

# INITIATION À LA PROGRAMMATION WEB

---

Thomas Boyer

Programmeur bio-informaticien

# PLAN

- Introduction
- HTML/CSS
- JavaScript
- La suite ?

# INTRODUCTION

---

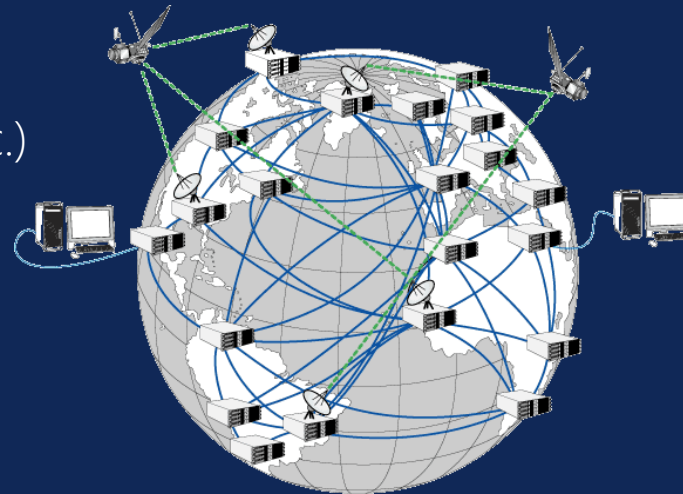
Internet : historique, presentation et preparation à la pratique

# QU'EST-CE QU'INTERNET ?

Un ensemble de réseaux informatiques locaux utilisant les mêmes protocoles de bas niveau standards (TCP/IP) et formant un réseau global.

## Applications d'internet

- Courrier électronique (gmail, yahoo, outlook, etc.)
- Partage de fichiers pair-à-pair (P2P)
- Réseaux sociaux (X, TikTok, etc.)
- ...
- Le « **World Wide Web** »
  - Un espace d'information global et décentralisé, basé sur la navigation par [hypertexte](#)



- Internet = réseau informatique qui permet de transporter les informations (matériel)
- Web = ensemble d'informations (applicatif)

# HISTOIRE DU WEB (EN BREF)

*Histoire partielle et subjective ...*

- [Paul Otlet](#) : Mondotheque / Traité de documentation
- [Vannevar Bush](#) : Memex / « As We May Think »
- [Douglas Engelbart](#) : La souris / Les interfaces graphiques
- [Ted Nelson](#) : Projet Xanadu / Hypertexte
- [Tim Berners-Lee](#) : Le World Wide Web / W3C
  - W3C = WWW consortium : Consortium d'entreprises et d'universités ayant pour mission la standardisation des technologies ouvertes et libres de droits du World Wide Web
- [Steve Jobs](#) : Smartphone / Web mobile

# COMPOSANTS DU WEB

HTTP

URLs

HTML

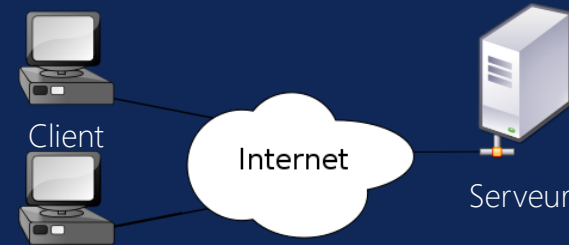
# COMPOSANTS DU WEB

## HTTP

- [HyperText Transfer Protocol](#)
- Décrit comment les données du Web sont échangées entre les machines

**Client :** un ordinateur/smartphone/tablette... utilisée pour afficher des ressources

**Serveur :** un ordinateur « contenant » des ressources, toujours connecté à Internet



**Ressource :** toute unité d'information (document, image, vidéo...) accessible sur le web

# COMPOSANTS DU WEB

## HTTP

- HyperText Transfer Protocol
- Décrit comment les données du Web sont échangées entre les machines



## URLs

- Uniform Resource Locator
- Structure :

https://httpwg.org/specs/rfc9110.html

Protocole

Nom du  
serveur

Nom local de  
la ressource

- N'importe qui peut lier à n'importe quoi
  - NB : bien que le sigle URL ait été remplacé par URI, la plupart des gens parlent encore d'URL



# COMPOSANTS DU WEB

## HTTP

- HyperText Transfer Protocol
- Décrit comment les données du Web sont échangées entre les machines



## URLs

- Uniform Resource Locator
- Structure :  

https://httpwg.org/specs/rfc9110.html  
Protocole    Nom du serveur    Nom local de la ressource
- N'importe qui peut lier à n'importe quoi

## HTML

- HyperText Markup Language
- Décrit comment les données peuvent être interprétées par le client
- Beaucoup de versions depuis 1991, mais toutes largement compatibles

# À QUOI BON ?

Pourquoi apprendre le HTML alors que tout le monde utilise des éditeurs de texte WYSIWYG et/ou de générateurs ?

- Les générateurs HTML ne permettent pas de tout faire
- Le HTML généré a souvent besoin d'être retouché à la main
- Vous pouvez être amenés à *écrire* des générateurs
- Vous pouvez être amenés à écrire des programmes qui *consomment* du HTML (pour le traiter, l'afficher, etc.)
- Le WYSIWYG n'est pas idéal pour HTML car le HTML n'est pas (plus) un langage de présentation, il décrit la *structure logique* des pages.
- Etc.

# VOS OUTILS

- Un éditeur de texte pour éditer vos fichiers HTML :
  - [Notepad/Notepad++](#) : Editeur de code gratuit de référence sur Windows.
  - [Visual Studio Code](#) : Editeur de code gratuit et extrêmement complet fourni par Microsoft.
  - Sublime Text, Atom, Nova, NetBeans, etc.
- Un navigateur web moderne pour les visualiser.
- Une documentation des éléments HTML/CSS:
  - [Devdocs](#) : Site combinant des multiples documentations dans une interface Web claire et organisée, dans laquelle il est possible de rapidement chercher des éléments d'un langage donné.
  - [W3schools](#) : Site créé en 1998, non affilié au W3C (malgré le nom), ayant pour mission de fournir des services gratuits liés à l'apprentissage et l'utilisation de nombreux langages de programmation (des tutoriels permettant d'apprendre les différents langages informatiques, de la documentation sur les différents éléments que proposent ces différents langages, etc.)



**“ EN OMETTANT DE VOUS PRÉPARER,  
VOUS VOUS PRÉPAREZ À L'ÉCHEC. ”**

- Benjamin Franklin

# HTML/CSS

---

Bases de programmation

# HTML : UN LANGAGE À BALISES

En anglais, « langage à balises » se dit « Markup language »

- HTML est format textuel : on peut donc le modifier dans un éditeur de texte
- Le texte non balisé sera affiché tel quel
- Les balises sont des codes qui ne sont pas affichées par le navigateur tel quel, mais qui servent à indiquer le rôle du texte balisé (et donc, indirectement, comment il sera présenté)

*Exemple :*

Qu'est-ce qu'une `<em>balise</em>`?

HTML

Qu'est-ce qu'une *balise*?

Résultat

# HTML : UN LANGAGE À BALISES

## Structure des balises

- Tout texte entre chevrons, '`<`' et '`>`', est considéré comme une balise
- `<em>` est une balise ouvrante, `</em>` est une balise fermante lui correspondant
  - Les deux balises délimitent le texte concerné

```
<p>Emboîtement <em>correct</em> </p>
```

```
<p>Emboîtement <em>incorrect</p> </em>
```

Qu'est-ce qu'une `<em>` balise `</em>`?

Correct

Qu'est-ce qu'une `<em>` balise?

Incorrect

## Emboîtement

- On peut appliquer plusieurs balises au même texte, à condition de respecter la règle d'emboîtement:
  - Toute balise ouverte à l'intérieur d'une autre doit être fermée dans cet autre balise

# HTML : UN LANGAGE À BALISES

## Attributs

- Certaines balises ont besoin d'information supplémentaire, mais qui n'apparaîtra pas dans le document.
- Cette information est donnée par des attributs, qui ont la forme « nom = valeur » et sont placés entre les chevrons de la balise ouvrante, après son nom

*Exemple :*

```
<a href="https://www.google.com/">Google</a>
```

*Affiche uniquement « [Google](https://www.google.com/) »*



# HTML : UN LANGAGE À BALISES

## Balises vides

Certaines balises particulières n'attendent pas de contenu textuel. Ces balises **vides** n'ont pas de balise fermante correspondante :

Cette balise `<Y a='v'>` n'a pas de contenu

## Gestion des espaces

Si vous écrivez :

Gestion des  
espaces.

... vous verrez :

Gestion des espaces.

Pourquoi ?

# HTML : UN LANGAGE À BALISES

- HTML considère tous les caractères d'espacements (espaces, retours à la ligne...) comme des séparations entre mots, et les affiche comme une *simple espace*.
- Cela donne de la souplesse dans la mise en page du *code* HTML (notamment en utilisant l'*indentation*, comme en programmation).
- On va voir dans la suite comment faire afficher des retours à la ligne.

## Séparation fond/forme

Depuis la version 4, HTML vise à décrire la *structure logique* du document, c'est à dire le **fond** et non la **forme**.

La mise en forme (qu'on appelle parfois *structure physique*) est gérée par une **feuille de style** qui sera décrite en CSS.

# HTML : STRUCTURE D'UN DOCUMENT

## Structure globale

Indique la version de HTML

La tête, qui contient les  
métadonnées (ne sont pas  
directement affichées)

Le corps, qui contient le  
contenu à proprement parler  
du document

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <title>Titre du document</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    ...
  </body>
</html>
```

Indique la langue utilisée

Le titre (<title>) est obligatoire

Spécifie l'encodage utilisé par le  
texte (non obligatoire mais  
recommandé)

# HTML : STRUCTURE D'UN DOCUMENT

## Titres et paragraphes

Un document typique est une séquence de titres et de paragraphes.

Les titres ont différents niveaux d'importance, repérables à leur typographie (et parfois à leur numérotation), et donnent une structure *hiérarchique* au document (en parties, sous-parties, *etc.*).

Quels sont les différents niveaux de titre dans l'exemple suivant ?

### Règles du jeu

Extrait des règles du jeu *Colons de Catane* par Klaus Teuber

Les mots ou groupes de mots suivis d'une astérisque (\*) sont expliqués plus en détail dans ce manuel.

#### Déroulement général de la partie

Le joueur le plus âgé commence. Le jouer dont c'est le tour peut effectuer les actions suivantes, **en respectant l'ordre indiqué** :

1. Il *doit* lancer les dés pour savoir où il y a des **revenus en matières premières\*** (ce résultat concerne tous les joueurs).
2. Il *peut* faire du **commerce\*** : il échange des cartes Matière première avec les autres joueurs.
3. Il *peut* effectuer des **constructions\*** : routes\*, colonies\*, ou villes\*, et/ou acheter des cartes de développement\*.

De plus, il peut jouer une (et une seule) carte Développement\* à n'importe quel moment pendant son tour.

Dès qu'un joueur a fini, le joueur situé à sa gauche commence son tour de jeu.

#### Déroulement du jeu en détail

##### 1. Revenus des matières premières

Chaque joueur commence son tour en lançant les deux dés. On fait la somme des dés et le résultat désigne la ou les régions où il y a des matières premières à obtenir !

**Chaque joueur** qui possède une **colonie\*** située au croisement\* d'une région désignée par le dé reçoit une **carte Matière première** correspondant à cette région.

**Important** : Si le résultat des dés désigne la case où se trouve le brigand noir\*, cette case ne produit **aucun revenu de matières premières** ce tour-là. Le brigand reste sur place.

(N.B. Pour des exemples, voir *Revenus des matières premières\** dans le Manuel.)

##### 2. Commerce\*

Le joueur dont c'est le tour peut faire ensuite du commerce, c'est à dire échanger des cartes Matière première. Il existe deux formes de commerce : ...

# HTML : STRUCTURE D'UN DOCUMENT

## Titres et paragraphes en HTML

- Les paragraphes sont délimités par la balise `<p>`
- Les titres sont délimités par les balises `<h1>`, `<h2>`..., `<h6>` ;
  - `<h1>` est le niveau le plus important, et `<h6>` le moins important.

```
<h1>Ceci est un titre</h1>
<p>Ceci est un paragraphe.</p>
<p>Ceci est un deuxième
paragraphe.</p>
```

```
<h1>Ma dissertation</h1>
<h2>Thèse</h2>
  <p>Paragraphe d'introduction</p>
  <h3>Argument 1</h3>
    <p>...</p> <p>...</p> <p>...</p>
  <h3>Argument 2</h3>
    <p>...</p> <p>...</p> <p>...</p>
<h2>Antithèse</h2>
  <h3>Contre-argument 1</h3>
    <p>...</p> <p>...</p> <p>...</p>
  <h3>Contre-argument 2</h3>
    <p>...</p> <p>...</p> <p>...</p>
<h2>Synthèse</h2>
  <p>...</p> <p>...</p> <p>...</p>
```

# HTML : STRUCTURE D'UN DOCUMENT

## Cohérence

L'enchaînement des niveaux de titre doit être cohérent :

- Le premier titre devrait toujours être de niveau 1 (<h1>Titre</h1>)
- Un titre ne devrait pas monter de plus d'un niveau par rapport au précédent

« Mais si le <h2> ne se différencie pas assez du <h1> visuellement, j'ai le droit de mettre un <h3> à la place ? »

→ non; HTML doit être cohérent avec la structure du document. Pour changer la typographie, on modifiera plutôt le CSS.



Ces règles de cohérence ne sont pas *normatives*, leur non-respect n'est *pas* signalé par les valideurs, mais il est tout de même à proscrire.

# HTML : STRUCTURE D'UN DOCUMENT

## Exercice

- 1) Récupérez le texte correspondant à l'exemple d'avant en utilisant les balises <p> et <h1>/<h2>/<h3>.
- 2) Récupérez le modèle HTML correspondant à l'exemple, et recopiez votre code à l'intérieur de la balise <body>. Ce modèle contient un lien vers la feuille de style appropriée.

Vous pouvez aussi essayer de mettre les mots en gras, italique ou souligné si vous le souhaitez :

<em>	emphase	italique
<strong>	emphase forte	gras
<dfn>	définition	italique
<cite>	titre d'ouvrage	italique
<q>	citation	guillemets

### Règles du jeu

Extrait des règles du jeu *Colons de Catane* par Klaus Teuber

Les mots ou groupes de mots suivis d'une astérisque (\*) sont expliqués plus en détail dans ce manuel.

#### Déroulement général de la partie

Le joueur le plus âgé commence. Le joueur dont c'est le tour peut effectuer les actions suivantes, **en respectant l'ordre indiqué** :

1. Il **doit** lancer les dés pour savoir où il y a des **revenus en matières premières\*** (ce résultat concerne tous les joueurs).
2. Il **peut** faire du **commerce\*** : il échange des cartes Matière première avec les autres joueurs.
3. Il **peut** effectuer des **constructions\*** : routes\*, colonies\*, ou villes\*, et/ou acheter des cartes de développement\*.

De plus, il peut jouer une (et une seule) carte Développement\* à n'importe quel moment pendant son tour.

Dès qu'un joueur a fini, le joueur situé à sa gauche commence son tour de jeu.

#### Déroulement du jeu en détail

##### 1. Revenus des matières premières

Chaque joueur commence son tour en lançant les deux dés. On fait la somme des dés et le résultat désigne la ou les régions où il y a des matières premières à obtenir !

Chaque **joueur** qui possède une **colonie\*** située au croisement\* d'une région désignée par le dé reçoit une **carte Matière première** correspondant à cette région.

**Important** : Si le résultat des dés désigne la case où se trouve le brigand noir\*, cette case ne produit **aucun revenu de matières premières** ce tour-là. Le brigand reste sur place.

(N.B. Pour des exemples, voir *Revenus des matières premières\** dans le Manuel.)

##### 2. Commerce\*

Le joueur dont c'est le tour peut faire ensuite du commerce, c'est à dire échanger des cartes Matière première. Il existe deux formes de commerce : ...

Valideur:

Le W3C fournit un service de validation en ligne :

<http://validator.w3.org/>

Son utilisation vous est vivement recommandée.

# HTML : STRUCTURE D'UN DOCUMENT

## Sections

Les titres définissent en fait une structure de plus haut niveau :

- chaque titre indique le début d'une section,
- qui se termine au prochain titre de même niveau ;

Une section contient donc

- un titre,
- éventuellement des paragraphes,
- éventuellement une ou plusieurs section(s) de niveau suivant.

Sections en HTML:

- Jusqu'à HTML 4.01, la structuration en sections était laissée implicite.
- Depuis HTML5, on peut est encouragé à utiliser la balise `<section>`,
  - d'autant que les anciens navigateurs l'ignoreront purement et simplement.



# HTML : STRUCTURE D'UN DOCUMENT

## Sections

```
<h1>Ma dissertation</h1>
<h2>Thèse</h2>
  <p>Paragraphe d'introduction</p>
  <h3>Argument 1</h3>
    <p>...</p> <p>...</p> <p>...</p>
  <h3>Argument 2</h3>
    <p>...</p> <p>...</p> <p>...</p>
<h2>Antithèse</h2>
  <h3>Contre-argument 1</h3>
    <p>...</p> <p>...</p> <p>...</p>
  <h3>Contre-argument 2</h3>
    <p>...</p> <p>...</p> <p>...</p>
<h2>Synthèse</h2>
  <p>...</p> <p>...</p> <p>...</p>
```



```
<h1>Ma dissertation</h1>
<section>
  <h2>Thèse</h2>
  <p>Paragraphe d'introduction</p>
  <section>
    <h3>Argument 1</h3>
    <p>...</p> <p>...</p> <p>...</p>
  </section>
  <section>
    <h3>Argument 2</h3>
    <p>...</p> <p>...</p> <p>...</p>
  </section>
</section>
<section>
  <h2>Antithèse</h2>
  <section>
    <h3>Contre-argument 1</h3>
    <p>...</p> <p>...</p> <p>...</p>
  </section>
  <section>
    <h3>Contre-argument 2</h3>
    <p>...</p> <p>...</p> <p>...</p>
  </section>
</section>
<section>
  <h2>Synthèse</h2>
  <p>...</p> <p>...</p> <p>...</p>
</section>
```

# HTML : STRUCTURE D'UN DOCUMENT

## Liste

Une liste est un paragraphe d'un type particulier, contenant une énumération d'éléments.

Liste non-ordonnée :

```
<ul>
  <li>sucre</li>
  <li>céréales</li>
  <li>lait</li>
</ul>
```



- Sucre
- Céréales
- lait

Liste ordonnée :

```
<ol>
  <li>sucre</li>
  <li>céréales</li>
  <li>lait</li>
</ol>
```



1. Sucre
2. Céréales
3. lait

Dans votre texte, pouvez-vous identifier une liste ?  
Reprenez votre code HTML reproduisant cet exemple  
et modifiez-le en conséquence.

# HTML : STRUCTURE D'UN DOCUMENT

## Liens

- La balise `<a>` transforme son contenu en lien hypertexte
- Elle doit contenir un attribut href, qui contient l'URL (absolue ou relative) vers lequel dirige ce lien.

Bienvenue sur `<a href='https://www.google.fr/'>Google</a>`

Bienvenue sur [Google](https://www.google.fr/)

# HTML : STRUCTURE D'UN DOCUMENT

## Liens

- La balise `<a>` transforme son contenu en lien hypertexte
- Elle doit contenir un attribut `href`, qui contient l'URL (absolue ou relative) vers lequel dirige ce lien.
- Rien n'empêche la balise `<a>` de contenir d'autres balises :

Bienvenue sur `<a href='https://www.google.fr/'>Google<img src='google.jpg'/'> </a>`

Bienvenue sur [Google](https://www.google.fr/)



# HTML : STRUCTURE D'UN DOCUMENT

## Liens interne

- Il peut être utile de faire pointer un lien vers un fragment particulier d'un document plutôt que vers son intégralité
- En HTML, toute balise délimite une partie du document. On peut donner un nom à ce fragment en ajoutant à la balise l'attribut 'id'
- L'URL du fragment est constituée en concaténant :
  - l'URL du document,
  - le caractère « # », et
  - l'identifiant du fragment.
- NB : une URL relative commençant par « # » désigne donc un fragment du document courant.

```
<a href = "#sec1">
  pointe vers le fragment #sec1 du
  document courant</a>

<a href = "other.html#sec2">
  pointe vers le fragment #sec2 d'un
  autre document</a>

<a href =
  "http://en.wikipedia.org/wiki/Lyon#Det
  ails">
  pointe vers la section "Détails" d'un
  article Wikipedia</a>

<section id = "sec1">
  La section vers laquelle pointe le
  1er lien ci-dessus.
...</section>
```

# HTML : STRUCTURE D'UN DOCUMENT

## Exercice

Je vous propose de travailler sur le fichier « sujet.html ». Il contient un livre dont vous êtes le héros minimaliste.

- 1) Faites-en une version interactive avec des liens internes.
- 2) Faites-en une version interactive où chaque entrée est une page HTML distincte (en utilisant [ces fichiers HTML](#)).

# HTML/CSS

---

Bases de programmation

# CSS : FEUILLE DE STYLE

## C'est quoi ?

- CSS : Cascading StyleSheet
- HTML décrit la structure logique des documents (le fond) tandis que la structure physique (la forme) est spécifiée par une feuille de style en CSS.

## La feuille de style CSS :

C'est un fichier texte écrit en .css qui permet de définir l'apparence de chaque élément sélectionné

```
em {  
    font-family: "Gill Sans Extrabold", sans-serif;  
}
```



Les éléments `<em></em>` sont alors écrits dans la police « Gil Sans Extrabold », sans-serif (pas d'empattement)



# CSS : FEUILLE DE STYLE

## Comment lier une feuille de style à un fichier HTML ?

Pour attacher une feuille de style à un document HTML, on ajoute dans l'élément `<head>` une balise `<link>` :

```
<head>  
  <link rel="stylesheet" type="text/css"  
        href="mystyle.css" />  
</head>
```

- `rel` : indique la relation entre le document et la ressource liée (toujours « `stylesheet` »)
- `type` : type de contenu auquel le lien fait référence (toujours `'text/css'`)
- `href` : url de la ressource (comme vu auparavant, chemin absolu ou relatif)

On peut avoir plusieurs feuilles de style (les effets se cumulent)

# CSS : FEUILLE DE STYLE

## Comment lier une feuille de style à un fichier HTML ?

On peut également spécifier la feuille de style directement dans le document HTML, à l'aide de la balise `<style>` ;

```
<head>  
  <style type="text/css">  
    /* your CSS here */  
  </style>  
</head>
```

# CSS : FEUILLE DE STYLE

## Pourquoi utiliser une feuille de style ?

- Cohérence au sein du document
- Cohérence au sein d'un ensemble de documents (charte graphique)
  - mutualisation de la feuille de style
- Séparation des tâches (développeur web / graphiste)

## Outils

Comme avant :

- Valideur : <http://jigsaw.w3.org/css-validator/>
- Inspecteur : « Clic droit » > « Inspecter »
- Liens utiles :
  - <https://www.w3schools.com/css/>
  - <https://devdocs.io/css/>

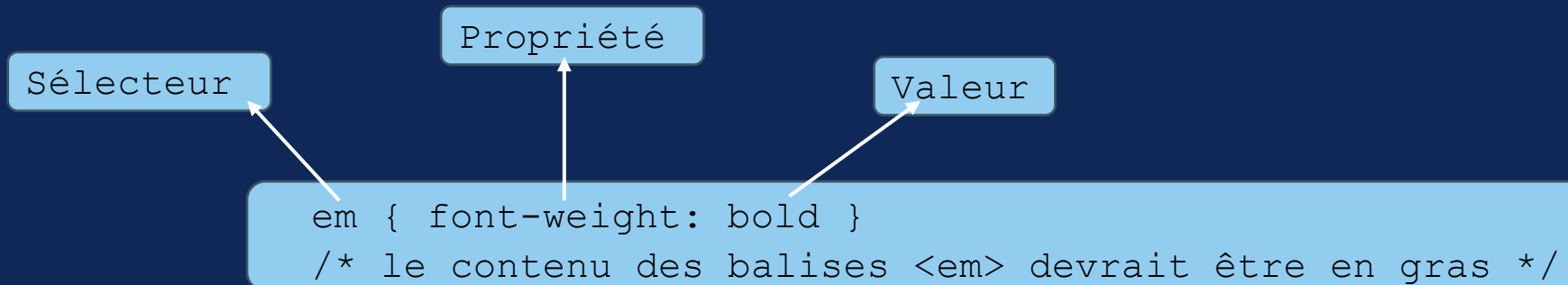
# CSS : DÉFINITION PAR RÈGLES

## Principe

En CSS, la mise en forme est spécifiée par un ensemble de règles.

Une règle typique est composée de trois parties :

- Un sélecteur
- Une propriété
- Une valeur



# CSS : DÉFINITION PAR RÈGLES

## Combinaison de règles

Plusieurs règles similaires peuvent coexister :

```
em    { font-weight: bold }  
em    { color: blue }  
cite  { font-weight: bold }  
cite  { color: blue }
```

## Regroupement par sélecteur

On peut regrouper des règles ayant le même contenu, en séparant les sélecteurs par une virgule ( « , » ) :

```
em    { font-weight: bold ; color: blue }  
cite  { font-weight: bold ; color: blue }
```

# CSS : DÉFINITION PAR RÈGLES

## Regroupement par contenu

On peut regrouper des règles ayant le même contenu, en séparant les sélecteurs par une virgule (« , »).

```
em, cite { font-weight: bold ; color: blue }
```

## Sensibilité aux espaces

```
em,  
cite  
{  
    font-weight : bold ;  
    color       : blue  ;  
}
```

On peut regrouper des règles ayant le même contenu, en séparant les sélecteurs par une virgule (« , »).

# CSS : PROPRIÉTÉS DU TEXTE

## Propriétés sur la police

- « Font-style » : Italique ou normal : « normal » ou « *italic* »
- « Font-weight » : Grosseur du trait : « normal », « **bold** », « **bolder** », « *lighter* »
  - ou une valeur entre 100 et 900 (400 = normal)
  - NB : beaucoup de polices ne supportent que normal et bold
- « Font-variant » : normal ou majuscule : « normal », « SMALL-CAPS »
- « Font-family » : police à utiliser : voir après
- « Font-size » : taille de la police : voir après

# CSS : PROPRIÉTÉS DU TEXTE

## Police avec « font-family »

Lorsqu'on publie sur le Web, on ne peut pas faire l'hypothèse que tous les clients auront les mêmes polices installées sur leur système.

Si on veut utiliser une police spécifique, il faut donc indiquer au navigateur comment la charger :

```
@font-face
{
  font-family: MyNiftyFont;
  src: url('http://example.org/nifty-font.ttf'),
       url('http://example.org/nifty-font.eot');
}
```



# CSS : PROPRIÉTÉS DU TEXTE

## Police avec « font-family » : Polices génériques

CSS définit des polices génériques :

- Inconvénient : leur apparence varie d'un système à l'autre (et d'un navigateur à l'autre).
- Avantage : elles garantissent un affichage *pas trop différent* des intentions de l'auteur.

```
@font-face { font-family: MyNiftyFont; src: /*...*/ }  
  
em {  
  font-family: MyNiftyFont, "Times New Roman", serif;  
}
```

Bonne pratique :

- Spécifier une liste de polices,
- par ordre croissant de probabilité qu'elle soit disponible,
- et en terminant toujours pas une police générique.

# CSS : PROPRIÉTÉS DU TEXTE

## Taille du texte avec font-size

La taille est généralement exprimée en points :

```
body { font-size: 12pt ; }
```

ou relativement à la taille de la police dans l'élément « général »:

```
strong { font-size: 150% ; }
```

ou relativement à la taille de la police de l'élément racine (<html>):

```
h1 { font-size: 2rem ; }
```

# CSS : PROPRIÉTÉS DU TEXTE

## Autres propriétés

- « text-decoration » : none, underline, overline, ~~line-through~~
- « text-transform » : none (Par exemple), capitalize (Par Exemple), uppercase (PAR EXEMPLE), lowercase (par exemple)
- « text-align » : left, center, right, justify (voir après)
- « color » : (voir après)
- « background-color » : (voir ci-après)

# CSS : PROPRIÉTÉS DU TEXTE

## Alignement du texte

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

« text-align » : left

« text-align » : center

« text-align » : right

« text-align » : justify

# CSS : COULEURS ET SÉLECTEURS

## Couleurs en CSS : RGB

RGB : format de couleur supporté par tous les navigateurs

Le RGB (Red Green Blue) correspond au mélange des trois couleurs rouge, vert et bleu avec des intensités définies pour chacune des trois couleurs primaires (optique).

RGB (Rouge, Vert, Bleu) avec rouge, vert et bleu étant un nombre compris dans l'intervalle [0, 255] correspondant à l'intensité de la couleur

- On peut aussi ajouter un paramètre alpha pour la transparence de la couleur



```
em { color: #f00 }  
em { color: #ff0000 }  
em { color: rgb(255,0,0) }  
em { color: rgb(100%, 0%, 0%) }
```

[https://www.w3schools.com/colors/colors\\_rgb.asp](https://www.w3schools.com/colors/colors_rgb.asp)

# CSS : COULEURS ET SÉLECTEURS

## Couleurs en CSS : autres moyens

- couleurs prédéfinies: « black », « white », « red », « green », « blue », « yellow »...
  - mais aussi « transparent »
  - la liste est très longue
- transparence (alpha): rgba(r,g,b,a) où a varie entre 0.0 (invisible) and 1.0 (opaque)
- pour en savoir plus : <http://www.w3.org/TR/css3-color/>

# CSS : COULEURS ET SÉLECTEURS

## Sélecteurs complexes

On a souvent besoin d'appliquer des règles de présentations différentes à la même balise en fonction de son contexte.

Ceci peut s'exprimer en combinant plusieurs sélecteurs :

```
X Y { /* s'applique à tout élément Y situé  
      à l'intérieur d'un X — même indirectement */ }
```

```
X > Y { /* s'applique à tout élément Y situé  
        directement à l'intérieur d'un X */ }
```

```
X + Y { /* s'applique à tout élément Y situé  
        immédiatement après un X */ }
```

# CSS : COULEURS ET SÉLECTEURS

## Sélecteurs complexes

Les sélecteurs complexes peuvent bien sûr être combinés à leur tour.

On peut également utiliser dans les combinaisons le sélecteur \*, qui est satisfait par n'importe quelle balise.

```
q      { font-style: italic; }  
q em   { font-weight: bold; }  
q strong { text-decoration: underline; }  
  
body>h1 { text-align: center; }  
  
h1+* { font-variant: small-caps; }  
  
ul ul li { font-size: 80%; }
```



# CSS : EXERCICES

## Exercices

### CSS Diner (jusqu'au niveau 14)

Reprenez le code HTML que vous avez écrit pour l'exercice sur les règles du jeu, et attachez-le à une nouvelle feuille de style.

Sans toucher au code HTML, ajoutez à votre feuille de style les règles qui permettront d'obtenir le résultat suivant:

### Règles Du Jeu

Extrait des règles du jeu *Colons de Catane* par Klaus Teuber

Les mots ou groupes de mots suivis d'une astérisque (\*) sont expliqués plus en détail dans ce manuel.

#### Déroulement Général De La Partie

Le joueur le plus âgé commence. Le joueur dont c'est le tour peut effectuer les actions suivantes, en respectant l'ordre indiqué :

1. Il doit lancer les dés pour savoir où il y a des revenus en matières premières\* (ce résultat concerne tous les joueurs).
2. Il peut faire du commerce\* : il échange des cartes Matière première avec les autres joueurs.
3. Il peut effectuer des constructions\* : routes\*, colonies\*, ou villes\*, et/ou acheter des cartes de développement\*.

De plus, il peut jouer une (et une seule) carte Développement\* à n'importe quel moment pendant son tour.

Dès qu'un joueur a fini, le joueur situé à sa gauche commence son tour de jeu.

#### Déroulement Du Jeu En Détail

##### REVENUS DES MATIÈRES PREMIÈRES

Chaque joueur commence son tour en lançant les deux dés. On fait la somme des dés et le résultat désigne la ou les régions où il y a des matières premières à obtenir !

Chaque joueur qui possède une colonie\* située au croisement\* d'une région désignée par le dé reçoit une carte Matière première correspondant à cette région.

Important : Si le résultat des dés désigne la case où se trouve le brigand noir\*, cette case ne produit aucun revenu de matières premières ce tour-là. Le brigand reste sur place.

(N.B. Pour des exemples, voir *Revenus des matières premières\** dans le Manuel.)

##### COMMERCE\*

Le joueur dont c'est le tour peut faire ensuite du commerce, c'est à dire échanger des cartes Matière première. Il existe deux formes de commerce : ...

# CSS : CLASSES ET IDENTIFIANTS

HTML autorise les attributs suivant dans n'importe quelle balise :

- « id » accepte comme valeur un nom unique (il est interdit d'utiliser le même « id » à deux endroits du même document)
  - Pourquoi ? Un identifiant identifie de façon unique un élément (carte d'identité, couple login/mdp pour un site, etc.)
- « class » accepte comme valeur une liste de noms séparés par des espaces (le même nom de classe peut être présent dans plusieurs balises).

```
<ol id="contents">...</ol>
```

```
<article class="post funny">...</article>
```

# CSS : CLASSES ET IDENTIFIANTS

## Sélecteurs associés

CSS permet de sélectionner un élément par son identifiant ou sa classe, en spécifiant ou non le type de la balise.

```
article.post { /* tout <article> de la classe 'post' */ }  
  
.funny      { /* tout élément de la classe 'funny' */ }  
  
ol#contents { /* toute <ol> avec l'id 'contents' */ }  
  
#contents   { /* tout élément avec l'id 'contents' */ }
```

# CSS : CLASSES ET IDENTIFIANTS

## Bonne utilisation des identifiants et des classes

### Identifiants

Un identifiant est unique au sein d'un document,

- mais il peut se répéter d'un document à l'autre
- par exemple, l'identifiant « contents » est peut-être utilisé pour identifier la table des matières sur toutes les pages d'un site.

Il y a donc un intérêt à mutualiser les règles de présentation pour un identifiant dans une feuille de style globale.

### Classes

Les classes permettent de définir des catégories sémantiques plus précises que celles fournies par HTML ; soit

- un cas particulier par rapport à une balise existante (e.g. post),
- une catégorie transverse (e.g. funny).

Le nom de la classe doit décrire ce que signifie la classe, et non la mise en forme qui lui sera appliquée :

- éviter par exemple rouge-souligné ou centré-16pt,
- qui deviendront obsolètes dès que votre charte graphique changera.

# CSS : CLASSES ET IDENTIFIANTS

## Priorité de règles

### CSS

```
em { font-style: italic;  
      color: red }
```

```
.summary em { font-style: normal;  
               font-weight: bold }
```

### HTML

```
<p class="summary">This summary  
is <em>short</em>.</p>
```

Quelle serait la mise en forme du HTML suivant ?

# CSS : CLASSES ET IDENTIFIANTS

## Priorité de règles

```
em { font-style: italic;  
      color: red }
```

```
.summary em { font-style: normal;  
               font-weight: bold }
```

```
<p class="summary">This summary  
is <em>short</em>.</p>
```

Quelle serait la mise en forme du HTML suivant ?



This summary is **short.**

- La règle la plus spécifique a toujours la priorité.
- En cas de spécificité égale, c'est la dernière règle (dans l'ordre du/des fichier.s) qui s'applique.
- Chaque attribut CSS est traité séparément (color dans l'exemple ci-dessus).

# CSS : CLASSES ET IDENTIFIANTS

## Entrainement

- [CSS Diner](#) (à partir du niveau 15)
- [Coloriage magique 1](#)
- [Coloriage magique 2](#)

Attention !

Les coloriages magiques utilisent des sélecteurs non vus en cours.

Pour une liste exhaustive, consultez <http://www.w3.org/TR/css3-selectors/>.

# JAVASCRIPT

---

Presentation et pratique



# JAVASCRIPT : SYNTAXE ET LOGIQUE

## C'est quoi ?

Langage de programmation qui complète HTML + CSS :

- Langage de programmation *généraliste*.
- Offre un spectre beaucoup plus large d'interactions.
- Syntaxe inspirée de celle de Java MAIS la similitude s'arrête là : Javascript n'est pas basé sur Java.

```
if (i < 10) {  
  j = j+1;  
  k += i;  
} else {  
  j = 0;  
}
```

**Condition**

```
for(i=0; i<10;  
i+=1) {  
  j = j*i;  
}
```

```
while (i < 10) {  
  j = j*i;  
  i += 1;  
}
```

**Itération/Boucle**

```
try {  
  i = riskyFunction();  
}  
catch (err) {  
  i = -1;  
}
```

**Exception**

# JAVASCRIPT : SYNTAXE ET LOGIQUE

## Comment ça fonctionne ?

Suite de déclarations (statements), qui se finissent par un « ; » et qui sont interprétés par l'ordinateur pour effectuer la tâche demandée :

```
console.log("Hello World!");  
console.log("I'm learning JavaScript");
```

On peut donc écrire plusieurs déclarations les unes après les autres sans retour à la ligne. Niveau lisibilité cependant ...

```
console.log("Hello World!"); console.log("I'm learning JavaScript");
```

↳ Ici, on appelle la fonction « log » de l'objet « console » pour afficher dans la console les mots compris dans les parenthèses.

↳ Pour l'instant, retenez juste que `console.log()` permet d'afficher des trucs dans la console.

# JAVASCRIPT : SYNTAXE ET LOGIQUE

## Commentaires

Les commentaires sont des lignes qui ne sont pas interprétés par la machine : elle ont donc pour but de documenter le code :

```
// This is hard to read  
console.log("Hello World!"); console.log("I'm  
learning JavaScript");
```

```
// Now it's better  
console.log("Hello World!");  
console.log("I'm learning JavaScript");
```

Flux d'exécution :

- De haut en bas



```
Hello World!  
I'm learning JavaScript  
Hello World!  
I'm learning JavaScript
```

# JAVASCRIPT : SYNTAXE ET LOGIQUE

## On essaye ?

Essayez de dire à la console d'afficher votre nom ainsi que votre âge dans la console. Rajoutez un commentaire pour vérifier que la console ne vous l'affiche pas.

Pour ça, je vous invite à aller sur le site suivant :

<https://jsbin.com/fosuzu/1/edit?js,console>

Comme pour les fois précédentes, je vous propose de vous aider de la documentation ainsi que de l'outil de vérification de code suivant :

<https://jshint.com/>

# JAVASCRIPT : SYNTAXE ET LOGIQUE

## Définition de variable

En programmation, une variable est un nom auquel on rattache une valeur pour l'utiliser plus tard :

```
message = "Hello World!"  
print(message)  
let message = "Nice weather!"  
print(message)
```



```
Hello World!  
Nice weather!
```

Pratique quand on veut sauvegarder un objet, le transformer, l'utiliser plusieurs fois dans un même programme, etc.

### Proposition :

Essayez d'afficher votre nom et votre age dans la console, mais cette fois ci en appelant des variables définies auparavant.

# JAVASCRIPT : TYPES DE DONNÉES

## Types de données

Un type de données spécifie les opérations que l'on peut faire avec cette donnée.

Javascript possède de nombreux types de données, donc les plus utilisés sont :

- String (ou chaîne de caractères) : texte alphanumérique
  - « Bonjour », « Blabla »
- Number : nombre compris entre -9007199254740991 et 9007199254740991
  - 50, 865, -501
- Boolean : Vrai ou Faux (true/false) → utilisé pour des tests logiques
  - true
- Null : Pas de valeur/vide
- Undefined : type donné aux variables auxquelles aucune valeur n'est rattachée

# JAVASCRIPT : TYPES DE DONNÉES

## Types de données

Exemple:

On est tous d'accord pour dire que  $2 + 2 = 4$  ?

Essayez de faire l'opération dans <http://jsbin.com/fosuzu/1/edit?js,console> , puis essayez de faire la même opération mais sous la forme  $2 + \text{« 2 »}$

Pourquoi on a ce résultat ? On a quoi comme type en sortie ? Vous pouvez utiliser la fonction `typeof()` pour déterminer le type d'un objet sous javascript.

Ici, on voit bien que le résultat d'une seule opération n'est pas le même en fonction des types de données.

# JAVASCRIPT : TYPES DE DONNÉES

## Le type « String » en Javascript

Les « string » sont du texte/des chaînes de caractères : vous en avez vu un exemple tout à l'heure avec la fonction `console.log()`

Une chaîne de caractère doit être comprise entre deux guillemets : « bonjour ».

Vous pouvez concaténer plusieurs strings à l'aide du signe `+` :

```
let message = "Hello " + "and " + "Goodbye!";  
console.log(message);
```



Hello and Goodbye!



# JAVASCRIPT : TYPES DE DONNÉES

## Le type « Number » en Javascript

Type de données utilisé pour faire de l'arithmétique, c'est-à-dire les opérations usuelles que vous connaissez sur les nombres :

- *Addition* : Signe +  
`console.log(4+2);` → 6
- *Soustraction* : Signe -  
`console.log(4-2);` → 2
- *Multiplication* : Signe \*  
`console.log(4*2);` → 8
- *Division* : Signe /  
`console.log(4/2);` → 2

Pour votre information :

- Les décimaux (nombres à virgule : 3,54) sont appelés des floats
- Les entiers (nombres sans virgule : 5) sont appelés des integers
  - Javascript se charge de faire la distinction entre les deux

# JAVASCRIPT : TYPES DE DONNÉES

## Le type « Boolean » en Javascript

Type pouvant prendre uniquement deux valeurs : true (Vrai) et false (Faux). Type utilisé pour des tests logiques.

Tout ce qui peut prendre deux états peut se représenté par un booléen :

- Lumière : Allumé ? Eteinte ?
- Mot de passe : Correct ? Mauvais ?
- Est-ce que la couleur présente dans ce cercle est le vert ? Vrai ? Faux ?

```
let on = true;  
let off = false;
```

Pas très parlant dans l'immédiat, mais très utile en pratique

# JAVASCRIPT : TYPES DE DONNÉES

## Le type « undefined » en Javascript

Valeur donnée à un variable sans valeur assignée :

```
let first_name;  
console.log(first_name);
```



undefined

## Le type « null » en Javascript

Type de donnée représentant le manque de valeur/le vide :

```
let first_name = null;
```

### Différence entre null et undefined ?

Les deux représentent du vide MAIS l'un représente une valeur par défaut pour une variable, l'autre représente un vide 'intentionnel'

- Utile dans la définition et l'utilisation de fonctions : on voit ça après

# JAVASCRIPT : TYPES DE DONNÉES

## Opérations de conversion de type

Admettons, on veut faire l'opération  $2 + 2$  MAIS ....

```
let x = "7";  
let y = 5;  
  
console.log(x + y);
```



"75"

Pour ça, il existe des fonctions qui permettent de changer le type de données d'un objet. Il en existe plusieurs en fonction de la transformation que l'on veut faire :

- Number()
- String()
- Boolean()

*En utilisant ces fonctions, comment est ce que je règle le problème au dessus ?*

# JAVASCRIPT : OPÉRATIONS

## Affectation de valeur à une variable

Vous vous souvenez que pour déclarer une variable on utilise « `x = valeur` » ?

Il existe des signes d'affectations permettant de rapidement changer la valeur de `x` :

Nom	Exemple	Signification
Affectation	<code>x = y</code>	<code>x = y</code>
Affectation addition	<code>x += y</code>	<code>x = x + y</code>
Affectation soustraction	<code>x -= y</code>	<code>x = x - y</code>
Affectation multiplication	<code>x *= y</code>	<code>x = x * y</code>
Affectation division	<code>x /= y</code>	<code>x = x / y</code>
Affectation reste	<code>x %= y</code>	<code>x = x % y</code>

```
x = 5  
y = 2
```

```
console.log(x += y);  
console.log(x -= y);  
console.log(x *= y);  
console.log(x /= y);  
console.log(x %= y);
```



```
7  
3  
10  
2,5  
1
```

# JAVASCRIPT : OPÉRATIONS

## Opérations de comparaison

Nom	Exemple
Egalité	<code>x == y</code>
Non égalité	<code>x != y</code>
Stricte égalité	<code>x === y</code>
Stricte non égalité	<code>x !== y</code>
Plus grand que	<code>x &gt; y</code>
Plus grand que ou égal à	<code>x &gt;= y</code>
Moins que	<code>x &lt; y</code>
Moins que ou égal à	<code>x &lt;= y</code>

Stricte : vérifie que le type de sortie est le même que celui des entrées

```
console.log(9 == 9);  
console.log(9 != 20);  
console.log(2 > 10);  
console.log(2 < 10);  
console.log(5 >= 10);  
console.log(10 <= 10);
```



```
true  
true  
false  
true  
false  
true
```

# JAVASCRIPT : OPÉRATIONS

## Opérateurs logiques

- Comment vérifier qu'une chaîne de caractères contient les mots « Bonjour » ET « je » ?
- Comment vérifier que cette même chaîne de caractère contient OU le mot « Bonjour », OU le mot « pouet » ?
- Comment changer un booléen sans réaffecter une valeur à la variable ?

Nom	Exemple	Signification
ET logique	<code>x &amp;&amp; y</code>	Retourne true si les deux tests sont vrais
OU logique	<code>x    y</code>	Retourne true si au moins l'un des deux est vrai
NON logique	<code>!x</code>	Inverse le résultat

```
console.log(7 > 2 && 5 > 4);  
console.log(1 > 2 || 5 > 4);  
console.log(!true);
```

# JAVASCRIPT : OPÉRATIONS

## Exercices

Essayez de faire les opérations suivantes :

- Affectation les valeurs suivantes à des variables : 2, 5, « Bonjour », votre nom
- Affichez dans la console le message suivant en utilisant les deux dernières variables :
  - « Bonjour, je m'appelle X » avec X étant votre nom.
- Additionnez, soustrayez, multipliez et divisez les deux variables « number »
  - Faites la même chose en utilisant des affectations de valeur
- Convertissez une variable « number » en variable de type « string »
  - Utilisez la fonction `typeof()` pour vérifier que le résultat est bon
- 2 est plus grand que 5 ? Comment le vérifier ?
- Utilisez la fonction `includes()` pour vérifier que « Bonjour » contient la lettre « o »
  - Faites la même chose, mais vérifiez que « Bonjour » contient les lettres « o » ET « p »
  - De même, pour tester que « Bonjour » contient soit la lettre « o », soit la lettre « i »



# JAVASCRIPT : UTILISATION EN WEB

## Alors ok, c'est bien joli, mais quel intérêt ?

Comme dit en introduction, dynamisation du site et proposition de fonctions impossibles à proposer avec du HTML/CSS pur :

[https://www.w3schools.com/js/js\\_examples.asp](https://www.w3schools.com/js/js_examples.asp)

# LA SUITE ?

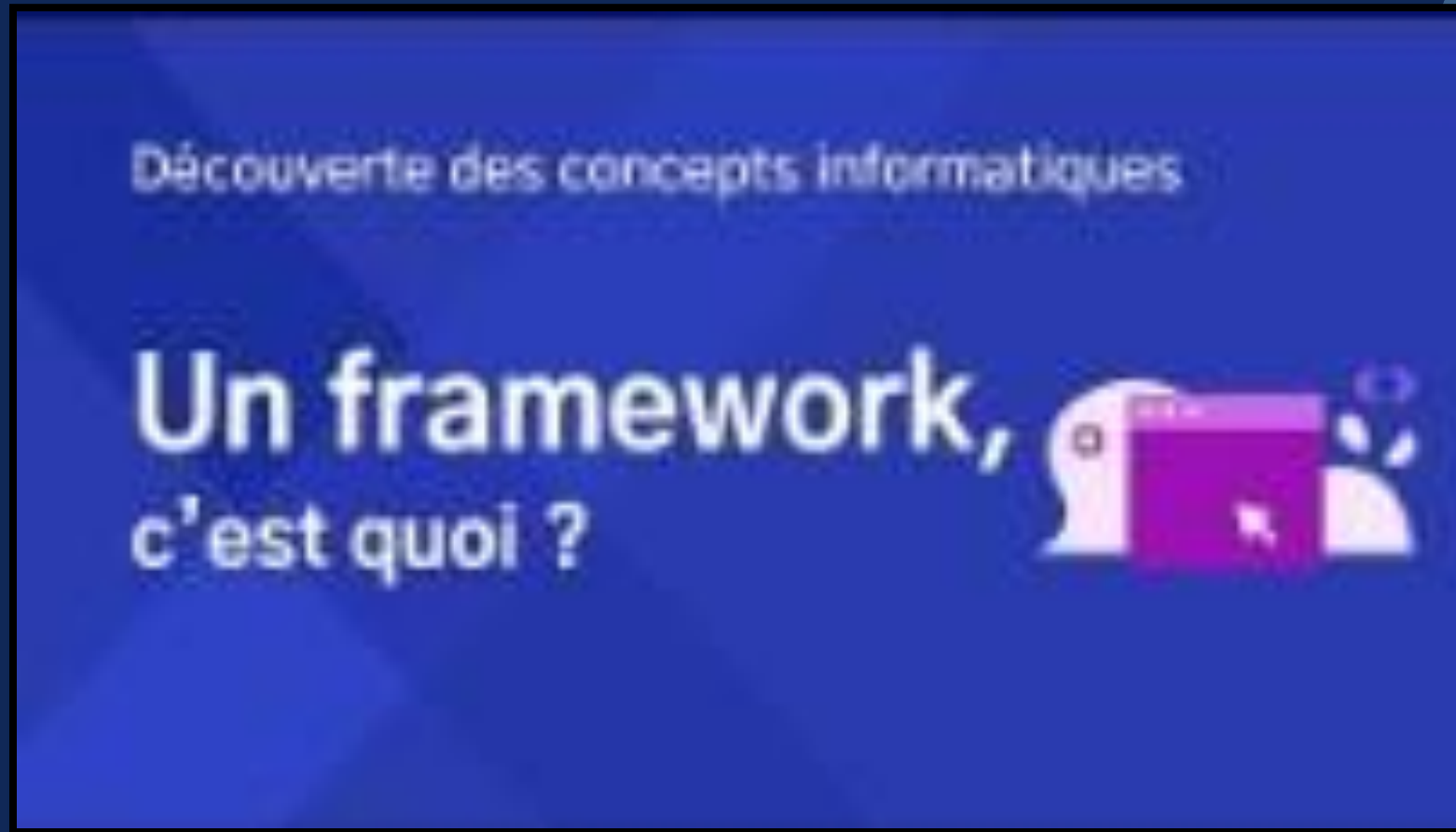
---

Déploiement de site web, Frameworks, etc.

# METTRE SON SITE SUR LE WEB



# FRAMEWORKS : C'EST QUOI ?



# FRAMEWORKS : POURQUOI ?

CGI

Développeur(se) Python Web F/H

Avignon, Provence-Alpes-Côte d'Azur, France

1 relation travaille ici

Promu(e)

es

Développeur Python / Expert Dask - F/H

CS GROUP

Toulouse, Occitanie, France

14 anciens élèves travaillent ici

Promu(e)

Développeur web

Smartfire

Lyon, Auvergne-Rhône-Alpes, France (Hybride)

Recrutement actif

Promu(e) · Candidature simplifiée

Développeur web

FarmLEAP

Vaulx-Milieu, Auvergne-Rhône-Alpes, France (Hybride)

Votre profil correspond à cette offre

Promu(e) · Candidature simplifiée

Software Developer

INTERPOL

Lyon, Auvergne-Rhône-Alpes, France (Sur site)

10 anciens élèves travaillent ici

Il y a 1 mois · 5 candidats

Full stack software developer Python/Angular

Evotec

Toulouse, Occitanie, France

1 ancien collègue travaille ici

Promu(e)

Développeur.se Python Senior

elmy

Lyon, Auvergne-Rhône-Alpes, France (Hybride)

4 anciens élèves travaillent ici

Promu(e)

AT&T

Développeur Full Stack (Python / Javascript) H/F

Ateme

Rennes, Bretagne, France (Hybride)

Promu(e)

AURIS

Développeur Full Stack

AURIS Gestion

Hoerdt, Grand Est, France (Sur site)

Recrutement actif

Promu(e) · Candidature simplifiée

Tu rejoins la brigade en charge du robot de trading automatique dédié au marché Intraday, composée de 2 développeurs.se et intégrée aux équipes métiers.

**Trading algorithmique Intraday**

Dans le cadre de notre activité, nous avons besoin d'équilibrer à chaque instant nos volumes d'approvisionnement et de vente d'énergie renouvelable. Pour cela, nous avons développé un robot de trading "Intraday". Il suit les évolutions des fondamentaux du marché afin de saisir automatiquement et à bon escient les opportunités offertes par le marché vis à vis de nos besoins d'équilibrage. Actuellement actif sur la période de présence des équipes en charge de son exploitation, l'un des enjeux majeurs d'évolution du robot, dans les prochains mois / prochaines années, est de mettre en place tous les garde-fous nécessaires à son activation sur du 24/7.

**Tes missions**

En intégrant l'une de nos brigades, tu participes pleinement à la construction du produit:

- Tu contribues activement à la conception et au développement des différentes fonctionnalités :
- Tu es responsable de la mise en production, l'exploitation et de la maintenance de ces fonctionnalités :
- Tu participes à la définition de l'architecture de nos applications :
- Tu as voix au chapitre sur la priorisation des chantiers de l'équipe :
- Tu es garante des bonnes pratiques de développement, notamment au niveau de la qualité et de la sécurité.

**Notre stack technique**

- Devops: Kubernetes, Docker, GCP...
- Languages: Node/TypeScript, Python...
- Front: React/TypeScript
- Test: pytest, TDD. Orientation forte sur la qualité...
- Data: PostgreSQL, Google Cloud Pub/Sub...
- CI/CD: Github, Jenkins, SonarQube, ...
- Monitoring: Datadog, Grafana, OpenTelemetry, ...

**Ce qui t'attend si tu nous rejoins**

Ce poste est ouvert dans nos locaux lyonnais avec du télétravail partiel possible. Nous proposons sur ce poste une rémunération annuelle à partir de 49.000 euros annuels en fonction de ton expérience + un variable de 10% indexé sur objectifs.

**semaine de 4 jours**

- un travail qui a du sens : avoir un impact positif sur l'environnement en œuvrant pour la transition énergétique
- de la flexibilité sur les horaires et le télétravail
- un pack sport de 200 euros par an
- un abonnement Deezer
- une mutuelle Alan prise en charge à 100% pour toi et ta famille
- une organisation horizontale inspirée de l'holacratie favorisant l'autonomie et la créativité
- un quotidien avec une équipe conviviale, bienveillante et optimiste
- des locaux où il fait bon vivre : cours de yoga, paniers de fruits, coin piano et salle de

Dans les faits, toutes les entreprises utilisent au moins un framework.

React JS



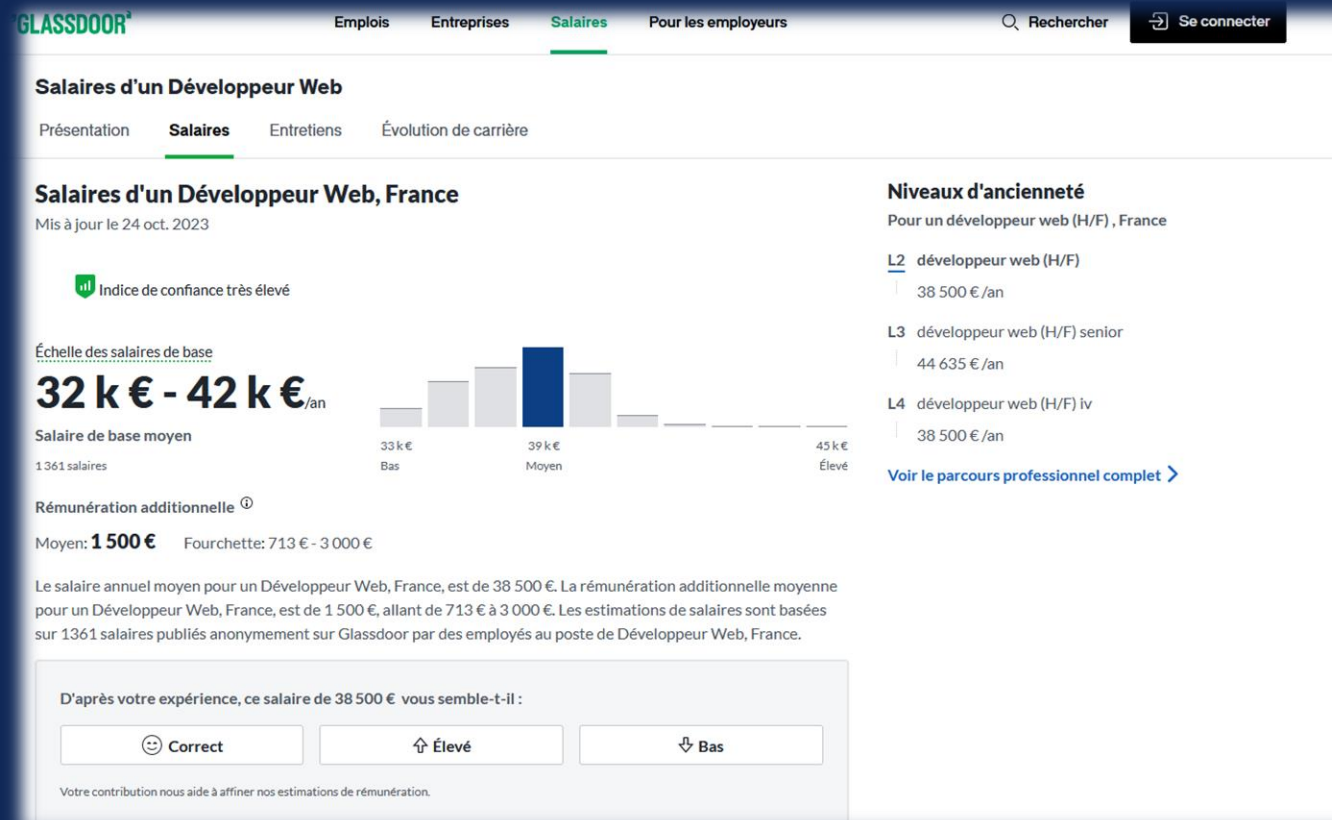
Symfony

Vue.js



# LE NERF DE LA GUERRE : \$\$\$

En date du mois d'octobre 2023, les chiffres sont les suivants :



Site : <https://www.glassdoor.fr/index.htm>



# LE NERF DE LA GUERRE : \$\$\$

En date du mois d'octobre 2023, les chiffres sont les suivants :

Chiffres à prendre avec des pincettes :








- Combien d'années d'expérience ?
- Quelle type d'entreprise ?
- Quelle école ?
- Négociation du salaire lors de l'entretien
- Etc.

30k€ – 35k€/an → 1800€ à 2100€/mois

**Salaires, France**

Lieu:   ou

Trier:

Entreprise	Salairé de base moyen en (EUR)	Fourchette
 <b>Capgemini</b> Développeur Web 3,6 ★ 23 salaires <a href="#">Voir 31 salaire(s) pour tous les lieux</a>	35 305 €/an	20 k € – 49 k €
 <b>Sopra Steria</b> Développeur Web 3,8 ★ 20 salaires <a href="#">Voir 29 salaire(s) pour tous les lieux</a>	36 529 €/an	31 k € – 42 k €
 <b>Web-atrío</b> Développeur Web 4 ★ 14 salaires <a href="#">Voir 14 salaire(s) pour tous les lieux</a>	31 410 €/an	25 k € – 52 k €
 <b>Atos</b> Développeur Web 3,7 ★ 13 salaires <a href="#">Voir 17 salaire(s) pour tous les lieux</a>	36 000 €/an	28 k € – 50 k €
 <b>Orange</b> Développeur Web - Stagiaire mensuel 4 ★ 9 salaires <a href="#">Voir 24 salaire(s) pour tous les lieux</a>	1 430 €/mois	777 € – 2 k €
 <b>onepoint</b> Développeur Web 4,2 ★ 9 salaires <a href="#">Voir 10 salaire(s) pour tous les lieux</a>	42 543 €/an	27 k € – 47 k €
 <b>Expectra</b> Développeur Web 3,9 ★ 9 salaires <a href="#">Voir 9 salaire(s) pour tous les lieux</a>	35 098 €/an	32 k € – 55 k €

# EMPLOI

Pour des chiffres concernant l'emploi :

<https://www.insee.fr/fr/accueil>

De manière globale :

- un très bon taux d'emploi, que ce soit dans des entreprises traditionnelles ou dans des sociétés de prestation de service.
- Possibilité d'auto-entrepreneuriat.
- Historiquement situé en Ile-de-France, mais gros pôle d'activité dans l'Isère et le Rhône.

<https://www.insee.fr/fr/statistiques/6797392>

<https://www.insee.fr/fr/statistiques/5350332>

<https://www.insee.fr/fr/statistiques/4259562>

## Près de la moitié des emplois du numérique localisés en Île-de-France

Marie-Christine Abboudi, Thérèse Ferré, Lynda entreprises, de la concurrence, de la consommation

En 2016, près d'un actif francilien sur dix fois plus qu'en province. Les non-salariés Les onze métiers spécifiques aux technologies en Île-de-France. Ils sont très concentrés en Seine-Saint-Denis, ils ont fortement augmenté de la province puisque, en neuf ans, les emplois de qualification et de féminisation. En outre largement dans les autres secteurs de l'économie.

## Économie du numérique - Une activité fortement concentrée et spécialisée dans l'Isère et le Rhône

Patricia Antoine, Sandra Bouvet (Insee), Olivier Jacod, Christine Jakse (Direction régionale de l'économie, de l'emploi, du travail et des solidarités)

En 2017, en Auvergne-Rhône-Alpes, 117 330 salariés travaillent dans un établissement relevant de l'économie du numérique. Par ailleurs, 62 700 personnes exercent une profession du numérique, tous secteurs confondus. Ces travailleurs sont plus qualifiés, plus diplômés et plus jeunes. La part des femmes y demeure réduite. Entre 2012 et 2017, le nombre de personnes exerçant un tel métier augmente fortement. De plus, la création d'entreprises dans les secteurs numériques est dynamique. Le bon positionnement de la région est largement dû à l'Isère, siège d'activités de fabrication de technologies de l'information et de la communication (TIC), et à la Métropole de Lyon, tournée vers les services des TIC. Complémentaires dans leurs spécialisations, ces deux zones bénéficient de la présence de gros établissements.

INSEE ANALYSES ÎLE-DE-FRANCE

N° 111

Paru le : 05/12/2019

[Découvrir la collection](#)

INSEE ANALYSES AUVERGNE-RHÔNE-ALPES

N° 117

Paru le : 06/04/2021

[Découvrir la collection](#)

VERSION IMPRIMABLE  
(pdf, 1 Mo)



DONNÉES  
(xlsx, 203 Ko)





# LES FORMATIONS

Concernant les formations, les missions locales sont là pour vous proposer, si vous le souhaitez, des formations financées dans le domaine du web.

Je vous laisse donc contacter votre conseiller pour en savoir plus.

Si vous souhaitez cependant plus sur le métier de développeur web, et les formations disponibles, vous avez toujours le site de l'Onisep (Office national d'information sur les enseignements et les professions) :

<https://www.onisep.fr/>



# MERCI

---

Thomas Boyer

[thomas.boyer@ividata.com](mailto:thomas.boyer@ividata.com)