

Technologies web

SÉMINAIRE DÉCOUVERTE – EFREI 2017

Je suis - je suis - je suis ?

- ☐ Pierrick Bignet
- ☐ Game Designer et développeur
- ☐ Président de Lone Stone Studio
- ☐ Projet actuel : City Invaders



LONESTONE.

Les technologies web pour vous





1. Le web 1.0 – HTML et CSS

- Couple inséparable toujours au cœur du web d'aujourd'hui
- HTML : langage de templating. Le contenu de la page web.
- CSS : langage de style. Ce qui décide de l'apparence de la page

```
body {  
  background-color: lightblue;  
}  
  
h1 {  
  color: white;  
  text-align: center;  
}  
  
p {  
  font-family: verdana;  
  font-size: 20px;  
}
```

```
<body>  
  
  <h1>My First CSS Example</h1>  
  <p>This is a paragraph.</p>  
  
  <h2>Introduction à l'internet des réseaux</h2>  
</body>
```



1.1. HTML - Hyper Text Markup Language

- Le HTML décrit le contenu de la page
- Le HTML utilise des « éléments » / « blocs » pour représenter le contenu
- Les éléments sont représentés par des tags, qui représentent le contenu à l'intérieur du bloc
- Exemples
 - `<p>` Est le tag utilisé pour les paragraphes `</p>`
 - `<h1>` Est le tag utilisé pour un titre principale. Il existe `h2`, `h3`, etc. `</h1>`
- On utilise (aujourd'hui souvent en bout de chaîne) des fichiers `.html`



Exemple de structure

```
<html>
```

```
<head>
```

```
<title>Page title</title>
```

```
</head>
```

```
<body>
```

```
<h1>This is a heading</h1>
```

```
<p>This is a paragraph.</p>
```

```
<p>This is another paragraph.</p>
```

```
</body>
```

```
</html>
```



Quelques tags importants

- Pour des images : ``
- Pour des tableaux : `<table>` , `<tr>` pour les lignes et `<td>` pour une cellule et `<th>` pour les headers
- Des listes : `` pour la liste et `` pour chaque entrée
- Un lien hypertexte, avec `<a>`
- Avec le HTML5 sont arrivés de nouveaux éléments : `<audio>` et `<video>`



Blocks, Inline et Class

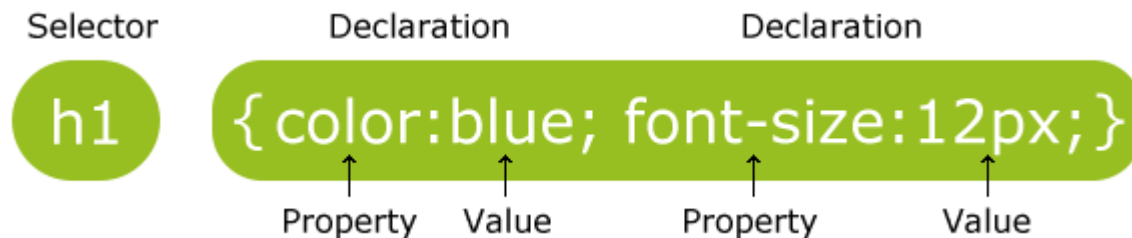
- En HTML on distingue les éléments de type « block » et ceux de type « inline »
 - Block : `<div>`, `<p>`, `<h1>`, etc.
 - Inline : `<a>`, ``, etc.
- Le block `<div>` est très souvent utilisé pour mieux organiser une page web. On lui applique des styles CSS, généralement via des classes
- Les classes permettent d'appliquer facilement des styles CSS à tous les éléments HTML qui ont cette classe.

1.2. Le CSS – Cascading Style Sheets



Le style, c'est important

- Sans CSS le web ressemblerait à de gros tableaux moches : c'est une partie indispensable
- Le CSS est stocké dans des fichiers séparés, .css



- Le selector peut être de plusieurs types différents :
 - Element Selector : p, h1, etc.
 - id Selector : #firstParagraphe, #bigTitle
 - Class Selector : .cities, h1.centers

Note: On peut insérer du CSS « inline », mais c'est très fortement déconseillé

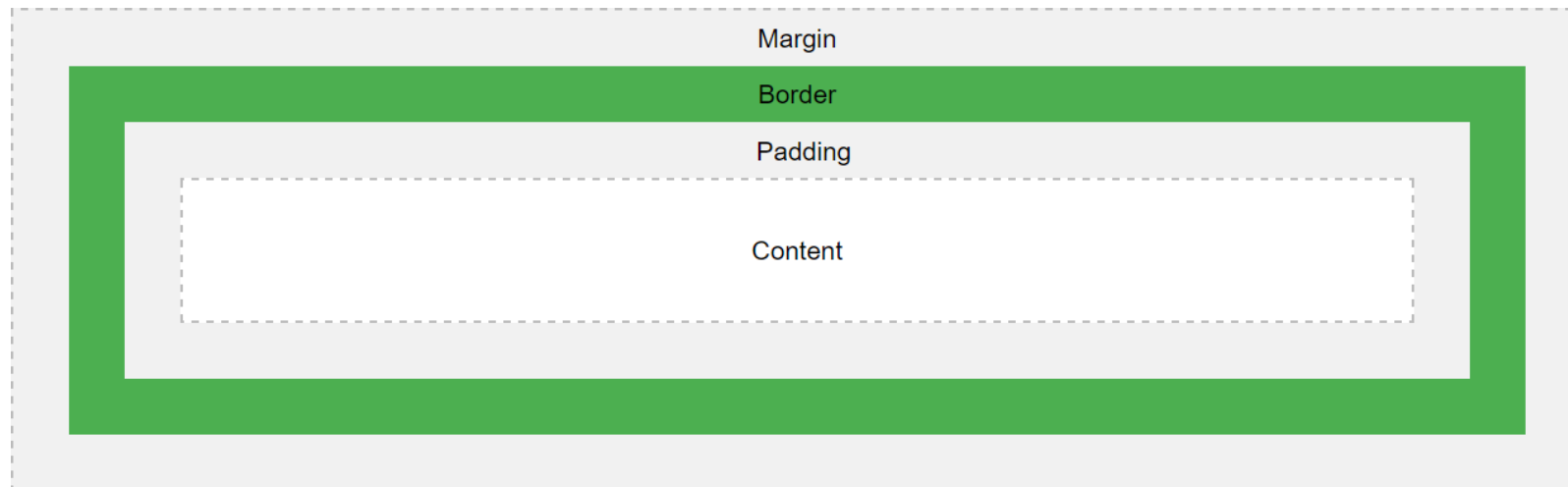


CSS : quelques exemples

- Background : color, image
- Border : style, width, color. On peut aussi le définir côté par côté
- Margin et padding
- Fonts
- Outline : style & color
- Display : block ou inline
- Align

Box Model

- Content – Le contenu de la box. C'est ici qu'apparaissent le texte, image, etc.
- Padding – Le padding est transparent ; il libère un espace autour du contenu
- Border – Un bord qui entour le **content + padding**
- Margin – La margin est transparente ; elle libère un espace autour du border





Height/Width

- Ce sont 2 valeurs primordiales : elles permettent de bien mieux organiser l'aspect visuel d'un site
- **Attention** ! La Height et Width n'incluent pas le padding, borders et margin ! Mais seulement le content de la box !
- Max-width



Position et Float

- Autres paramètres indispensables pour une bonne mise en page
- Il existe 4 méthodes de positionnement pour un élément :
 - Static : pas affecté par top, bottom, left et right. Positionnement par défaut
 - Relative : positionné relativement à son top, bottom, left et right
 - Fixed : Reste toujours à la même place par rapport au viewport (fenêtre globale)
 - Absolute : Positionné de façon absolue par rapport au dernier bloc positionné (c-a-d bloc non static)
- Float Bill !

Responsive



Solutions

- Depuis CSS2 on peut tout faire à la main via les media queries

```
@media screen and (min-width: 480px) {  
  body {  
    background-color: lightgreen;  
  }  
}
```

- Le plus simple pour commencer : utiliser un style css avec les classes adaptées. Le plus répandu est Bootstrap (par l'équipe de Twitter). Bootstrap utilise un système de grille.





2. PHP : Sortir de la page « statique »

- Avec du HTML pur on ne peut produire que des pages statiques, avec tout leur contenu écrit dans les fichiers HTML
- PHP est un des premiers langages à offrir la possibilité de générer du HTML à partir de données dynamiques (par exemple depuis une base de donnée)
- Malgré son âge et ses (nombreuses) lacunes, PHP est toujours le langage serveur le plus utilisé aujourd'hui
- Wordpress, Facebook, et bien d'autres services sont ou ont été basés sur du PHP
- Contrairement à HTML qui n'est qu'un langage de templating, PHP est un « vrai » langage de programmation





PHP : usages

- Formulaires (exemple)
- Vérification des données
- Pages dynamiques avec chargement de données depuis une BDD
- Le PHP permet aussi de lire et écrire sur des fichiers sur le serveur. Donc d'enregistrer des images, des sons, etc. uploadés par l'utilisateur
- PHP permet la gestion de sessions



3. Le Javascript et le Web 2.0

- Le Javascript est au centre du développement web aujourd'hui
- Il est sur toute la stack: front-end comme back-end
- Côté front-end le javascript a la capacité de changer dynamiquement le contenu d'une page HTML :
 - Le contenu d'un `<p>`
 - Le style d'un `<bouton>`
 - On peut créer de nouvelles `<div>`

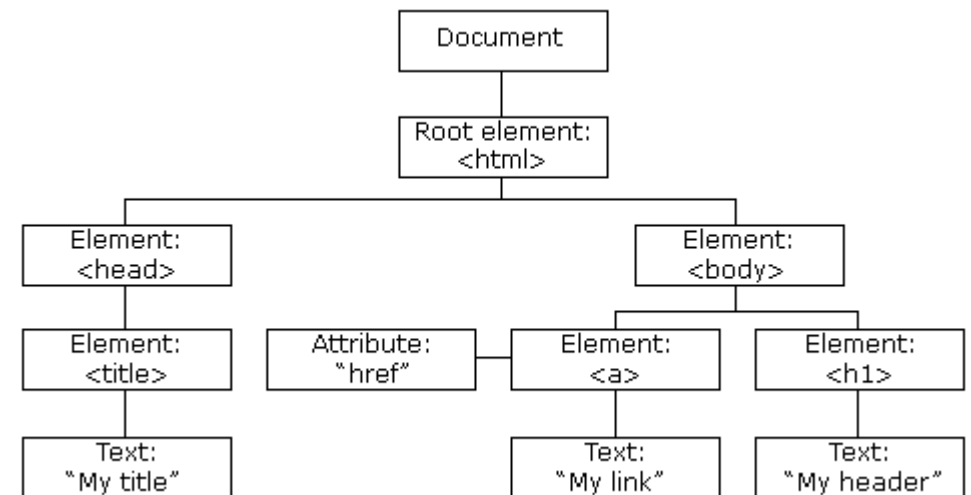


Une histoire chaotique

- Prototype based : on peut ajouter et enlever des propriétés d'un objet au runtime. Le javascript n'est pas basé sur le concept de classes et d'héritage de Java ou C#
- Des features qui arrivent au fur et à mesure (amélioration du scoping, etc.)

JavaScript HTML DOM

- On peut modifier tous les éléments HTML d'une page
- On peut modifier tous les attributs HTML d'une page
- On peut modifier tous les styles CSS d'une page
- On peut ajouter des éléments ou des attributs HTML
- On peut aussi en retirer
- JavaScript peut réagir à tous les Events HTML
- Ou en créer de nouveaux



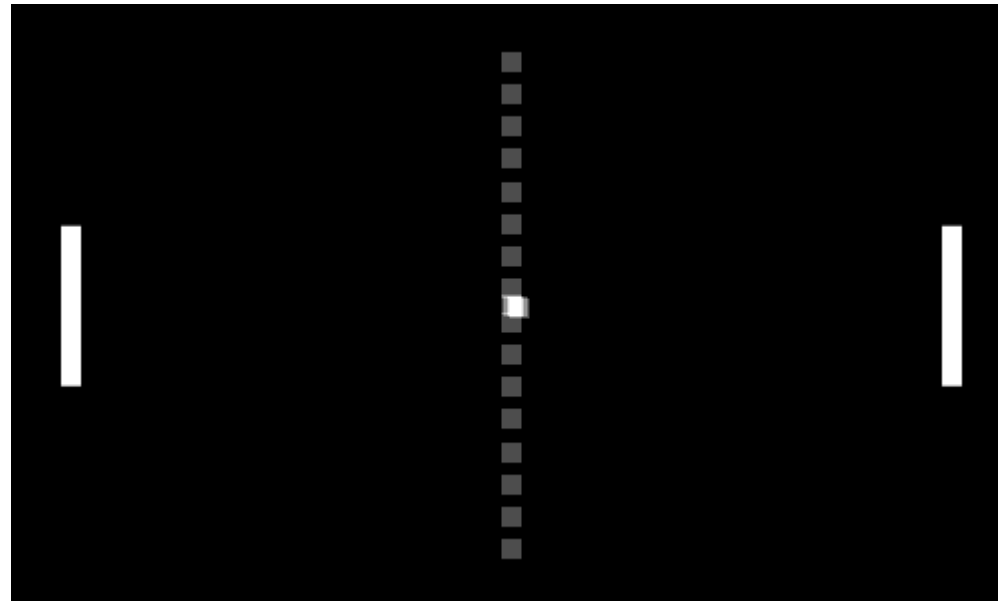


Accéder au DOM

- `getElementById` : on récupère un élément du DOM via son ID
- `getElementById('title').innerHTML` : le contenu HTML de l'élément récupéré
- Une fois qu'on a un élément on peut tout modifier :
 - `element.setAttribute(attribute, value)`
 - `element.style.property = new style`
- On peut aussi ajouter / enlever des élément au document (le D du DOM)
 - `document.createElement(element)`
 - `document.removeChild(element)`
- On a d'autres manière que l'id de retrouver un élément :
 - Tag name
 - Class name
 - CSS selectors
 - Object collections

Events, animations

- En utilisant les events onclick, onmouseover, onmouseout, onchange, etc. on peut réagir aux input utilisateur très facilement et lancer des méthodes javascript
- On peut animer des éléments en Javascript. Javascript peut, frame par frame, modifier la position d'un objet, ce qui le rend donc adapté à des usages très varies. Notamment le jeu video !



Ajax

- Le PHP permet de récupérer des données stockées, le javascript de transformer la page
- Mais pour le moment on ne sait pas rafraichir la page chargée en utilisant des données du serveur ...

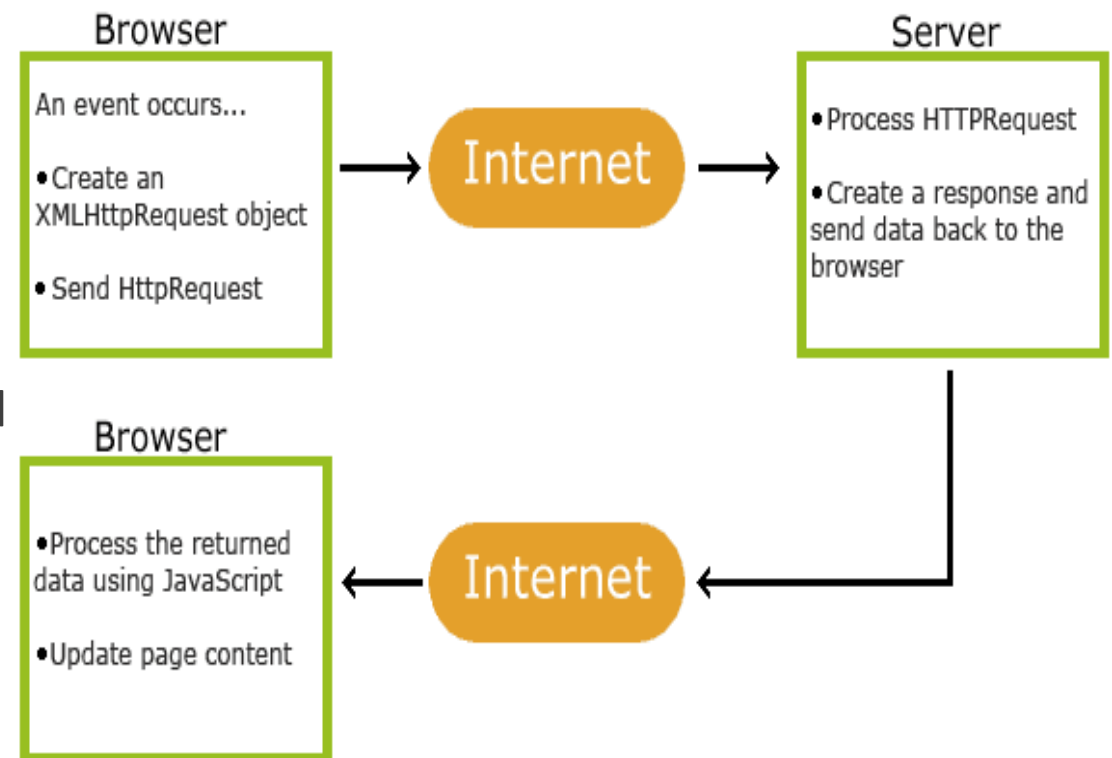


C'est là qu'Ajax intervient !

Ajax : page dynamique sans reload

- Mettre à jour une page sans la recharger complètement
- **Demander** des données au serveur **après** le chargement de la page
- **Recevoir** des données depuis le serveur **après** le chargement de la page
- **Envoyer** des données au serveur en tâche de fond

Note : Malgré « XML » dans son nom, aujourd'hui AJAX est surtout utilisé avec le format JSON





Json : Le format d'échange par défaut

- Suivant la montée en puissance de Javascript, le Json s'est rapidement imposé
- On peut transformer des données Json en objets Javascript et inversement de façon complètement transparente

4. Javascript côté serveur (backend)

- Assez naturellement le javascript est passé du client (browser) aux serveur
- 1 seul langage sur toute la stack simplifie le développement et les échanges (notamment en json)
- Utilisé par ;
 - Paypal
 - Yahoo
 - Groupon
 - LinkedIn
 - Netflix
 - Uber





Découverte de Node.JS

- Exemple de cityinvaders-server
- Exemple de cityinvaders-website
- Exemple de cityinvaders-admin

5. Le JS et le développement web en ~~2016~~ 2017





5.1. Le front-end

- De nombreuses technologies différentes pour différents usages
- Bien réfléchir avant de choisir une techno ! Marteau vs Mouche



5.1.1. CSS Pre-processors

- 3 choix principaux

- Sass
- Less
- Stylus

- Des features partagés:

- Variables
- Mixins
- Nesting
- Loops, if/else


Note : Les fichiers .less, .scss ou .stylus ne peuvent pas être interprétés par un navigateur !

Ils doivent d'abord être converti en fichiers .css avant d'être envoyés au client.

Variables

- Changer rapidement les couleurs d'un site
- Clarifier le code en évitant les string répétées un peu partout
- Simplifie le refactoring

```
/* Scss */
$primary-color: #3bbfce;
$margin: 16px;
.border {
  padding: $margin / 2;
  margin: $margin / 2;
  border-color: $primary-color;
}
```



```
/*CSS*/
border {
  padding : 8px;
  margin : 8px;
  border-color: #3bbfce;
}
```


Mixins

- Eviter les répétitions et simplifier le refactoring
- Créer des styles visuels basés sur 1 seule couleur (darken, lighten, etc.)
- Ce sont des fonctions qui retournent du code less/sass, et peuvent prendre des arguments

```
/*Less*/
.rounded-corners (@radius: 5px 10px 8px 2px) {
  -webkit-border-radius: @radius;
  -moz-border-radius: @radius;
  border-radius: @radius; }

#header {
  .rounded-corners;
}

#footer {
  .rounded-corners(10px 25px 35px 0px);
}
```



```
/*CSS*/
#header {
  -webkit-border-radius: 5px 10px 8px 2px;
  -moz-border-radius: 5px 10px 8px 2px;
  border-radius: 5px 10px 8px 2px; }

#footer {
  -webkit-border-radius: 10px 25px 35px 0px;
  -moz-border-radius: 10px 25px 35px 0px;
  border-radius: 10px 25px 35px 0px; }
```



```
1 @import 'modules/normalize.scss';
2 @import 'modules/reset.scss';
3 @import '/js/vendor/bootstrap/dist/css/bootstrap.css';
4 @import '/js/vendor/bootstrap/dist/css/bootstrap-theme.css';
5
6 @mixin getLang($lang) {
7   @at-root {
8     @if ($lang==chinese) {
9       [lang="zh-Hans"] & {
10        [lang="zh-Hant"] & {
11          @content;
12        }
13      }
14    @else if ($lang==jp) {
15      [lang="ja"] & {
16        @content;
17      }
18    }
19    @else if ($lang==ru) {
20      [lang="ru"] & {
21        @content;
22      }
23    }
24  }
25 }
26
27 %font-georgia {
28   font-family: Georgia, 'Times New Roman', Serif;
29   font-weight: normal;
30   font-style: italic;
31   -webkit-font-smoothing: antialiased;
32   -moz-osx-font-smoothing: greyscale;
33   @at-root {
34     @include getLang(chinese) {
35       font-family: Georgia, 'Times New Roman', "FangSong", "仿宋", STFangSong, "华文仿宋", "Arial Unicode MS";
36     }
37     @include getLang(jp) {
38       font-family: Georgia, 'Times New Roman', "メイリオ", "MS PGothic", Verdana, Helvetica, sans-serif;
39     }
40     @include getLang(ru) {
41       font-family: Georgia, 'Times New Roman', "Times CY", "Times New Roman", "Arial Unicode MS";
42     }
43   }
44 }
45
46 p {
47   color: orange;
48   @extend %font-georgia;
49 }
50
51
```

ence !

```
/* =====
Author's custom styles
===== */

p {
  font-family: Georgia, 'Times New Roman', Serif;
  font-weight: normal;
  font-style: italic;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: greyscale;
}

[lang="zh-Hans"] p,
[lang="zh-Hant"] p {
  font-family: Georgia, 'Times New Roman', "FangSong", "仿宋", STFangSong, "华文仿宋", "Arial Unicode MS";
}

[lang="ja"] p {
  font-family: Georgia, 'Times New Roman', "メイリオ", "MS PGothic", Verdana, Helvetica, sans-serif;
}

[lang="ru"] p {
  font-family: Georgia, 'Times New Roman', "Times CY", "Times New Roman", "Arial Unicode MS";
}

p {
  color: orange;
}
```



Loops

- En combinaison avec des frameworks (VueJS, pug, etc.) permet de générer rapidement des listes d'éléments
- Chaque élément peut avoir des attributs différents (couleur, vitesse d'animation, etc.)

```
/*Scss*/
$delay-class-slug: delay !default;
@for $i from 1 through 50 {
  $delay: $i / 10;
  .#{ $delay-class-slug }--#{ $i } {
    animation-duration: #{ $delay }s;
  }
}
```

5.1.2. Les framework front-end

Angular

- Framework complet avec beaucoup d'outils out of the box
- Angular 2 utilise TypeScript par défaut

React

- Utilise le “language” **JSX**
- Pas uniquement dédié au Web (React native)
- Permet de créer des composants qui gèrent leur état de façon autonome
- Manipule un “Virtual DOM” et non pas le DOM directement, et patch le DOM en cas de changements
- Pas un framework, mais une library !

VueJS

- Similaires à React, en plus “propre” (pas de langage dédié)
- Sépare chaque fichier vue en 3 parties : template, style, code
- Separation of concerns
- Pas un framework, mais une library !

5.2. Le back-end

API

- Une API => plusieurs applications clients
- RESTful

Notes

- Il est conseillé de bien séparer son code pour une relecture et des modifications plus simples

Tips and tricks

- Les tags headings sont très importants pour le référencement d'un site web
- Il est très facile de voir les sources d'un site web. Toutefois le css et le js sont souvent minifiés
- Si on définit 2 fois des valeurs pour le même selector en CSS, c'est la dernière configuration qui écrase les anciennes (en général le dernier fichier CSS chargé)