



# Formation HTML5, CSS3 JavaScript

Du 04 - 08 Octobre 2018



Jours | et 2  
HTML5 - CSS3

# Introduction

Dans cette partie :

- HTML, XHTML et CSS
- Les navigateurs Internet
- Les éditeurs XHTML/CSS

# HTML, XHTML et CSS

HTML est le langage de base permettant de définir des pages Web. Il est composé de balises (aussi appelées marqueurs ou tags) qui décrivent et mettent en forme des contenus.

XHTML est un langage assez proche d'HTML, à quelques nuances près : toutes les balises doivent avoir un parent, et le parent de tous les parents est <html>. Voici la structure minimale d'une page XHTML :

```
<html>
  <head>
    ...
  </head>
  <body>
    ...
  </body>
</html>
```

Quelques restrictions :

- pas de majuscules dans les noms et attributs des balises ;
- balises de fin obligatoires (les balises HTML non terminées prennent un slash de fin, comme par exemple <br> ou <img> ;
- les balises doivent être correctement imbriquées. A titre d'exemple, seule la seconde ligne de code est correcte en XHTML :

```
<b><i>Ce texte est gras et italique</b></i>
```

```
<b><i>Ce texte est gras et italique</i></b>
```

Le langage HTML ou XHTML est utilisé pour définir le contenu des pages Web.

Le langage CSS est utilisé pour définir la mise en forme et la mise en page des contenus HTML. Il peut être utilisé directement dans les balises HTML/XHTML, dans l'en-tête des pages ou dans un fichier externe appelé "feuille de styles".



## Editeurs HTML/XHTML/CSS

La saisie du code HTML peut se faire dans un simple éditeur de texte, comme le Bloc-Notes de Windows. Mais vous lui préférerez un éditeur de code à coloration syntaxique.

Plusieurs éditeurs de ce type sont diffusés sous la forme de freewares. Vous utiliserez par exemple NotePad ++ ou Sublime Text 2.

# Pourquoi utiliser HTML5 et CSS3 ?

Le *World Wide Web Consortium* (W3C) est un organisme de standardisation chargé de promouvoir la compatibilité des technologies du Web, telles que HTML, XHTML, CSS, PNG, SVG, SOAP, etc. La dernière spécification du langage HTML 4.01 date de 1999.

Il était temps qu'une nouvelle version du langage voie le jour, car le Web a bien évolué depuis presque 15 ans !

Vous en saurez plus sur les spécifications de ce tout nouveau langage en accédant aux pages [www.w3.org/TR/html5/](http://www.w3.org/TR/html5/) et [www.w3.org/TR/html-markup/](http://www.w3.org/TR/html-markup/).

# Pourquoi HTML5 et CSS3 ?

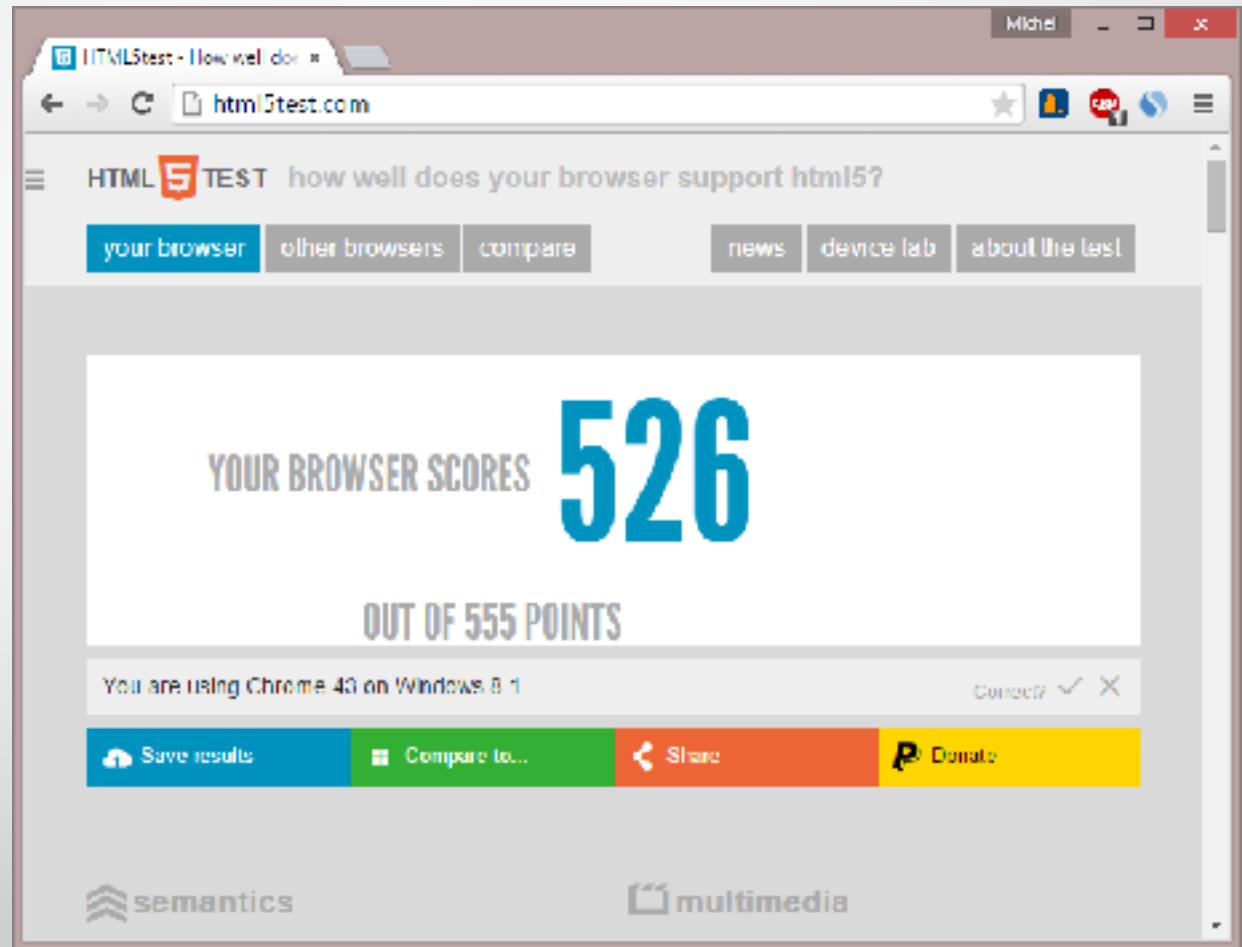
HTML est en mesure d'afficher des éléments textuels, graphiques et vidéos dans un navigateur web et d'assurer leur mise en page. Cette seconde fonction est cependant assez limitée. C'est la raison d'être du langage CSS, qui se charge de la mise en forme et de la mise en page des éléments HTML. La version 3 du langage apporte son lot de nouveautés, toutes plus impressionnantes les unes que les autres.

La standardisation de ce langage est assurée par la section CSS3 Working Group du W3C. Pour suivre l'état d'avancement des différents modules qui la composent, rendez-vous à [www.css3.info/modules/](http://www.css3.info/modules/).

# Compatibilité avec les navigateurs

Les nouveaux éléments propres au langage HTML5 ne sont pas encore entièrement implémentés sur les principaux navigateurs.

Pour connaître le taux de compatibilité de votre navigateur, rendez-vous à la page <http://html5test.com/>.



Testez la compatibilité de vos navigateurs

# Compatibilité avec les navigateurs

Les principaux navigateurs utilisent un préfixe pour (pré)implémenter les nouvelles propriétés CSS3 :

- -moz pour les navigateurs Mozilla (Firefox) ;
- -webkit pour les navigateurs Webkit (Safari, OmniWeb, Midori, etc.) ;
- -khtml pour les navigateurs Konqueror ;
- -o pour les navigateurs Opera ;
- -ms pour le navigateur Internet Explorer 9 et supérieur.

Tant que la spécification du langage n'a pas atteint au moins le statut de *recommandation candidate*, vous devrez utiliser plusieurs préfixes dans les propriétés CSS3 pour assurer la plus grande compatibilité possible. À terme, ces préfixes ne devraient plus avoir cours et une seule instruction devrait être interprétée à l'identique dans tous les navigateurs.

# Différences HTML5/prédécesseurs

HTML5 est rétrocompatible

La déclaration de type de document (DTD) est simplifiée à son extrême : <!DOCTYPE html>

Le jeu de balises propre à HTML5 introduit une nouvelle logique de formulation, plus sémantique et plus intuitive : la balisage ne décrit plus le contenu du document. Il détaille sa structure et désigne le rôle de chaque section.

Enfin, les développeurs web peuvent désormais utiliser des API JavaScript gérées nativement par les navigateurs. Le but est d'obtenir des niveaux de performances comparables à ceux des applications Desktop, mais aussi de tirer parti des possibilités offertes par les périphériques mobiles, qui vont prendre une part de plus en plus importante dans le paysage Internet.

# Les bases du HTML5/CSS3

Dans cette partie :

- Déclaration de type de document (DTD)
- Syntaxes HTML et XHTML
- Jeux de caractères
- Définir la langue dans un document HTML5
- Structure d'un document HTML5
- Valider du code HTML5
- Styles CSS dans les balises
- Feuille de styles interne
- Feuille de styles externe
- Éléments, sélecteurs, propriétés et valeurs CSS
- Sélecteurs CSS 2.1
- Sélecteurs CSS3
- id ou class ?
- Pseudo-classes et pseudo-éléments
- Unités CSS

# Déclaration de type de document

La balise <!DOCTYPE> est utile pour que les navigateurs adaptent le rendu d'un document HTML.

La balise <!DOCTYPE> des documents écrits en HTML5 est réduite à sa plus simple expression :

```
<!DOCTYPE html>
```

En HTML4.01, les trois variantes généralement utilisées étaient les suivantes :

- HTML 4.01 strict (mode rigoureux) :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

- HTML 4.01 transitionnel (langage XHTML utilisé) :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

- HTML 4.01 frameset (la page contient des frames) :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
```

# DTD

Voici le squelette d'un document écrit en HTML5 :

```
<!DOCTYPE html>

<html>

  <head>
    </head>

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

# Jeu de caractères

Les *charsets* (jeux de caractères) sont associés à la notion de claviers nationaux. Pour indiquer aux navigateurs dans quel jeu de caractères vous travaillez, vous pouvez insérer une balise dans l'en-tête de votre document :

<meta charset="jeu-à-utiliser">

Vous utiliserez le jeu de caractères :

- **ISO-8859-1** pour accéder directement à la majorité des caractères des langues occidentales (français, anglais, allemand, espagnol, etc.).
- **utf-8** pour afficher sur une même page des caractères issus de plusieurs langues (français et japonais par exemple). Consultez la page [www.columbia.edu/kermit/utf8.html](http://www.columbia.edu/kermit/utf8.html) pour vous rendre compte des immenses possibilités du jeu de caractères utf-8.

# Squelette standard HTML5

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta ...>
    ...
    <meta ...>
    <title>...</title>
    <link ...>
    <style>...</style>
  </head>
  <body>
  </body>
</html>
```

Les éléments meta, s'ils sont présents, permettent, entre autres, d'associer un jeu de caractères, une description et un nom d'auteur au document. Par exemple :

```
<meta charset="utf-8">

<meta name="description" content="Description de la page en quelques phrases.">

<meta name="author" content="Samuel Michaux">
```

link lie le document courant à un autre document (HTML, CSS, image, XML). Par exemple, pour définir une feuille de styles externe :

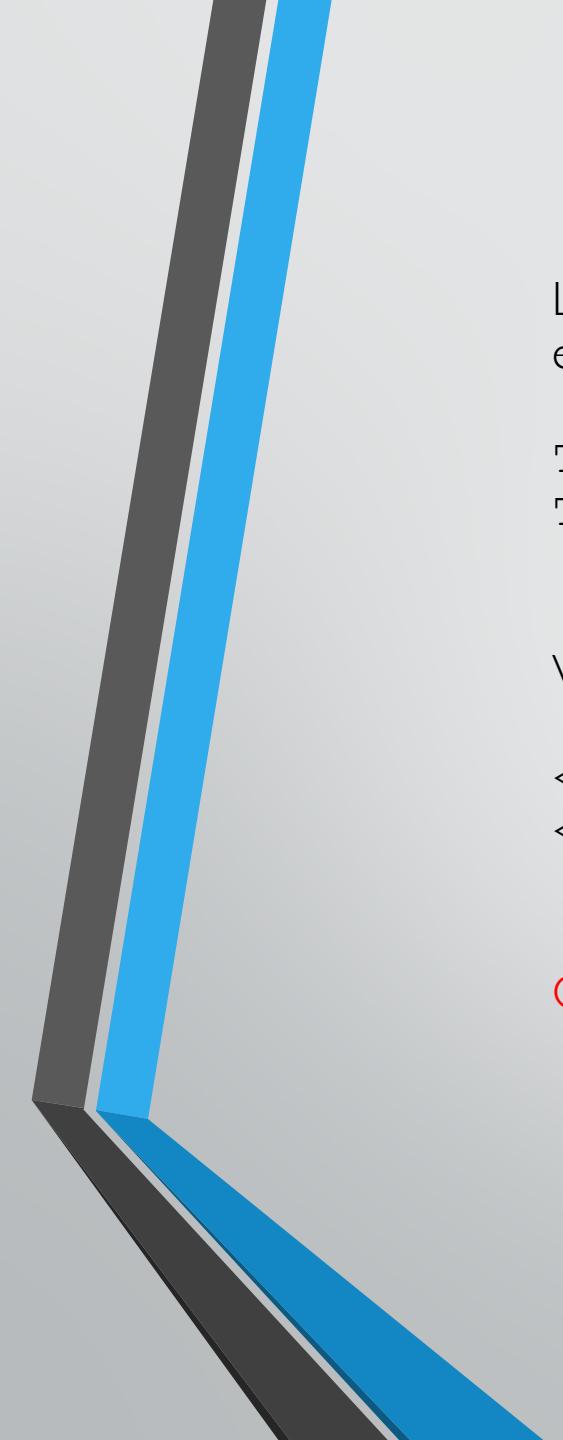
```
<link rel="stylesheet" href="style.css">
```

# Valider le code HTML/XHTML

Avant de tester un code HTML/XHTML dans votre navigateur, je vous conseille de vous assurer de sa conformité. Pour cela, rendez-vous à la page <http://validator.w3.org/>.

Pour faire apparaître du texte dans une page Web, il suffit de le taper tel qu'il doit apparaître, entre les balises <body> et </body> :

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    ...
  </head>
  <body>
    Ce texte apparaît dans le navigateur.
  </body>
</html>
```



Le texte est affiché tel qu'il est saisi dans le navigateur. Si nécessaire, vous pouvez effectuer un passage à la ligne avec la balise `<br>` :

Texte sur la ligne 1  
Texte sur la ligne 2

Vous pouvez aussi définir des paragraphes avec la balise `<p> </p>` :

<p>Ceci est un paragraphe</p>  
<p>Ceci est un autre paragraphe</p>

Quelle est la différence entre `<br>` et `<p>` ?

## ***Des commentaires dans le texte***

Si vous concevez des pages longues et/ou complexes, vous voudrez peut-être y insérer des commentaires pour faciliter leur maintenance. Pour cela, vous utiliserez le marqueur `<!-->` :

```
<!-- Commentaires -->
```

Le texte placé dans le marqueur n'est pas affiché dans le navigateur : il est uniquement visible dans le code source.

# ***Titres et sous-titres***

Vous pouvez affecter jusqu'à six niveaux de titre à vos documents HTML. Pour cela, vous utiliserez les marqueurs <hx> et </hx>, où x est un chiffre compris entre 1 et 6.

## **Exemple :**

Ce petit exemple montre comment utiliser les marqueurs <hx>. Remarquez qu'un marqueur de fin de titre (</hx>) provoque un passage à la ligne. Il est donc inutile de faire appel aux marqueurs <br> ou <p>.

```
<html>
<head>
<title>Les balises de titre</title>
</head>
<body>
<h1>titre de niveau 1</h1>
<h2>titre de niveau 2</h2>
<h3>titre de niveau 3</h3>
<h4>titre de niveau 4</h4>
<h5>titre de niveau 5</h5>
<h6>titre de niveau 6</h6>
texte normal
</body>
</head>
</html>
```

# ***Liens hypertexte***

Une grande part de la magie inhérente au Web est liée à la présence de liens hypertexte dans les pages HTML : en cliquant sur certains mots ou sur certaines images, on peut avoir accès à d'autres parties du document en cours de consultation, à d'autres pages du même site ou d'un autre site, ou encore à d'autres services Internet (mail, ftp, gopher, telnet).

Avant de passer à la pratique, vous devez savoir que l'intérêt d'un site tient à la qualité et à la quantité de ses liens.

Au niveau de la qualité, il est nécessaire de placer les liens à des endroits logiques et de bien spécifier vers quoi ils pointent. N'oubliez pas que les informations qui se trouvent dans vos pages sont consultées on-line. La rapidité d'accès aux données recherchées est donc essentielle.

Au niveau quantité, veillez à ne pas submerger l'utilisateur sous une profusion de liens. Un document aéré est toujours plus facile à lire. Sauf cas particuliers (tableau, sommaire, etc.), limitez-vous à quelques liens par page et révisez-les fréquemment pour que votre site soit toujours proche de l'actualité (c'est un des intérêts majeurs d'Internet).

# ***Lien dans la même page***

Si une de vos pages HTML vous semble un peu longue, vous pouvez définir quelques liens pour en faciliter la lecture.

Pour chaque lien, vous procéderez en deux étapes :

1. Définition d'un signet pour marquer la partie à accéder.
2. Définition d'un lien hypertexte vers ce signet.

Pour placer un signet dans le document courant, vous insérerez un id dans une balise quelconque. Par exemple, une balise `<p></p>` :

```
<p id="Signet">Texte</p>
```

où Signet est le nom du signet et Texte le texte vers lequel vous désirez pointer.

Pour créer un lien hypertexte vers le signet que vous venez de définir, vous utiliserez une autre variante de la balise  :

```
<a href = "#Signet">  
    Texte  
</a>
```

où Signet est le nom de la balise à pointer et Texte le texte qui servira de lien.

## Exercice

Définissez un document qui contient plusieurs paragraphes de texte. Ce texte pourra être pris sur le site <http://fr.lipsum.com/>.

Placez ce texte dans des paragraphes. Définissez des signets pour chacun d'entre eux et des liens hypertextes pour accéder aux différents paragraphes.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Liens et signets</title>
  </head>
  <body>
    <a href="#par1">Paragraphe 1</a> <a href="#par2">Paragraphe 2</a> <a href="#par3">Paragraphe
3</a> <a href="#par4">Paragraphe 4</a> <a href="#par5">Paragraphe 5</a>
    <p id="par1">Paragraphe 1 : Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
    <p id="par2">Paragraphe 2 : Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
    <p id="par3">Paragraphe 3 : Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
    <p id="par4">Paragraphe 4 : Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>
    <p id="par5">Paragraphe 5 : Lorem ipsum dolor sit amet, consectetur adipiscing elit...</p>

  </body>
</html>
```

## ***Lien vers un document local***

Comme nous venons de le voir, les signets facilitent l'accès à un passage déterminé dans un même document. Lorsqu'un lien doit donner accès à des informations qui ne se trouvent pas dans le document courant mais dans un autre document du site, les paramètres de la balise `<a>` doivent être légèrement différents :

```
<a href = "fichier">  
    Texte  
</a>
```

où Fichier est le nom du fichier HTML auquel accéder et Texte le texte qui servira de lien.

## ***Lien vers une partie d'un document local***

Vous savez maintenant définir un lien hypertexte vers un signet du même document et vers un autre document HTML. En combinant les deux syntaxes, il est possible d'accéder à un signet d'un autre document :

```
<a href = "fichier#signet">  
    Texte  
</a>
```

où Fichier est le nom du fichier HTML auquel accéder, Signet le nom du signet dans le fichier HTML, et Texte le texte qui servira de lien.

## **Lien vers un document distant**

Jusqu'à présent, vous avez appris à utiliser un lien hypertexte pour pointer vers un passage du même document, un autre document du site ou un passage d'un autre document du site. En utilisant une syntaxe légèrement différente, il est possible d'accéder à un document d'un autre site :

```
<a href = "https://adresse_url">  
    Texte  
</a>
```

où [http://Adresse\\_URL](#) est l'adresse URL de la page HTML et Texte le texte qui servira de lien.

Un URL (*Uniform Ressource Locator*) est une adresse permettant d'identifier de manière unique un objet Internet. Sa syntaxe "façon UNIX" détermine le nom du serveur, le chemin d'accès dans le serveur et le nom du fichier HTML à accéder : **http://serveur/chemin/document#signet**

## ***Lien vers un fichier***

Pour donner accès à un fichier, il suffit de préciser son nom dans le chemin URL. Par exemple, pour accéder au document ebook.pdf dans le dossier documents du serveur [www.monsite.com](http://www.monsite.com), vous utiliserez ce lien :

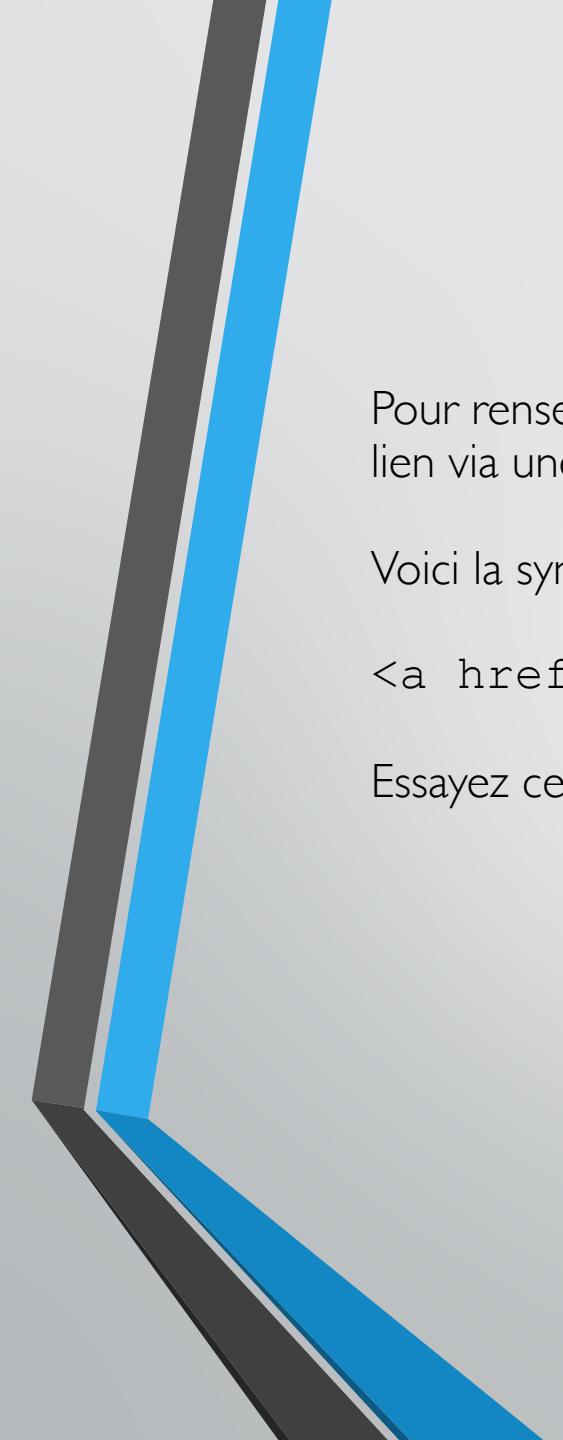
```
<a href="http://www.monsite.com/documents/ebook.pdf">Cliquez ici</a>
```

# **Accéder à un service Internet**

Dans les lignes précédentes, vous avez appris à accéder à une page Web située sur un site distant. Comme le montre ce tableau, il suffit de modifier la première partie de l'adresse pour accéder tout aussi simplement à un autre service Internet.

<b>Service</b>	<b>Marqueur</b>
Site ftp	<a href = "ftp://adresse ">Texte</a>
Serveur e-mail	<a href = "mailto:adresse ">Texte</a>
Gopher	<a href = "gopher://adresse ">Texte</a>
Telnet	<a href = "telnet://adresse ">Texte</a>

Dans chacun des marqueurs précédents, adresse désigne l'adresse URL du service auquel accéder et texte le texte utilisé pour réaliser le lien.



Pour renseigner le visiteur de votre site, vous pouvez ajouter des informations concernant une image ou un lien via une infobulle.

Voici la syntaxe à utiliser pour un lien :

```
<a href="adresse" title="Texte de l'infobulle">Texte du lien</a>
```

Essayez ce code

# Tableaux

Certaines pages Web contiennent des tableaux. Ceux-ci ont été réalisés à l'aide de balises `<table> </table>`.

La syntaxe du marqueur `<table>` est assez complexe :

```
<table  
    [border = "bordure"]  
    [cellpadding = "espace"]  
    [cellspacing = "epaisseur"]  
    [width = "largeur"]  
    [height = "hauteur"]>  
</table>
```

Bordure est un entier qui définit l'épaisseur de la bordure autour du tableau. Si BORDER n'apparaît pas dans le marqueur ou si Bordure vaut 0, le tableau n'a pas de bordure.

Espace définit l'espace entre le contenu des cellules et la bordure gauche de la cellule, en pixels.

Epaisseur définit l'épaisseur du trait entre les cellules, en pixels.

Largeur force la largeur du tableau, en pixels. Si ce paramètre n'est pas spécifié, la largeur du tableau est calculée automatiquement.

Hauteur force la hauteur du tableau, en pixels. Si ce paramètre n'est pas spécifié, la hauteur du tableau est calculée automatiquement.

Vous utiliserez le marqueur <tr> </tr> pour créer chacune des lignes du tableau :

```
<tr  
    [align = alignement]>  
</tr>
```

où alignement définit l'alignement des données sur la ligne. Il peut prendre la valeur left, center ou right selon l'effet recherché. Si ce paramètre n'est pas spécifié, les données sont alignées à gauche.

Enfin, pour définir les caractéristiques d'une cellule, vous utiliserez le marqueur **<td> </td>** dont voici la syntaxe :

```
<td  
    [colspan = "nbcol"]  
    [rowspan = "nblig"]  
    [align = "alignhoriz"]  
    [valign = "alignvert"]  
    [width = "largeur"]>  
élément  
</td>
```

**NbCol** est le nombre de colonnes occupées par la cellule.

**NbLig** est le nombre de lignes occupées par la cellule.

**AlignHoriz** indique l'alignement horizontal du texte dans la cellule. Ce paramètre peut prendre la valeur **LEFT**, **CENTER** ou **RIGHT**. S'il n'est pas spécifié, les données sont alignées à gauche.

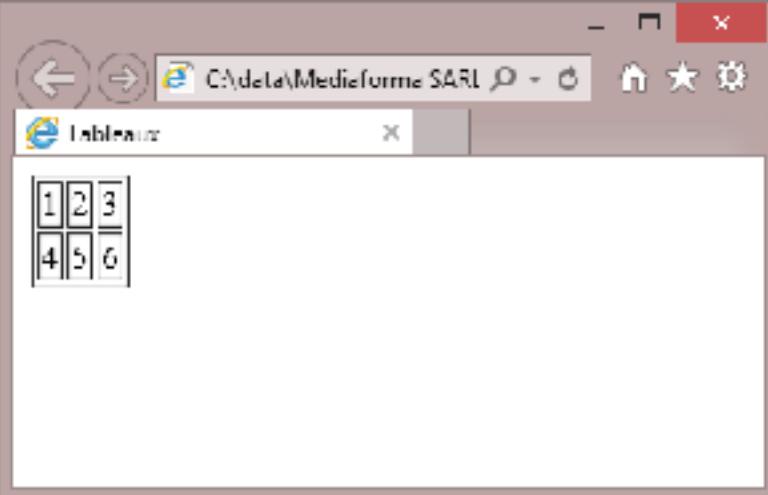
**AlignVert** indique l'alignement vertical du texte dans la cellule. Ce paramètre peut prendre la valeur **TOP**, **MIDDLE** ou **BOTTOM**. S'il n'est pas spécifié, les données sont alignées sur la partie supérieure de la cellule.

**Largeur** indique la largeur de la cellule, en pixels.

**Elément** décrit le contenu de la cellule. Il peut s'agir d'un texte ou d'une image (marqueur **IMG**).

## Exercice 1

Définissez le tableau suivant :



A screenshot of a web browser window titled "Tableaux". The address bar shows the path "C:\data\Medieforme SARL". The main content area displays a 2x3 grid of numbers:

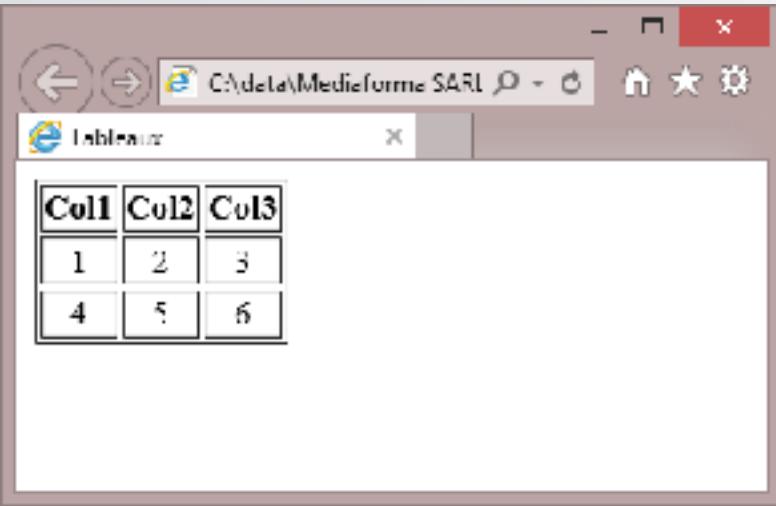
1	2	3
4	5	6

## Solution

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Tableaux</title>
  </head>
  <body>
    <table border="1">
      <tr>
        <td>1</td><td>2</td><td>3</td>
      </tr>
      <tr>
        <td>4</td><td>5</td><td>6</td>
      </tr>
    </table>
  </body>
</html>
```

## Exercice 2

Définissez le tableau suivant :



The screenshot shows a web browser window with the title 'Tableaux'. The address bar displays 'C:\dele\Mediforme SARL'. The main content area shows a 2x3 grid table. The first row contains three columns labeled 'Col1', 'Col2', and 'Col3'. The second row contains three cells with the numbers 1, 2, and 3 respectively. The third row contains three cells with the numbers 4, 5, and 6 respectively.

Col1	Col2	Col3
1	2	3
4	5	6

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Tableaux</title>
  </head>
  <body>
    <table border="1">
      <tr>
        <th>Col1</th><th>Col2</th><th>Col3</th>
      </tr>
      <tr>
        <td align="center">1</td><td align="center">2</td><td align="center">3</td>
      </tr>
      <tr>
        <td align="center">4</td><td align="center">5</td><td align="center">6</td>
      </tr>
    </table>
  </body>
</html>
```

## Exercice 3

Définissez le tableau suivant :

Titre centré sur les 3 colonnes		
Col1	Col2	Col3
1	2	3
4	5	6

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Tableaux</title>
  </head>
  <body>
    <table border="1">
      <tr><th width="500" colspan="3" align="center">Titre centré sur les 3 colonnes</th></tr>
      <tr><th>Col1</th><th>Col2</th><th>Col3</th></tr>
      <tr><td align="center">1</td><td align="center">2</td><td align="center">3</td></tr>
      <tr><td align="center">4</td><td align="center">5</td><td align="center">6</td></tr>
    </table>
  </body>
</html>
```

Exercice : Définissez le tableau suivant

The screenshot shows a web browser window with a title bar reading "C:\data\Mediaforma SARL\Formation\Déjà données\2015-06-08 au 12 HTML5 CSS3 JavaScript Xerox Clermont Fd - 0". The main content area displays a table with the following data:

	En millions d'Euros	2015 en €/hab.
	2014	2015
Ile-de-france	399258	415920
Midi-pyrénées	73961	65826
Centre	43638	55802

```
<center>
    Recettes budgétaires par région
    <table border>
        <tr>
            <th></th>
            <th colspan="2">En millions d'Euros</th>
            <th rowspan="2">2015 en €/hab.</th>
        </tr>
        <tr>
            <th></th>
            <th>2014</th>
            <th>2015</th>
        </tr>
        <tr align="center">
            <td>Ile-de-france</td>
            <td>399258</td>
            <td>415920</td>
            <td>178,8</td>
        </tr>
        <tr align="center">
            <td>Midi-pyrénées</td>
            <td>73961</td>
            <td>63826</td>
            <td>219,4</td>
        </tr>
        <tr align="center">
            <td>Centre</td>
            <td>43638</td>
            <td>55802</td>
            <td>185,9</td>
        </tr>
    </table>
</center>
```

# ***Insertion d'une image dans un document HTML***

L'insertion d'une image dans un document HTML s'effectue en trois étapes :

1. Définition ou choix de l'image.
2. Sauvegarde au format GIF (.GIF), JPEG (.JPG) ou PNG (.PNG) et stockage dans le même dossier que le fichier HTML.
3. Insertion d'une balise <img> dans le document HTML.

## Avantages et inconvénients de ces trois types d'images

GIF : limité à 256 couleurs, peut posséder une couleur de transparence, non compressé

JPG : 16 millions de couleurs, pas de couleur de transparence, compressé

PNG : 16 millions de couleurs, peut posséder une couleur de transparence, non compressé

Téléchargez l'application PhotoFiltre 7 sur  
<http://www.photofiltre-studio.com/news.htm>

Entraînez-vous à créer des images au format GIF, JPG (avec plusieurs formats de compression) et PNG et comparez leurs tailles en octets

Lorsque l'image a été sauvegardée au bon format, insérez un marqueur <IMG> dans la page HTML, à l'endroit où vous désirez l'afficher. La syntaxe du marqueur est la suivante :

```
<img  
    src = "nom de l'image"  
    [align = {top|bottom|middle|left|right} ]  
    [border = epaisseur]  
    [alt = "texte"]>
```

1. Nom de l'image est le nom complet de l'image (nom du fichier et extension).
2. ALIGN définit l'alignement de l'image par rapport au texte : TOP, BOTTOM et MIDDLE alignent le texte respectivement par rapport à la partie supérieure, à la partie inférieure et au centre de l'image ; LEFT et RIGHT alignent l'image respectivement à gauche et à droite du texte.
3. Epaisseur définit l'épaisseur du cadre autour de l'image, en pixels.
4. Texte est un court texte qui apparaît à la place de l'image lorsque l'affichage des images a été désactivé.



Entraînez-vous à insérer les images GIF, JPG et PNG définies précédemment et comparez leur rendu.

## Rendre une image cliquable

Pour insérer un lien sur une image, il suffit de l'entourer d'une balise `<a></a>` :

```
<a href="http://www.site.com"></a>
```

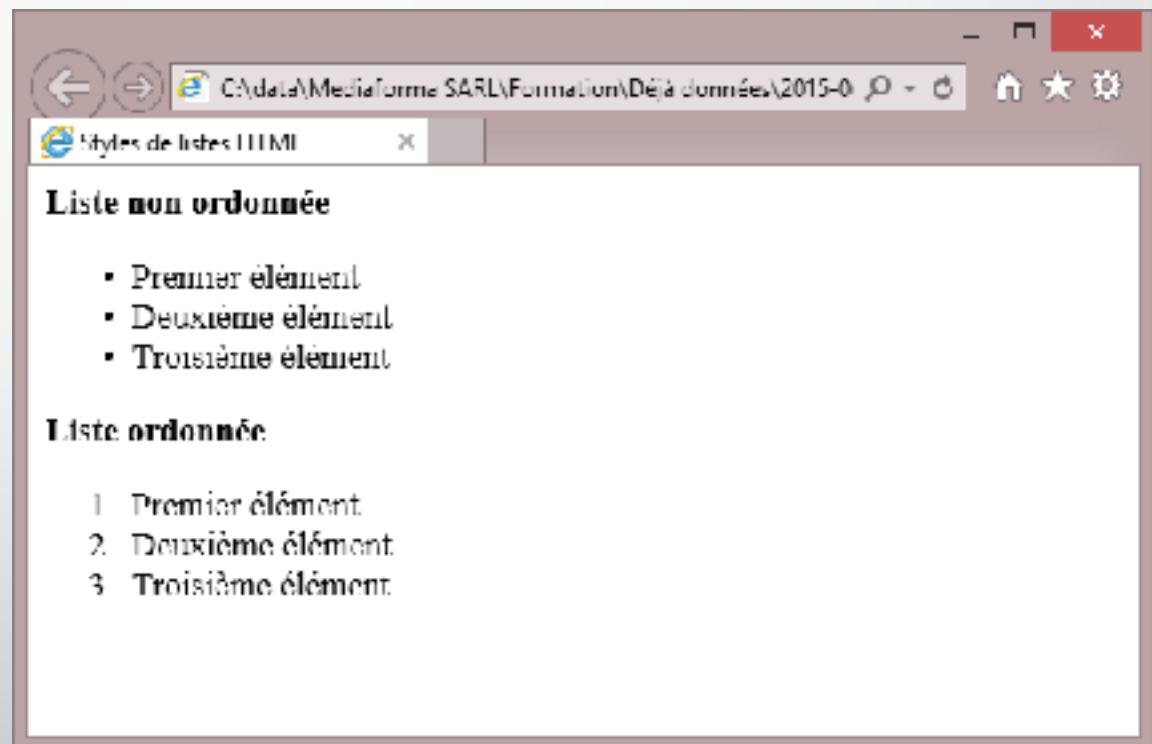
# Listes

Le langage HTML compte trois types de listes :

- non ordonnées ou listes à puces : éléments ul et li ;
- ordonnées : éléments ol et li ;

Cet exemple illustre ces deux types de listes :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Styles de listes HTML</title>
  </head>
  <body>
    <b>Liste non ordonnée</b>
    <ul>
      <li>Premier élément</li>
      <li>Deuxième élément</li>
      <li>Troisième élément</li>
    </ul>
    <b>Liste ordonnée</b>
    <ol>
      <li>Premier élément</li>
      <li>Deuxième élément</li>
      <li>Troisième élément</li>
    </ol>
  </body>
</html>
```



# Formulaires

Si la plupart des pages Web se contentent de diffuser des informations, elles sont également en mesure d'en recueillir par l'intermédiaire de formulaires. Les données sont stockées sur le serveur. Elles peuvent être traitées à l'aide d'un programme spécifique écrit par vos soins, fourni par le prestataire ou tout simplement expédiées sans traitement dans votre boîte aux lettres.

Pour définir un formulaire, vous utiliserez le marqueur <form> </form> dont voici la syntaxe :

```
<form  
    action = "traitement"  
    method = "{get|post}">  
</form>
```

1. traitement est le nom d'un programme de traitement, par exemple <http://www.site.com/traitement.php>, ou encore l'adresse de votre boîte aux lettres, précédée de mailto. Par exemple. <mailto:pierre.durand@Bertold.fr>.
2. method définit la méthode à utiliser pour prendre en charge les données du formulaire. La méthode GET étant limitée quant à la quantité de données transmissibles, la méthode POST est presque toujours utilisée.

Entre les balises <form> et </form>, vous utiliserez des balises <input> pour définir les données à saisir :

```
<input  
[type = {"text|password|checkbox|radio|submit|reset|hidden|image"}]  
name = "nom"  
[value = "valeur"]  
[size = "taille1[,taille2]"]  
[maxlength = "longueur"]  
[checked]>
```

Le paramètre optionnel type définit le type de l'objet :

Valeur	Effet
<b>text</b>	Champ de saisie
<b>password</b>	Mot de passe
<b>checkbox</b>	Case à cocher
<b>radio</b>	Bouton radio
<b>submit</b>	Bouton d'envoi, pour envoyer les données du formulaire au serveur
<b>reset</b>	Bouton de réinitialisation du formulaire
<b>hidden</b>	Champ caché qui n'apparaît pas sur le formulaire mais qui est envoyé lors de l'appui sur le bouton SUBMIT ou sur une image cliquable

Exercice :

Définissez un formulaire pour obtenir le résultat suivant :

The screenshot shows a Windows application window titled "Un premier formulaire". The window contains the following elements:

- A toolbar at the top with icons for back, forward, stop, and refresh.
- A title bar with the window title and standard window controls (minimize, maximize, close).
- A main content area with the following fields:
  - "Entrez votre nom" followed by a text input field.
  - "Entrez votre adresse e mail" followed by a text input field.
  - A question "Etes-vous client de la fnac ?" with two radio button options: "OUI" (selected) and "NON".
- At the bottom are two buttons: "Envoyer" and "Annuler".

## Solution

```
<!DOCTYPE html>
<html>
  <head>
    <title>Un premier formulaire</title>
  </head>
  <body>
    <form action = "mailto:bertrand.oscar@oracle.fr" method = "post">
      Entrez votre nom <input type = "text" name = nom><br>
      Entrez votre adresse e-mail <input type = "text" name = nom><br>
      Etes-vous client de la fnac ?
      OUI <input type = "radio" name = "client" value = "oui" checked>
      NON <input type = "radio" name = "client" value = "non"><br>
      <input type = "submit" value = "Envoyer">
      <input type = "reset" value = "Annuler">
    </form>
  </body>
</html>
```

## Exercice

Utilisez un tableau pour améliorer le formulaire comme ceci :

The screenshot shows a Windows application window titled "Un premier formulaire". The window contains the following elements:

- A toolbar at the top with standard icons for back, forward, stop, and refresh.
- A title bar with the window title and system controls.
- A main content area with the following fields:
  - "Entrez votre nom" followed by a text input field.
  - "Entrez votre adresse e-mail" followed by a text input field.
  - A question "Etes-vous client de la fnac ?" followed by two radio buttons labeled "OUI" and "NON".
- Buttons at the bottom: "Envoyer" on the left and "Annuler" on the right.

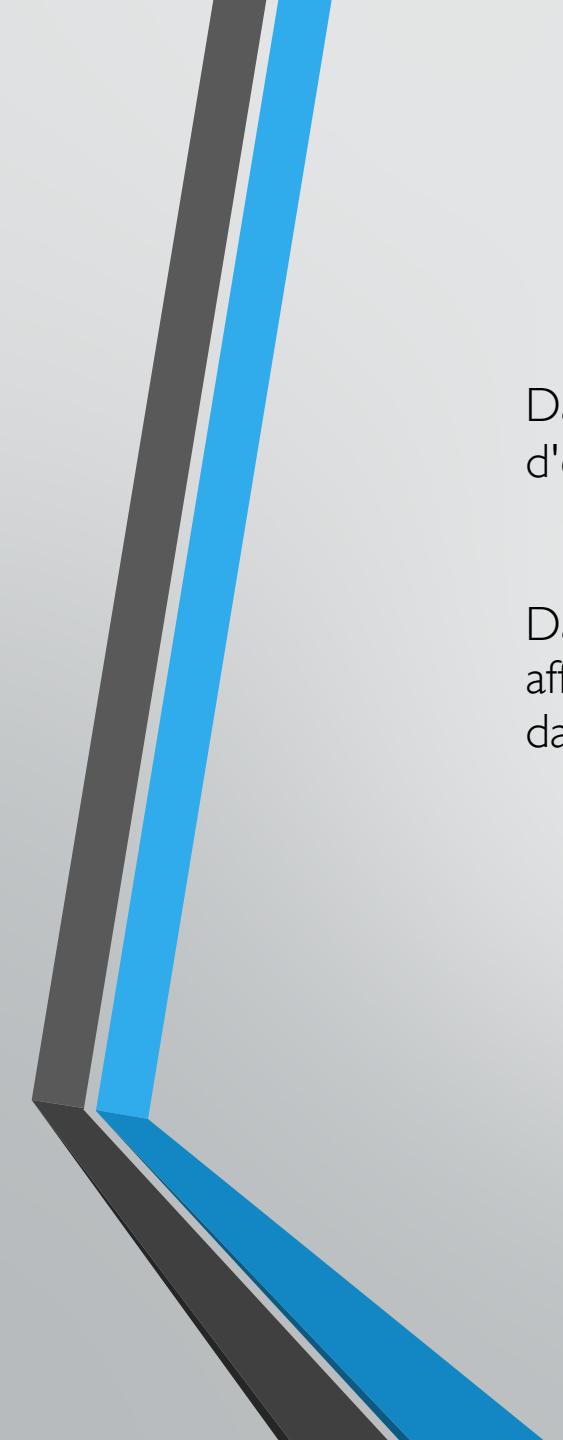
## Solution

```
<!DOCTYPE html>
<html>
  <head>
    <title>Un premier formulaire</title>
  </head>
  <body>
    <form action = "mailto:bertrand.oscar@oracle.fr" method = "post">
      <table>
        <tr><td>Entrez votre nom</td><td><input type = "text" name="nom"></td></tr>
        <tr><td>Entrez votre adresse e-mail</td><td><input type = "text" name="email"></td></tr>
        <tr><td>Etes-vous client de la fnac ?</td>
          <td>OUI <input type = "radio" name = "client" value = "oui" checked>
          NON <input type = "radio" name = "client" value = "non"></td></tr>
        <tr><td><input type = "submit" value = "Envoyer"></td>
          <td><input type = "reset" value = "Annuler"></td></tr>
      </table>
    </form>
  </body>
</html>
```

## Une liste déroulante dans un formulaire

Pour définir une liste déroulante, vous utiliserez la balise <select>. Chacun des éléments de la liste sera placé dans une balise enfant <option>

```
<form>
  <select name="jour">
    <option>Lundi</option>
    <option>Mardi</option>
    <option>Mercredi</option>
    <option>Jeudi</option>
    <option>Vendredi</option>
  </select>
</form>
```



Dans la balise **<select>**, vous pouvez définir l'attribut **size** pour indiquer combien d'éléments sont visibles dans la liste (par défaut, **size** vaut **1**).

Dans l'une des balises **<option>** vous pouvez définir l'attribut **selected**, sans lui affecter aucune valeur. L'élément correspondant est alors sélectionné par défaut dans la liste.

Outre les boutons de formulaires **submit** et **reset** :

```
<input type="submit">  
<input type="reset">
```

Vous pouvez utiliser de simples boutons à l'intérieur ou à l'extérieur d'un formulaire avec la balise **<button>** :

```
<button>Texte</button>
```

## Découpage d'une page

Pour découper une page en blocs, vous utiliserez les balises `<div>` et `</div>`.

Ces balises sont de type **block**. Elles s'affichent automatiquement l'une sous l'autre, sans qu'il soit nécessaire d'utiliser une balise `<br>` pour passer à la ligne ou de les insérer dans une balise `<p> </p>`.

Les balises `<div>` peuvent être personnalisées en utilisant des instructions CSS. Nous étudierons ces instructions dans la suite de la formation.

# Mise en forme en CSS

Dans cette partie :

- Styles dans les balises, dans une feuille de styles interne ou externe
- Sélecteurs, propriétés et valeurs
- Sélecteurs CSS 2.1
- id ou class ?
- Sélecteurs CSS 3
- Pseudo-classes et pseudo-éléments
- Unités CSS



Vous utiliserez le langage CSS pour mettre en forme les balises d'un ou de plusieurs documents, à travers des attributs ou des feuilles de styles.

Les prochaines diapositives vont vous montrer comment utiliser ce langage.

# Styles CSS dans les balises

style=""

où :

- balise est un nom de balise : <p> ou <h1> par exemple.
- propriété1 sont des propriétés de style de la balise.
- valeur1 sont les valeurs affectées aux propriétés.

Cette technique n'est pas optimale, car limitée au seul élément qui l'utilise. Dans la mesure du possible, privilégiez l'utilisation d'une feuille de styles externe, d'extension .css et reliée au document à l'aide d'un élément link. Cette même feuille de styles pourra être utilisée dans tous les documents apparentés pour uniformiser leur apparence.

# Styles CSS dans les balises

À titre d'exemple, pour affecter la couleur jaune à l'arrière-plan d'un élément p, vous utiliserez le code suivant :

```
<p style="background: yellow;">  
Ce texte a un arrière-plan jaune  
</p>
```

Ou encore, pour utiliser la police Verdana corps 18 dans un titre h2, vous utiliserez le code suivant :

```
<h2 style="font-family: Verdana; font-size: 18px;">  
Ce titre est en Verdana corps 18  
</h2>
```

Essayez ces deux codes

# Feuille de styles interne

Pour étendre la portée des définitions CSS, vous pouvez les regrouper dans l'en-tête du document HTML.

Voici la syntaxe permettant d'affecter des propriétés à une balise :

élément {propriété1:valeur1; ... propriétéN:valeurN}

où :

- élément est un nom d'élément : p ou h1 par exemple.
- propriété1 sont des propriétés de style de l'élément.
- valeur1 sont les valeurs affectées aux propriétés.

Ces lignes de code doivent être insérées entre les balises <style> et </style>, à l'intérieur de l'élément head.

# Exercice

Ecrivez une feuille de styles interne pour :

- Définir dans tout le document un arrière-plan de couleur jaune pour les éléments `p`
- Utiliser la police Verdana corps 18 dans tous les titres `h2`

# Solution

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Une feuille de styles interne</title>
    <style>
      p {background-color:yellow; }
      h2 {font-family:Verdana; font-size: 18px; }
    </style>
  </head>
  <body>
    ...
  </body>
</html>
```

# Feuille de styles externe

Pour augmenter encore le champ d'action des styles définis dans l'en-tête d'un document, vous pouvez les stocker dans un fichier CSS. Dans ce cas, plusieurs documents HTML peuvent référencer cette "feuille de styles" (le fichier CSS) pour utiliser les styles qui y sont définis.

L'opération se fait avec un élément link (ici, la feuille de styles a pour nom *moncss.css*) :

```
<link rel="stylesheet" href="moncss.css">
```

# Exercice

Définissez une feuille de styles externe qui reprend les règles définies dans l'exercice précédent et faites référence à cette feuille de styles dans un document HTML.

# Solution

```
p {background-color:yellow;}  
h2 {font-family:Verdana; font-size: 18px;}
```

La feuille de styles  
moncss.css

```
<!DOCTYPE html>  
<html>  
<head>  
  <meta charset="utf-8">  
  <title>Une feuille de styles externe</title>  
  <link rel="stylesheet" href="moncss.css">  
</head>  
<body>  
  <!-- Les instructions HTML5 peuvent utiliser les styles -->  
  <!-- définis dans la feuille de styles moncss.css -->  
</body>  
</html>
```

Le document HTML

# Sélecteurs, propriétés et valeurs CSS

La syntaxe à utiliser est la suivante :

```
sélecteur {propriété1: valeur1; ... propriétéN: valeurN;}  
élément {propriété1: valeur1; ... propriétéN: valeurN;}
```

Pour définir le style d'un élément, il suffit d'entrer le nom de l'élément, d'ouvrir des accolades, d'énumérer des propriétés et de leur affecter les valeurs souhaitées.

# Exercice

Définissez un document HTML5 qui contient une liste à puces.

Définissez une feuille de styles interne qui donne les caractéristiques de la liste à puces :

- Couleur : blue
- Arrière-plan : #FF4040
- Marges : 14px
- Style des puces : square

# Solution

code001.htm

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Style des li</title>
    <style>
      li
      {
        color: blue;
        background: #FF4040;
        margin: 14px 14px 14px 14px;
        list-style: square;
      }
    </style>
  </head>
  <body>
    Une liste à puces :
    <ul>
      <li>Premier</li>
      <li>Deuxième</li>
      <li>Troisième</li>
      <li>Quatrième</li>
    </ul>
  </body>
</html>
```

# Sélecteurs CSS 2.1

Le sélecteur universel étoile (\*) s'adresse à tous les éléments. Vous l'utiliserez essentiellement pour modifier le style de tous les éléments de la page. Par exemple, la police.

```
* { font-family: Verdana; }
```

Le sélecteur de type permet de changer le style d'un élément. Ici, par exemple, le style de tous les éléments h1 :

```
h1 { font-size: 120%; }
```

Le sélecteur de classe est représenté par un point (.). Il cible les éléments en fonction de la valeur de leur attribut class. Par exemple, le style ci-après s'applique à tous les éléments p dont l'attribut class vaut info.

```
p.info { background: yellow; }
```

Si aucun élément ne restreint le sélecteur de classe, tous les éléments dont l'attribut class a la valeur spécifiée sont concernés. Ici, par exemple, il s'agira de tous les éléments dont l'attribut class vaut info :

```
.info { background: yellow; }
```

# Sélecteurs CSS 2.1

Il est possible d'affecter plusieurs classes à un élément (l'attribut class peut recevoir plusieurs valeurs séparées par des espaces). On parle alors de "sélecteur de classes multiples". Ici par exemple, seuls les éléments p dont l'attribut class vaut info et premier sont affectés :

```
p.info.premier {background: yellow;}
```

Voici un exemple d'élément p concerné par cette règle CSS :

```
<p class="info premier">texte</p>
```

# Sélecteurs CSS 2.1

Le sélecteur d'identificateur est représenté par le caractère dièse (#). Il cible l'élément dont l'attribut id a la valeur spécifiée. Ici, seul l'élément dont la propriété id vaut corne est concerné :

```
#corne {font-family: Serif;}
```

Vous pouvez restreindre le ciblage en précisant un élément devant le sélecteur d'id. Ici, par exemple, seul l'élément p dont le marqueur id vaut corne est concerné :

```
p#corne {font-family: Serif;}
```

Un sélecteur descendant est construit avec deux sélecteurs traditionnels (ou plus) séparés par un espace. Il cible les enfants du premier sélecteur. Par exemple, pour affecter un arrière-plan jaune à tous les éléments li enfants d'un élément div, vous utiliserez le sélecteur descendant ci-après :

```
div li {color: yellow;}
```

# Sélecteurs CSS 2.1

Un sélecteur d'attribut cible les éléments en fonction de la présence d'attributs et/ou de leurs valeurs. Par exemple, pour affecter une taille de 120% à tous les éléments qui possèdent un attribut perso, vous utiliserez le sélecteur suivant :

```
[perso] {font-size: 120%;}
```

Pour cibler les seuls éléments dont l'attribut perso a pour valeur vert, vous utiliserez le sélecteur suivant :

```
[perso=vert] {font-size: 120%;}
```

# Sélecteurs CSS 2.1

Enfin, pour sélectionner les éléments **el2** immédiatement précédés par une élément **el1**, vous utiliserez le sélecteur suivant :

`el1+el2`

# Exercice

- 1) Affectez la couleur rouge aux balises p de classe rouge
- 2) Affectez la police Courrier aux balises div de classe code ou exemple
- 3) Affectez la couleur d'arrière-plan jaune aux balises <li> contenues dans une balise <ol>

Voici le code HTML sur lequel appliquer ces règles :

```
<body>
    <p class="rouge">Ceci est un paragraphe de classe rouge</p>
    <p class="vert">Ceci est un paragraphe de classe vert</p>
    <p class="noir">Ceci est un paragraphe de classe noir</p>
    <div class="code">Ce texte est affiché dans une balise div de classe code</div>
    <div>Ce texte est affiché dans une balise div sans classe</div>
    <div class="exemple">Ce texte est affiché dans une balise div de classe exemple</div>
    Une liste UL
    <ul>
        <li>Premier</li>
        <li>Deuxième</li>
        <li>Troisième</li>
    </ul>
    Une liste OL
    <ol>
        <li>Premier</li>
        <li>Deuxième</li>
        <li>Troisième</li>
    </ol>
</body>
```

# Exercice

Et voici le résultat à obtenir :

Ceci est un paragraphe de classe rouge

Ceci est un paragraphe de classe vert.

Ceci est un paragraphe de classe noir

Ce texte est affiché dans une balise div de classe code

Ce texte est affiché dans une balise div sans classe

Ce texte est affiché dans une balise div de classe exemple

Une liste UL.

- Premier
- Deuxième
- Troisième

Une liste OL

1. Premier
2. Deuxième
3. Troisième

# Solution

code002.htm

```
<style>
    p.rouge {color: red; }
    div.code {font-family:Courier New; }
    div.exemple {font-family:Courier New; }
    ol li {background: yellow; }
</style>
```

# id ou class ?

L'attribut id est généralement utilisé :

- Sur les éléments qui structurent le document et qui sont désignés comme uniques : par exemple, les éléments header, container, content, nav, footer, etc. Une fois ces éléments identifiés avec l'attribut id, ils peuvent être stylés avec des propriétés CSS.
- Sur d'autres éléments qui doivent être accessibles en JavaScript.
- Pour accéder à une ancre.

```
<header id="hperso">  
  
<nav id="nperso">  
  
<p id="position1">, puis <a href="#position1">Aller au repère 1</a>
```

L'attribut class est utilisé dans tous les autres cas.

```
<div class="perso">  
  
<p class="perso">
```

# Sélecteurs CSS3

Tout ce qui a été dit sur les sélecteurs à la section précédente reste valable en CSS3. Cependant, de nouveaux sélecteurs sont maintenant disponibles, qui permettent de cibler encore plus précisément les éléments auxquels vous vous adressez.

# Sélecteurs CSS3

Syntaxe	Signification
<code>nom élément[attr^="valeur"]</code>	Tout élément <u>nom élément</u> dont la valeur de l'attribut <u>attr</u> commence exactement par la chaîne <u>valeur</u>
<code>nom élément[attr\$="valeur"]</code>	Tout élément <u>nom élément</u> dont la valeur de l'attribut <u>attr</u> finit exactement par la chaîne <u>valeur</u>
<code>nom élément[attr*="valeur"]</code>	Tout élément <u>nom élément</u> dont la valeur de l'attribut <u>attr</u> contient la sous-chaîne <u>valeur</u>
<code>nom élément:root</code>	Un élément <u>nom élément</u> , racine du document
<code>nom élément:nth-child(n)</code>	Un élément <u>nom élément</u> qui est le n-ième enfant de son parent
<code>nom élément:nth-last-child(n)</code>	Un élément <u>nom élément</u> qui est le n-ième enfant de son parent en comptant depuis le dernier enfant
<code>nom élément:nth-of-type(n)</code>	Un élément <u>nom élément</u> qui est le n-ième enfant de son parent et de ce type
<code>nom élément:nth-last-of-type(n)</code>	Un élément <u>nom élément</u> qui est le nième enfant de son parent et de ce type en comptant depuis le dernier enfant
<code>nom élément:last-child</code>	Un élément <u>nom élément</u> , dernier enfant de son parent
<code>nom élément:first-of-type</code>	Un élément <u>nom élément</u> , premier enfant de son type

# Sélecteurs CSS3

Syntaxe	Signification
<code>nom élément:last-of-type</code>	Un élément <u>nom élément</u> , dernier enfant de son type
<code>nom élément:only-child</code>	Un élément <u>nom élément</u> , seul enfant de son parent
<code>nom élément:only-of-type</code>	Un élément <u>nom élément</u> , seul enfant de son type
<code>nom élément:empty</code>	Un élément <u>nom élément</u> qui n'a aucun enfant
<code>nom élément:target</code>	Un élément <u>nom élément</u> qui est la cible de l'URL d'origine
<code>nom élément:enabled</code>	Un élément d'interface utilisateur <u>nom élément</u> qui est actif ou inactif
<code>nom élément:checked</code>	Un élément d'interface utilisateur <u>nom élément</u> qui est coché ou dont l'état est indéterminé (bouton radio ou case à cocher par exemple)
<code>nom élément:contains("attr")</code>	Un élément <u>nom élément</u> dont le contenu textuel concaténé contient la sous-chaîne <u>attr</u>
<code>nom élément:selection</code>	La partie d'un élément <u>nom élément</u> qui est actuellement sélectionnée par l'utilisateur
<code>nom élément:not(sel)</code>	Un élément <u>nom élément</u> qui n'est pas représenté par le sélecteur simple <u>sel</u>
<code>nom élément ~ F</code>	Un élément <u>F</u> précédé par un élément <u>nom élément</u>

# Pseudo-classes, pseudo-éléments

Le sélecteur de pseudo-classe est représenté par le caractère deux-points (`:`).

Vous ferez appel aux pseudo-classes pour cibler des éléments en fonction de caractéristiques inaccessibles aux sélecteurs traditionnels : premier enfant ou focus par exemple.

La pseudo-classe `:first-child` permet de cibler le premier enfant d'un élément. Par exemple, pour mettre en gras le premier élément `p` enfant de l'élément `div`, vous utiliserez le sélecteur ci-après :

```
div p:first-child {font-weight: bold;}
```

Les pseudo-classes `:link` et `:visited` ciblent les éléments `a` dont (respectivement) le lien n'a pas été visité/a été visité. Les deux lignes suivantes définissent la couleur des liens :

```
:link {color: fuchsia;}
```

```
:visited {color: navy;}
```

# Pseudo-classes, pseudo-éléments

La pseudo-classe `:focus` cible les éléments qui ont le focus (par défaut, les liens et les éléments de formulaires). Elle permet très simplement de modifier la couleur d'arrière-plan (ou un autre style) d'un élément. Ici, par exemple, nous affectons un arrière-plan rouge à l'élément `input` de type `text` qui a le focus :

```
input[type=text]:focus {background: lightgrey;}
```

La pseudo-classe `:hover` cible les éléments dont le contenu est survolé. Cela permet par exemple de changer la bordure d'une image lorsqu'elle est pointée par la souris :

```
img:hover {border: black 5px solid;}
```

# Pseudo-classes, pseudo-éléments

CSS3 s'enrichit de plusieurs pseudo-classes :

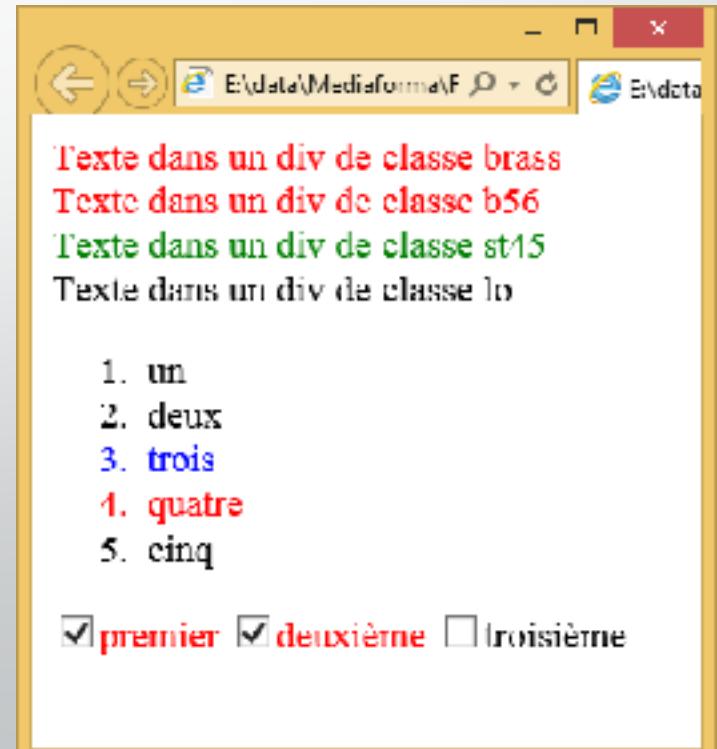
Pseudo-classe	Signification
<u>:nth-child(position)</u>	Éléments repérés par leur position dans le parent
<u>:nth-last-child(position)</u>	Dernier élément du parent
<u>:type:nth-of-type(position)</u>	Éléments du type spécifié repérés par leur position dans le parent
<u>:type:nth-last-of-type(N)</u>	Dernier élément du type spécifié du parent
<u>:last-child</u>	Dernier élément du parent
<u>:type:first-of-type</u>	Premier élément du type spécifié
<u>:last-of-type</u>	Dernier élément du type spécifié
<u>:only-child</u>	Élément qui est l'enfant unique du parent
<u>:type:only-of-type</u>	Élément qui est l'enfant unique du type spécifié du parent
<u>:root</u>	Éléments à la racine du document
<u>:empty</u>	Éléments sans enfant
<u>:target</u>	Élément dont l'identifiant est l'ancre spécifiée dans l'URL de la page
<u>:enabled</u>	Éléments validés (enabled)
<u>:disabled</u>	Éléments dévalidés (disabled)
<u>:checked</u>	Éléments (cases à cocher ou boutons radio) cochés
<u>:not(Sélecteur)</u>	Éléments qui sont différents du sélecteur

# Exercice

Quelle syntaxe utiliser pour s'adresser :

- 1) Aux éléments div dont l'attribut class commence par "br"
- 2) Aux éléments div dont l'attribut class contient la lettre "t"
- 3) L'avant-dernier élément li enfant de ol
- 4) Le troisième enfant li de l'élément ol de classe "t"
- 5) Les labels associés aux éléments checkbox qui sont cochés

Définissez le code HTML5 et CSS3 nécessaire pour obtenir ce résultat :



# Solution

- 1) div[class^="br"]
- 2) div[class\*="t"]
- 3) ol li:nth-last-child(2)
- 4) ol li:nth-child(3)
- 5) input[type=checkbox]:checked+label

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <div class="brass">Texte dans un div de classe brass</div>
    <div class="b56">Texte dans un div de classe b56</div>
    <div class="st45">Texte dans un div de classe st45</div>
    <div class="lo">Texte dans un div de classe lo</div>
    <ol class="t">
      <li>un</li>
      <li>deux</li>
      <li>trois</li>
      <li>quatre</li>
      <li>cinq</li>
    </ol>
    <form>
      <input type="checkbox" checked><label>premier</label>
      <input type="checkbox" checked><label>deuxième</label>
      <input type="checkbox"><label>troisième</label>
    </form>
  </body>
</html>
```

# Solution

```
div[class^="br"] {color: red;}  
div[class*= "t"] {color: green;}  
ol li:nth-last-child(2) {color: red;}  
ol li:nth-child(3) {color: blue;}  
input[type=checkbox]:checked+label {color:red; }
```

# Unités CSS

Voici la liste des unités exprimées de façon absolue :

- cm : centimètre ;
- in : pouce (2,54 cm) ;
- mm : millimètre ;
- pt : point (1/62 inch) ;
- pc : pica (12 points).

À titre d'information, 1 in = 2,54 cm = 25,4 mm = 72 pt = 6 pc.

# Unités CSS

Il existe également des unités exprimées de façon relative, qui dépendent de leur élément parent :

- em : cadratin. Un em correspond grossièrement à la hauteur de la lettre "C" de l'élément parent (taille du texte définie dans la propriété font-size).
- ex : fraction de la hauteur des caractères. Dans les polices occidentales, ex correspond à la hauteur de la lettre minuscule "x".
- px : pixel. Cette unité dépend de la résolution du périphérique d'affichage.
- % : pourcentage de la taille de l'élément ou de son parent.

L'unité em n'est pas à exclure. Elle permet en effet aux utilisateurs finaux de redimensionner le texte proportionnellement à la taille de police, tout en conservant la mise en page intacte. C'est la raison pour laquelle je ne saurais trop vous recommander de l'utiliser !

# Ossature d'un document HTML 5

Dans cette partie :

- Nouvelle organisation des documents HTML
- Sections et niveaux de titre
- hgroup
- address

# Organisation des documents HTML5

Bien souvent, les pages web sont constituées d'un en-tête et d'un pied de page, d'un menu, d'une zone réservée au contenu (article, images, vidéos, etc.) et, éventuellement, d'une zone annexe, n'ayant aucun rapport direct avec le contenu de la page.

Le langage HTML5 a de nouvelles balises pour répondre à ces besoins.

# HTML5

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Un document HTML5 avec en-tête, bas de page, menu, zone annexe et Zone de contenu</title>
    <link rel="stylesheet" href="styles004.css">
    <script src="Creation-elements-HTML5.js"></script>
  </head>
  <body>
    <header>
      <h1>En-tête du document</h1>
      Texte de l'en-tête
    </header>
    <nav>
      <h2>Menu</h2>
      <ul>
        <li><a href="page1.htm">Page 1</a></li>
        <li><a href="page2.htm">Page 2</a></li>
        <li><a href="page3.htm">Page 3</a></li>
      </ul>
      <br><br><br>
    </nav>
    <aside>
      Texte affiché dans la partie droite de la page avec la balise &lt;aside&gt;<br><br>
    </aside>
    <article>
      <h2>Premier article</h2>
      <p>Texte du premier article</p>
    </article>
    <article>
      <h2>Deuxième article</h2>
      <p>Texte du deuxième article</p>
    </article>
    <footer>
      <p>Copyright et e-mail du Webmaster</p>
    </footer>
  </body>
</html>
```

Les balises <div> n'ont pas disparu du langage HTML5. Elles sont simplement moins souvent utilisées, car plusieurs nouvelles balises plus spécialisées viennent lui prêter main-forte.

Saisissez ce code et sauvegardez-le sous le nom code004.htm

## CSS3

Saisissez ce code et sauvegardez-le sous le nom styles004.htm

```
header, nav, article, section, footer, aside
{
    display: block;
}
```

Le rendu block (`display: block;`) est affecté aux nouveaux éléments HTML5 pour assurer le support de ces nouveaux éléments dans les navigateurs peu ou pas compatibles.

```
header
{
    background-color: red;
}
```

header défini l'en-tête d'une section ou d'une page web.

```
nav
{
    float:left;
    width:20%;
    background-color:yellow;
}
```

nav est destiné à contenir des liens de navigation dans le site.

```
article
{
    background-color: #66FFFF;
    width:80%;
}
```

article correspond à une entité autonome du document : par exemple, un article (ou un ensemble d'articles) extrait d'un blog ou d'un forum.

```
aside
{
    float:right;
    width:20%;
    background-color:yellow;
}
```

aside définit un contenu qui n'a pas de rapport direct avec la page. Il peut être utilisé pour définir un menu ou pour donner accès aux archives du site.

```
footer
{
    clear:both;
    background-color: #99FF66;
}
```

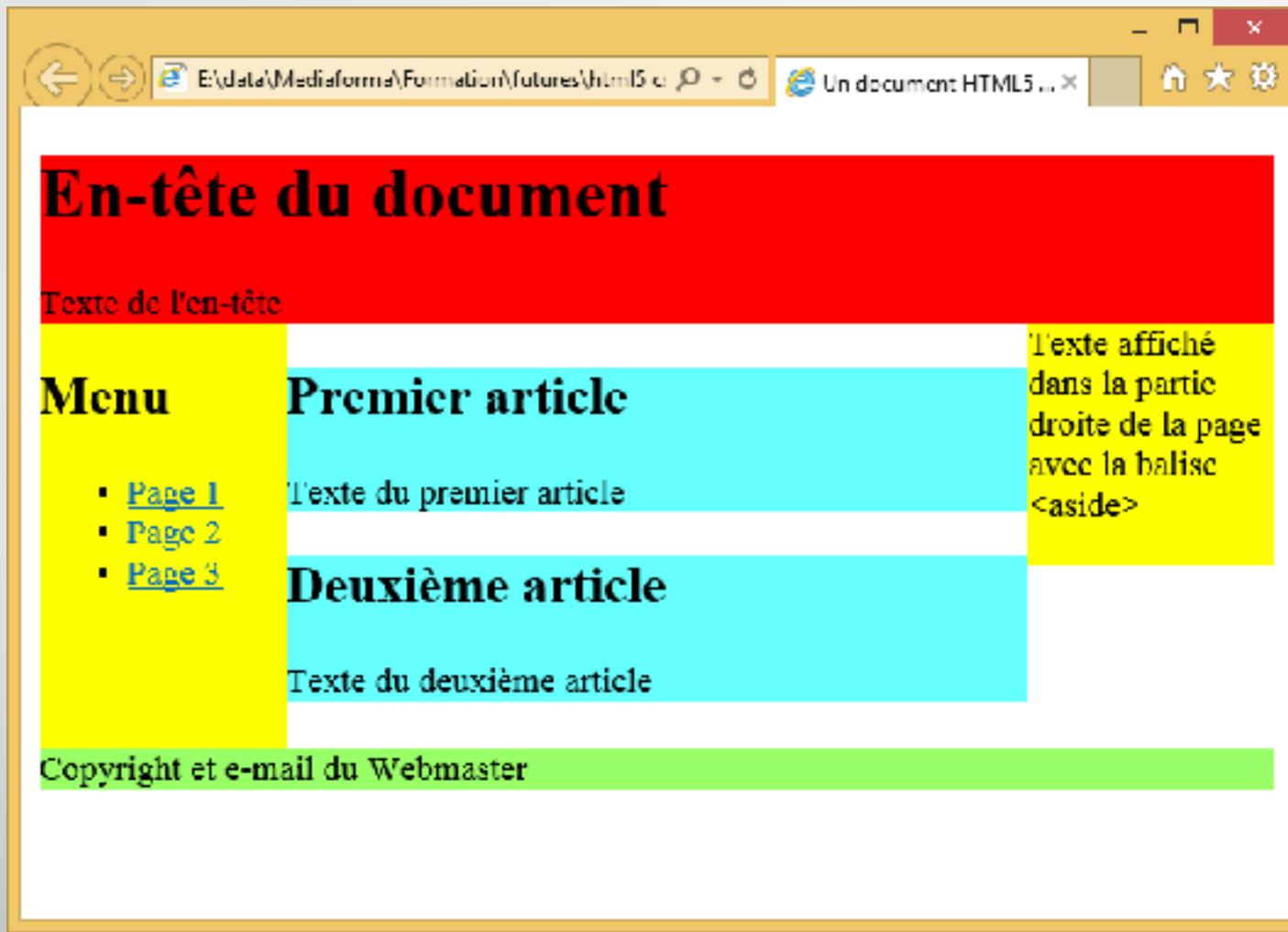
footer définit le pied de page d'une section ou d'une page web.

Les navigateurs Internet Explorer 8 et inférieur ne sont pas en mesure d'affecter un style aux nouveaux éléments HTML5. En effet, puisqu'ils ne les connaissent pas, ils doivent les intégrer au DOM *via* la méthode `createElement`. Pour ce faire, quelques lignes de JavaScript sont nécessaires :

```
document.createElement("footer");  
document.createElement("header");  
document.createElement("article");  
document.createElement("nav");  
document.createElement("aside");
```

Saisissez ce code et  
sauvegardez-le sous le nom  
`Creation-elements-HTML5.js`

Exécutez le code HTML. Voici ce que vous devriez obtenir :



# Sections et niveaux de titre

Les éléments header, nav, aside, article et footer ne sont pas les seuls à faire leur apparition dans le langage HTML5. Vous pouvez également utiliser l'élément section pour définir des sections dans un document.

Il sert généralement pour segmenter une page ou un article en plusieurs sujets.

Il peut être employé conjointement aux éléments de titre h1 à h6 pour définir la structure du document.

Il correspond plus ou moins à l'ancien élément div du HTML 4.0x/XHTML 1.x, à ceci près qu'il y ajoute une signification sémantique. En effet, il regroupe des éléments qui traitent d'un même sujet.

# Sections et niveaux de titre

Ici, par exemple, nous utilisons un élément section pour englober plusieurs articles au sein du document :

```
<section>  
    <article>  
        Contenu du premier article  
    </article>  
    ...  
    <article>  
        Contenu du dernier article  
    </article>  
</section>
```

# Sections et niveaux de titre

Vous pouvez également inclure plusieurs contenus dans un article en utilisant plusieurs sections.

```
<article>

    <section>
        Contenu de la première section
    </section>

    ...
    <section>
        Contenu de la dernière section
    </section>

</article>
```

# Sections et niveaux de titre

Prenons l'exemple d'un blog, constitué d'un empilement de plusieurs articles (ou *posts*), non nécessairement liés les uns aux autres. Ces composants pourraient être insérés dans des éléments article, eux-mêmes insérés dans un élément section :

```
<section>
  <article>
    <h1>Titre de la première section</h1>
    Texte ou autre contenu qui présente la section
  </article>
  ..
  <article>
    <h1>Titre de la dernière section</h1>
    Texte ou autre contenu qui présente la section
  </article>
</section>
```

# Sections et niveaux de titre

Si vous avez du mal à faire la différence entre <section> et <article>, dites-vous qu'une section peut être composée d'un ou de plusieurs articles qui traitent d'un même sujet.

Par exemple, si une page web contient plusieurs sujets sur les baleines blanches et plusieurs autres sur les arbres fruitiers, les sujets sur les baleines blanches seront regroupés dans une même section et décomposés en autant d'articles que nécessaire.

De même en ce qui concerne les sujets sur les arbres fruitiers.



```
<section>
    <h1>Articles sur les baleines blanches</h1>
    <article>
        <h2>Les baleines blanches, habitat</h2>
        <p>texte de l'article</p>
    </article>
    <article>
        <h2>Les baleines blanches, population</h2>
        <p>texte de l'article</p>
    </article>
    <article>
        <h2>Les baleines blanches, une espèce menacée</h2>
        <p>texte de l'article</p>
    </article>
</section>
<section>
    <h1>Articles sur les arbres fruitiers</h1>
    <article>
        <h2>Les arbres fruitiers, catégories</h2>
        <p>texte de l'article</p>
    </article>
    <article>
        <h2>Les arbres fruitiers, taille</h2>
        <p>texte de l'article</p>
    </article>
    <article>
        <h2>Les arbres fruitiers, récolte</h2>
        <p>texte de l'article</p>
    </article>
</section>
```

# Sections et niveaux de titre

Un ou plusieurs niveaux de titre `<h1>` à `<h6>` peuvent être utilisés à l'intérieur d'un élément section, mais aussi article, aside ou nav. Dans le code suivant, des niveaux de titre `<h2>` sont utilisés dans les sections et un niveau de titre `<h1>` dans l'article qui englobe les sections :

```
<article>
  <h1>Titre de l'article</h1>
  Texte ou autre contenu qui présente les grandes lignes de l'article
  <section>
    <h2>Titre de la section 1</h2>
    <p>Contenu texte pour la section 1.</p>
  </section>
  ...
  <section>
    <h2>Titre 2 pour dernière section </h2>
    <p>Contenu texte pour la dernière section</p>
  </section>
</article>
```

# Mise en forme d'un document en HTML5 et CSS3

Dans cette partie :

- Polices exotiques
- Couleur et image d'arrière-plan
- Gradients linéaires
- Gradient radiaux
- Masquer une image avec un gradient
- RGBA() et HSLA()
- Regrouper des éléments
- etc..

# Polices exotiques

CSS3 permet d'utiliser des polices qui ne sont pas installées sur les "postes clients". C'est la propriété CSS3 @font-face qui autorise cette prouesse.

```
@font-face
{
    font-family: 'nom-police';
    [font-style: style-police;]
    [font-variant: petites-capitales;]
    [font-weight: graisse-police;]
    [font-stretch: condensé-étendu;]
    [font-size: taille-police;]
    src: [local('nom-local'),] url('url-police')
[format(format-police)];
}
```

- nom-police est le nom de la police, tel qu'il sera utilisé dans la feuille de styles.
- style-police définit le style de la police : all, normal, italic ou oblique.
- petites-capitales indique la casse des caractères : normal ou small-caps.
- graisse-police indique la graisse de la police : all, normal, bold, 100, 200, 300, 400, 500, 600, 700, 800 ou 900.
- condensé-étendu définit l'état de compression/extension horizontale de la police : all, normal, ultra-condensed, extra-condensed, condensed, semi-condensed, semi-expanded, expanded, extra-expanded ou ultra-expanded.
- taille-police définit les tailles possibles pour la police : all pour toutes les polices proportionnelles, une ou deux tailles pour les polices bitmap ou devant être rendues dans une taille ou une fourchette de tailles spécifiques.
- nom-local définit le nom local de la police, au cas où cette dernière serait installée sur l'ordinateur.
- url-police définit l'adresse URL de la police accessible sur le serveur.
- format-police définit le format de la police : truedoc-pfr (TrueDoc™ Portable Font Resource, .pfr), embedded-opentype (Embedded OpenType, .eot), type-1 (PostScript™ Type 1, .pfb ou .pfa), truetype (TrueType, .ttf) ou opentype (OpenType, .ttf).

# Polices exotiques

Cette formation va vous montrer comment :

1. Installer un kit de polices en utilisant le site Font Squirrel
2. Utiliser des Google Fonts
3. Insérer des caractères Font Awesome dans vos documents HTML

# Polices exotiques - Font Squirrel

Rendez-vous sur le site <https://www.fontsquirrel.com/> et trouvez la police que vous voulez utiliser en vous aidant des catégories sous **FIND FONTS**.

Attention, les polices qui portent la mention **OFF SITE** sont payantes.

The screenshot shows a web browser window displaying the Fontsquirrel website. The address bar shows 'freesquirrel.com'. The page content includes a heading 'and presenting them in an easy-to-use format. Here are some of our favorites:' followed by four font preview sections:

- Intro Rust**: Shows the text 'INTRO RUST AABBCDDDEEFFG' in a textured, blocky font. Below it is a navigation bar with icons for desktop, mobile, and tablet, followed by 'Intro Rust > Fontfabric > 3 Styles >' and a blue 'DOWNLOAD OTF [OFF SITE]' button.
- Source Sans Pro**: Shows the text 'Source Sans Pro AaBbCcDdEeFfGgH' in a clean sans-serif font. Below it is a navigation bar with icons for desktop, mobile, and tablet, followed by 'Source Sans Pro > Adobe > 12 Styles >' and a blue 'DOWNLOAD OTF' button.
- Nexa Rust**: Shows the text 'NEXA RUST ABCDEFHIJ' in a bold, textured font. Below it is a navigation bar with icons for desktop, mobile, and tablet, followed by 'Nexa Rust > FontFabric > 5 Styles >' and a blue 'DOWNLOAD OTF [OFF SITE]' button.
- Open Sans**: Shows the text 'Open Sans AaBbCcDdEeFfGgHhij' in a clean sans-serif font. Below it is a navigation bar with icons for desktop, mobile, and tablet, followed by 'Open Sans > FontFabric > 100 Styles >' and a blue 'DOWNLOAD OTF' button.

At the bottom of the screenshot, the URL 'https://www.fontsquirrel.com/fonts/nexa-rust' is visible.

# Polices exotiques - Font Squirrel

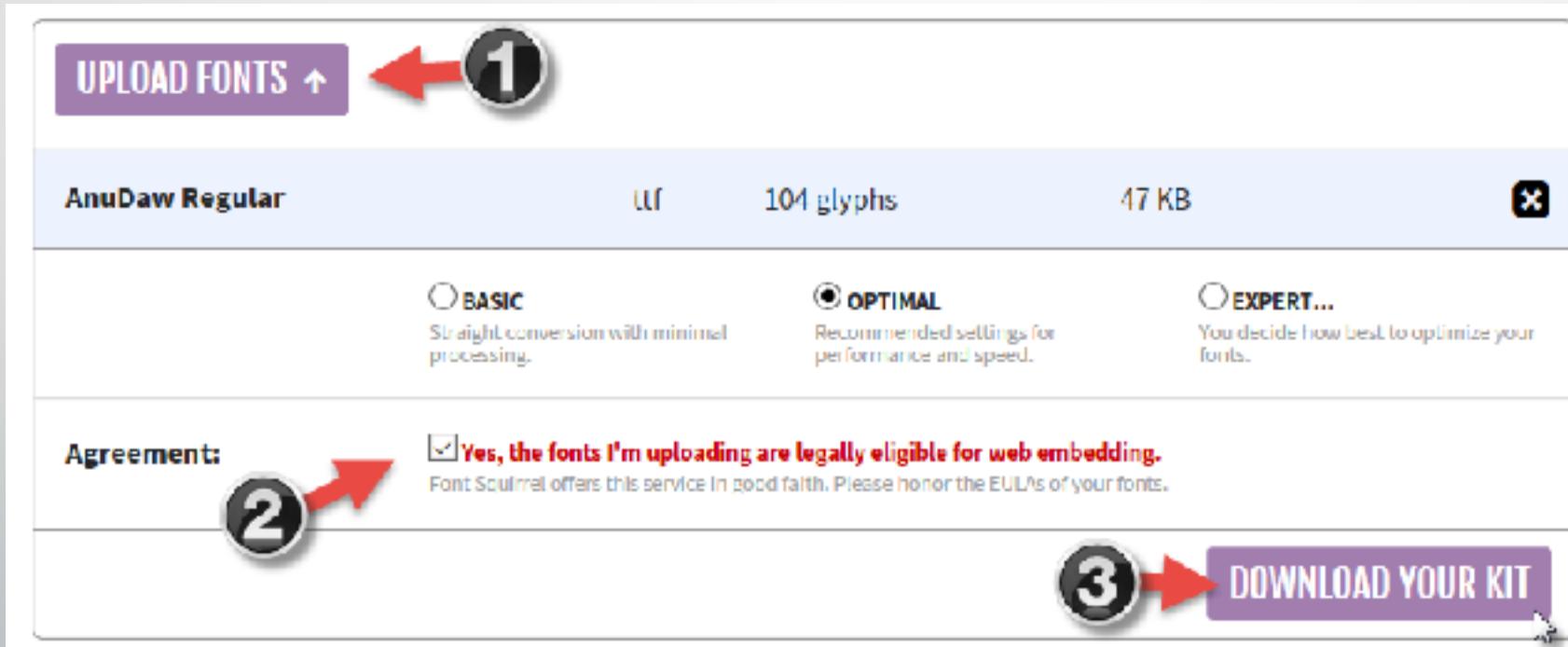
Cliquez sur le bouton **DOWNLOAD** de la police que vous voulez utiliser. Vous obtenez un fichier ZIP. Dézippez ce fichier. Vous obtiendrez un ou plusieurs fichiers de police et un fichier de licence. Lisez bien ce fichier pour savoir dans quel cadre vous pouvez utiliser la police.

Malheureusement, il existe plusieurs formats de polices et tous les navigateurs ne supportent pas tous les formats. Vous allez donc devoir décliner la police que vous avez téléchargée en plusieurs variantes pour assurer une compatibilité aussi grande que possible avec les divers navigateurs disponibles sur le marché.

Cette opération se fera sur le site Font Squirrel, en utilisant l'onglet **Generator**.

# Polices exotiques - Font Squirrel

Uploadez la police que vous avez téléchargée en cliquant sur **UPLOAD FONTS**, sélectionnez l'option **OPTIMAL** et cliquez sur **DOWNLOAD YOUR KIT** :



# Police exotiques - Font Squirrel

Le kit de Font Squirrel est un fichier ZIP. Dézipez-le. Vous obtiendrez plusieurs fichiers de polices et un fichier CSS qui contient une instruction @font-face prête à l'emploi. Par exemple :

```
@font-face {  
    font-family: 'anudawregular';  
    src: url('anudrg_-webfont.woff2') format('woff2'),  
         url('anudrg_-webfont.woff') format('woff');  
    font-weight: normal;  
    font-style: normal;  
}
```

Utilisez cette instruction dans vos pages et faites référence à la police (ici anudawregular) avec une instruction du type suivant :

```
h1 {  
    font-family: 'anudawregular';  
}
```

# Polices exotiques - Font Squirrel

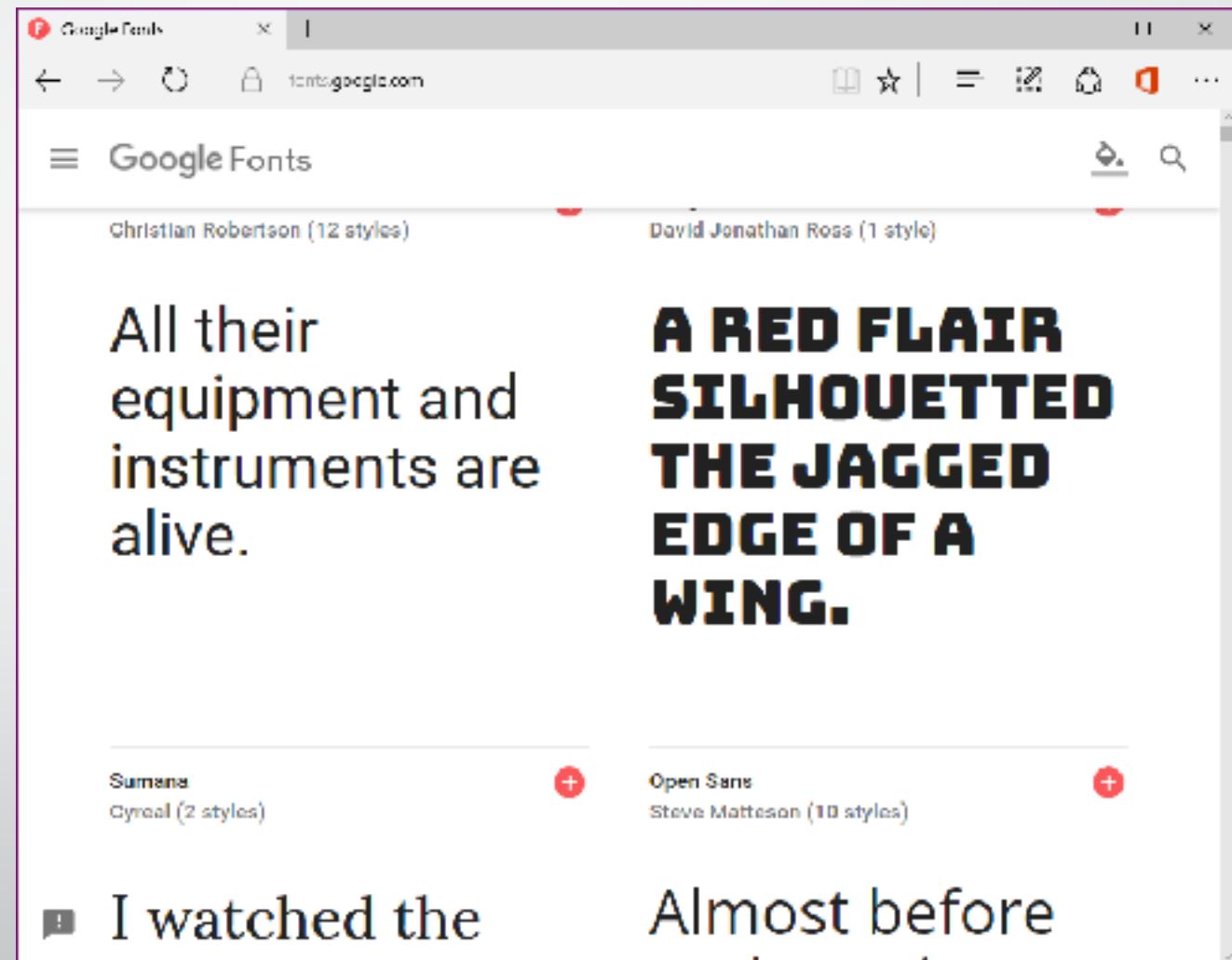
## Exercice

Téléchargez une police gratuite sur Font Squirrel, créez un kit de polices à partir de cette police et utilisez-le dans un document HTML.

# Polices exotiques - Google Fonts

Google propose un grand nombre de polices libres de droits. Il est très simple de les utiliser dans vos sites Web.

Commencez par vous rendre sur le site <https://fonts.google.com/> et trouvez la police que vous voulez utiliser.



# Polices exotiques - Google Fonts

Ici par exemple, nous allons utiliser la police **Bungee Inline**.

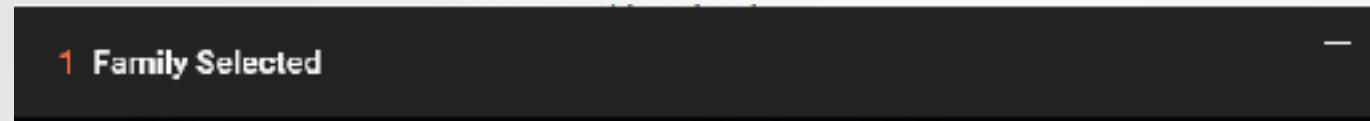
Cliquez sur le signe "+" correspondant :



# Police exotiques - Google Fonts

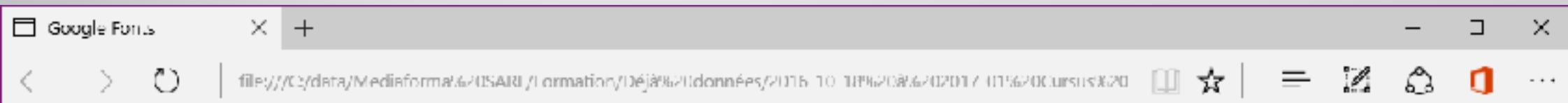
La barre de titre d'une fenêtre popup s'affiche dans la partie inférieure de la page :

Cliquez dessus pour la développer la fenêtre. Vous trouverez les deux instructions à utiliser pour accéder à la police :

A screenshot of the Google Fonts interface, identical to the one above but with a red box highlighting the entire window. It shows the "Bungee Inline" font selected, with options to "EMBED" or "CUSTOMIZE" and a "Load Time" set to "Fast". Below the selection, there are sections for "Embed Font" (with code provided) and "Specify in CSS" (with code provided). A note at the bottom says: "For examples of how fonts can be added to webpages, see the [getting started guide](#)".

# Polices exotiques - Google Fonts

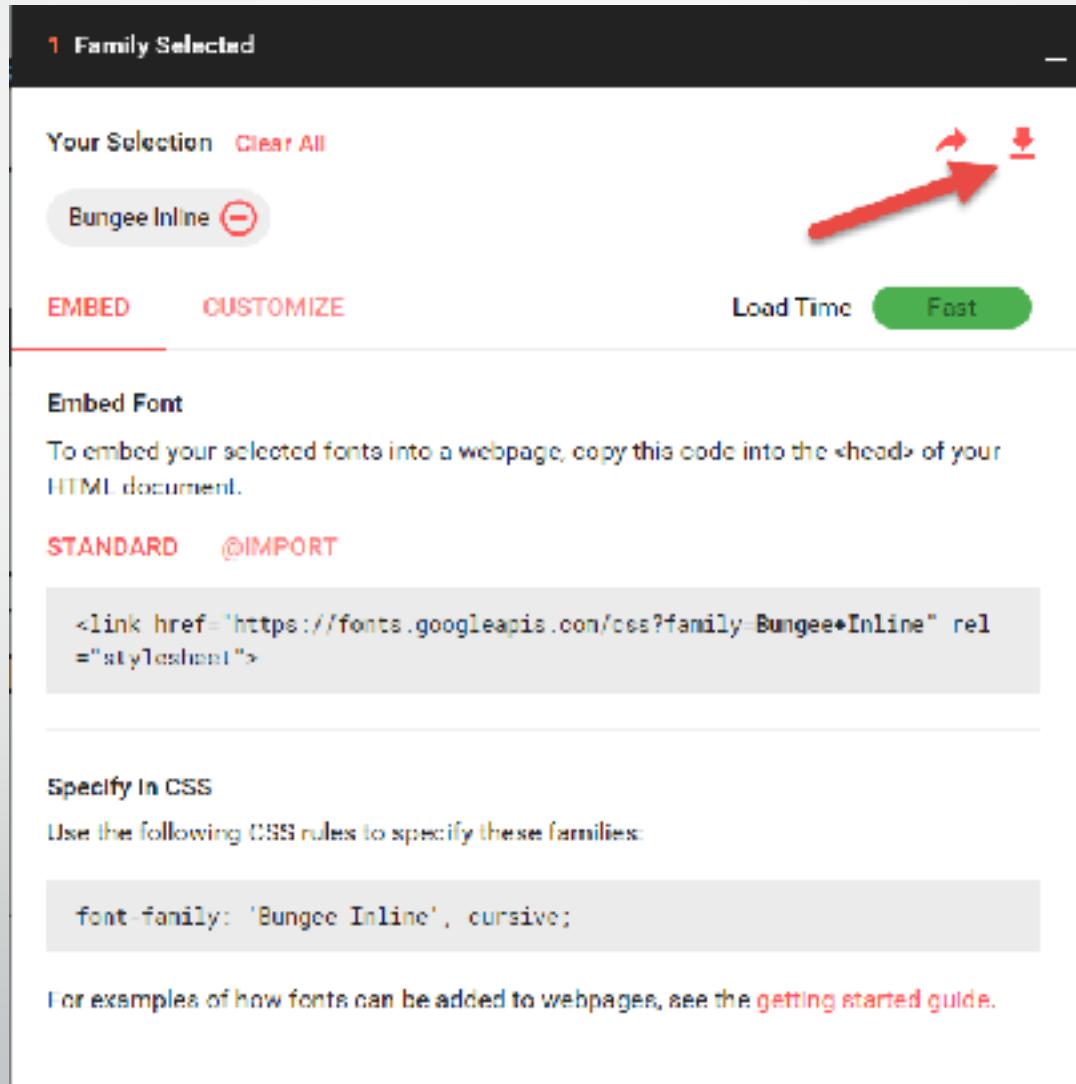
```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Google Fonts</title>
    <link href="https://fonts.googleapis.com/css?family=Bungee+Inline" rel="stylesheet">
    <style>
      h1 {
        font-family: 'Bungee Inline', cursive;
      }
    </style>
  </head>
  <body>
    <h1>Ce titre utilise la Google Font Bungee Inline</h1>
  </body>
</html>
```



**CE TITRE UTILISE LA GOOGLE FONT BUNGEE INLINE**

# Polices exotiques - Google Fonts

Si vous le souhaitez, vous pouvez également télécharger la police et l'inclure dans votre site en cliquant sur l'icône de téléchargement :



1 Family Selected

Your Selection [Clear All](#)

Bungee Inline [-](#)

[EMBED](#) [CUSTOMIZE](#) Load Time [Fast](#)

**Embed Font**  
To embed your selected fonts into a webpage, copy this code into the <head> of your HTML document.

**STANDARD** [@IMPORT](#)

```
<link href="https://fonts.googleapis.com/css?family=Bungee+Inline" rel="stylesheet">
```

**Specify In CSS**  
Use the following CSS rules to specify these families:

```
font-family: 'Bungee Inline', cursive;
```

For examples of how fonts can be added to webpages, see the [getting started guide](#).

# Polices exotiques - Google Fonts

## Exercice

Entraînez-vous à utiliser une ou plusieurs polices Google Fonts.

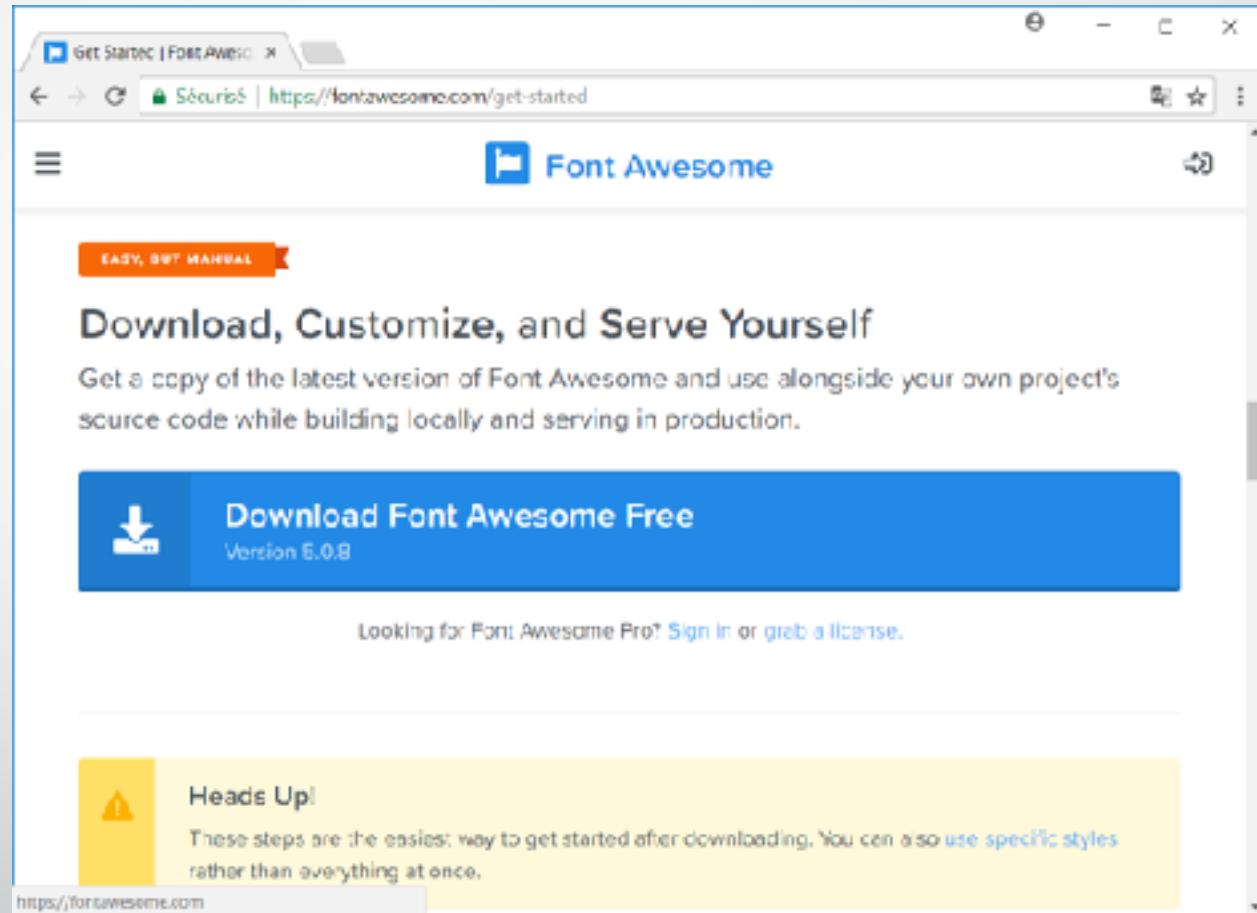
# Font Awesome

Font Awesome donne accès à de nombreuses icônes dignes d'intérêt.

Pour y accéder, commencez par vous rendre sur le site

<https://fontawesome.com/get-started>

Et téléchargez Font Awesome en cliquant sur le bouton **Download Font Awesome Free** :



Extrayez le fichier **fontawesome-all.js** de cette archive et placez-le dans un dossier de votre ordinateur.

Vous y ferez référence avec une balise <script> dans le <head> :

```
<script defer src="fontawesome-all.js"></script>
```

Ca y est, vous pouvez utiliser les icônes de Font Awesome dans votre code HTML !

# Font Awesome

Pour accéder aux icônes de FontAwesome, allez sur la page

<https://fontawesome.com/icons?d=gallery>

Cliquez sur une des icônes proposées pour savoir comment l'intégrer dans votre code HTML. Vers le bas de la page, vous obtiendrez le code de l'icône. Par exemple :

```
<i class="fab fa-angellist"></i>
```

# Font Awesome

Exercice :

Insérez le code nécessaire pour afficher une bicyclette.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Font Awesome</title>
    <script defer src="fontawesome-all.js"></script>
    <style>

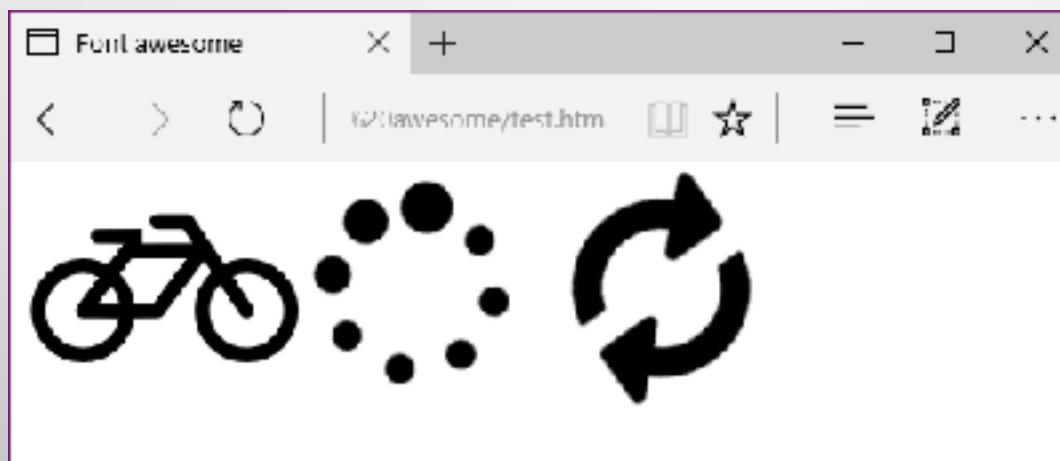
      </style>
  </head>
  <body>
    <p>ceci est un texte</p>
    <i class="fas fa-cubes"></i>
    <i class="fas fa-camera-retro"></i>
  </body>
</html>
```

Solution

# Font Awesome

## Exercice

En consultant les pages du site <http://fontawesome.io/> ; trouvez comment changer la couleur des icônes Font Awesome, comment choisir leur taille ou encore comment avoir des icônes animées.



# Font Awesome

## Solution

```
<i class="fa fa-bicycle fa-5x" aria-hidden="true"></i>
<i class="fa fa-spinner fa-spin fa-5x"></i>
<i class="fa fa-refresh fa-spin fa-5x fa-fw"></i>
```

Un exemple de remplacement des puces d'une liste à puces par des caractères FontAwesome.

Ici, on utilise le sélecteur li::before pour insérer un contenu devant les <li>. Le contenu est inséré sous la forme d'un caractère Unicode de police FontAwesome.

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Arriere plan fluide</title>
        <link rel="stylesheet"
            href="https://maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">
    <style>
        ul {
            list-style-type: none;
            padding-left: 20px;
        }
        li{color:#ff00ff;font-size: 30px;list-style:none; }
        li {
            position: relative;
            padding-left: 20px;
            margin-bottom: 10px
        }
        li::before{
            font-family: FontAwesome;
            content: "\f1b0";
        }
    </style>
    </head>
    <body>
        <ul>
            <li>
                un
            </li>
            <li>
                deux
            </li>
            <li>
                trois
            </li>
        </ul>
    </body>
</html>
```

# Couleur et image d'arrière-plan

La couleur d'arrière-plan d'un élément est définie par la propriété background-color. Voici sa syntaxe :

```
background-color: couleur;
```

La valeur couleur peut être spécifiée :

- "En dur", c'est-à-dire sous la forme d'un nom normalisé : yellow ou skyblue par exemple.
- À l'aide d'un code hexadécimal sur 6 digits : #RRVVBB (où RR, VV et BB sont les composantes rouge, vert et bleu de la couleur, codées en hexadécimal entre 00 et FF).
- À l'aide de la fonction RGB(R,V,B), où R, V et B sont des valeurs comprises entre 0 et 255.

L'image d'arrière-plan d'un élément est définie par la propriété background-image. Voici sa syntaxe :

```
background-image: url (image);
```

Où image est l'adresse URL de l'image. Par exemple img/fond.jpg, ou encore http://monsite.com/image/fond.png.

Il n'est pas recommandé d'utiliser des guillemets pour délimiter l'adresse URL de l'image d'arrière-plan car ils peuvent provoquer des dysfonctionnements dans les navigateurs anciens.

# Couleur et image d'arrière-plan

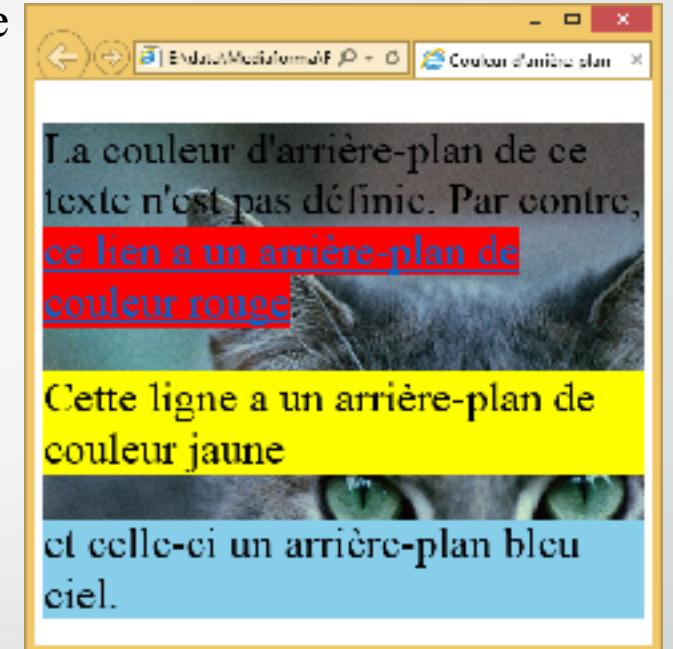
Une image d'arrière-plan peut être répétée horizontalement et/ou verticalement. Pour cela, on donne la valeur repeat (répétition horizontale et verticale), repeat-x (répétition horizontale seulement) ou repeat-y (répétition verticale seulement), à la propriété CSS background-repeat.

La position d'une image d'arrière-plan peut être définie horizontalement et/ou verticalement : affectez la position horizontale (left, center ou right) et éventuellement la position verticale (top, center ou bottom) à la propriété CSS background-position.

# Couleur et image d'arrière-plan

Dans cet exemple (code005.htm), du texte est affiché au-dessus d'une image en arrière-plan. Nous modifions tour à tour la couleur d'arrière-plan d'un lien hypertexte, d'un paragraphe, puis d'un élément `div`. Les styles correspondants sont directement définis dans les éléments concernés :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Couleur d'arrière-plan</title>
  </head>
  <body>
    <div style="background-image: url(chat.jpg); font-size: 40px; color: black;">
      <p>La couleur d'arrière-plan de ce texte n'est pas définie. Par contre, <a href="#" style="background-color: red;">ce lien a un arrière-plan de couleur rouge</a>.</p>
      <p style="background-color: yellow;">Cette ligne a un arrière-plan de couleur jaune</p>
      <div style="background-color: skyblue;">et celle-ci un arrière-plan bleu ciel.</div>
    </div>
  </body>
</html>
```



# Gradient vertical

Pour définir un gradient vertical, vous utiliserez ces propriétés CSS :

```
-webkit-gradient(linear, p1, p2, from(c1), to(c2));  
-ms-linear-gradient(p1, c1, c2) no-repeat;  
-moz-linear-gradient(p1, c1, c2) no-repeat;
```

Où :

- p1 représente les coordonnées du point de départ du gradient : top, left, right et/ou down sur les navigateurs IE et Firefox ; pourcentage horizontal et pourcentage vertical sur les navigateurs Webkit.
- p2 représente les coordonnées du point d'arrivée du gradient sur les navigateurs Webkit. Il est défini par un pourcentage horizontal et un pourcentage vertical de la zone cible.
- c1 est la couleur de départ du gradient.
- c2 est la couleur de fin du gradient.

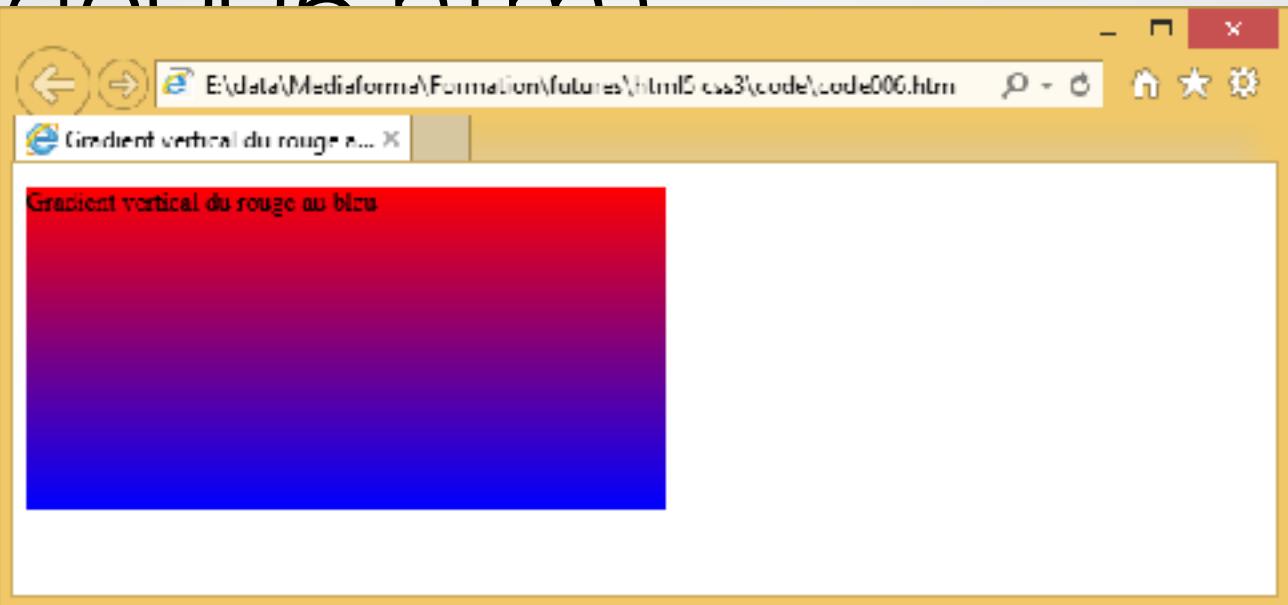
Cette solution n'est pas encore implémentée sur tous les navigateurs et il est nécessaire d'utiliser des préfixes pour s'adresser à chaque navigateur : -webkit pour Chrome, -ms pour IE, -moz pour Firefox.

# Exercice

Définissez un gradient vertical de la couleur rouge à la couleur bleue.

# Solution (code006.htm)

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Gradient vertical du rouge au bleu</title>
    <link rel="stylesheet" href="gradient-vertical.css" />
  </head>
  <body>
    <p class="gradient-vertical">Gradient vertical du rouge au bleu</p>
  </body>
</html>
```



Code CSS3

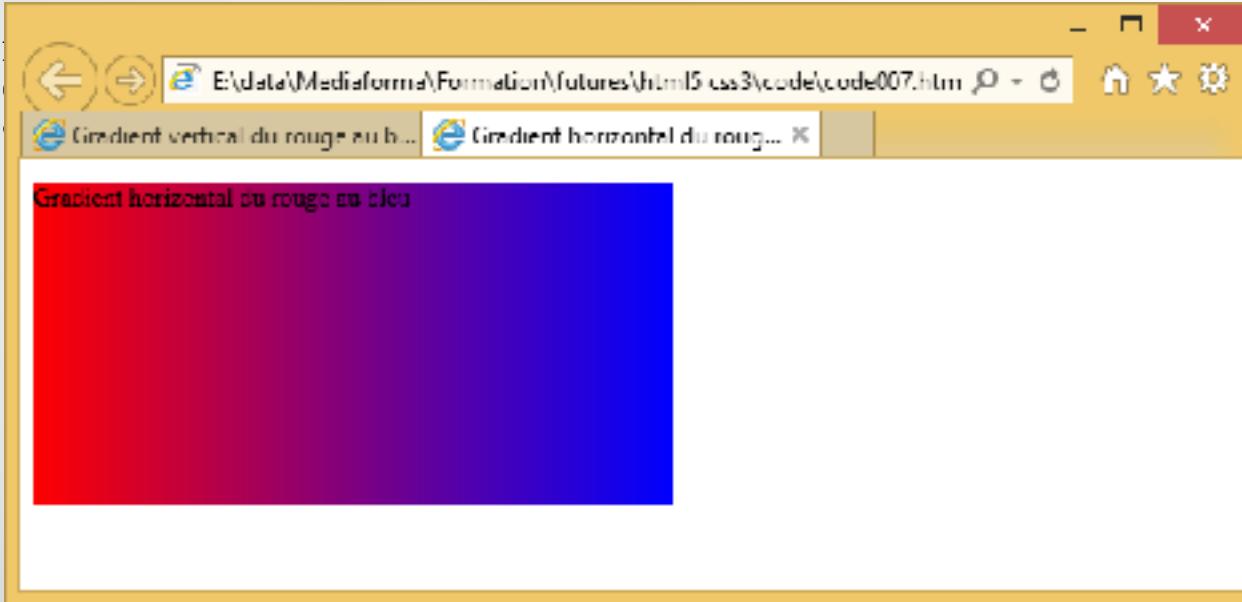
```
.gradient-vertical
{
  width:400px;
  height: 200px;
  background: -ms-linear-gradient(top, red, blue);
  background: -moz-linear-gradient(top, red, blue);
  background: -webkit-linear-gradient(top, red, blue);
}
```

# Exercice

Définissez un gradient horizontal de la couleur rouge à la couleur bleue.

# Solution (code007.htm)

```
<!doctype html>
<html>
  <head>
    <meta c...
  <title>...
  <link r...
  </head>
  <body>
    <p clas...
au bleu</p>
  </body>
</html>
```



Code CSS3

```
.gradient-horizontal
{
  width:400px;
  height: 200px;
  background: -ms-linear-gradient(left, red, blue);
  background: -moz-linear-gradient(left, red, blue);
  background: -webkit-gradient(linear, 0% 0%, 100% 0%, from(red),
to(blue));
}
```

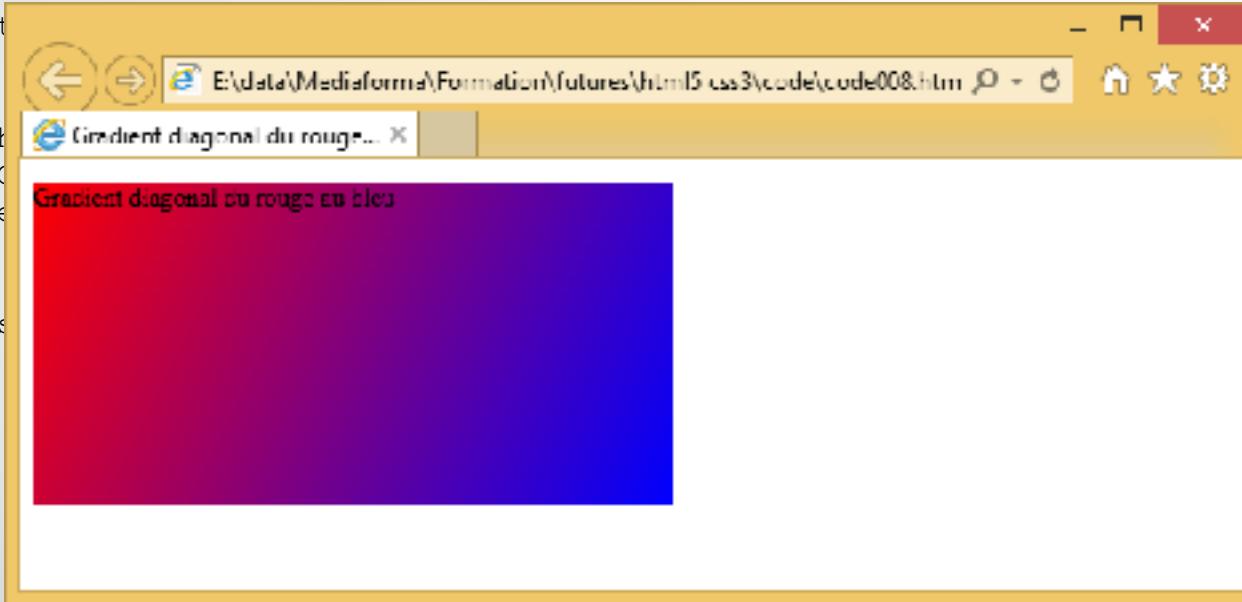
Code HTML5

# Exercice

Définissez un gradient diagonal de la couleur rouge à la couleur bleue.

# Solution (code008.htm)

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Gradient diagonal du rouge au bleu</title>
    <link rel="stylesheet" type="text/css" href="code008.css" />
  </head>
  <body>
    <p class="gradient"></p>
  </body>
</html>
```



Code CSS3

```
width:400px;
height: 200px;
background: -moz-linear-gradient(top left, red, blue);
background: -ms-linear-gradient(top left, red, blue);
background: -webkit-gradient(linear, 0% 0%, 100% 100%,
from(red), to(blue));
}
```

Code HTML5

# Plusieurs couleurs dans un gradient linéaire

Vous savez définir un gradient linéaire à deux couleurs. Il est possible de lui adjoindre des couleurs intermédiaires, grâce à l'ajout de quelques paramètres dans les propriétés `-webkit-gradient`, `-ms-linear-gradient` et `-moz-linear-gradient`.

# Plusieurs couleurs dans un gradient linéaire

## **Navigateurs Gecko et IE**

`-moz-linear-gradient(p1, c1 p1, c2 p2, ..., cN);`

`-ms-linear-gradient(p1, c1 p1, c2 p2, ..., cN);`

Où :

- p1 à pN-1 sont les coordonnées des différentes transitions de couleurs, exprimées en pourcentages.
- c1 à cN sont les différentes couleurs du gradient.

# Plusieurs couleurs dans un gradient linéaire

## Navigateurs Webkit

```
-webkit-gradient(linear, p1, p2, from(c1), color-stop(v2, c2), color-stop(v3, c3), ..., to(cN));
```

Où :

- p1 et p2 sont les coordonnées des points de départ et d'arrivée du gradient, exprimées en pourcentages.
- c1 à cN sont les différentes couleurs du gradient.
- v1 sont les positions des différentes transitions de couleurs, exprimées dans un nombre décimal compris entre 0 et 1.

# Exercice

Définissez un gradient linéaire horizontal – qui passe du rouge au bleu au jaune et au blanc – en répartissant de façon égale les différentes transitions. Ce gradient doit être compatible Gecko, IE et Mozilla.

# Solution (code009.htm)

Deux points intermédiaires doivent être définis : un à 33 % et un à 66 % de la zone cible.

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Gradient horizontal multiple</title>
  </head>
  <body>
    <p class="gradient-horizontal-multiple">Gradient horizontal multiple, du rouge au blanc, couleurs intermédiaires bleu et jaune</p>
  </body>
</html>
```



Code CSS3

```
.gradient-horizontal-multiple
{
  width:400px;
  height: 200px;
  background: -moz-linear-gradient(left, red 0%, blue 33%, yellow 66%, white);
  background: -ms-linear-gradient(left, red 0%, blue 33%, yellow 66%, white);
  background: -webkit-gradient(linear, 0% 0%, 100% 0%, from(red), color-stop(0.33, blue), color-stop(.66, yellow), to(white));
}
```

Code HTML5

# Plusieurs couleurs dans un gradient linéaire

Pour créer vos gradients plus aisément, vous pouvez vous rendre à la page <http://www.colorzilla.com/gradient-editor/>, où quelques réglages élémentaires permettent d'obtenir le code CSS3 correspondant.

# Gradient radial

Les gradients linéaires ne sont pas les seuls utilisables : on peut également définir des gradients radiaux, qui fixent un dégradé de couleurs d'une zone circulaire vers le reste de l'élément ciblé. Les propriétés à utiliser sont les suivantes :

- `-webkit-gradient(radial, ...)` sur les navigateurs Webkit (Safari, Chrome) ;
- `-ms-radial-gradient` sur les navigateurs Internet Explorer ;
- `-moz-radial-gradient` sur les navigateurs Gecko (Mozilla Firefox).

# Gradient radial

## Navigateurs Webkit

```
background: -webkit-gradient(radial, p1, r1, p2, r2, from(c1), to(c2));
```

Où :

- p1 représente les coordonnées du point intérieur du gradient.
- r1 est le rayon intérieur du gradient.
- p2 représente les coordonnées du point extérieur du gradient.
- r2 est le rayon extérieur du gradient.
- c1 est la couleur intérieure du gradient.
- c2 est la couleur extérieure du gradient.

# Gradient radial

## Navigateurs Gecko et Internet Explorer

```
background: -moz-radial-gradient(p1, forme taille, c1, c2);
```

```
background: -ms-radial-gradient(p1, forme taille, c1, c2);
```

Où :

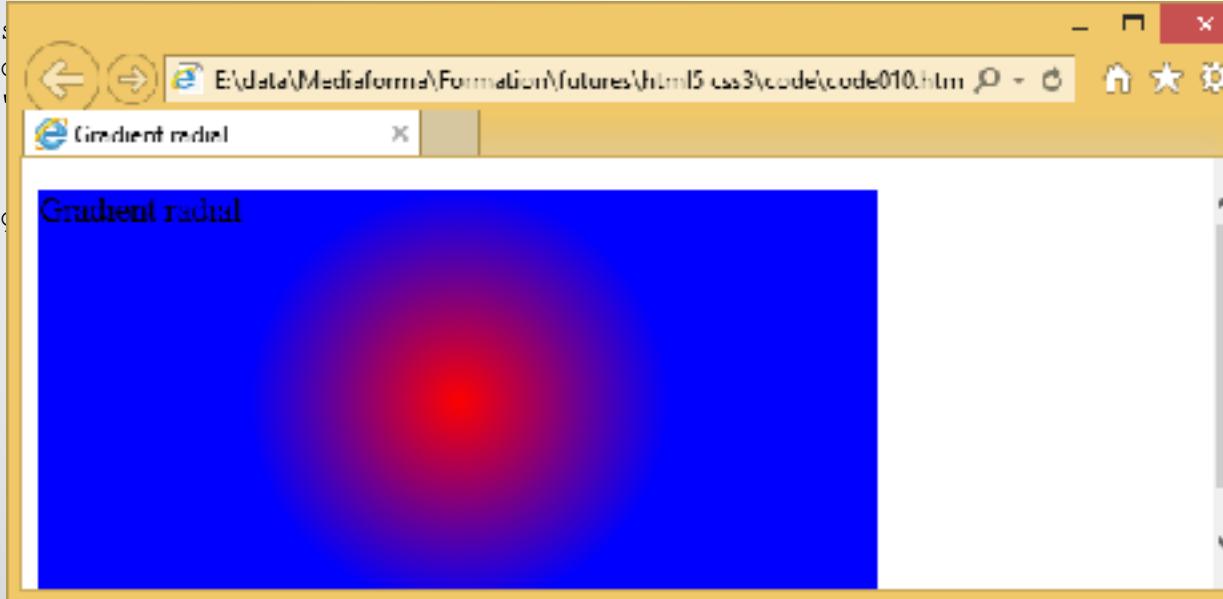
- p1 représente les coordonnées du point intérieur du gradient.
- forme représente la forme du gradient : circle ( cercle) ou ellipse ( ellipse).
- taille est la taille du gradient. Ce paramètre peut prendre d'une des valeurs suivantes :
  - closest-side (ou contain) pour que le gradient occupe toute la taille du conteneur.
  - closest-corner pour que le gradient occupe tout l'espace disponible jusqu'au coin le plus proche du conteneur.
  - farthest-side (ou cover) pour que le gradient occupe tout l'espace disponible jusqu'au coin le plus éloigné du conteneur.
- c1 est la couleur intérieure du gradient.
- c2 est la couleur extérieure du gradient.

# Exercice

Définissez un gradient vertical du rouge au bleu, compatible Mozilla, Gecko et IE.

# Solution (code010.htm)

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Gradient radial</title>
    <link rel="stylesheet" type="text/css" href="style.css"/>
  </head>
  <body>
    <p class="gradient">Gradient radial</p>
  </body>
</html>
```



HTML5

Code CSS3

```
height: 200px;
background: -webkit-gradient(radial, 50% 0%, 20, 50% 100%, 150, from(red),
to(blue));
background: -ms-radial-gradient(50% 50%, circle closest-side, red, blue);
background: -moz-radial-gradient(50% 50%, circle closest-side, red, blue);
}
```

# Gradient radial

Pour que le gradient n'occupe pas tout le conteneur, la méthode est totalement différente dans les navigateurs Webkit, Internet Explorer et Gecko.

## Navigateurs Webkit

Définissez un rayon extérieur de plus petite taille dans le code CSS3. Ici, 50 pixels au lieu de 150 :

```
background: -webkit-gradient(radial, 50% 50%, 0, 50% 50%, 50,  
from(red), to(blue));
```

# Gradient radial

## Navigateurs Internet Explorer et Gecko

Il est nécessaire d'ajouter deux pourcentages lors de la définition des couleurs. Ici, le rouge constitue le point de départ du gradient (0 %) et le bleu s'arrête à 30 % de la taille maximale du gradient (30 % de farthest-side) :

```
background: -moz-radial-gradient(50% 50%, circle farthest-side, red 0%, blue 30%);
```

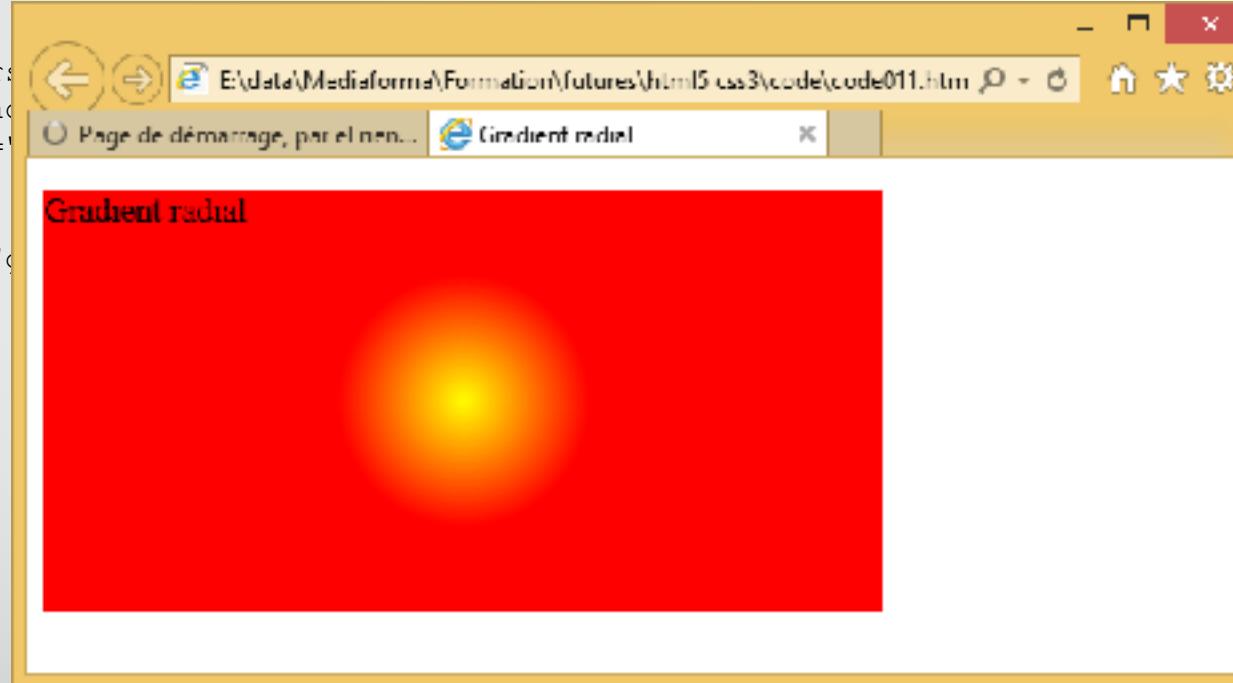
```
background: -ms-radial-gradient(50% 50%, circle farthest-side, red 0%, blue 30%);
```

# Exercice

Définissez un gradient radial de 50 pixels du jaune au rouge. Ce gradient sera compatible avec les navigateurs Gecko, IE et Mozilla.

# Solution (code011.htm)

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Gradient radial</title>
    <link rel="stylesheet" type="text/css" href="style.css"/>
  </head>
  <body>
    <p class="gradient">Gradient radial</p>
  </body>
</html>
```



Code CSS3

```
height: 200px;
background: -webkit-gradient(radial, 50% 50%, 0, 50% 50%, 50, from(yellow), to(red));
background: -moz-radial-gradient(50% 50%, circle farthest-side, yellow 0%, red 30%);
background: -ms-radial-gradient(50% 50%, circle farthest-side, yellow 0%, red 30%);
}
```

# Plusieurs couleurs dans un gradient radial

Tout comme pour les gradients linéaires, il est possible de définir plusieurs couleurs dans un gradient radial. La syntaxe à utiliser est très différente selon qu'on s'adresse à un navigateur Webkit, IE ou Gecko.

## Navigateurs Webkit

Un ou plusieurs color-stop doivent être définis entre la couleur de départ (from) et la couleur d'arrivée (to). Ici, par exemple, le gradient va du rouge au noir en passant par le jaune et le bleu :

```
background: -webkit-gradient(radial, 50% 50%, 0, 50% 50%, 150,  
from(red), color-stop(.25,yellow), color-stop(.50, blue), to (black));
```

# Plusieurs couleurs dans un gradient radial

## **Navigateurs IE et Gecko**

La syntaxe est beaucoup plus simple dans les navigateurs Gecko. Il suffit de préciser les différentes couleurs comme paramètres de la propriété `-moz-radial-gradient`. Ici, le gradient va du rouge au noir en passant par le jaune et le bleu :

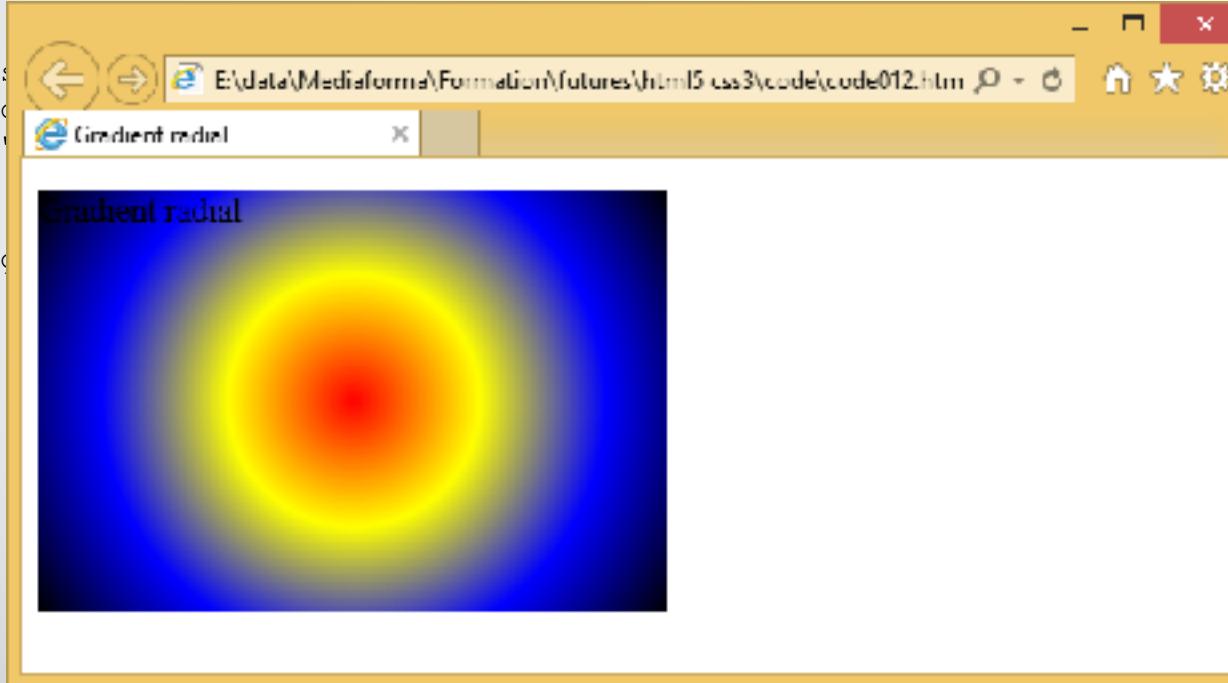
```
background: -moz-radial-gradient(circle, red, yellow, blue, black);
```

# Exercice

Utilisez les instructions précédentes pour définir un gradient radial multicolore compatible avec IE, Gecko et Mozilla.

# Solution (code012.htm)

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Gradient radial</title>
    <link rel="stylesheet" type="text/css" href="style.css"/>
  </head>
  <body>
    <p class="gradient">Gradient radial</p>
  </body>
</html>
```



HTML5

Code CSS3

```
height: 200px;
background: -webkit-gradient(radial, 50% 50%, 0, 50% 50%, 150, from(red), color-stop(.25,yellow), color-stop(.50, blue), to (black));
background: -moz-radial-gradient(circle, red, yellow, blue, black);
background: -ms-radial-gradient(circle, red, yellow, blue, black);
}
```

# Fonctions RGBA() et HSLA()

En utilisant les fonctions RGBA() ou HSLA(), on peut appliquer un effet de transparence aux couleurs manipulées dans le code CSS3. Ces fonctions s'appliquent à n'importe quelle propriété dont la valeur est une couleur : color, background-color, border-color, etc.

La fonction RGBA() admet quatre paramètres :

RGBA(red, green, blue, alpha)

Où :

- red, green et blue sont les composantes rouge, vert et bleu de la couleur. Ces trois informations peuvent prendre 256 valeurs, codées entre 0 et 255.
- alpha est le degré d'opacité de la couleur. Cette information est un nombre décimal codé entre 0 (transparent) et 1 (opaque).

# Fonctions RGBA() et HSLA()

La fonction HSLA() admet également quatre paramètres :

HSLA(hue, sat, light, alpha)

Où :

- hue représente la teinte. Les valeurs autorisées pour ce paramètre sont comprises entre 0 et 360. La valeur 0(ou 360) correspond au rouge, la valeur 120 correspond au vert et la valeur 250 au bleu.
- sat représente la teinte de la couleur, entre 0% et 100%.
- light représente la luminosité de la couleur, entre 0% (noir) et 100% (blanc).
- alpha est le degré d'opacité de la couleur. Cette information est un nombre décimal codé entre 0 (transparent) et 1 (opaque).

# Fonctions RGBA() et HSLA()

À titre d'exemple, nous allons définir plusieurs éléments div contenant du texte et dont la couleur d'arrière-plan est plus ou moins transparente. Voici le code HTML5 utilisé :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Facteur Alpha avec RGBA</title>
    <link rel="stylesheet" href="alpha.css">
  </head>
  <body>
    <div id="alpha00">Texte affiché sur un fond vert avec alpha à 0%</div>
    <div id="alpha20">Texte affiché sur un fond vert avec alpha à 20%</div>
    <div id="alpha50">Texte affiché sur un fond vert avec alpha à 50%</div>
    <div id="alpha80">Texte affiché sur un fond vert avec alpha à 80%</div>
  </body>
</html>
```

Ce code se contente d'afficher quatre blocs <div> de classe alpha00, alpha20, alpha50 et alpha80.

Saisissez ce code dans le fichier code013.htm

# Fonctions RGBA() et HSLA()

Voici le code CSS associé :

```
div
{
    width: 200px;
    height:100px;
}

.alpha00
{
    background-color: rgba(0, 100, 100, 0);
}

.alpha20
{
    background-color: rgba(0, 100, 100, .2);
}

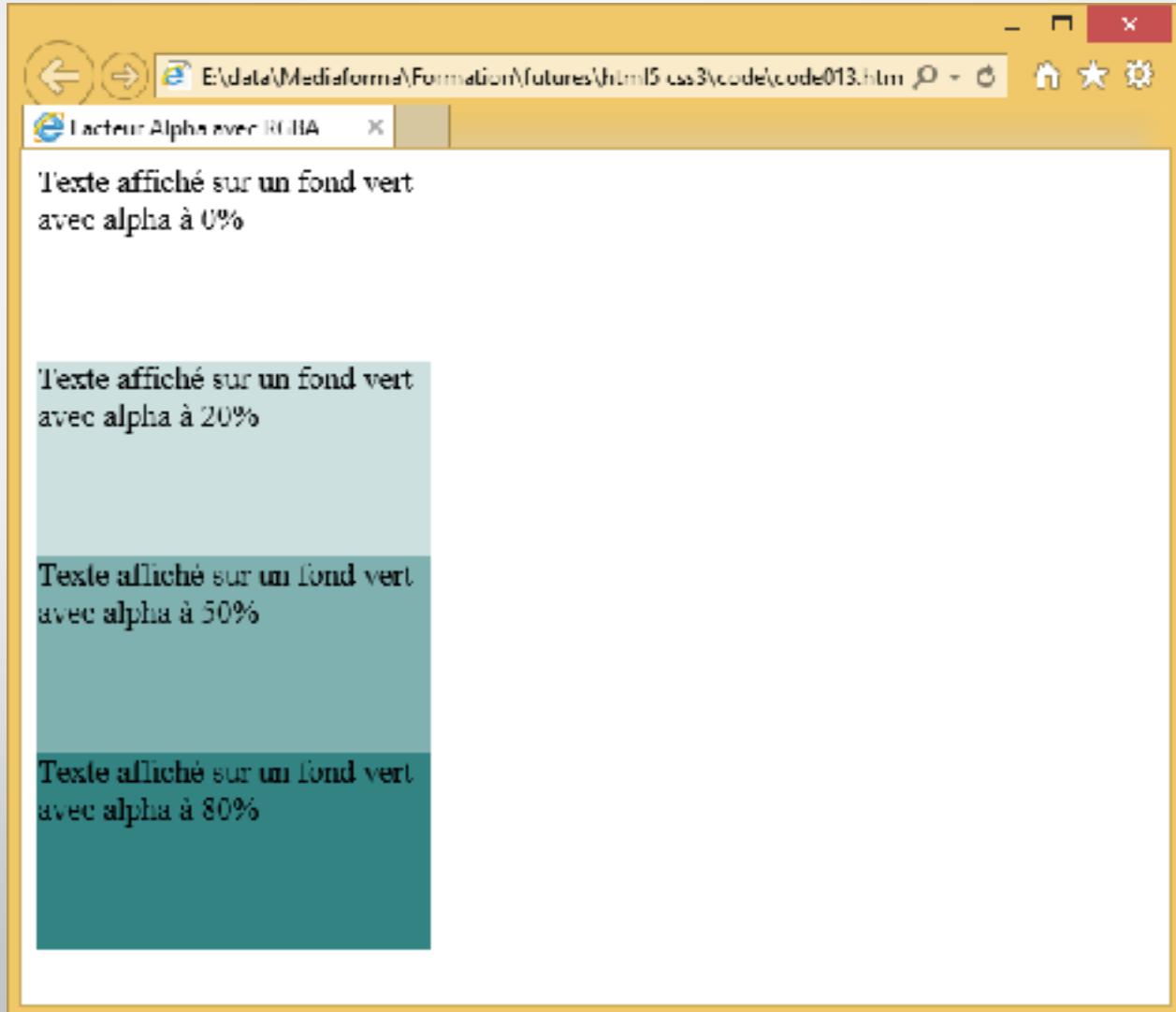
.alpha50
{
    background-color: rgba(0, 100, 100, .5);
}

.alpha80
{
    background-color: rgba(0, 100, 100, .8);
}
```

Saisissez ce code dans le fichier alpha.css

# Fonctions RGBA() et HSLA()

Voici ce que vous devriez obtenir :



# Fonctions RGBA() et HSLA()

La fonction HSLA() est une alternative à la fonction RGBA(). Le fichier *alpha.css* précédent pourrait être réécrit comme suit avec la fonction HSLA() :

```
div
{
    width: 200px;
    height:100px;
}

.alpha00
{
    background-color: hsla(120, 94%, 18%, 0);
}

.alpha20
{
    background-color: hsla(120, 94%, 18%, .2);
}

.alpha50
{
    background-color: hsla(120, 94%, 18%, 0.4);
}

.alpha80
{
    background-color: hsla(120, 94%, 18%, .8);
}
```

Pour choisir une couleur plus facilement, vous pouvez vous rendre sur les pages [www.colorschemer.com/online.html](http://www.colorschemer.com/online.html) (RGB) ou [www.workwithcolor.com/hsl-color-schemer-01.htm](http://www.workwithcolor.com/hsl-color-schemer-01.htm) (HSL).

# Positionner en flux

Lorsque la position d'un élément n'est pas spécifiée, il se place d'après l'ordre dans lequel il apparaît dans le code, en ligne ou en bloc selon son rendu.

Par exemple, l'élément a va tout naturellement se placer à la suite du texte qui le précède.

Cliquez [pour accéder au site de formation Mediaforma](http://www.mediaforma.com)

Cliquez [pour accéder au site de formation Mediaforma](http://www.mediaforma.com)

Par contre, l'élément de rendu block <h2> va s'afficher sur la ligne suivante :

Ce texte est suivi<h2> par un titre de style H2</h2>

Ce texte est suivi

**par un titre de style H2**

# Positionner en flux

Il en va de même des objets placés à l'intérieur d'un conteneur. Ici, par exemple, deux images, un texte, puis deux autres images sont affichés à l'intérieur d'un élément div. Ces éléments se succèdent en ligne. Le passage à la ligne après le mot "templum" a été provoqué par un manque de place dans le navigateur

```

```

```

```

Iam summus Pater architectus Deus hanc quam videmus mundanam  
domum, divinitatis templum augustissimum, archanae legibus  
sapientiae fabrefecerat.

```

```

```

```



Iam summus Pater architectus Deus hanc quam videmus mundanam domum, divinitatis templum  
augustissimum, archanae legibus sapientiae fabrefecerat.



# Mise en page avec la propriété display

La propriété display permet de spécifier la façon dont un élément est rendu. Elle peut prendre de nombreuses valeurs. Voici les plus courantes :

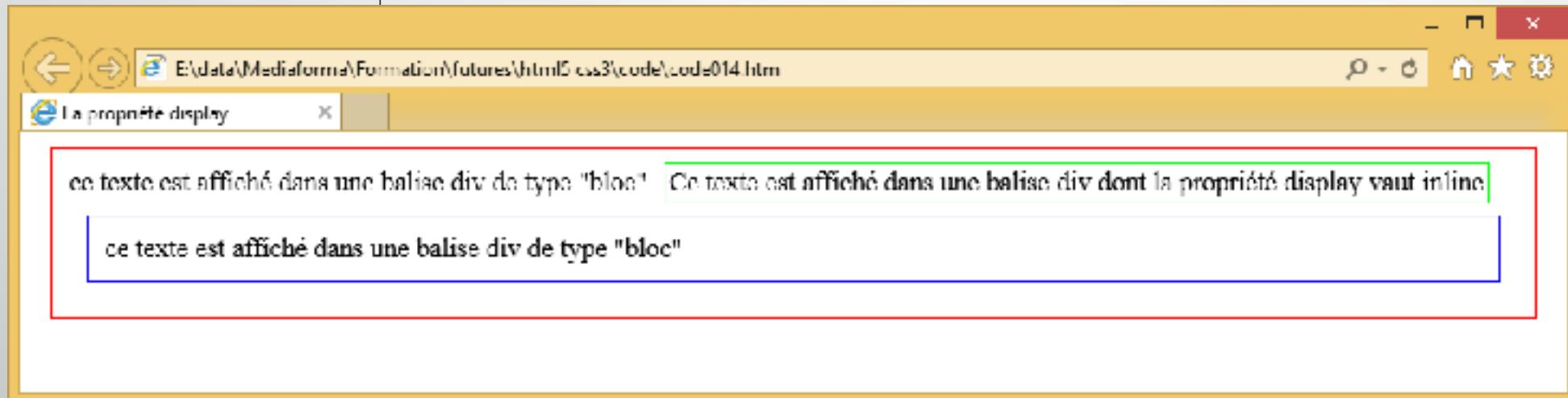
- none : l'élément n'est pas rendu.
- inline : l'élément est rendu sur la même ligne que son parent.
- block : l'élément est rendu en mode bloc.
- inline-block : l'élément se comporte comme une "boîte en ligne" : il peut être dimensionné (width, height) et avoir des marges (margin, padding), mais il se place également sur la ligne de texte de son parent. Vous utiliserez -moz-inline-box pour assurer la compatibilité avec Firefox 2 et supérieur.
- list-item : l'élément est affiché sous la forme d'une liste.
- table, inline-table, table-header-group, table-footer-group, table-row, table-row-group, table-column, table-column-group, table-cell, table-caption : l'élément se comporte comme les composants d'un tableau (attention, ces valeurs sont incompatibles avec Internet Explorer).

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>La propriété display</title>
    <style>
        .un
        {
            border: 1px solid #ff0000;
            margin: 10px;
            padding: 10px;
        }
    </style>

```

Premier exemple (code014.htm)

Saisissez ce code.  
Qu'obtenez-vous ?

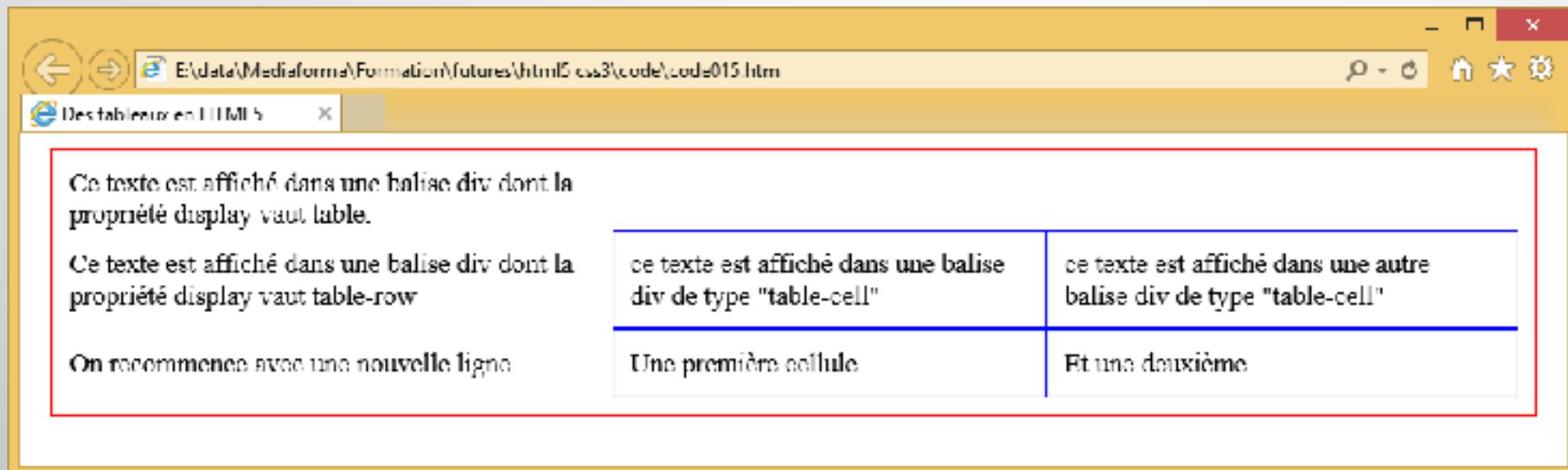


```
    </head>
    <body>
        <div class="un" style="display: block;">
            ce texte est affiché dans une balise div de type "bloc".
            <div class="deux" style="display: inline">Ce texte est affiché dans une balise div dont la
            propriété display vaut inline</div>
            <div class="trois">ce texte est affiché dans une balise div de type "bloc"</div>
        </div>
    </body>
</html>
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Des tableaux en HTML5</title>
    <style>
      .un
      {
        border: 1px solid #ff0000;
        margin: 10px;
        padding: 10px;
      }
      .deux
```

## Deuxième exemple (code015.htm)

Saisissez ce code  
Qu'obtenez-vous ?



```
      <div class="trois" style="display: table-cell">ce texte est affiché dans une autre balise div de type "table-cell"</div>
    </div>
    <div class="deux" style="display: table-row">On recommence avec une nouvelle ligne
      <div class="trois" style="display: table-cell">Une première cellule</div>
      <div class="trois" style="display: table-cell">Et une deuxième</div>
    </div>
  </body>
</html>
```

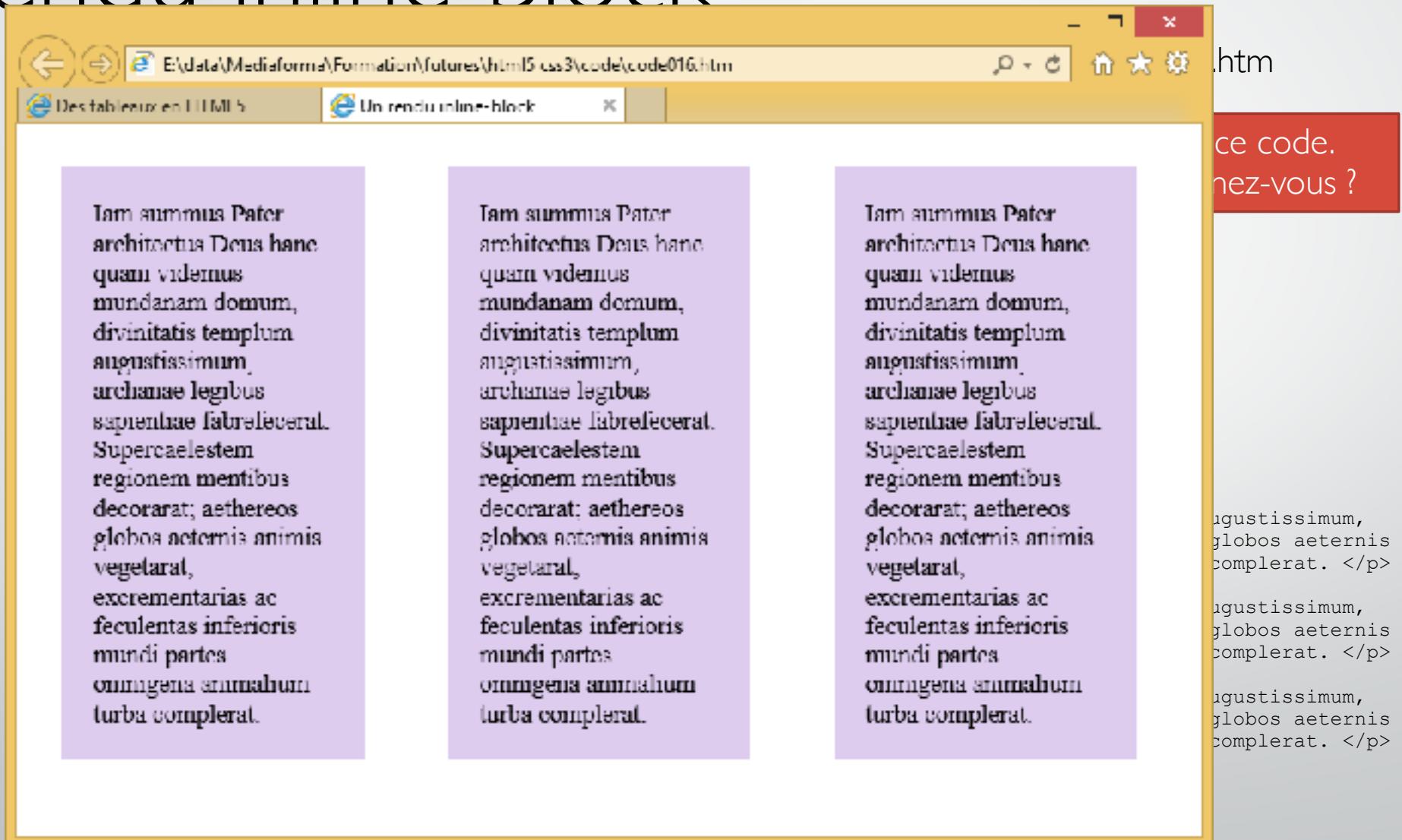
: display  
:cell" />

# Rendu inline-block

Le rendu inline-block profite des propriétés propres aux rendus inline et block. En effet, il permet aux éléments concernés de s'afficher côte à côte (rendu inline) et d'être dimensionnés et de recevoir des marges externes (rendu block).

Pour définir des blocs de texte de largeur fixe à l'aide de simples éléments p, rien de tel qu'un affichage inline-block.

# Rendu inline-block



.htm

ce code.  
nez-vous ?

agustissimum,  
globos aeternis  
complerat. </p>

agustissimum,  
globos aeternis  
complerat. </p>

agustissimum,  
globos aeternis  
complerat. </p>

# Rendu inline-block

Cette technique ne se limite pas aux éléments p.

Elle peut être étendue à tous les éléments dont le rendu par défaut est inline. Essentiellement a, b, br, code, em, font, i, img, input, s, select, small, span, strike, strong, sub, sup, textarea et u.

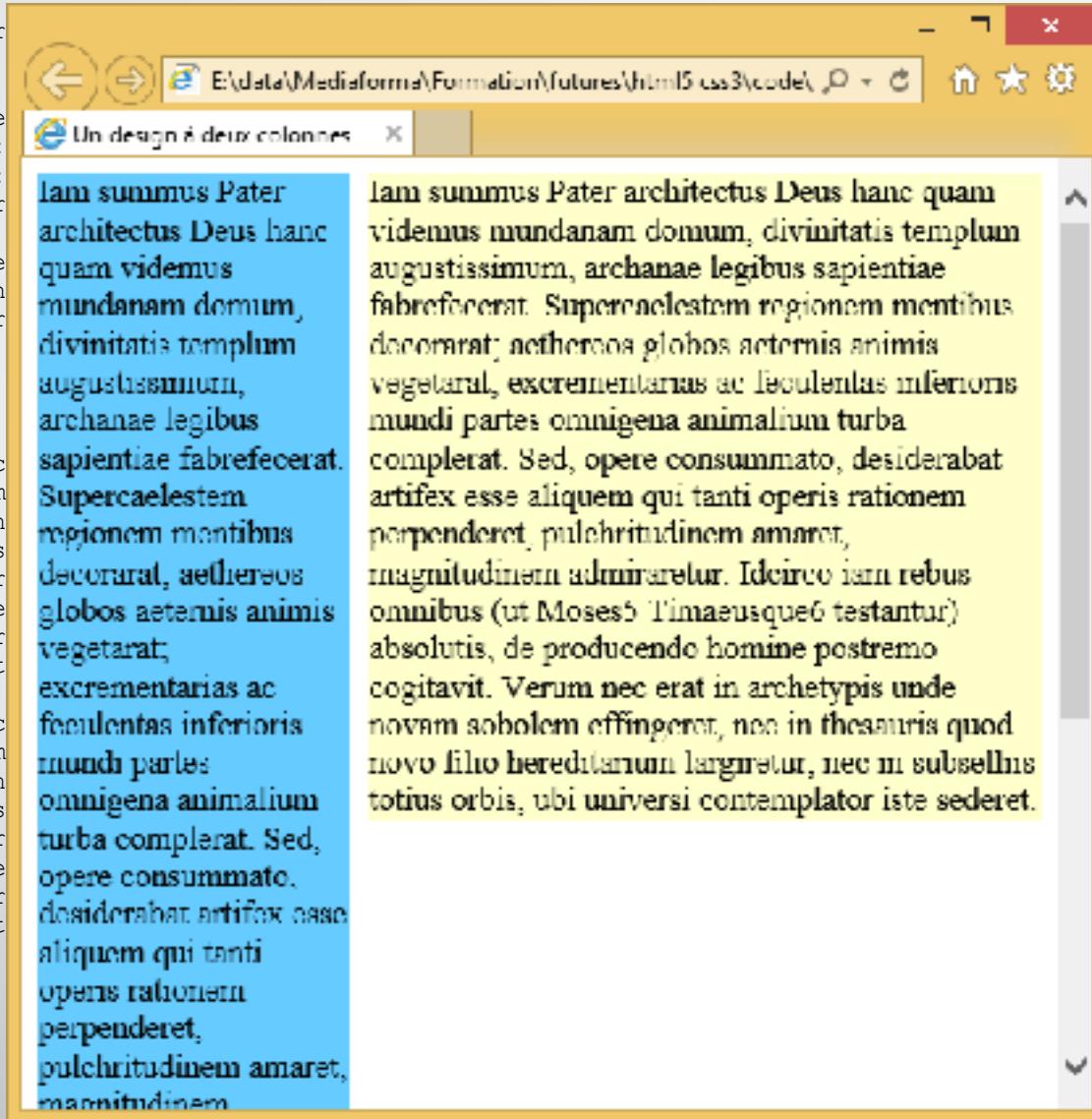
# Design à deux colonnes

Le principe consiste à définir une boîte <div> flottante (à gauche ou à droite selon les besoins) et une zone non flottante, dont la marge est légèrement supérieure à la largeur de l'élément flottant.

Dans cet exemple, nous définissons une "colonne" flottante à gauche de largeur 150 pixels, et une colonne non flottante qui occupe le reste de la page. Cette colonne a une marge gauche de 160 pixels, de façon à se détacher légèrement vers la droite de la colonne gauche.

# Design à deux colonnes

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Un design à deux colonnes</title>
    <style>
      #colonne_gauche {
        float: left;
        width: 300px;
        background-color: #f0f0ff;
      }
      #colonne_droite {
        margin-left: 300px;
        background-color: #fffff0;
      }
    </style>
  </head>
  <body>
    <div id="colonne_gauche">
      Iam summus Pater architectus Deus hanc quam videnus mundanam domum, divinitatis templum augustissimum, archanae legibus sapientiae fabrefecerat. Supercaelestem regionem mentibus decorarat; aetheros globos aeternis animis vegetarum, excrementarias ac feculentas inferiores mundi partes omnigena animalium turba complerat. Sed, opere consummato, desiderabat artifex esse aliquem qui tanti operis rationem perpendere, pulchritudinem amaret, magnitudinem admiraretur. Idcirco iam rebus omnibus (ut Moses5 Timaeus6 testantur) absolutis, de producendo homine postremo cogitavit. Verum nec erat in archetypis unde novam sibolem effingere, non in thesauris quod novo filio hereditarium largiretur, nec in subsellis totius orbis, ubi universi contemplator iste sederet.
    </div>
    <div id="colonne_droite">
      Iam summus Pater architectus Deus hanc quam videnus mundanam domum, divinitatis templum augustissimum, archanae legibus sapientiae fabrefecerat. Supercaelestem regionem mentibus decorarat; aetheros globos aeternis animis vegetarum, excrementarias ac feculentas inferiores mundi partes omnigena animalium turba complerat. Sed, opere consummato, desiderabat artifex esse aliquem qui tanti operis rationem perpendere, pulchritudinem amaret, magnitudinem admiraretur. Idcirco iam rebus omnibus (ut Moses5 Timaeus6 testantur) absolutis, de producendo homine postremo cogitavit. Verum nec erat in archetypis unde novam sibolem effingere, non in thesauris quod novo filio hereditarium largiretur, nec in subsellis totius orbis, ubi universi contemplator iste sederet.
    </div>
  </body>
</html>
```



code016.htm

Saisissez ce code.  
Qu'obtenez-vous ?

s templum augustissimum, archanae os globos aeternis animis vegetarat; rat. Sed, opere consummato, em amaret, magnitudinem admiraretur. homine postremo cogitavit. Verum io hereditarium largiretur, nec in

s templum augustissimum, archanae os globos aeternis animis vegetarat; rat. Sed, opere consummato, em amaret, magnitudinem admiraretur. homine postremo cogitavit. Verum io hereditarium largiretur, nec in

# Design à 3 colonnes dont 2 flottantes

Pour réaliser un design à trois colonnes dont deux sont flottantes, ancrées sur les bords gauche et droit de la page, vous utiliserez trois éléments div :

- un premier de largeur fixe (width), ancré sur le bord gauche de la page (float: left) ;
- un deuxième de largeur fixe (width), ancré sur le bord droit de la page (float: right) ;
- un troisième qui occupera librement l'espace situé entre les deux autres.

## Exercice

Ecrivez le code pour réaliser un design à 3 colonnes :

- Colonne 1 : 150 px flottante à gauche
- Colonne 2 : 300 px flottante à droite
- Colonne 3 : La place restante

# Solution

```
<!DOCTYPE html>
<html>
  <head>
```



E:\data\Mediaforma\Formation\futures\html5\css3\code\code018.htm

Un design fluid à 3 colonnes

Iam summus Pater architectus Deus hanc quam videmus mundanam domum, divinitatis templum augustissimum, archanae legibus sapientiae fabrefecerat. Supercaelestem regionem montibus decorarat; aethereos globos aeternis annis vegetarunt; excrementarias ac feculentas inferioris mundi partes omnigena animalium turba complerat. Sed, opere consummato, desiderabat artifex esse aliquem qui tanti operis rationem perpendaret, pulchritudinem amaret, magnitudinem admiraretur. Idcirco iam rebus omnibus (ut Moses⁹ Timaeusque⁹ testantur) absolutis, de producendo homine postremo cogitavit. Verum nec erat in archetypis unde novam sobolem effingeret, nec in thesauris quod novo filio hereditarium largiretur, nec in subsellis totius orbis, ubi universi contemplator iste sederet.

Iam summus Pater architectus Deus hanc mundanam domum, divinitatis templum augustissimum, quam videmus mundanam domum, divinitatis archanae legibus sapientiae fabrefecerat. Supercaelestem regionem montibus decorarat; aethereos globos aeternis annis vegetarunt; excrementarias ac feculentas inferioris mundi partes omnigena animalium turba complerat. Sed, opere consummato, desiderabat artifex esse aliquem qui tanti operis rationem perpendaret, pulchritudinem amaret, magnitudinem admiraretur. Idcirco iam rebus omnibus (ut Moses⁹ Timaeusque⁹ testantur) absolutis, de producendo homine postremo cogitavit. Verum nec erat in archetypis unde novam sobolem effingeret, nec in thesauris quod novo filio hereditarium largiretur, nec in subsellis totius orbis, ubi universi contemplator iste sederet.

Iam summus Pater architectus Deus hanc templum augustissimum, archanae legibus sapientiae fabrefecerat. Supercaelestem regionem montibus decorarat; aethereos globos aeternis annis vegetarunt; excrementarias ac feculentas inferioris mundi partes omnigena animalium turba complerat. Sed, opere consummato, desiderabat artifex esse aliquem qui tanti operis rationem perpendaret, pulchritudinem amaret, magnitudinem admiraretur. Idcirco iam rebus omnibus (ut Moses⁹ Timaeusque⁹ testantur) absolutis, de producendo homine postremo cogitavit. Verum nec erat in archetypis unde novam sobolem effingeret, nec in thesauris quod novo filio hereditarium largiretur, nec in subsellis totius orbis, ubi universi contemplator iste sederet.

chanae legibus sapientiae arias ac feculentas quem qui tanti operis aeusque⁹ testantur) nec in thesauris quod

anae legibus sapientiae arias ac feculentas quem qui tanti operis aeusque⁹ testantur) nec in thesauris quod

chanae legibus sapientiae arias ac feculentas quem qui tanti operis aeusque⁹ testantur) nec in thesauris quod

# Design à 3 colonnes de même hauteur

Pour obtenir des colonnes de même hauteur, et ce, indépendamment de leur contenu, vous utiliserez la technique dite des "colonnes factices", qui consiste à :

1. Insérer les boîtes des colonnes dans un conteneur commun.
2. Définir un arrière-plan graphique qui se répète en hauteur dans le conteneur, de façon à tracer un séparateur entre les colonnes.
3. Modifier les marges de la colonne centrale pour obtenir un léger décalage de part et d'autre des colonnes séparatrices.
4. Conférer au conteneur un nouveau contexte de formatage *via* la déclaration overflow: hidden, ce qui permet à ce conteneur d'englober ses enfants flottants.

A titre d'exemple, nous allons définir une colonne gauche de largeur 160 pixels, une colonne droite de largeur 300 pixels et une colonne centrale qui occupe le reste de l'espace disponible, soit  $800 - 160 - 300 = 340$  pixels.

L'arrière-plan est une image de 800 pixels sur 3 pixels :



E:\data\Mediaformat\Formation\francais\html5.css2\code\code010.htm

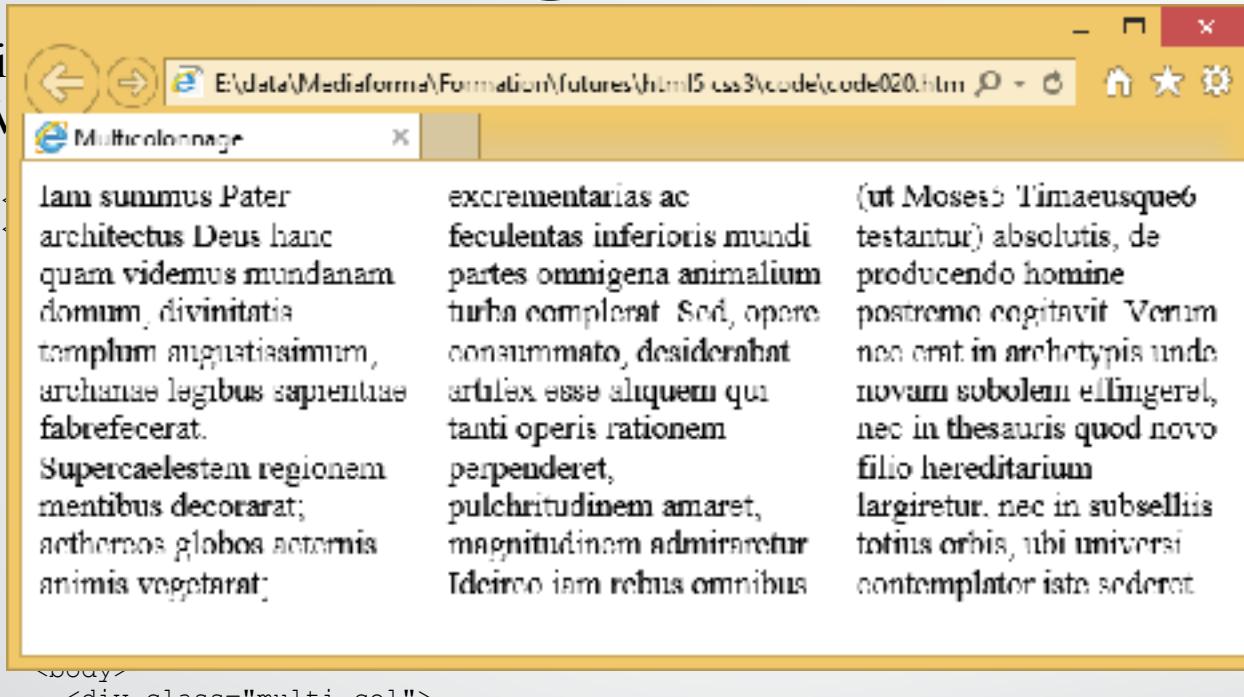
Un design à 3 colonnes de...

Iam summus Pater architectus Deus hanc quam videmus mundanam dominum, divinitatis templum augustissimum, archanae legibus sapientiae fabrefecerat. Supercaelestem regionem mentibus decorarat; aethereos globos aeternis animis vegetarunt; excrementarias ac feculentas inferioris mundi partes omnigena animalium turba complerat. Sed, opere consummato, desiderabat artifex esse aliquem qui tanti operis rationem perpendere, pulchritudinem amaret, magnitudinem admiraretur. Idcirco iam rebus omnibus (ut Moses <sup>5</sup> Timaeusque <sup>6</sup> testantur) absolutis, de producendo homine postremo cogitavit. Verum nec erat in archetypis unde novam sibolem effingeret, nec in thesauris quod novo filio hereditarium largiretur, nec in subsellis totius orbis, ubi universi contemplator iste sederet.	Iam summus Pater architectus Deus hanc quam videmus mundanam dominum, divinitatis templum augustissimum, archanae legibus sapientiae fabrefecerat. Supercaelestem regionem mentibus decorarat; aethereos globos aeternis animis vegetarunt; excrementarias ac feculentas inferioris mundi partes omnigena animalium turba complerat. Sed, opere consummato, desiderabat artifex esse aliquem qui tanti operis rationem perpendere, pulchritudinem amaret, magnitudinem admiraretur. Idcirco iam rebus omnibus (ut Moses <sup>5</sup> Timaeusque <sup>6</sup> testantur) absolutis, de producendo homine postremo cogitavit. Verum nec erat in archetypis unde novam sibolem effingeret, nec in thesauris quod novo filio hereditarium largiretur, nec in subsellis totius orbis, ubi universi contemplator iste sederet.	Iam summus Pater architectus Deus hanc quam videmus mundanam dominum, divinitatis templum augustissimum, archanae legibus sapientiae fabrefecerat. Supercaelestem regionem mentibus decorarat; aethereos globos aeternis animis vegetarunt; excrementarias ac feculentas inferioris mundi partes omnigena animalium turba complerat. Sed, opere consummato, desiderabat artifex esse aliquem qui tanti operis rationem perpendere, pulchritudinem amaret, magnitudinem admiraretur. Idcirco iam rebus omnibus (ut Moses <sup>5</sup> Timaeusque <sup>6</sup> testantur) absolutis, de producendo homine postremo cogitavit. Verum nec erat in archetypis unde novam sibolem effingeret, nec in thesauris quod novo filio hereditarium largiretur, nec in subsellis totius orbis, ubi universi contemplator iste sederet.
--	--	--

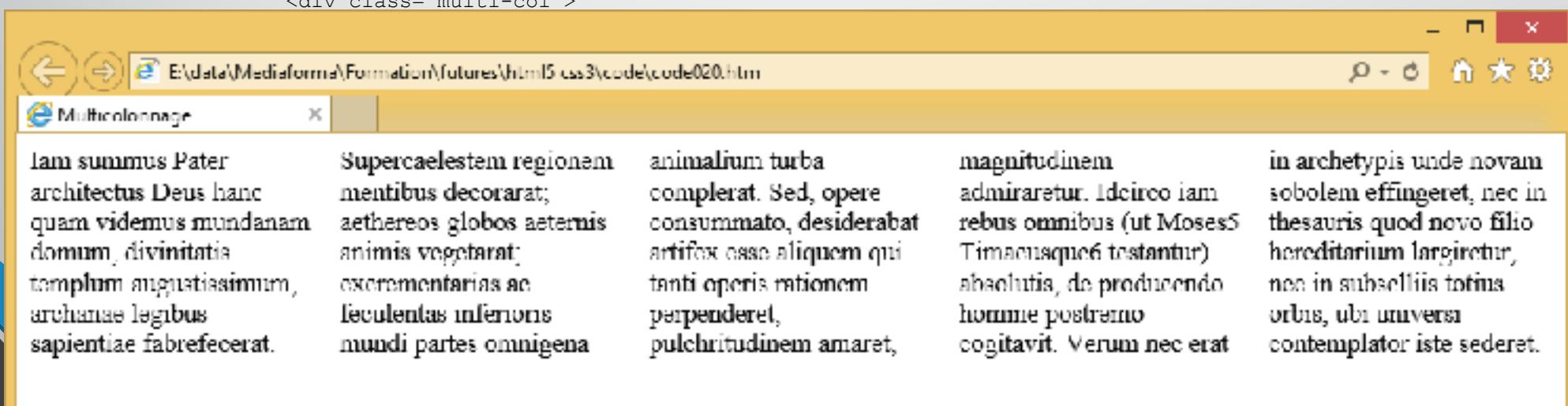
# Multicolonnage CSS3

Pour définir les colonnes. V

es et l'espace entre deux



```
<body>
<div class="multi-col">
```



# Multicolonnage CSS3

Si le nombre de colonnes doit rester fixe, utilisez les propriétés CSS3 column-count, -moz-column-count et -webkit-column-count. Par exemple, pour définir trois colonnes, quelle que soit la largeur de la fenêtre, ajoutez les deux lignes suivantes dans la classe multi-col :

```
column-count: 3;  
-moz-column-count: 3;  
-webkit-column-count: 3;
```

## Exercice

Modifiez le code précédent pour que le nombre de colonne soit fixe et égal à 4.

# Solution

Modifiez le style .multi-col comme ceci :

```
<style>
  .multi-col
  {
    column-width: 150px;
    -moz-column-width: 150px;
    -webkit-column-width: 150px;
    -moz-column-gap: 20px;
    -webkit-column-gap: 20px;
    column-count: 4;
    -moz-column-count: 4;
    -webkit-column-count: 4;
  }
</style>
```

# Multicolonnage CSS3

Si nécessaire, vous pouvez également insérer un trait séparateur entre les colonnes en utilisant les propriétés CSS3 column-rule, -moz-column-rule et -webkit-column-rule dont voici la syntaxe :

```
column-rule: largeur style couleur;  
-moz-column-rule: largeur style couleur;  
-webkit-column-rule: largeur style couleur;
```

Où :

- largeur est la largeur du trait en pixels.
- style est le style du trait. Il peut prendre l'une des valeurs suivantes : dotted, dashed, solid, insert, double, groove, ridge ou outset.
- couleur est la couleur du trait.

Par exemple, pour définir un trait séparateur continu noir et d'épaisseur 1 pixel, vous utiliserez les deux propriétés suivantes :

```
column-rule: 1px solid black;  
-moz-column-rule: 1px solid black;  
-webkit-column-rule: 1px solid black;
```

Entraînez-vous à utiliser ces instructions dans le code précédent.

# Multicolonnage CSS3

Pour terminer, sachez qu'il est possible de définir la couleur d'arrière-plan du texte avec la propriété background et l'alignement du texte dans les colonnes avec la propriété text-align. Par exemple, pour affecter un arrière-plan gris et pour justifier le texte dans les colonnes, servez-vous des propriétés suivantes :

```
background:#ccc;  
text-align:justify;
```

# Exercice

En utilisant le code HTML précédent, définissez un multicolonnage fixe sur trois colonnes séparées entre elles par un trait de 1 pixel, justifiées et d'arrière-plan #ccc.

## Gestion des débordements de contenus

Qu'il s'agisse de texte, d'images ou de tout autre type de contenu, les éléments placés dans un conteneur peuvent déborder de ce conteneur. Vous devez alors gérer les débordements, en particulier si le contenu peut provenir de sources que vous ne contrôlez pas.

## Examinez le code suivant :

```
<!DOCTYPE html>
<html>
  <head>
    <meta>
    <title></title>
    <style>
      #ma_box {
        width: 300px;
        height: 200px;
        border: 1px solid black;
        border-radius: 10px;
        background-color: #f0f0f0;
        padding: 10px;
      }
    </style>
  </head>
  <body>
    <div id="ma_box">
      Lorem ipsum dolor sit amet,  

      ce_mot_est_extrêmement_long_intentionnellement_pour_illustrer_la_coupure_de_mots  

      consectetur adipiscing elit, sed  

      do eiusmod tempor incididunt  

      ut labore et dolore magna  

      aliqua. Ut enim ad minim  

      veniam, quis nostrud  

      exercitation ullamco laboris  

      nisi ut aliquip ex ea commodo  

      consequat. Duis aute irure  

      dolor in reprehenderit in  

      voluptate velit esse cillum  

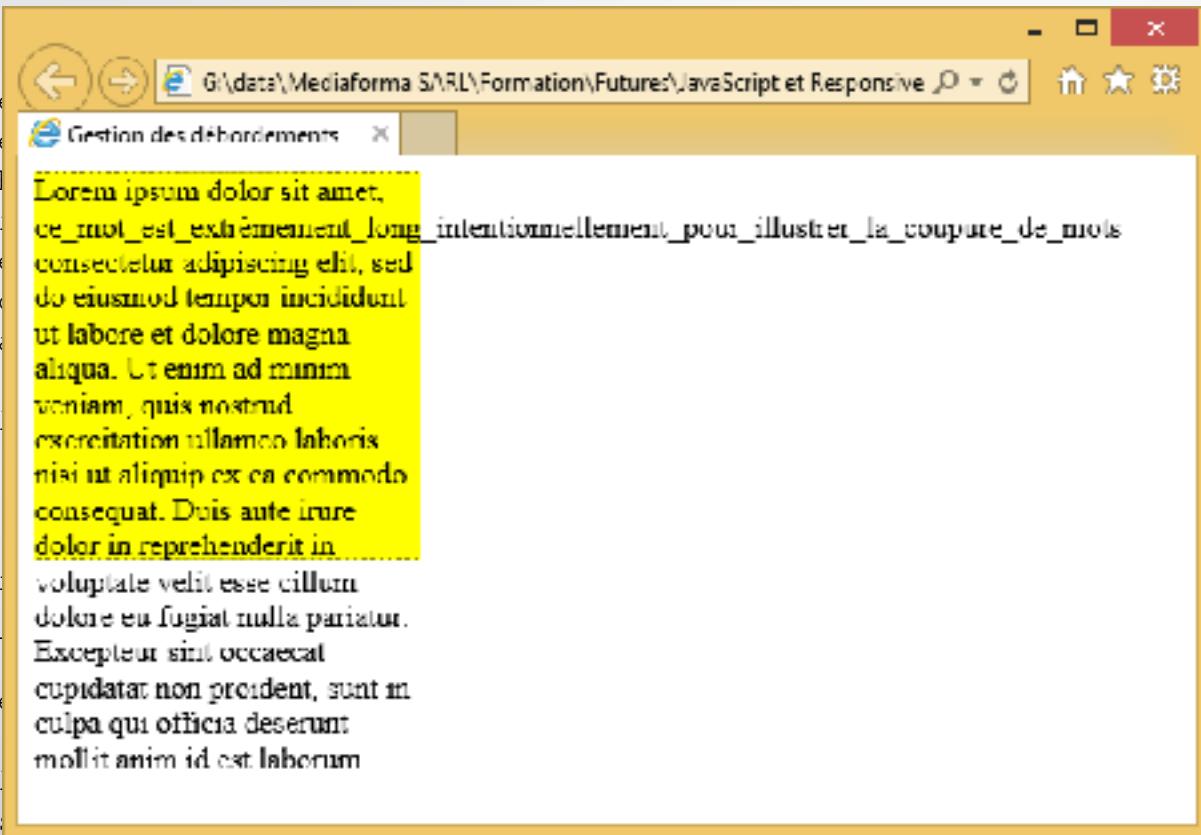
      dolore eu fugiat nulla pariatur.  

      Excepteur sint occaecat  

      cupidatat non proident, sunt in  

      culpa qui officia deserunt  

      mollit anim id est laborum
    </div>
  </body>
</html>
```



de\_mots consectetur  
na aliqua. Ut enim  
o ex ea commodo  
cillum dolore eu  
nt in culpa qui

Manifestement, la balise <div> conteneur n'est pas assez grande pour héberger tout le texte. Etant donné que ses dimensions ont été fixées "en dur" (width et height), le texte déborde.

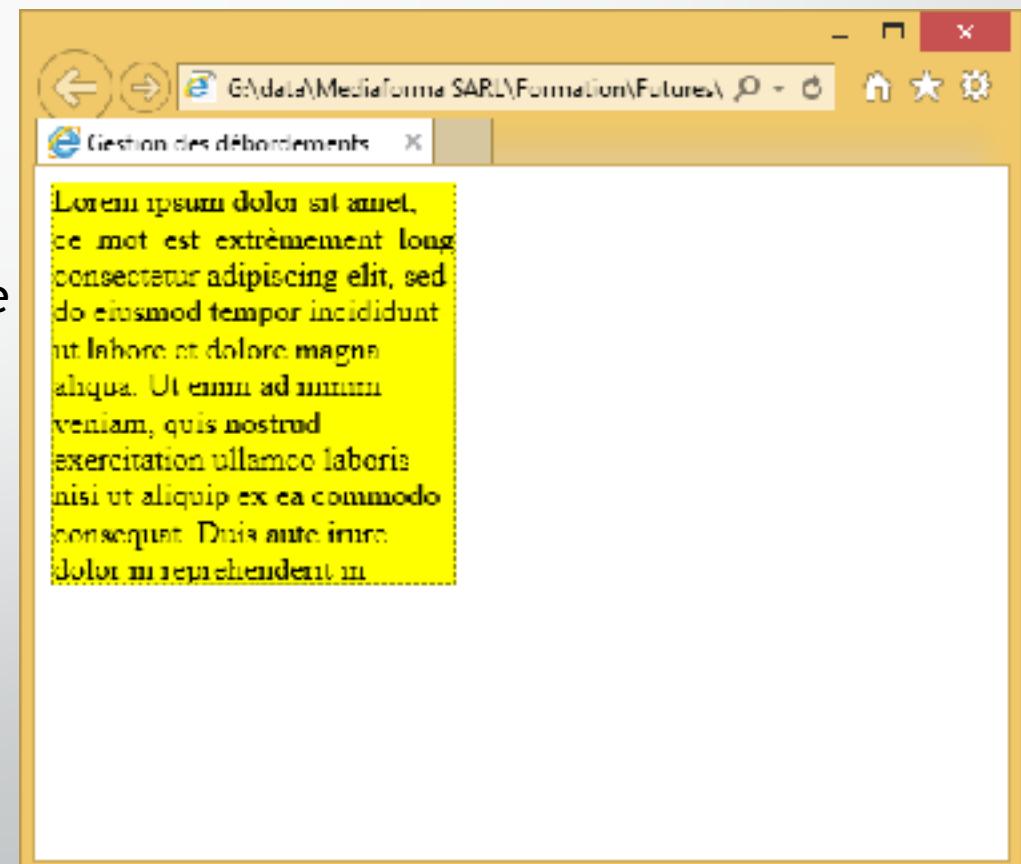
Plusieurs technique vont permettre de gérer les débordements.

## Première technique : overflow: hidden

Tout ce qui dépasse du conteneur est affacé avec cette instruction CSS :

```
#mabox {  
    overflow: hidden;  
}
```

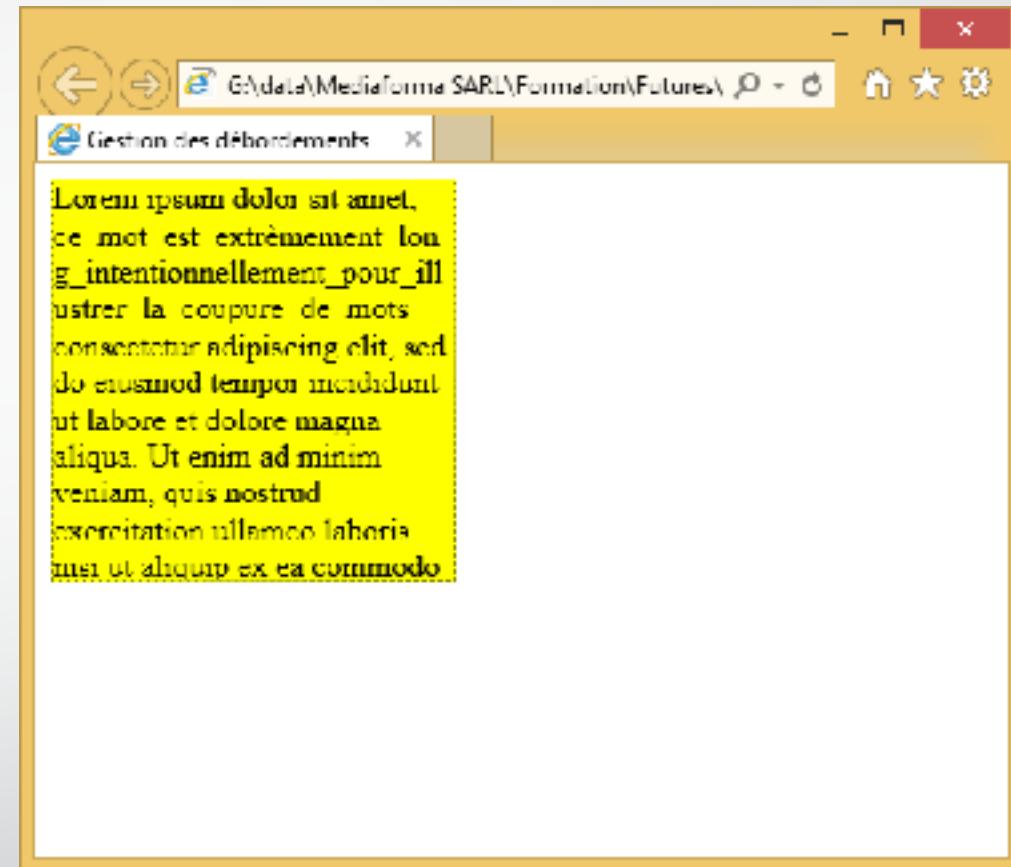
Voici le résultat :



## Deuxième technique : passage à la ligne des mots longs

Lorsqu'elle est initialisée à break-word, la propriété CSS overflow-wrap permet de découper proprement les mots trop longs pour le conteneur :

```
#mabox {  
    overflow: hidden;  
    word-wrap: break-word;
```



Comme vous pouvez le constater, le mot long n'est pas partiellement dissimulé par la propriété overflow: hidden;. Au contraire, le mot est renvoyé à la ligne autant de fois que nécessaire pour s'afficher dans son intégralité.

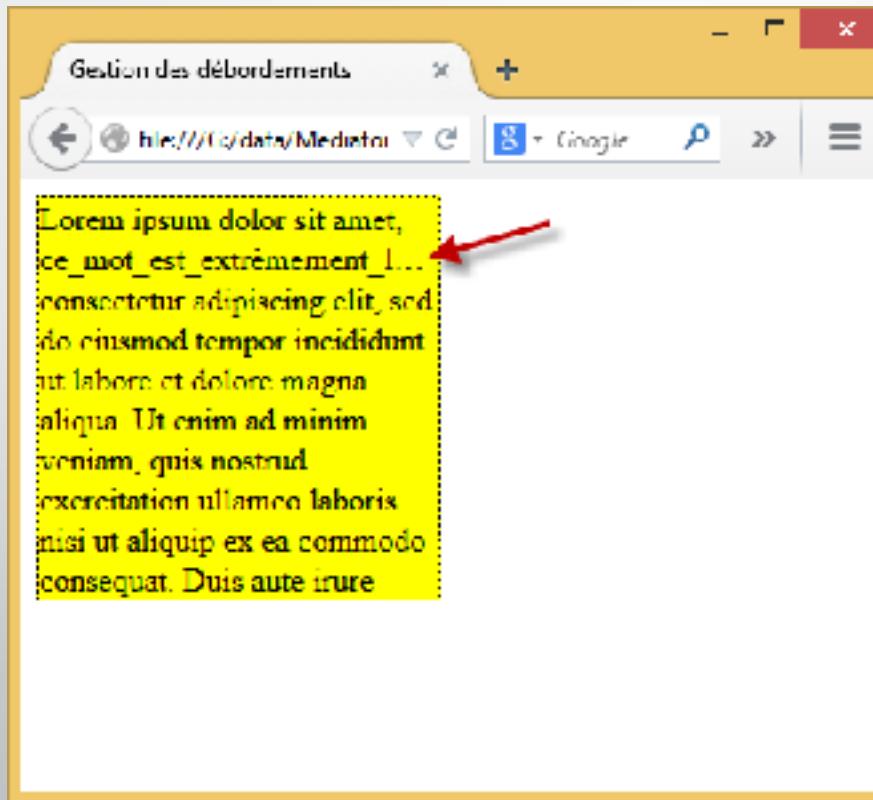
## Troisième technique : points de suspension

En affectant la valeur **ellipsis** à la propriété **text-overflow**, des points de suspension seront affichés chaque fois qu'un texte est rogné :

```
#mabox {  
    overflow: hidden;  
    text-overflow: ellipsis;
```

### Attention

Cette propriété n'est pas supportée par tous les navigateurs.



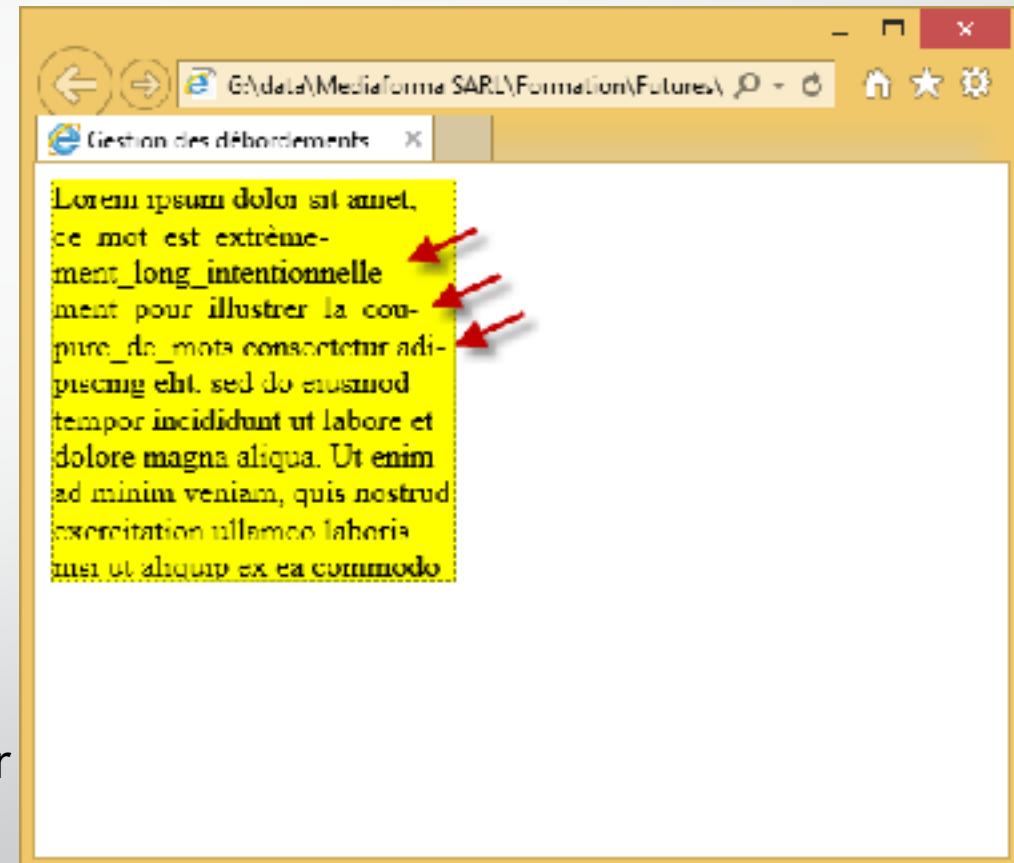
## Quatrième technique : césure de mots

Affectez la valeur **auto** à la propriété **hyphens** pour activer la césure de mots :

```
#mabox {  
    overflow: hidden;  
    hyphens: auto;  
    -ms-hyphens: auto;  
    -moz-hyphens: auto;  
    -webkit-hyphens: auto;
```

### Attention

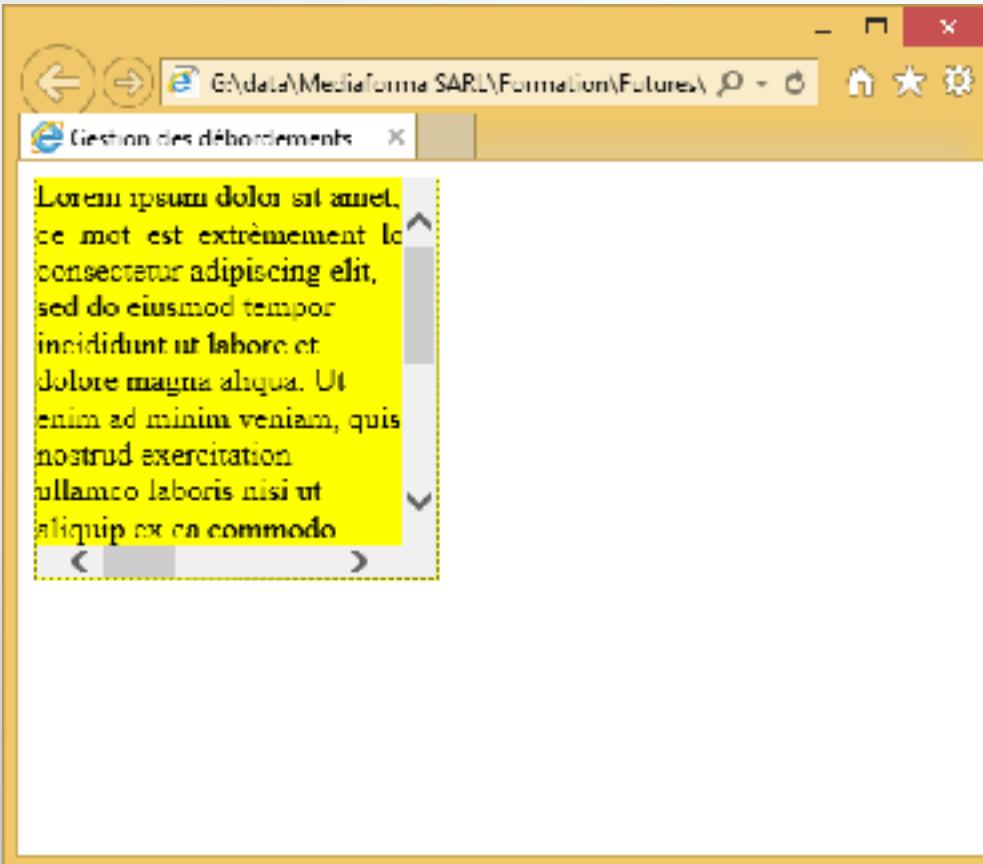
Cette propriété n'est pas encore supportée par tous les navigateurs. Pensez à utiliser les préfixes habituels pour améliorer sa compatibilité.



## Cinquième technique : barres de défilement

En affectant la propriété **auto** à la propriété **overflow**, des barres de défilement permettent d'accéder au contenu qui déborder du conteneur :

```
#mabox {  
    overflow: auto;
```

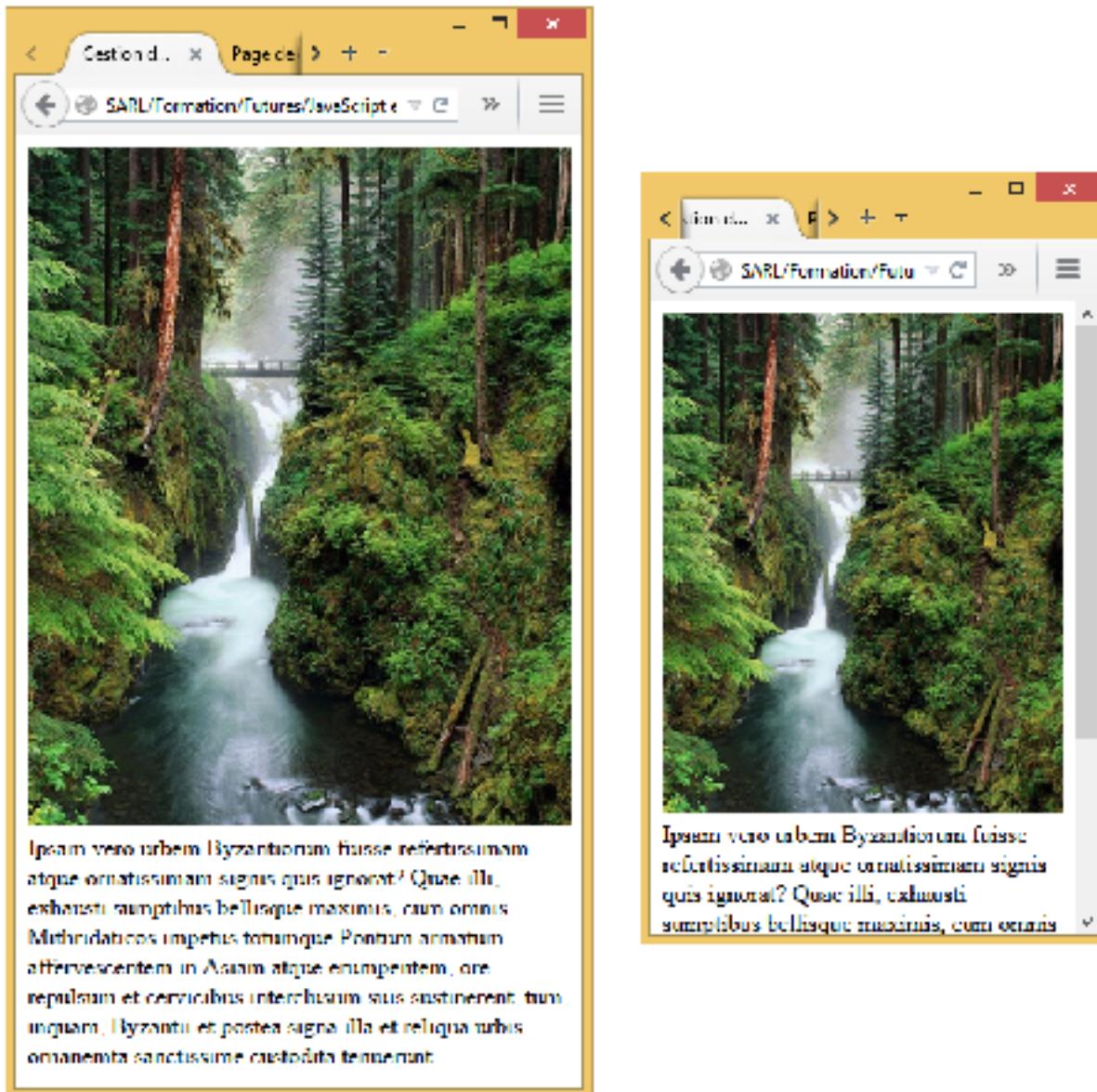


# Composants graphiques

## Image

Pour que la telle dispose, code :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Générer une image dans une fenêtre</title>
  </head>
  <body>
    
    Ipsam vero urbem Byzantium fuisse refertissimum atque ornatissimum signis quis ignorat? Quae illi, exhausti sumptibus bellisque maximis, cum omnis Mithridaticos impebus totumque Pontum armatum affervescentem in Asiam atque enumpentem, ore repulsum et cervicibus interclusum suis sustinerent: tum inquam, Byzantii et postea signa illa et reliqua turbis romanem in sanctissime castellis tenuerunt.
  </body>
</html>
```



Redimensionnez la fenêtre. Comme vous pouvez le voir, l'image se redimensionne automatiquement en fonction des dimensions de la fenêtre :

ction de l'espace dont il. Voici un exemple de

gnis quis ignorat? Quae totumque Pontum armatum usum suis sustinerent, ssime custodita

# Arrière-plan de page fluide

Pour ce faire :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Arrière-plan de la page fluide</title>
  </head>
  <body>
    <h1>Arrière-plan de la page fluide</h1>
  </body>
</html>
```



dépendante :

# Affichage fluide, Grid Layout et FlexBox

## Principe d'un affichage fluide

On parle d'**affichage fluide** lorsque le contenu et la disposition sont élastiques. Pour arriver à ce résultat, on utilise des blocs dont la largeur est exprimée en pourcentages de la largeur totale de la fenêtre (ordinateur) ou de l'écran (mobile).

## Exercice

Définissez quelques lignes de HTML5 et de CSS3 pour obtenir la mise en page suivante :



Voici la largeur des différents blocs :

- Couleur jaune : 20% ;
- Couleur bleue : 80% ;
- Couleurs rouget et verte : 100%.

# Solution

Le code HTML5 est très simple : il met en place quatre zones distinctes : **header**, **article**, **aside** et **footer** :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Un document HTML5 avec en-tête, bas de page, zone de contenu et zone annexe</title>
    <link rel="stylesheet" href="styles004.css">
  </head>
  <body>
    <header>
      <h1>En-tête du document</h1>
      Texte de l'en-tête
    </header>
    <article>
      <h2>Premier article</h2>
      <p>Texte du premier article</p>
    </article>
    <aside>
      Texte affiché dans la partie droite de la page avec la balise &lt;aside&gt;<br><br>
    </aside>
    <footer>
      <p>Copyright et e-mail du Webmaster</p>
    </footer>
  </body>
</html>
```

Le code CSS3 est également très simple. Il indique que les quatre zones seront affichées en mode block, puis il définit les caractéristiques de chaque zone :

```
header, article, footer, aside {  
    display: block;  
}  
  
header {  
    background-color: red;  
}  
article {  
    height: 300px;  
    float: left;  
    background-color: #66FFFF;  
    width: 80%;  
}  
  
aside {  
    height: 300px;  
    float: right;  
    width: 20%;  
    background-color: yellow;  
}  
  
footer {  
    clear: both;  
    background-color: #99FF66;  
}
```

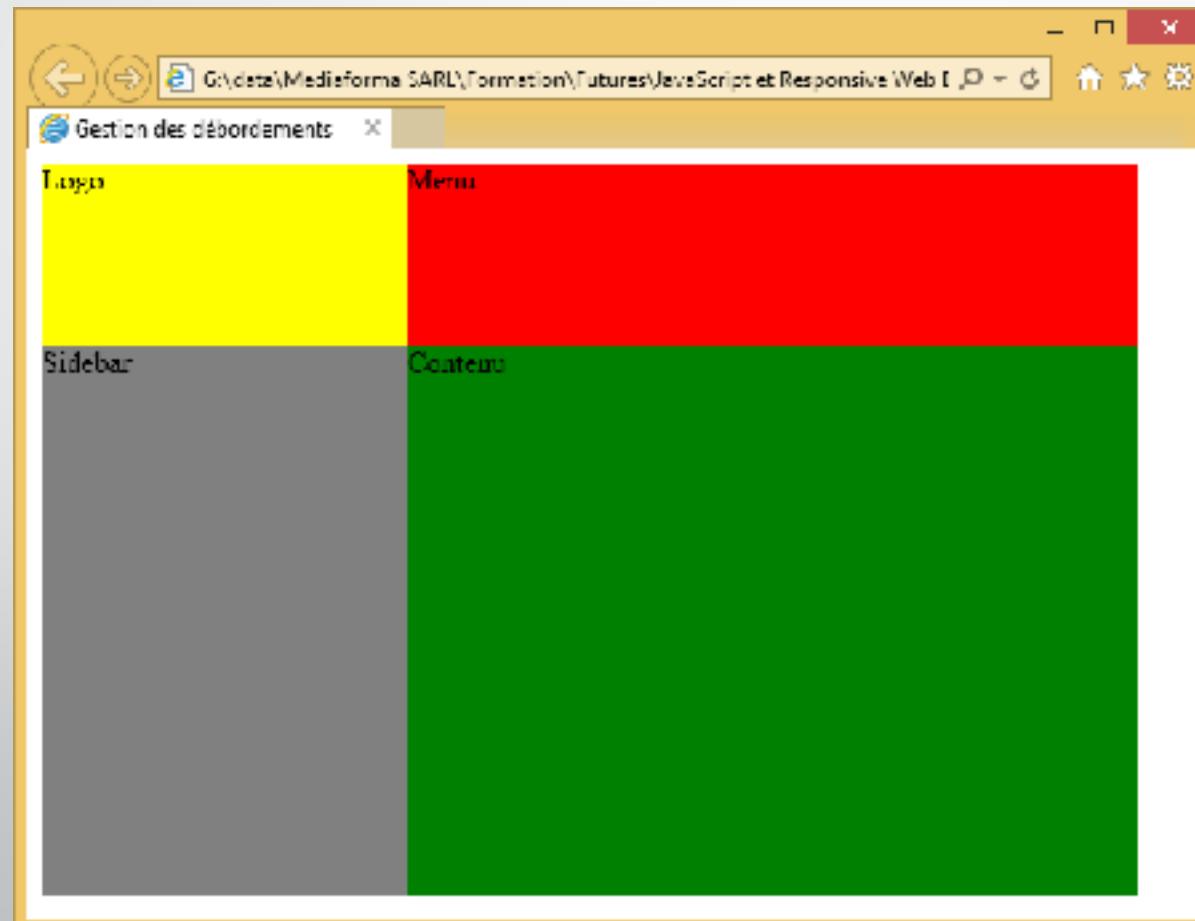
Vérifiez que l'affichage s'adapte bien à l'espace horizontal disponible en redimensionnant la fenêtre du navigateur.

## Remarque

Il est possible de définir les marges extérieures (**margin**) et intérieures (**padding**) d'un élément HTML5 quelconque (**img**, **div**, **nav**, etc.) sous la forme d'un pourcentage de leur conteneur pour leur conférer un caractère responsive.

## Principe d'un affichage Grid Layout

Le principe de fonctionnement du Grid layout consiste à diviser l'écran en plusieurs zones, comme dans un tableau, à ceci près que le découpage se fait en CSS et non en HTML. Voici un exemple de découpage Grid layout (propre à Internet Explorer et Microsoft Edge) :



Par exemple, pour découper la page en quatre zones (logo, menu, sidebar et contenu) comme dans la figure précédente, vous utiliserez ce code :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Gestion des débordements</title>
    <style>
      body {
        display: -ms-grid;
        -ms-grid-columns: 200px 400px;
        -ms-grid-rows: 100px 300px;
      }
      header, menu, aside, article { display: block; margin: 0; padding: 0; }
      header { -ms-grid-column: 1; -ms-grid-row: 1; background-color: yellow; }
      menu { -ms-grid-column: 2; -ms-grid-row: 1; background-color: red; }
      aside { -ms-grid-column: 1; -ms-grid-row: 2; background-color: grey; }
      article { -ms-grid-column: 2; -ms-grid-row: 2; background-color: green; }

    </style>
  </head>
  <body>
    <header>
      Logo
    </header>
    <menu>
      Menu
    </menu>
    <aside>
      Sidebar
    </aside>
    <article>
      Contenu
    </article>
  </body>
</html>
```

## Principe de la grille flexible (flexbox)

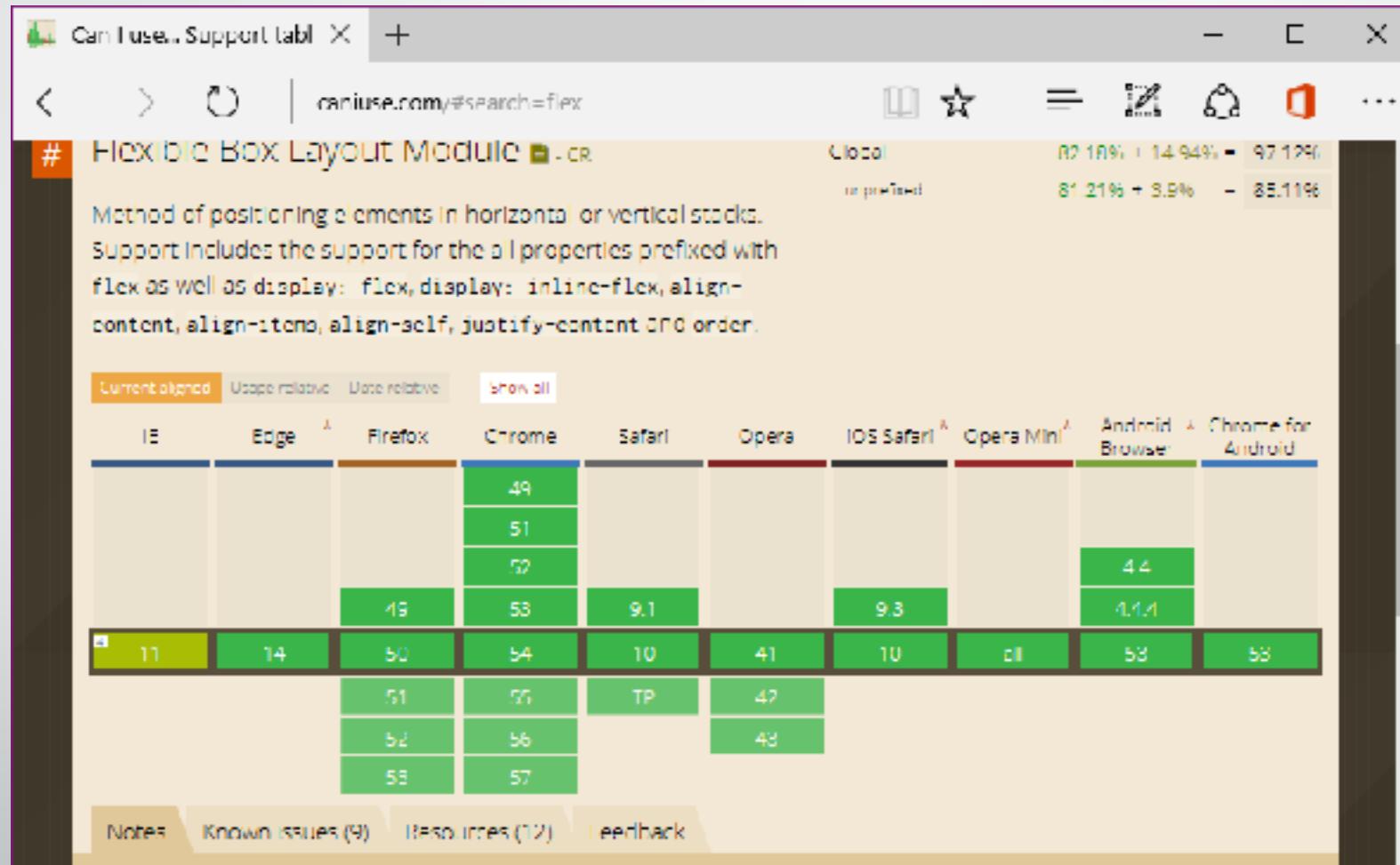
L'affichage **Flexbox** est un nouveau type de rendu défini dans la propriété CSS **display** :

```
display: flex;
```

Il permet de gérer la mise en page de plusieurs éléments enfants dans un élément parent. Avec un affichage flexbox, vous pouvez entre autres :

- mettre en place plusieurs éléments sur une ligne sans vous soucier de leur taille et/ou de leurs marges ;
- modifier le sens de lecture vertical/vertical ;
- aligner les éléments à gauche, à droite ou au centre du parent ;
- modifier l'ordre d'affichage des éléments enfants.

Le mode d'affichage flex est utilisable sur tous les navigateurs récents. Pour vous assurer de sa compatibilité avec votre navigateur, allez sur la page <http://caniuse.com/#search=flex> :



## Affichage flexbox – display: flex

Pour mettre en place un affichage flexbox, commencez par définir le mode d'affichage **flex** pour le conteneur :

```
.conteneur { display: flex; }
```

## En ligne ou en colonne – flex-flow

Si vous voulez que les enfants s'affichent en ligne, affectez la valeur **row** à la propriété **flex-flow** du conteneur :

```
.conteneur {  
    display: flex;  
    flex-flow: row;  
}
```

Si vous voulez que les enfants s'affichent en colonne, affectez la valeur **column** à la propriété **flex-flow** du conteneur :

```
.conteneur {  
    display: flex;  
    flex-flow: column;  
}
```

## Direction de l'affichage – flex-direction

La propriété **flex-direction** définit l'ordre d'affichage des enfants dans le parent. Elle peut prendre les valeurs suivantes :

Valeur de flex-direction	Signification
row	de gauche à droite
row-reverse	de droite à gauche
column	de haut en bas
column-reverse	de bas en haut

## Wrapping des enfants – flex-wrap

Par défaut, les enfants essayeront de s'afficher sur une même ligne (ou une même colonne). Vous pouvez modifier ce comportement en agissant sur la propriété `flex-wrap` :

Valeur de <code>flex-wrap</code>	Signification
<code>nowrap</code>	Sur une seule ligne ou une seule colonne
<code>wrap</code>	Sur plusieurs lignes (ou plusieurs colonnes) si nécessaire
<code>wrap-reverse</code>	Sur plusieurs lignes (ou plusieurs colonnes) si nécessaire, dans l'ordre inverse d'affichage

## Alignment des enfants sur l'axe principal – justify-content

La propriété `justify-content` définit l'alignment des enfants sur l'axe principal (horizontal ou vertical selon la valeur affectée à la propriété `flex-flow`). Elle peut prendre les valeurs suivantes :

Valeur de <code>justify-content</code>	Signification
<code>flex-start</code>	Alignment au début du conteneur
<code>flex-end</code>	Alignment à la fin du conteneur
<code>center</code>	Centré dans le conteneur
<code>space-between</code>	Espacement entre les enfants
<code>space-around</code>	Espacement avant, entre et après les enfants

## Alignement des enfants sur l'axe secondaire– align-items

La propriété align-items définit l'alignement des enfants sur l'axe secondaire (l'axe perpendiculaire à l'axe principal). Elle peut prendre les valeurs suivantes :

Valeur de align-items	Signification
<b>flex-start</b>	Alignement au début du conteneur
<b>flex-end</b>	Alignement à la fin du conteneur
<b>center</b>	Centré dans le conteneur
<b>stretch</b>	Etiré pour occuper toute la dimension de l'axe secondaire
<b>baseline</b>	Aligné sur la baseline

## Alignment des enfants sur l'axe secondaire– align-content

La propriété align-content définit l'alignment des enfants sur l'axe secondaire (l'axe perpendiculaire à l'axe principal). Elle n'a aucun effet lorsque les enfants occupent une seule ligne. Cette propriété peut prendre les valeurs suivantes :

Valeur de align-items	Signification
flex-start	Alignment au début du conteneur
flex-end	Alignment à la fin du conteneur
center	Centré dans le conteneur
stretch	Etiré pour occuper toute la dimension de l'axe secondaire
space-between	Espacement entre les enfants
space-around	Espacement avant, entre et après les enfants

## Ordre d'affichage d'un enfant - order

Par défaut, les enfants s'affichent dans l'ordre où ils apparaissent dans le code.

Cependant, vous pouvez modifier cet ordre en définissant la propriété **order** dans un des enfants :

```
#enfant3 { order: 5; }
```

## Taille des enfants – flex-grow

La propriété **flex-grow** définit la taille des éléments affichés sur une même ligne (ou une même colonne). Si tous les éléments d'une même ligne ont une propriété **flex-grow** initialisée à 1, ils auront la même largeur. Si l'un d'entre eux a une propriété **flex-grow** initialisée à 2 et les autres à 1, il aura une largeur deux fois plus grande que les autres éléments.

## Rétrécissement des enfants – flex-shrink

La propriété **flex-shrink** indique si un élément peut être rétréci.

## Taille d'un élément – flex-basis

La propriété **flex-basis** définit la taille par défaut d'un élément avant que l'espace restant ne soit distribué :

```
#enfant2 { flex-basis: 200px; }
```

ou :

```
#enfant2 { flex-basis: auto; // Valeur par défaut}
```

## Alignment d'un enfant – align-self

La propriété **align-self** permet de définir l'alignement d'un enfant. Elle peut prendre les valeurs suivantes : **auto**, **flex-start**, **flex-end**, **center**, **baseline** ou **stretch**.

Et maintenant , vous allez vous entraîner pour comprendre le fonctionnement des **flexbox**.

Commencez par saisir ce code HTML :

```
<!DOCTYPE html>

<html>
    <head>
        <meta charset="utf-8">
        <title>Media Queries</title>
        <style>
        </style>
    </head>
```

```
<body>
  <div class="conteneur">
    <div class="enfant">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
    </div>
    <div class="enfant">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
    </div>
    <div class="enfant">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
    </div>
    <div class="enfant">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit.
    </div>
    <div class="enfant">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
    </div>
    <div class="enfant">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip.
    </div>
  </div>
</body>
</html>
```

## Exercice

Définissez les classes conteneur et enfant pour mettre en place une mise en page flexbox dans laquelle tous les enfants ont une largeur de 200 pixels, une couleur d'arrière-plan yellow et se partagent équitablement l'espace horizontal restant.

