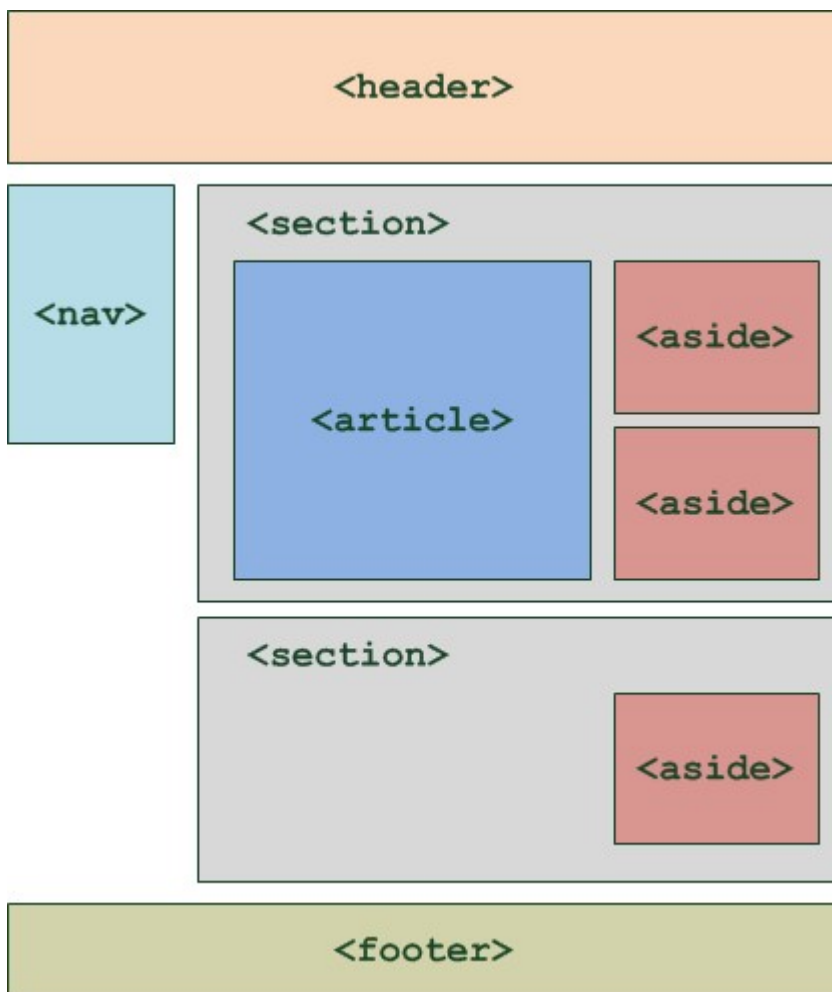


# HTML5

## Introduction

Vérification des fonctionnalités des navigateurs (html5 css3) : <http://caniuse.com/>

### Nouvelle architecture de page html5 :



## Balises de structuration du texte

Balise	Description
<abbr>	Abréviation
<blockquote>	Citation (longue)
<cite>	Citation du titre d'une œuvre ou d'un évènement
<q>	Citation (courte)
<sup>	Exposant
<sub>	Indice
<strong>	Mise en valeur forte
<em>	Mise en valeur normale
<mark>	Mise en valeur visuelle
<h1>	Titre de niveau 1
<h2>	Titre de niveau 2
<h3>	Titre de niveau 3
<h4>	Titre de niveau 4
<h5>	Titre de niveau 5
<h6>	Titre de niveau 6
<img />	Image
<figure>	Figure (image, code, etc.)
<figcaption>	Description de la figure
<audio>	Son

Balise	Description
<video>	Vidéo
<source>	Format source pour les balises <audio> et <video>
<a>	Lien hypertexte
 	Retour à la ligne
<p>	Paragraphe
<hr />	Ligne de séparation horizontale
<address>	Adresse de contact
<del>	Texte supprimé
<ins>	Texte inséré
<dfn>	Définition
<kbd>	Saisie clavier
<pre>	Affichage formaté (pour les codes sources)
<progress>	Barre de progression
<time>	Date ou heure

## Balises de listes

Cette section énumère toutes les balises HTML permettant de créer des listes (listes à puces, listes numérotées, listes de définitions...)

Balise	Description
<code>&lt;ul&gt;</code>	Liste à puces, non numérotée
<code>&lt;ol&gt;</code>	Liste numérotée
<code>&lt;li&gt;</code>	Élément de la liste à puces
<code>&lt;dl&gt;</code>	Liste de définitions
<code>&lt;dt&gt;</code>	Terme à définir
<code>&lt;dd&gt;</code>	Définition du terme

## Balises de tableau

Balise	Description
<code>&lt;table&gt;</code>	Tableau
<code>&lt;caption&gt;</code>	Titre du tableau
<code>&lt;tr&gt;</code>	Ligne de tableau
<code>&lt;th&gt;</code>	Cellule d'en-tête
<code>&lt;td&gt;</code>	Cellule
<code>&lt;thead&gt;</code>	Section de l'en-tête du tableau
<code>&lt;tbody&gt;</code>	Section du corps du tableau
<code>&lt;tfoot&gt;</code>	Section du pied du tableau

## Balises de formulaire

Balise	Description
<code>&lt;form&gt;</code>	Formulaire
<code>&lt;fieldset&gt;</code>	Groupe de champs
<code>&lt;legend&gt;</code>	Titre d'un groupe de champs
<code>&lt;label&gt;</code>	Libellé d'un champ
<code>&lt;input /&gt;</code>	Champ de formulaire (texte, mot de passe, case à cocher, bouton, etc.)
<code>&lt;textarea&gt;</code>	Zone de saisie multiligne
<code>&lt;select&gt;</code>	Liste déroulante
<code>&lt;option&gt;</code>	Élément d'une liste déroulante
<code>&lt;optgroup&gt;</code>	Groupe d'éléments d'une liste déroulante

## Balises sectionnantes

Ces balises permettent de construire le squelette de notre site web.

Balise	Description
<code>&lt;header&gt;</code>	En-tête
<code>&lt;nav&gt;</code>	Liens principaux de navigation
<code>&lt;footer&gt;</code>	Pied de page
<code>&lt;section&gt;</code>	Section de page
<code>&lt;article&gt;</code>	Article (contenu autonome)
<code>&lt;aside&gt;</code>	Informations complémentaires

## Balises génériques

Les balises génériques sont des balises qui n'ont pas de sens sémantique.

En effet, toutes les autres balises HTML ont un *sens* : `<p>` signifie « Paragraphe », `<h2>` signifie « Sous-titre », etc.

Parfois, on a besoin d'utiliser des balises génériques (aussi appelées **balises universelles**) car aucune des autres balises ne convient. On utilise le plus souvent des balises génériques pour construire son design.

Il y a deux balises génériques : l'une est inline, l'autre est block.

Balise	Description
<code>&lt;span&gt;</code>	Balise générique de type inline
<code>&lt;div&gt;</code>	Balise générique de type block

Ces balises ont un intérêt uniquement si vous leur associez un attribut `class`, `id` ou `style` :

- **class**: indique le nom de la classe CSS à utiliser.
- **id**: donne un nom à la balise. Ce nom doit être unique sur toute la page car il permet d'identifier la balise. Vous pouvez vous servir de l'ID pour de nombreuses choses, par exemple pour créer un lien vers une ancre, pour un style CSS de type ID, pour des manipulations en JavaScript, etc.
- **style**: cet attribut vous permet d'indiquer directement le code CSS à appliquer. Vous n'êtes donc pas obligés d'avoir une feuille de style à part, vous pouvez mettre directement les attributs CSS. Notez qu'il est préférable de ne pas utiliser cet attribut et de passer à la place par une feuille de style externe, car cela rend votre site plus facile à mettre à jour par la suite.

Ces trois attributs ne sont pas réservés aux balises génériques : vous pouvez aussi les utiliser sans aucun problème dans la plupart des autres balises.

# CSS3

## Insertion d'un fichier CSS dans un fichier HTML

```
<html>
<head>
<link href="css/site.css" rel="stylesheet" type="text/css" />
</head>
<body>
...
</body>
</html>
```

## Propriétés de mise en forme du texte

Je résume ici la plupart des propriétés de **mise en forme du texte**.

Qu'est-ce que la mise en forme de texte ? C'est tout ce qui touche à la présentation du texte proprement dit : le gras, l'italique, le souligné, la police, l'alignement, etc.

Propriété	Valeurs (exemples)	Description
font-family	<i>police1, police2, police3</i> , serif, sans-serif, monospace	Nom de police
@font-face	<i>Nom et source de la police</i>	Police personnalisée
font-size	1.3em, 16px, 120%...	Taille du texte
font-weight	bold, normal	Gras
font-style	italic, oblique, normal	Italique
text-decoration	underline, overline, line-through, blink, none	Soulignement, ligne au-dessus, barré ou clignotant
font-variant	small-caps, normal	Petites capitales
text-	capitalize, lowercase,	Capitales

Propriété	Valeurs (exemples)	Description
transform	uppercase	
font	-	Super propriété de police. Combine : font-weight, font-style, font-size, font-variant, font-family.
text-align	left, center, right, justify	Alignement horizontal
vertical-align	baseline, middle, sub, super, top, bottom	Alignement vertical (cellules de tableau ou éléments inline-block uniquement)
line-height	18px, 120%, normal...	Hauteur de ligne
text-indent	25px	Alinéa
white-space	pre, nowrap, normal	Césure
word-wrap	break-word, normal	Césure forcée
text-shadow	5px 5px 2px blue (horizontale, verticale, fondu, couleur)	Ombre de texte



## Propriétés de couleur et de fond

Propriété	Valeurs (exemples)	Description
color	<i>nom</i> , rgb(rouge,vert,bleu), rgba(rouge,vert,bleu,transparence), #CF1A20...	Couleur du texte
background-color	<i>Identique à color</i>	Couleur de fond
background-image	url('image.png')	Image de fond
background-attachment	fixed, scroll	Fond fixe
background-repeat	repeat-x, repeat-y, no-repeat, repeat	Répétition du fond
background-position	(x y), top, center, bottom, left, right	Position du fond
background	-	Super propriété du fond. Combine : background- image, background- repeat, background- attachment, background- position
opacity	0.5	Transparence

## Propriétés des boîtes

Propriété	Valeurs (exemples)	Description
width	150px, 80%...	Largeur
height	150px, 80%...	Hauteur
min-width	150px, 80%...	Largeur minimale
max-width	150px, 80%...	Largeur maximale
min-height	150px, 80%...	Hauteur minimale
max-height	150px, 80%...	Hauteur maximale
margin-top	23px	Marge en haut
margin-left	23px	Marge à gauche
margin-right	23px	Marge à droite
margin-bottom	23px	Marge en bas
margin	23px 5px 23px 5px (haut, droite, bas, gauche)	Super-propriété de marge. Combine : margin-top, margin-right, margin-bottom, margin-left.
padding-left	23px	Marge intérieure à gauche
padding-right	23px	Marge intérieure à droite

Propriété	Valeurs (exemples)	Description
padding-bottom	23px	Marge intérieure en bas
padding-top	23px	Marge intérieure en haut
padding	23px 5px 23px 5px (haut, droite, bas, gauche)	Super-propriété de marge intérieure. Combine : padding-top, padding-right, padding-bottom, padding-left.
border-width	3px	Épaisseur de bordure
border-color	nom, rgb(rouge,vert,bleu), rgba(rouge,vert,bleu,transparence), #CF1A20...	Couleur de bordure
border-style	solid, dotted, dashed, double, groove, ridge, inset, outset	Type de bordure
border	3px solid black	Super-propriété de bordure. Combine border-width, border-color, border-style. Existe aussi en version border-top, border-right, border-bottom, border-left.
border-radius	5px	Bordure arrondie
box-shadow	6px 6px 0px black (horizontale, verticale, fondu, couleur)	Ombre de boîte

## Propriétés de positionnement et d'affichage

Propriété	Valeurs (exemples)	Description
display	block, inline, inline-block, table, table-cell, none...	Type d'élément (block, inline, inline-block, none...)
visibility	visible, hidden	Visibilité
clip	rect (0px, 60px, 30px, 0px) <i>rect (haut, droite, bas, gauche)</i>	Affichage d'une partie de l'élément
overflow	auto, scroll, visible, hidden	Comportement en cas de dépassement
float	left, right, none	Flottant
clear	left, right, both, none	Arrêt d'un flottant
position	relative, absolute, static	Positionnement
top	20px	Position par rapport au haut
bottom	20px	Position par rapport au bas
left	20px	Position par rapport à la gauche
right	20px	Position par rapport à la droite
z-index	10	Ordre d'affichage en cas de superposition. La plus grande valeur est affichée par-dessus les autres.

## Propriétés des listes

Propriété	Valeurs (exemples)	Description
<code>list-style-type</code>	disc, circle, square, decimal, lower-roman, upper-roman, lower-alpha, upper-alpha, none	Type de liste
<code>list-style-position</code>	inside, outside	Position en retrait
<code>list-style-image</code>	<code>url('puce.png')</code>	Puce personnalisée
<code>list-style</code>	-	Super-propriété de liste. Combine <code>list-style-type</code> , <code>list-style-position</code> , <code>list-style-image</code> .

## Propriétés des tableaux

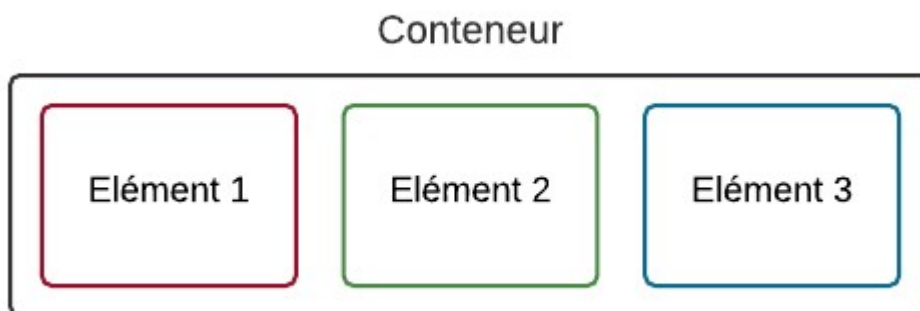
Propriété	Valeurs (exemples)	Description
<code>border-collapse</code>	collapse, separate	Fusion des bordures
<code>empty-cells</code>	hide, show	Affichage des cellules vides
<code>caption-side</code>	bottom, top	Position du titre du tableau

## Autres propriétés

Propriété	Valeurs (exemple)	Description
cursor	crosshair, default, help, move, pointer, progress, text, wait, e-resize, ne-resize, auto...	Curseur de souris

## Positionnement FLEXBOX

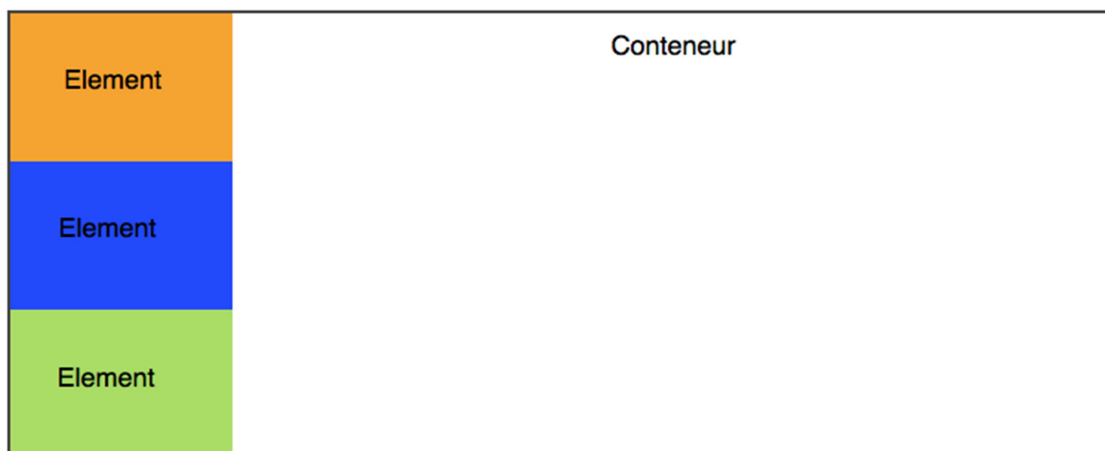
Le principe de la mise en page avec Flexbox est simple : vous définissez un conteneur, et à l'intérieur vous placez plusieurs éléments. Sur une même page web, vous pouvez sans problème avoir plusieurs conteneurs. Ce sera à vous d'en créer autant que nécessaire pour obtenir la mise en page que vous voulez.



Le conteneur est une balise HTML, et les éléments sont d'autres balises HTML à l'intérieur :

```
<div id="conteneur">
  <div class="element">Elément 1</div>
  <div class="element">Elément 2</div>
  <div class="element">Elément 3</div>
</div>
```

Par défaut, les blocs se placent les uns en-dessous des autres



Découvrons maintenant Flexbox. Si je mets une propriété CSS, tout change. Cette propriété, c'est `flex`, et je l'applique au conteneur :

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

alors les blocs se placent par défaut côte à côte.



Enormément de possibilités s'offrent à vous avec flexbox :

<https://openclassrooms.com/courses/apprenez-a-creeer-votre-site-web-avec-html5-et-css3/la-mise-en-page-avec-flexbox>



## Illustration par l'exemple

Voici un fichier HTML :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <link rel="stylesheet" href="style.css" />
    <title>Le Site Web</title>
  </head>

  <body>
    <header>
      <h1>Ceci est un h1</h1>
      <h2>Carnets de voyage</h2>
    </header>

    <nav>
      <ul>
        <li><a href="#">Accueil</a></li>
        <li><a href="#">Blog</a></li>
        <li><a href="#">CV</a></li>
      </ul>
    </nav>

    <section>
      <aside>
        <h1>À propos de l'auteur</h1>
        <p>C'est moi.</p>
      </aside>
      <article>
        <h1>Je suis un grand voyageur</h1>
        <p>texte de l'article</p>
      </article>
    </section>

    <footer>
      <p>Copyright Auteur<br />

      <a href="#">Me contacter !</a></p>
    </footer>

  </body>
</html>
```

Voici le résultat :

# Ceci est un h1

## Carnets de voyage

- [Accueil](#)
- [Blog](#)
- [CV](#)

### À propos de l'auteur

C'est moi.

### Je suis un grand voyageur

texte de l'article

Copyright Auteur  
[Me contacter !](#)

Nous allons essayer de placer le menu à gauche et le reste du texte à droite. Pour cela, nous allons faire flotter le menu à gauche et laisser le reste du texte se placer à sa droite.

Nous voulons que le menu occupe 150 pixels de large. Nous allons aussi rajouter une bordure noire autour du menu et une bordure bleue autour du corps (à la balise `<section>`) pour bien les distinguer :

```
nav
{
  float: left;
  width: 150px;
  border: 1px solid black;
}

section
{
  margin-left: 170px;
  border: 1px solid blue;
}
```

# Ceci est un h1

## Carnets de voyage

- [Accueil](#)
- [Blog](#)
- [CV](#)

### À propos de l'auteur

C'est moi.

### Je suis un grand voyageur

texte de l'article

Copyright Auteur  
[Me contacter !](#)

Une meilleure technique consiste à transformer vos éléments en `inline-block` avec la propriété `display`.

Quelques petits rappels sur les éléments de type `inline-block` :

- Ils se positionnent les uns à côté des autres (exactement ce qu'on veut pour placer notre menu et le corps de notre page !).
- On peut leur donner des dimensions précises (là encore, exactement ce qu'on veut !).

Nous allons transformer en `inline-block` les deux éléments que nous voulons placer côte à côte : le menu de navigation et la section du centre de la page.

```
nav
{
  display: inline-block;
  width: 150px;
  border: 1px solid black;
}

section
{
  display: inline-block;
  border: 1px solid blue;
}
```

# Ceci est un h1

## Carnets de voyage

- [Accueil](#)
- [Blog](#)
- [CV](#)

### À propos de l'auteur

C'est moi.

### Je suis un grand voyageur

texte de l'article

Copyright Auteur  
[Me contacter !](#)

Ce n'est pas tout à fait ce qu'on voulait. Et en fait, c'est normal : les éléments inline-block se positionnent sur une même ligne de base (appelée **baseline**), en bas.

Heureusement, le fait d'avoir transformé les éléments en inline-block nous permet d'utiliser une nouvelle propriété, normalement réservée aux tableaux : `vertical-align`. Cette propriété permet de modifier l'alignement vertical des éléments. Voici quelques-unes des valeurs possibles pour cette propriété :

- `baseline` : aligne de la base de l'élément avec celle de l'élément parent (par défaut) ;
- `top` : aligne en haut ;
- `middle` : centre verticalement ;
- `bottom` : aligne en bas ;
- (valeur en px ou %) : aligne à une certaine distance de la ligne de base (baseline).

Il ne nous reste plus qu'à aligner nos éléments en haut et le tour est joué.

```
nav
{
  display: inline-block;
  width: 150px;
  border: 1px solid black;
  vertical-align: top;
}

section
{
  display: inline-block;
  border: 1px solid blue;
  vertical-align: top;
}
```

# Ceci est un h1

## Carnets de voyage

<ul style="list-style-type: none"><li>• <a href="#">Accueil</a></li><li>• <a href="#">Blog</a></li><li>• <a href="#">CV</a></li></ul>	<b>À propos de l'auteur</b> C'est moi. <b>Je suis un grand voyageur</b> texte de l'article
---	---

Copyright Auteur  
[Me contacter !](#)

## Les positionnements absolu, fixe et relatif

Il existe d'autres techniques un peu particulières permettant de positionner avec précision des éléments sur la page :

- **Le positionnement absolu** : il nous permet de placer un élément n'importe où sur la page (en haut à gauche, en bas à droite, tout au centre, etc.).
- **Le positionnement fixe** : identique au positionnement absolu mais, cette fois, l'élément reste toujours visible, même si on descend plus bas dans la page. C'est un peu le même principe que `background-attachment: fixed;` (si vous vous en souvenez encore).
- **Le positionnement relatif** : permet de décaler l'élément par rapport à sa position normale.

Comme pour les flottants, les positionnements absolu, fixé et relatif fonctionnent aussi sur des balises de type inline.

Toutefois, vous verrez qu'on l'utilise bien plus souvent sur des balises block que sur des balises inline.

Il faut d'abord faire son choix entre les trois modes de positionnement disponibles. Pour cela, on utilise la propriété CSS `position` à laquelle on donne une de ces valeurs :

- `absolute` : positionnement absolu ;
- `fixed` : positionnement fixe ;
- `relative` : positionnement relatif.

Nous allons étudier un à un chacun de ces positionnements.

### Le positionnement absolu

Le positionnement absolu permet de placer un élément (réellement) n'importe où sur la page. Pour effectuer un positionnement absolu, on doit écrire :

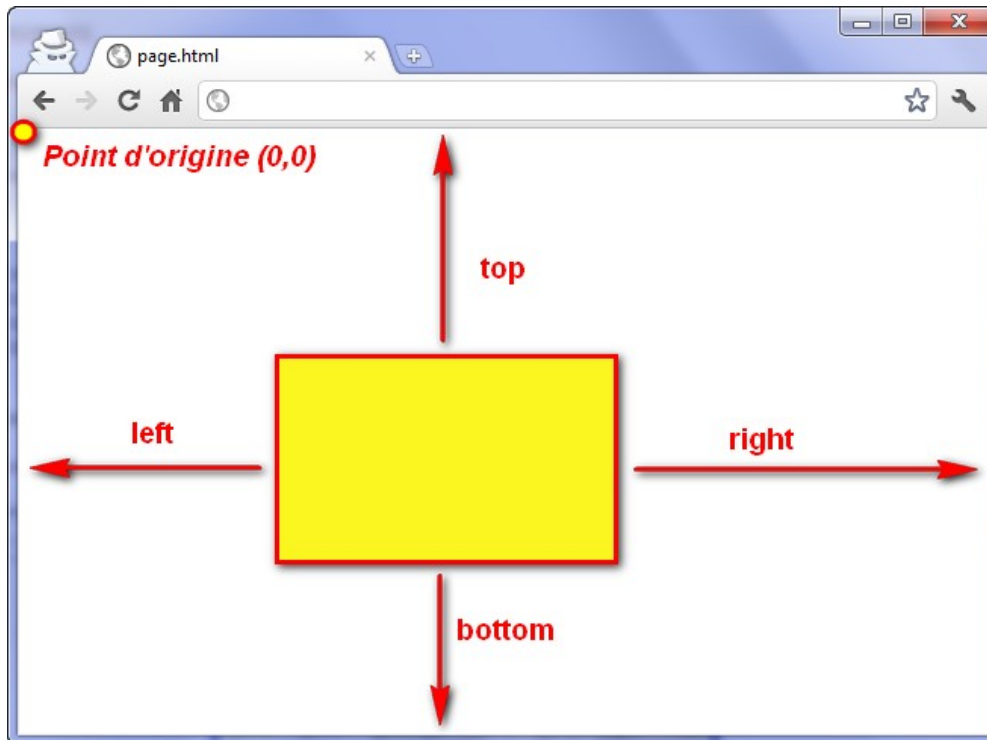
```
element
{
  position: absolute;
}
```

Mais cela ne suffit pas ! On a dit qu'on voulait un positionnement absolu, mais encore faut-il dire où l'on veut que le bloc soit positionné sur la page. Pour ce faire, on va utiliser quatre propriétés CSS :

- `left` : position par rapport à la gauche de la page ;

- `right` : position par rapport à la droite de la page ;
- `top` : position par rapport au haut de la page ;
- `bottom` : position par rapport au bas de la page.

On peut leur donner une valeur en pixels, comme 14px, ou bien une valeur en pourcentage, comme 50%.



Positionnement absolu de l'élément sur la page

Avec cela, vous devriez être capables de positionner correctement votre bloc.

Il faut donc utiliser la propriété `position` et au moins une des quatre propriétés ci-dessus (`top`, `left`, `right` ou `bottom`). Si on écrit par exemple :

```
element
{
  position: absolute;
  right: 0px;
  bottom: 0px;
}
```

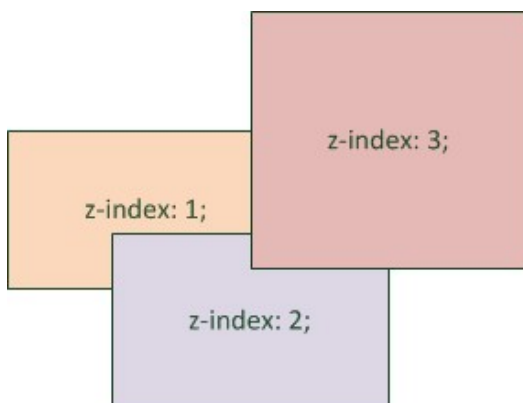
... cela signifie que le bloc doit être positionné tout en bas à droite (0 pixel par rapport à la droite de la page, 0 par rapport au bas de la page).

Si on essaye de placer notre bloc `<nav>` en bas à droite de la page, on obtient le même résultat qu'à la figure suivante.

Les éléments positionnés en absolu sont placés par-dessus le reste des éléments de la page ! Par ailleurs, si vous placez deux éléments en absolu vers le même endroit, ils risquent de se chevaucher. Dans ce cas, utilisez la propriété z-index pour indiquer quel élément doit apparaître au-dessus des autres.

```
element
{
  position: absolute;
  right: 0px;
  bottom: 0px;
  z-index: 1;
}
element2
{
  position: absolute;
  right: 30px;
  bottom: 30px;
  z-index: 2;
}
```

L'élément ayant la valeur de z-index la plus élevée sera placé par dessus les autres, comme le montre la figure suivante.



#### Positionnement des éléments absolus

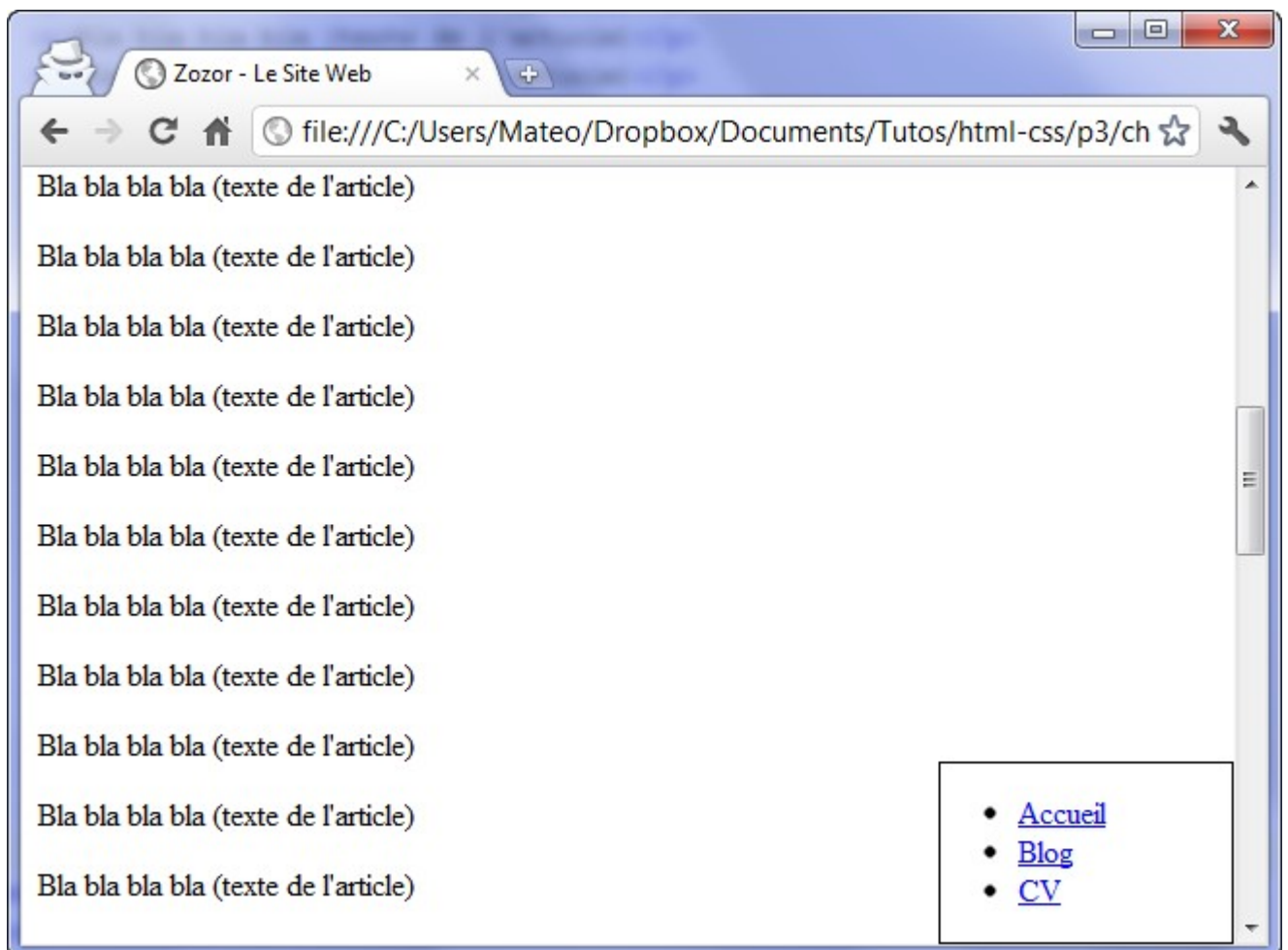
Une petite précision technique qui a son importance : le positionnement absolu ne se fait pas forcément toujours par rapport au coin en haut à gauche de la fenêtre ! Si vous positionnez en absolu un bloc A qui se trouve dans un autre bloc B, lui-même positionné en absolu (ou fixe ou relatif), alors votre bloc A se positionnera par rapport au coin supérieur gauche du bloc B.

## Le positionnement fixe

Le principe est *exactement le même* que pour le positionnement absolu sauf que, cette fois, le bloc reste fixe à sa position, même si on descend plus bas dans la page.

```
element
{
    position: fixed;
    right: 0px;
    bottom: 0px;
}
```

Essayez d'observer le résultat, vous verrez que le menu reste dans le cas présent affiché en bas à droite même si on descend plus bas dans la page (figure suivante).



Le menu reste affiché en bas à droite en toutes circonstances



## Le positionnement relatif

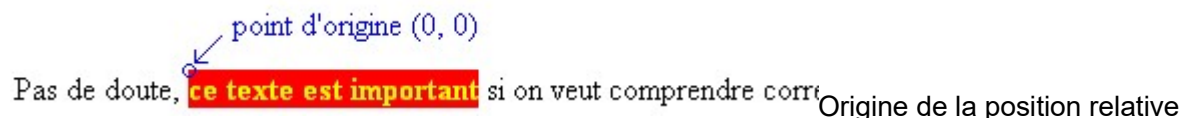
Plus délicat, le positionnement relatif peut vite devenir difficile à utiliser. Ce positionnement permet d'effectuer des « ajustements » : l'élément est décalé par rapport à sa position initiale.

Prenons par exemple un texte important, situé entre deux balises `<strong>`. Pour commencer, je le mets sur fond rouge pour qu'on puisse mieux le repérer :

```
strong
{
  background-color: red; /* Fond rouge */
  color: yellow; /* Texte de couleur jaune */
}
```

Cette fois, le schéma que je vous ai montré tout à l'heure pour les positions absolue et fixe ne marche plus. Pourquoi ? Parce que l'origine a changé : le point de coordonnées (0, 0) ne se trouve plus en haut à gauche de votre fenêtre comme c'était le cas tout à l'heure. Non, cette fois l'origine se trouve en haut à gauche... de la position actuelle de votre élément.

Tordu n'est-ce pas ? C'est le principe de la position relative. Le schéma en figure suivante devrait vous aider à comprendre où se trouve l'origine des points.



Pas de doute, **ce texte est important** si on veut comprendre corr

Origine de la position relative

Donc, si vous faites un `position: relative;` et que vous appliquez une des propriétés `top`, `left`, `right` ou `bottom`, le texte sur fond rouge va se déplacer par rapport à la position où il se trouve.

Prenons un exemple : je veux que mon texte se décale de 55 pixels vers la droite et de 10 pixels vers le bas. Je vais donc demander à ce qu'il soit décalé de 55 pixels par rapport au « bord gauche » et de 10 pixels par rapport au « bord haut » (lignes 6 à 8) :

```
strong
{
  background-color: red;
  color: yellow;

  position: relative;
  left: 55px;
  top: 10px;
}
```

Le texte est alors décalé par rapport à sa position initiale, comme illustré à la figure suivante.



Pas de doute, **ce texte est important** si on veut com

Le texte est décalé !

## En résumé

---

- La technique de mise en page la plus récente et la plus puissante est Flexbox. C'est celle que vous devriez utiliser si vous en avez la possibilité.
- D'autres techniques de mise en page restent utilisées, notamment sur des sites plus anciens : le positionnement flottant et le positionnement `inline-block`. Il est conseillé de les connaître.
- Le positionnement flottant (avec la propriété `float`) est l'un des plus utilisés à l'heure actuelle. Il permet par exemple de placer un menu à gauche ou à droite de la page. Néanmoins, cette propriété n'a pas été initialement conçue pour cela et il est préférable, si possible, d'éviter cette technique.
- Le positionnement `inline-block` consiste à affecter un type `inline-block` à nos éléments grâce à la propriété `display`. Ils se comporteront comme des inlines (placement de gauche à droite) mais pourront être redimensionnés comme des blocs (avec `width` et `height`). Cette technique est à préférer au positionnement flottant.
- Le positionnement absolu permet de placer un élément où l'on souhaite sur la page, au pixel près.
- Le positionnement fixe est identique au positionnement absolu mais l'élément restera toujours visible même si on descend plus bas dans la page.
- Le positionnement relatif permet de décaler un bloc par rapport à sa position normale.
- Un élément A positionné en absolu à l'intérieur d'un autre élément B (lui-même positionné en absolu, fixe ou relatif) se positionnera par rapport à l'élément B, et non par rapport au coin en haut à gauche de la page.

Références de base de ce document :

<https://openclassrooms.com/courses/apprenez-a-creer-votre-site-web-avec-html5-et-css3>

<http://www.alsacreations.com/tuto/lire/608-initiation-positionnement-css.html>

<http://www.w3schools.com/html/default.asp>

<http://www.w3schools.com/css/default.asp>

<http://www.w3schools.com/bootstrap/default.asp>

<http://getbootstrap.com/css/>