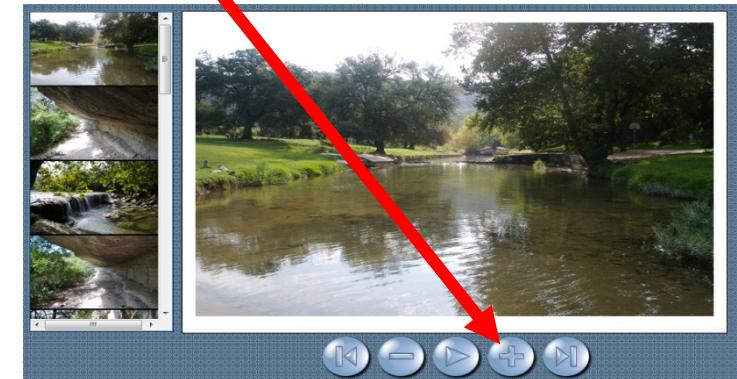
# jQuery : extension

```
<div id="photoPane">  
    <img id="photoDisplay" src="" />  
</div>
```



# jQuery : extension

```
<div id="buttonBar">
    
    
    
    
    
</div>
```



# jQuery : extension

- Options
  - firstControl
    - Saut à la première diapo
  - lastControl
  - nextControl
  - previousControl
  - playControl
  - photoElement
  - delay
  - *transformer*

```
<script type="text/javascript">
$(function() {
    $('#thumbnailsPane img').photomatic({
        photoElement: '#photoDisplay',
        previousControl: '#previousButton',
        nextControl: '#nextButton',
        firstControl: '#firstButton',
        lastControl: '#lastButton',
        playControl: '#playButton',
        delay: 1000
    });
});
</script>
```

# jQuery : extension

Vérification que \$ est bien un alias de jQuery

```
(function($) {  
    $.fn.photomatic = function(options) {  
    };  
}) (jQuery);
```

Rajoute une méthode à jQuery

# jQuery : extension

Code de l'extension :

Compléter les options par des valeurs par défaut (si non renseignées)

```
var settings = $.extend( Méthode de class jQuery
    photoElement: 'img.photomaticPhoto',
    transformer: function(name) {
        return name.replace(/thumbnail/, 'photo');
    },
    nextControl: null,
    previousControl: null,
    firstControl: null,
    lastControl: null,
    playControl: null,
    delay: 3000
}, options || {}); Fusion des options avec des valeurs de hashcode par défaut
```

# jQuery : extension

Nécessité de conserver la liste des vignettes  
mais aussi la vignette courante et sa position dans la liste des vignettes  
settings = variable d'état (options)  
extension de settings :

```
settings-thumbnails$ = this.filter('img');
```

... dans le multi ensemble courant, on sélectionne seulement les images  
(this.filter('img'))

Par convention, on écrit un multi ensemble en terminant son nom par \$

# jQuery : extension

L'idéal serait d'avoir un indice par image avec son numéro d'index pour permettre de conserver le numéro de l'image courante (et de se déplacer)

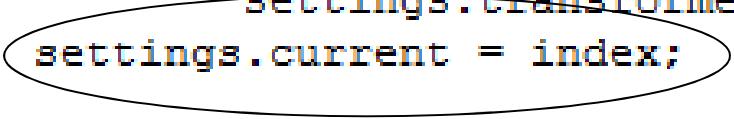
```
settings-thumbnails$.each(  
    function(n) { $(this).data('photomatic-index',n); }  
)
```

... **each** permet d'itérer sur l'ensemble et d'appliquer la fonction. Cette fonction prend en argument le numéro d'index. **data** permet d'associer une valeur à chaque élément du multi ensemble sous un nom donné.

# jQuery : extension

Chaque action provoque l'affichage d'une (grande) image d'index courant.

```
function showPhoto(index) {  
    $(settings.photoElement)  
        .attr('src',  
            settings.transformer(settings-thumbnails[index].src));  
    settings.current = index;  
}
```



`settings.photoElement` est la valeur de l'option `photoElement` (l'ID de l'endroit où placer l'image)

`$(settings.photoElement)` est le multi ensemble des éléments

`.attr` change l'attribut de l'image

par le nom de l'image obtenu par application de `transformer` sur le nom de la vignette d'indice `index`

(convertir le nom URL d'une vignette en URL d'une image)

# jQuery : extension

...restent les clicks d'interaction...

```
settings.current = 0;
settings-thumbnails$ = this.filter('img');
settings-thumbnails$  

    .each(  

        function(n) { $(this).data('photomatic-index', n); }  

    )  

    .click(function() {  

        showPhoto($(this).data('photomatic-index'));  

    });
}
```

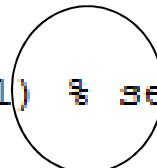
Chaque click sur une vignette provoque l'affichage de l'indexe de la vignette courante

# jQuery : extension

...click sur photoElement -> image suivante...

```
$ (settings.photoElement)
    .attr('title','Click for next photo')
    .css('cursor','pointer')
    .click(function() {
        showPhoto( (settings.current+1) % settings-thumbnails$.length);
    });

```



modulo

# jQuery : extension

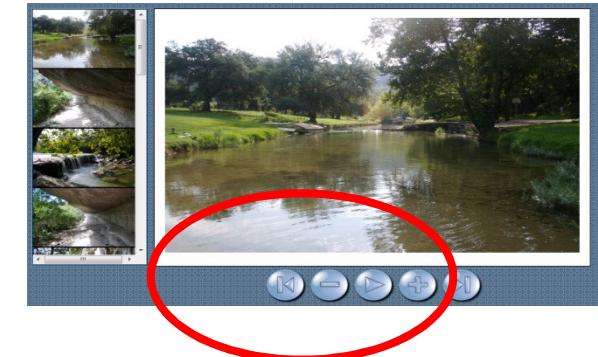
...contrôles...

```
$ (settings.nextControl).click(function() {
    showPhoto ((settings.current+1) % settings.thumbnails$.length);
});

$(settings.previousControl).click(function() {
    showPhoto ((settings.thumbnails$.length+settings.current-1) %
        settings.thumbnails$.length);
});

$(settings.firstControl).click(function() {
    showPhoto (0);
});

$(settings.lastControl).click(function() {
    showPhoto (settings.thumbnails$.length-1);
});
```



# jQuery : extension

...bouton *play* : passage automatique d'une image à l'autre

```
$ (settings.playControl).toggle(  
    function(event) {  
        settings.tick = window.setInterval(      START  
            function(){ $(settings.nextControl).triggerHandler('click') ; },  
            settings.delay);  
        $(event.target).addClass('photomatic-playing');  
        $(settings.nextControl).click();  
    },  
    function(event) {  
        window.clearInterval(settings.tick);  
        $(event.target).removeClass('photomatic-playing');  
    }) ;
```

STOP

# jQuery : extension

Premier *listener* : START

```
function(event) {
    settings.tick = window.setInterval(
        function() { $(settings.nextControl).triggerHandler('click'); },
        settings.delay);
    $(event.target).addClass('photomatic-playing');
    $(settings.nextControl).click();
},
```

On avance immédiatement sur la diapo suivante (sans attendre setinterval)

Déclenchement d'un événement click sur le bouton 'suivant' chaque settings.delay

L'ajout de la classe 'photomatic-playing' permet à l'utilisateur de détecter les objets en cours d'action (par exemple pour changer une couleur)

# jQuery : extension

Premier *listener* : **STOP**

```
function(event) {
    window.clearInterval(settings.tick);
    $(event.target).removeClass('photomatic-playing');
```

On stoppe le déclenchement de l'intervalle (on a stocké le gestionnaire dans settings.tick)

On enlève la classe ‘photomatic-playing’ à l’objet actif



```
(function($){  
    $.fn.photomatic = function(options) {  
        var settings = $.extend({  
            photoElement: 'img.photomaticPhoto',  
            transformer: function(name) {  
                return name.replace(/thumbnail/,'photo');  
            },  
            nextControl: null,  
            previousControl: null,  
            firstControl: null,  
            lastControl: null,  
            playControl: null,  
            delay: 3000  
        },options||{});  
  
        function showPhoto(index) {  
            $(settings.photoElement)  
                .attr('src',  
                    settings.transformer(settings.thumbnails$[index].src));  
            settings.current = index;  
        }  
  
        settings.current = 0;  
        settings.thumbnails$ = this.filter('img');  
        settings.thumbnails$  
            .each(  
                function(n){ $(this).data('photomatic-index',n); }  
            )  
            .click(function(){  
                showPhoto($(this).data('photomatic-index'));  
            });  
  
        $(settings.photoElement)  
            .attr('title','Click for next photo')  
            .css('cursor','pointer')  
            .click(function(){  
                showPhoto((settings.current+1) % settings.thumbnails$.length);  
            });  
    };  
});
```

```
$ (settings.nextControl).click(function(){
    showPhoto((settings.current+1) % settings-thumbnails$.length);
});

$(settings.previousControl).click(function(){
    showPhoto((settings-thumbnails$.length+settings.current-1) %
        settings-thumbnails$.length);
});

$(settings.firstControl).click(function(){
    showPhoto(0);
});

$(settings.lastControl).click(function(){
    showPhoto(settings-thumbnails$.length-1);
});

$(settings.playControl).toggle(
    function(event){
        settings.tick = window.setInterval(
            function(){ $(settings.nextControl).triggerHandler('click'); },
            settings.delay);
        $(event.target).addClass('photomatic-playing');
        $(settings.nextControl).click();
    },
    function(event){
        window.clearInterval(settings.tick);
        $(event.target).removeClass('photomatic-playing');
    });
}

showPhoto(0);
return this;
};

}) (jQuery);
```

 jQuery  
write less, do more.

photomatic.js

```
<!DOCTYPE html>
<html>
  <head>
    <title>Photomatic Test</title>
    <link rel="stylesheet" type="text/css"
          href="../../styles/core.css">
    <link rel="stylesheet" type="text/css" href="photomatic.css">
    <script type="text/javascript"
           src="../../scripts/jquery-1.4.js"></script>
    <script type="text/javascript"
           src="jquery.jqia.photomatic.js"></script>
    <script type="text/javascript">
      $(function() {
        $('#thumbnailsPane img').photomatic({
          photoElement: '#photoDisplay',
          previousControl: '#previousButton',
          nextControl: '#nextButton',
          firstControl: '#firstButton',
          lastControl: '#lastButton',
          playControl: '#playButton',
          delay: 1000
        });
      });
    </script>
  </head>
```

```
<body class="fancy">

  <div id="pageContainer">
    <div id="pageContent">

      <h1>Photomatic Tester</h1>

      <div id="thumbnailsPane">
        
        
        
        
        
        
        
        
        
        
        
        
        
        
        
        
      </div>

      <div id="photoPane">
        <img id="photoDisplay" src="" />
      </div>
```

```
<div id="buttonBar">
    
    
    
    
    
</div>

</div>
</div>

</body>
</html>
```

```
#pageContainer {  
    width: 900px;  
    background-color: transparent;  
    border-width: 0;  
}  
  
h1 {  
    margin: 0 0 32px 64px !important;  
}  
  
#thumbnailsPane {  
    float: left;  
    width: 175px;  
    height: 462px;  
    overflow: auto;  
    border: 2px ridge #00457b;  
    margin-right: 10px;  
    background-color: black;  
}  
  
#buttonBar {  
    clear: both;  
    text-align: center;  
    margin-left: 188px;  
}  
  
#buttonBar img {  
    cursor: pointer;  
}  
  
#photoDisplay {  
    border: 1px solid black;  
    background-color: white;  
    padding: 16px 16px 24px 16px;  
}  
  
img#firstButton {  
    background: url(button.first.png) no-repeat;  
}  
  
img#lastButton {  
    background: url(button.last.png) no-repeat;  
}  
  
img#previousButton {  
    background: url(button.previous.png) no-repeat;  
}  
  
img#nextButton {  
    background: url(button.next.png) no-repeat;  
}  
  
img#playButton {  
    background: url(button.play.png) no-repeat;  
}  
  
img#playButton.photomatic-playing {  
    background: url(button.pause.png) no-repeat;  
}
```



# Ajax



# Ajax : principes de base

- Ajax = Javascript + XML + HTTP GET/POST
  - Javascript de base (ECMA 5 6)
  - Jquery
  - Promesses et extensions async



# Ajax : principes de base

- Ajax = Javascript + XML + HTTP GET/POST
  - CSS
  - XSLT
  - SVG



# Ajax : principes de base

- **Asynchronous JAvascript and Xml**
  - Javascript + objet non standard XMLHttpRequest
  - = scripter le GET/POST du protocole HTTP
  - RFC 2616
  - Communication avec des scripts coté serveur
  - ‘Asynchrone’ :
    - Sans recharger la page (CGI)
    - Non bloquant coté Javascript



# Ajax : principes de base

Rappel : HTTP à étudier !!!

- **POST/GET**

- Communication formulaire / Serveur
- **GET** : lancement d'un script côté Serveur + communication en clair dans l'URL
- **POST** : les paramètres ne sont pas passés dans l'URL mais ds protocole de com  
HTTP    /script.php ?nomVariable1=valeurVariable1&nomVariable2=valeurVariable2
- Attente de réponse
- Réponse du serveur : autre page HTML qui remplace la précédente



# Ajax : principes de base

- Ajax & HTTP
  - Requêtes **GET** ou **POST** à partir de Javascript SANS modifier la page courante (communication ‘masquée’ / ‘cachée’)
  - Communication avec le serveur (et ses scripts) en utilisant les variables de **POST** / **GET**
  - Objet dédié capable de se comporter comme un mini-moteur de requêtes HTTP
  - Objet Javascript
  - Requêtes + récupération + gestion Asynchronisme

# Ajax : principes de base



- Ajax & XMLHttpRequest
  - Origine :
    - IE XMLHTTP (activeX)
  - Mozilla, Safari etc
    - XMLHttpRequest (objet Javascript)

```
if (window.XMLHttpRequest) { // FireFox, Safari, Mozilla, Opera etc..
    http_request = new XMLHttpRequest();
} else { // IE
    http_request = new ActiveXObject(" Microsoft.XMLHTTP ");
}
```



# Ajax : principes de base

- En fait :

```
var http_request = false;  
if (window.XMLHttpRequest) {  
    http_request = new XMLHttpRequest();  
    if (http_request.overrideMimeType) {  
        http_request.overrideMimeType('text/xml');  
    }  
} else if (window.ActiveXObject) {  
    try {  
        http_request = new ActiveXObject("Msxml2.XMLHTTP");  
    } catch (e) {  
        try {  
            http_request = new ActiveXObject("Microsoft.XMLHTTP");  
        } catch (e) {}  
    }  
}
```

*Opera etc..*

*IE*



# Ajax : principes de base

- Mime : `http_request.overrideMimeType('text/xml');`
  - Force le retour Mime du serveur
  - Mozilla (pb pour la gestion XML)
- Requête :
  - `http_request.open('GET','http://.....',true);`
    - Appel méthode HTTP (*Majuscule !*)
      - Codage des données GET possible dans l'URL
    - URL limité au site d'origine
    - true : asynchrone
      - Javascript n'attend pas la réponse de la page

# *Multipurpose Internet Mail Extensions*



« *MIME* » redirige ici. Pour les autres significations, voir [Mime \(homonymie\)](#).

**Multipurpose Internet Mail Extensions** (MIME) ou **Extensions multifonctions du courrier internet** qui étend le [format de données des courriels](#) pour supporter des textes en différentes codifications que l'[ASCII](#), des contenus non textuels, des contenus multiples, et des informations d'encodage. Les courriels étant généralement envoyés via le protocole [SMTP](#) au format MIME, ces courriels sont nommés **courriels SMTP/MIME**.

À l'origine, SMTP avait été prévu pour ne transférer que des fichiers textes (codés en [ASCII](#)) et l'utilisation croissante des applications bureautiques, le besoin s'est fait sentir d'échanger des fichiers binaires (format des applications bureautiques, images, sons, fichiers compressés).

Les types de contenus définis par le standard MIME peuvent être utilisés à d'autres fins que les protocoles de communication comme le [HTTP](#) pour le [World Wide Web](#).

MIME est initialement spécifié dans cinq [RFC](#) : RFC 2045<sup>2</sup>, RFC 2046<sup>3</sup>, RFC 2047<sup>4</sup>, RFC 2048<sup>5</sup> et complétée par la RFC 2077<sup>7</sup>. La RFC 2048, maintenant obsolète, est remplacée par la RFC 2049<sup>6</sup>.

Sommaire [masquer]

- 1 [Introduction](#)
- 2 [En-têtes MIME](#)
  - 2.1 [MIME-Version](#)



# Ajax : principes de base

- GET/POST :
  - POST : les variables sont envoyées via protocole HTTP
  - `http_request.send(' nom1=valeur1&nom2=valeur2');`
  - Dans le cas d'un GET :
    - `http_request.send(null);`
- Un fois la requête lancée, le code Javascript continue à être interprété
- Non bloquant
  - Temps Web ? Erreur ? Etc..
- *Listener* / observation de la progression du chargement
- Méthode de l'objet `http_request`



# Ajax : principes de base

- Méthode de ‘surveillance’ de chargement

- onreadystatechange
  - `http_request.onreadystatechange = nomMéthode;`
  - Cf cours sur Javascript & Méthodes
  - Exemple :

Pas d'arguments !

```
http_request.onreadystatechange = function maFonction() {  
    ...  
}
```

- A chaque fois qu'un nouvel evt est déclenché, la fonction maFonction est appelée.

# Ajax : principes de base

- Etat :
  - État de l'objet `http_request`
  - `http_request.readyState`
    - 0 : pas de requête encore
    - 1 : en cours de chargement (`send` non encore exécutée)
    - 2 : chargement (`send` exécutée)
    - 3 : des morceaux ont été chargés (on peut commencer à les lire)
    - 4 : chargement terminé
  - `http_request.status`
    - Status Protocol HTTP
    - Voir spécif HTTP (200 = *ok*, 404 = *non trouvée*, 500 = *pépin serveur*)
  - `http_request.statusText`
    - Ligne de status HTTP (code + commentaire HTTP)



# Ajax : principes de base

- Donc...

```
var http_request = false;
if (window.XMLHttpRequest) {
    http_request = new XMLHttpRequest();
    if (http_request.overrideMimeType) {
        http_request.overrideMimeType('text/xml');
    }
} else if (window.ActiveXObject) {
    try {
        http_request = new ActiveXObject("Msxml2.XMLHTTP");
    } catch (e) {
        try {
            http_request = new ActiveXObject("Microsoft.XMLHTTP");
        } catch (e) {}
    }
}
if (http_request != false) {

    http_request.onreadystatechange = function maFonction() {
        ... agir en fonction état du chargement...
    }
    http_request.open('GET','http://.....',true);
    http_request.send(null);

} else alert("probleme XMLHttpRequest");
```



# Ajax : principes de base

- Action & contenu chargé
  - `http_request.responseText`
    - *Body* complet comme une chaîne de caractères
  - `http_request.responseXML`
    - Contenu analysé comme du XML (attention ‘text/xml’)
  - `http_request.responseBody`
    - Tableau de caractères
  - `http_request.responseStream`
    - IStream

# Ajax : principes de base



```
<script type="text/javascript" language="javascript">

function lanceRequete(url) {
    var http_request = false;
    if (window.XMLHttpRequest) { // Mozilla, Safari, ...
        http_request = new XMLHttpRequest();
        if (http_request.overrideMimeType) {
            http_request.overrideMimeType('text/xml');
        }
    } else if (window.ActiveXObject) { // IE
        try {
            http_request = new ActiveXObject("Msxml2.XMLHTTP");
        } catch (e) { try { http_request = new ActiveXObject("Microsoft.XMLHTTP");
            } catch (e) {}}
        }
    if (!http_request) {
        alert('Pépin : impossible de créer l\'objet XMLHttpRequest');
        return false;
    }

    http_request.onreadystatechange = function() {
        surveille_reponse_pour(http_request);
    };
    http_request.open('GET', url, true);
    http_request.send(null);
}
```

# Ajax : principes de base



```
function surveille_reponse_pour(ma_requete) {  
    if (ma_requete.readyState == 4) { // terminé  
        if (ma_requete.status == 200) { // pas d'erreur  
            alert(ma_requete.responseText); // affiche le texte  
        } else { alert('Un problème est survenu avec la requête.'); }  
    }  
}  
</script>
```

```
<a href="#" onclick="lanceRequete('essai.html')">  
    Lancer une requête via AJAX  
</a>
```

-> récupération du fichier ‘essai.html’ sur le serveur et affichage de son contenu (texte) dans une boîte d’alerte

# Ajax : principes de base



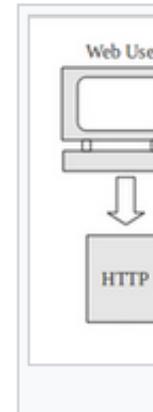
- Appels concurrents
  - attention aux variables globales
- Sémaphores ?
- Exemple donné seulement sur des données statiques (!)
  - Le fichier existe déjà
- Scripts **CGI** / production de fichier ‘dynamiquement’
  - Xml produit par serveur (CGI / PHP / ASP)
  - ‘Object Expected’ (IE)
    - Content-Type: application/xml
    - Cache-Control: no-cache

# Common Gateway Interface

La **Common Gateway Interface** (littéralement « Interface de passerelle commune »), généralement abrégée **CGI**, est une interface utilisée par les serveurs HTTP. Elle a été normalisée par la [RFC 3875<sup>1</sup>](#).

**Sommaire** [[masquer](#)]

- [1 Principe de fonctionnement](#)
  - [1.1 Limitations et évolutions](#)
- [2 Historique](#)
- [3 Exemple d'utilisation](#)
- [4 Notes et références](#)
- [5 Voir aussi](#)
  - [5.1 Articles connexes](#)
  - [5.2 Liens externes](#)



## Principe de fonctionnement [ [modifier](#) | [modifier le code](#) ]

Au lieu d'envoyer le contenu d'un [fichier](#) (fichier HTML, image), le serveur HTTP exécute un [programme](#) qui génère le contenu. CGI est le [standard industriel](#) qui indique comment transmettre la requête du serveur et comment récupérer la réponse générée. Un exemple classique de paramètre est la [chaîne de caractères](#) recherchées auprès d'un [moteur de recherche](#).

# Ajax : principes de base



- Gestion de données structurées (XML)
  - Récupération document XML
  - + Analyse document
  - Accès au contenu des balises
    - Même principe que pour DOM
  - Exemple :

```
<?xml version="1.0" ?>
<racine>
    info a recuperer
</racine>
```

# Ajax : principes de base



- Gestion de données structurées (XML)

```
var xmldoc = http_request.responseXML;
```

Récupération de la réponse ‘analysée’

```
var root_node = xmldoc.getElementsByTagName('racine').item(0);
```

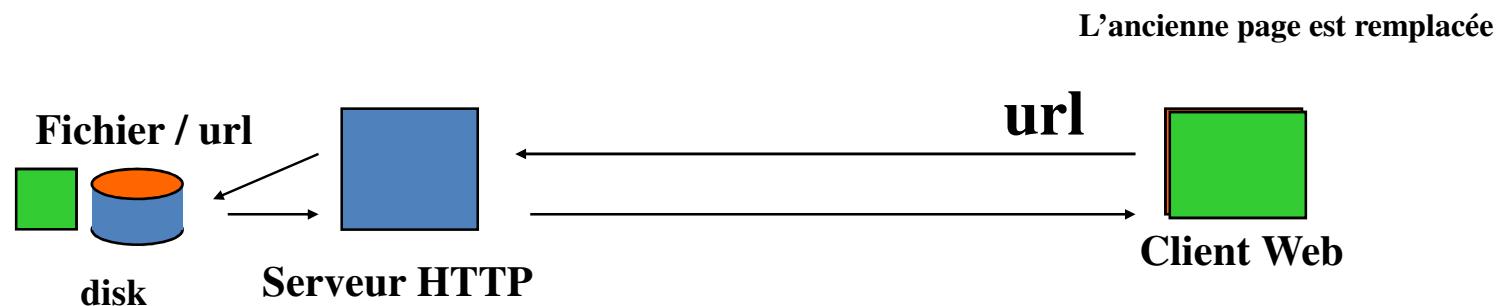
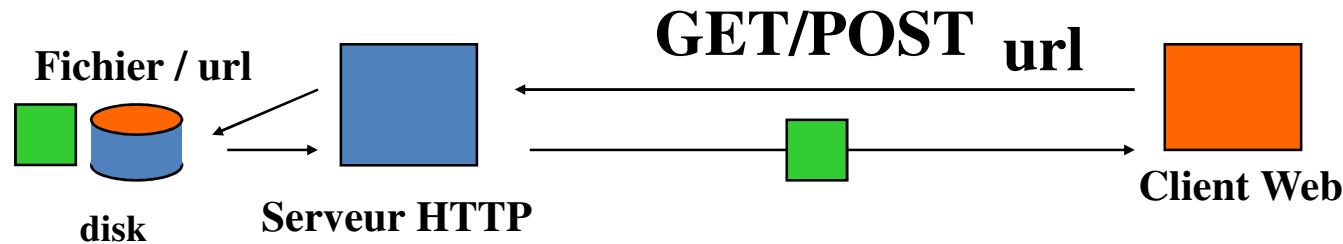
Variable ‘racine’ : tab[0]

```
alert(root_node.firstChild.data);
```

Contenu (ici) du premier fils

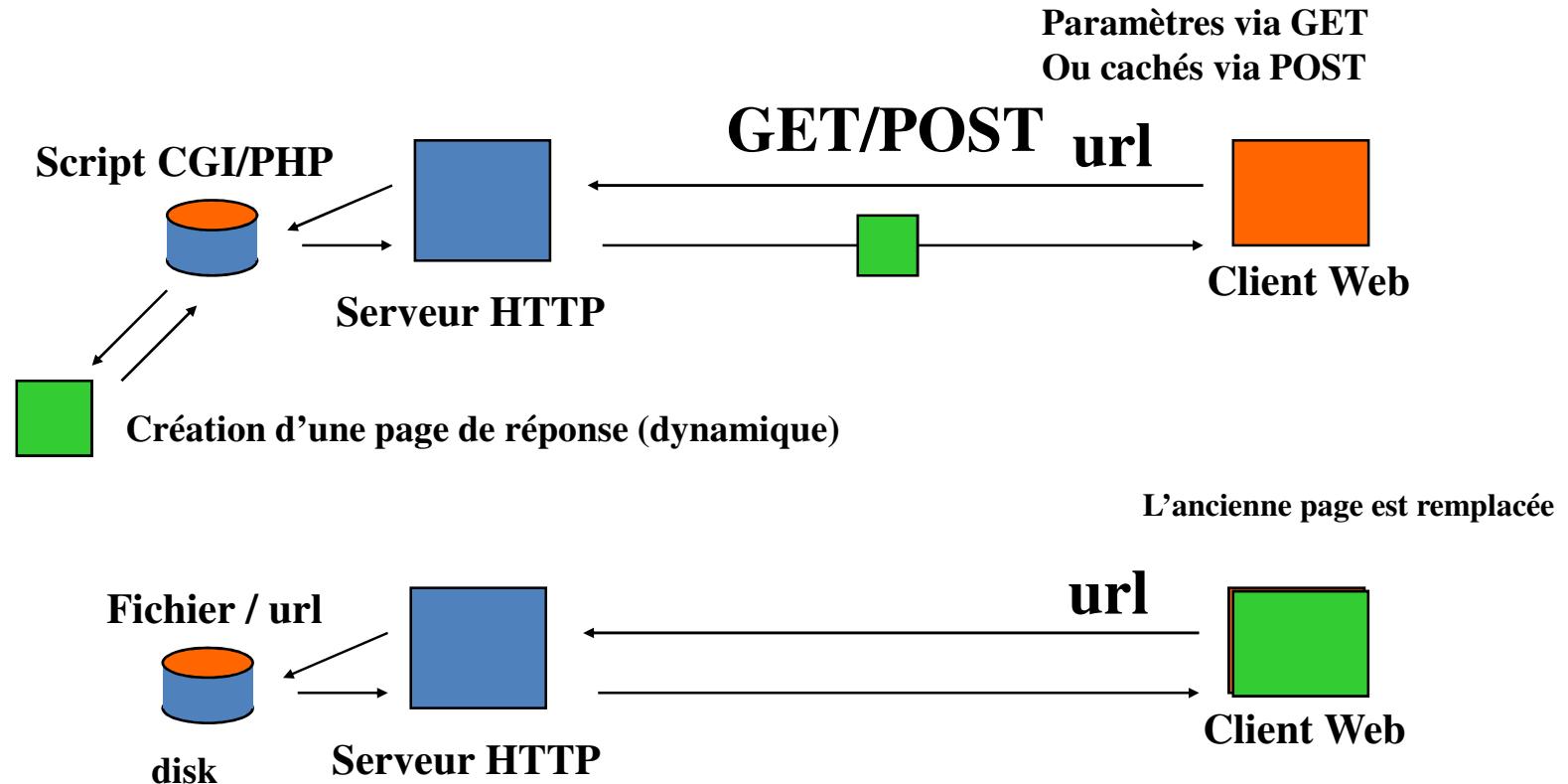
*Même mécanisme pour accéder  
aux autres éléments de l’arbre XML DOM*

# Ajax : principes de base



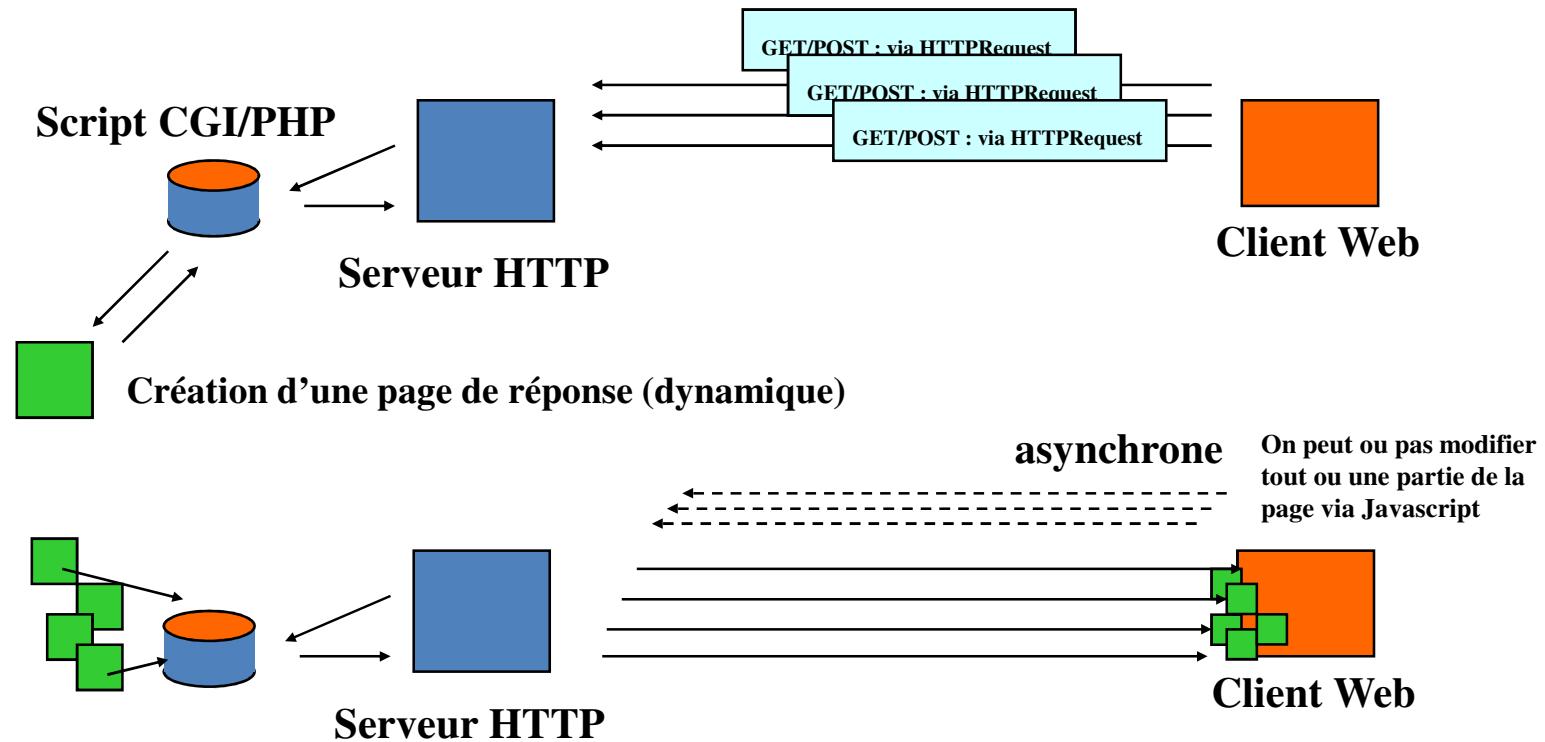
**Le fichier récupéré remplace la page de départ**

# Ajax : principes de base



**Le fichier récupéré remplace la page de départ**

# Ajax : principes de base



**AJAX :** Javascript permet de contrôler un script Distant SANS forcer le rechargement de la page  
**Client -> Serveur** = données via GET/POST  
**Serveur -> Client** = données dans une structure XML

**Client / Serveur Sockets / Applet Java**



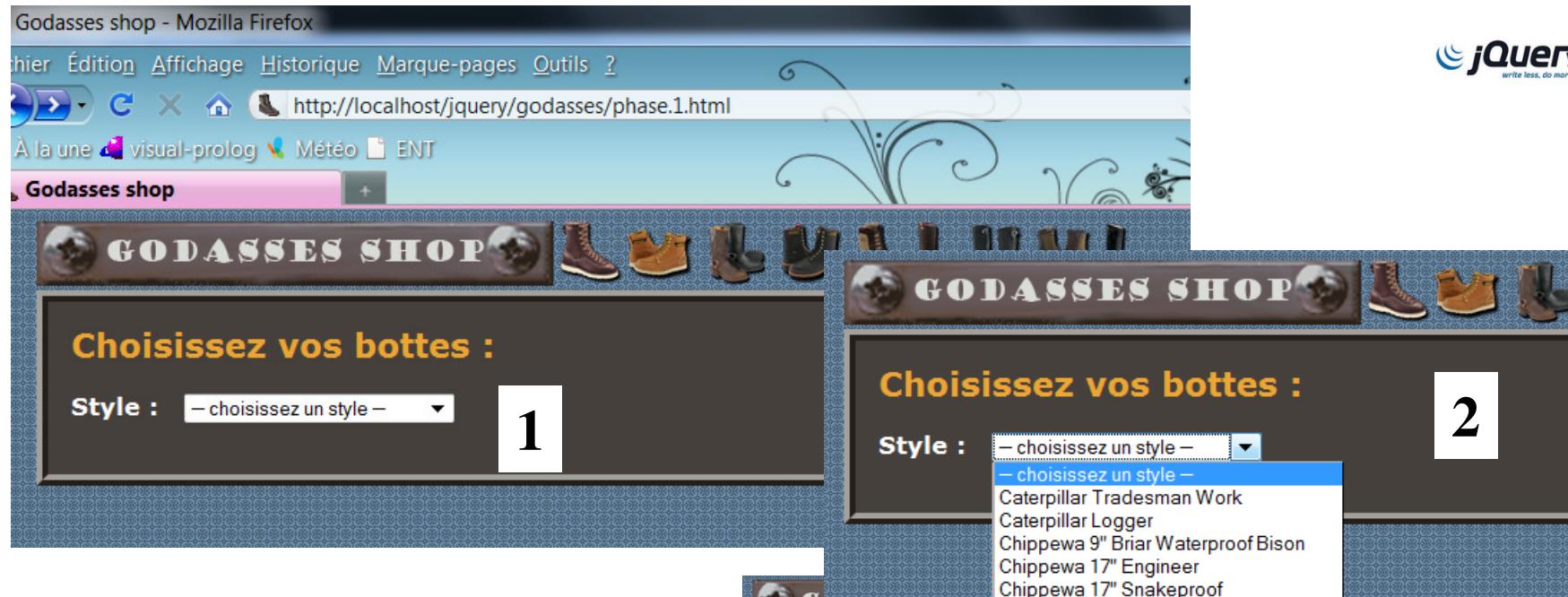
# Ajax : principes de base

```
function construitRequete(sUrl, oParams) {  
    var res = "";  
    for (sName in oParams) {  
        if (sUrl.indexOf("?") > -1) {  
            res += "&";  
        } else {  
            res += "?";  
        }  
        res += encodeURIComponent(sName) + "=" + encodeURIComponent(oParams[sName]);  
    }  
    return res  
}  
...  
var oParams = {  
    "param1": "value1",  
    "param2": "value2"  
};  
var r = construitRequete("/path/to/myjs.php", oParams)
```

**Encodage des params dans un GET**

# jQuery : Utilisation d'Ajax

- Projet :
  - ◆ Exemple d'un magasin en ligne de bottes
  - ◆ Utilisation d'ajax pour mettre à jour les choix / descriptions de bottes
  - ◆ Exemple :
    - WAMP / LAMP / MAMP
    - Prérequis : Php / MySQL
    - La base de données mySQL contient
      - ◆ La liste des types de bottes
      - ◆ Les descriptions de bottes
      - ◆ Stocks etc..



**1.Ouverture de la page  
ajax récupère les options via  
une requête sql sur la base de données**

**2.Choix de l'utilisateur dans les options**

**3.Requête ajax pour récupérer la description**

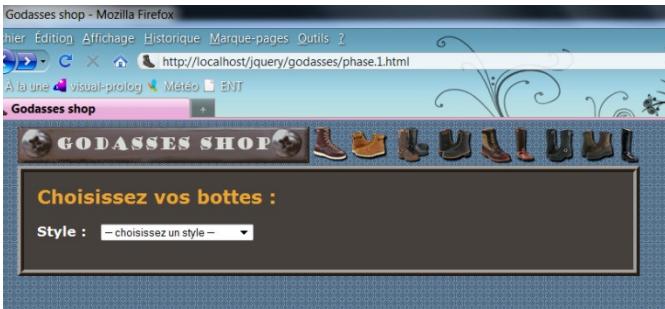
# jQuery : Utilisation d'Ajax

## ■ Principe

- Sélection : <select>
- Remplissage des options de <select> avec du code fabriqué par PHP à partir du contenu de la base
- (*voir les cours TechnoWeb de L2*)

```
<option value="">&mdash; choisissez un style &mdash;</option>
<option value="7177382">Caterpillar Tradesman Work </option>
<option value="7269643">Caterpillar Logger </option>
<option value="7332058">Chippewa 9" Briar Waterproof Bison</option>
<option value="7141832">Chippewa 17" Engineer </option>
<option value="7141833">Chippewa 17" Snakeproof </option>
<option value="7173656">Chippewa 11" Engineer </option>
<option value="7141922">Chippewa Harness </option>
<option value="7141730">Danner Foreman Pro Work </option>
<option value="7257914">Danner Grouse GTX </option>
```

# jQuery : Utilisation d'Ajax



```
<div id="selectionsPane">
  <label for="bootChooserControl">Style : </label>&ampnbsp;
  <select id="bootChooserControl" name="bootStyle"></select>
</div>
```

Requête Ajax sur la page '*fetchBootStyleOptions.php*'  
 Le résultat est inséré dans **#bootChooserControl**

```
<div id="selectionsPane">
  <label for="bootChooserControl">Style : </label>&ampnbsp;
  <select id="bootChooserControl" name="bootStyle"></select>
</div>
```

Injecter du code

```
<option value="">&mdash; choisissez un style &mdash;</option>
<option value="7177382">Caterpillar Tradesman Work </option>
<option value="7269643">Caterpillar Logger </option>
<option value="7332058">Chippewa 9" Briar Waterproof Bison</option>
<option value="7141832">Chippewa 17" Engineer </option>
<option value="7141833">Chippewa 17" Snakeproof </option>
<option value="7173656">Chippewa 11" Engineer </option>
<option value="7141922">Chippewa Harness </option>
<option value="7141730">Danner Foreman Pro Work </option>
<option value="7257914">Danner Grouse GTX </option>
```

par exemple la clé  
 Sql de l'objet (référence)

# jQuery : Utilisation d'Ajax

```
<body>

    <div id="banner">
        
    </div>

    <div id="pageContent">

        <h1>Choisissez vos bottes :</h1>

        <div>

            <div id="selectionsPane">
                <label for="bootChooserControl">Style : </label>&nbsp;
                <select id="bootChooserControl" name="bootStyle"></select>
            </div>

            <div id="productDetailPane"></div>

        </div>

    </div>

</body>
```



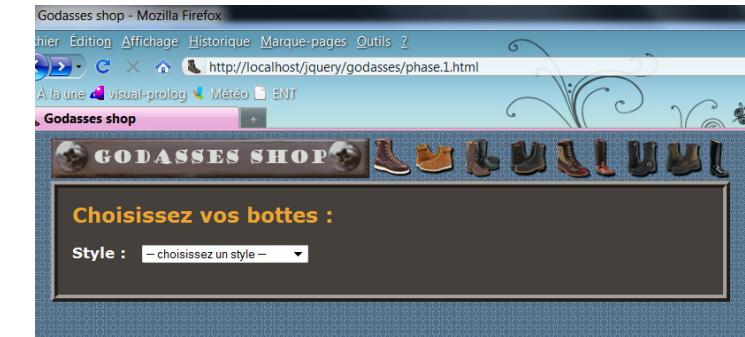
# jQuery : Utilisation d'Ajax

```
<script type="text/javascript">
$(function() {
    $('#bootChooserControl')
        .load('actions/fetchBootStyleOptions.php');
})
</script>
```

**load :**

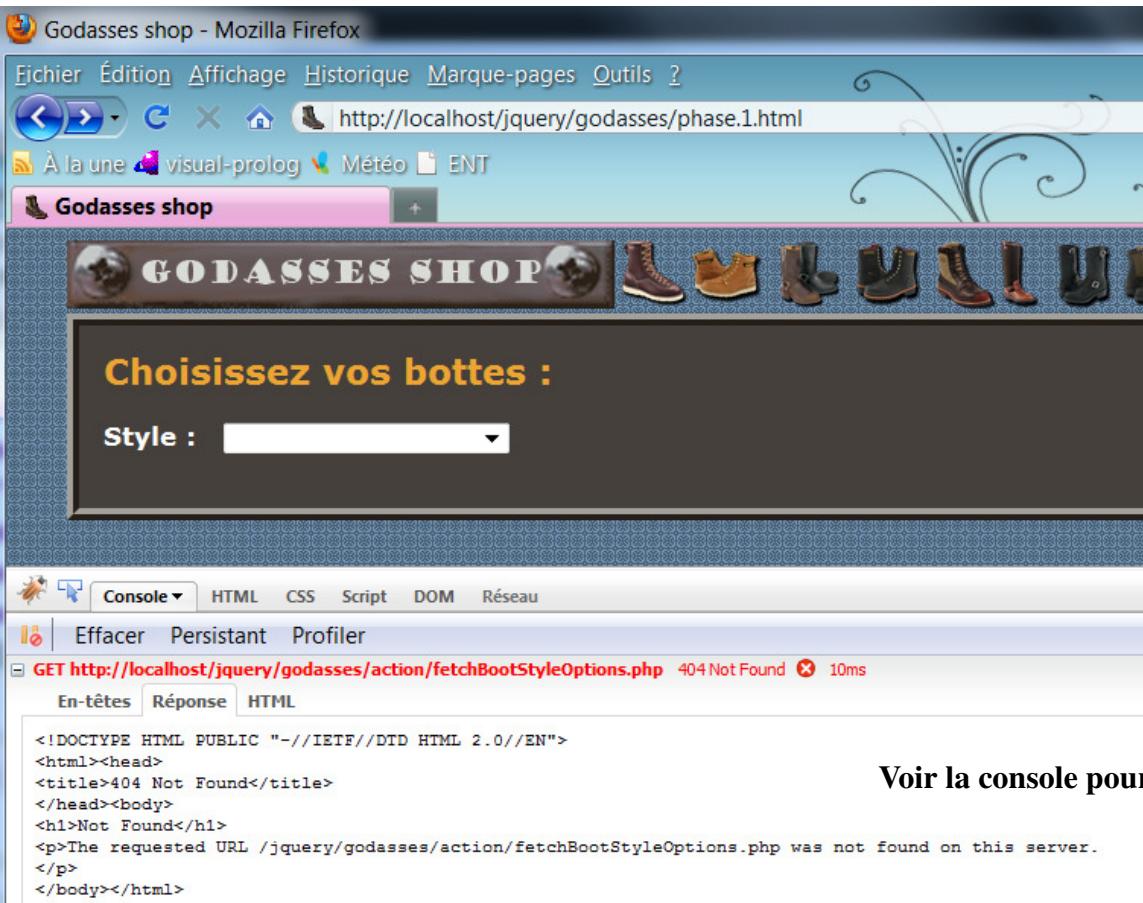
requête **ajax** avec *actions/fetchBootStyleOptions.php* comme url  
**non bloquant**  
**quand le contenu est reçu, il est inséré dans #bootChooserControl**  
**(cad dans le <select>)**

exercice (**à savoir faire**) : écrire le code équivalent en Javascript  
sans jQuery.



# jQuery : Utilisation d'Ajax

## Mise au point : utilisation de firebug



The screenshot shows a Firefox browser window displaying a website for "Godasses shop". The main content area shows a banner with various boot styles and a dropdown menu labeled "Choisissez vos bottes : Style :". Below this is a "Console" tab in the Firebug toolbar, which shows a 404 Not Found error for the URL `/jquery/godasses/action/fetchBootStyleOptions.php`. The error message in the console is:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /jquery/godasses/action/fetchBootStyleOptions.php was not found on this server.</p>
</body></html>
```

**Voir la console pour contrôler le fonctionnement d'Ajax**

Godasses shop - Mozilla Firefox

Fichier Édition Affichage Historique Marque-pages Outils ?

jQuery write less, do more.

À la une visual-prolog Météo ENT

Godasses shop

**GODASSES SHOP**

**Choisissez vos bottes :**

Style :

Console ▾

HTML CSS Script DOM Réseau

Effacer Persistant Profiler

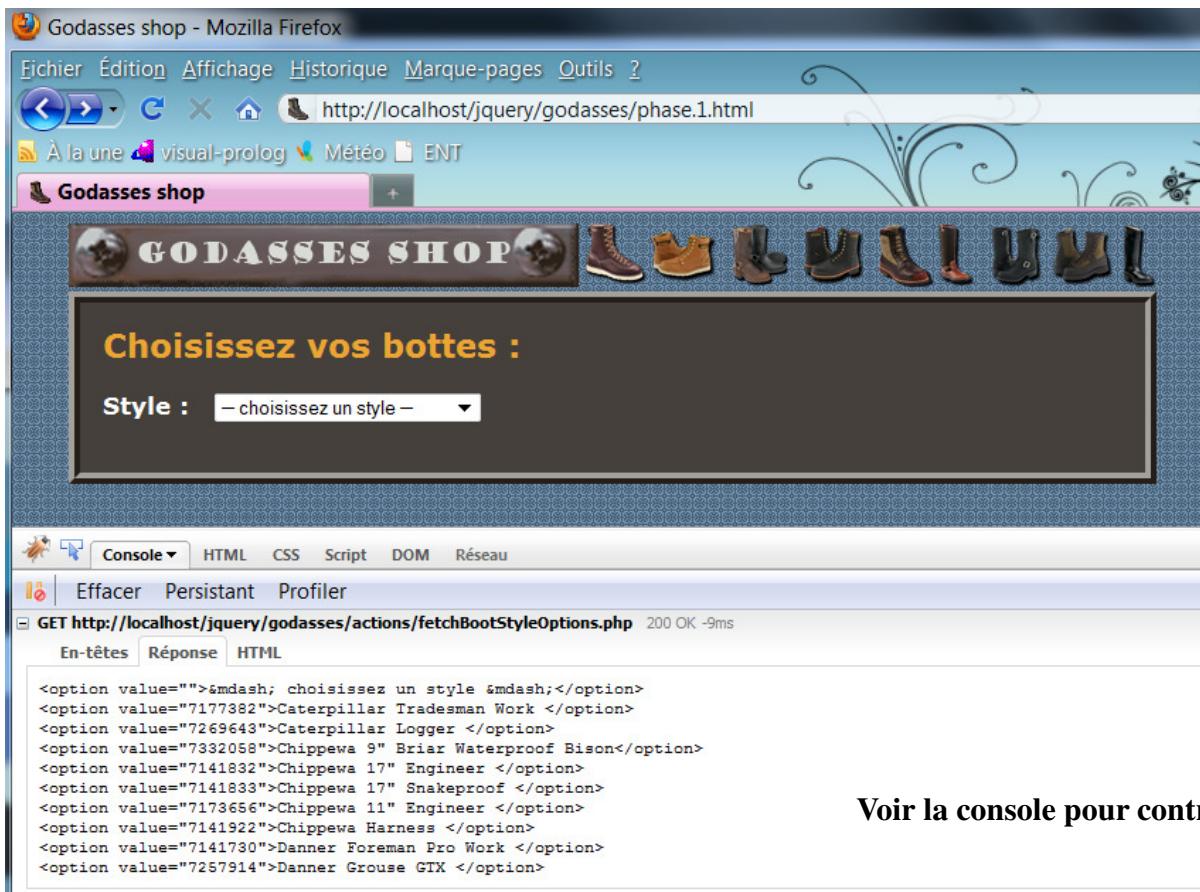
GET http://localhost/jquery/godasses/action/fetchBootStyleOptions.php 404 Not Found 10ms

En-têtes Réponse HTML

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /jquery/godasses/action/fetchBootStyleOptions.php was not found on this server.
</p>
</body></html>
```

# jQuery : Utilisation d'Ajax

## Mise au point : utilisation de firebug



The screenshot shows a Firefox browser window with the title "Godasses shop - Mozilla Firefox". The main content area displays a website for "GODASSES SHOP" featuring a row of various boots. Below this, a section titled "Choisissez vos bottes :" contains a dropdown menu with the placeholder text "Style : - choisissez un style -".

At the bottom of the browser window, the Firebug developer tool is open. The "Console" tab is selected, showing the following XMLHttpRequest:

```
GET http://localhost/jquery/godasses/actions/fetchBootStyleOptions.php 200 OK -9ms
En-têtes    Réponse    HTML
```

```
<option value="">&mdash; choisissez un style &mdash;</option>
<option value="7177382">Caterpillar Tradesman Work </option>
<option value="7269643">Caterpillar Logger </option>
<option value="7332058">Chippewa 9" Briar Waterproof Bison</option>
<option value="7141832">Chippewa 17" Engineer </option>
<option value="7141833">Chippewa 17" Snakeproof </option>
<option value="7173656">Chippewa 11" Engineer </option>
<option value="7141922">Chippewa Harness </option>
<option value="7141730">Danner Foreman Pro Work </option>
<option value="7257914">Danner Grouse GTX </option>
```

**Voir la console pour contrôler le fonctionnement d'Ajax**

Godasses shop - Mozilla Firefox

Fichier Édition Affichage Historique Marque-pages Outils ?

jQuery write less, do more.

À la une visual-prolog Météo ENT

Godasses shop

**GODASSES SHOP**

Choisissez vos bottes :

Style : – choisissez un style – ▾

Console ▾

HTML CSS Script DOM Réseau

Effacer Persistant Profiler

GET http://localhost/jquery/godasses/actions/fetchBootStyleOptions.php 200 OK -9ms

En-têtes Réponse HTML

```
<option value="">&mdash; choisissez un style &mdash;</option>
<option value="7177382">Caterpillar Tradesman Work </option>
<option value="7269643">Caterpillar Logger </option>
<option value="7332058">Chippewa 9" Briar Waterproof Bison</option>
<option value="7141832">Chippewa 17" Engineer </option>
<option value="7141833">Chippewa 17" Snakeproof </option>
<option value="7173656">Chippewa 11" Engineer </option>
<option value="7141922">Chippewa Harness </option>
<option value="7141730">Danner Foreman Pro Work </option>
<option value="7257914">Danner Grouse GTX </option>
```

Console ▾ HTML CSS Script DOM Réseau

Effacer Persistant Profiler

GET http://localhost/jquery/godasses/actions/fetchBootStyleOptions.php 20

En-têtes Réponse HTML

Réponse

```
Date Sun, 21 Nov 2010 12:40:23 GMT
Server Apache/2.2.11 (Win32) PHP/5.3.0
X-Powered-By PHP/5.3.0
Content-Length 575
Keep-Alive timeout=5, max=95
Connection Keep-Alive
Content-Type text/html
```

Requête

```
Host localhost
User-Agent Mozilla/5.0 (Windows; U; Windows NT 6.1; fr; rv:1.9.2.13) Gecko/20100907 Firefox/3.6.13
Accept text/html, */*
Accept-Language fr,fr-fr;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding gzip,deflate
Accept-Charset ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive 115
Connection keep-alive
X-Requested-With XMLHttpRequest
Referer http://localhost/jquery/godasses/phase.1.html
```

# jQuery : Utilisation d'Ajax

## Interaction/sélection : afficher les infos sur l'item sélectionné

```
$('#bootChooserControl').change(function(event) {
    $('#productDetailPane').load(
        'actions/fetchProductDetails.php',
        {style: $(event.target).val()},
        function() { $('[value=""]').remove(); }
    );
});
```

```
<div id="banner">
    
</div>

<div id="pageContent">

    <h1>Choisissez vos bottes :</h1>

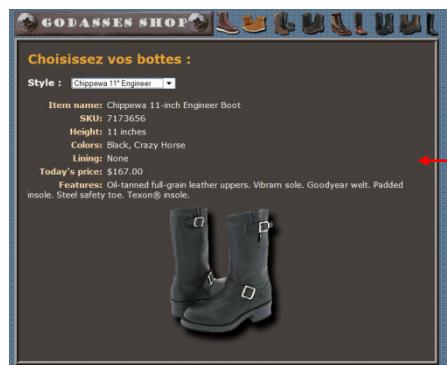
    <div>

        <div id="selectionsPane">
            <label for="bootChooserControl">Style : </label>&ampnbsp
            <select id="bootChooserControl" name="bootStyle"></select>
        </div>

        <div id="productDetailPane"></div>
    </div>
</div>
```

Chargement dans la div

*#productDetailPane*



# jQuery : Utilisation d'Ajax

- `load(url, paramètres, callback)`
  - Si non spécifié, l'appel ajax est un GET
  - Si spécifié
    - { `style` : 123456, `color` : 4, `name` : "jojo" }
    - POST avec les `variables` indiquées
  - `callback`
    - Fonction exécutée quand le transfert est terminé
    - `function(contenu, status, xmlhttprequest)`
      - Exécutée pour chaque élément multi ensemble
      - » `this`

"success"

# jQuery : Utilisation d'Ajax

```
$('#bootChooserControl').change(function(event) {
    $('#productDetailPane').load(
        'actions/fetchProductDetails.php',
        {style: $(event.target).val()},
        function() { $('[value=""]').remove(); }
    );
});
```

Éxécution d'un **POST** avec **style** comme paramètre  
valeur : **\$(event.target).val()**

---

```
<option value="">&mdash; choisissez un style &mdash;</option>
<option value="7177382">Caterpillar Tradesman Work </option>
<option value="7269643">Caterpillar Logger </option>
<option value="7332058">Chippewa 9" Briar Waterproof Bison</option>
<option value="7141832">Chippewa 17" Engineer </option>
<option value="7141833">Chippewa 17" Snakeproof </option>
```

# jQuery : Utilisation d'Ajax

**La sélection expédie une requête Php + la clé de l'objet**

**Coté serveur :**

**Php** : requête SQL avec la clé de l'objet  
récupération de la description de l'objet  
**Php** : fabrication du contenu HTML/description  
Expédition du code

**Coté Client :**

réception du code HTML à insérer dans la div  
exécution du callback



# jQuery : Utilisation d'Ajax

Accès BD ↑

```
$style = isset($_REQUEST['style']) ? $_REQUEST['style'] : 'default';
$item = $items[$style];
?>

<div>
    <label>Item name:</label> <?php echo $item['name']; ?>
</div>
<div>
    <label>SKU:</label> <?php echo $item['sku']; ?>
</div>
<div>
    <label>Height:</label> <?php echo $item['height']; ?>
</div>
<div>
    <label>Colors:</label> <?php echo $item['colors']; ?>
</div>
<div>
    <label>Lining:</label> <?php echo $item['lining']; ?>
</div>
<div>
    <label>Today's price:</label> <?php echo $item['price']; ?>
</div>
<div>
    <label>Features:</label> <?php echo $item['features']; ?>
</div>
<div align="center">
    <?php if ($style != 'default') { ?>
        
    <?php } ?>
</div>
```

## Exemple PHP

# jQuery : Utilisation d'Ajax

- Gestion des paramètres
  - POST : hashtable ou syntaxe JSON
  - GET :
    - `$.param(parametres)`

```
$.param({ 'a thing':'it&s=value',
          'another thing':'another value',
          'weirds': '!@#$%^&()_+'
        })
```

donne "a+thing=it%26s%3Dvalue&another+thing=anther+value&weirds..."

- À concaténer avec l'URL pour obtenir un GET syntaxiquement correct
- Forme de sérialisation

# jQuery : Utilisation d'Ajax

- Gestion des paramètres

- Javascript : encodeURIComponent

```
1 // encodes characters such as ?,=,/,&,:  
2 console.log(encodeURIComponent('?x=шельвы'));  
3 // expected output: "%3Fx%3D%D1%88%D0%B5%D0%BB%D0%BB%D1%8B"  
4  
5 console.log(encodeURIComponent('?x=test'));  
6 // expected output: "%3Fx%3Dtest"  
7
```

- jQuery : Cas d'un formulaire :

- `$(input).serialize()`

- Fabrique une chaîne de paramètres à partir des éléments de formulaire du multi ensemble

- `$(input).serializeArray()`

- Fabrique un tableau d'objets { name: , value : }
        - Possible de passer directement cet argument à `load`

# jQuery : Utilisation d'Ajax

- TD Ajax / jQuery / Php / MySQL
  - Réaliser en jQuery + Php + MySQL
  - Même modèle que *Godasses Shop* :
    - Base SQL de livres
      - Titre, auteur, genre, disponibilité, résumé, nombre d'exemplaires, image de couverture
    - Une page Web avec options/select pour le titre / pour l'auteur
      - Changement automatique des titres/auteurs avec Ajax
      - Chargement avec Ajax des informations (et affichage)
        - » Titre, auteur, genre, image etc..
      - Le chargement d'un nouveau texte se fait avec une transition graphique

# jQuery : Utilisation d'Ajax

GET ou POST ?

# jQuery : Utilisation d'Ajax

- GET ou POST ?
  - En théorie...
    - Get pour récupérer une page Web
      - Mise en cache
      - Toujours même réponse
    - Post pour envoyer des données au serveur
      - En fonction des valeurs, pages différentes

# jQuery : Utilisation d'Ajax

- Load = Get/Post + chargement
  - `$.get(url, paramètres, callback, type)`
    - Paramètres : hashtable
    - `callback( contenu, status, XHR )`
    - `type`
      - html
      - text
      - xml
      - json
      - script

# jQuery : Utilisation d'Ajax

```
..... .change(function(event) {  
    $.get(  
        'fetchProduitDetails',  
        {style: $(event.target).val() },  
        function(reponse) {  
            $('#productDetailPane').html(reponse);  
            $('[value=""'], event.target).remove();  
        }  
    );  
});
```

# jQuery : Utilisation d'Ajax

- Get & XML
  - `$.get(url, paramètres, callback, type)`
    - `callback( contenu, status, XHR )`
    - `type`
      - `xml`
    - Contenu est la racine de l'arbre DOM du document XML

# jQuery : Utilisation d'Ajax

- Get & json : désérialisation
  - `$.getJSON(url, paramètres, callback)`
    - `callback( contenu, status, XHR )`
    - Le contenu de l'URL est interprété comme une chaîne JSON
    - Contenu est l'objet fabriqué par JSON
    - Pas de paramètre type (implicitement JSON)

```
$.getJSON('ajax/test.json', function(data) {  
    var items = [];  
  
    $.each(data, function(key, val) {  
        items.push('<li id="' + key + '">' + val + '</li>');  
    });  
  
    $('<ul/>', {  
        'class': 'my-new-list',  
        html: items.join("")  
    }).appendTo('body');  
});
```



Conversion tableau en  
chaine (séparateur)

```
<!DOCTYPE html>
<html>
<head>
<style>img{ height: 100px; float: left; }</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
<div id="images">

</div>
<script>
$.getJSON("http://api.flickr.com/services/feeds/photos_public.gne?jsoncallback=?",
{
  tags: "bike",
  tagmode: "any",
  format: "json"
},
function(data) {
  $.each(data.items, function(i,item){
    $("<img/>").attr("src", item.media.m).appendTo("#images");
    if ( i == 3 ) return false;
  });
}
);
</script>

</body>
</html>
```

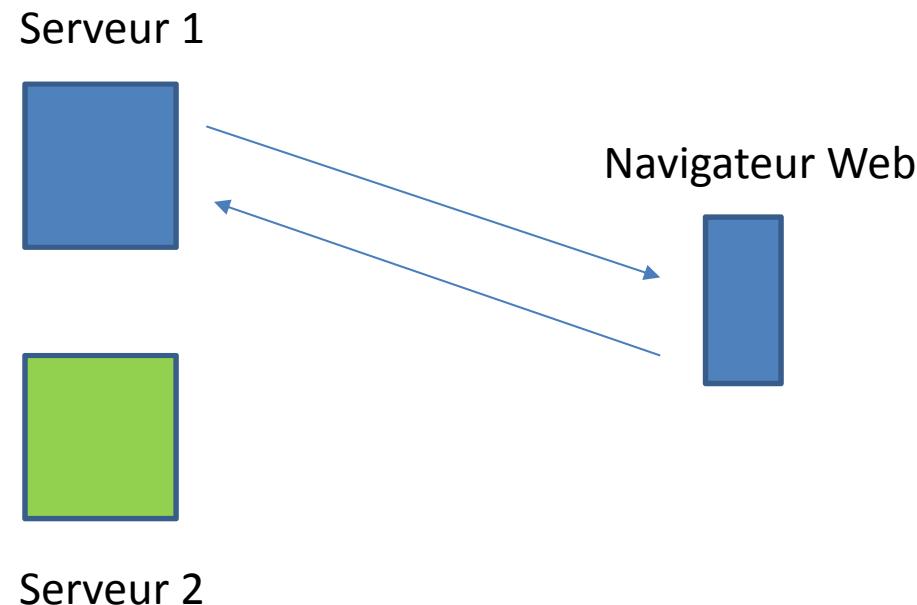
# Cross domain

*cross domain* (non : niveau 1 XMLHttpRequest)

Impossible de faire une requête croisée (sur un autre site)

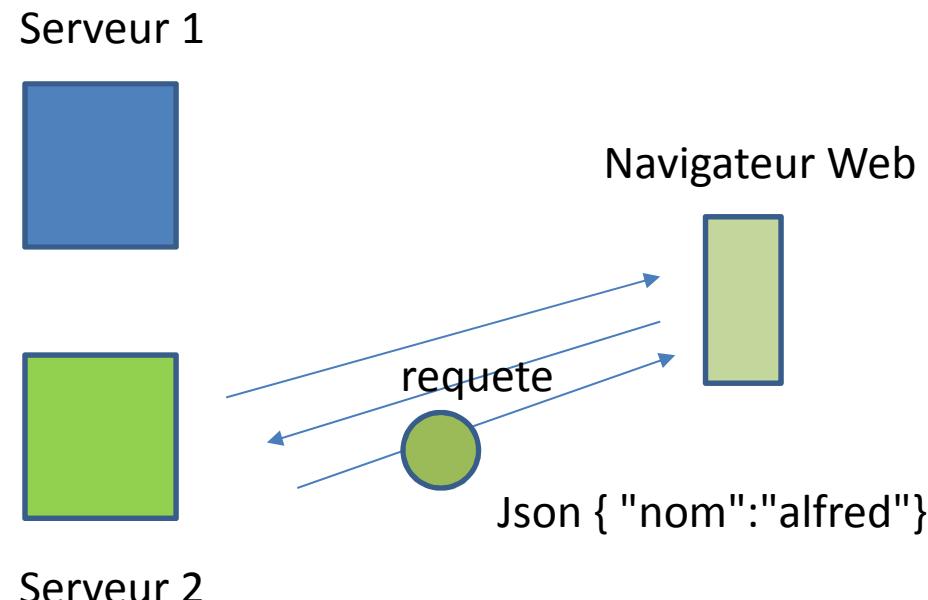
monServer.com -> page web avec JS : ce JS ne peut faire de requêtes sur jojoServer.com

Restriction : seulement sur monServer.com



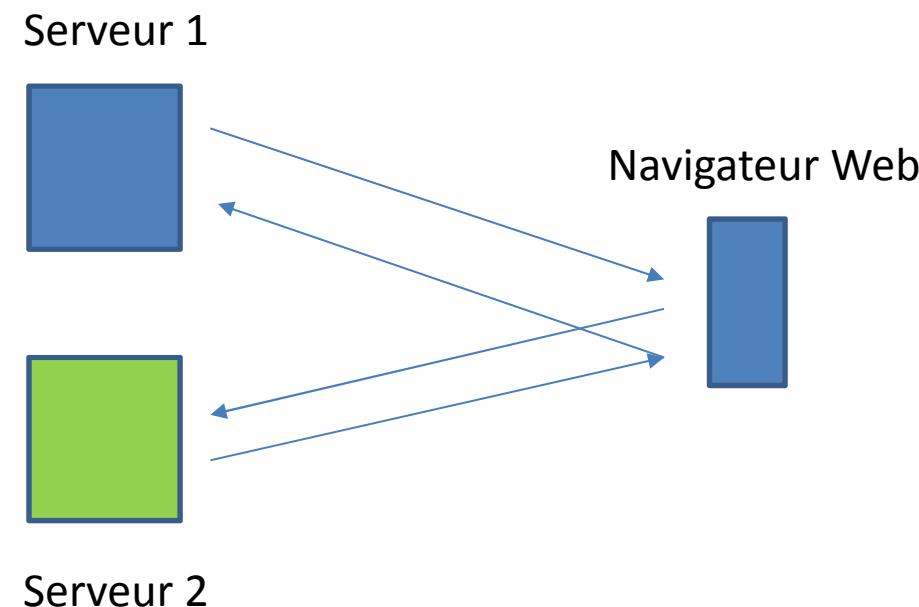
# Cross domain

*Supposons que le Serveur 2 contienne une donnée JSON { "nom" : "alfred" } générée par l'url Serveur2/requete*



# Cross domain

*On veut récupérer cette information JSON à partir d'une page qui vient de S1*



# JSON / JSONP

*Au niveau de S2, on fabrique un texte de fonction à exécuter :*

**maFonction( { "nom" : "alfred" } )**

JSON + Padding = JSONP

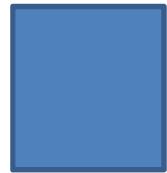
Dans la page bleu :

```
<script>
    function maFonction(data) {
        console.log(data.nom) // on a le JSON
    }
</script>
<script src="http://www.serveur2.com/requete"></script>
```

# JSON / JSONP

*On veut récupérer cette information JSON à partir d'une page qui vient de S1*

Serveur 1

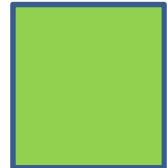


Navigateur Web

```
<script>
    function maFonction(data) {
        console.log(data.nom) // on a le JSON sans le P
    }
</script>
<script src="http://www.serveur2.com/requete"></script>
```

*maFonction( { "nom" : "alfred" } )*

Serveur 2



# JSON / JSONP

Ou bien en jQuery, on peut écrire :

```
<script>

function maFonction(data){
    // à vous de voir ce que vous faites des données
}

$(document).ready(function() {
    $.getJSON(" http://www.serveur2.com/requete?maFonction=?");
});

</script>
```

```
<!DOCTYPE html>
<html>
<head>
<style>img{ height: 100px; float: left; }</style>
<script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
<div id="images">

</div>
<script>
$.getJSON("http://api.flickr.com/services/feeds/photos_public.gne?jsoncallback=?",
{
  tags: "bike",
  tagmode: "any",
  format: "json"
},
function(data) {
  $.each(data.items, function(i,item){
    $("<img/>").attr("src", item.media.m).appendTo("#images");
    if ( i == 3 ) return false;
  });
});</script>

</body>
</html>
```

```
$.getJSON("http://api.flickr.com/services/feeds/photos_public.gne?jsoncallback=?",
{
  tags: "paris louvre",
  tagmode: "any",
  format: "json"
},
function(data) {
  $.each(data.items, function(i,item){
    $("<img/>").attr("src", item.media.m).appendTo("#un");
  });
});
```

# JSONP

From Wikipedia, the free encyclopedia

**JSONP** (**JSON** with **Padding** or **JSON-P**)<sup>[1]</sup> is a javascript pattern to request data by loading by Bob Ippolito in 2005.<sup>[2]</sup> JSONP enables sharing of data bypassing [same-origin policy](#). The [JavaScript](#) to read media [DOM](#) elements or [XHR](#) data fetched from outside the page's origin scheme, port number and host name identifies as its origin.

## Contents [hide]

- [1 How JSONP works](#)
- [2 Script element injection](#)
- [3 Security concerns](#)
  - [3.1 Untrusted third-party code](#)
  - [3.2 Callback name manipulation and reflected file download attack](#)
    - [3.2.1 Cross-site request forgery](#)
  - [3.3 Rosetta Flash](#)
- [4 History](#)
- [5 See also](#)
- [6 References](#)
- [7 External links](#)

## How JSONP works [edit]

The HTML `<script>` element is allowed to execute content retrieved from foreign origins. JSON data were not able to share the data across domain before the adoption of CORS (Cross-Origin Resource Sharing). For example, a request to a foreign service <http://server.example.com/Users/1234> must be made via JSONP.

# jQuery : Utilisation d'Ajax

- `$.ajax(options)`
  - Options : hashtable
    - url :
    - type : GET /POST (GET par défaut)
    - data : hashtable de propriétés/valeurs
    - dataType :
      - xml : un dom est passé au callback
      - html : code ( les <script> sont évalués)
      - json : un objet est passé au callback
      - script
      - text

# jQuery : Utilisation d'Ajax

- `$.ajax(options)`
  - Options : hashtable
    - cache : false : pas de mise en cache
    - timeout : en ms (le callback d'erreur est appelé)
    - success : callback succès
    - error : callback erreur
    - complete
    - beforeSend
    - async : true/false
    - username
    - password
    - xhr
    - Etc...

## `$.ajaxSetup(options)`

# jQuery : Utilisation d'Ajax

- Evenements & Ajax
  - ajaxStart
  - ajaxStop
  - ajaxSend
  - ajaxSuccess
  - ajaxError
  - ajaxComplete

```
$('body').on(  
    'ajaxStart ajaxStop ajaxSend ajaxSuccess ajaxError ajaxComplete',  
    function(event) {  
        $('#console').append(event.type+"<br>");  
    }  
);
```

# jQuery et Ajax

- Encapsulation d'Ajax
- Callbacks
- Evenements
  - Body etc..
  - Div etc..
- Simplification des appels XHR
- Portabilité
- Timeout
- Implémentation timeout dans Ajax/Javascript ?

# jQuery et jQuery UI

- Librairie de plugins
- Taille importante
- <http://jqueryui.com/download>
- Page web : création du paquet
- Zip :
  - Index.html
    - Demo des widgets sélectionnées
  - /css
  - /development-bundle
  - /js
    - Jquery core + jquery UI

# jQuery UI

- Utilisation très importante de CSS
- jQuery : noms de classe ui-...
  - ui-state-active
  - ui-autocomplete
  - ui-widget
    - ui-widget-header
    - ui-widget-content
- Icons
  - Image / grille d'icônes
    - Rapidité / cache client

# jQuery UI

- Notion de thèmes
- Modifier les descriptions CSS
- <http://jqueryui.com/themeroller>
- Aide à la conception / customisation des éléments d'interface
  - Préférable à la modification sauvage de CSS

# jQuery UI

- Animations jQuery Core
- jQuery UI :
  - `effect(type, options, vitesse, callback)`
    - Type : blind, bounce, clip, drop, explode, fade, fold...
    - Options : les mêmes que pour `animate`
    - Vitesse : en ms ou bien 'slow', 'normal', 'fast'
    - Callback : appel à la fin de l'animation
  - Extension des fonctions de base de jQuery Core :
    - `show(effet, options, vitesse, callback)`
    - `hide...`
    - `toggle...`

# jQuery UI

- Extension des fonction jQuery Core de changement de classe
  - Morphing / blending sur les paramètres CSS
- addClass(classe, vitesse, easing, callback)
- removeClass...
- toggleClass...
- switchClass(enlever, ajouter, vitesse, easing, callback)

# jQuery UI

- Positionnement des objets :

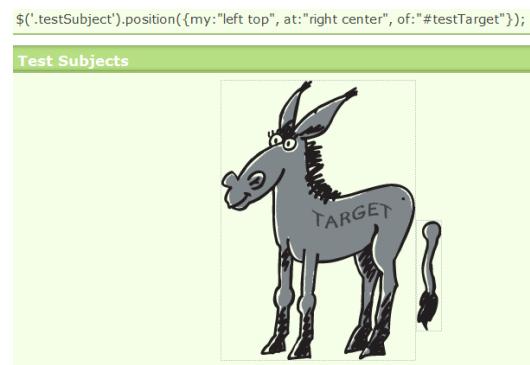
```
$("#monElement").css({
    position : 'absolute',
    top: 200,
    left : 200
});
```

- jQuery UI : extension : 'position'

- at :
- my :
- of :

```
$('#monElement').position({
    my: 'top center',
    at: 'bottom right',
    of: '#unAutreElement'
});
```

- offset : '10 20'
- collision / using



# jQuery UI

- Déplacer des objets
- `$(...).draggable(options)`
  - disable
  - enable
  - destroy
  - draggable('option', nomOption, Valeur)
  - Hashtable
- Rajoute la classe CSS *ui-draggable*



C10

# jQuery UI

- Déplacer des objets
- *Événements*
  - *dragstart*
    - *Début déplacement*
  - *drag*
    - *Constamment durant le déplacement*
  - *dragstop*



C10

# jQuery UI

- 'Droppable'
- *Un objet qui régit quand on 'lache' un **draggable** dessus*

## Draggable options

**accept:**  flower  dog  motorcycle  water

**activeClass:**  none  redBorder  blueBorder  greenBorder

**hoverClass:**  none  bronze  silver  gold

**tolerance:**  none  intersect  pointer  fit  touch

# jQuery UI

- 'Droppable'
- Evénements
  - `dropactivate` (option activate)
  - `dropdeactivate` (option deactivate)
  - `dropover` (option over)
  - `dropout` (option out)
  - `drop` (option drop)

# jQuery UI

- 'selectable'
- `$('#test').resizable('enable')`



**Resizing options**

**alsoResize:**  unspecified  other test subject

**animate:**  unspecified  true  false

**animateDuration:**  unspecified  slow  normal  fast  milliseconds:

**aspectRatio:**  unspecified  false  true  float:

**autoHide:**  unspecified  true  false

**containment:**  none  document  parent  window

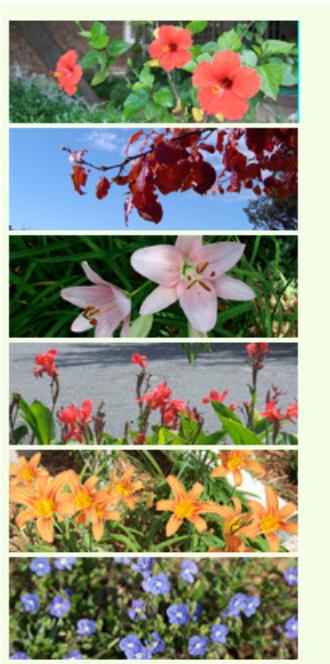
**ghost:**  unspecified  true  false

**handles:**  n  ne  e  se  s  sw  w  nw  
 **minHeight:**  **minWidth:**  **maxHeight:**  **maxWidth:**

(possible de créer soi même les poignées via CSS)

# jQuery UI

- 'Sortable'
- Appliquer `$('#test').sortable('enable')` au parent de la liste à trier.



**sort**  
**sortactivate**  
**sortbeforestop**  
**Etc..**

# jQuery UI

- 'Resizable'
- \$('#test').selectable('enable')

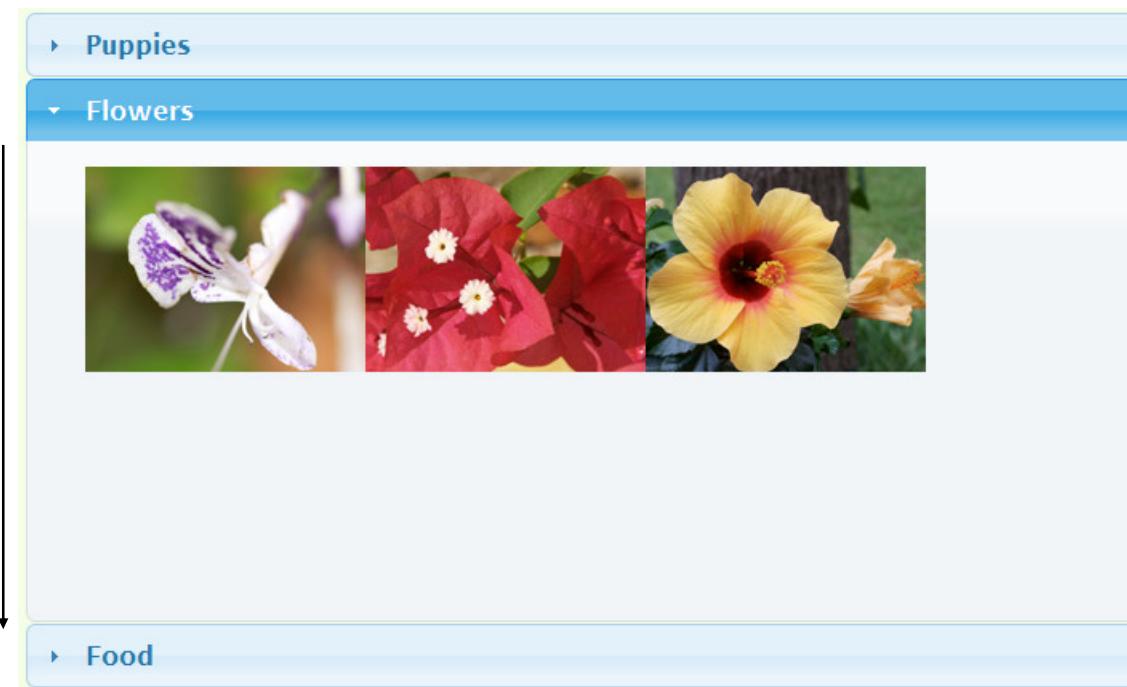
**Événements :**

**resizablestart (start)**  
**resize (resize)**  
**resizestop (stop)**

Name	Year	Binder	Page	Slot	Viewed?
20 Million Miles to Earth	1957	4	12	1	X
2001: A Space Odyssey	1968	2	20	4	X
Alien	1979	23	1	3	X
Aliens	1986	23	1	4	X
Alien Nation	1988	17	2	2	X
Blade Runner	1982	12	22	4	X
Close Encounters of the Third Kind	1977	3	5	1	X
Cocoon	1985	11	13	3	X
Communion	1989	4	2	1	
Creature from the Black Lagoon	1954	6	1	4	X
Day of the Triffids	1962	7	17	2	X
Dragonslayer	1981	8	3	3	X
Dreamscape	1984	10	3	1	X

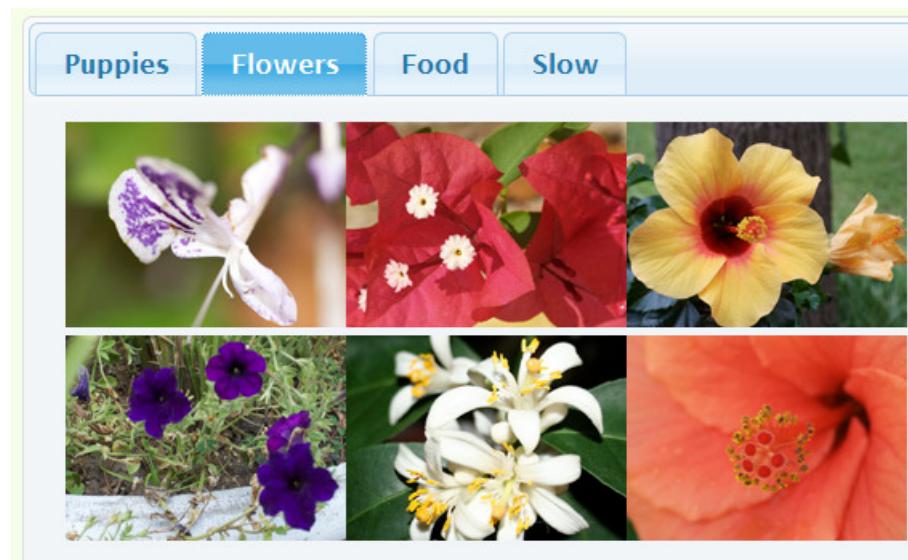
# jQuery UI

- Contrôles HTML supplémentaires
  - Accordéons



# jQuery UI

- Contrôles HTML supplémentaires
  - Tabs



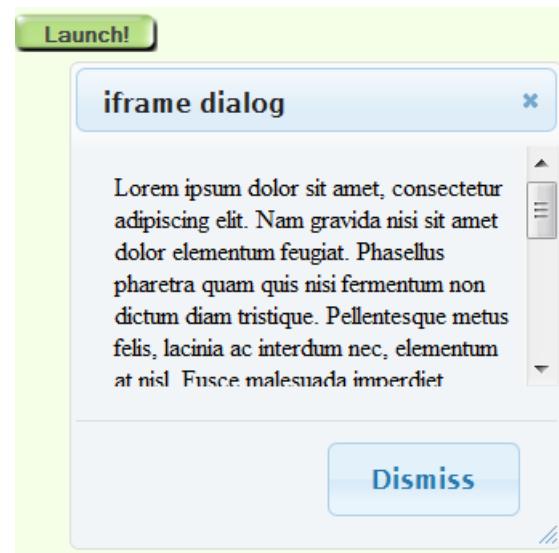
# jQuery UI

- Contrôles HTML supplémentaires
  - Sliders



# jQuery UI

- Contrôles HTML supplémentaires
  - Boites de dialogue



# jQuery UI

- Contrôles HTML supplémentaires
  - Calendrier (datepicker)



# jQuery UI

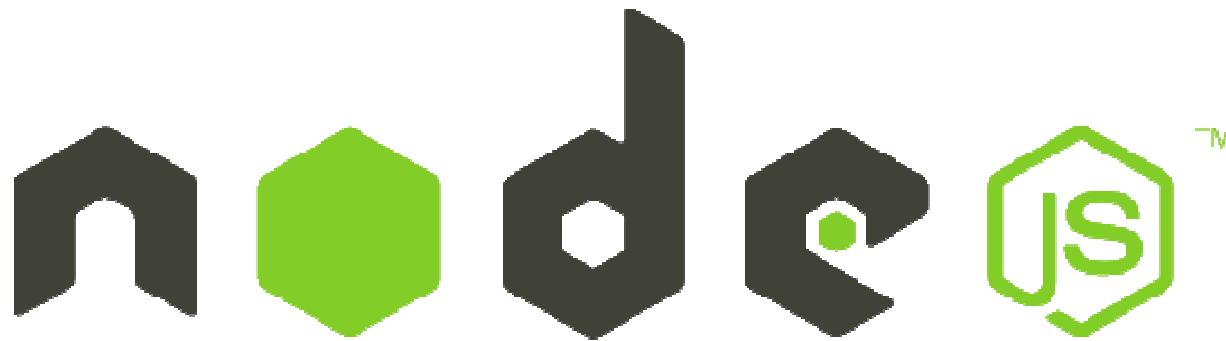
- Contrôles HTML supplémentaires
  - Boutons
  - Auto completer
  - Barres de progressions etc..

# jQuery : conclusion

- Javascript + DomScript + interpréteur CSS 3
- jQuery Core
  - Support d'Ajax
- Extensions interfaces avec jQuery UI
- Pattern de programmation par réécriture de multi ensembles
- Utilisation massive de CSS
  - Sélecteurs
  - Classes et comportements

# jQuery : conclusion

- 3 manières d'utiliser jQuery
  - Seulement les sélecteurs
    - Puis Javascript pour les actions
  - Sélecteurs + actions
  - Extension et UI
- Bonne séparation du code, feuille de style et code HTML
  - Plus 'propre' que Javascript
  - Plus portable



Gildas Ménier  
[gildas.menier@univ-ubs.fr](mailto:gildas.menier@univ-ubs.fr)



# Node.js

- En résumé :

- Interpréteur Javascript :
  - WebAssembly (WASM) unity
  - JIT /JsVM
- Ryan Dahl
- Interpréteur V8 Google
- + partie client / serveur
- Pourquoi faire :
  - web services (REST)
  - jeux multijoueur temps réels
  - communication multiclients
  - applications Web
- Un seul langage serveur et client
- Passage à l'échelle

The screenshot shows the WasmFiddle interface for WebAssembly development. It features two main code editors: one for C and one for JavaScript (JS). Below the editors are tabs for different formats: Text Format, Wat, Wasm, and Output.

**C Editor:**

```
int main() {
    return 42;
}
```

**JS Editor:**

```
var wasmModule = new WebAssembly.Module(wasmCode);
var wasmInstance = new WebAssembly.Instance(wasmModule, wasmImports);
log(wasmInstance.exports.main());
```

**Output Tab:**

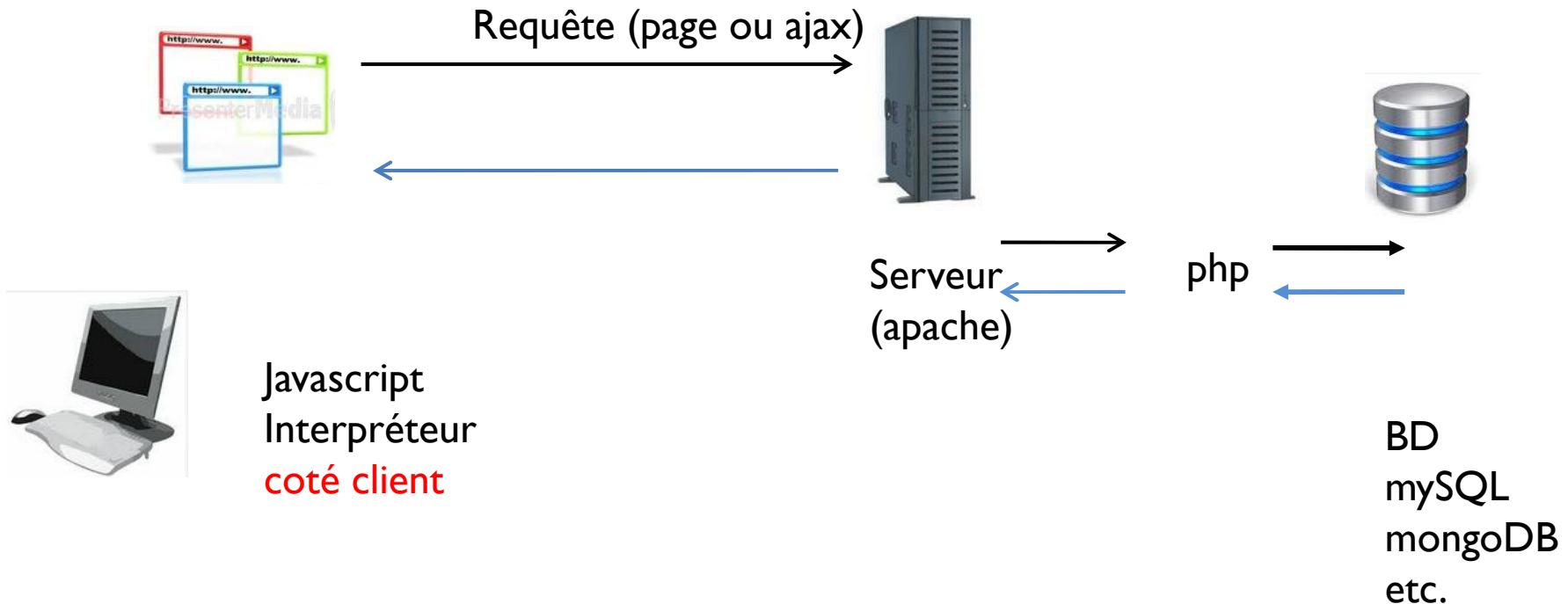
```
(module
  (table 0 anyfunc)
  (memory $0 1)
  (export "memory" (memory $0))
  (export "main" (func $main))
  (func $main (; 0 ;) (result i32)
    (i32.const 42)
  )
)
```

```
wasm-function[0]:  
    sub rsp, 0x18          ; 0x0000000 48 83 ec 18  
    mov qword ptr [rsp + 8], r14 ; 0x0000004 4c 89 74 24 08  
    mov rax, rsp           ; 0x0000009 48 8b c4  
    add rax, 0              ; 0x000000c 48 05 00 00 00 00  
    cmp qword ptr [r14 + 0x28], rax ; 0x000012 49 39 45 28  
    jae 0x27               ; 0x000016 0f 83 0b 00 00 00  
0x00001c:  
    mov eax, 0x2a           ; 0x00001c b8 2a 00 00 00  
    jmp 0x30               ; 0x000021 e9 0a 00 00 00  
0x000026:  
    int3                  ; 0x000026 cc  
0x000027:  
    add rsp, 0x10           ; 0x000027 from: [0x000016]  
    jmp 0x61               ; 0x00002b e9 31 00 00 00  
0x000030:  
    mov r14, qword ptr [rsp + 8] ; 0x000030 4c 8b 74 24 08  
    nop                   ; 0x000035 66 90  
    add rsp, 0x18           ; 0x000037 48 83 c4 18  
    ret                   ; 0x00003b c3
```



# Node.js

- Principe : interpréteur Javascript





# Node.js

## ● Principe





# Node.js

- Principe
  - Framework de développement
    - basé sur le compilateur / interpréteur V8 de google
  - Ecriture du code serveur ET code de gestion de page
    - en javascript
  - Javascript côté client et serveur
  - *Un seul thread de traitement*
    - Pas de performance si pas d'asynchronisme !!!!
  - Extensible
  - Vitesse de développement



[HOME](#) | [ABOUT](#) | [DOWNLOADS](#) | [DOCS](#) | [GET INVOLVED](#) | [SECURITY](#) | [NEWS](#)

Node.js® is a JavaScript runtime built on [Chrome's V8 JavaScript engine](#).

## Download for Windows (x64)

**12.16.1 LTS**

Recommended For Most Users

**13.10.1 Current**

Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#)

[Other Downloads](#) | [Changelog](#) | [API Docs](#)

Or have a look at the [Long Term Support \(LTS\) schedule](#).

Sign up for [Node.js Everywhere](#), the official Node.js Monthly Newsletter.



# Node.js

- Interpréteur REPL (*Read Eval Print Loop*)

```
C:\Users\gildas>node
> console.log("ca roule")
ca roule
undefined
> .help
break  Sometimes you get stuck, this gets you out
clear  Alias for .break
exit   Exit the repl
help   Show repl options
load   Load JS from a file into the REPL session
save   Save all evaluated commands in this REPL session to a file
> .exit

C:\Users\gildas>
```

- node fichier.js



# Node.js

- Javascript
  - Programmation fonctionnelle
  - Définition de fonctions anonymes
  - Asynchronisme
    - Méthodes 'callback' (ou listeners) / promesses § async
- Programmation Client + Serveur
  - **HTTP RFC 7230-7237** à étudier !!!
  - HTTP est à connaître
- Sockets



# Node.js

- **HTTP RFC 7230-7237 à étudier !!!**
- HTTP est à connaître



# Node.js

## • HTTP exemple

HTTP protocole de base du Web  
Voir partie Php Web  
HTTP à connaître

Client : socket sur le port 80 URL <http://www.linux.org/>

**GET /index.html HTTP/1.1**

**Connection: Keep-Alive**

**User-Agent: Mozilla/4.7 [en] (WinNT; I)**

**Host: [www.linux.org](http://www.linux.org)**

**Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, \*/\***

**Accept-Encoding: gzip**

**Accept-Language: en**

**Accept-Charset: iso-8859-1,\* ,utf-8**

**Ligne vide**



# Node.js

## • HTTP exemple

HTTP protocole de base du Web  
Voir partie Php Web  
[HTTP à connaître](#)

Réponse du serveur (socket port 80)

**HTTP/1.1 200 OK**

**Date: Thu, 09 Dec 1999 12:23:29 GMT**

**Server: Apache/1.3.9 (Linux Debian 2.2) ApacheJServ/1.0**

**Last-Modified: Mon, 04 Oct 1999 09:33:15 GMT ETag: "0-374-37f8745b"**

**Accept-Ranges: bytes**

**Content-Length: 884**

**Content-Type: text/html**

**<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"> [...]**



# Node.js

## •Base Web

<http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html>

W3 Consortium : <http://www.w3.org/>

RFC catalogue IETF : <http://www.ietf.org/rfc.html>

RFC 2616: HTTP specifications

RFC 1630: URI definition.

RFC 1866: HTML specifications

RFC 2045: MIME extensions.

RFC 2046: MIME types.

RFC 2109: Cookie standard draft.

RFC 821: SMTP Simple Mail Transfert Protocol.

RFC 4315 : IMAP Internet Message Access Protocol



# Node.js

```
1 var http = require('http');
2
3 http.createServer(function (req, res) {
4     res.writeHead(200, {'Content-Type': 'text/plain'});
5     var d = new Date();
6     res.end('Il est '+d.getHours()+':'+d.getMinutes()+'\n');
7 }) .listen(8080, '127.0.0.1');
8
9 console.log('Le serveur est accessible au http://127.0.0.1:8080/');
```

Réponse du serveur (socket port 80)

**HTTP/1.1 200 OK**

Date: Thu, 09 Dec 1999 12:23:29 GMT

Server: Apache/1.3.9 (Linux Debian 2.2) ApacheJServ/1.0

Last-Modified: Mon, 04 Oct 1999 09:33:15 GMT ETag: "0-374-37f8745b" serveur1.js

Accept-Ranges: bytes

Content-Length: 884

Content-Type: text/plain

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"> [...]



# Node.js

```

1 var http = require('http');
2
3 http.createServer(function (req, res) {
4     res.writeHead(200, {'Content-Type': 'text/plain'});
5     var d = new Date();
6     res.end('Il est '+ d.getHours() + ':' + d.getMinutes() + '\n');
7 }).listen(8080, '127.0.0.1');
8
9 console.log('Le serveur est accessible au http://127.0.0.1:8080/');

```

accès au module 'http'

Fonction anonyme  
req : requête  
res : réponse

écriture HTTP

réponse

http.createServer(...).listen

C:\Users\gildas\Desktop\node>node serveur1.js  
Le serveur est accessible au http://127.0.0.1:8080/



# Node.js

- Ecriture en même temps

- Serveur
- Génération des pages
- Gestion du *backend*

- Asynchrone

- La fonction définit le comportement du serveur
  - Requête
  - Réponse

- Simple

- HTTP !

```
require('http').createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  var d = new Date();
  res.end('Il est ' + d.getHours() + ':' + d.getMinutes() + '\n');
}).listen(8080, '127.0.0.1');
```



# Node.js

- **req**

- req.url
- req.protocol
- req.ip
- req.path
- req.host
- req.method
- etc..

typescript

```
require('http').createServer(function (req, res) {  
    res.writeHead(200, {'Content-Type': 'text/plain'});  
    var d = new Date();  
    res.end('Il est '+d.getHours()+ ':' + d.getMinutes()+'\n');  
}).listen(8080, '127.0.0.1');
```

- **res**

- res.write(...)
- res.writeHead(...)
- res.end



# Node.js

## •Require ?

- Javascript pour node
- En fait, pas vraiment du js

Fichier mymodule.js

```
var myFunc1 = function() { ... };
var myFunc2 = function() { ... };
exports.myFunc1 = myFunc1;
exports.myFunc2 = myFunc2;
```

- j = require("mymodule")

```
var m = require('mymodule');
m.myFunc1();
```

exports / module.export



# Node.js

- Packages node.js
  - `http = require("http")`
  - Node package manager
    - Npm
  - Librairies en ligne de packages
  - http package bas niveau
    - HTTP



# Node.js

```
var url = require('url')
var resParse = url.parse("http://user:pass@host.fr:80/dir1/dir2/?question=test#hash")

console.log(resParse)
```

```
< protocol: 'http',
  slashes: true,
  auth: 'user:pass',
  host: 'host.fr:80',
  port: '80',
  hostname: 'host.fr',
  hash: '#hash',
  search: '?question=test',
  query: 'question=test',
  pathname: '/dir1/dir2/',
  path: '/dir1/dir2/?question=test',
  href: 'http://user:pass@host.fr:80/dir1/dir2/?question=test#hash' >
```



# Node.js

- Format

```
console.log(url.format(resParse))
```

- querystring

```
http://user:pass@host.fr:80/dir1/dir2/?question=test#hash
```

- qs.stringify(params)

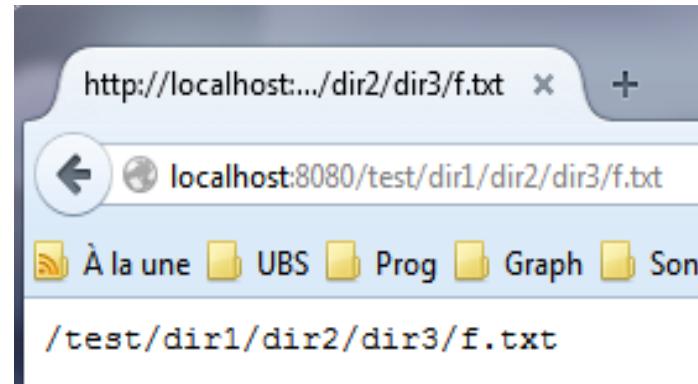
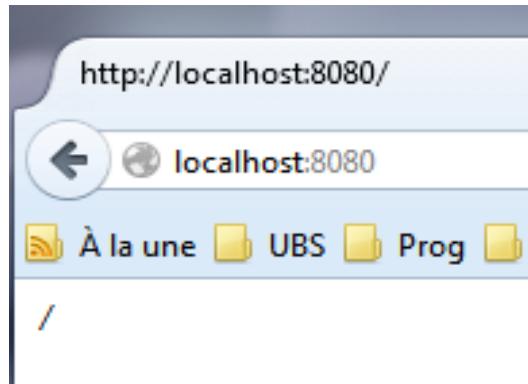
- Refabrique la chaîne

```
var qs = require ('querystring')
var params =
qs.parse("nom=jojo&prenom=bob&ville=toulouse")
// params = { nom:'jojo', prenom:'bob', ville:'toulouse'}
```

# Node.js

```
http.createServer(function (req, res) {  
    res.writeHead(200, {'Content-Type': 'text/plain'});  
    res.end(req.url);  
}).listen(8080);
```

- Package http



- Utilisation des packages url  
etc.

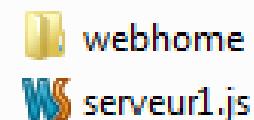


# Node.js

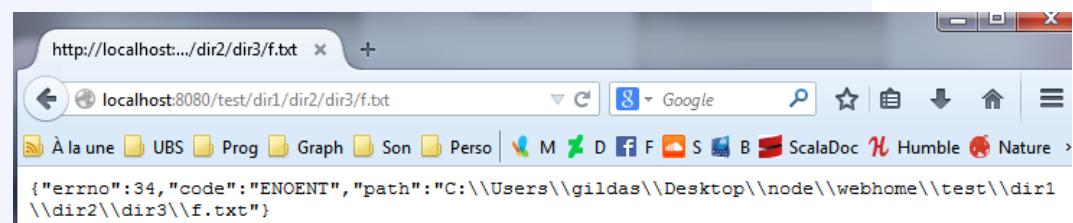
```
var fich = require('fs')
var url = require('url')
var http = require('http')
var repBase = "webhome"

http.createServer(function (req, res) {
    var reqParse = url.parse(req.url)
    fich.readFile(repBase+ reqParse.pathname, function(err,data) {
        if (err) {
            res.writeHead(404);
            res.end(JSON.stringify(err))
            return;
        }
        res.writeHead(200);
        res.end(data);
    })
}).listen(8080);

console.log('Le serveur est accessible au http://127.0.0.1:8080/');
```

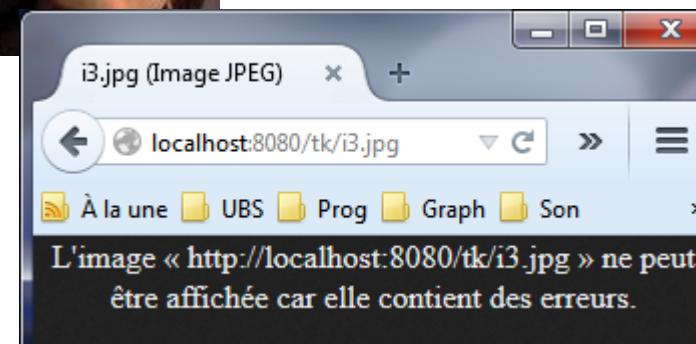
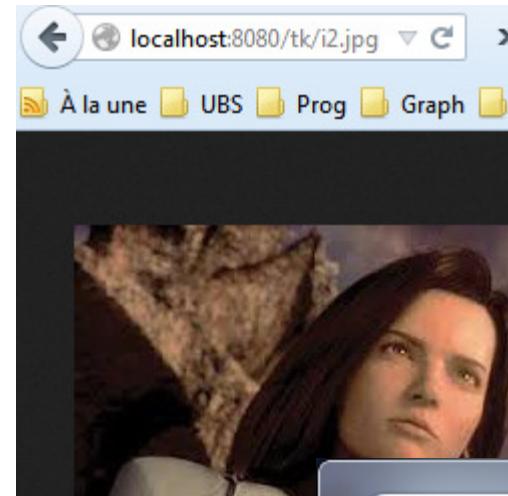
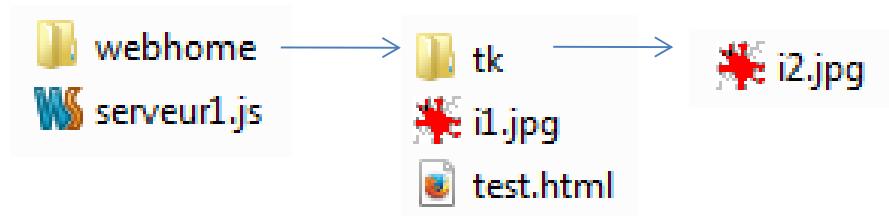
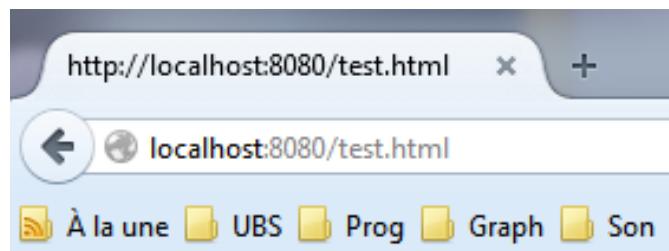


ASYNCHRONE





# Node.js





# Node.js

- package fs

- Écriture / lecture synchrone ou pas
- Passage de fonctions callback  
(fonctionnel !...)
- `open` / `openSync`
- `writeFile` / `writeFileSync`
- `readFile` / `readFileSync`
- Streams
- `exists`, `readdir`, `readdirSync`, `unlink`,
- `unlinkSync`, `mkdir`, `mkdirSync`,
- `rename`,
- `renameSync`, `watchFile` etc..

EVITER LES SYNC !

# Node.js

SERVEUR

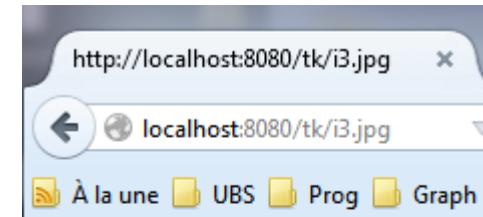
- Serveur dynamique
  - Construction du résultat en fonction d'une requête (bd ou autre)

```

var http = require('http')
var infos = [ 'et voila',
              'des infos',
              'qui viennent',
              'de loin']

http.createServer(function(req, res) {
    res.setHeader("Content-Type", "text/html")
    res.writeHead(200)
    res.write("<html>")
    res.write("<body>")
    for(var i in infos) {
        | res.write("<h1> " + infos[i] + "</h1>")
    }
    res.end("</body></html>")

}).listen(8080)
  
```



**et voila**

**des infos**

**qui viennent**

**de loin**



# Node.js

```
var http = require('http')
var acces = {
  hostname: 'localhost',
  port: '8080'
}

http.request(acces, function(reponse) {

  var donneeServeur = ''
  reponse.on('data', function(morceau) {
    donneeServeur += morceau
  })

  reponse.on('end', function() {
    console.log("status : ", reponse.statusCode)
    console.log("headers: ", reponse.headers)
    console.log("donnees: ", donneeServeur)
  })
}) .end()
```

REQUETE (client)

C:\Users\gildas\Desktop\node>node client.js

```
status : 200
headers: { 'content-type': 'text/html',
  date: 'Tue, 09 Dec 2014 09:43:17 GMT',
  connection: 'keep-alive',
  'transfer-encoding': 'chunked' }
donnees: <html><body><h1> et voila</h1><h1> des infos</h1><h1> qui viennent</h1>
<h1> de loin</h1></body></html>
```



# Node.js

- Package `net`
  - Sockets services
  - `net.socket`
  - `net.server`
    - Socket TCP
  - TLS
  - SSL etc..



# Node.js

- Autres packages

- process
- os
- util
- dns
- mongodb
- express

# npm



- Installation de packages node.js
- Réinstaller les dépendances pour chaque projet
  - Avantage : versions
  - Déploiement (et localisation)
  - Inconvénient : place
- Ou utiliser l'installation globale (-g)



# npm



- Installation de packages node.js
- Pas seulement !!!!
- Installé avec node.js
- **npm i -g express**
  - Installation des librairies en global
  - Installation de binaires exécutables (global)
  - Par exemple
    - **sudo npm i --g cordova**
    - Commande cordova rajoutée au bash
    - **npm i cordova**
      - Seulement répertoire local (pas d'exe)
      - Hiérarchie
      - **node\_modules**



Remarque :

```
C:\Users\menier>npm
npm WARN npm npm does not support Node.js v11.11.0
npm WARN npm You should probably upgrade to a newer version of node as we
npm WARN npm can't make any promises that npm will work with this version.
npm WARN npm Supported releases of Node.js are the latest release of 4, 6, 7, 8
9.
npm WARN npm You can find the latest version at https://nodejs.org/
```

Remarque :

```
C:\Users\menier>npm cache clean -f
npm WARN npm does not support Node.js v11.11.0
npm WARN You should probably upgrade to a newer version of node as
npm WARN can't make any promises that npm will work with this versi
npm WARN Supported releases of Node.js are the latest release of 4,
9.
npm WARN You can find the latest version at https://nodejs.org/
npm WARN using --force I sure hope you know what you are doing.

C:\Users\menier>npm install -g npm@latest
npm WARN npm does not support Node.js v11.11.0
npm WARN You should probably upgrade to a newer version of node as
npm WARN can't make any promises that npm will work with this versi
npm WARN Supported releases of Node.js are the latest release of 4,
9.
npm WARN You can find the latest version at https://nodejs.org/
C:\Users\menier\AppData\Roaming\npm\npx -> C:\Users\menier\AppData\Roam
ode_modules\npm\bin\npx-cli.js
C:\Users\menier\AppData\Roaming\npm\npm -> C:\Users\menier\AppData\Roam
```

# npm



- Installation de packages node.js
- Problème
  - sudo
  - Installation locale
  - Accès aux exécutables

# npm



- npm config

- List

```
C:\Users\menier>npm config list
; cli configs
metrics-registry = "https://registry.npmjs.org/"
scope = ""
user-agent = "npm/5.6.0 node/v8.9.3 win32 x64"

; builtin config undefined
prefix = "C:\\\\Users\\\\menier\\\\AppData\\\\Roaming\\\\npm"

; node bin location = D:\\Program Files\\nodejs\\node.exe
; cwd = C:\\Users\\menier
; HOME = C:\\Users\\menier
; "npm config ls -l" to show all defaults.

C:\Users\menier>
```



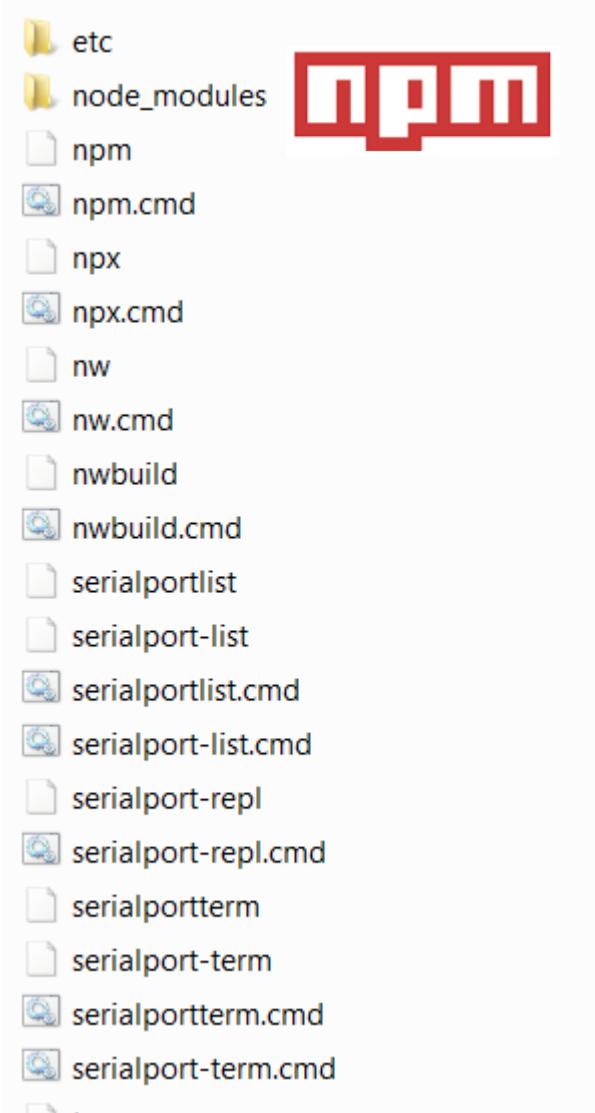
- Prefix est la racine d'installation de node\_module
- En global

# npm



- npm config list
  - node\_modules
  - Scripts !
- npm config get prefix

```
; builtin config undefined
prefix = "C:\\\\Users\\\\menier\\\\AppData\\\\Roaming\\\\npm"
```



# npm



- npm list
  - node\_modules
  - du rep courant !
- npm -g list

```
| | |   | | ``-- readable-stream@2.3.3 deduped
| | |   | +-- readable-stream@2.3.3 deduped
| | |   | +-- timed-out@3.1.3
| | |   | +-- unzip-response@1.0.2
| | |   | ``-- url-parse-lax@1.0.2
| | |   |   ``-- prepend-http@1.0.0
| | |   +-- registry-auth-token@1.0.0
| | |   | +-- rc@1.2.5
| | |   |   +-- deep-extend@0.4.2
| | |   |   +-- ini@1.3.5
| | |   |   +-- minimist@1.2.0
| | |   |   ``-- strip-json-comments@2.0.1
| | |   | ``-- safe-buffer@5.1.1 deduped
| | |   +-- registry-url@3.1.0
| | |   | ``-- rc@1.2.5 deduped
| | |   |   ``-- semver@5.5.0
| | |   +-- lazy-req@1.1.0
| | |   +-- semver-diff@2.1.0
| | |   | ``-- semver@5.5.0
| | |   +-- xdg-basedir@2.0.0
| | |   | ``-- os-homedir@1.0.2
| | |   | ``-- winresourcer@0.9.0
| | +-- serialport@6.0.5
| | +-- bindings@1.3.0
| | +-- commander@2.14.0
| | +-- debug@3.1.0
| | | ``-- ms@2.0.0
| | +-- nan@2.8.0
| | +-- prebuild-install@2.5.0
| | | +-- detect-libc@1.0.3
| | | +-- expand-template@1.1.0
| | | +-- github-from-package@0.0.0
| | | +-- minimist@1.2.0
| | | +-- mkdirp@0.5.1
| | | | ``-- minimist@0.0.8
| | | +-- node-abi@2.2.0
| | | | ``-- semver@5.5.0
| | | +-- noop-logger@0.1.1
| | +-- npmlog@4.1.2
```



# Node.js



- Modifier la configuration prefix

```
[menier@ens-dsiva-0026 ~]$ npm config list
; cli configs
user-agent = "npm/3.10.10 node/v6.12.0 linux x64"

; project config /ubs/fukuisaurus/home.1/m/menier/.npmrc
http-proxy = "http://squidva.univ-ubs.fr:3128/"
https-proxy = "http://squidva.univ-ubs.fr:3128/"
prefix = "/ubs/home/prof/menier"
proxy = "http://squidva.univ-ubs.fr:3128/"
registry = "http://registry.npmjs.org/"
strict-ssl = true
```

- npm config set prefix /ubs/home/prof/menier
- Installation node\_module dans ~/lib
- Exemple : cowsay



# Node.js

- npm i --g cowsay

```
[menier@ens-dsiva-0026 ~]$ npm i --g cowsay
/ubs/home/prof/menier/bin/cowsay -> /ubs/home/
y/cli.js
/ubs/home/prof/menier/bin/cowthink -> /ubs/home/
say/cli.js
/ubs/home/prof/menier/lib
└── cowsay@1.2.1
    ├── get-stdin@5.0.1
    ├── optimist@0.6.1
    │   └── minimist@0.0.10
    ├── wordwrap@0.0.3
    ├── string-width@2.1.1
    └── is-fullwidth-code-point@2.0.0
        └── strip-ansi@4.0.0
            └── ansi-regex@3.0.0
```

- Installation locale sans sudo
- Accès aux exécutables

```
[menier@ens-dsiva-0026 ~]$ cowsay "hello"
```

```
< hello >
-----
 \  ^__^
  (oo)\_____
  (__)\       )\/\
      ||----w |
      ||     ||
```

```
$ lolcatjs --help

Usage: lolcatjs [OPTION]... [FILE]...

Concatenate FILE(s), or standard input, to standard output.
With no FILE, or when FILE is -, read standard input.

--spread, -p <f>: Rainbow spread (default: 8.0)
--freq, -F <f>: Rainbow frequency (default: 0)
--seed, -S <i>: Rainbow seed, 0 = random (default)
--animate, -a: Enable psychedelics
--duration, -d <i>: Animation duration (default: 10)
--speed, -s <f>: Animation speed (default: 20.0)
--force, -f: Force color even when stdout is not a terminal
--version, -v: Print version and exit
--help, -h: Show this message
```

#### Examples:

```
lolcatjs f - g      Output f's contents, then stdin, then a rainbow cookie.
lolcatjs             Copy standard input to standard output, then a rainbow cookie.
fortune | lolcatjs Display a rainbow cookie.
```

```
Report lolcatjs bugs to <https://github.com/robertboloc/lolcatjs>
lolcatjs home page: <https://github.com/robertboloc/lolcatjs>
Report lolcatjs translation bugs to <http://speaklolc.com>
```

```
rboloc at nurbert in /git/lolcatjs on master
$ fortune | cowsay | lolcatjs
```

```
/ This was the most unkindest cut of all. \
|  |
\-- William Shakespeare, "Julius Caesar" /
-----
          \  ^__^
           (oo)\_____
              (__)\       )\/\
                 ||----w |
                  ||     ||
```

```
rboloc at nurbert in /git/lolcatjs on master
$ █
```



# Node.js

```
[menier@ens-dsiva-0026 ~]$ npm -help
```



Usage: npm <command>

where <command> is one of:

access, adduser, bin, bugs, c, cache, completion, config,  
ddp, dedupe, deprecate, dist-tag, docs, edit, explore, get,  
help, help-search, i, init, install, install-test, it, link,  
list, ln, login, logout, ls, outdated, owner, pack, ping,  
prefix, prune, publish, rb, rebuild, repo, restart, root,  
run, run-script, s, se, search, set, shrinkwrap, star,  
stars, start, stop, t, tag, team, test, tst, un, uninstall,  
unpublish, unstar, up, update, v, version, view, whoami

```
npm <cmd> -h      quick help on <cmd>
npm -l            display full usage info
npm help <term>   search for help on <term>
npm help npm      involved overview
```

Specify configs in the ini-formatted file:

  /ubs/home/prof/menier/.npmrc

or on the command line via: npm <command> --key value

Config info can be viewed via: npm help config



# Node.js



- npm en ligne de commande
- ... ou bien via package.json
  - Sorte de *Manifest*
- Dans le cas d'un projet local
  - Librairies spécifiques au projet
  - Options d'exécutions spécifiques (script)
  - Version, nom etc..
  - Dépendances
  - Publier le module vers les modules node.js
  - Etc..



# Node.js



- Package.json

- *npm init*

- Pose des questions à l'utilisateur
  - Construction d'un package.json
  - par défaut
- 
- Minimum :
    - "name" : "monprog"
    - "version" : "1.0.0"

```
{  
  "name": "my_package",  
  "description": "",  
  "version": "1.0.0",  
  "main": "index.js",  
  "scripts": {  
    "test": "echo \\"$Error: no"  
  },  
  "repository": {  
    "type": "git",  
    "url": "https://github.com"  
  },  
  "keywords": [],  
  "author": "",  
  "license": "ISC",  
  "bugs": {  
    "url": "https://github.com"  
  },  
  "homepage": "https://github."  
}
```



# Node.js



- Package.json

- "main" : "prog.js"

- Point d'entrée dans le programme

- "repository" :

- où vit le code ?

```
"repository": {  
    "type": "git",  
    "url": "https://github.com/bob/supertop.git"  
}
```

HTTP : à connaître

GIT : à connaître



# Node.js



- Package.json

- "script" :

- Définitions de scripts de lancement, test etc..

```
"scripts": {  
    "build": "node app.js",  
    "test": "standard"  
}
```

- npm run build
- npm build



# Node.js

- Package.json



- "dependencies" :
  - Librairies à installer en local

```
"dependencies": {  
    "async": "^0.2.10",  
    "npm2es": "~0.4.2",  
    "optimist": "~0.6.0",  
    "request": "~2.30.0",  
    "skateboard": "^1.5.1",  
    "split": "^0.3.0",  
    "weld": "^0.2.2"  
},
```

SEMVER

*Semantic versioning*

MAJOR.MINOR.PATCH



# Node.js

- Package.json



SEMVER

- "dependencies" : *Semantic versioning*  
MAJOR.MINOR.PATCH

- 2.4.\*

- 1.2.3 - 2.3.4 ou  $\geq 1.2.3 \leq 2.3.4$
- $\sim 1.2.3$  ou bien  $\geq 1.2.3 < 1.3.0$
- $\wedge 1.2.3$  ou bien  $\geq 1.2.3 < 2.0.0$
- - prerelease

Sauf 0.x.x



# Node.js

- Package.json



- "engines" :

```
{ "engines" : { "node" : ">=0.10.3 <0.12" } }
```

- Spécifier le nom :

```
{ "engines" : { "npm" : "~1.0.20" } }
```

<https://www.npmjs.com>

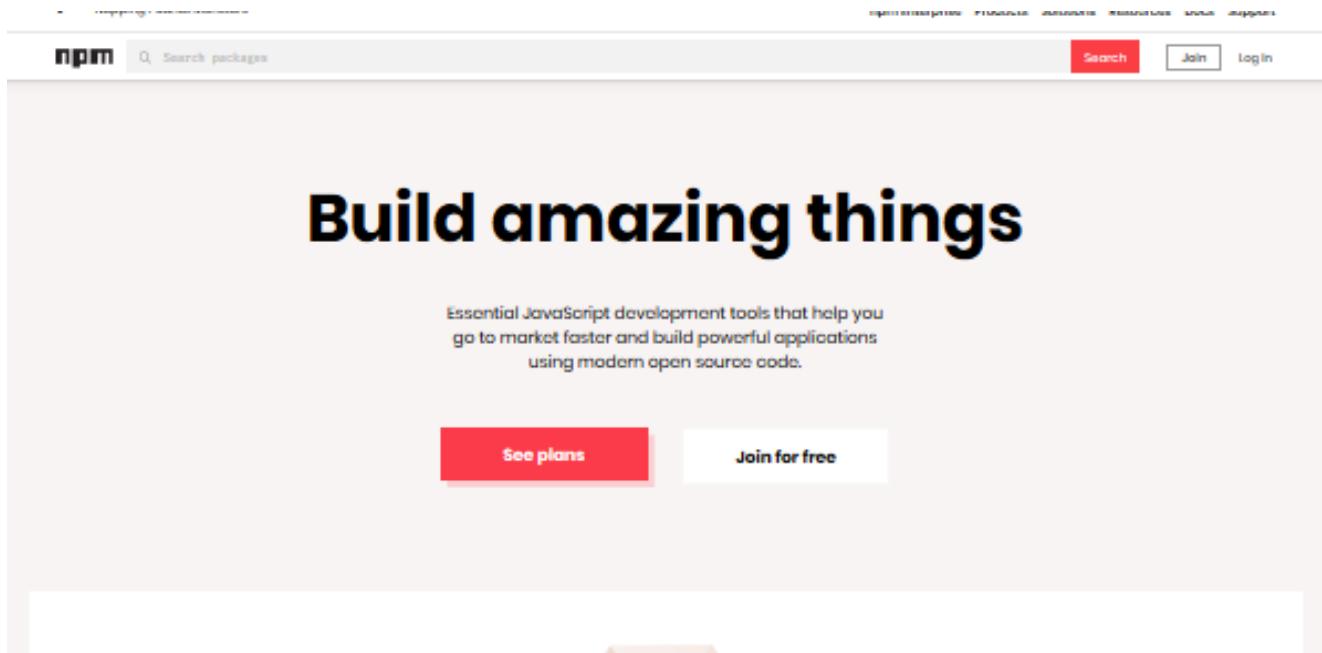


# Node.js

- Package.json



- "os": [ "darwin", "linux", "!win32" ]
- npm peut publier la librairie sur <https://www.npmjs.com>





# Node.js

- **npm install express**

- Micro framework
- Basé sur http
- Beaucoup plus simple d'utilisation
- L'un n'empêche pas l'autre (cf javascript et jquery)
- Syntaxe réactive (attention – cf apply-template de xslt)
- Principe
  - Définition de routes
  - Avec paramètres



# Node.js

- Npm

```
C:\Users\gildas>npm
Usage: npm <command>

where <command> is one of:
  access, add-user, adduser, apihelp, author, bin, bugs, c,
  cache, completion, config, ddp, dedupe, deprecate, dist-tag,
  dist-tags, docs, edit, explore, faq, find, find-dupes, get,
  help, help-search, home, i, info, init, install, issues, la,
  link, list, ll, ln, login, logout, ls, outdated, owner,
  pack, ping, prefix, prune, publish, r, rb, rebuild, remove,
  repo, restart, rm, root, run-script, s, se, search, set,
  show, shrinkwrap, star, stars, start, stop, t, tag, team,
  test, tst, un, uninstall, unlink, unpublish, unstar, up,
  update, upgrade, v, verison, version, view, whoami

npm <cmd> -h      quick help on <cmd>
npm -l            display full usage info
npm faq          commonly asked questions
npm help <term>  search for help on <term>
npm help npm     involved overview

Specify configs in the ini-formatted file:
  C:\Users\gildas\.npmrc
or on the command line via: npm <command> --key value
Config info can be viewed via: npm help config

npm@3.3.6 F:\Program Files\nodejs\node_modules\npm

C:\Users\gildas>_
```

- Installations dans *node\_modules*



# Node.js

- Npm

```
C:\Users\gildas>npm config list
; cli configs
user-agent = "npm/3.3.6 node/v5.0.0 win32 x64"

; builtin config undefined
prefix = "C:\\\\Users\\\\gildas\\\\AppData\\\\Roaming\\\\npm"

; node bin location = F:\\Program Files\\nodejs\\node.exe
; cwd = C:\\Users\\gildas
; HOME = C:\\Users\\gildas
; "npm config ls -l" to show all defaults.
```

- Changer les chemins (cf doc)



## • Npm

```
C:\Users\gildas>npm list
C:\Users\gildas
└─<empty>

C:\Users\gildas>npm list -g
C:\Users\gildas\AppData\Roaming\npm
  babel@5.8.23
    babel-core@5.8.23
      babel-plugin-constant-folding@1.0.1
      babel-plugin-dead-code-elimination@1.0.2
      babel-plugin-eval@1.0.1
      babel-plugin-inline-environment-variables@1.0.1
      babel-plugin-jscript@1.0.4
      babel-plugin-member-expression-literals@1.0.1
      babel-plugin-property-literals@1.0.1
      babel-plugin-proto-to-assign@1.0.4
      babel-plugin-react-constant-elements@1.0.3
      babel-plugin-react-display-name@1.0.3
      babel-plugin-remove-console@1.0.1
      babel-plugin-remove-debugger@1.0.1
      babel-plugin-runtime@1.0.7
      babel-plugin-undeclared-variables-check@1.0.2
        leven@1.0.2
      babel-plugin-undefined-to-void@1.1.6
      babylon@5.8.23
      bluebird@2.9.34
      chalk@1.1.1
        ansi-styles@2.1.0
        escape-string-regexp@1.0.3
        has-ansi@2.0.0
          ansi-regex@2.0.0
        strip-ansi@3.0.0
          ansi-regex@2.0.0
        supports-color@2.0.0
      core-js@1.1.4
      debug@2.2.0
        ms@0.7.1
      detect-indent@3.0.1
        get-stdin@4.0.1
        minimist@1.2.0
      esutils@2.0.2
      globals@6.4.1
      home-or-tmp@1.0.0
        os-tmpdir@1.0.1
        user-home@1.1.1
      is-integer@1.0.6
        is-finite@1.0.1
          number-is-nan@1.0.0
      js-tokens@1.0.1
      json5@0.4.0
```



# Node.js

- Npm
  - Installation locale : npm install express
  - Installation globale : --globale
  - npm uninstall express
  - npm update express
- Possibilité de publier une librairie
  - Fichier JSON : package.json
  - Commande npm
  - Mise à disposition de la communauté



express

Search

Join

Log In

16687 packages found

1 2 3 ... 835 »

Sort Packages

Optimal

Popularity

Quality

Maintenance

Who's Hiring?

Apply Digital, Overpass, Hired and lots of other companies are hiring javascript developers.

See all 19 companies

express exact match

Fast, unopinionated, minimalist web framework

express framework sinatra web rest restful router app api

dougwilson published 4.16.4 • 5 months ago

path-to-regexp

Express style path to RegExp utility

express regexp route routing

blakeembrey published 3.0.0 • 2 months ago

cors

Node.js CORS middleware

cors express connect

dougwilson published

morgan

HTTP request logger middleware

express http logger

dougwilson published

apollo-server-express

## Node.js's npm Is Now The Largest Package Registry in the World



133

Posted by EditorDavid on Saturday January 14, 2017 @11:34AM from the

Linux.com highlights [some interesting statistics about npm](#), the package manager for Node.js.

- "At over 350,000 packages, the npm registry contains more than double the next most populated package registry (which is the Apache Maven repository). In fact, it is currently the largest package registry in the world."



# Node.js

npm is the package manager for javascript.



205 780  
total packages



44 294 761  
downloads in the last day



661 087 970  
downloads in the last week



2 474 649 747  
downloads in the last month

## packages people 'npm install' a lot



### browserify

browser-side require() the node way  
10.2.6 published 4 months ago by substack



### grunt-cli

The grunt command line interface.  
0.1.13 published 2 years ago by tkellen



### bower

The browser package manager  
1.4.1 published 8 months ago by sheerun



### gulp

The streaming build system  
3.9.0 published 6 months ago by phated



### grunt

The JavaScript Task Runner

### express

Fast, unopinionated, minimalist web framework  
4.13.1 published 4 months ago by dougwilson



### npm

a package manager for JavaScript  
2.13.0 published 4 months ago by zkat



### cordova

Cordova command line interface tool  
5.1.1 published 5 months ago by stevegill



### forever

A simple CLI tool for ensuring that a given node...  
0.14.2 published 5 months ago by indexzero



### less

Leaner CSS



### pm2

Production process manager for Node.JS appli...  
0.14.3 published 5 months ago by jshkurti



### karma

Spectacular Test Runner for JavaScript.  
0.13.1 published 4 months ago by dignifiedquire



### coffee-script

Unfancy JavaScript  
1.9.3 published 6 months ago by jashkenas



### statsd

A simple, lightweight network daemon to colle...  
0.7.2 published a year ago by pkhzzrd



### yo

CLI tool for running Yeoman generators



Gestion de paquets  
Déploiements  
Gestion de dépendance / yarn (à voir)  
Mise à jour etc



*Process Manager*  
Lancement  
Gestion des erreurs  
Métrie etc..



# Node.js

Quick Start   Documentation   Tutorials   Modules   [Star](#)   [Monitor PM2](#)

# PM2

Advanced, production process manager for Node.js

```
npm install pm2 -g
```

---

## Features

A Complete feature set for production environment, built with a worldwide community of developers and enterprises

```
pm2 start app.js
```

- ✓ Behavior configuration
- ✓ Source map support
- ✓ Container Integration
- ✓ Watch & Reload
- ✓ Log management
- ✓ Monitoring
- ✓ Cluster Mode
- ✓ Hot reload
- ✓ Development workflow
- ✓ Startup Scripts
- ✓ Deployment workflow
- ✓ PaaS Compatible



```
# Fork mode
pm2 start app.js --name my-api # Name process

# Cluster mode
pm2 start app.js -i 0          # Will start maximum processes with LB depending on avai.
pm2 start app.js -i max        # Same as above, but deprecated.

# Listing
pm2 list                      # Display all processes status
pm2 jlist                      # Print process list in raw JSON
pm2 prettylist                  # Print process list in beautified JSON

pm2 describe 0                 # Display all informations about a specific process

pm2 monit                      # Monitor all processes

# Logs
pm2 logs [--raw]               # Display all processes logs in streaming
pm2 flush                      # Empty all log files
pm2 reloadLogs                 # Reload all logs
```



```
# Actions

pm2 stop all          # Stop all processes
pm2 restart all        # Restart all processes

pm2 reload all         # Will 0s downtime reload (for NETWORKED apps)

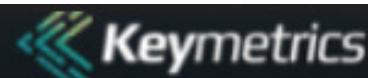
pm2 stop 0             # Stop specific process id
pm2 restart 0          # Restart specific process id

pm2 delete 0           # Will remove process from pm2 list
pm2 delete all          # Will remove all processes from pm2 list

# Misc

pm2 reset <process>    # Reset meta data (restarted time...)
pm2 updatePM2          # Update in memory pm2
pm2 ping                # Ensure pm2 daemon has been launched
pm2 sendSignal SIGUSR2 my-app # Send system signal to script
pm2 start app.js --no-daemon
pm2 start app.js --no-vizion
pm2 start app.js --no-autorestart
```

pm2 startup



GeoFleet

Filter by Server name

Filter by App name



Server #1 - LWS (IP 31.207.33.100 - Hostname vps40349)



Last beat



7 minutes ago

System

Linux

Hostname

vps40349

CPU load average

5.11



app.geofleet.ma



online

online for 2 days

More

Alerts

Logs

CPU  
2 %MEM  
77.3 ...Errors  
0 ⚠Restarts  
21 ⚡HTTP avg.  
22.46...  
Processes  
1 ⚙

App 1 - 77.3 MB

Loop delay  
2.45msHTTP  
95.73req/...

Add a metric



Commands

Add an action

preprod.geofle...



online

online for 2 days

More

Alerts

Logs

CPU  
0 %MEM  
34.8 ...Errors  
0 ⚠Restarts  
1 ⚡HTTP avg.  
0ms  
Processes  
1 ⚙

pm2-server-monit

Support



Issues

CPU usage

Free mem...

Avail. Disk

Total Proc...  
TTY/SSH ...

ne

0 ⚠

14 29%

92 2%

N/A

39  
0

0

# Express

4.16.4

Fast, unopinionated, minimalist  
web framework for [Node.js](#)

```
$ npm install express --save
```

## Web Applications

Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.

## APIs

With a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy.

## Performance

Express provides a thin layer of fundamental web application features, without obscuring Node.js features that you know and love.

# Node.js : express

## •Création du serveur

```
var express = require('express')
var https = require('https')
var http = require('http')
var fs = require('fs')
var app = express()

var options = {
  host : '127.0.0.1',
  key : fs.readFileSync('ssl/server.key'),
  cert : fs.readFileSync('ssl/server.crt')
}

http.createServer(app).listen(80)

https.createServer(options, app).listen(443)

app.get('/', function(req, res) {
  res.send('express est lancé')
})
```

Bloquant  
(volontairement)

# Node.js

- Définition des routes
  - `app.<methode>(chemin, callback)`

```
[-] app.get('/', function(req, res) {
      res.send('express est lancé')
    })

[-] app.get('/login', function(req, res) {
      res.send('page de login')
    })

[-] app.post('/panier', function(req, res) {
      res.send('paiement')
    })

[-] app.get('/logout', function(req, res) {
      res.send('au revoir')
    })
```



# Node.js

```
└─ app.all('/*', function(req, res) {  
    })  
  
└─ app.all('/utilisateur/*', function (req, res) {  
    })
```

- get et post
- \*



# Node.js

- Paramètres de route

- Impossible de définir toutes les routes si nom variable
- `/utilisateur/menier`
- `/utilisateur/courtrai`
- Base de données utilisateurs
- Route paramétrées par `:param`

```
app.get('/utilisateur/:identifiant', function(req, res) {  
    res.send("utilisateur d'identifiant : "+ req.param("identifiant"))  
})
```

- Utilisateur d'identifiant : `menier`

- Texte à ‘trous’ et unification
- *pattern matching*
- (Scala+ match, Prolog+varlibres)



# Node.js

```
app.get('/utilisateur/:identifiant', function(req, res) {
  res.send("utilisateur d'identifiant : " + req.param("identifiant"))
})
req.params.identifiant
(req.param() deprecated)

app.param(':identifiant', function(req, res, next, value) {
  console.log('je viens de voir passer '+ value)
  next();
})
})
```

- Notez *next*
- Si Express trouve un paramètre dans un chemin et qu'un *callback* est défini,
  - alors *param* est exécuté avant le *get*
- *req.param* permet de récupérer la valeur du param
  - Ou bien *value* dans *app.param*



# Node.js

- **Response**

- Construction d'une réponse HTTP complète

- Headers

- res.set('Content-Type', 'text/plain')

- Status

- res.status(200) // pas de pb
    - res.status(300) // redirection
    - res.status(400) // pb de requête
    - res.status(401) // non autorisé
    - res.status(403) // interdit
    - res.status(404) // non trouvé
    - res.status(500) // erreur serveur



# Node.js

## • HTTP exemple

HTTP protocole de base du Web  
Voir partie Php Web  
HTTP à connaître

Client : socket sur le port 80 URL <http://www.linux.org/>

**GET /index.html HTTP/1.1**

**Connection: Keep-Alive**

**User-Agent: Mozilla/4.7 [en] (WinNT; I)**

**Host: [www.linux.org](http://www.linux.org)**

**Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, \*/\***

**Accept-Encoding: gzip**

**Accept-Language: en**

**Accept-Charset: iso-8859-1,\* ,utf-8**

**Ligne vide**



# Node.js

```
var express = require('express')
var url = require('url')
var app = express()
app.listen(80)
app.get('/', function(req, res) {
  var reponse = '<html><head>test</head></head>' +
    '<body><h1>Hello</h1></body></html>'
  res.status(200)
  res.set({
    'Content-Type' : 'text/html',
    'Content-Length' : reponse.length
  })
  res.send(reponse)
})
app.get('/error', function(req, res) {
  res.status(400)
  res.send("mauvaise requete")
})
```



# Node.js

- **JSON**

- Serveur expédie un objet sérialisé par JSON

- Le client récupère les données puis peuple la page HTML avec le contenu

- jquery

- Coté serveur

- `res.json(500, {status:false, message:'pb serveur'})`
    - `res.json( { nom:'paul', prenom:'henry'} )`

- Coté client

- `JSON.parse(resultat)`



# Node.js

- Envoyer un fichier complet au client
  - Image par exemple

```
var express = require('express')
var url = require('url')
var app = express()
app.listen(80)
app.get('/image', function(req, res) {
    res.sendfile('monimage.jpg',
        { root : './img/' },
        function(err) {
            // à faire
        }
    )
})
```

◦ res.download



# Node.js

```
var express = require('express')
var url = require('url')
var app = express()
app.listen(80)
app.get('/google', function(req, res) {
  res.redirect('http://google.com')
})
```

- Cookies, etc..



# Node.js

- Templates
- Jade - pug / EJS (Embedded Javascript)
- Pages prédefinies
  - P -> render -> HTML
- Philosophies différentes
  - Jade : codage
  - EJS : code HTML
- Installation
  - Jade -> npm install jade
  - EJS -> npm install ejs



# Node.js

- Enregistrer le moteur de *template* pour Express

```
var app = express()
app.set('view', './mesVues')
app.set('view engine', 'jade')

app.engine('jade', require('jade').__express)
```

Répertoire *templates*

extension



# Node.js

- Template Jade
  - Codage HTML (plus compacte)

main\_jade.jade

```
doctype html
html(lang="fr")
  head
    title="exemple de template jade"
  body
    block content
```

test\_jade.jade

```
extends main_jade
block content
  h1 test utilisation jade
  ul
    li Nom : #{nom}
    li Prenom : #{prenom}
```

- res.render('test\_jade')
  - Lit le template **mesVues/test\_jade** et génère en sortie le code HTML

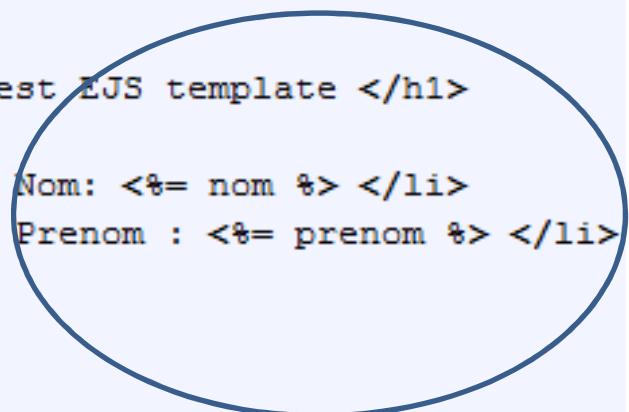


# Node.js

- Template EJS

- HTML      t\_test.html

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title> test ejs </title>
</head>
<body>
<h1> test EJS template </h1>
<ul>
<li> Nom: <%= nom %> </li>
<li> Prenom : <%= prenom %> </li>
</ul>
</body>
</html>
```





# Node.js

## test EJS template

- Variables

- Jade : # { nom }
- EJS : <%= nom %>
- Nom: Dupond
- Prenom : joe

- app.locals

```
app.locals({  
    nom : 'dupond',  
    prenom : 'joe'  
})
```



# Node.js

```
var express = require('express')
var url = require('url')
var app = express()

app.set('view', './mesVues')
app.set('view engine', 'jade')
app.engine('jade', require('jade').__express)
app.engine('html', require('ejs').renderFile)

app.listen(80)

app.locals({
    nom : 'dupond',
    prenom : 'joe'
})

app.get('/jade', function(req, res) {
    res.render('test_jade.jade')
})

app.get('/ejs', function(req, res) {
    app.render('t_test.html', function(err, rendered) {
        res.send(rendered)
    })
})
```

*res.render*

*app.render*



# Node.js

- **Middleware Express**

- cf param
- Méthodes et classes qui s'intègrent dans le mécanisme Express

- **Exemples**

- static
  - GET : express.static
- express-logger
  - Log des requêtes au serveur
- basic-auth-connect
  - Authentification de base pour HTTP
- cookie-parser
  - Lecteur / écriture de cookies
- cookie-session
  - Sessions contrôlées par cookies
- express-session
- body-parser
  - Analyse de JSON via POST
- compression
  - Gzip

# Node.js

- Exemples :

- Serveur statique (délivre des pages directement)

```

var express = require("express"),
app = express();

hostname = "192.168.1.89" || 'localhost',
port = 80,
publicDir = __dirname + '/public'; // rep contenant les fichiers

app.get("/", function (req, res) { // index par défaut
  res.redirect("/index.html");
});

app.use(express.static(publicDir)); // repertoire d'accès

// Si req non couverte par les chemins
app.use(function(req, res) {
  res.status(400);
  res.render('404.jade', {title: '404: File Not Found'});
});

console.log("Mini Serveur %s ouvert en http://%s:%s", publicDir, hostname, port);

app.listen(port, hostname);

```

Oops!  
you found a  
**Dead Link**



block content  
img(src='404.jpg')

- Noter MW pour la page 404



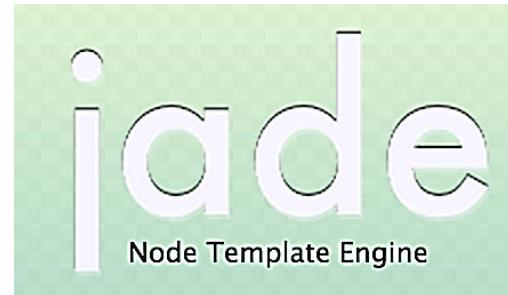
# Node.js

- Node.js = serveur ?
  - Node.js n'est pas limité au côté serveur
  - Package 'request' (pas exemple)
  - Client
    - CF TD Node.js Client
    - Analyse de page
    - Téléchargement
    - Pas de limitation / Ajax
- Besoins machine
  - Très léger
  - Raspberry PI
- MAIS : **ASYNCHRONE !!!!!!!**



# Moteurs de templates

- Jade
- Pug

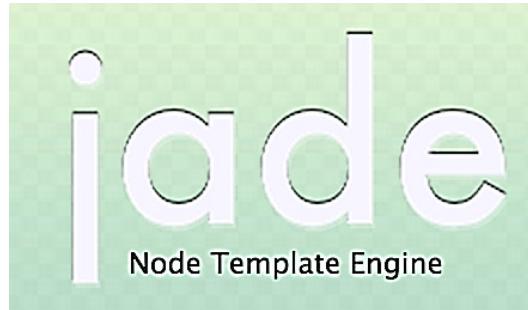


- **Principes de base** (Sanjay Guruprasad )

```
div  
  p Bonjour  
  p tout le monde !
```

```
<div>  
  <p>Bonjour</p>  
  <p>tout le monde !</p>  
</div>
```

attention aux indentations !!!



## •Attributes

```
div(class="movie-card", id="oceans-11")
  h1(class="movie-title") Ocean's 11
  img(src="/img/oceans-11.png", class="movie-poster")
  ul(class="genre-list")
    li Comedy
    li Thriller
```

```
<div class="movie-card" id="oceans-11">
  <h1 class="movie-title">Ocean's 11</h1>
  
  <ul class="genre-list">
    <li>Comedy</li>
    <li>Thriller</li>
  </ul>
</div>
```



- Attributes / pseudo CSS

```
div.movie-card#oceans-11
  h1.movie-title Ocean's 11
  img.movie-poster(src="/img/oceans-11.png")
  ul.genre-list
    li Comedy
    li Thriller
```

```
<div class="movie-card" id="oceans-11">
  <h1 class="movie-title">Ocean's 11</h1>
  
  <ul class="genre-list">
    <li>Comedy</li>
    <li>Thriller</li>
  </ul>
</div>
```



- tag + text
- comment faire pour plusieurs lignes de texte ?

div

p How are you?

p.

I'm fine thank you.

And you? I heard you fell into a lake?

That's rather unfortunate. I hate it when my shoes get wet.

notez le point .

tout ce qui suit devient du texte  
(plus de nouveau tag HTML retours ligne)



- Utiliser Javascript pour produire du code

```
- var x = 5;  
div  
ul  
-for (var i=1; i<=x; i++) {  
    li Hello  
-}
```

```
<div>
  <ul>
    <li>Hello</li>
    <li>Hello</li>
    <li>Hello</li>
    <li>Hello</li>
    <li>Hello</li>
  </ul>
</div>
```



- Utiliser Javascript pour produire du code
- tag=
  - indique qu'on souhaite calculer un contenu

```
- var x = 5;
div
  ul
    - for (var i=1; i<=x; i++) {
      li= i + ". Hello"
    - }
```

```
<div>
  <ul>
    <li>1. Hello</li>
    <li>2. Hello</li>
    <li>3. Hello</li>
    <li>4. Hello</li>
    <li>5. Hello</li>
  </ul>
</div>
```



## •Boucles

```
- var droids = ["R2D2", "C3PO", "BB8"];
div
  h1 Les Droïds de Star Wars
  for name in droids
    div.card
      h2= name
```

```
<div>
  <h1>Les Droïds de Star Wars</h1>
  <div class="card">
    <h2>R2D2</h2>
  </div>
  <div class="card">
    <h2>C3PO</h2>
  </div>
  <div class="card">
    <h2>BB8</h2>
  </div>
</div>
```



- Variables :#{ variable }

```
- var profileName = "Danny Ocean";
div
  p Hi there, #{profileName}. How are you doing?
```



- Mixin (~ macro)

appel :

```
+thumbnail("oceans-eleven", "Danny Ocean makes an  
elevator pitch.")  
+thumbnail("pirates", "Introducing Captain Jack Sparrow!")
```

```
mixin thumbnail( imageName, caption)
```

```
div.thumbnail
```

```
img(src="/img/#{imageName}.jpg")
```

```
h4.image-caption= caption
```

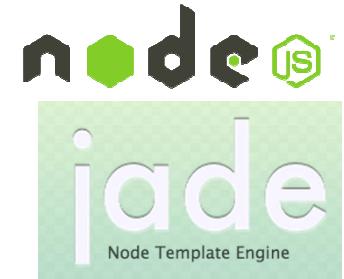
```
<div class="thumbnail">  
    
  <h4 class="image-caption">  
    Danny Ocean makes an elevator pitch.  
  </h4>  
</div>  
<div class="thumbnail">  
    
  <h4 class="image-caption">  
    Introducing Captain Jack Sparrow!  
  </h4>  
</div>
```

## • Exemple complet

```
- var movieList = [
{
  title: "Ocean's Eleven",
  cast: ["Julia Roberts", "George Clooney", "Brad Pitt", "Andy Garcia"],
  genres: ["Comedy", "Thriller"],
  posterImage: "/img/oceans-eleven",
  imdbURL: "http://www.imdb.com/title/tt0240772/",
  rating: 7
}
// etc...
];
```

attention ne pas confondre  
movie.cast (objet)  
div.rating (class = )

```
mixin movie-card(movie)
div.movie-card
  h2.movie-title= movie.title
  img.movie-poster(src=movie.posterImage)
  h3 Cast
  ul.cast
    each actor in movie.cast
      li= actor
  div.rating
    if movie.rating > 5
      img(src="img/thumbs-up")
    else
      img(src="img/thumbs-down")
  ul.genre
    each genre in movie.genres
      li= genre
```





```
- var movieList = [
  {
    title: "Ocean's Eleven",
    cast: ["Julia Roberts", "George Clooney", "Brad Pitt", "Andy Garcia"],
    genres: ["Comedy", "Thriller"],
    posterImage: "/img/oceans-eleven",
    imdbURL: "http://www.imdb.com/title/tt0240772/",
    rating: 9.2
  },
  ...
];
```

```
mixin movie-card(movie)
  div.movie-card
    h2.movie-title= movie.title
    img.movie-poster(src=movie.posterImage)
    h3 Cast
    ul.cast
      each actor in movie.cast
        li= actor
    div.rating
      if movie.rating > 5
        img(src="img/thumbs-up")
      else
        img(src="img/thumbs-down")
    ul.genre
      each genre in movie.genres
        li= genre
```

```
for movie in movieList
  +movie-card(movie)
```

- Appel et génération :



```
<div class="movie-card">
  <h2 class="movie-title">Ocean's Eleven</h2>
  
  <h3>Cast</h3>
  <ul class="cast">
    <li>Julia Roberts</li>
    <li>George Clooney</li>
    <li>Brad Pitt</li>
    <li>Andy Garcia</li>
  </ul>
  <div class="rating">
    
  </div>
  <ul class="genre">
    <li>Comedy</li>
    <li>Thriller</li>
  </ul>
</div>
```

```
<div class="movie-card">
  <h2 class="movie-title">Pirates of the Caribbean</h2>
  
  <h3>Cast</h3>
  <ul class="cast">
    <li>Johnny Depp</li>
    <li>Keira Knightley</li>
    <li>Orlando Bloom</li>
  </ul>
  <div class="rating">
    
  </div>
  <ul class="genre">
    <li>Adventure</li>
    <li>Comedy</li>
  </ul>
</div>
```



WHO WE ARE

SOLUTIONS FOR... ▾

CAPABILITIES

...



Our company is built on innovation

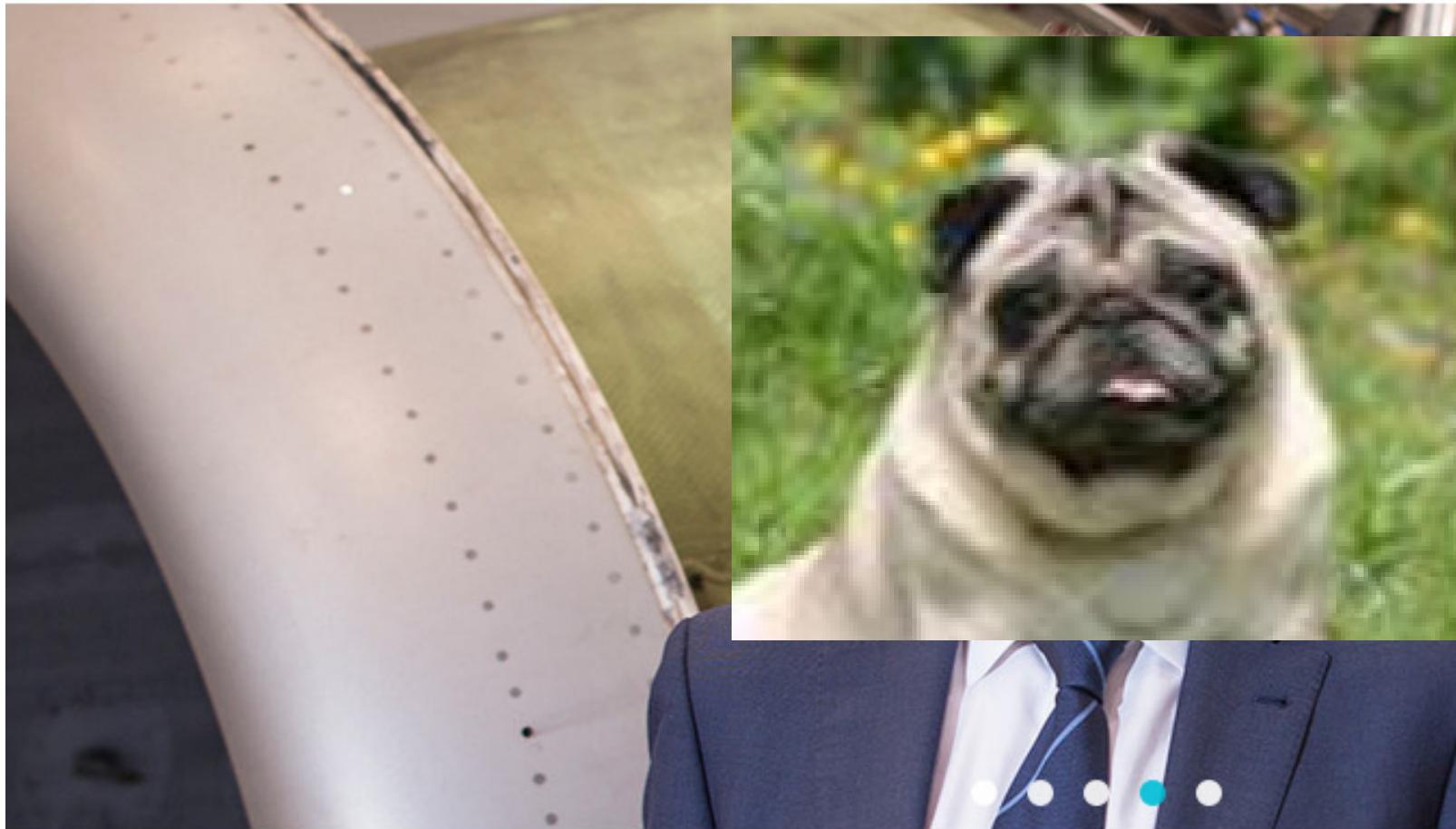


WHO WE ARE

SOLUTIONS FOR... ▾

CAPABILITIES

...



• • • •

Our company is built on innovation.



pug



pug

- Jade renommée en Pug

<https://pugjs.org/api/getting-started.html>

```
const pug = require('pug');

// Compile the source code
const compiledFunction =
pug.compileFile('template.pug');

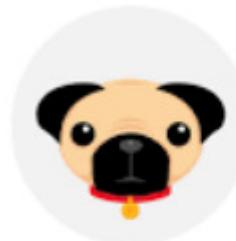
// Render a set of data
console.log(compiledFunction({
  name: 'Timothy'
}));
// "<p>Timothy's Pug source code!</p>

// Render another set of data
console.log(compiledFunction({
  name: 'Forbes'
}));
```

```
//- template.pug
p #{name}'s Pug source code!
```

```
const pug = require('pug');

// Compile + render
console.log(pug.renderFile('template.pug', {
  name: 'Timothy'
}));
// "<p>Timothy's Pug source code!</p>"
```



pug

- Pug2

## Legacy Mixin Call

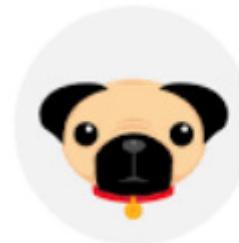
```
//- old  
  
mixin foo('whatever')
```

```
//- new  
  
+foo('whatever')
```

## Attribute Interpolation

```
//- old  
  
a(href='#{link}')  
  
a(href='before#{link}after')
```

```
//- new  
  
a(href=link)  
  
// - (on Node.js/io.js ≥ 1.0.0)  
a(href=`before${link}after`)  
// - (everywhere)  
a(href='before' + link + 'after')
```



pug

- Août 2016 Pug2

## Prefixed `each` Syntax

```
//- old  
  
- each a in b  
  = a  
  
- for a in b  
  = a
```

```
//- new  
  
each a in b  
  = a  
  
for a in b  
  = a
```

attention à : doctype, nodes, selfClosing, utils, compiler, lexer, parser



- Autre parseur : <https://mustache.github.io/>

```
var person = {  
    firstName: "Christophe",  
    lastName: "Coenraets",  
    blogURL: "http://coenraets.org"  
};  
var template = "<h1>{{firstName}} {{lastName}}</h1>Blog: {{blogURL}}";  
var html = Mustache.to_html(template, person);  
$('#sampleArea').html(html);
```

Not Only  
**NOSQL**



# Introduction

- Système d'information
  - *Consistency* (cohérence)
  - *Availability* (disponibilité)
  - *Partition tolerance* (tolérance au partitionnement)

# Introduction

- **Consistency (cohérence)**
  - Tous les nœuds du système ont accès aux mêmes informations et chaque lecture prend en compte chaque écriture antérieure
  - Pas d'incohérences locales sur le contenu de la base

# Introduction

- **Availability** (disponibilité)
  - Les lectures et écritures réussissent toujours
  - Aucune erreur : réponse à toutes les requêtes

# Introduction

- *Partition tolerance* (tolérance au partitionnement)
  - Aucune panne locale ne doit empêcher le système de fonctionner correctement
  - Aucune erreur : réponse à toutes les requêtes

# Introduction

- Montée en charge BD
- Distribution
- Théorème de Brewer (ou de CAP)
  - Impossible d'avoir les 3 en même temps dans un système distribué
  - CA ou AP ou CP mais pas CAP

# Introduction

- Base de données
  - Opération sur les données
  - Transaction
- Fiabilité de ces transactions
  - Modèle **ACID** : une transaction doit avoir les propriétés
    - Atomicité
    - Cohérence
    - Isolation
    - Durabilité

# Introduction

- **Atomicité**

- Tout ou rien
- Une transaction se fait complètement
- Ou bien est annulée (complètement)
- Interdiction absolue de faire un morceau de transaction
- Si un morceau de transaction se fait et est interrompue, il est nécessaire de remettre la BD dans l'état avant la transaction !

# Introduction

- **Cohérence**

- Chaque transaction modifie l'état général de la BD qui doit rester cohérente
- Une transaction se termine quand toutes les parties de la BD ont les mêmes informations conséquences de la transaction

# Introduction

- **I**solation

- Chaque transaction est **indépendante** des autres transactions
- Même résultat si deux transactions se déroulent en // ou en séquence

# Introduction

- Durabilité
  - Quand une transaction est terminée (réalisée ou confirmée), ses effets perdurent même en cas de panne

# Introduction

## • Montée en charge

- Passage à l'échelle (*scalability*)
- Capacité d'extension du système
- Verticale
  - Augmenter la puissance CPU
  - La mémoire, l'espace de stockage

### • Horizontale

- Système distribué : ajouter des nœuds

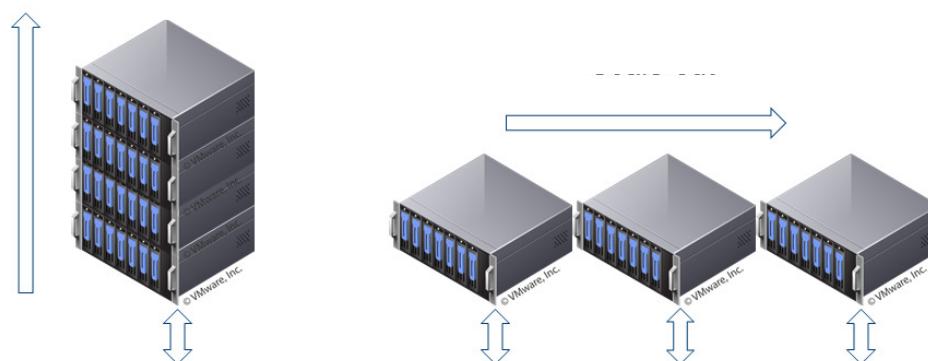
vertical



horizontal



VS



# Introduction

- Système distribué
  - Avantages
    - Tolérances aux fautes /pannes
      - Redondance
    - Passage à l'échelle
      - Rajouter un nœud
    - Partage de certaines ressources
    - Flexibilité
      - Installation simplifiée
    - Vitesse, performance et coûts

# Introduction

- Système distribué
  - Inconvénients
  - Diagnostique et correction des erreurs
  - Réseau
    - Communication
    - Gestion des mises à jour
    - Cohérence
  - Sécurité

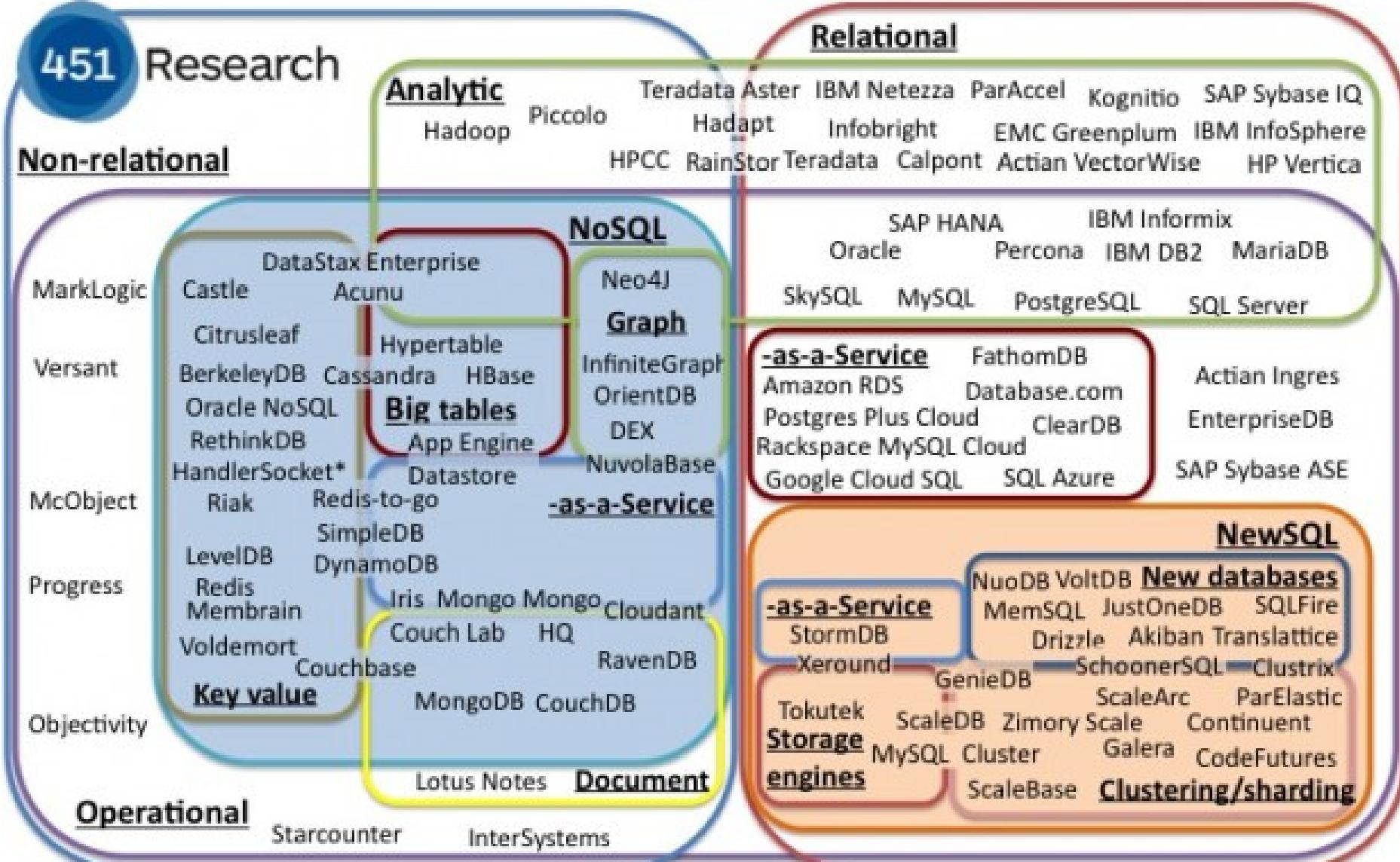
# Introduction

- NoSql : Not (**Only**) Relationnal
- DBMS :
  - Relational Database Management System
  - NoSql = NDBMS
- Pas d'accès via SQL
- Simplification
- MAIS ...
  - Pas de modèle de transaction

# The evolving database landscape

## 451 Research

### Non-relational



# Introduction

- NoSQL : plusieurs possibilités

- Clé -> valeur
    - put, get, delete
    - [Redis, riak et voldemort \(linkedin\)](#)

- Orienté colonne

- Cf table mais avec nombre de colonnes dynamique
    - Hbase (= BigTable de Google)
    - [Cassandra \(Apache\)](#)

- Orienté graphe

- Théorie des graphes
    - Nœuds + relations + propriétés
    - [Neo4j](#)

- Orienté document

# Introduction

- NoSQL : orienté document

- Clé -> document
  - clé
  - Document = XML ou JSON (ou BSON)
  - CouchDB (Apache)
  - RavenDB (Microsoft)
  - MongoDB

- Attention :

- Ne remplace pas SQL
- Dépend du problème à résoudre
- Contraintes
- Jointures
- Transactions

# MongoDB

- Principe et documents
- Opérations Shell
  - Insert
  - Find
  - Update
  - Remove
  - Utilitaires
  - Indexes
- Node.js
  - Driver de base
  - Mongoose
- Administration
  - Comptes
  - Sécurité
- Sharding
- Java

# MongoDB

- Humongous (énorme)
- Depuis 2007
- Liens avec les langages :

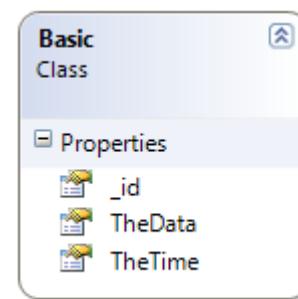
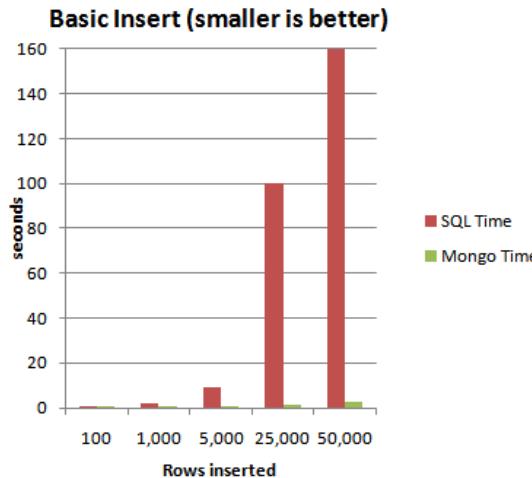
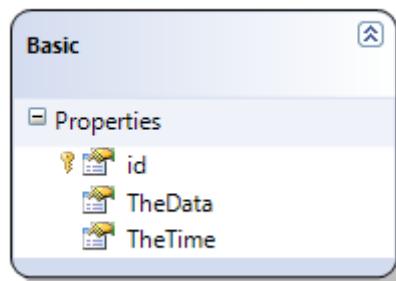
- Javascript
- Java
- Scala
- C, C++
- Erlang,
- PHP,
- Perl
- Python
- Ruby
- Go
- .Net
- Etc..

- Sharding
- Tolérance aux pannes
- 2.6.x
- 3.x

- Rapide ... très rapide !

# MongoDB

- Concurrent SQL ?



```
MongoDB Client
Warming up ...
Building insert data...
Waiting on mutex
Running!
Finished with 10000 MongoDB inserts in 2.032 sec.
Done
```

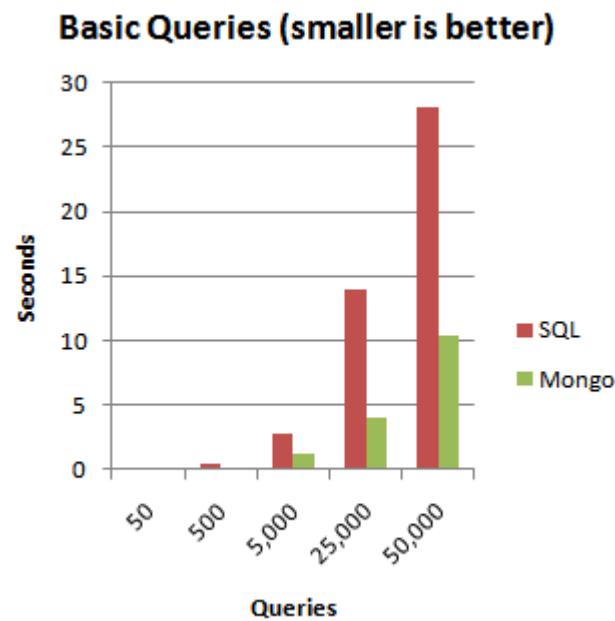
```
SQL Server Client
Warming up ...
Building insert data...
Waiting on mutex
Running!
Finished with 10000 SQL inserts in 204.215 sec.
Done
```

... mais requêtes simples en accès

... mais requêtes complexes  
(jointures etc) ???

# MongoDB

- Concurrent SQL ?

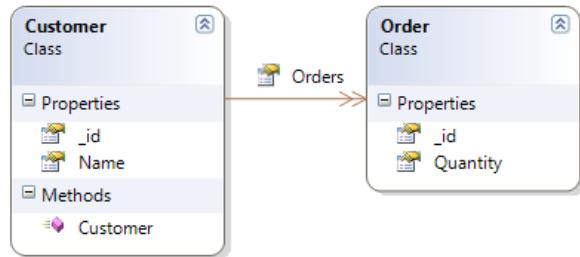


Requêtes (simples) sur le même objet  
... sans jointures

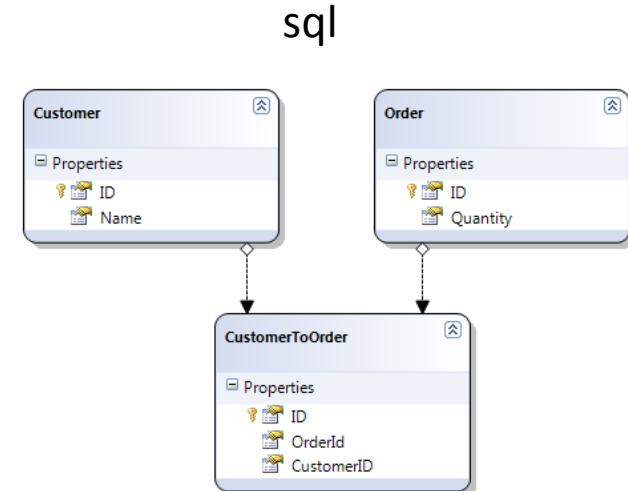
# MongoDB

- Concurrent SQL ?

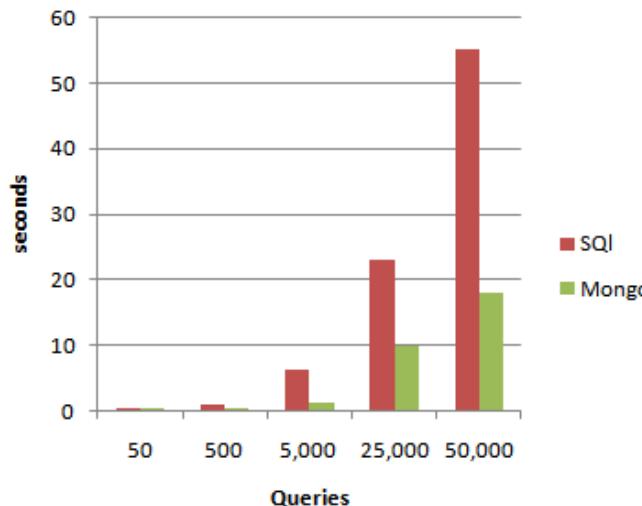
Requête avec une jointure



mongodb



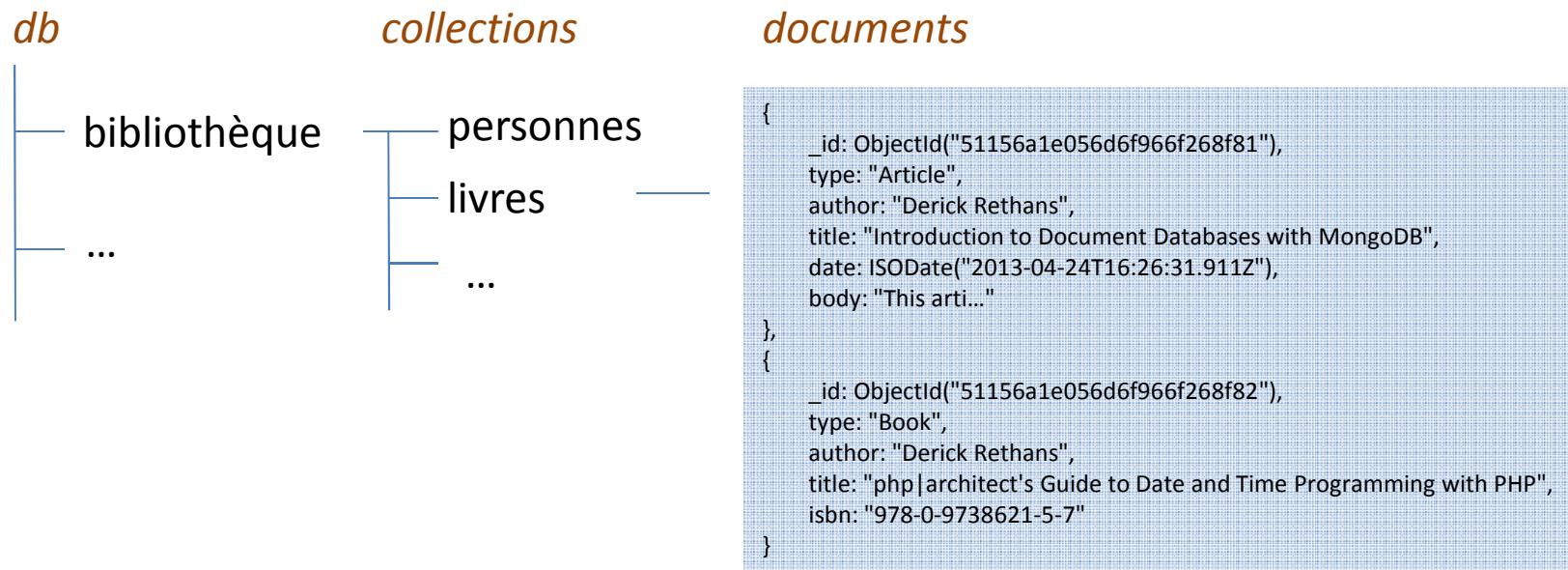
Complex Queries (smaller is better)



MAIS : pas de transaction  
(exemple)

# MongoDB

- Organisation



# MongoDB

- Installation : <https://www.mongodb.com/>

**MongoDB  
multi-document  
ACID transactions  
are coming**

[Read the blog](#)

[Sign up for the beta](#)



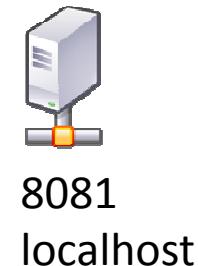
# MongoDB

- CRUD
  - Creation
  - Retrieval
  - Update
  - Deletion
- REST (**R**Epresentational **S**tate **T**ransfert)
  - PUT
  - GET
  - POST
  - DELETE

# MongoDB

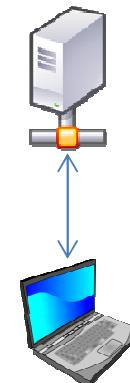
- Lancement simple

- Un seul nœud
- **mongod - - port 8081**
- Serveur local sur le port 8081
- DB dans data/db
  - Utiliser - - dbpath
- Voir - - rest
- Voir **mongod - -help**
- En particulier voir la sortie log



- Client simple

- **mongo - -port 8081**
- **mongo - -host 192.168.1.67 - -port 8081**
- Interface console pour tester et administrer
- Console Javascript (simplifiée)



# MongoDB

- Création d'une base vide
  - mongo -- port 8081 -- host hostname
  - Contient le nécessaire pour la création et l'administration de la base
  - Une seule base par défaut

```
odroid@white:~/mongodb/data$ ls
local.0  local.ns  mongod.lock
```

```
> show dbs
admin  <empty>
local  0.031GB
```

```
> help
    db.help()
    db.mycoll.help()
    sh.help()
    rs.help()
    help admin
    help connect
    help keys
    help misc
    help mr

    show dbs
    show collections
    show users
    show profile
    h time >= 1ms
    show logs
    show log [name]
    'global' is default
    use <db_name>
    db.foo.find()
    db.foo.find( { a : 1 } )
    it
    further iterate
    DBQuery.shellBatchSize = x
    hell
    exit

                                help on db methods
                                help on collection method
                                sharding helpers
                                replica set helpers
                                administrative help
                                connecting to a db help
                                key shortcuts
                                misc things to know
                                mapreduce

                                show database names
                                show collections in current
                                show users in current dat
                                show most recent system.p
                                show the accessible logge
                                prints out the last segme
                                set current database
                                list objects in collectio
                                list objects in foo where
                                result of the last line e
                                set default number of ite
                                quit the mongo shell
```

# MongoDB

- Création d'une base
- Machine à état
  - use **maBase**
  - Par défaut : **maBase**
- Création d'une collection

```
> use biblio
switched to db biblio
> show collections
>
```

```
> db.personnes.insert(
... {nom:"menier",
... prenom:"gildas",
... metier:"maitre de conferences",
... ville:"vannes"}
...
)
Cannot use commands write mode, degrading to compatibility mode
WriteResult({ "nInserted" : 1 })
>
```

- Document = JSON
- Dès qu'un objet est inséré dans une collection, la collection est créée
- db = collection courante définie par **use**
- Collection **personnes** créée dans la base **maBase**

```
> show dbs
admin    <empty>
biblio   0.063GB
local    0.031GB
> show collections
personnes
system.indexes
>
```

admin	
biblio (use)	
personnes	
system.indexes	

# MongoDB

find

- Documents et collection

```
> db.personnes.find()
{ "_id" : ObjectId('56521589adbff04658d0046f'), "nom" : "menier", "prenom" : "gi
ldas", "metier" : "maitre de conferences", "ville" : "vannes" }
>
```

- db.personnes.find()
- Recherche tous les documents
- "\_id" : champs automatique: clé

```
> db.system.indexes.find()
{ "v" : 1, "key" : { "_id" : 1 }, "ns" : "biblio.personnes", "name" : "_id_" }
```

- Rajout :

```
> db.personnes.insert(
... {nom:"bond",
... prenom:"james",
... metier:"espion",
... ville:"Londres",
... acteurs:["Craig", "Connery", "Moore", "Dalton"] })
WriteResult({ "nInserted" : 1 })
>
```

- Structure de document différente (différent de SQL)

# MongoDB

find

- Documents et collection

```
> db.personnes.find()
{ "_id" : ObjectId("56521589adbff04658d0046f"), "nom" : "menier", "prenom" : "gi
ldas", "metier" : "maitre de conferences", "ville" : "vannes" }
{ "_id" : ObjectId("56521888adbff04658d00470"), "nom" : "bond", "prenom" : "jame
s", "metier" : "espion", "ville" : "Londres", "acteurs" : [ "Craig", "Connery",
"Moore", "Dalton" ] }
```

- Sélection : filtre JSON

```
> db.personnes.find(
... { ville:"vannes" } )
{ "_id" : ObjectId("56521589adbff04658d0046f"), "nom" : "menier", "prenom" : "gi
ldas", "metier" : "maitre de conferences", "ville" : "vannes" }
>
```

- db.personnes.find ( **{ ville: "vannes" }** )
- Multiples critères
  - db.personnes.find( **{ ville:"vannes", metier:"boulanger"}** )

# MongoDB

find

- db.restaurants.find( { "ville": "Paris" } )
- Valeur des propriétés
  - { adresse: {num:10, rue:"mollet",ville:"paris"} }
  - db.restaurants.find({"adresse.num":10})
  - " obligatoires
- Conditions sur les champs
  - replacer le champs par une condition
  - db.hotels.find( { etoiles: { \$gt: 2} } )
  - \$gt : *greater than (>)*

# MongoDB

find

- Combinaison
  - and :
    - db.restaurants.find( { "cuisine": "Chinoise", "departement": "75" } )
  - or
    - db.restaurants.find( { \$or: [ { "cuisine": "Chinoise" }, { "departement": "75" } ] } )
  - \$or:[ ]
    - tableau

# MongoDB

find

- Conditions

- `$eq` égalité sur un champ
- `$gt >`
- `$gte >=`
- `$lt <`
- `$lte <=`
- `$ne <>`
- `$in` une valeur dans un tableau
- `$nin` aucune des valeurs d'un tableau
- `$or, $and, $not, $nor`
- `$exist` : si un champ existe
- `$type` : si un champ a le type indiqué

# MongoDB

find

- \$where
  - db.myCollection.find(  
  { **\$where: "this.credits == this.debits || this.credits > this.debits"** } );
  - db.myCollection.find(  
  **"this.credits == this.debits || this.credits > this.debits"** );
    - Si seulement \$where
    - Expression Javascript
  - Fonction possible (pas de "")
    - db.myCollection.find(  
    function() { return ( **this.credits == this.debits || this.credits > this.debits** )  
    }  
  );

# MongoDB

find

- Tri résultat
  - db.restaurants.find().**sort**( { "etoiles": 1, "ville": 1 } )
  - Premier critère étoile + ville
  - Attention
    - 100 Mo / 32 Mo
    - Possibilité de tri sur disque
    - Voir cursor et limit()
  - Voir \$orderby

# MongoDB

update

- Mise à jour
  - db.restaurants.update(  
  { "nom" : "la bonne fourchette" },  
  { **\$set**: { "cuisine": "Française" , "proprio":"dubouchon" },  
    **\$currentDate**: { "lastModified": true } }  
  )
  - **\$set** : remplace la valeur
  - **\$currentDate** : met à jour le champ avec la date courante
    - true : date
    - false : timeStamp

# MongoDB

update

- **\$inc**
  - db.produits.update({ "id":"LFO3456"}, { \$inc : { quantite: -2, prix : 4.5} })
  - quantite = quantite -2
  - prix = prix + 4.5
- **\$mul**
- **\$rename**
  - db.etudiants.update( { \_id: 1 },  
                          { \$rename: { 'nom': 'alias', 'telephone': 'mobile' } } )
- **\$unset**
  - db.etudiants.update( { \_id:1}, { \$unset : {"nom":""}} )

# MongoDB

Update  
tableau

- Tableaux
  - db.collection.update( { <query selector> },  
{ <update operator>: { "array.\$field" : value } } )
- Par exemple :
  - { \_id: 4,  
mesures: [  
  { taille: 80, couleur: 75 },  
  { taille: 85, couleur: 90 },  
  { taille: 90, couleur: 85 }  
] }

# MongoDB

update  
tableau

- db.students.update(  
  { \_id: 4, "mesures.taille": 85 },                      }  
  { \$set: { "mesures.\$couleur" : "bleu" } } )

## \$elemMatch

```
db.students.update(  
  { _id: 4,  
    mesures: { $elemMatch: { taille: { $lte: 90 } } },  
    { $set: { "mesures.$couleur" : "bleu" } }  
  })
```

# MongoDB

Update  
tableau

- **\$addToSet** : rajoute un (ou plusieurs) élément(s) au tableau
- **\$pop** : enlève le premier ou dernier élément d'un tableau
- **\$pushAll**, **\$popAll**, etc..
- **\$each**, **\$position**, etc..
- Voir la doc

# MongoDB

update

- update
  - Un seul document
  - Plusieurs documents : utiliser l'option **multi**

```
db.restaurants.update(  
  { "dept": "75",  
    cuisine: "?" },  
  
  { $set: { cuisine: "Inconnu" },  
    $currentDate: { "lastModified": true } },  
  
  { multi: true }  
)
```

# MongoDB

update

- Attention : piège !!!



```
db.restaurants.update(  
  { "restaurant_id" : "41704620" },  
  { "nom" : "La bonne fourchette" } )
```

```
db.restaurants.update(  
  { "restaurant_id" : "41704620" },  
  { $set: { "nom" : "La bonne fourchette" } } )
```

# MongoDB

update

- update
  - **upsert** : insère si pas trouvé
  - **multi** : plusieurs éléments
  - *writeConcern*: sécurité écriture

```
db.collection.update(  
  <query>,  
  <update>,  
  
  { upsert: <boolean>,  
    multi: <boolean>,  
    writeConcern: <document> }  
)
```

# MongoDB

remove

- remove
  - db.restaurants.remove( { "ville": "Paris" } )
    - Efface tous les documents
  - db.restaurants.remove( { "ville": "Londres" }, { justOne: true } )
    - Efface un seul
- Efface tous les documents
  - db.restaurants.remove({}) ... ou bien *remove()*
    - les documents sont effacés, mais la collection est présente (vide)
  - db.restaurants.drop()
    - la collection est éliminée
- Efface la base de donnée : db.**dropDatabase()**

# MongoDB

indexes

- Indexes
  - par défaut, création de "`_id`" automatique
  - Accès rapide pour un `find / update` avec "`_id`"
    - sinon balayage
  - Prévoir quel champ sert aux recherches
- Simple champ
  - `db.amis.createIndex( { "nom" : 1 } )`
  - clé secondaire (`_id` clé primaire)
  - 1 : indexes croissants

# MongoDB

indexes

- Indexes composés
  - avec un, ou plusieurs champs (ordres croissants ou décroissants)
  - db.produts.createIndex( { "objet": 1, "stock": 1 } )
  - 1 ou -1 (croissants et décroissants)
- Multi clé
  - Indexé par un tableau

# MongoDB

indexes

- Geospatial
  - objets pour représenter :
    - des positions
    - des régions ...
  - opérateurs
    - localisation
    - recherche de régions proches
    - etc..
  - Par exemple :
    - rechercher un magasin proche de coordonnées GPS

# MongoDB



indexes

```
db.places.insert( {  
    name: "Central Park",  
    location: { type: "Point", coordinates: [ -73.97, 40.77 ] },  
    category: "Parks"  
} );
```

```
db.places.insert( {  
    name: "Sara D. Roosevelt Park",  
    location: { type: "Point", coordinates: [ -73.9928, 40.7193 ] },  
    category: "Parks"  
} );
```

```
db.places.insert( {  
    name: "Polo Grounds",  
    location: { type: "Point", coordinates: [ -73.9375, 40.8303 ] },  
    category: "Stadiums"  
} );
```

## Géospatial

# MongoDB



indexes

Création d'un indexe :

```
db.places.createIndex( { location: "2dsphere" } )
```

Requête :

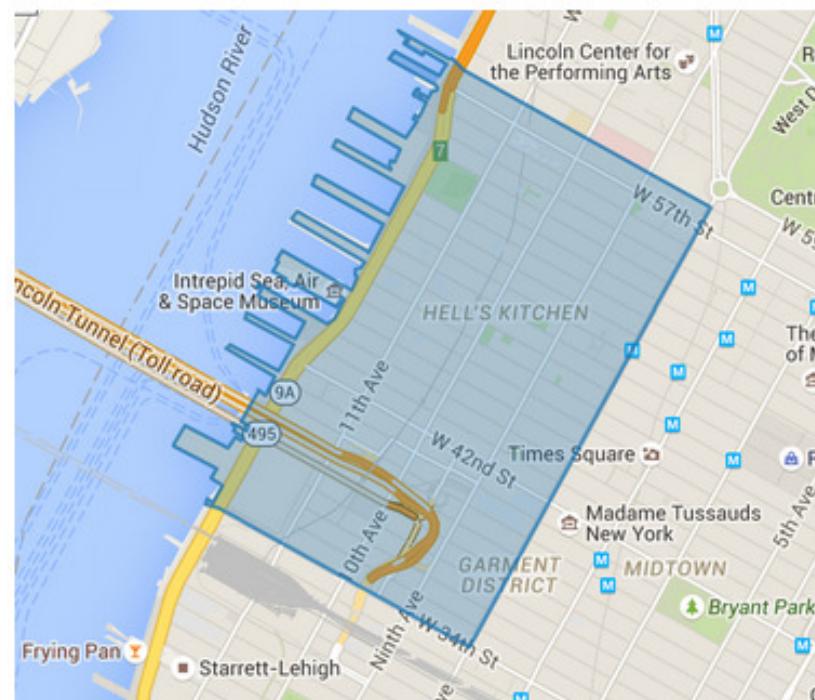
```
db.places.find(
{
  location:
    { $near:
      {
        $geometry: { type: "Point", coordinates: [ -73.9667, 40.78 ] },
        $minDistance: 1000,
        $maxDistance: 5000
      }
    }
}
)
```

```
db.places.insert( {
  name: "Central Park",
  location: { type: "Point", coordinates: [ -73.97, 40.77 ] },
  category: "Parks"
} );
```

## Géospatial

indexes

```
{  
  geometry: {  
    type: "Polygon",  
    coordinates: [[  
      [ -73.99, 40.75 ],  
      ...  
      [ -73.98, 40.76 ],  
      [ -73.99, 40.75 ]  
    ]]  
  },  
  name: "Hell's Kitchen"  
}
```



# MongoDB

indexes

- Indexes et texte (\$text)
  - db.reviews.createIndex( { commentaires: "text" } )
  - Le champ "commentaires" doit être optimisé pour la recherche \$text
  - db.collection.createIndex( { "\$\*\*": "text" } )
    - tous les champs qui contiennent du texte doivent être indexés pour \$text
  - La recherche ignore les mots *stopList*
    - Anglais : the, and, is etc..
    - Français : le la et etc..

# MongoDB

indexes

- Exemple de recherche \$text

- db.articles.createIndex( { sujet: "text" } )
- db.articles.find( { \$text: { \$search: "café", \$language:"fr"} } )
  - Cherche les documents qui contiennent le mot "café" dans le champ "sujet"
- db.articles.find( { \$text: { \$search: "café chocolat thé", \$language:"fr"} } )
  - un des trois
- db.articles.find( { \$text: { \$search: "\"John Crichton\"" } } )
  - phrase
- db.articles.find( { \$text: { \$search: "café chocolat -thé", \$language:"fr"} } )
  - café ou chocolat, mais PAS thé

# MongoDB

- Cursors
  - Bases de données très grandes (4 To = moyen)
  - Résultats de recherche
    - Ne tiennent pas nécessairement en mémoire
    - Pas non plus sur le disque
  - Accès au résultat
    - CURSOR
    - **Itérateur**
  - Find
    - Crédit d'un *cursor*
    - Itérateur sur les résultats
    - Convertir en tableau = idiot

# MongoDB

cursors

- Cursors
  - recherche : accès par itérateur == cursor
  - par défaut dans le REPL : affichage sur les 20 premiers éléments
  - sauvegarde itérateur
    - Javascript

```
var monCurseur = db.films.find( { type: 'horreur' } );
while (monCurseur.hasNext()) {
    print(tojson(monCurseur.next())); // ou printjson
}
```
  - Dans REPL :
    - ou mieux :

```
var monCurseur = db.films.find( { type: 'horreur' } );
monCurseur.forEach(printjson);
```

# MongoDB

cursors

- Méthodes Javascript sur curseur MongoDB
  - `count()` : nombre de documents
  - `forEach( )` : itérateur fonctionnel
  - `hasNext()`
  - `itcount()` : nombre de documents qui restent dans le curseur
  - `limit(n)` : limite aux n premiers
  - `next()` : passe au document suivant (renvoie le document)
  - `sort()` :
  - `toArray()` : convertir le curseur en tableau de document
    - (attention pour la mémoire)

# MongoDB

cursors

- Méthodes Javascript pour les curseurs :

- map
- db.restaurants.find()

```
{ ville:"paris" } ).map( (r) => { return r.nom } )
```

- récupérer l'ensemble des noms des restaurants de Paris
- tableau de noms

# MongoDB

utils

- Utilitaires DB
  - `db.cloneCollection()`
    - copie d'un nœud MongoDB à l'autre
  - `db.cloneDatabase()`
    - copie d'un nœud MongoDB à l'autre
- Utilitaires ligne de commande
  - `mongoexport --db test --collection traffic --out traffic.json`
    - Exporte la collection ‘traffic’ de la base ‘test’ sous la forme d’un fichier json
    - JSON, CSV, TSV
    - Local ou pas
  - `mongoimport --db test --collection traffic --type json --file seed.json`
    - --host --port
    - Local ou pas

# MongoDB

- Utilitaires ligne de commande
  - Imports / exports binaires
    - mongodump / mongorestore
      - BSON (sérialisation binaire de JSON)
    - bsondump
      - Conversion BSON vers JSON
  - Diagnostique
    - mongostat
    - mongosniff
    - mongotop
    - mongoperf
  - GridFS
    - mongofiles (voir plus loin) – BSON > 16 Mo

# MongoDB

**utils**

- Interface REST & HTTP
  - Option --rest
  - Attention à l'administration / sécurité
  - Numéro de port 1000 de plus : -- port 8080 -> 9080

**mongod white:8081**

[List all commands](#) | [Replica set status](#)

Commands: [buildInfo](#) [cursorInfo](#) [features](#) [hostInfo](#) [isMaster](#) [listDatabases](#) [replSetGetStatus](#) [serverStatus](#) [top](#)

```
db version v2.4.9
git hash: nogitversion
sys info: Linux kishi05 3.2.0-60-highbank #91-Ubuntu SMP PREEMPT Wed Feb 19 04:47:26 UTC 2014 armv7l BOC
uptime: 39679 seconds
```

**overview** (only reported if can acquire read lock quickly)

```
time to get readlock: 0ms
# databases: 4
# Cursors: 0
replication:
master: 0
slave: 0
```

## clients

Client	Opid	Locking	Waiting	SecsRunning	Op	Namespace	Query
initandlisten	6		{ waitingForLock: false }		2002	local.startup_log	
DataFileSync	0		{ waitingForLock: false }		0		
snapshotthread	2		{ waitingForLock: false }		0		
clientcursormon	3		{ waitingForLock: false }		0		
TTLMonitor	2681		{ waitingForLock: false }		2004	restos.system.indexes	{ expireAfter
websvr	7		{ waitingForLock: false }		0	admin.system.users	

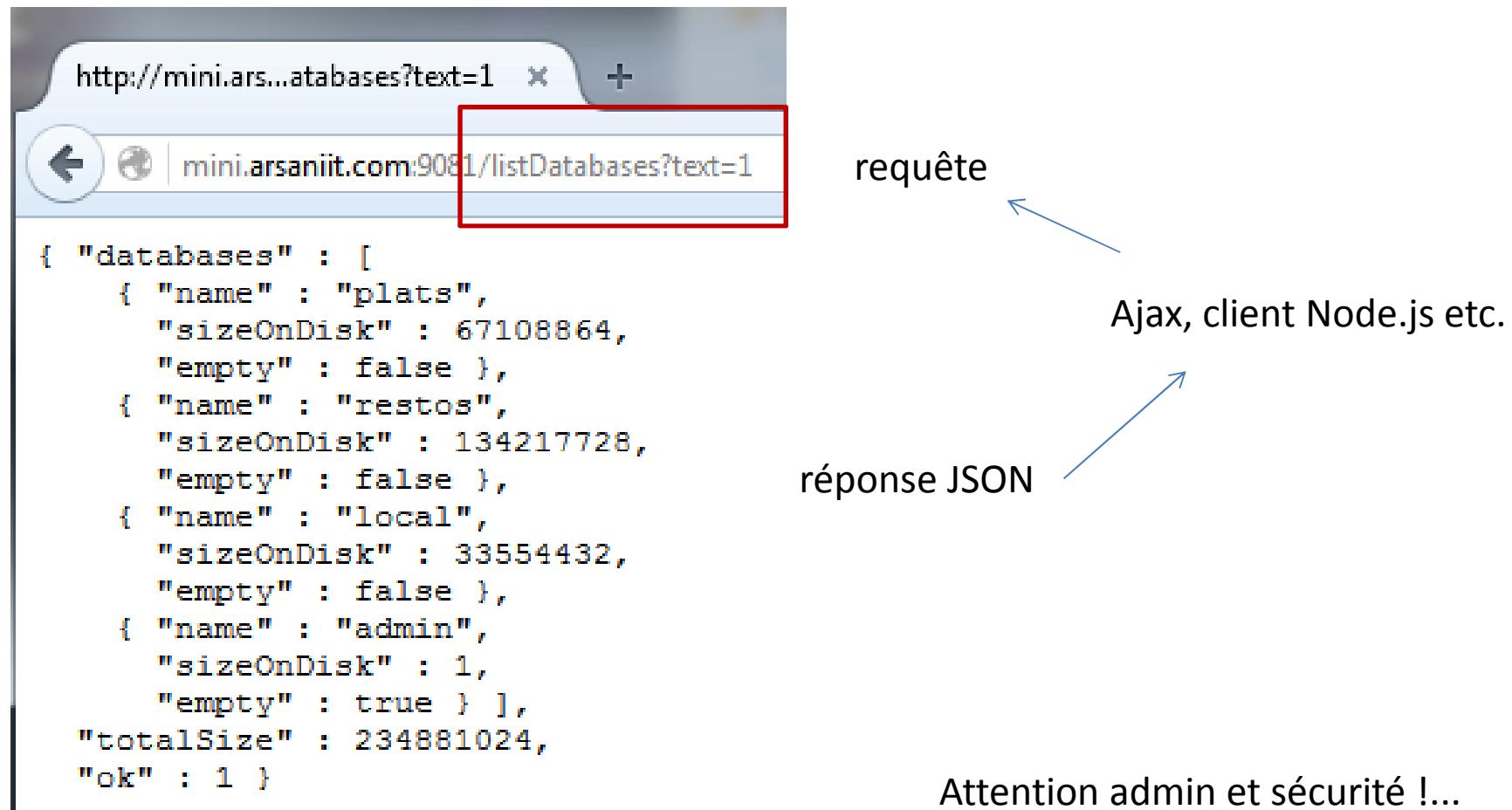
**dbtop** (occurrences|percent of elapsed)

NS	total	Reads	Writes	Queries	GetMores	Inserts	Updates	Removes
TOTAL	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%

write lock % time in write lock by 1 sec periods

# MongoDB

- Interface REST & HTTP



# MongoDB

- Node.js & MongoDB
  - **npm install mongodb**
  - Connection simple à la base ‘restos’

```
var MongoClient = require('mongodb').MongoClient;
```

```
= MongoClient.connect("mongodb://lo.testserver.com:87/restos", function(err, db) {  
=   if(!err) {  
=     console.log("Connecté");  
=   }  
});
```

- host : lo.testserver.com
- port : 87
- Plusieurs manières
  - MongoClient = simple

# MongoDB

- Node.js & MongoDB

```
var MongoClient = require('mongodb').MongoClient;

// Connection à la base
MongoClient.connect("mongodb://host:89/restos", (err, db) => {
  if(err) { return console.dir(err); }

  db.createCollection('villes', (err, collection) => {});
  var collection = db.collection('villes')
  var doc1 = { nom: "paris", pays:"France" }
  var doc2 = { nom: "londres", pays:"Angleterre" }
  collection.insert(doc1)
  collection.insert(doc2)

  collection.find().forEach( v => console.log(v))
});

});
```

?

```
F:\archives\__Desk\Cours\testmongodb>node cnx.js
{ nom: 'paris', pays: 'France', _id: 56562eb20e1f58d00357ebef }
{ nom: 'londres',
  pays: 'Angleterre',
  _id: 56562eb20e1f58d00357ebef }
```

# MongoDB

```

var MongoClient = require('mongodb').MongoClient;

// Connection à la base
MongoClient.connect("mongodb://host.com:87/", (err, db) => {
  if(err) { return console.dir(err); }

  var nouvDB = db.db("plats")
  nouvDB.createCollection("desserts", (err, db) => {
    if (!err) {
      var collection = nouvDB.collection("desserts")
      collection.insert( { nom:"creme", type:"renverse" } )
      collection.insert( { nom:"glace", type:"lait" } )
      collection.find().forEach( v=>console.log(v))
    }
  })
});
```

?

Insert : writeconcern...

```

F:\archives\__Desk\Cours\testmongodb>node cnx.js
{ nom: 'creme', type: 'renverse', _id: 56563264034b5b3813ee82ad }
{ nom: 'glace', type: 'lait', _id: 56563264034b5b3813ee82ae }
```

Récupérer les noms des desserts :

```
collection.find().map( v => v.nom ).forEach( v=>console.log("dessert : " + v))
```

(fonctionnel !)

# MongoDB

Récupérer la liste des bases

```
var MongoClient = require('mongodb').MongoClient;

// Connection à la base
MongoClient.connect("mongodb://mini.arsaniit.com:8081/admin", (err, db) => {
  if(err) { return console.dir(err); }

  var adminDB = db.admin();
  adminDB.listDatabases( (err, dbs) => {
    |   dbs.databases.map( v=> v.name).forEach( v=> console.log(v))
    |
  }
}

});
```

```
{ databases:
  [ { name: 'plats', sizeOnDisk: 67108864, empty: false },
    { name: 'restos', sizeOnDisk: 134217728, empty: false },
    { name: 'local', sizeOnDisk: 33554432, empty: false },
    { name: 'admin', sizeOnDisk: 1, empty: true } ],
  totalSize: 234881024,
  ok: 1 }
```

```
plats
restos
local
admin
```

Voir collectionNames

# MongoDB

Création d'une base

une collection est créée à l'ajout d'un document

une base est créée à l'ajout d'une collection...

```
var MongoClient = require('mongodb').MongoClient;

// Connection à la base
MongoClient.connect("mongodb://mini.arsaniit.com:8081/", (err, db) => {
  if(err) { return console.log(err); }

  var newDB = db.db("nouv")
  newDB.createCollection("maCollection", (err, collection) => {
    if (!err) {
      console.log("base créée")
    }
  })
});
```

# MongoDB

Parenthèse : Javascript et callback

```
f1(param, (err, res) => {
  if (err) {
    // gestion erreur 1
  } else {
    f2(res, (err, res) => {
      if (err) {
        // gestion erreur 2
      } else {
        f3(res, (err, res) => {
          if (err) {
            // gestion erreur 3
          } else {
            f4(res)
          }
        })
      }
    })
  }
})
```

Remarque 1

Critique callback

# MongoDB

Parenthèse : Javascript et callback

```
f1(param, then_f2)
    ↓
function then_f2(err, res) {
  if (err) {
    // gestion erreur de f1
  } else f2(res, then_f3)
}

function then_f3(err, res) {
  if (err) {
    // gestion erreur de f2
  } else f3(res, then_f4)
}

function then_f4(err, res) {
  if (err) {
    // gestion erreur de f3
  } else f4(res)
}
```

Continuation

(autre solution plus simple : ...promises – voir plus loin)

# MongoDB

## Remarque 2 :

Fermeture des ressources !!!!

Requêtes asynchrones :

    chaque requête ouvre un client  
    fermeture par db.close !!!

```
MongoClient.connect( ...., (err, db) => {  
    if (err) {  
        // gestion erreur  
    } else {  
        // traitement  
        // ... et  
        db.close()  
    }  
})
```



# MongoDB

Connection à une base à partir de node.js

MongoClient

Utilisation d'une chaîne de connexion = le plus simple

MongoClient.connect( chaineCnx, options, callback)

chaineCnx **mongodb://username:password@host/database?option**

**option :**

```
MongoClient.connect( "mongodb://menier:carambar@localhost:27017/test", {  
  db: {w: 1, native_parser: false},  
  server: { poolSize:5, socketOptions: { connectTimeoutMS: 500 },  
            auto_reconnect: true },  
  replSet:{},  
  mongos:{}  
}, (err, db) => {  
  if (err) {  
  } else {  
    // traitements...  
    db.logout( (err, res) => {  
      if (err){  
      } else {  
        db.close()  
      }  
    })  
  }  
});
```

# MongoDB

Connection à une base à partir de node.js

**db: {w: 1, ...}**

**niveau de garantie en écriture (important)**

*Write concern*

niveau de garantie d'écriture

-1 : ignore les erreurs réseau

0 : pas de vérification d'écriture

1 : vérification d'écriture

2 : vérification pour le nœud root + secondaire

majorité : demandé pour une majorité de noeuds

# MongoDB

Comment obtenir un **db** ?

Objet db :

`MongoClient.connect( ...., (err, db)=> { ... } )`

- db.admin()** : création d'un objet admin (voir plus loin)
- db.open(cb)** : voir Server
- db.db(nom)** : nouvelle instance sur la base
- db.collectionInfo([nom], cb)** : curseur sur les collections (ou la collection)
- db.collectionNames(cb)** : liste des noms de collection
- db.collection(nom, [opt], cb)** : objet Collection
- db.collections(cb)** :
- db.logout(cb)** : déconnection
- db.authenticate(u, p, cb)** : changement d'utilisateur
- db.addUser(u, p, cb)** : rajout d'un utilisateur
- db.removeUser(u, cb)** :
- db.createCollection(nom, cb):**
- db.dropCollection(nom, cb) :**
- db.renameCollection(anc,nouv,cb)**
- db.dropDatabase(nom, cb)**

# MongoDB

## Objet Admin

`var ad = db.admin()`

Comment obtenir un **ad** ?

: création d'un objet admin (voir plus loin)

`ad.serverStatus(cb)`

(hôte, version, temps d'arrêt etc..)

: infos sur le serveur en cours

`ad.ping(cb)`

: ping le serveur

`ad.listDatabases(cb)`

: liste des bases de données

`ad.authenticate(u,p,cb)`

`ad.logout(cb)`

`ad.addUser(u,p,cb)`

`ad.removeUser(u,p,cb)`

# MongoDB

## Objet Collection

Comment obtenir un col ?

```
var col = db.collection() // cf URL
var col = new Collection(db, "test")
db.createCollection("test", (err, col) => { ... })
```

- col.insert(docs, cb)** : insérer un ou plusieurs documents dans la col
- col.remove([q],[o],[cb])** : enlever des documents.
- col.rename(nouvNom, cb)** : renomme la collection
- col.save([doc],[o],[cb])** : préférer update ou findAndModify
- col.update(q, up, [o],[cb])** : query, update
- col.find(q,[o],cb)** : fabrique un cursor
- col.findOne(q, [o], cb)** : un cursor avec le premier trouvé
- col.findAndModify(q,s,u,[o],cb)** : modifications en place (ordre avec sort)
- col.findAndRemove(q,s,[o],cb)** : cf remove (+ sort)
- col.distinct(clé, [q], cb)** :

# MongoDB

## Exemple d'utilisation de *distinct*

```
{ "_id": 1, "dept": "A", "item": { "sku": "111", "color": "red" }, "sizes": [ "S", "M" ] }  
{ "_id": 2, "dept": "A", "item": { "sku": "111", "color": "blue" }, "sizes": [ "M", "L" ] }  
{ "_id": 3, "dept": "B", "item": { "sku": "222", "color": "blue" }, "sizes": "S" }  
{ "_id": 4, "dept": "A", "item": { "sku": "333", "color": "black" }, "sizes": [ "S" ] }
```

```
distinct( "dept" )  
[ "A", "B" ]
```

```
distinct( "item.sku" )  
[ "111", "222", "333" ]
```

```
distinct( "item.sku" , { dept:"B"} )  
[ "222" ]
```

# MongoDB

## Objet Collection

Comment obtenir un **col** ?

```
var col = db.collection()    // cf URL  
var col = new Collection(db, "test")  
db.createCollection("test", (err, col) => { ... })
```

... (suite)

**col.count([q], cb)**

: combien de documents ?

**col.drop(cb)**

: vide la collection

**col.stats(cb)**

: infos sur la collection

nb éléments,  
taille sur le disque,  
taille moyenne obj etc..

# MongoDB

## Objet Cursor

Comment obtenir un **cur** ?

```
col.find( ..., (err, cur) => { ... } )
db.collectionInfo(..., (err, cur) => { ... })
```

<code>cur.each(cb)</code>	: itération <code>cb = (err, item) =&gt; { ... }</code>
<code>cur.toArray(cb)</code>	: fabrique un tableau (attention)
<code>cur.nextObject(cb)</code>	: objet suivant
<code>cur.rewind()</code>	: revient au début des objets
<code>cur.count(cb)</code>	: combien d'éléments
<code>cur.close(cb)</code>	: libère mémoire client + serveur
<code>cur.isClosed()</code>	
<code>cur.sort(kl, dir, cb)</code>	: tri

`kl = "nom" ou bien ["nom", "prenom"]`

`dir = 1 (croissant) ou -1 (décroissant)`

# MongoDB

## Options

w	: write concern
wtimeout	: temps maxi pour l'écriture
fsync	: si vrai, attente de fsync
journal	: si vrai attente sync journal
serializeFunctions	: sérialisation des fonctions ou pas
forceServerObjectId	: pas toucher
checkKeys	: pas toucher
upsert	: si update et pas de doc : création
multi	: si vrai, modification pour les docs trouvés
new	: findAndModify renvoie le nouveau (défaut = false)

# Promesses

- Encapsule le mécanisme des callbacks
- Un appel asynchrone peut dépendre d'un appel asynchrone qui peut dépendre ...
- Imbrication



# Promesses



- Mécanisme classique

```
function caMarcheCallback(resultat) {  
    console.log("L'appel async donne " + resultat);  
}
```

Attention à l'ordre

```
function rateCallback(erreur) {  
    console.log(" problème code " + erreur);  
}
```

```
fonctionAsync(caMarcheCallback, rateCallback);
```

La fonctionAsync n'est pas bloquante et calcule **resultat** (ou se termine **mal**)

# Promesses



- Une promesse (promise)

```
function caMarcheCallback(resultat) {  
    console.log("L'appel async donne " + resultat);  
}
```

```
function rateCallback(erreur) {  
    console.log("problème code " + erreur);  
}
```

```
fonctionAsync().then(caMarcheCallback, rateCallback);
```

**then** est une méthode de fonction (!) qui renvoie une autre promesse  
(cf prototypes + une fonction est un objet comme les autres)

# Promesses



- Chaînage des promesses

```
fonctionAsyncRouge(function(resultat1) {  
    fonctionAsyncVerte(resultat1, function(resultat2) {  
        fonctionAsyncBleue(resultat2, function(finalResultat) {  
            console.log('resultat final ' + finalResultat);  
  
        }, bleueRateCallback);  
    }, verteRateCallback);  
}, rougeRateCallback);
```

# Promesses



- Chaînage des promesses

```
OuvrirUnLienVersLaBDDistante(function(socket) {  
    trouverLaBase(socket, function(lienBase) {  
        lireLeNom(lienBase, function(leNom) {  
            console.log('son nom est ' + leNom);  
  
        }, bleueRateCallback); // le nom n'existe pas dans la base  
    }, verteRateCallback); // la base n'existe pas  
}, rougeRateCallback); // impossible d'ouvrir socket
```

# Promesses



- Chaînage des promesses

```
OuvrirUnLienVersLaBDDistante().then(function(socket) {  
    return trouverLaBase (socket);  
})  
.then(function(lienBase) {  
    return lireLeNom(lienBase);  
})  
.then(function(leNom) {  
    console.log('le nom est ' + leNom);  
})  
.catch(rateCallback);  
        .then(null, rateCallback)
```

Un seul

# MongoDB

Parenthèse : Javascript et callback

```
f1(param, (err, res) => {
  if (err) {
    // gestion erreur 1
  } else {
    f2(res, (err, res) => {
      if (err) {
        // gestion erreur 2
      } else {
        f3(res, (err, res) => {
          if (err) {
            // gestion erreur 3
          } else {
            f4(res)
          }
        })
      }
    })
  }
})
```

Remarque 1

Critique callback

# MongoDB

Parenthèse : Javascript et callback

```
f1(param, then_f2)
    ↓
function then_f2(err, res) {
  if (err) {
    // gestion erreur de f1
  } else f2(res, then_f3)
}

function then_f3(err, res) {
  if (err) {
    // gestion erreur de f2
  } else f3(res, then_f4)
}

function then_f4(err, res) {
  if (err) {
    // gestion erreur de f3
  } else f4(res)
}
```

Continuation

(autre solution plus simple : ...promises – voir plus loin)

# Promesses



- Chaînage des promesses

```
ouvrirUnLienBDDistance()  
.then(socket => trouverLaBase(socket))  
.then(lienBD => lireLeNom(lienBD))  
.then(leNom => {  
    console.log(`le nom est: ${leNom}`);  
})  
.catch(rateCallback);
```

```
.then(null, rateCallback)
```

# Promesses



- Chaînage des promesses

```
ouvrirUnLienBDDistance()
.then(socket => trouverLaBase(socket))
.then(lienBD => lireLeNom(lienBD))
.then(leNom => {
  console.log(`le nom est: ${leNom}`);
})
.catch(rateCallback)
.then( () => ... ) // équivalent finally
```

# Promesses



- Définition

```
var promise = new Promise(function(resolve, reject) {  
    // faire une action (éventuellement asynchrone)
```

```
    if /* ok */ {  
        resolve("ca marche");  
    }  
    else {  
        reject(Error("pb"));  
    }  
});
```

# Promesses

- Chaînage des promesses



```
async function myFirstAsyncFunction() {  
    try {  
        const fulfilledValue = await promise;  
    }  
    catch (rejectedValue) {  
        // ...  
    }  
}
```

Ecmascript 2017

# Promesses



- Chaînage des promesses

```
async function lireLeNomDansLaBaseDistante() {  
    try {  
        let socket = await ouvrirLienBDDistante();  
        let lienBD = await ouvrirLaBaseDonnee(socket);  
        let leNom = await lireLeNom(lienBD);  
        console.log(`le nom est : ${leNom}`);  
    } catch(erreur) {  
        rateCallback(erreur);  
    }  
}
```

# Promises



```
function getResponseSize(url) {  
    return fetch(url).then(response => {  
        const reader = response.body.getReader();  
        let total = 0;  
  
        return reader.read().then(function processResult(result) {  
            if (result.done) return total;  
  
            const value = result.value;  
            total += value.length;  
            console.log('Received chunk', value);  
  
            return reader.read().then(processResult);  
        })  
    });  
}
```

# Promises



```
async function getResponseSize(url) {  
    const response = await fetch(url);  
    const reader = response.body.getReader();  
    let result = await reader.read();  
    let total = 0;  
  
    while (!result.done) {  
        const value = result.value;  
        total += value.length;  
        console.log('Received chunk', value);  
        // get the next result  
        result = await reader.read();  
    }  
  
    return total;  
}
```

# Moongoose

Création de schemas

Similaire à XML et schemas

Grammaire de structure

```
var mongoose = require('mongoose');
var Schema = mongoose.Schema;

var blogSchema = new Schema({
  title: String,
  author: String,
  body: String,
  comments: [{ body: String, date: Date }],
  date: { type: Date, default: Date.now },
  hidden: Boolean,
  meta: {
    votes: Number,
    favs: Number
  }
});
```

```
var Blog = mongoose.model('Blog', blogSchema);
```

# Moongoose

```
// define a schema
var animalSchema = new Schema({ name: String, type: String });

// assign a function to the "methods" object of our animalSchema
animalSchema.methods.findSimilarTypes = function(cb) {
  return this.model('Animal').find({ type: this.type }, cb);
};
```