

# Web : HTML/CSS/JavaScript

© 2020 tv <tvaira@free.fr> - v.1.0

<b>HTML5</b>	<b>2</b>
<b>W3C</b>	<b>4</b>
<b>CSS</b>	<b>4</b>
Bootstrap . . . . .	6
<b>DOM</b>	<b>11</b>
<b>Ajax</b>	<b>12</b>
XML . . . . .	12
JSON . . . . .	13
<b>JavaScript</b>	<b>15</b>
jQuery . . . . .	17
Plugins . . . . .	21

# Web : HTML/CSS/JavaScript

Les objectifs de ce TP sont de s'initier à la programmation web (HTML/CSS/JavaScript).

## HTML5



### Définition

**HTML** (*HyperText Markup Language*) est un langage de description de format de document qui se présente sous la forme d'un langage de balisage conçu pour représenter les pages web. Il permet d'écrire de l'hypertexte et également de structurer sémantiquement la page, de mettre en forme le contenu, de créer des formulaires de saisie, d'inclure des ressources multimédias dont des images, des vidéos, et des programmes informatiques.



HTML5 est la dernière révision majeure (finalisée en 2014) du HTML.

Le langage HTML utilise d'**éléments** (ou balise) pour **structurer et décrire un document** :

```
<BALISE>...</BALISE> ou <BALISE> ou <BALISE />
```

Les **attributs** permettent de **préciser les propriétés des éléments HTML** :

```
<BALISE NOM_ATTRIBUT="Valeur">...</BALISE>
```

Les fonctionnalités implémentées par HTML peuvent être réparties ainsi : structure générale d'un document HTML, informations sur la langue, marquage sémantique, listes, tables, hyperliens, inclusion d'images, d'applets et d'objets divers, éléments de regroupement, style de la présentation, marquage de présentation du texte, cadres, formulaire pour l'insertion interactive de données, scripts.

🔗 La spécification HTML5 en anglais : <https://html.spec.whatwg.org/multipage/>

Un document HTML est défini par une **balise racine** nommée HTML. Cette balise accepte l'attribut **lang** qui précise la langue utilisée pour le contenu de la page : `<html lang="fr">`

Le document comprend deux parties : un **en-tête** (pour les métadonnées sur le document) délimité par HEAD et le **corps** du document défini par BODY.

```
document
<html>
  <head>                                en-tête
    <title>Un titre</title>
  </head>
  <body>                                corps
    Ceci est une phrase avec un <a href="cible.html">hyperlien</a>.
    <p>
      Ceci est un paragraphe o&ugrave; il n'y a pas d'hyperlien.
    </p>
    <!-- un commentaire -->
  </body>
</html>
```

Les principaux changements :

- La **version** d'un document HTML : l'instruction DOCTYPE. Un *doctype* (« type de document ») est une instruction au début des documents SGML et XML (et donc HTML) spécifiant sa DTD (*Document Type Definition*). La DTD est un document permettant de décrire un modèle de document SGML, XML ou HTML en indiquant les noms des éléments pouvant apparaître, leur contenu et les attributs. Le document sera jugé valide lorsqu'il possède et respecte sa DTD. Un document HTML valide doit donc déclarer la version HTML qu'il utilise. Pour HTML 5, il faut préciser : `<!DOCTYPE html>`
- Le processus de détection de l'**encodage** a été modifié et s'effectue dans l'ordre :
  - la vérification de la présence d'un en-tête HTTP *Content-Type* (le choix de l'UTF-8 est désormais préconisé par le W3C) :

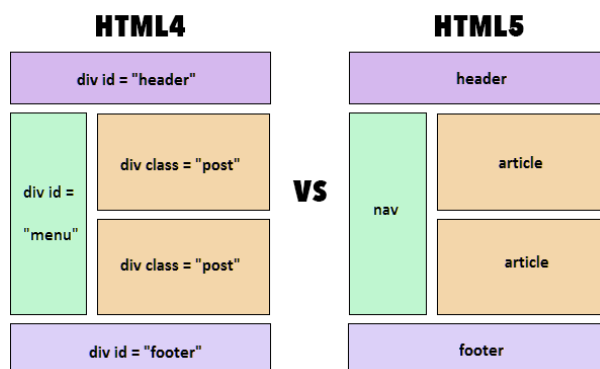
```
// version courte en HTML5
<meta charset="utf-8">

// ou version longue
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

- ensuite la détection du BOM (*Byte Order Mark*) en début de fichier, qui indique l'utilisation d'un encodage unicode ainsi que l'ordre des octets. En UTF-8, l'indicateur d'ordre des octets est codé par la séquence EF BB BF.
- Suppression d'anciennes balises, par exemple :
  - **basefont**, **big**, **center**, **font**, **strike**, **tt** et **u** car leurs effets étaient purement représentatifs ce qui est le rôle de CSS.
  - **frame**, **frameset** et **noframes** pour des problèmes d'accessibilité et d'utilisation
  - **acronym**, **isindex** et **dir** (utilisé alors **ul**)
  - **applet** est remplacé par **object**
- Ajout de nouvelles balises, notamment celles-ci pour la structure d'un document :
  - **main** : définit le contenu principal de la page, il doit être unique dans la page.
  - **section** : définit les sections du document, telles que les chapitres, en-têtes, pieds de page
  - **article** : partie indépendante d'un site, comme un commentaire
  - **header** : spécifie un en-tête (comme une introduction ou des éléments de navigation)
  - **footer** : définit le pied de page d'un article ou un document
  - **nav** : définit une section dans la navigation.

D'autres nouvelles balises importantes : **figure** (et **figcaption**), **audio**, **video**, **embed**, **time**, **canvas**, ...

- HTML4 vs HTML5 : <https://www.w3.org/TR/html5-diff/>



🔗 <https://developer.mozilla.org/fr/docs/Web/Guide/HTML/HTML5>

# W3C



## Définition

**W3C** (*World Wide Web Consortium*) est un organisme de standardisation à but non lucratif (regroupant des centaines d'entreprises partenaires) chargé de promouvoir la compatibilité des technologies du *World Wide Web* telles que HTML5, HTML, XML, CSS, ...

📄 <https://www.w3c.fr/>

Outils de validation (<http://www.w3.org/QA/Tools/#validators>) :

- HTML : <http://validator.w3.org/>
- CSS : <http://jigsaw.w3.org/css-validator/>
- Lien (*link*) : <http://validator.w3.org/checklink>
- Correction, conversion de code : <http://www.w3.org/People/Raggett/tidy/>

# CSS



## Définition

Les **feuilles de style CSS** (*Cascading Style Sheets*) servent **à décrire la présentation des documents HTML et XML**. Les standards définissant CSS sont publiés par le **W3C**.

Les enjeux sont :

- Séparer la structure d'un document de ses styles de présentation
- Décliner les styles de présentation selon le récepteur
- Permettre la cascade des styles

Une **règle CSS** consiste en deux parties : un **sélecteur** ('H1') et une **déclaration** ('color: blue').

La déclaration comprend deux parties : la **propriété** ('color') et la **valeur** associée ('blue').

```
sélecteur[, sélecteur2] { propriété1: valeur; propriété2: valeur }  
H1 { color: blue }
```

Le sélecteur est le lien entre le document HTML et la feuille de style, tous les types d'éléments HTML pouvant être des sélecteurs.

Un sélecteur est une chaîne de caractères qui identifie les éléments sur lesquels s'applique la règle correspondante. Un sélecteur est :

- simple (ex. 'H1') ou
- contextuel (fait de plusieurs sélecteurs simples, p.ex. 'H1 B').

Les différentes syntaxes pour les sélecteurs :

- L'attribut **class** : on l'adresse en le faisant précéder d'un point '.'

```
H1.pastoral { color: #00FF00 }
.pastoral { color: green } /* tous les éléments de la classe pastoral */
```

- L'attribut **id** : on l'adresse en le faisant précéder d'un dièse '#'

```
#prenom { letter-spacing: 0.3em }
```

- Les sélecteurs contextuels peuvent être spécifiés au travers des types d'élément, des attributs **class**, des attributs **id** ou d'une combinaison de ceux-ci :

```
DIV P { font: small sans-serif } /* les balises appartenant à DIV */
#x78y CODE { background: blue }
DIV.sidenote H1 { font-size: large }
H1 B, H2 B, H1 EM, H2 EM { color: red } /* regroupement */
```

- Les pseudo-classes d'ancre (élément <A>) :

```
A:link { color: red } /*lien non-visité*/
A:visited { color: blue } /*lien visité*/
...
```

Pour que les feuilles de style puissent avoir une influence sur la présentation, les agents utilisateurs (les navigateurs par exemple) doivent en connaître l'existence.

Cet exemple regroupe quatre façons de combiner le style CSS à un document HTML :

```
<!doctype html>
<html lang="fr">
  <head>
    <title>Titre</title>
    <link rel=stylesheet type="text/css" href="cool.css" title="Cool"> <!-- 1-->
    <style type="text/css"> <!-- 2-->
      @import url(http://style.com/basic.css); /* 3*/
      h1 { color: blue }
    </style>
  </head>
  <body>
    <h1>Ce titre est en bleu</h1>
    <p style="color: green">et le paragraphe est en vert.</p> <!-- 4-->
  </body>
</html>
```

# Bootstrap



## Définition

**Bootstrap** est une collection d'outils utiles à la création du *design* (graphisme, animation et interactions avec la page dans le navigateur, etc.) de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs.

🔗 <https://getbootstrap.com/>

Bootstrap est essentiellement un **fichier CSS** que l'on intègre dans le bloc `<head>` d'un document HTML :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/
      bootstrap.min.css" integrity="sha384-9aIt2nRpC12Uk9gS9baD1411NQApFmC26EwAOH8WgZl5MYXxPfc
      +NcPb1dKGj7Sk" crossorigin="anonymous">
    <title></title>
  </head>
  <body>
  </body>
</html>
```

Beaucoup des composants de Bootstrap nécessitent tout de même l'utilisation de JavaScript pour fonctionner et plus précisément de jQuery et Popper.js. Il faut placer leur intégration à la fin du document juste avant la balise de fermeture `</body>` (jQuery doit venir en premier, puis Popper.js, puis les *plugins* JavaScript) :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/
      bootstrap.min.css" integrity="sha384-9aIt2nRpC12Uk9gS9baD1411NQApFmC26EwAOH8WgZl5MYXxPfc
      +NcPb1dKGj7Sk" crossorigin="anonymous">
    <title></title>
  </head>
  <body>
    <!-- ... -->
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0
      lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity=
      "sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo" crossorigin="
      anonymous"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"
      integrity="sha384-0gVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI"
      crossorigin="anonymous"></script>
  </body>
</html>
```

Bootstrap comporte de nombreuses classes que l'on utilise directement dans les balises HTML.

➡ Il est possible de tester Bootstrap en ligne : <https://www.codeply.com/p?starter=Bootstrap>

Bootstrap est « *responsive* » en s'adaptant à la taille de l'écran. Il offre un système de grille découpée en 12 colonnes. Chaque colonne représente un pourcentage de la largeur de la fenêtre de visualisation.

La classe `row` représente une ligne. Il existe un ensemble de classe `col` qui définissent le nombre de colonnes en fonction de la largeur de la fenêtre de visualisation.

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	Extra large ≥1200px
Max container width	None (auto)	540px	720px	960px	1140px
Class prefix	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
# of columns	12				
Gutter width	30px (15px on each side of a column)				
Nestable	Yes				
Column ordering	Yes				

🔗 <https://getbootstrap.com/docs/4.5/layout/grid/>

La grille de Bootstrap doit être placée dans un **conteneur**. Bootstrap propose les classes `container` et `container-fluid` :

- la classe `container` aura une largeur fixe pour chaque taille d'écran (xs, sm, md, lg, xl)
- la classe `container-fluid` aura 100% de la largeur disponible

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	Extra large ≥1200px
.container	100%	540px	720px	960px	1140px
.container-sm	100%	540px	720px	960px	1140px
.container-md	100%	100%	720px	960px	1140px
.container-lg	100%	100%	100%	960px	1140px
.container-xl	100%	100%	100%	100%	1140px
.container-fluid	100%	100%	100%	100%	100%

🔗 <https://getbootstrap.com/docs/4.5/layout/overview/>

Exemples de grille :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/
      bootstrap.min.css" integrity="sha384-9aIt2nRpC12Uk9gS9baD1411NQApFmC26EwAOH8WgZl5MYXxPfC
      +NcPb1dKGj7Sk" crossorigin="anonymous">
```

```
<title>Exercice n°6</title>
<style>
/* * {padding: 0;margin: 0;} */
body {padding: 0;margin: 10px;}
[class*="col-"] {
    text-align: center;
}
</style>
</head>
<body>
    <div class="container">
        <div class="row">
            <div class="col-3 border bg-light">col-3</div>
            <div class="col-3 border bg-light">col-3</div>
            <div class="col-3 border bg-light">col-3</div>
            <div class="col-3 border bg-light">col-3</div>
        </div>
    </div>
    <div class="container-lg">
        <div class="row">
            <div class="col-lg-3 border bg-light">col-lg-3</div>
            <div class="col-lg-3 border bg-light">col-lg-3</div>
            <div class="col-lg-3 border bg-light">col-lg-3</div>
            <div class="col-lg-3 border bg-light">col-lg-3</div>
        </div>
    </div>
    <div class="container-md">
        <div class="row">
            <div class="col-md-3 border bg-light">col-md-3</div>
            <div class="col-md-3 border bg-light">col-md-3</div>
            <div class="col-md-3 border bg-light">col-md-3</div>
            <div class="col-md-3 border bg-light">col-md-3</div>
        </div>
    </div>
    <div class="container-sm">
        <div class="row">
            <div class="col-sm-3 border bg-light">col-sm-3</div>
            <div class="col-sm-3 border bg-light">col-sm-3</div>
            <div class="col-sm-3 border bg-light">col-sm-3</div>
            <div class="col-sm-3 border bg-light">col-sm-3</div>
        </div>
    </div>
    <div class="container-fluid">
        <div class="row">
            <div class="col-3 border bg-light">col-3</div>
            <div class="col-3 border bg-light">col-3</div>
            <div class="col-3 border bg-light">col-3</div>
            <div class="col-3 border bg-light">col-3</div>
        </div>
    </div>
    <div class="container-fluid">
        <div class="row">
            <div class="col-lg-3 border bg-light">col-lg-3</div>
            <div class="col-lg-3 border bg-light">col-lg-3</div>
            <div class="col-lg-3 border bg-light">col-lg-3</div>
            <div class="col-lg-3 border bg-light">col-lg-3</div>
        </div>
    </div>
    <div class="container-fluid">
        <div class="row">
```



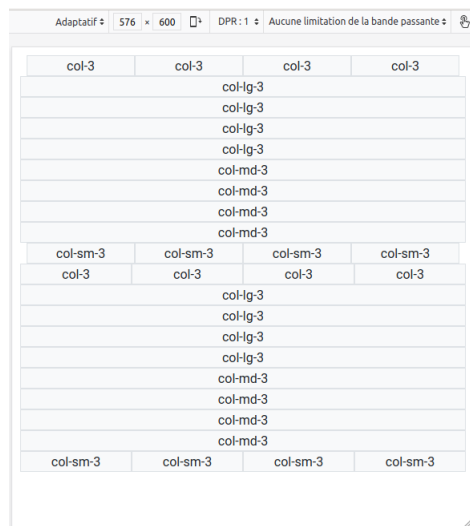
```

        <div class="col-md-3 border bg-light">col-md-3</div>
        <div class="col-md-3 border bg-light">col-md-3</div>
        <div class="col-md-3 border bg-light">col-md-3</div>
        <div class="col-md-3 border bg-light">col-md-3</div>
    </div>
</div>
<div class="container-fluid">
    <div class="row">
        <div class="col-sm-3 border bg-light">col-sm-3</div>
        <div class="col-sm-3 border bg-light">col-sm-3</div>
        <div class="col-sm-3 border bg-light">col-sm-3</div>
        <div class="col-sm-3 border bg-light">col-sm-3</div>
    </div>
</div>
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="sha384-Q6E9RHvbiYzFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo" crossorigin="anonymous"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" integrity="sha384-OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI" crossorigin="anonymous"></script>
</body>
</html>

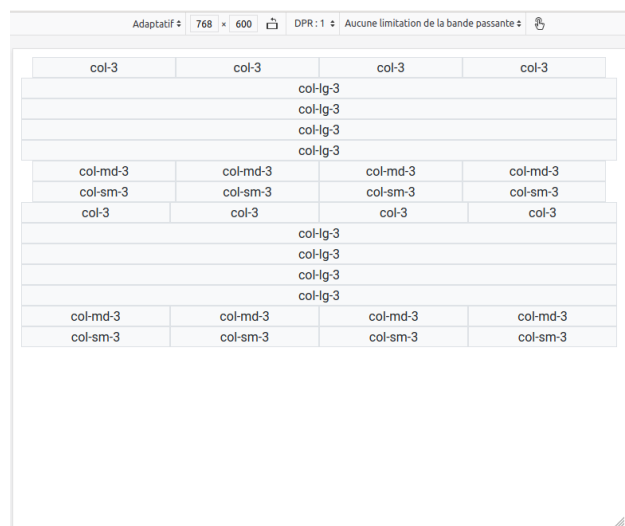
```



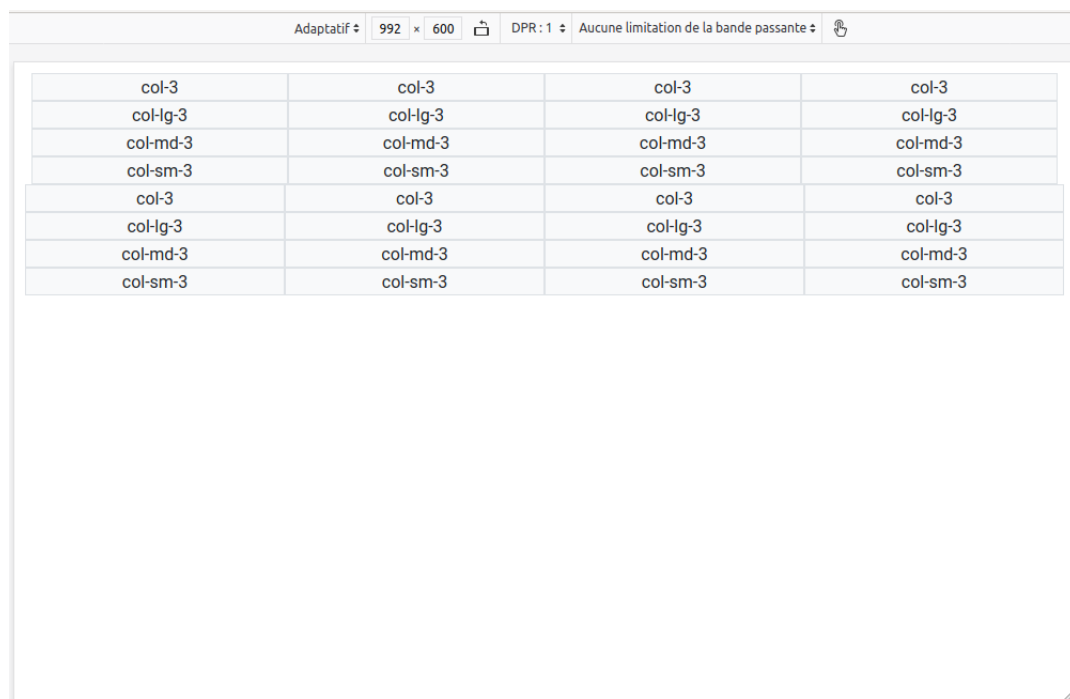
*Extra small*



Small 576 px



Medium 768 px



Large 992 px



Il est possible de combiner plusieurs classes `col-*` dans un élément HTML : `<div class="col-xs-4 col-sm-3 col-md-2">...</div>`

Bootstrap propose de nombreux composants à intégrer dans un document HTML : formulaires, boutons, etc ... <https://getbootstrap.com/docs/4.0/components/>

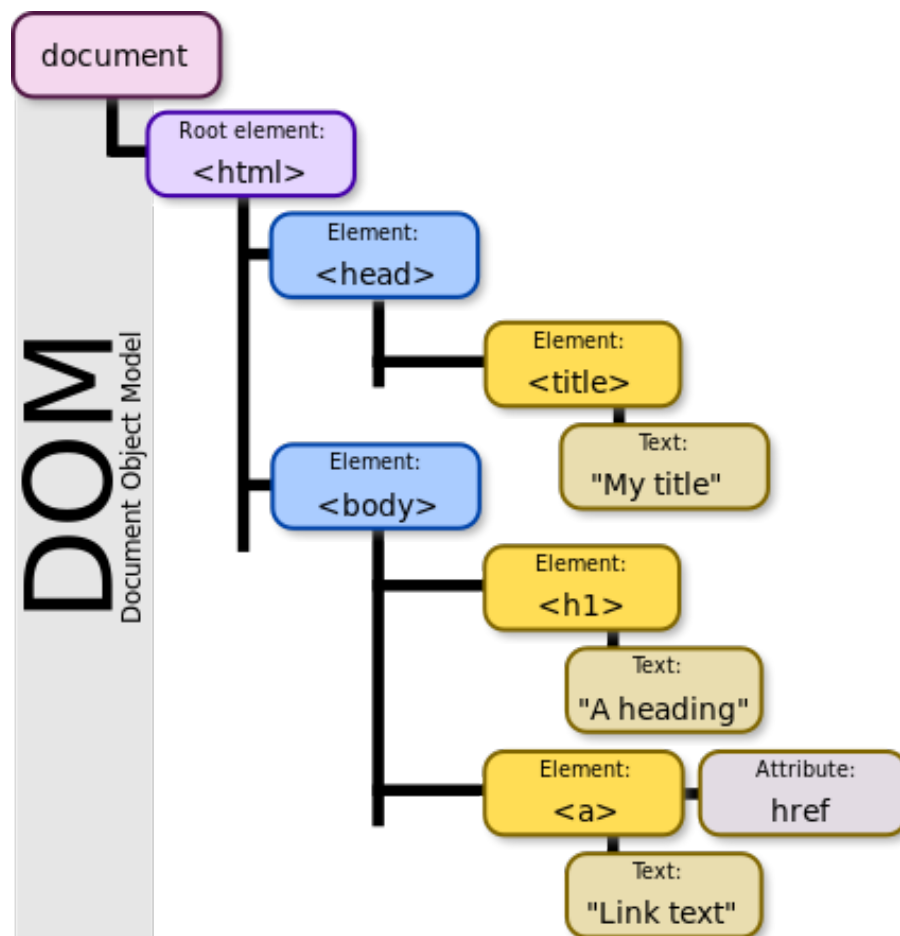
🔗 Voir aussi : **Foundation** (<https://foundation.zurb.com/>), **Blueprint** (<http://www.blueprintcss.org/>), **jQuery Mobile** (<http://jquerymobile.com/>), ...

# DOM



## Définition

**DOM** (*Document Object Model*) est une interface de programmation normalisée par le W3C, qui permet à des scripts d'examiner et de modifier le contenu du navigateur web. Le DOM représente le document affiché dans le navigateur par une structure en arbre, comportant des nœuds (branches et feuilles). Les objets du DOM peuvent représenter une fenêtre, un document, une phrase, un style ...



# Ajax



## Définition

**AJAX** est un acronyme signifiant *Asynchronous JavaScript and XML* et désignant une solution informatique libre pour le développement d'applications Web.

AJAX n'est pas une technologie en elle-même, mais un terme qui évoque l'utilisation conjointe d'un ensemble de technologies libres couramment utilisées sur le Web :

- HTML (ou XHTML) pour la structure sémantique des informations ;
- CSS pour la présentation des informations ;
- DOM et JavaScript pour afficher et interagir dynamiquement avec l'information présentée ;
- l'objet `XMLHttpRequest` pour échanger et manipuler les données de manière asynchrone avec le serveur Web.
- XML/JSON pour le format des données informatives et visuelles

L'avantage principal est dans le côté **asynchrone**. La page entière ne doit plus être rechargée en totalité lorsqu'une partie doit changer ce qui entraîne un gain de temps et une meilleure interaction avec le serveur et donc le client.

Pour faciliter l'utilisation de ces technologies, de nombreux *frameworks* (jQuery est le plus souvent rencontré) ont été mis en place. Il s'agit en général d'un ensemble de bibliothèques JavaScript permettant de réaliser les traitements asynchrones et d'offrir une ergonomie avancée grâce à une palette d'objets graphiques aboutis.

# XML



## Définition

**XML** (*eXtensible Markup Language*) est un métalangage informatique de balisage générique pour créer d'autres langages et destiné à l'échange d'informations et de documents.

Un document XML respecte généralement certains principes :

- « bien formé » signifie que le document obéit aux règles syntaxiques de la spécification XML.
- « valide » signifie que le document est « bien formé » et répond à une structure définie par une DTD.

**XML** (*eXtensible Markup Language*) est standardisé par la spécification W3C : <http://www.yoyodesign.org/doc/w3c/xml11/>.



XML est plus qu'un langage, c'est une famille de langages. Actuellement on estime que plusieurs centaines de « langages » basés sur XML ont été décrits : XHTML, XSL, XSLT, Xpath, XLink, XPointer, XML-Schema, RSS, MathML, SVG, OpenDocument, ...

Exemple :

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<menu id="file" value="File">
  <popup>
    <menuitem value="New" onclick="CreateNewDoc()" />
    <menuitem value="Open" onclick="OpenDoc()" />
    <menuitem value="Close" onclick="CloseDoc()" />
  </popup>
</menu>
```

## JSON



### Définition

**JSON** (*JavaScript Object Notation*) est un format de données textuelles dérivé de la notation des objets du langage JavaScript. Il permet de représenter de l'information structurée comme le permet XML par exemple.

Un document JSON comprend deux types d'éléments structurels : **des ensembles de paires « nom » (alias « clé ») / « valeur » et des listes ordonnées de valeurs.**

Ces mêmes éléments représentent trois types de données : des objets, des tableaux et des valeurs génériques de type tableau, objet, booléen, nombre, chaîne de caractères ou *null*.

Exemple :

```
{
  "menu": {
    "id": "file",
    "value": "File",
    "popup": {
      "menuitem": [
        { "value": "New", "onclick": "CreateNewDoc()" },
        { "value": "Open", "onclick": "OpenDoc()" },
        { "value": "Close", "onclick": "CloseDoc()" }
      ]
    }
  }
}
```

Il est notamment utilisé comme langage de transport de données par AJAX et les services Web.

Exemple :

**Openweather** fournit des API simples et rapides pour travailler avec leur base de données de données météorologiques, d'images satellite et d'autres données environnementales. Il existe notamment une API pour les données météorologiques.

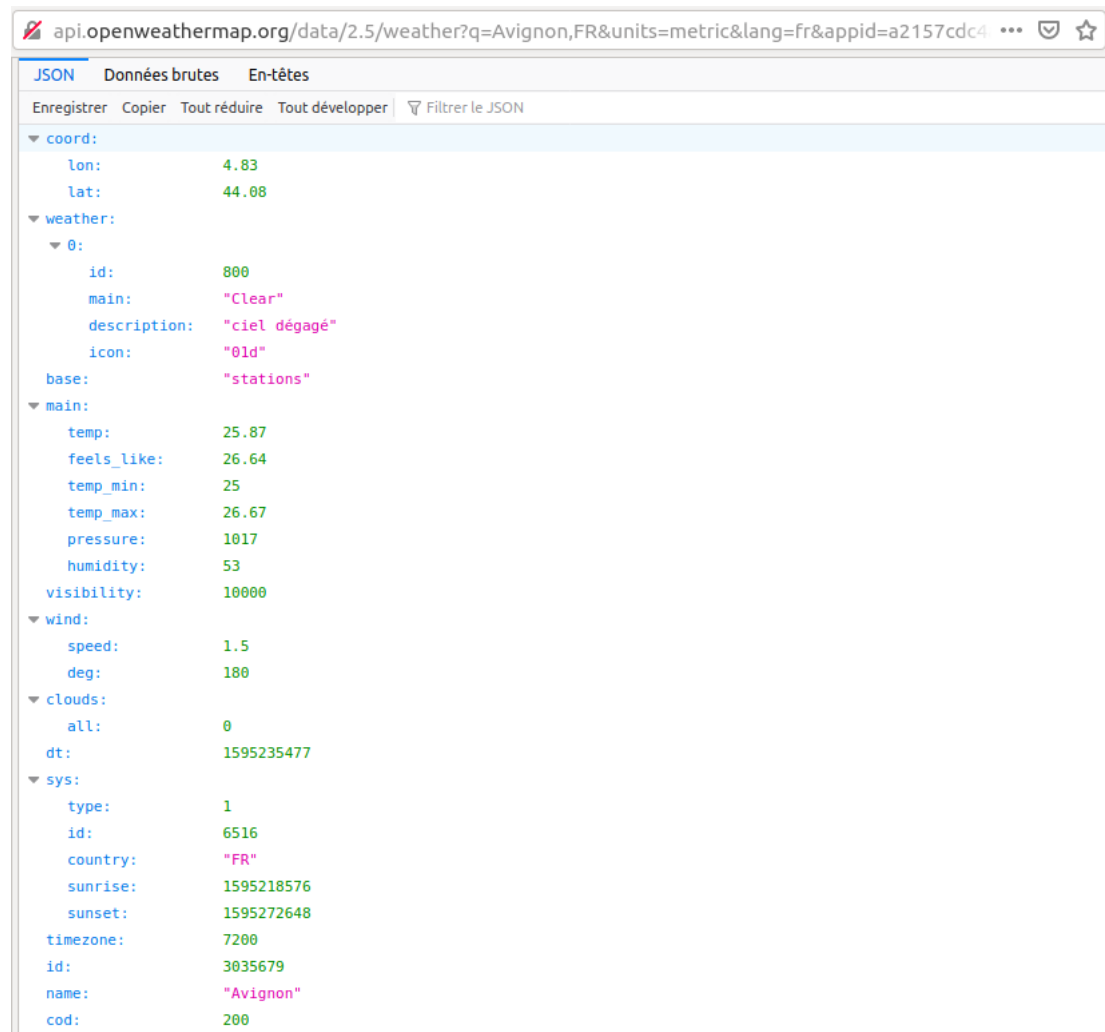
⇒ Obtenir les données météorologiques d'Avignon au format JSON :

- <http://api.openweathermap.org/data/2.5/weather?q=Avignon,FR&units=metric&lang=fr&APPID=xxxx>
- <http://api.openweathermap.org/data/2.5/forecast?id=6455379&APPID=xxxx>



```
{
  "coord": {
    "lon": 4.83,
    "lat": 44.08
  },
  "weather": [
    {
      "id": 800,
      "main": "Clear",
      "description": "ciel d\u00e9gag\u00e9",
      "icon": "01d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 25.87,
    "feels_like": 26.64,
    "temp_min": 25,
    "temp_max": 26.67,
    "pressure": 1017,
    "humidity": 53,
    "visibility": 10000,
    "wind": {
      "speed": 1.5,
      "deg": 180
    },
    "clouds": {
      "all": 0
    },
    "dt": 1595235477,
    "sys": {
      "type": 1,
      "id": 6516,
      "country": "FR",
      "sunrise": 1595218576,
      "sunset": 1595272648,
      "timezone": 7200,
      "id": 3035679,
      "name": "Avignon",
      "cod": 200
    }
  }
}
```

### Donn\u00e9es brutes



```
{
  "coord": {
    "lon": 4.83,
    "lat": 44.08
  },
  "weather": [
    {
      "id": 800,
      "main": "Clear",
      "description": "ciel d\u00e9gag\u00e9",
      "icon": "01d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 25.87,
    "feels_like": 26.64,
    "temp_min": 25,
    "temp_max": 26.67,
    "pressure": 1017,
    "humidity": 53,
    "visibility": 10000,
    "wind": {
      "speed": 1.5,
      "deg": 180
    },
    "clouds": {
      "all": 0
    },
    "dt": 1595235477,
    "sys": {
      "type": 1,
      "id": 6516,
      "country": "FR",
      "sunrise": 1595218576,
      "sunset": 1595272648,
      "timezone": 7200,
      "id": 3035679,
      "name": "Avignon",
      "cod": 200
    }
  }
}
```

### JSON

# JavaScript



## Définition

**JavaScript** (qui est souvent abrégé en « JS ») est un langage de script léger, orienté objet, principalement connu comme le langage de script des pages web. Mais il est aussi utilisé dans de nombreux environnements extérieurs aux navigateurs web tels que **Node.js**. Le code JavaScript est interprété ou compilé à la volée (JIT).

Le standard pour JavaScript est **ECMAScript**. Le langage JavaScript est un des langages les plus populaires actuellement.



JavaScript a été créé en 1995 par Brendan Eich. Il a été standardisé sous le nom d'ECMAScript en juin 1997 par Ecma International dans le standard ECMA-262. JavaScript n'est depuis qu'une implémentation d'ECMAScript, celle mise en œuvre par la fondation Mozilla. L'implémentation d'ECMAScript par Microsoft (dans Internet Explorer jusqu'à sa version 9) se nomme JScript, tandis que celle d'Adobe Systems se nomme ActionScript. JavaScript ne doit pas être confondu avec le langage de programmation Java.

C'est un langage à objets utilisant le concept de **prototype** (une POO sans classe, fondée sur la notion de prototype qui est un objet à partir duquel on crée de nouveaux objets) :

- Tous les objets sont des instances ;
- On définit et on crée un ensemble d'objets avec des fonctions qui sont des constructeurs ;
- On crée un seul objet grâce à l'opérateur **new** ;
- Le constructeur ou le prototype définit un ensemble de propriétés initiales. Il est possible d'ajouter ou de retirer des propriétés dynamiquement.



Les classes JavaScript ont été introduites avec ECMAScript 2015 : <https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Classes>.

## Ressources :

- Site officiel : <https://developer.mozilla.org/fr/docs/Web/JavaScript>
- Standard ECMA-262 : [www.ecma-international.org](http://www.ecma-international.org)
- Node.js : <https://nodejs.org/en/>
- [https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Utiliser\\_les\\_objets](https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Utiliser_les_objets)
- [https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Le\\_modèle\\_objet\\_JavaScript\\_en\\_détails](https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Le_modèle_objet_JavaScript_en_détails)
- Cours JavaScript : <http://tvaira.free.fr/web/cours-javascript.pdf> et <http://tvaira.free.fr/web/>

Le langage JavaScript est doté d'un **typage dynamique faible**.



Le typage dynamique consiste à laisser l'interpréteur associer un type « à la volée » lors de l'exécution du code. C'est la valeur affectée à la variable qui précisera son type. Un langage de programmation est dit faiblement typé lorsqu'il ne considère pas comme une erreur les changements de types.

```
var a = 1; // un nombre
var b = 2.5; // un nombre
var c = "hello"; // une chaine de caracteres
var d; // undefined

console.log(typeof a);
console.log(typeof b);
console.log(typeof c);
console.log(typeof d);

d = a; // un nombre

console.log(a);
console.log(b);
console.log(c);
console.log(d);

var x = "5"; // une chaine de caracteres
var y = 1+ y; // un nombre
console.log(typeof x);
console.log(typeof y);

console.log(y); // NaN (Not a Number)
```

#### *Exemple d'utilisation des types en Javascript*

Le code JavaScript a besoin d'un objet global pour y rattacher les déclarations (variables et fonctions) avant d'exécuter des instructions. La situation la plus connue est celle de l'objet **window** obtenu dans le contexte d'une page web. Du code JavaScript peut être intégré directement au sein des pages web (avec les balises `<script></script>`, pour y être exécuté sur le poste client. C'est alors le navigateur web qui prend en charge l'exécution de ces programmes appelés scripts.

```
<!DOCTYPE html>
<html dir="ltr" lang="fr">
  <head>
    <meta charset="utf8" />
    <script>
      alert('Hello world!');
    </script>
  </head>
  <body>
  </body>
</html>
```

#### *Du code JavaScript intégré directement dans une page web*



JavaScript peut également être utilisé comme langage de programmation sur un serveur HTTP à l'image des langages comme PHP. Il existe par ailleurs des projets indépendants et Open Source d'implémentation de serveurs en JavaScript. Parmi eux, on pourra distinguer Node.js, une plateforme polyvalente de développement d'applications réseau.

Le langage JavaScript utilise une syntaxe très proche de celle utilisée en C/C++.

En C, chaque instruction se termine par un point-virgule (;). JavaScript est plus souple, permettant à une fin de ligne de marquer implicitement la fin d'une instruction. Cet usage est déconseillé.

➡ Il est possible de tester JavaScript en ligne : <https://jsfiddle.net/>



# jQuery



## Définition

**jQuery** est une bibliothèque JavaScript libre et multiplateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web.

Le but de la bibliothèque étant le parcours et la modification du DOM. Elle contient de nombreuses fonctionnalités : notamment des animations, la manipulation des feuilles de style en cascade (accessibilité des classes et attributs), la gestion des événements, etc.

L'utilisation d'Ajax est facilitée et de nombreux *plugins* sont présents.

jQuery a connu un large succès auprès des développeurs Web et son apprentissage est aujourd'hui un des fondamentaux de la formation aux technologies du Web. Il est à l'heure actuelle la bibliothèque *front-end* la plus utilisée au monde (plus de la moitié des sites Internet en ligne intègrent jQuery).

Émergence de nouvelles librairies : React (JavaScript) et Vue.js

→ Il est possible de tester jQuery en ligne : <https://jsfiddle.net/boilerplate/jquery>

Pour utiliser jQuery, il faut l'intégrer au document HTML :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title></title>
  </head>
  <body>
    <!-- ... -->
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0
      lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>
  </body>
</html>
```

🔗 <https://api.jquery.com/>

jQuery est utilisé pour manipuler en lecture et en écriture le DOM (*Document Object Model*). Il faudra donc attendre l'évènement *ready* qui indique que le document a été complètement créé :

```
jQuery(document).ready(function() {
  alert('DOM prêt');
});

// Ou en utilisant l'alias $
$(document).ready(function() {
  alert('DOM prêt');
});

// Ou encore
$(function() {
  alert('DOM prêt');
});
```



Dans certains cas, il faut attendre l'évènement *load* qui indique que le document a été complètement chargé dans la fenêtre (objet window).

À chaque fois que la fonction `$()` est appelée, elle retourne un objet `jQuery` qui est un tableau de type liste de noeuds DOM (la propriété `length` indiquera le nombre d'éléments). Il est possible de sélectionner un ou plusieurs éléments en passant en argument un sélecteur :

- par balise : `$("p")`
- par id : `$("#paragraphe")`
- par classe : `$(".paragraphe")`

Les fonctionnalités de jQuery s'utilisent ensuite en appelant les méthodes (des fonctions) de l'objet retourné :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Exemples jQuery</title>
    <style>
      body {padding: 0;margin: 10px;}
    </style>
  </head>
  <body>
    <p></p>
    <p id="paragraphe"></p>
    <p class="paragraphe"></p>
    <script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
    <script>
      $(document).ready(function() {
        //alert('Dom prêt !');
        // Modification HTML
        $("p").html("Un paragraphe");
        $("#paragraphe").html("Un deuxième paragraphe");
        $(".paragraphe").html("Un autre paragraphe");
        // Modification CSS
        $("p").css("color", "red");
        $("#paragraphe").css("color", "blue");
        $(".paragraphe").css("color", "green");
        // Évènement et effet
        $("p").click(function(){
          alert('Clic sur p !');
        });
        $("#paragraphe").click(function(){
          console.log($(this).text());
          $(this).fadeToggle( "slow", "linear" );
          $(this).fadeToggle( "slow", "linear" );
          if ( $(".paragraphe").is( ":hidden" ) ) {
            $(".paragraphe").show();
          }
        });
        $(".paragraphe").on('click', function(event) {
          console.log("Type : " + event.type);
          $(this).hide();
        });
      });
    </script>
  </body>
</html>
```

## jQuery et AJAX :

```

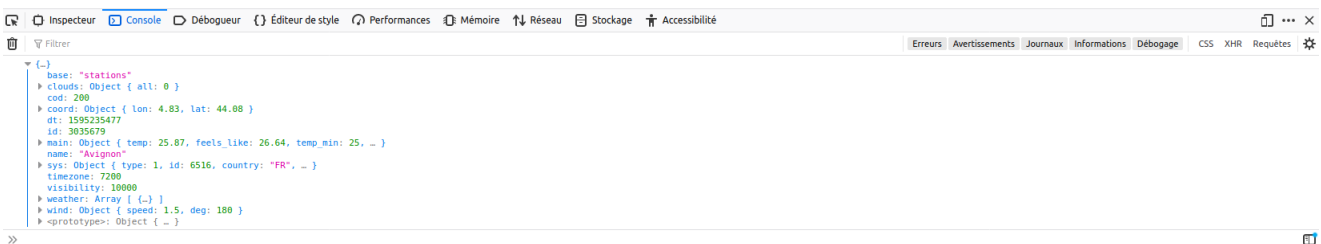
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>jQuery et AJAX</title>
    <style>
      body {padding: 0;margin: 10px;}
    </style>
  </head>
  <body>
    <p id="meteo"></p>
    <script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
    <script>
      $(document).ready(function() {
        $.ajax({
          type: "GET",
          url: "http://api.openweathermap.org/data/2.5/weather?q=Avignon,FR&units=metric&lang=fr&appid=a2157cdc4a03c47c79e8414161c59762",
          dataType: 'json',
          success: function(data)
          {
            console.log(data);
            $('#meteo').html(JSON.stringify(data));
          },
          error: function (xhr, status, error) {
            console.log("Erreur : " + status + " " + error + " " + xhr.status + " " +
              xhr.statusText);
            alert("Erreur : " + status + " " + error + " " + xhr.status + " " + xhr.
              statusText);
          }
        });
      });
    </script>
  </body>
</html>

```

```

{"coord":{"lon":4.83,"lat":44.08},"weather":[{"id":800,"main":"Clear","description":"ciel dégagé","icon":"01d"}],"base":"stations","main":
{"temp":25.87,"feels_like":26.64,"temp_min":25,"temp_max":26.67,"pressure":1017,"humidity":53},"visibility":10000,"wind":{"speed":1.5,"deg":180},"clouds":{"all":0},"dt":1595235477,"sys":
{"type":1,"id":6516,"country":"FR","sunrise":1595218576,"sunset":1595272648},"timezone":7200,"id":3035679,"name":"Avignon","cod":200}

```



## JSON

## Extraction des données au format JSON :

```

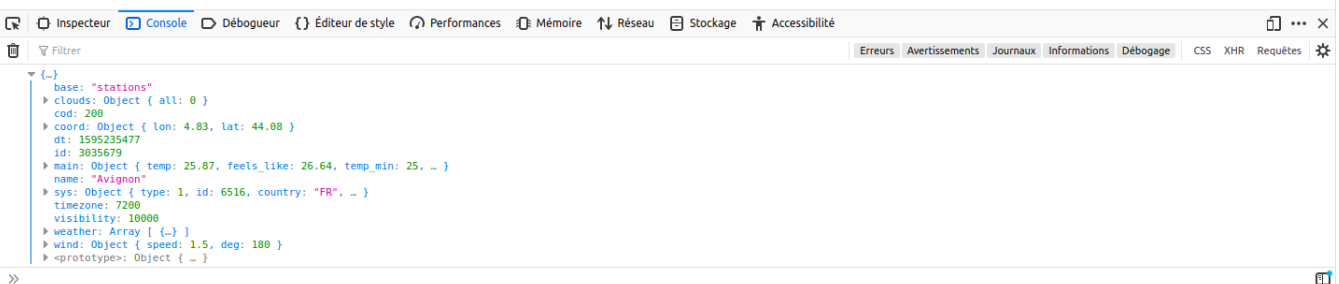
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>jQuery et AJAX/JSON</title>
    <style>
      body {padding: 0;margin: 10px;}
    </style>

```

```
</head>
<body>
  <p id="meteo"></p>
  <script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
  <script>
    function getMeteo(ville) {
      $.ajax({
        type: "GET",
        url: "http://api.openweathermap.org/data/2.5/weather?q=" + ville + ",FR&units=metric&lang=fr&appid=a2157cdc4a03c47c79e8414161c59762",
        dataType: 'json',
        success: function(data)
        {
          console.log(data);
          //$('#meteo').html(JSON.stringify(data));

          cityName = data.name;
          var date = new Date();
          date.setTime(data.dt*1000);
          horodatage = date.toLocaleDateString("fr-FR") + " à " + date.
            toLocaleTimeString("fr-FR");
          temperature = data.main.temp;
          humidity = data.main.humidity;
          pressure = data.main.pressure;
          weather = data.weather[0].description;
          result = cityName + " " + horodatage + " : " + weather + "<br />" + "
            Température : " + temperature + "°C Humidité : " + humidity + "%
            Pression : " + pressure + "hPa";
          $('#meteo').html(result);
        },
        error: function (xhr, status, error) {
          console.log("Erreur : " + status + " " + error + " " + xhr.status + " " +
            xhr.statusText);
          alert("Erreur : " + status + " " + error + " " + xhr.status + " " + xhr.
            statusText);
        }
      });
    }
    $(document).ready(function() {
      getMeteo("Avignon");
      // ou périodiquement :
      setInterval(function() {
        getMeteo("Avignon");
      }, 1000);
    });
  </script>
</body>
</html>
```

Avignon 20/07/2020 à 10:57:57 : ciel dégagé  
Température : 25.87°C Humidité : 53% Pression : 1017hPa



## Plugins

Il existe de très nombreux *plugins* JavaScript.

Par exemple, **Google Charts** fournit un moyen pour visualiser les données dans un document Web. Des graphiques linéaires simples aux cartes arborescentes hiérarchiques complexes, la galerie de graphiques fournit un grand nombre de types de graphiques prêts à l'emploi.

🔗 <https://developers.google.com/chart/interactive/docs>

Exemple *Gauge* :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Google Charts Gauge</title>
    <style>
      body {padding: 0;margin: 10px;}
    </style>
  </head>
  <body>
    <p id="meteo"></p>
    <div id="chart_div" style="width: 400px; height: 120px;"></div>
    <script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
    <script>
      google.charts.load('current', {'packages':['gauge'], 'language': 'fr'});
      google.charts.setOnLoadCallback(initChart);
      var options = {
        width: 600, height: 200,
        minorTicks: 5
      };

      function initChart() {
        chart = new google.visualization.Gauge(document.getElementById('chart_div'));
        datas = google.visualization.arrayToDataTable([
          ['Label', 'Value'],
          ['Température', 0],
          ['Humidité', 0]
        ]);
      }

      function getMeteo() {
        $.ajax({
          type: "GET",
          url: "http://api.openweathermap.org/data/2.5/weather?q=Avignon,FR&units=metric&lang=fr&appid=a2157cdc4a03c47c79e8414161c59762",
          dataType : 'json',
          success: function(data)
          {
            console.log(data);
            //$('#meteo').html(JSON.stringify(data));
            cityName = data.name;
            var date = new Date();
            date.setTime(data.dt*1000);
            horodatage = date.toLocaleDateString("fr-FR") + " à " + date.
              toLocaleTimeString("fr-FR");
            temperature = data.main.temp;
            humidity = data.main.humidity;
            pressure = data.main.pressure;
          }
        });
      }
    </script>
  </body>
</html>
```

```

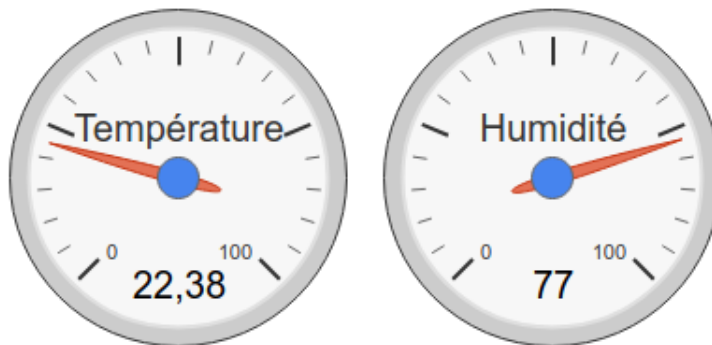
        weather = data.weather[0].description;
        result = cityName + " " + horodatage + " : " + weather + "<br />" + "
          Température : " + temperature + "°C Humidité : " + humidity + "%
          Pression : " + pressure + "hPa";
        $('#meteo').html(result);

        datas.setValue(0, 1, temperature);
        datas.setValue(1, 1, humidity);
        chart.draw(datas, options);
      },
      error: function (xhr, status, error) {
        console.log("Erreur : " + status + " " + error + " " + xhr.status + " " +
          xhr.statusText);
        alert("Erreur : " + status + " " + error + " " + xhr.status + " " + xhr.
          statusText);
      }
    });
  }

  $(document).ready(function() {
    getMeteo();
    setInterval(function() {
      getMeteo();
    }, 1000);
  });
</script>
</body>
</html>

```

Avignon 21/07/2020 à 08:49:13 : ciel dégagé  
 Température : 22.38°C Humidité : 77% Pression : 1020hPa



Exemple *Line Chart* :

```

<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Google Charts Line Chart</title>
    <style>
      body {padding: 0;margin: 10px;}
    </style>
  </head>
  <body>
    <p id="meteo"></p>
    <div id="chart_div"></div>

```

```
<script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
<script>
    google.charts.load('current', {'packages':['corechart'], 'language': 'fr'});
    google.charts.setOnLoadCallback(initChart);
    var options = {
        title: 'Météo',
        width: 900,
        height: 500,
        series: {
            0: {targetAxisIndex: 0},
            1: {targetAxisIndex: 1}
        },
        vAxes: {
            0: {title: 'Température (Celsius)'},
            1: {title: 'Humidité'}
        },
        vAxis: {
            maxValue: 100,
            minValue: 0,
            viewWindow: {max: 100},
            gridlines: {minSpacing: 5}
        }
    };
    var lastHorodatage = 0;

    function initChart() {
        chart = new google.visualization.LineChart(document.getElementById('chart_div'));
        datas = new google.visualization.DataTable();
        datas.addColumn('string', 'Horodatage');
        datas.addColumn('number', 'Température');
        datas.addColumn('number', 'Humidité');
    }

    function getMeteo() {
        $.ajax({
            type: "GET",
            url: "http://api.openweathermap.org/data/2.5/weather?q=Avignon,FR&units=metric&lang=fr&appid=a2157cdc4a03c47c79e8414161c59762",
            dataType: 'json',
            success: function(data)
            {
                console.log(data);
                //$('##meteo').html(JSON.stringify(data));
                cityName = data.name;
                var date = new Date();
                date.setTime(data.dt*1000);
                horodatage = date.toLocaleDateString("fr-FR") + " à " + date.
                    toLocaleTimeString("fr-FR");
                temperature = data.main.temp;
                humidity = data.main.humidity;
                pressure = data.main.pressure;
                weather = data.weather[0].description;
                result = cityName + " " + horodatage + " : " + weather + "<br />" + "
                    Température : " + temperature + "°C Humidité : " + humidity + "%
                    Pression : " + pressure + "hPa";
                $('##meteo').html(result);
                if(lastHorodatage != data.dt)
                {
                    datas.addRow([date.toLocaleTimeString("fr-FR"), temperature, humidity]);
                }
            }
        });
    }
}
```

```
        chart.draw(datas, options);
        lastHorodatage = data.dt;
    }
},
error: function (xhr, status, error) {
    console.log("Erreur : " + status + " " + error + " " + xhr.status + " " +
        xhr.statusText);
    alert("Erreur : " + status + " " + error + " " + xhr.status + " " + xhr.
        statusText);
}
});
}

$(document).ready(function() {
    getMeteo();
    setInterval(function() {
        getMeteo();
    }, 5000);
});
</script>
</body>
</html>
```

Avignon 21/07/2020 à 10:13:30 : nuageux  
Température : 25.22°C Humidité : 69% Pression : 1020hPa

