



**formation.tech**

# CSS - Cascading Style Sheet



**formation.tech**

# Introduction

# Introduction - Historique



- Première apparition en 1992 dans le navigateur Viola
- Norme CSS 1 par le W3C en 1996
- CSS 2 en 1998
- CSS 2.1 en 2001
- Après CSS 2.1 la norme est découpée en modules qui sont aujourd'hui en version 1 à 5

# Introduction - Comment inclure du CSS ?



- Le CSS peut s'inclure de 3 façons :
  - via un fichier externe et la balise link

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

- via une balise style

```
<style>
  body {
    background-color: yellow;
    font-weight: bold;
  }
</style>
```

- via l'attribut style (déconseillé)

```
<div id="introduction" style="background-color: greenyellow">
```

# Introduction - Comment inclure du CSS ?



- Dans un fichier externe ou dans une balise style on définit des règles CSS c'est à dire
  - un ensemble de propriétés
  - un sélecteur qui spécifie à quelles balises appliquer ces propriétés

```
sélecteur {  
    propriété: valeur;  
    propriété: valeur;  
}
```

# Introduction - Compatibilité



- Les modules CSS sont des normes qui évoluent en permanence
  - Les navigateurs les implémentent au fur et pas forcément avec les mêmes priorités
  - Un sélecteur, une propriété ou une valeur peut fonctionner dans un navigateur mais pas un autre
  - Certaines propriétés ou valeurs sont préfixées par -webkit-, -moz-, -ms-, -o- c'est le signe que l'implémentation est expérimentale ou qu'il s'agit d'une extension à la norme
  - Pour connaître la compatibilité on peut utiliser
    - Can I Use -> <https://caniuse.com/>
    - MDN -> <https://developer.mozilla.org/fr/docs/Web/CSS/flex>
    - Chrome Status -> <https://www.chromestatus.com/features/schedule>

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Opera Mobile
					10-11.5				
6-9		1 2-21 3 22-27	1 4-20 21-28	1 3.1-6 6.1-8	12.1	1 3.2-6.1 7-8.4		1 2.1-4.3	12
2 4 10	12-85	28-81	29-85	9-13.1	17-71	9-13.7		4.4-4.4.4	12.1
4 11	86	82	86	14	72	14	all	81	59
		83-84	87-89	TP					





# Introduction - Cascade et héritage

- Lorsque 2 règles entre en conflit :
  - à spécificité équivalente (poids du sélecteur), la 2e l'emporte
  - sinon la règle dont le sélecteur est le plus fort l'emporte
- Certaines règles sont hérités, par exemple color, la définir au niveau d'un élément l'applique à tout ces descendants



formation.tech

# Sélecteurs CSS

# Sélecteurs CSS



- Les sélecteurs CSS est un langage de recherche de noeud dans un arbre
- Ils permettent de sélectionner les éléments de la page (balises) auxquels s'appliqueront des propriétés CSS
- Norme actuelle : Selectors Level 3  
<http://www.w3.org/TR/selectors/>  
<http://caniuse.com/#feat=css-sel3>
- Norme en cours de redaction : Selectors Level 4  
<http://www.w3.org/TR/selectors4/>  
<http://css4-selectors.com/browser-selector-test/>

# Sélecteurs CSS



- ▶ Exemple d'utilisation des Sélecteurs CSS

- En CSS

```
#compteur {  
    width: 100px;  
    height: 100px;  
    background: red;  
}
```

- En JavaScript

```
var compteur = document.querySelector('#compteur');  
compteur.addEventListener('click', function() {  
    compteur.innerHTML++;  
});
```

- Avec jQuery

```
var $compteur = $('#compteur');  
$compteur.click(function() {  
    $compteur.toggleClass('inverse');  
});
```

# Sélecteurs CSS



## ▶ Sélecteurs Basiques

*	tous les éléments	*
E	un élément de type E	div p h1
E.warning	un élément E dont l'une des classes est warning	div.important *.center .center button.btn.btn-primary
E#myid	un élément E dont l'id est myId	div#boite *#boite #boite
E, F	les éléments E et les éléments F	div, p h1, h2, h3, h4, h5, h6 button.btn, a.btn div.center, #box, p

# Sélecteurs CSS



## ▶ Sélecteurs Hiérarchiques

E F	un élément F descendant de E	#boite a a span.icon
E > F	un élément F fils de E	body > h1 ul > li
E + F	un élément F immédiatement précédent par un élément E	h2 + p li + li
E ~ F	un élément F précédent par un élément E	h2 ~ form

# Sélecteurs CSS



## ▶ Sélecteurs par pseudo-classes hiérarchiques

E:root	un élément E, racine du document	html:root :root
E:nth-child(n)	un élément E, nième enfant de son parent	li:nth-child(3) li:nth-child(odd) li:nth-child(even) li:nth-child(2n+1)
E:nth-last-child(n)	un élément E, nième enfant en partant de la fin	li:nth-last-child(3)
E:nth-of-type(n)	un élément E, nième enfant du même type	h2:nth-of-type(3) h2:nth-of-type(odd) h2:nth-of-type(3n)
E:nth-last-of-type(n)	un élément E, nième enfant du même type en partant de la fin	h2:nth-last-of-type(3)
E:first-child	un élément E, premier enfant	li:first-child

# Sélecteurs CSS



## ▶ Sélecteurs par pseudo-classes hiérarchiques (suite)

E:last-child	un élément E, dernier enfant	li:last-child
E:first-of-type	un élément E, premier enfant du même type	h2:first-of-type
E:last-of-type	un élément E, dernier enfant du même type	h2:last-of-type
E:only-child	un élément E, fils unique	div:only-child
E:only-of-type	un élément E, fils unique du même type	form:only-of-type
E:empty	un élément E, qui n'a pas d'enfant, excepté des noeuds de texte	div:empty

# Sélecteurs CSS



## ▶ Sélecteurs par attributs

E[foo]	un élément E avec un attribut foo	*[id] [id]
E[foo="bar"]	un élément E dont l'attribut foo vaut bar	img[src="monimage.png"]
E[foo~="bar"]	un élément E dont l'attribut foo contient le mot bar	a[rel~="nofollow"] button[class~="btn"]
E[foo^="bar"]	un élément E dont l'attribut foo commence par bar	a[href^="https://"]
E[foo\$="bar"]	un élément E dont l'attribut foo finit par bar	img[src\$=".png"]
E[foo*="bar"]	un élément E dont l'attribut foo contient bar	div[id*="box"]
E[foo = "en"]	un élément E dont l'attribut foo contient en ou en- suivi de quelque chose	a[hreflang = "en"]

# Sélecteurs CSS



## ▶ Sélecteurs par pseudo-classes de contexte

E:link	un lien non visité	a:link
E:visited	un lien visité	a:visited
E:active	un élément actif (clic en cours)	a:active div:active
E:hover	un élément survolé	a:hover div:hover
E:focus	un élément qui reçoit le focus	input:focus

# Sélecteurs CSS



## ▶ Sélecteurs par pseudo-classes de contexte

E:target	un élément cible d'un lien	div:target
E:lang(fr)	un élément dont la langue est fr	div:lang(fr)
E:enabled	un élément activé	input:enabled
E:disabled	un élément désactivé	input:disabled
E:checked	un élément coché	input[type="checkbox"]:checked
E:not(s)	une élément E qui ne correspond pas au sélecteur simple s	div:not(.important)



# Sélecteurs CSS

## ▶ Sélecteurs par pseudo-élément

E::first-line	la premier ligne de l'élément E	p::first-line
E::first-letter	la premier lettre de l'élément E	p::first-letter
E::before	contenu généré avant l'élément E	div::before
E::after	contenu généré après l'élément E	div::after



# Sélecteurs CSS

- Priorités / Spécificité

<http://www.standardista.com/css3/css-specificity/>

<https://cssspecificity.com/>

## CSS SPECIFISHITY

WITH PLANKTON, FISH AND SHARKS

* universal selector 0 - 0 - 0	div 1 element 0 - 0 - 1	li > ul 2 elements 0 - 0 - 2	body div ... ul li p a 12 elements 0 - 0 - 12
.myClass 1 class 0 - 1 - 0	*.myClass 1 universal selector 1 class 0 - 1 - 0	[type=checkbox] 1 attribute selector 0 - 1 - 0	:only-of-type 1 pseudo-class 0 - 1 - 0
li.myClass 1 element 1 class 0 - 1 - 1	li[attr] 1 element 1 attribute 0 - 1 - 1	li:nth-of-type(3n)~li 2 elements 1 pseudo-class 0 - 1 - 2	form input[type=email] 2 elements 1 attribute 0 - 1 - 2
li.class:nth-of-type(3n) 1 element 1 class 1 pseudo-class 0 - 2 - 1	input[type]:not(.class) 1 element 1 class 1 attribute 0 - 2 - 1	cl:nth-child(3n)~chk[type]... 10 class/attribute, pseudo-classes 0 - 10 - 0	#myDiv ID Selector 1 - 0 - 0
#myDiv li.class a[href] 2 types 2 class/attribute 1 ID Selector 1 - 2 - 2	#divitis #myDiv a 2 ID Selectors 1 type selector 2 - 0 - 1	style="" inline style 1 - 0 - 0 - 0	!important !important 1 - 0 - 0 - 0 - 0

X-0-0: The number of ID selectors

0-Y-0: The number of class selectors, attributes selectors, and pseudo-classes

0-0-Z: The number of type selectors and pseudo-elements

\*, +, >, ~ : Universal selector and combinator do not increase specificity

:not(x): Negation selector has no value. Argument increases specificity





# Sélecteurs CSS

- ▶ Apprendre en s'amusant  
<http://flukeout.github.io/>

The screenshot shows a web browser window for 'CSS Diner - Where we feast on...' at <http://flukeout.github.io/>. The title bar includes a 'Roman' button. The page has a dark background with a central illustration of two white plates on a yellow napkin. A 'Help, I'm stuck!' button is visible. The main content area is titled 'Level 1 of 26' with a checkmark. It features a 'Type Selector' section with the text 'Select elements by their type'. It shows examples for 'A' (Selects all elements of type `A`) and 'div' (Selects all `<div>` elements). The 'HTML Viewer' shows the following code:

```
1 <div class="table">
2   <plate></plate>
3   <plate></plate>
4 </div>
```

The 'CSS Editor' shows the following code:

```
1 Type in a CSS selector [enter]
2 {
3 /* Styles would go here. */
4 }

5
6 /*
7 Type a number to skip to a level.
8 Ex → "5" for level 5
9 */
10
```



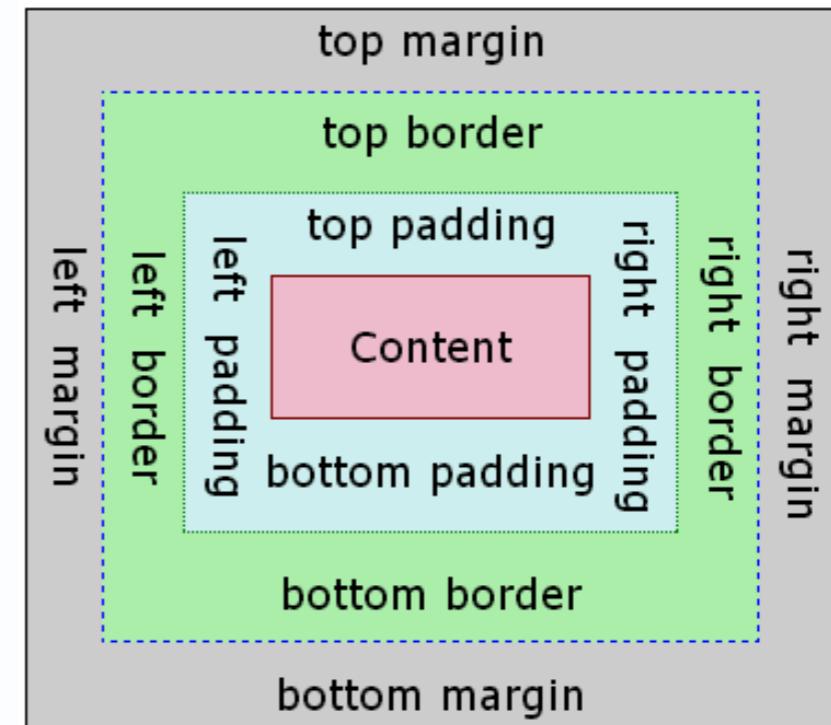
**formation.tech**

# Modèle de boîte CSS

# Modèle de boîte CSS - Introduction



- ▶ Modèle de boîte CSS
  - Chaque élément d'un document est matérialisé par une boîte qui peut être ajustée grâce à des propriétés CSS spécifiques.
- ▶ Cette boîte possède différentes dimensions :
  - Content : Contenu
  - Padding : Marge interne
  - Border : Bordure
  - Margin : Marge externe



# Modèle de boîte CSS - Content



- Le contenu d'une boîte, par exemple du texte peut se dimensionner grâce aux propriétés width et height
- Les valeurs par défaut dépendent du type :
  - pour les balises inline :  
`width: auto; height: auto;`
  - pour les balises block :  
`width: 100%; height: auto;`
  - (100% est relatif au parent, auto est relatif au contenu)
- Pour les tailles relatives on peut définir des largeurs et hauteurs minimales et maximales avec min-width, min-height, max-width et max-height
- On ne peut pas redimensionner un élément inline (mais on peut le transformer en block)

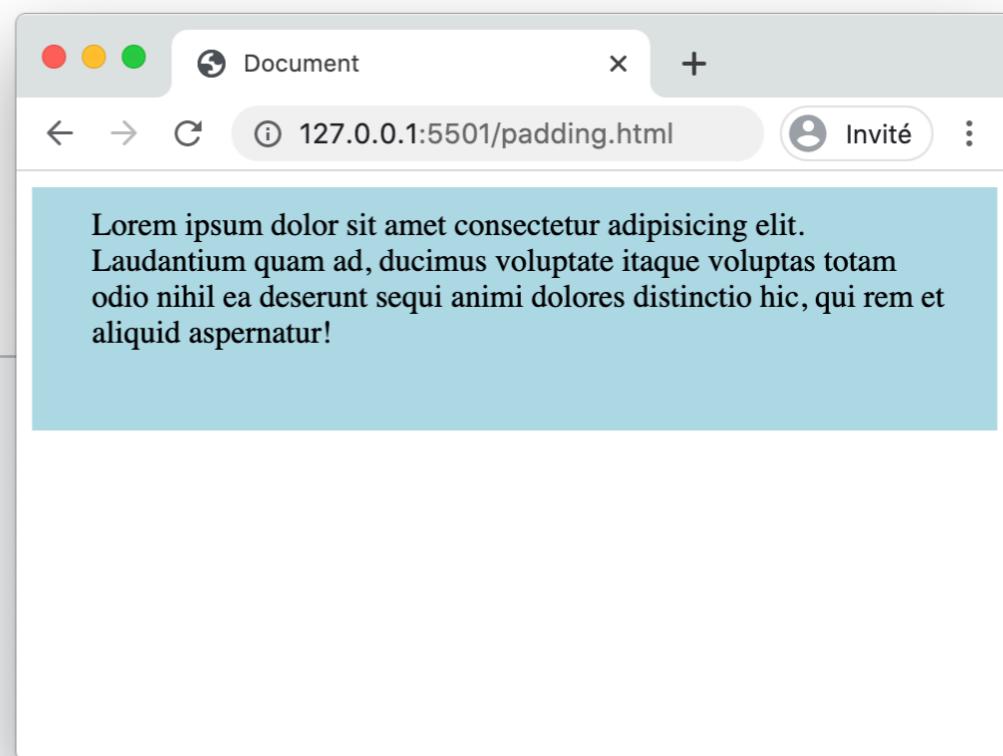


# Modèle de boîte CSS - Padding

- › Le padding est une marge interne à l'élément
- › Le padding inclus le background
- › Sa valeur se définit à l'aide des propriétés :  
padding-top, padding-right, padding-bottom, padding-left

```
<div>
    Lorem ipsum dolor sit amet consectetur adipisicing elit. Laudantium quam
    ad, ducimus voluptate itaque voluptas totam odio nihil ea deserunt sequi
    animi dolores distinctio hic, qui rem et aliquid aspernatur!
</div>
```

```
div {
    background-color: lightblue;
    padding-top: 10px;
    padding-right: 20px;
    padding-bottom: 30px;
    padding-left: 40px;
}
```



# Modèle de boîte CSS - Padding



- › La propriété padding est un raccourci de padding-top, padding-right, padding-bottom, padding-left
- › Si on lui passe 4 valeurs, elles se lisent dans le sens des aiguilles d'une montre (haut, droite, bas, gauche) :

```
padding: 10px 20px 30px 40px;  
/* est équivalent à : */  
padding-top: 10px;  
padding-right: 20px;  
padding-bottom: 30px;  
padding-left: 40px;
```

- › Si on lui passe 3 valeurs : haut droite+gauche bas
- › Si on lui passe 2 valeurs : haut+bas droite+gauche
- › Si on lui passe 1 valeur : haut+droite+bas+gauche

# Modèle de boîte CSS - Border

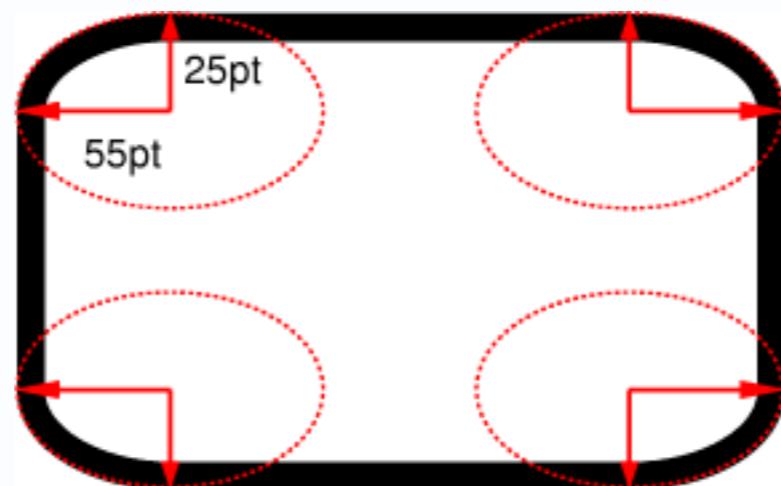


- ▶ Une bordure se définit à l'aide des propriétés :
  - border-width (border-top-width, border-right-width, border-bottom-width, border-left-width) : l'épaisseur du trait
  - border-color (border-top-color, border-right-color, border-bottom-color, border-left-color) : la couleur du trait
  - border-style (border-top-style, border-right-style, border-bottom-style, border-left-style) : le type de trait parmi les mots clés
    - none, hidden (masqués)
    - dotted, dashed, solid, double (pointillés, tirets, trait plein, trait double)
    - groove, ridge, inset, outset (effets 3d)
- ▶ Chaque propriétés raccourci peut prendre 1, 2, 3 ou 4 valeurs avec les mêmes règles que padding
- ▶ La propriété border est le raccourci de border-width border-style border-color  
ex : border: 1px solid black; (voir aussi border-top|right|bottom|left)

# Modèle de boîte CSS - Bords arrondis



- border-top-left-radius: x [ y ]
- border-top-right-radius
- border-bottom-left-radius
- border-bottom-right-radius
- border-radius: top-left top-right bottom-right bottom-left



# Modèle de boîte CSS - Margin



- margin est une marge externe
- De nombreuses balises HTML ont une ou plusieurs marges par défaut (h1, h2, h3, h4, h5, h6, p, ul, ol)
- On peut utiliser les propriétés margin-top, margin-left, margin-bottom, margin-right
- La propriété margin est une propriété raccourci qui peut prendre 1, 2, 3 ou 4 valeurs avec les mêmes règles que padding
- Attention à la fusion des marges !

# Modèle de boîte CSS - Fusion des marges

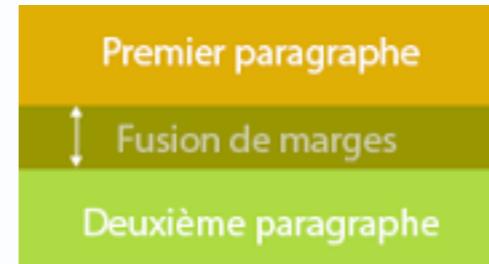


## ▶ Fusion des marges

Les marges haute et basse des blocs sont parfois fusionnées en une seule marge dont la taille est la plus grande des deux marges fusionnées. C'est ce qu'on appelle la fusion des marges.

## ▶ 3 cas possibles

- Des éléments voisins adjacents  
La marge inférieure d'un élément fusionne avec la marge supérieure de l'élément suivant (ex : les paragraphes).
- Un élément parent et son premier/dernier élément fils  
La marge supérieure du parent fusionne avec la marge supérieure de ses descendants (si le parent n'a pas de padding et de border), idem pour les marges inférieures
- Des blocs vides  
Les marges du haut et du bas fusionnent avec les précédents suivants (si pas de padding et de border)





# Modèle de boîte CSS - box-sizing

- ▶ La propriété box-sizing permet de configurer comme se calcule la largeur et la hauteur d'un élément
  - content-box  
Par défaut, la hauteur totale d'un élément sera de :  
$$\text{margin-top} + \text{border-top} + \text{padding-top} + \text{height} + \text{padding-bottom} + \text{border-bottom} + \text{margin-bottom}$$
  - border-box  
En box-sizing border-box, la hauteur de l'élément inclus le padding et la bordure, ainsi la hauteur total sera de :  
$$\text{margin-top} + \text{height} + \text{margin-bottom}$$
- ▶ (idem pour le calcul de la largeur dans les 2 cas)
- ▶ Les calculs étant simplifiés en box-sizing border-box, certains bibliothèques CSS l'activent pour tous les éléments :

```
* {  
  -webkit-box-sizing: border-box;  
  -moz-box-sizing: border-box;  
  box-sizing: border-box;  
}
```

# Modèle de boîte CSS - Débordement



## ▶ Débordement

Lorsque la hauteur et/ou la largeur d'un élément ne permet pas d'afficher son contenu, on peut décider que faire du "débordement" avec overflow, overflow-x ou overflow-y (overflow est la propriété raccourcie)

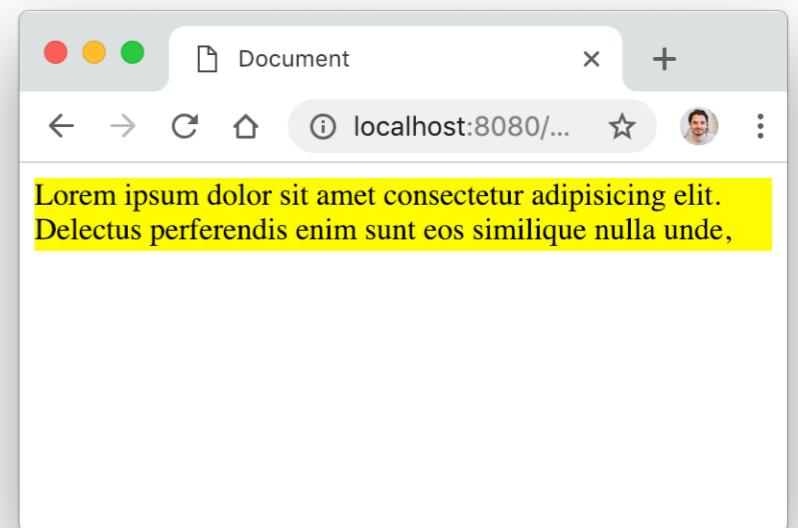
## ▶ Valeurs

- visible: le contenu débordant s'affiche (défaut)
- hidden : le contenu débordant ne s'affiche pas.
- scroll : le conteneur ajoute une barre de scroll
- auto : le navigateur décide s'il faut une barre de scroll

```
<div>
```

  Lorem ipsum dolor sit amet consectetur adipisicing elit.  
  Delectus preferendis enim sunt eos similique nulla unde,  
  dignissimos numquam magni incidunt quae quis veniam dolorem  
  in voluptatem? Quibusdam fugit a maxime!

```
</div>
```



```
div {  
  height: 2.4em;  
  background-color: yellow;  
  overflow: hidden;  
}
```

# Modèle de boîte CSS - Ombres



- box-shadow: x y blur-x blur-y couleur [inset];
- On peut en mettre plusieurs séparées par des virgules.
- Elle s'applique de la dernière à la première.
- inset = ombre intérieure
- S'adapte aux arrondis.



**formation.tech**

# Unités

# Unités - Introduction



- ▶ Documentation
  - <https://www.w3.org/Style/Examples/007/units.fr.html>
  - [https://developer.mozilla.org/en-US/docs/Learn/CSS/Building\\_blocks/Values\\_and\\_units](https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Values_and_units)
- ▶ Attention à ne pas mettre d'espace entre la valeur et son unité :

```
div {  
  width: 100%;  
  height: 200px;  
  font-size: 2em;  
}
```

# Unités - Les unités de tailles



- px : une largeur absolue exprimée en pixels  
Exemple : 10px est une largeur  
(site dont le conteneur est en pixel : fixed design)
- % : une largeur relative exprimée en pourcentage du parent  
(site dont le conteneur est en % : fluid design)
- em : multiple de la taille de la police du parent (font-size)  
exemple : 2em est équivalent à 2 fois la police du parent (32px si la police parent vaut 16px), à utiliser pour font-size, parfois margin
- rem : multiple de la taille de la police de la balise racine
- vw : 1% de la largeur du viewport
- vh : 1% de la hauteur du viewport
- vmin : 1% du plus petit entre la largeur et la hauteur du viewport
- vmax : 1% du plus grand entre la largeur et la hauteur du viewport (pas IE11)



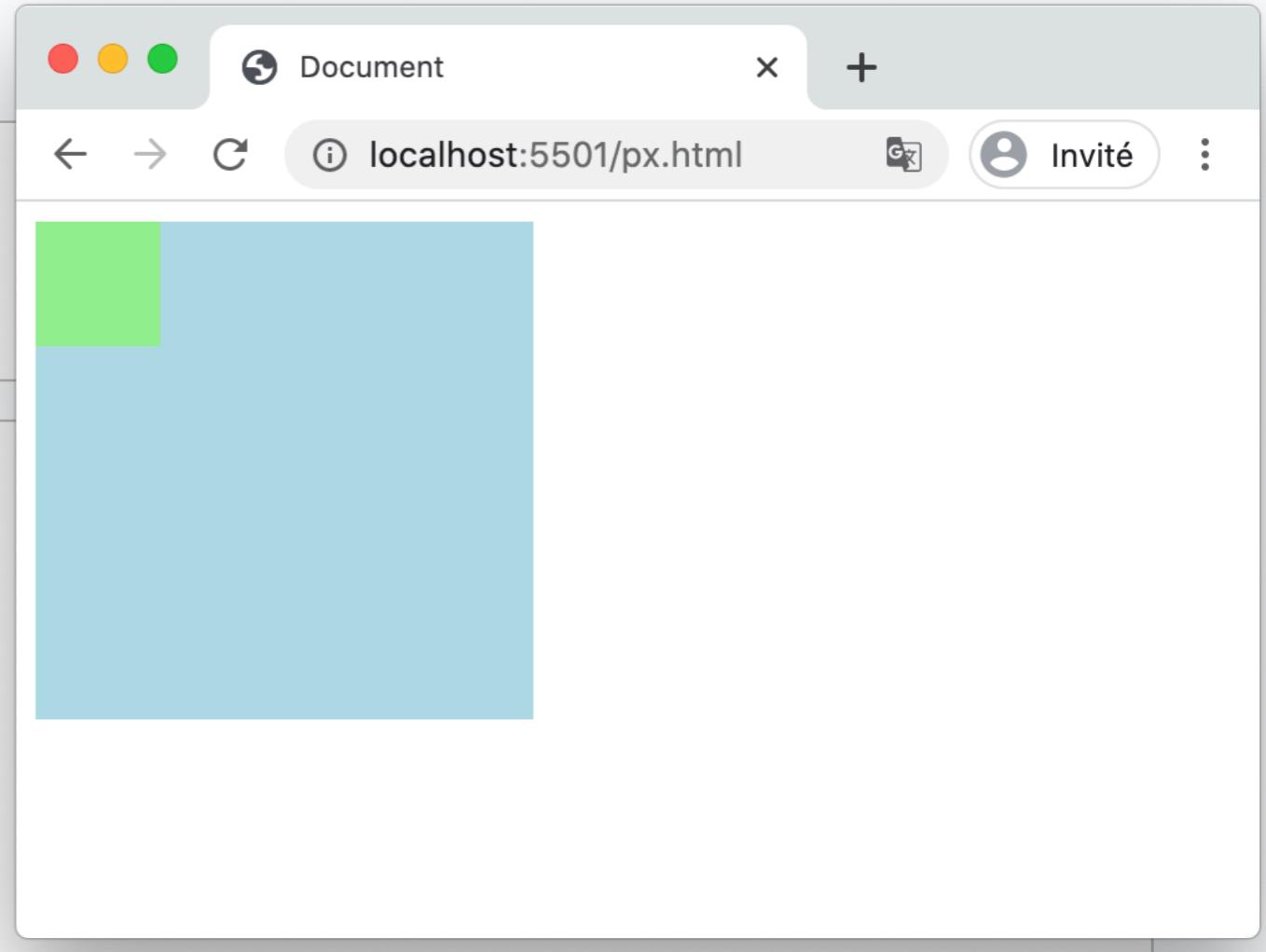
# Unités - Les unités de tailles

## ► px

```
<body>
  <div class="outer">
    <div class="inner"></div>
  </div>
</body>
```

```
.outer {
  background-color: lightblue;
  width: 200px;
  height: 200px;
}

.inner {
  background-color: lightgreen;
  width: 50px;
  height: 50px;
}
```





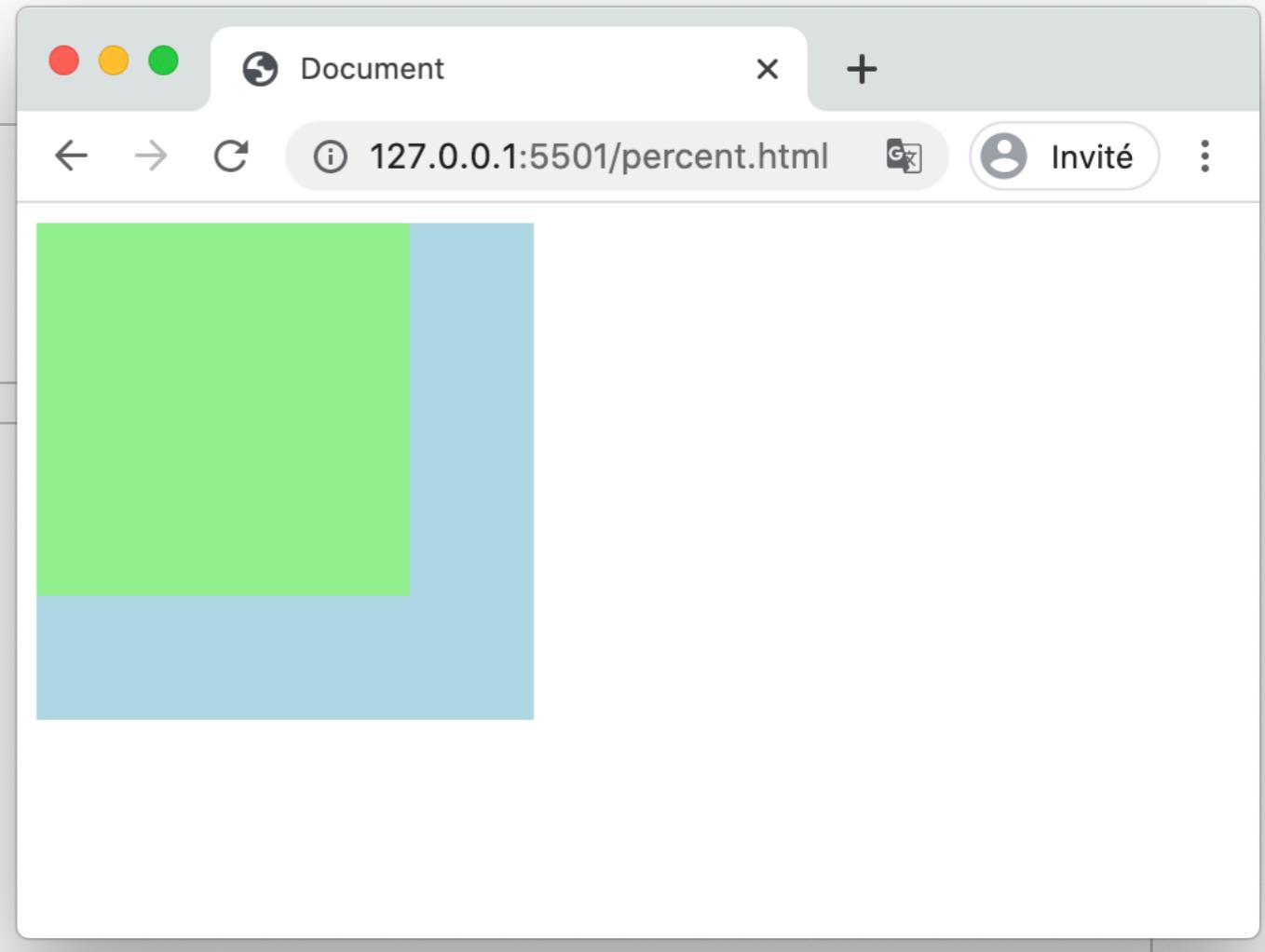
# Unités - Les unités de tailles

▶ %

```
<body>
  <div class="outer">
<div class="inner"></div>
  </div>
</body>
```

```
.outer {
  background-color: lightblue;
  width: 200px;
  height: 200px;
}

.inner {
  background-color: lightgreen;
  width: 75%; /* 75% of parent width */
  height: 75%; /* 75% of parent height */
}
```





# Unités - Les unités de tailles

## ► em

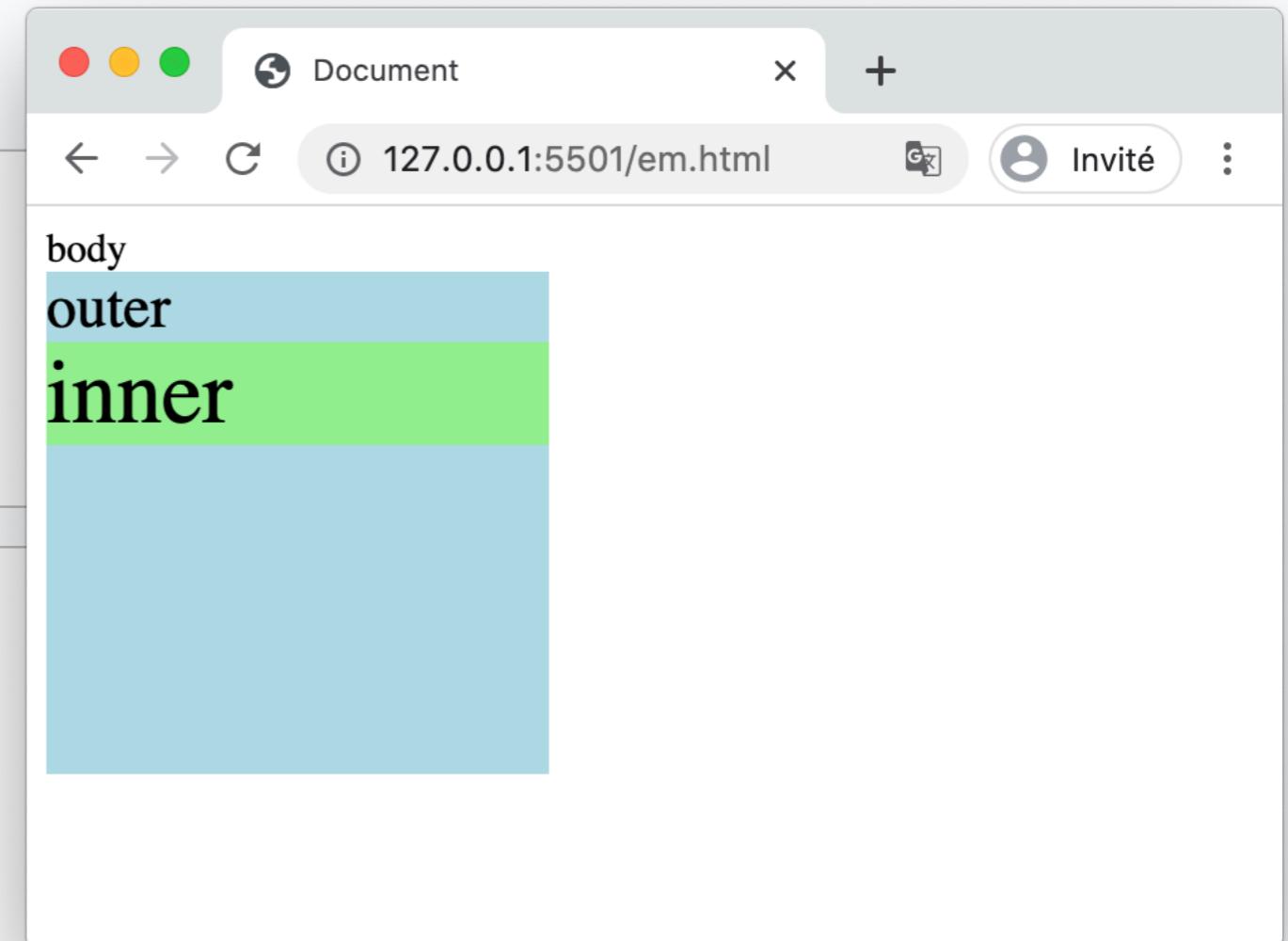
```
<body>
  body
    <div class="outer">
      outer
        <div class="inner">inner</div>
    </div>
</body>
```

```
:root {
  font-size: 12px;
}

body {
  font-size: 16px;
}

.outer {
  background-color: lightblue;
  width: 200px;
  height: 200px;
  font-size: 1.5em; /* 1.5×16px (parent's font-size) = 24px */
}

.inner {
  background-color: lightgreen;
  font-size: 1.5em; /* 1.5×24px (parent's font-size) = 36px */
}
```





# Unités - Les unités de tailles

## ► rem

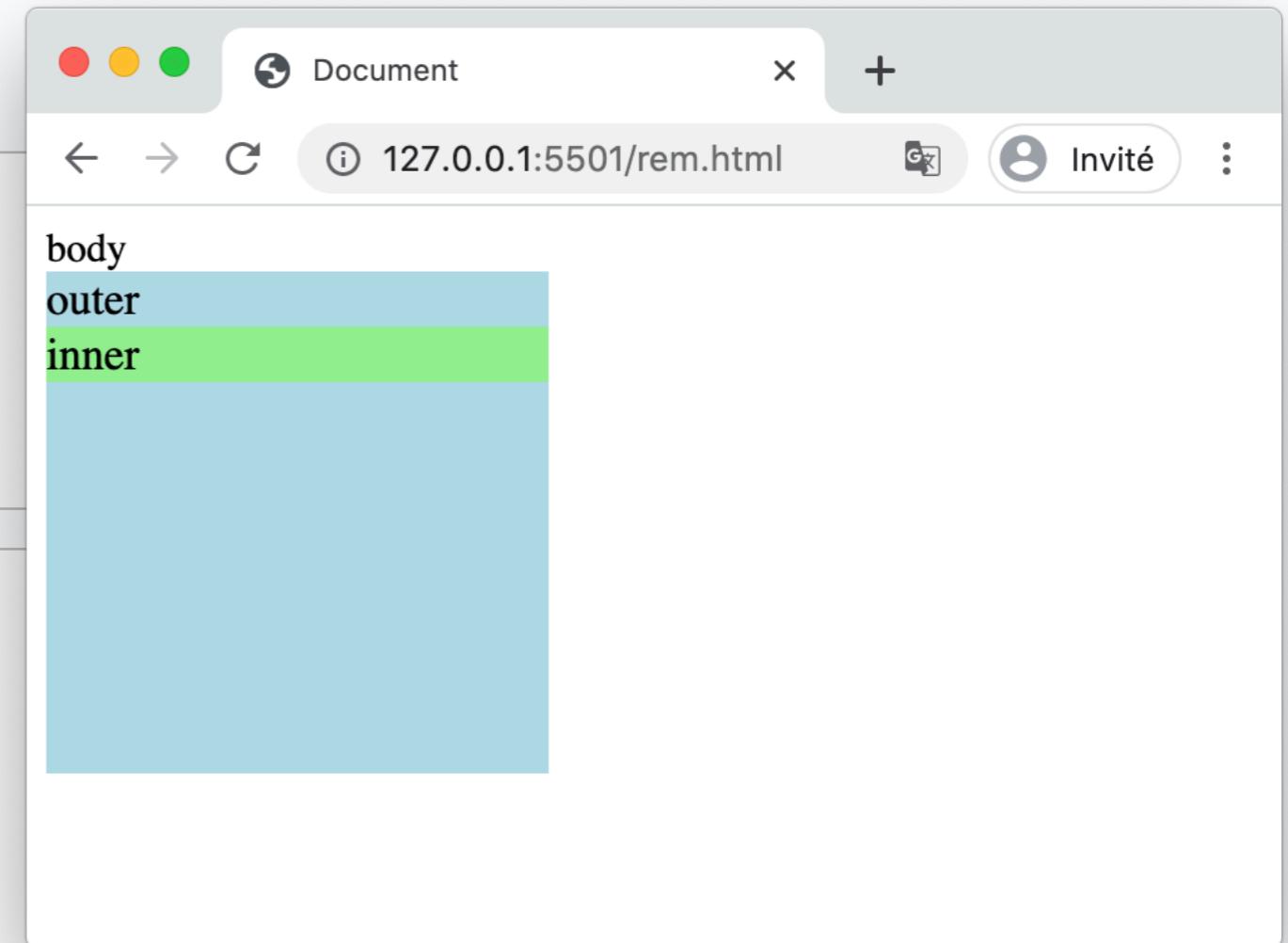
```
<body>
  body
    <div class="outer">
      outer
        <div class="inner">inner</div>
    </div>
</body>
```

```
:root {
  font-size: 12px;
}

body {
  font-size: 16px;
}

.outer {
  background-color: lightblue;
  width: 200px;
  height: 200px;
  font-size: 1.5rem; /* 1.5×12px (root's font-size) = 18px */
}

.inner {
  background-color: lightgreen;
  font-size: 1.5rem; /* 1.5×12px (root's font-size) = 18px */
}
```





# Unités - Les unités de tailles

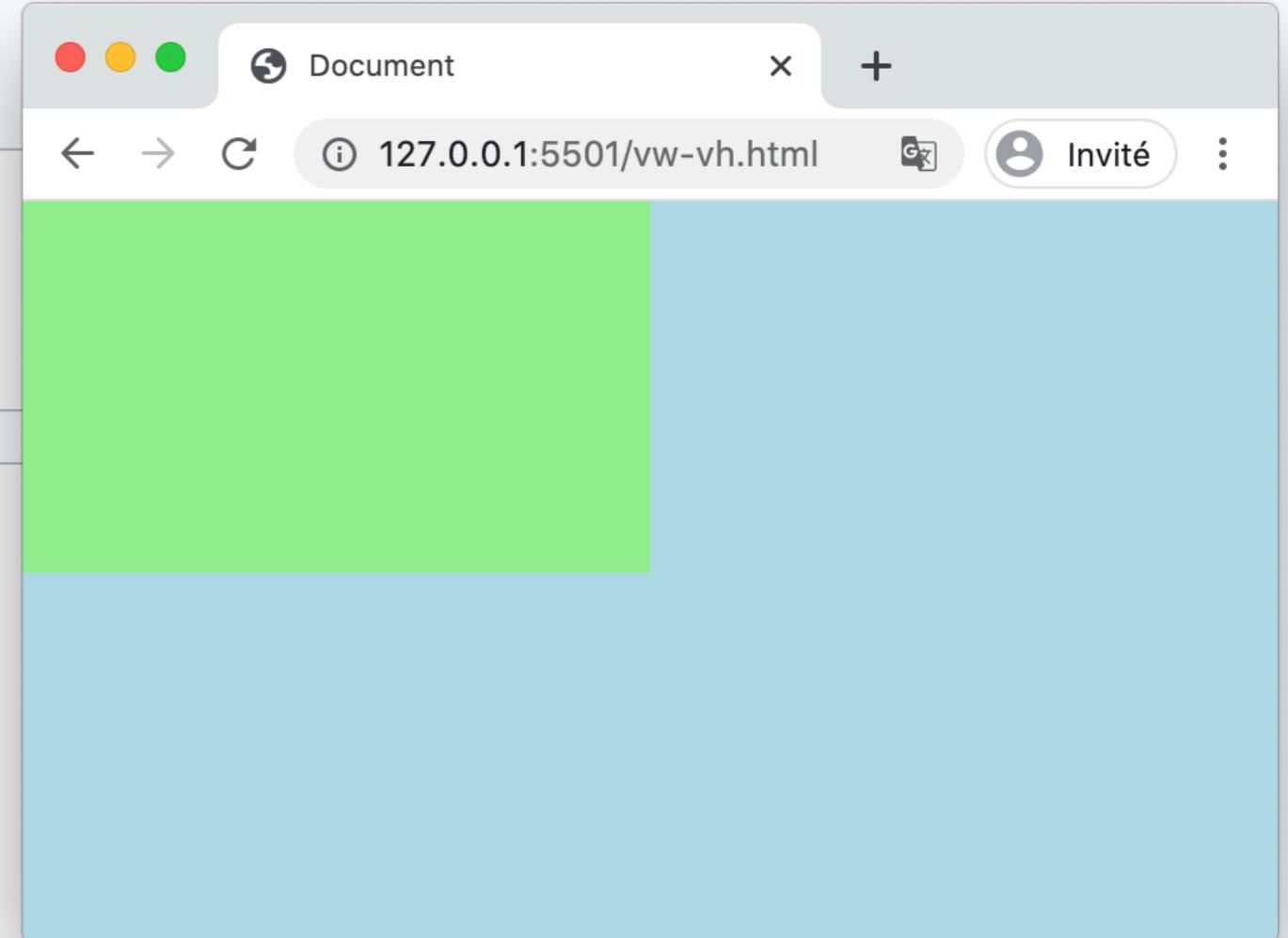
## ▶ vw / vh

```
<body>
  <div class="outer">
<div class="inner"></div>
  </div>
</body>
```

```
body {
  margin: 0;
}

.outer {
  background-color: lightblue;
  width: 100vw;
  height: 100vh;
}

.inner {
  background-color: lightgreen;
  width: 50vw;
  height: 50vh;
}
```



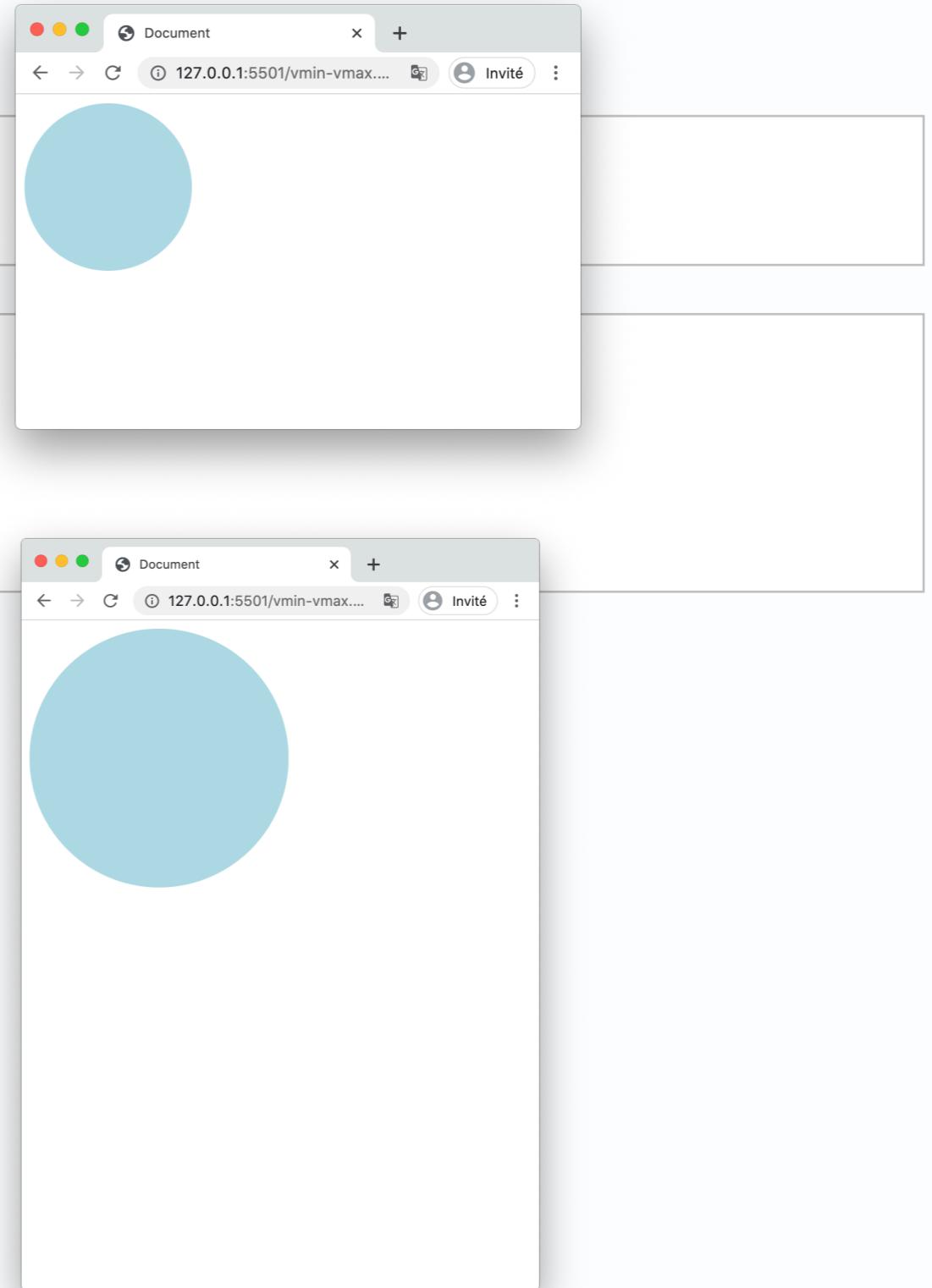


# Unités - Les unités de tailles

## ▶ vmin / vmax

```
<body>
  <div class="circle"></div>
</body>
```

```
.circle {
  background-color: lightblue;
  width: 50vmin;
  height: 50vmin;
  border-radius: 25vmin;
}
```



# Unités - Les couleurs



- ▶ Les couleurs peuvent se définir :
  - avec un mot clé (blue, transparent...)
  - en système RGB (Rouge, Vert, Bleu)
  - en système HSL (Teinte, Saturation, Luminosité)

# Unités - Les couleurs (mot clés)



- aliceblue, antiquewhite, aqua, aquamarine, azure, beige, bisque, black, blanchedalmond, blue, blueviolet, brown, burlywood, cadetblue, chartreuse, chocolate, coral, cornflowerblue, cornsilk, crimson, cyan, darkblue, darkcyan, darkgoldenrod, darkgray, darkgreen, darkgrey, darkkhaki, darkmagenta, darkolivegreen, darkorange, darkorchid, darkred, darksalmon, darkseagreen, darkslateblue, darkslategray, darkslategrey, darkturquoise, darkviolet, deeppink, deepskyblue, dimgray, dimgrey, dodgerblue, firebrick, floralwhite, forestgreen, fuchsia, gainsboro, ghostwhite, gold, goldenrod, gray, green, greenyellow, grey, honeydew, hotpink, indianred, indigo, ivory, khaki, lavender, lavenderblush, lawngreen, lemonchiffon, lightblue, lightcoral, lightcyan, lightgoldenrodyellow, lightgray, lightgreen, lightgrey, lightpink, lightsalmon, lightseagreen, lightskyblue, lightslategray, lightslategrey, lightsteelblue, lightyellow, lime, limegreen, linen, magenta, maroon, mediumaquamarine, mediumblue, mediumorchid, mediumpurple, mediumseagreen, mediumslateblue, mediumspringgreen, mediumturquoise, mediumvioletred, midnightblue, mintcream, mistyrose, moccasin, navajowhite, navy, oldlace, olive, olivedrab, orange, orangered, orchid, palegoldenrod, palegreen, paleturquoise, palevioletred, papayawhip, peachpuff, peru, pink, plum, powderblue, purple, rebeccapurple, red, rosybrown, royalblue, saddlebrown, salmon, sandybrown, seagreen, seashell, sienna, silver, skyblue, slateblue, slategray, slategrey, snow, springgreen, steelblue, tan, teal, thistle, tomato, turquoise, violet, wheat, white, whitesmoke, yellow, yellowgreen
- transparent

# Unités - Les couleurs (RGB)

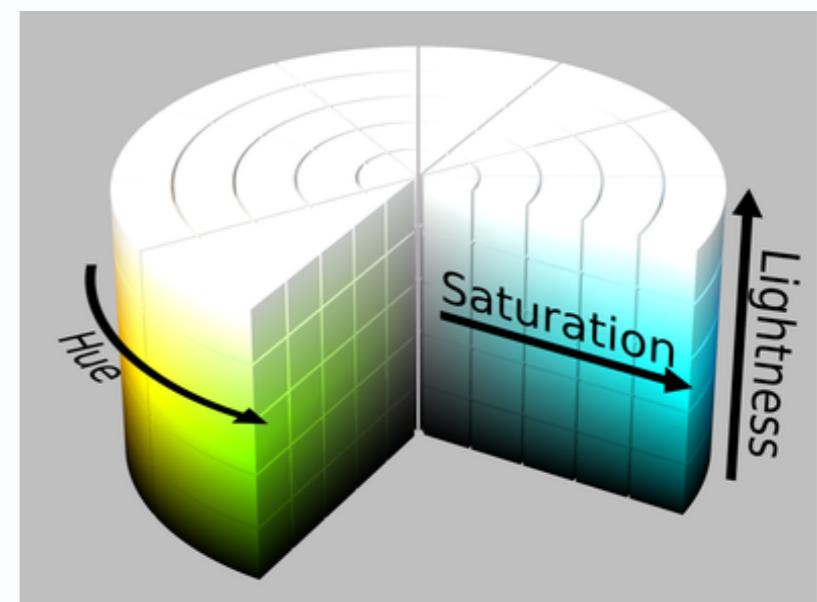
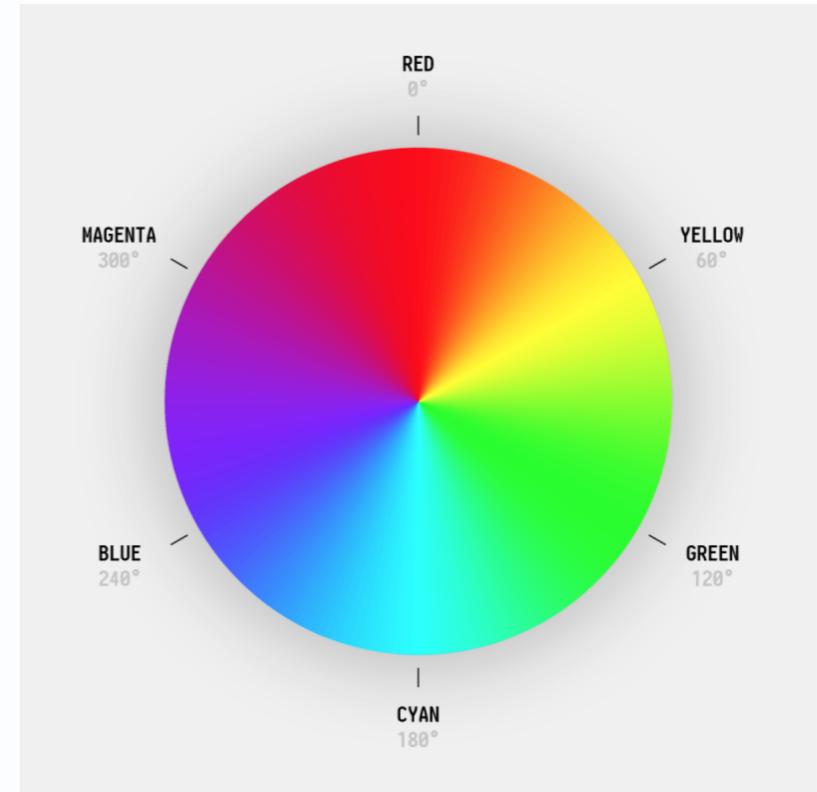


- ▶ Dans le système RGB les couleurs peuvent se définir
  - en hexadécimal sur 6 octets :  
#000000 (noir), #ffffff (blanc), #ff0000 (rouge)
  - en hexadécimal sur 3 octets :  
#000 (noir), #fff (blanc), #f00 (rouge)
  - en appelant la fonction rgb :  
rgb(0, 0, 0) (noir), rgb(255, 255, 255) (blanc), rgb(255, 0, 0) (rouge)  
rgb(0%, 0%, 0%) (noir), rgb(100%, 100%, 100%) (blanc), rgb(100%, 0%, 0%) (rouge)
- ▶ Pour ajouter de la transparence
  - en hexadécimal sur 6 octets : #ff00007f (rouge transparent à 50%)
  - en hexadécimal sur 3 octets : #f007 (rouge transparent à 50%)
  - en appelant la fonction rgba : rgba(255, 0, 0, 0.5), rgba(100%, 0%, 0%, 50%)

# Unités - Les couleurs (HSL)



- ▶ Dans le système HSL on utilise la fonction hsl qui prend 3 paramètres
  - la teinte comprise en 0 et 360
  - la saturation
  - la luminosité
- ▶ Exemple :  
`hsl(0, 100%, 50%)`
- ▶ Avec de la transparence :  
`hsl(0, 100%, 50%, 0.5)`



# Unités - Autres unités



- Les tailles pour l'impression :  
pt, cm, mm, in, pc
- Les autres tailles :  
cap, ch, ex, ic, lh, rlh, vi, vb, Q
- Les angles :  
deg, rad, turn, grad
- Le temps :  
ms, s
- Les fréquences :  
hz
- Les fractions :  
fr



**formation.tech**

# Positionnement CSS

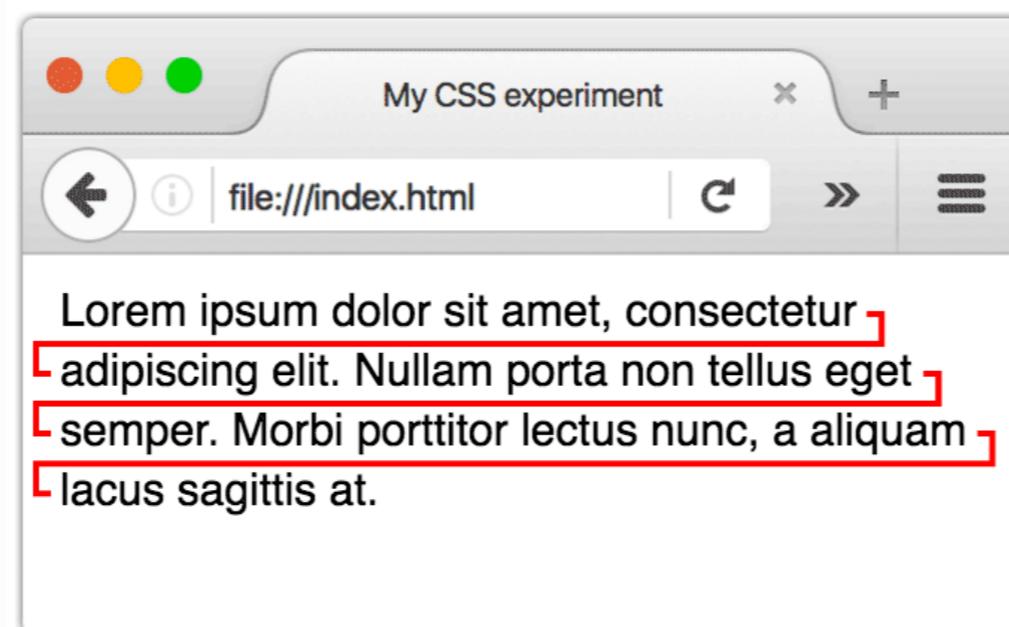


# Positionnement CSS

## ▶ Flux

Les balises HTML se positionnent dans leur ordre d'arrivée, de gauche à droite, puis de haut en bas s'il n'y a plus de place sur la ligne (comme des mots au sein d'un texte).

Certaines balises peuvent forcer le passage à la ligne.





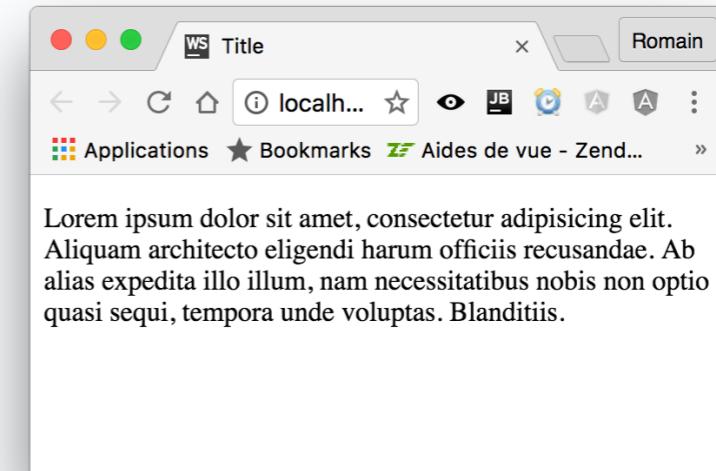
# Positionnement CSS

## ▶ Gestion des espaces

Le navigateur supprime les espaces et retour à la ligne qui précèdent le premier mot, idem après ceux du dernier mots.

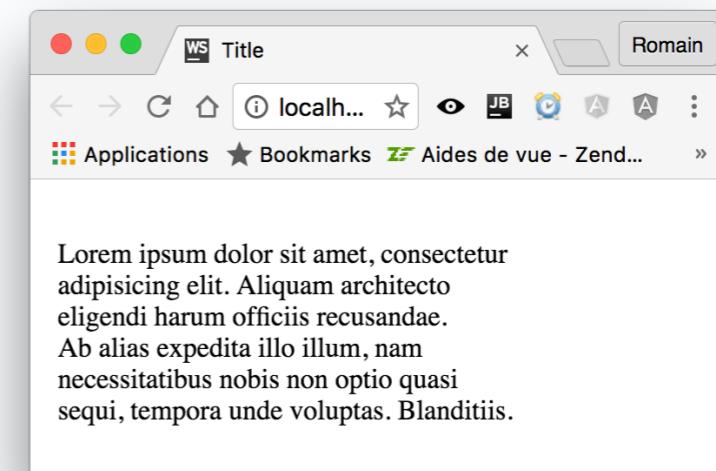
Entre 2 mots, plusieurs espaces ou retour à la ligne sont transformés en un seul espace.

```
<p>
    Lorem ipsum dolor sit amet, consectetur
    adipisicing elit. Aliquam architecto
    eligendi harum officiis recusandae.
    Ab alias expedita illo illum, nam
    necessitatibus nobis non optio quasi
    sequi, tempora unde voluptas. Blanditiis.
</p>
```



## ▶ Il est possible de préserver les espaces avec la propriété white-space (comportement par défaut des balises pre et textarea) :

```
p {
    white-space: pre;
}
```





# Positionnement CSS

## ▶ Display

Il y a 4 valeurs historiques pour la propriétés display.

- none

Cette valeur retire l'élément du flux, comme si l'élément et son contenu n'existaient pas.

- inline

Cette valeur rend l'élément transparent au sens où il s'inscrit dans le flux de texte global, il est donc associé au texte l'environnant. Il n'est pas possible de modifier sa largeur et sa hauteur.

Comportement par défaut des balises : a, span, b, i, strong, em...

- block

Cette valeur cassera le flux de texte pour insérer l'élément. Cela provoquera donc un saut de ligne avant et après. Le contenu de cet élément ne fait donc pas partie du flux global et suit donc les contraintes de l'élément définies par le modèle de boîte. Largeur par défaut 100%, hauteur auto.

Comportement par défaut des balises : div, p, h1, h2, h3, form, header, footer...

- inline-block

Cette valeur est en quelque sorte un intermédiaire entre inline et block. Comme avec inline, les boîtes seront placées dans le flux global mais , comme avec block, le contenu ne fera pas partie du texte environnant. Il est possible de modifier sa largeur et sa hauteur.



# Positionnement CSS

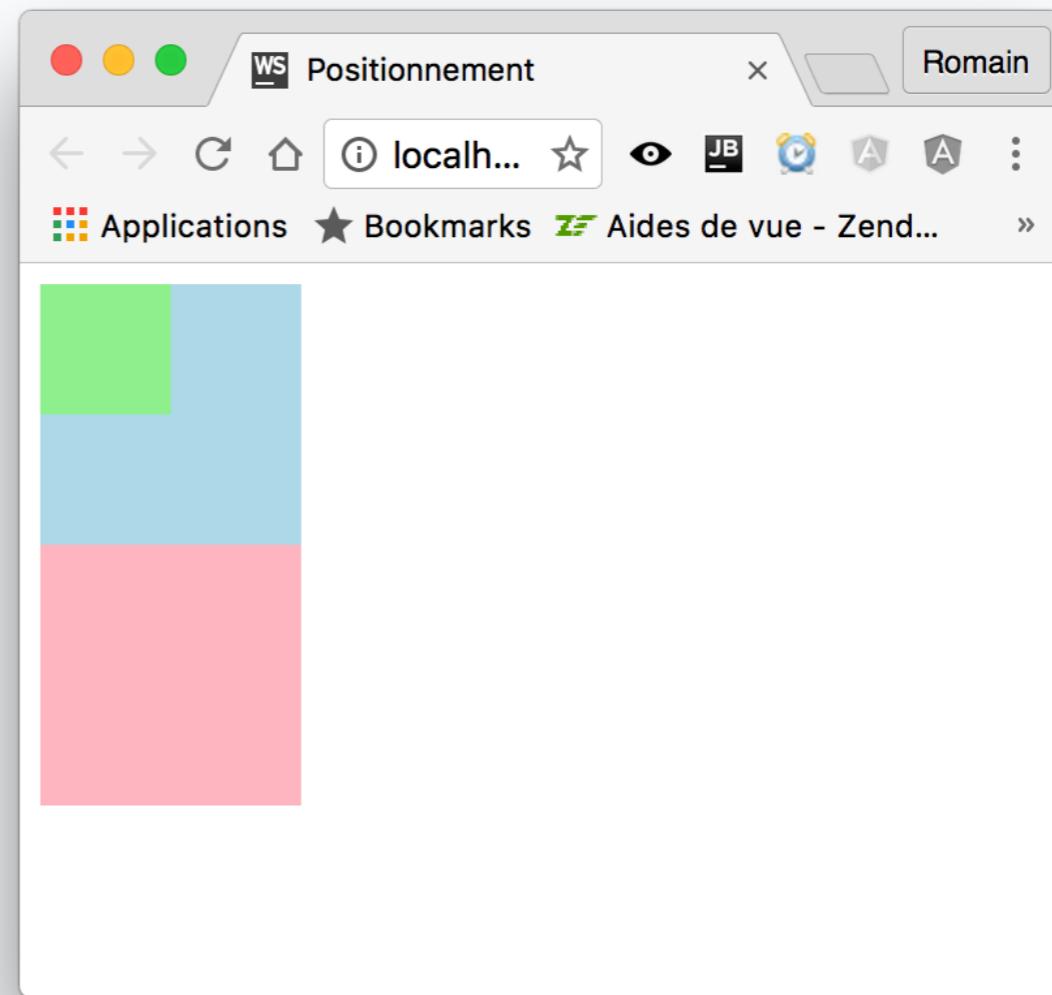
- ▶ Soit le HTML + CSS suivant

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Position Static</title>
    <style>
        .externe {
            width: 100px;
            height: 100px;
            background-color: lightblue;
        }

        .interne {
            width: 50px;
            height: 50px;
            background-color: lightgreen;
        }

        .suivant {
            width: 100px;
            height: 100px;
            background-color: lightpink;
        }

    </style>
</head>
<body>
    <div class="externe">
        <div class="interne"></div>
    </div>
    <div class="suivant"></div>
</body>
</html>
```

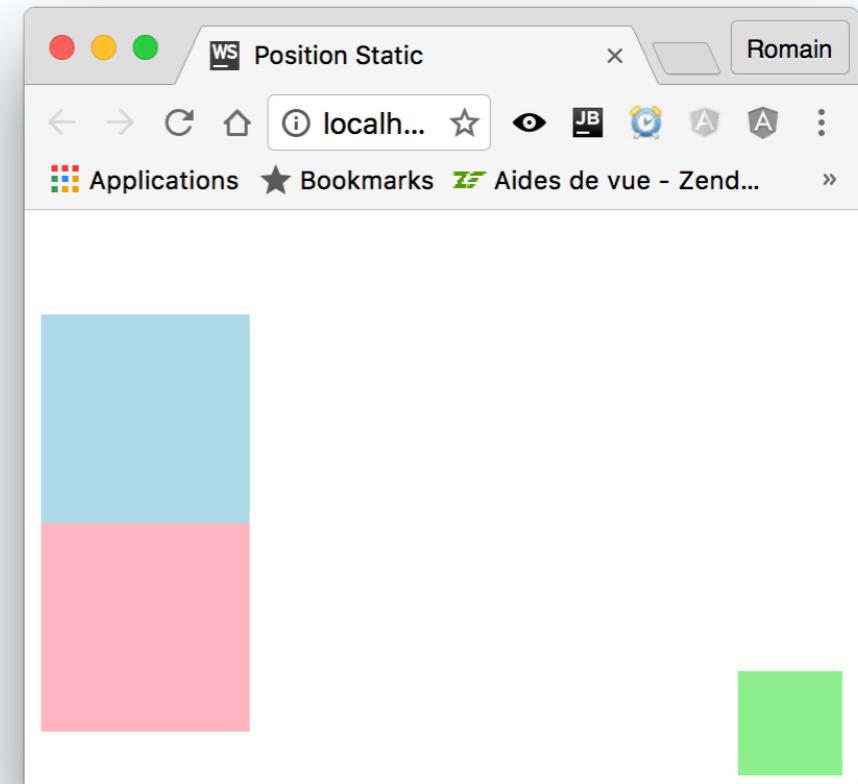




# Positionnement CSS

## ▶ Positionnement static (valeur par défaut)

```
.externe {  
    position: static; /* valeur par défaut */  
    margin-top: 50px;  
    top: 20px; /* inactif */  
    left: 20px; /* inactif */  
}  
  
.interne {  
    position: absolute;  
    bottom: 8px;  
    right: 8px;  
}
```



## ▶ Remarques :

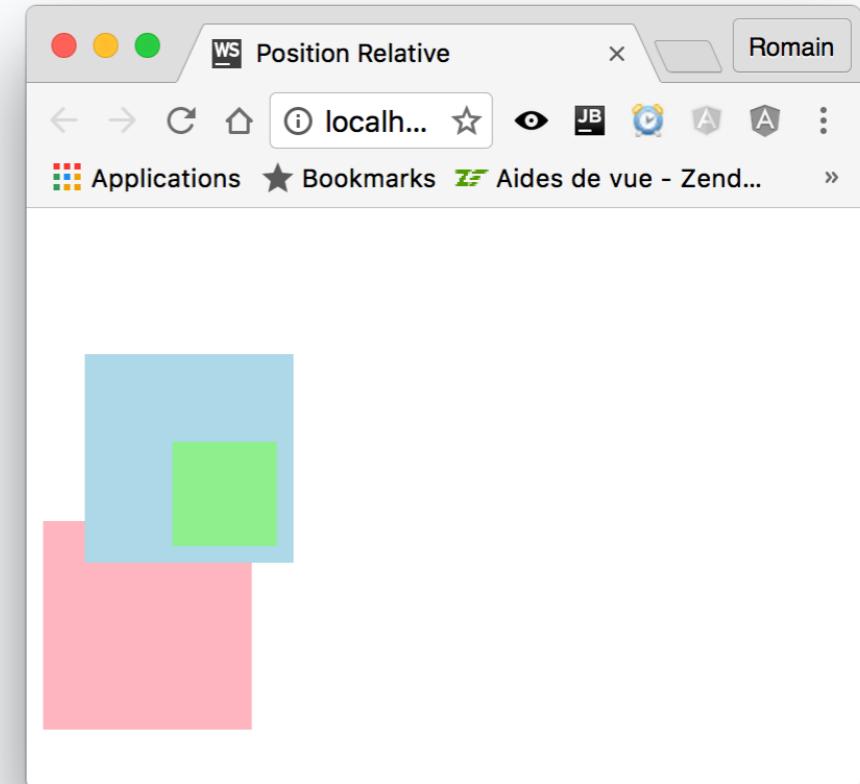
- positionnement dans le flux, sa taille et ses marges impactent la suite du flux
- top, left, bottom, right inactifs
- l'élément interne (position absolu) s'est positionné par rapport à la page



# Positionnement CSS

## ▶ Positionnement relative

```
.externe {  
    position: relative;  
    margin-top: 50px;  
    top: 20px;  
    left: 20px;  
}  
  
.interne {  
    position: absolute;  
    bottom: 8px;  
    right: 8px;  
}
```



## ▶ Remarques :

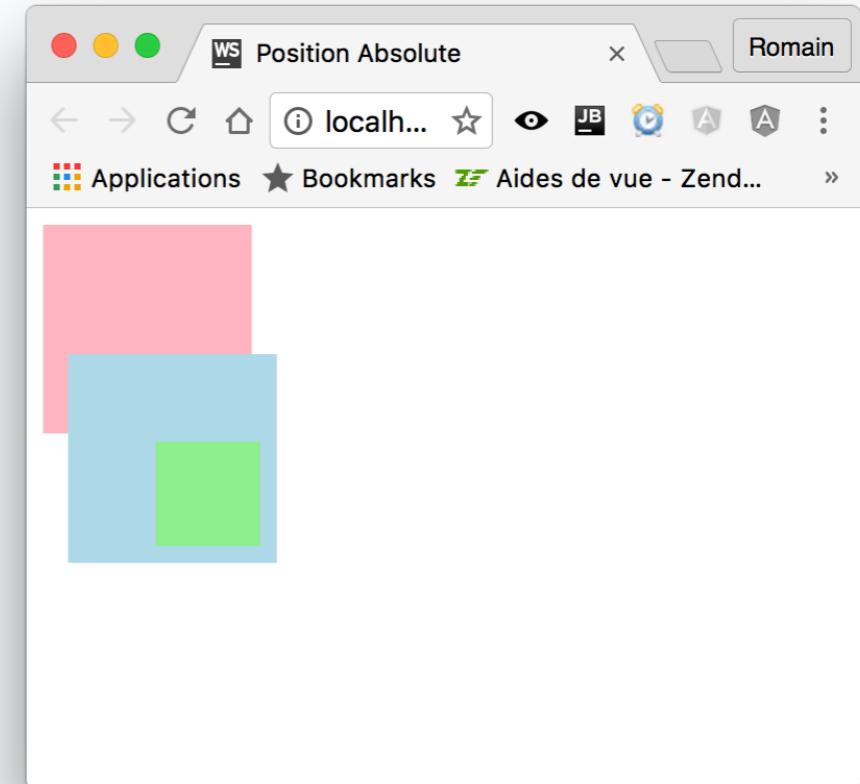
- positionnement dans le flux, sa taille et ses marges impactent la suite du flux
- top, left, bottom, right déplacent l'élément par rapport à sa position d'origine, sans impacter la suite du flux
- l'élément interne (position absolu) s'est positionné par rapport à son ancêtre non static le plus proche



# Positionnement CSS

## ▶ Positionnement absolute

```
.externe {  
    position: absolute;  
    margin-top: 50px;  
    top: 20px;  
    left: 20px;  
}  
  
.interne {  
    position: absolute;  
    bottom: 8px;  
    right: 8px;  
}
```



## ▶ Remarques :

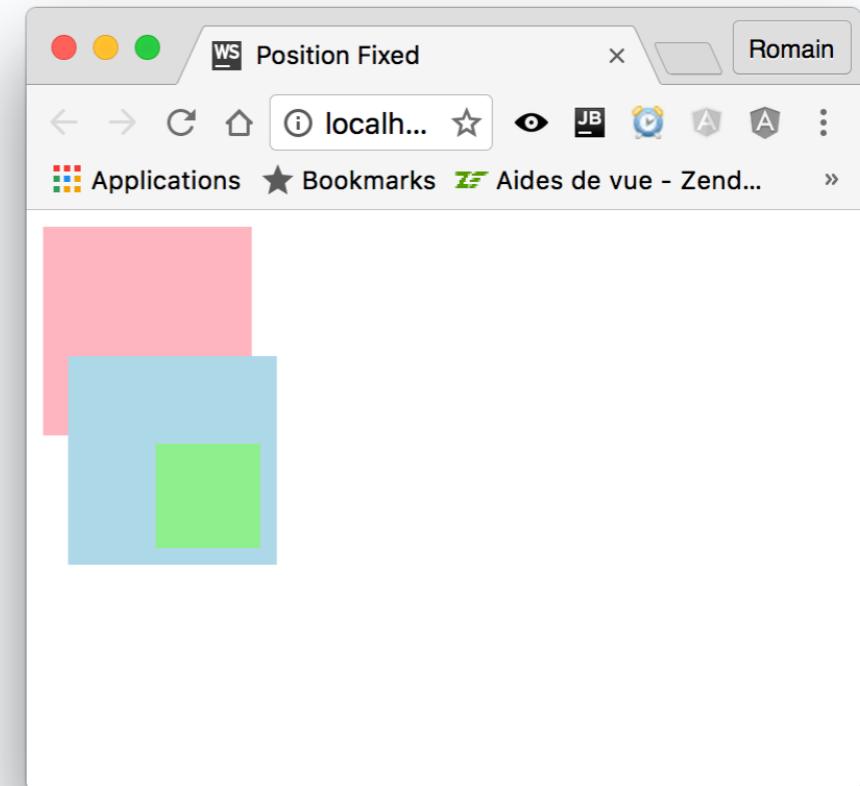
- positionnement hors flux, les éléments suivants se positionnent comme cet élément n'existe pas
- top, left, bottom, right place l'élément par rapport à la page (si que des ancêtres static)
- l'élément interne (position absolu) s'est positionné par rapport à son ancêtre non static le plus proche



# Positionnement CSS

## ▶ Positionnement fixed

```
.externe {  
    position: fixed;  
    margin-top: 50px;  
    top: 20px;  
    left: 20px;  
}  
  
.interne {  
    position: absolute;  
    bottom: 8px;  
    right: 8px;  
}
```



## ▶ Remarques :

- positionnement hors flux, les éléments suivants se positionnent comme cet élément n'existe pas
- top, left, bottom, right placent l'élément par rapport à la fenêtre
- l'élément interne (position absolu) s'est positionné par rapport à son ancêtre non static le plus proche



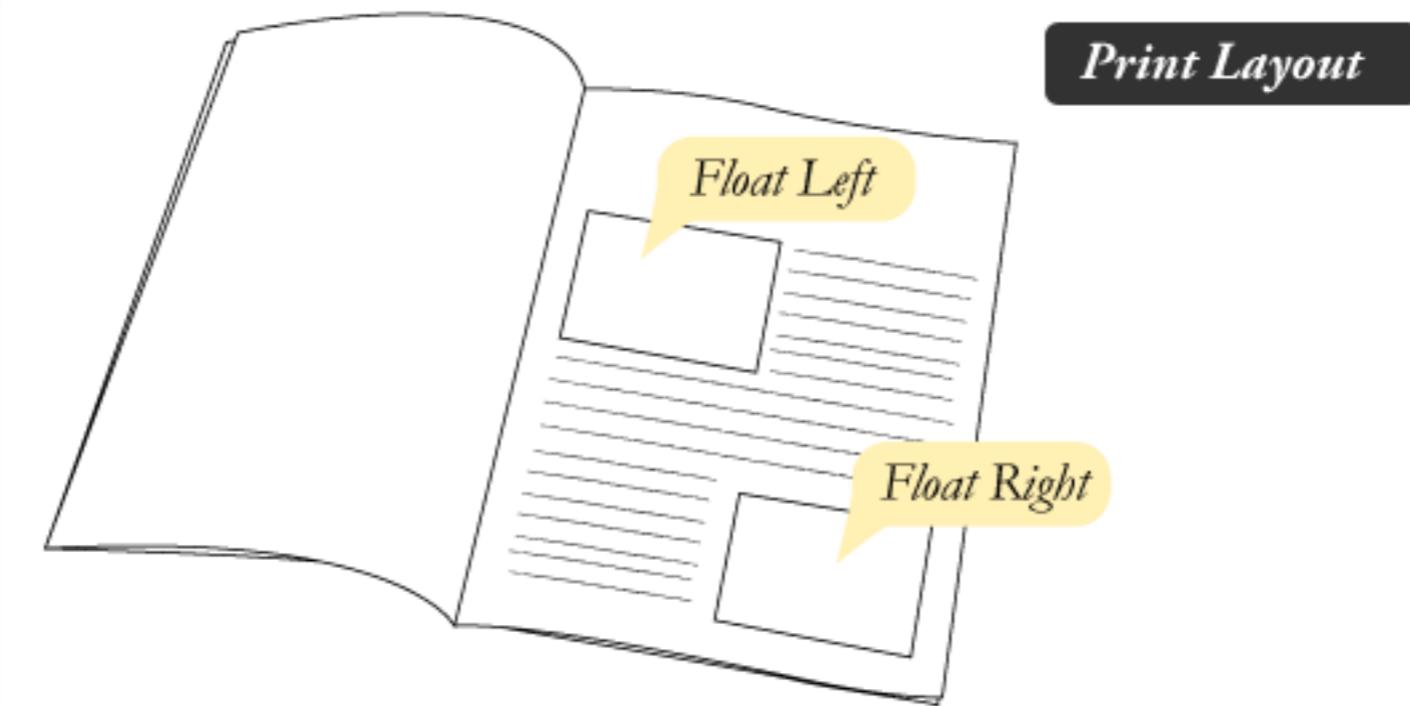
# Positionnement CSS

## ▶ Float

Permet d'encadrer du texte autour d'un élément.

## ▶ Clear

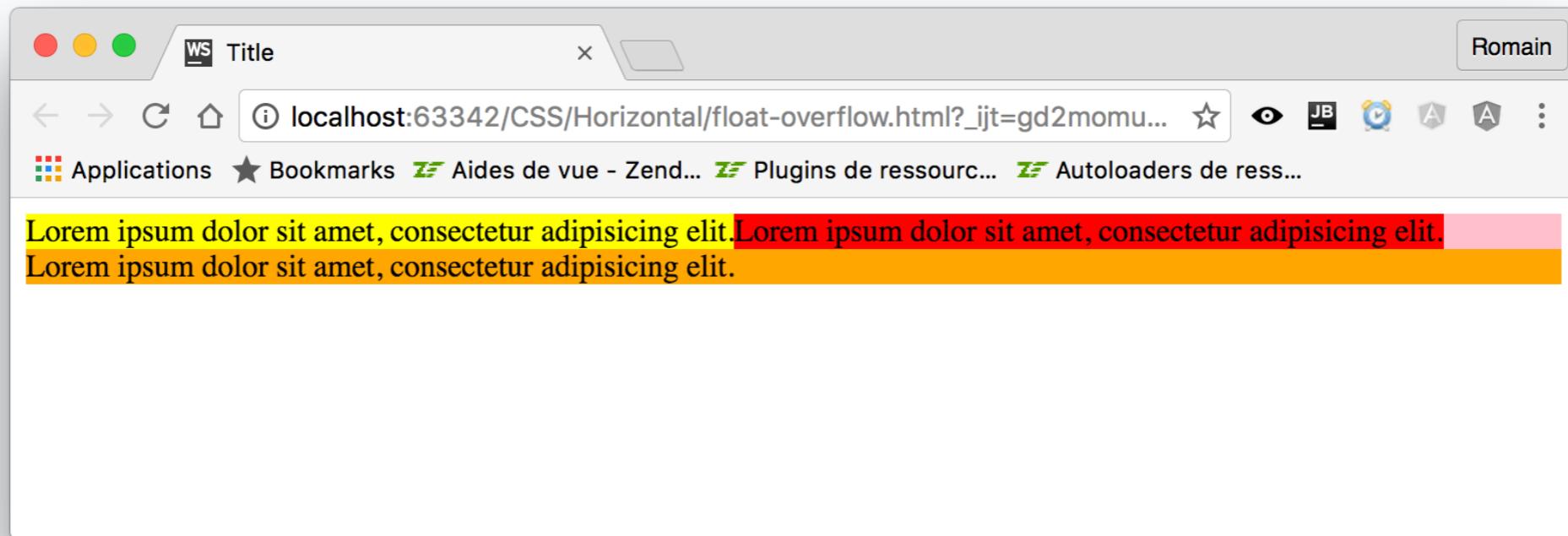
On utilise la propriété clear pour qu'un élément arrête d'encadrer le texte. Il est également possible d'utiliser overflow sur un élément conteneur.





# Positionnement CSS

## ▶ Placer 2 éléments block horizontalement

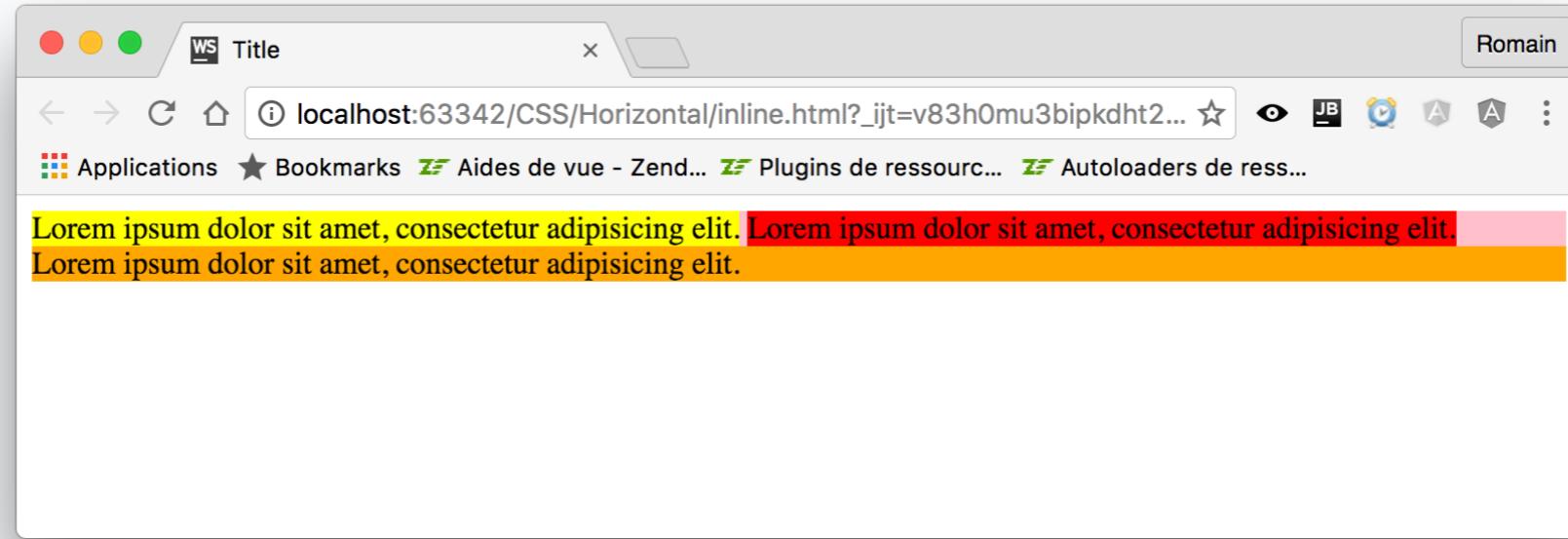


```
<div class="container pink">
  <div class="col yellow">Lorem ipsum dolor sit amet, consectetur adipisicing elit.</div>
  <div class="col red">Lorem ipsum dolor sit amet, consectetur adipisicing elit.</div>
</div>
<div class="next orange">Lorem ipsum dolor sit amet, consectetur adipisicing elit.</div>
```



# Positionnement CSS

- ▶ Placer 2 éléments block horizontalement



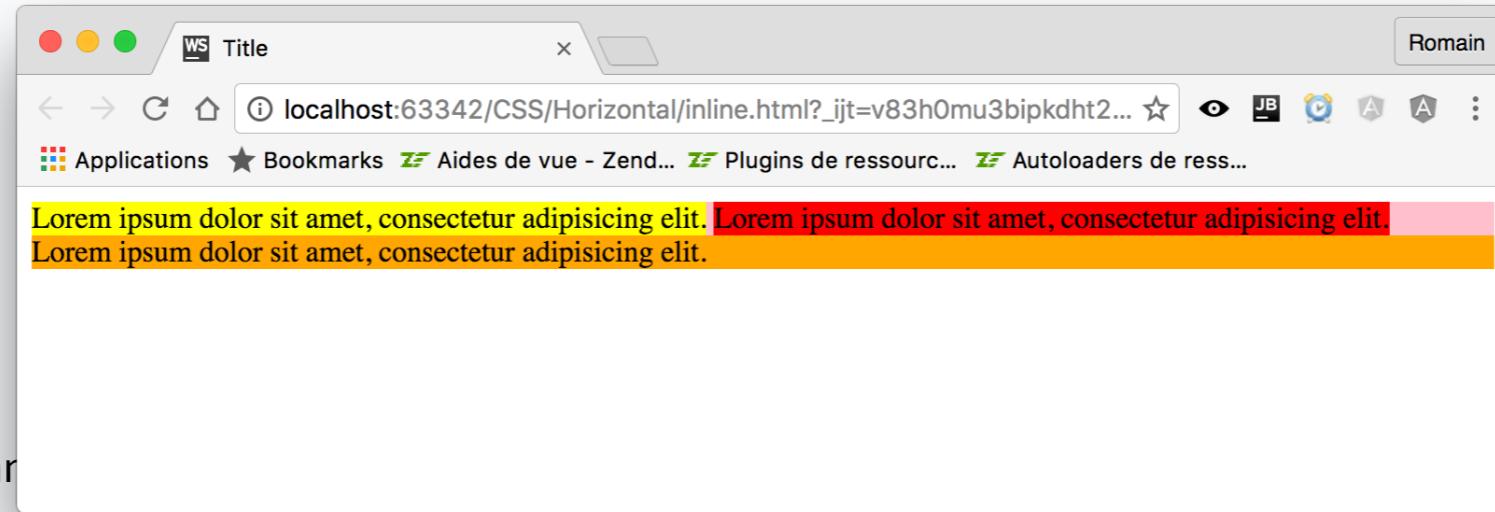
- ▶ Pas possible de modifier la largeur et hauteur
- ▶ Espace les 2 colonnes

```
.col {  
  display: inline;  
}
```



# Positionnement CSS

- ▶ Placer 2 éléments block horizontalement



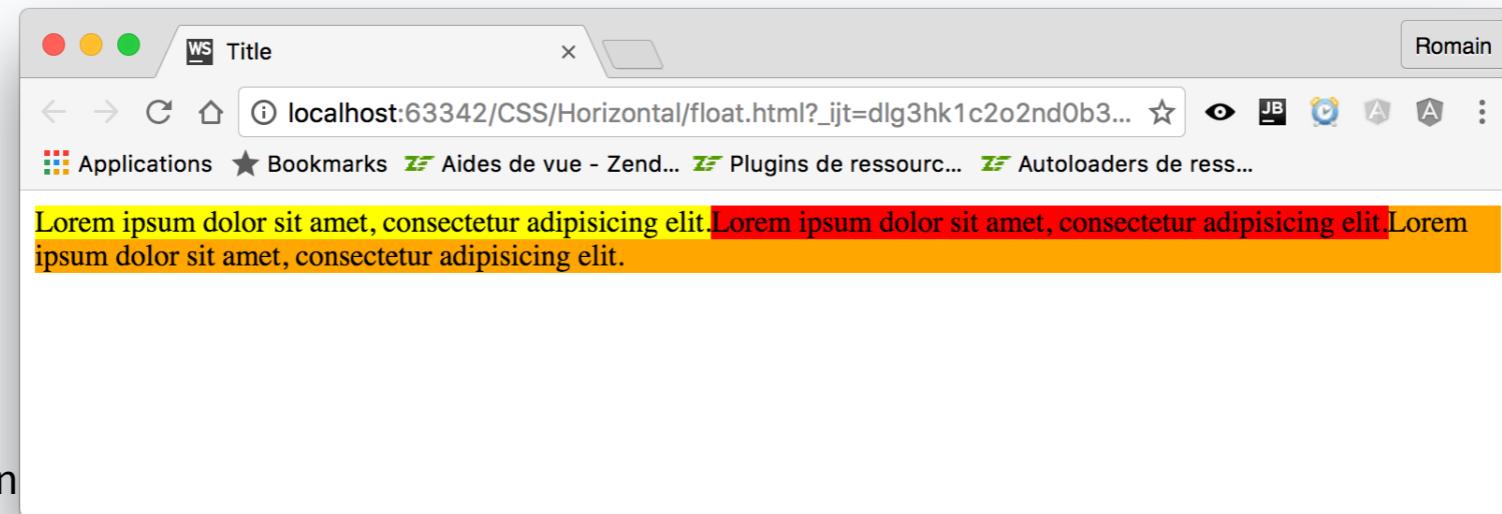
- ▶ Espace les 2 colonnes

```
.col {  
  display: inline-block;  
}
```



# Positionnement CSS

- ▶ Placer 2 éléments block horizontalement



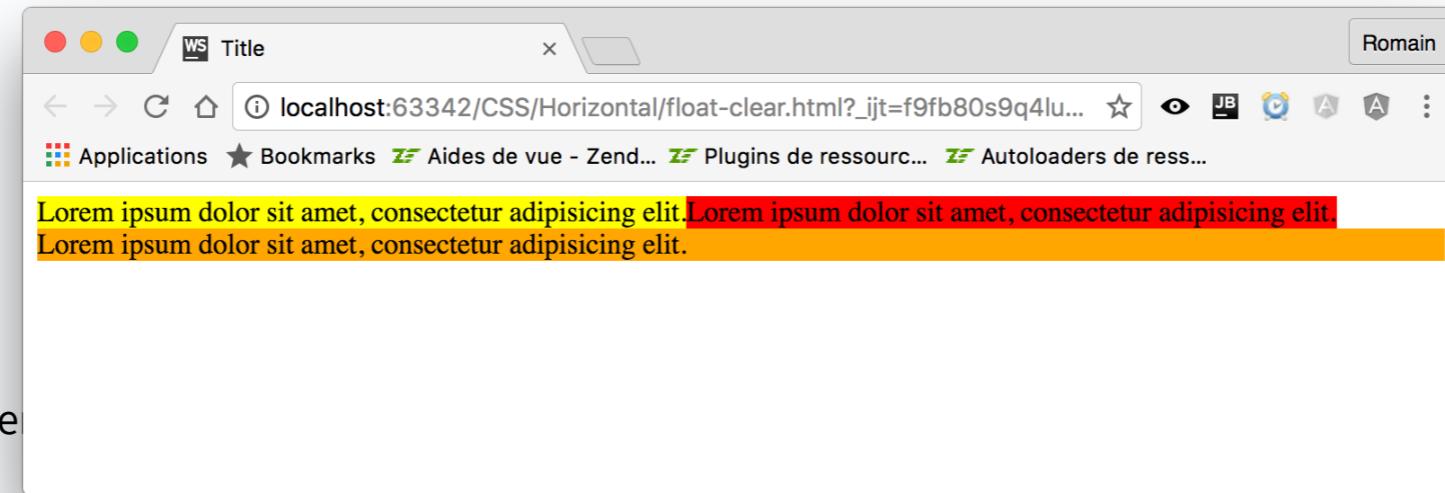
- ▶ Le texte suivant en

```
.col {  
  float: left;  
}
```



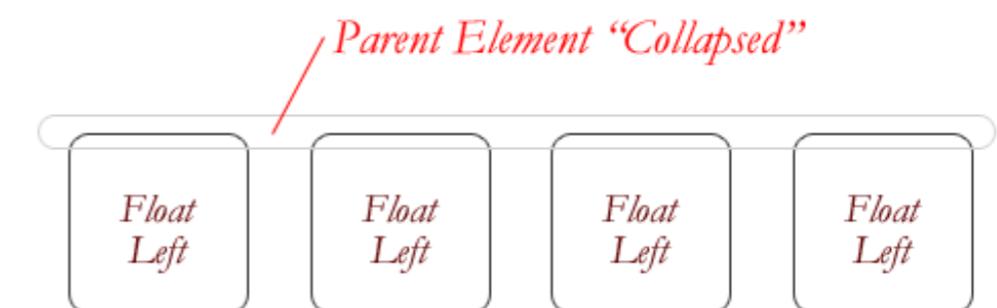
# Positionnement CSS

- ▶ Placer 2 éléments block horizontalement



- ▶ La hauteur du conte

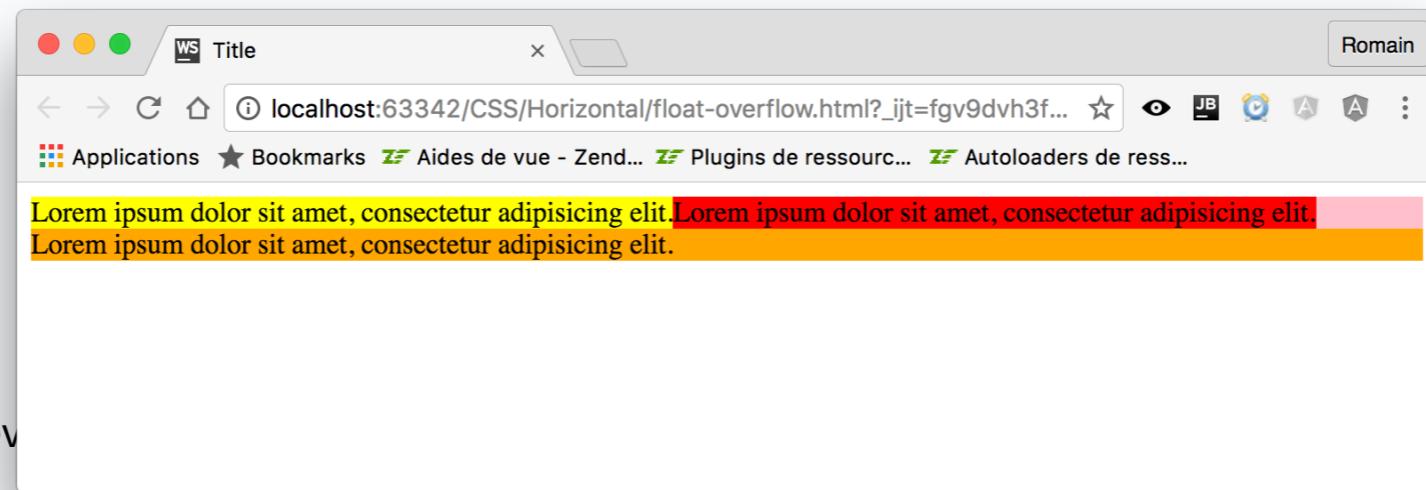
```
.col {  
  float: left;  
}  
  
.next {  
  clear: left;  
}
```





# Positionnement CSS

- ▶ Placer 2 éléments block horizontalement



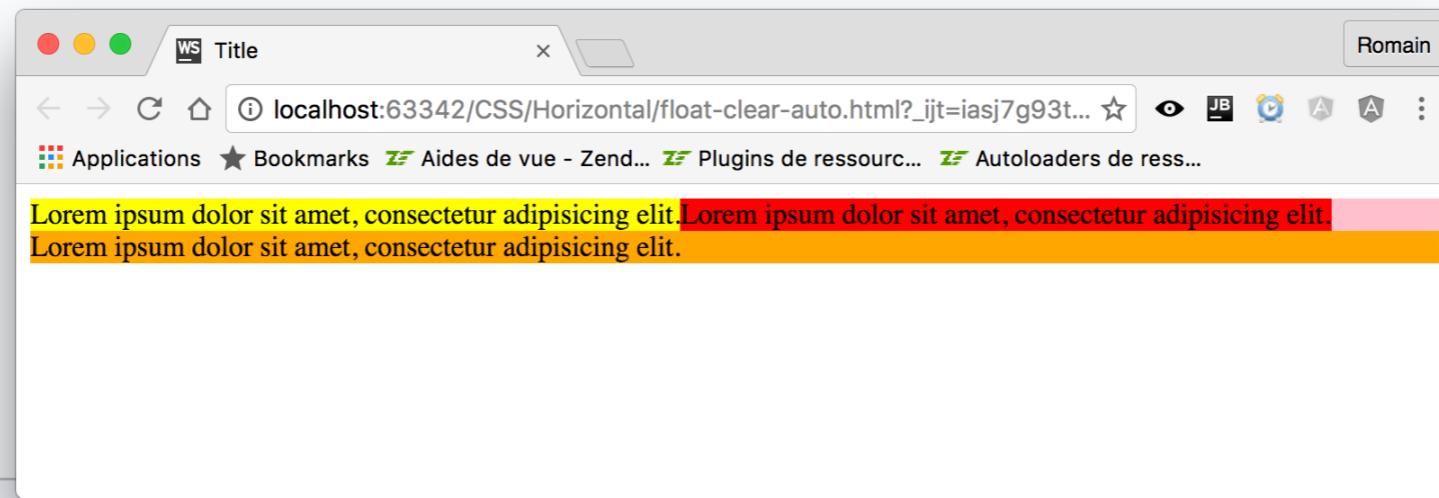
- ▶ Un dépassement dev

```
.container {  
    overflow: hidden;  
}  
  
.col {  
    float: left;  
}
```



# Positionnement CSS

- ▶ Placer 2 éléments block horizontalement

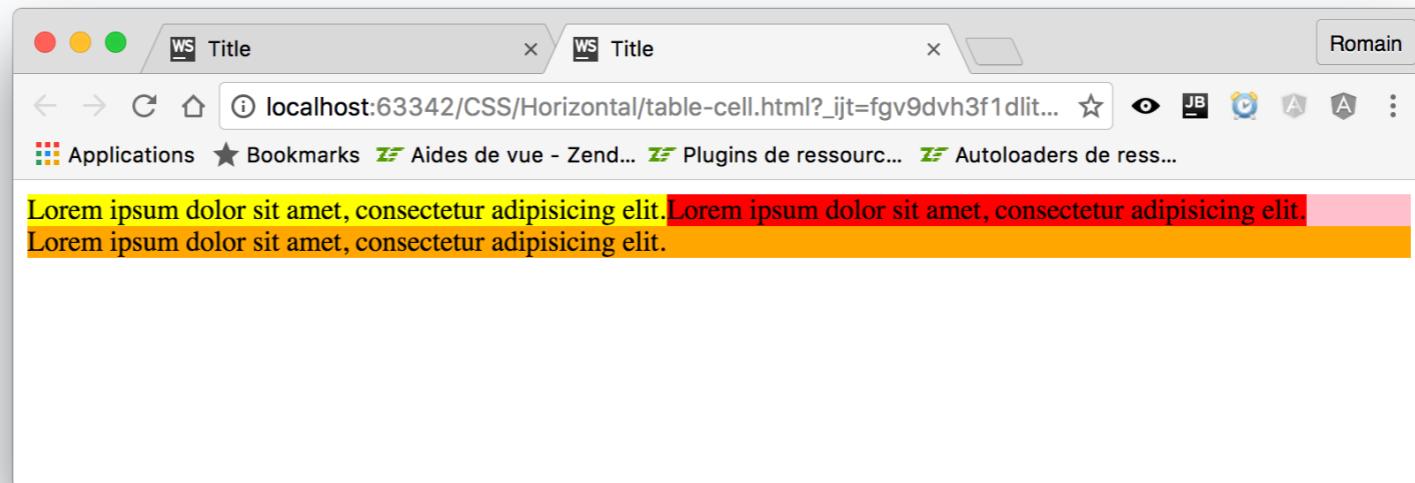


```
.col {  
    float: left;  
}  
  
.container::after {  
    content: "";  
    display: block;  
    clear: left;  
}
```

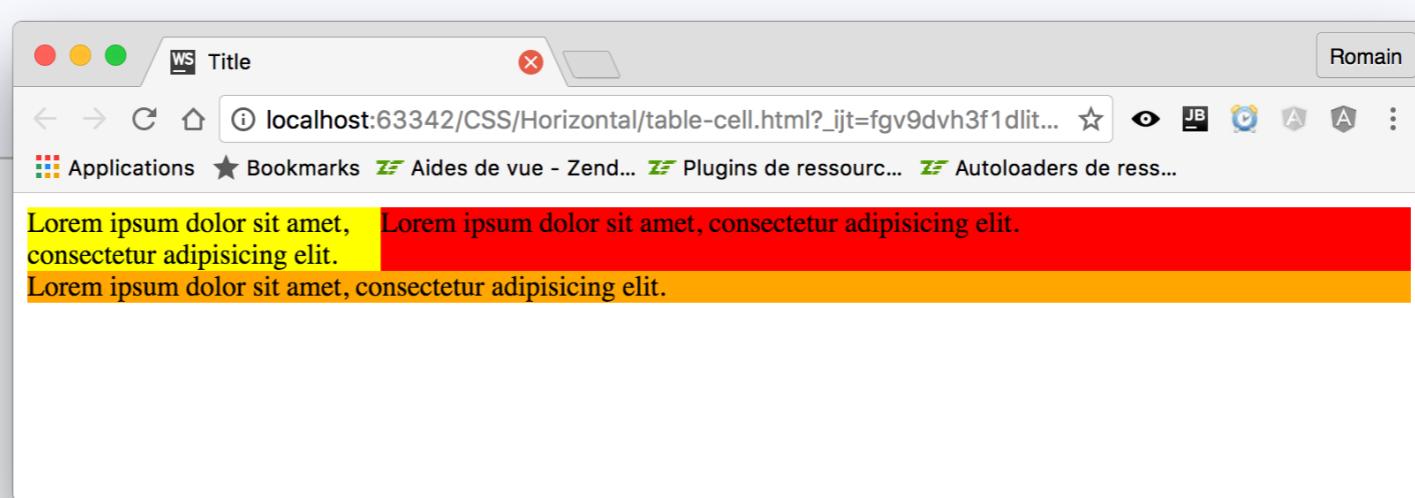


# Positionnement CSS

- ▶ Placer 2 éléments block horizontalement



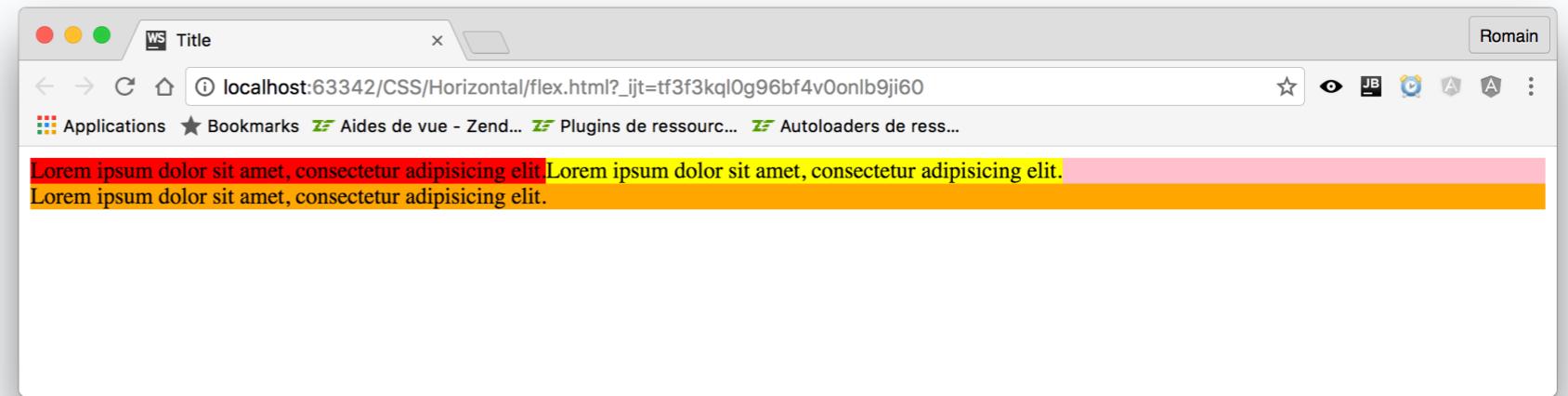
```
.container {  
    display: table;  
    width: 100%;  
}  
  
.col {  
    display: table-cell;  
}  
  
.col.yellow {  
    width: 200px;  
}
```





# Positionnement CSS

- ▶ Placer 2 éléments block horizontalement



IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser	Chrome for Android
8	13	47	49	51		9.2		4.4	
11	14	48	52	9.1	39	9.3	all	4.4.4	51
		49	53	10	40				
		50	54	TP	41				
		51	55						



# Exercice

Titre		Login / Register		
Lien 1	Home > Catégorie > Produit			
Lien 2	img	img	img	img
Lien 3	img	img		



# Exercice - Les containers à placer

Titre

Login / Register

Lien 1

Lien 2

Lien 3

Home > Catégorie > Produit

img

img

img

img

img

# Exercice



- La balise verte doit se positionner en haut à droite de la balise bleue
- Les containers violet et bleu sont en display flex
- Le fil d'ariane doit utiliser display: inline ou display: inline-block et les pseudo-éléments ::after ou ::before pour créer le symbole >



**formation.tech**

# Flexbox



# Flexbox

- ▶ Nouveau type de positionnement

IE	Edge	*	Firefox	Chrome	Safari	Opera	iOS Safari	*	Opera Mini	*	Android Browser	*	Chrome for Android
				49							4.4		
8	13		47	51							4.4.4		
11	14		48	52	9.1	39	9.2						51
4				49	53	10	40		9.3	all			51
				50	54	TP	41						
				51	55								



# Flexbox

- ▶ 4 fonctionnalités principales :
  - Distribution des éléments horizontale ou verticale, avec passage à la ligne autorisé ou non,
  - Alignements et centrages horizontaux et verticaux, justifiés, répartis,
  - Réorganisation des éléments indépendamment de l'ordre du flux (DOM),
  - Gestion des espaces disponibles ou manquants (fluidité).

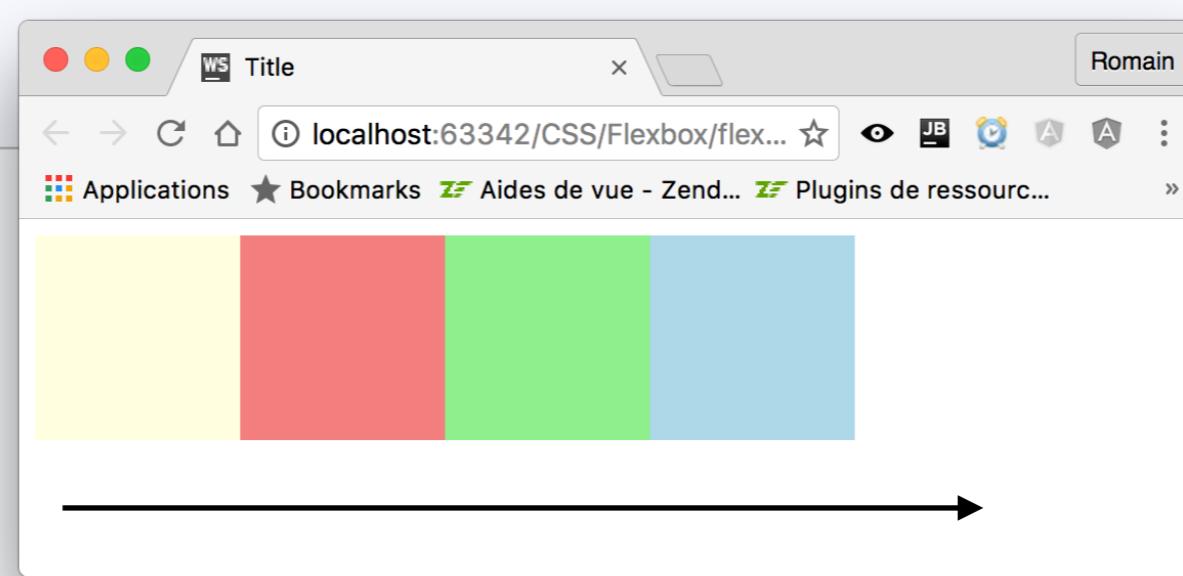


# Flexbox

- ▶ Mise en place

```
<div class="container">
  <div class="col yellow"></div>
  <div class="col red"></div>
  <div class="col green"></div>
  <div class="col blue"></div>
</div>
```

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

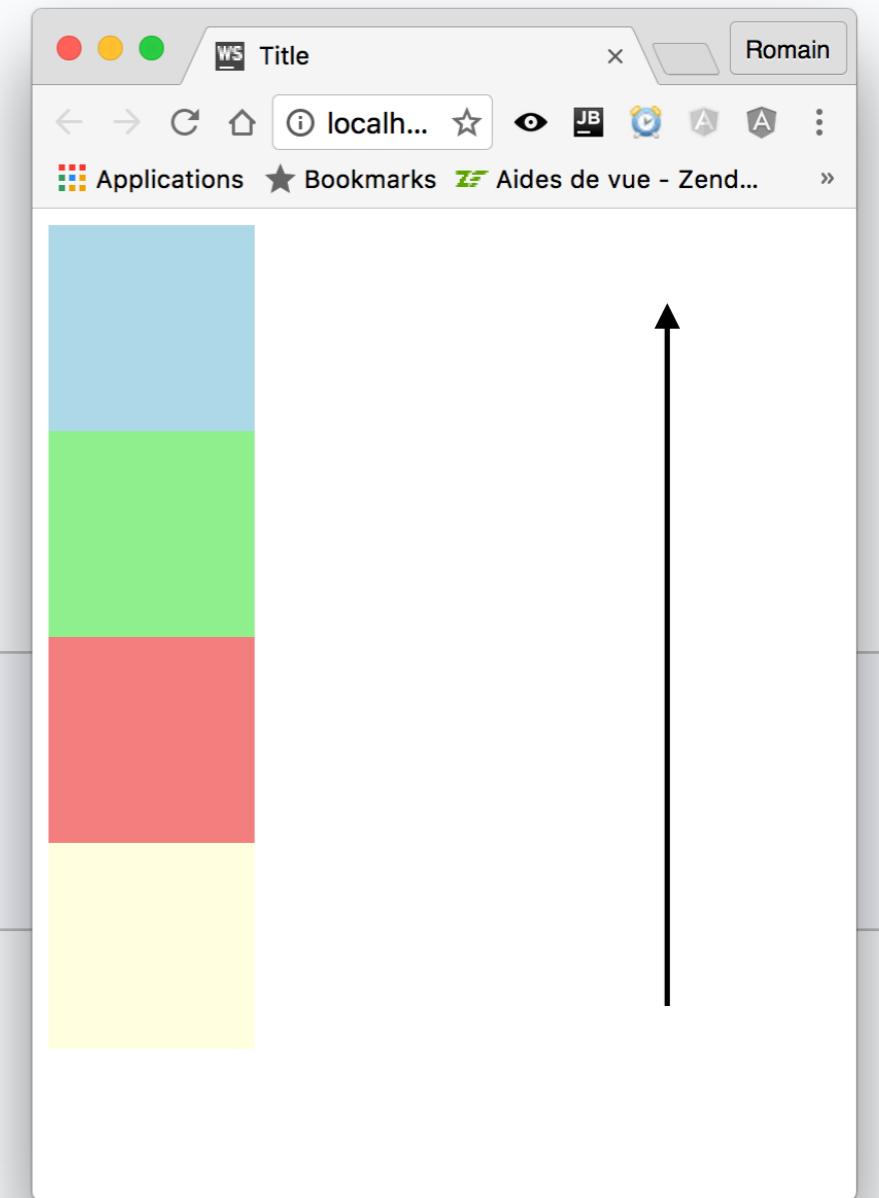




# Flexbox

## ▶ Direction

```
.container {  
  display: flex;  
  flex-direction: column-reverse;  
  /* row column row-reverse column-reverse */  
}
```



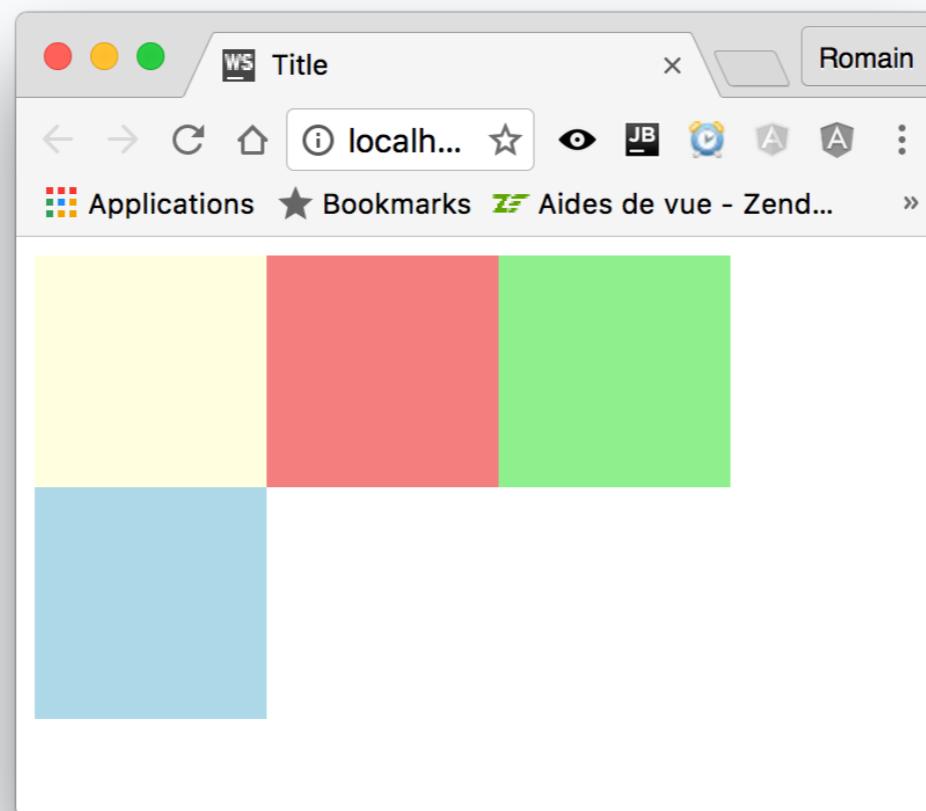


# Flexbox

- ▶ Passage à la ligne

```
.container {  
  display: flex;  
  flex-wrap: wrap;  
}
```

- ▶ Par défaut les éléments ne passent pas à la ligne
- ▶ Avec flex-wrap: wrap-reverse, la nouvelle ligne sera créée au dessus.



# Flexbox



- ▶ Répartir le contenu

```
.container {  
  display: flex;  
  justify-content: space-around;  
}
```

- ▶ flex-start (éléments positionnés au début du sens de lecture, valeur par défaut)
- ▶ flex-end (éléments positionnés à la fin)
- ▶ center (position centrale)
- ▶ space-between (répartition “justifiée”)
- ▶ space-around (variante de répartition “justifiée”)





# Flexbox

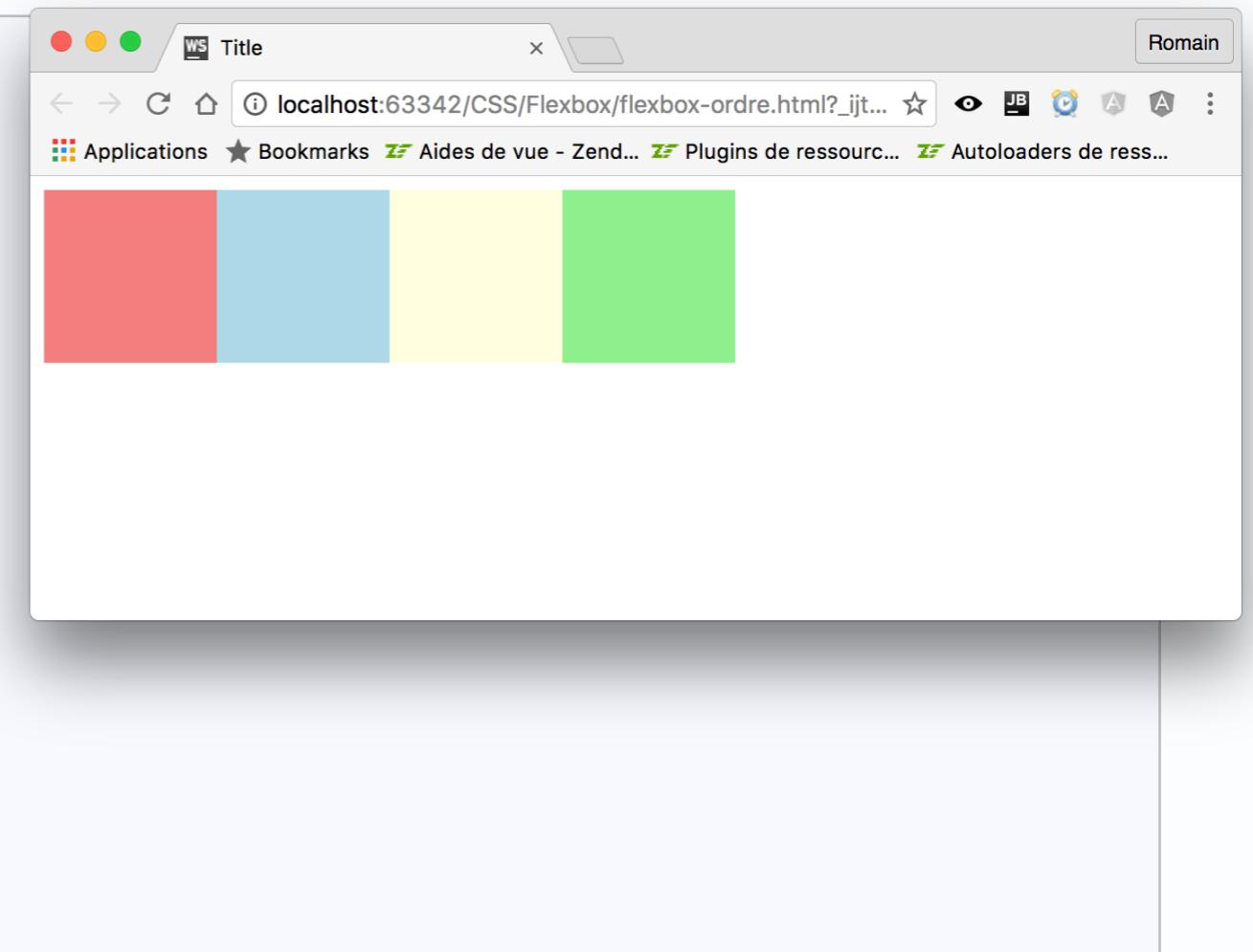
- Axe secondaire: align-items
- Dans l'axe secondaire, les alignements sont régis via la propriété align-items, dont les valeurs sont :
  - flex-start (au début)
  - flex-end (à la fin)
  - center (au centre)
  - baseline (généralement identique à flex-start)
  - stretch (étirés dans l'espace disponible, valeur par défaut)



# Flexbox

- ▶ Ordonnancement  
Il est possible de changer l'ordre. (par défaut 0)

```
.yellow {  
  background-color: lightyellow;  
  order: 3;  
}  
  
.red {  
  background-color: lightcoral;  
  order: 1;  
}  
  
.green {  
  background-color: lightgreen;  
  order: 4;  
}  
  
.blue {  
  background-color: lightblue;  
  order: 2;  
}
```

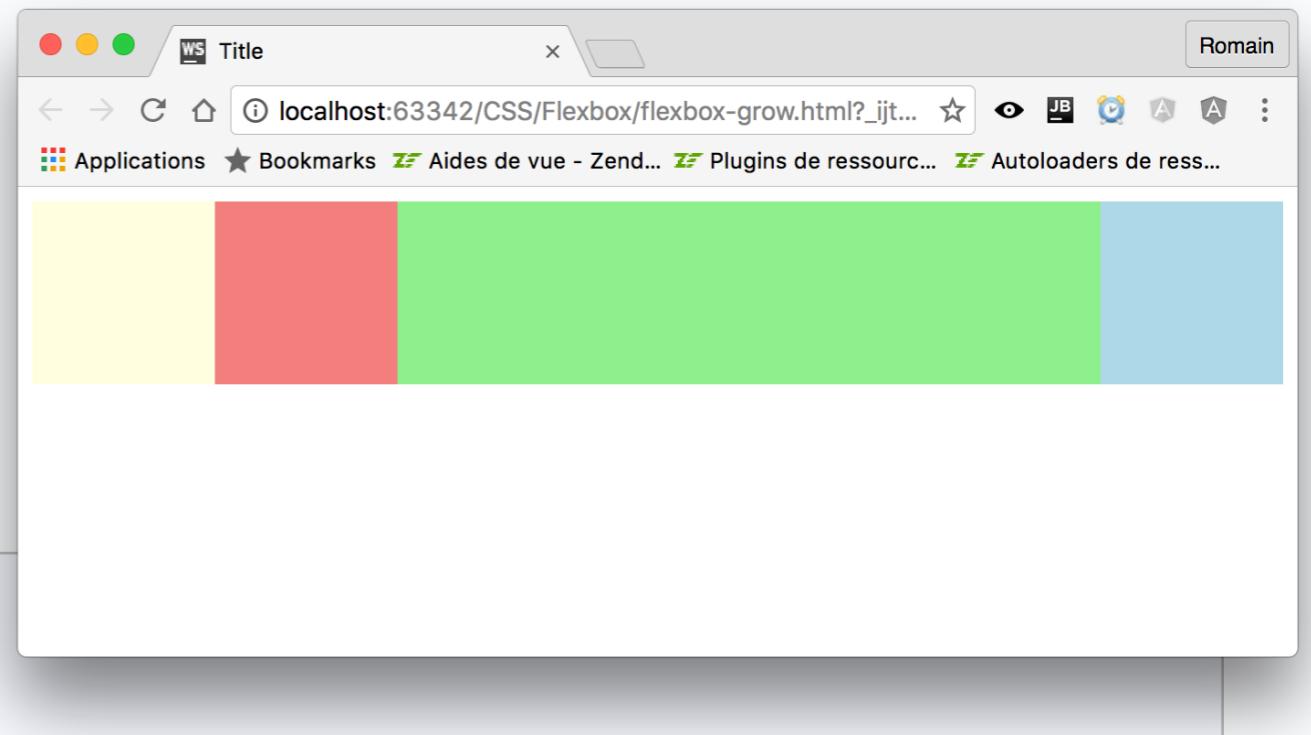




# Flexbox

- ▶ Flexibilité

Il est possible l'allouer des fractions de l'espace restant (ici 1/1) à un élément

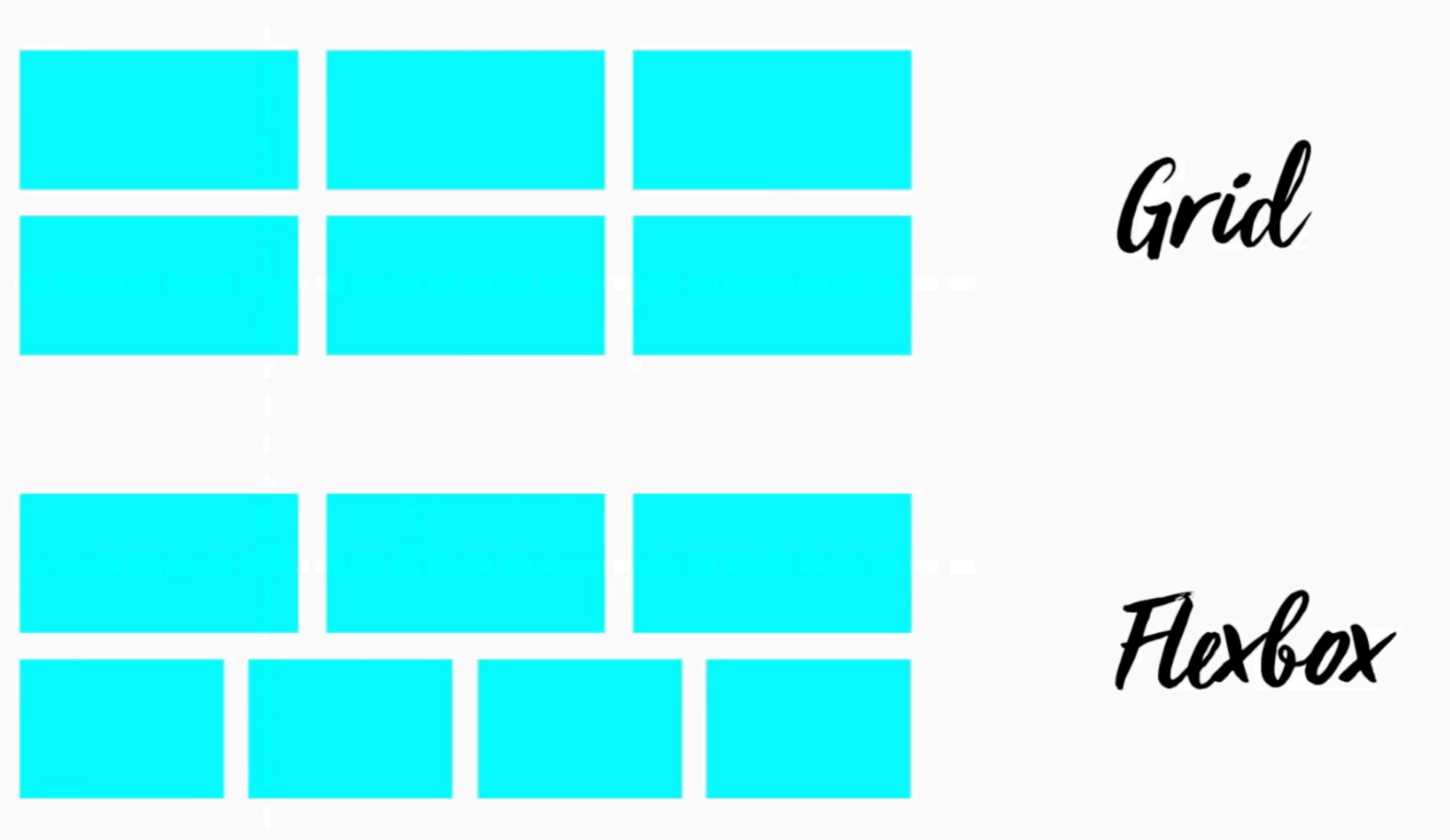




**formation.tech**

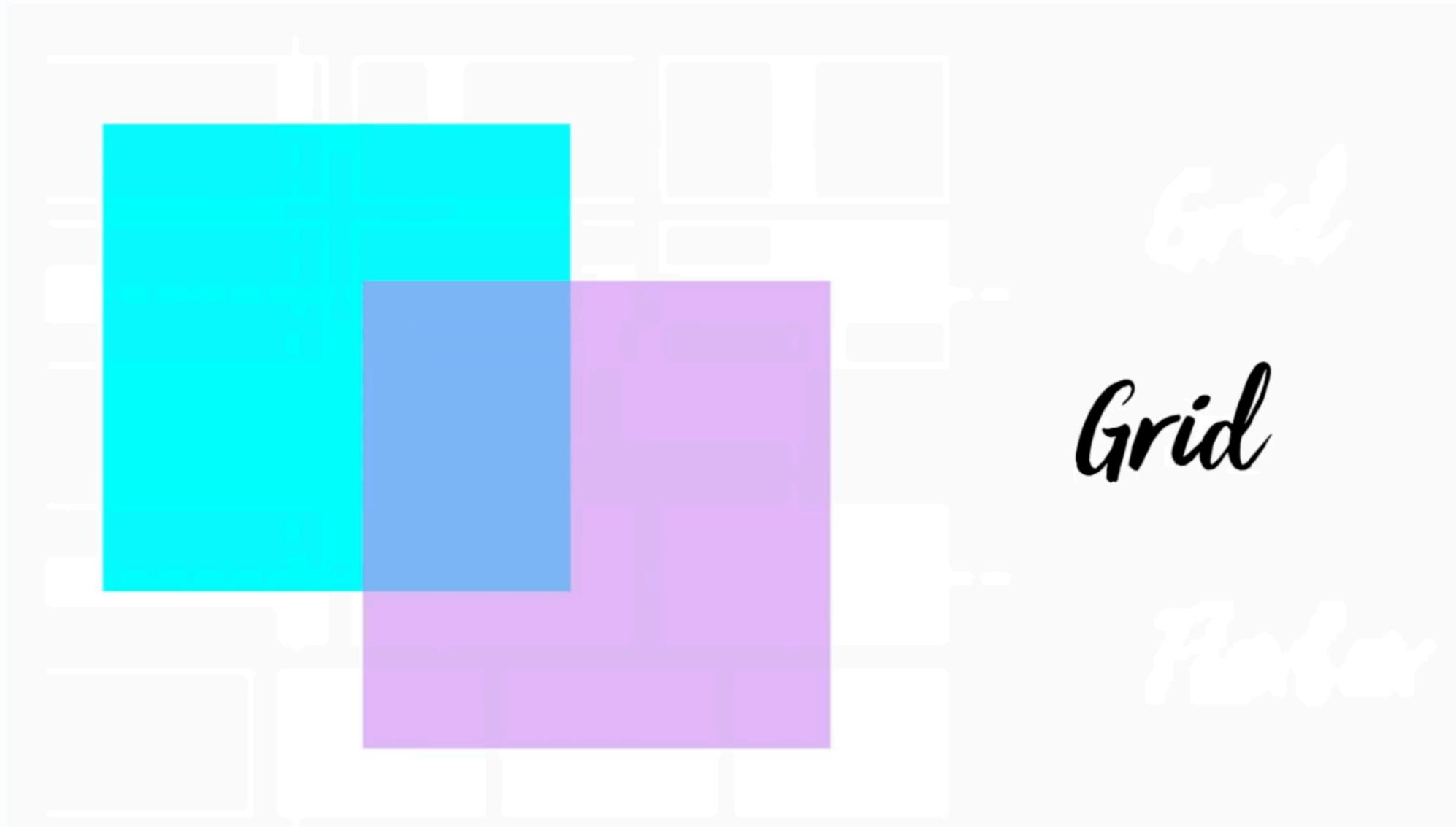
# CSS Grid

# CSS Grid - Et les Flexbox ?





# CSS Grid - Et les Flexbox ?



# CSS Grid - min/max-content



This is a phrase with several words.

This is a phrase with several words.



**max-content**

This is  
a  
phrase  
with  
several  
words.



**min-content**



**formation.tech**

# Texte



# Texte - Principales propriétés

- ▶ **color**

La couleur du texte

```
color: blue;
```

- ▶ **font-family**

Défini la police à utiliser, on peut définir une liste de préférence

```
font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial, "Noto Sans", sans-serif;
```

- ▶ **font-size**

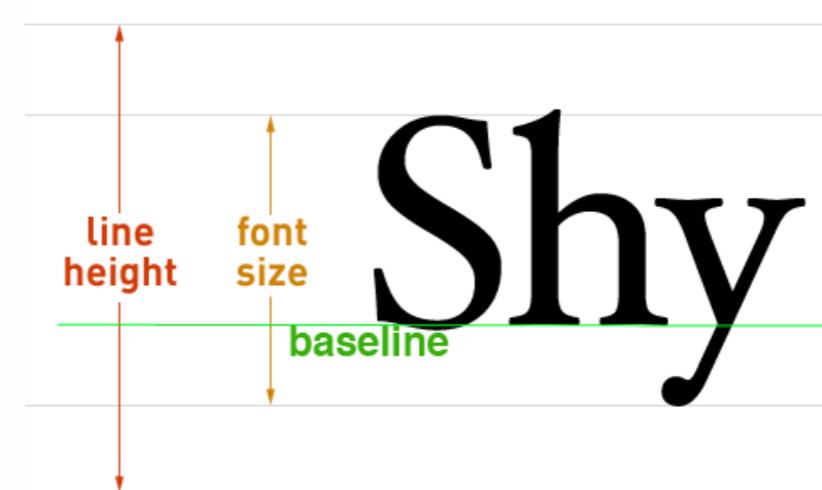
La taille de la police, privilégier les unités em ou rem

```
font-size: 1.4em;
```

- ▶ **line-height**

La hauteur de la ligne

```
line-height: 1.4;
```





# Texte - Principales propriétés

- ▶ **font-weight**

Graisse utilisée pour le texte (100 à 900, normal, bold)

`font-weight: bold;`

- ▶ Valeurs communes

- 100 - Thin (Hairline)
- 200 - Extra Light (Ultra Light)
- 300 - Light
- 400 (normal) - Normal
- 500 - Medium
- 600 - Semi Bold (Demi Bold)
- 700 (bold) - Bold
- 800 - Extra Bold (Ultra Bold)
- 900 - Black (Heavy)



# Texte - Principales propriétés

- ▶ **font-style**  
Mettre du texte en italique (normal, italic, oblique)  
**font-style: italic;**
- ▶ **text-align**  
Aligner le texte (left, right, center, justify)  
**text-align: center;**
- ▶ **text-decoration**  
Ajoute des lignes dessous, dessus et en travers du texte (none, underline, overline, line-through) peut contenir plusieurs valeurs séparées par des espaces  
**text-decoration: underline overline;**
- ▶ **text-transform**  
Modifier la casse du texte (none, capitalize, uppercase, lowercase)  
**text-transform: uppercase;**
- ▶ **text-shadow**  
Ombre (x, y, blur, color) peut contenir plusieurs valeurs séparées par des virgules  
**text-shadow: 1px 1px 2px pink;**

# Texte - Web Fonts



- Possibilité de charger des polices dans le navigateur pour une page web sans les installer sur le système
- Attention aux Licences et au temps de chargement
- Où trouver des polices ?
  - Google Font Directory : polices libres de droit  
<https://fonts.google.com>
  - FontSquirrel : Polices + pour préparer ses polices  
<http://www.fontsquirrel.com/>
  - Adobe Fonts : Polices professionnelles avec licences Web  
<https://fonts.adobe.com>
  - En les installant via npm <https://github.com/fontsource/fontsource>



# Texte - Web Fonts

- ▶ Une police se charge à l'aide de la règle @font-face

```
@font-face {  
    font-family: 'CalligraffitiRegular';  
    src: url('Calligraffiti.eot');  
    src: url('Calligraffiti.eot?#iefix') format('embedded-opentype'),  
        url('Calligraffiti.woff') format('woff'),  
        url('Calligraffiti.ttf') format('truetype'),  
        url('Calligraffiti.svg#CalligraffitiRegular') format('svg');  
    font-weight: normal;  
    font-style: normal;  
}  
  
body {  
    font-family: CalligraffitiRegular;  
}
```

- ▶ Il est possible de les précharger

```
<link rel="preload" href="fonts/cicle_fina-webfont.woff" as="font" type="font/woff"  
crossorigin="anonymous">
```



**formation.tech**

# Transformations



# Transformations

- transform-origin: centre des transformations précisées dans la propriété transform.
- transform: transformation1 ... transformation(n);
- les transformations sont séparées par un espace.
- Elles sont réalisées de la droite vers la gauche.
- Une matrice de transformation est calculée puis appliquée.



# Transformations

- `translate(tx, ty)`, `translateX(tx)`, `translateY(ty)`
- `rotate(angle)` : unité deg, rad ou grad
- `scale(facteur)`, `scale(facteurX, facteurY)`, `scaleX(fx)`, `scaleY(fy)`
- `skew(sx, sy)`, `skewX(sx)`, `skewY(sy)`
- `matrix(m11, m12, m21, m22, tx, ty)` : matrice de transformation 2x2 + translation(tx, ty)



# Transformations

- `translate3d(tx, ty, tz)`, `translateX(tx)`, `translateY(ty)`, `translateZ(tz)`
- `rotate3d(ax, ay, az)`, `rotateX(ax)`, `rotateY(ay)`, `rotateZ(az)`
- `scale3d(fx, fy, fz)`, `scaleX(fx)`, `scaleY(fy)`, `scaleZ(fz)`
- `matrix3d(m11, ..., m44)` : matrice de transformation 3D 4x4
- `perspective(profondeur)` : distance du plan z=0 par rapport à l'observateur.



**formation.tech**

# Transitions



# Transitions

- Permet d'animer presque tous les changements de valeur des propriétés CSS.
- Etapes :
  - 1. on place la valeur de départ de la propriété + la transition
  - 2. on change la valeur de la propriété à sa valeur d'arrivée.
  - 3. le navigateur applique automatiquement la(les) transition(s)



# Transitions

- transition: propriété durée easing délai, ...
- property: none, all, nom d'une propriété
- duration: 0 = pas d'animation, unité : s (secondes) ou ms (millisecondes)
- delay: 0 ou now (au moment du changement de propriété), unité s ou ms.
- timing-function : courbe de timing de l'animation



# Transitions

- ▶ Basées sur des courbes de bázier :
  - `cubic-bezier(point de contrôle 1, point de contrôle 2)`
  - `linear = cubic-bezier(0,0, 1,1)`
  - `ease = cubic-bezier(0.25, 1, 0.25, 1)` - valeur par défaut
  - `ease-in = cubic-bezier(0.42, 0, 1, 1)`
  - `ease-out = cubic-bezier(0, 0, 0.58, 1)`
  - `ease-in-out = cubic-bezier(0.42, 0, 0.58, 1)`
- ▶ <https://cubic-bezier.com/>
- ▶



**formation.tech**

# Animations



# Animations

- Définir des animations plus complexes avec des étapes
- On peut animer les mêmes propriétés que pour les transitions.
- Utilisation d'une directive @keyframes
- Une série de propriétés :
- animation-name, animation-duration, animation-iteration-count, animation-delay, animation-direction, animation-timing-function



# Animations

```
div {  
  width: 100px;  
  height: 100px;  
  background-color: red;  
  animation-name: pulse;  
  animation-delay: 2s;  
  animation-duration: 1s;  
  animation-iteration-count: 10;  
  animation-direction: alternate;  
}  
  
@keyframes pulse {  
  from {  
    transform: scale(1);  
  }  
  30% {  
    transform: scale(0.7);  
  }  
  70% {  
    transform: scale(1.3);  
  }  
  to {  
    transform: scale(1);  
  }  
}
```



**formation.tech**

# Préprocesseurs CSS

# Préprocesseurs CSS



- Les préprocesseurs CSS sont des technologies qui permettent à des langages proches de CSS de transpiler en CSS en y ajoutant des fonctionnalités
- Comparateurs de préprocesseurs CSS  
<http://csspre.com/compare/>



# Préprocesseurs CSS

- Apparu en 2009, inspiré par SASS  
<http://lesscss.org/>
- Variables



```
@link-color: #428bca; // sea blue

a, .link {
  color: @link-color;
}
.widget {
  color: #fff;
  background: @link-color;
}
```

- Héritage

```
nav ul {
  &:extend(.inline);
  background: blue;
}
```



# Préprocesseurs CSS

## ► Mixins (fonctions)

```
.border-radius(@radius) {  
    -webkit-border-radius: @radius;  
    -moz-border-radius: @radius;  
    border-radius: @radius;  
}  
  
.header {  
    .border-radius(4px);  
}  
.button {  
    .border-radius(6px);  
}
```

## ► Imports

```
.foo {  
    background: #900;  
}  
@import "this-is-valid.less";
```



# Préprocesseurs CSS

## ▶ Imbriquer

```
#header {  
    color: black;  
    .navigation {  
        font-size: 12px;  
    }  
    .logo {  
        width: 300px;  
    }  
}
```

# Préprocesseurs CSS



## ▶ Transpiler

- Côté client :

```
<link rel="stylesheet/less" type="text/css" href="styles.less" />
<script src="less.js" type="text/JavaScript"></script>
```

- En ligne de commande
  - Installer

```
npm install -g less
```

- Transpiler

```
lessc styles.less > styles.css
```

# Préprocesseurs CSS



Sass

- Apparu en 2007  
<http://sass-lang.com>
- A peu de choses près fonctionnalités égales à Less
- 2 syntaxes :
  - SASS

```
$primary-color: #333

body
  color: $primary-color
```

- SCSS (inspirée de Less)

```
$primary-color: #333;

body {
  color: $primary-color;
}
```



# Préprocesseurs CSS

- Apparu en 2010, inspiré par SASS et LESS  
<http://learnboost.github.io/stylus/>
- Syntaxe encore plus concise

```
body
  font 12px Helvetica, Arial, sans-serif

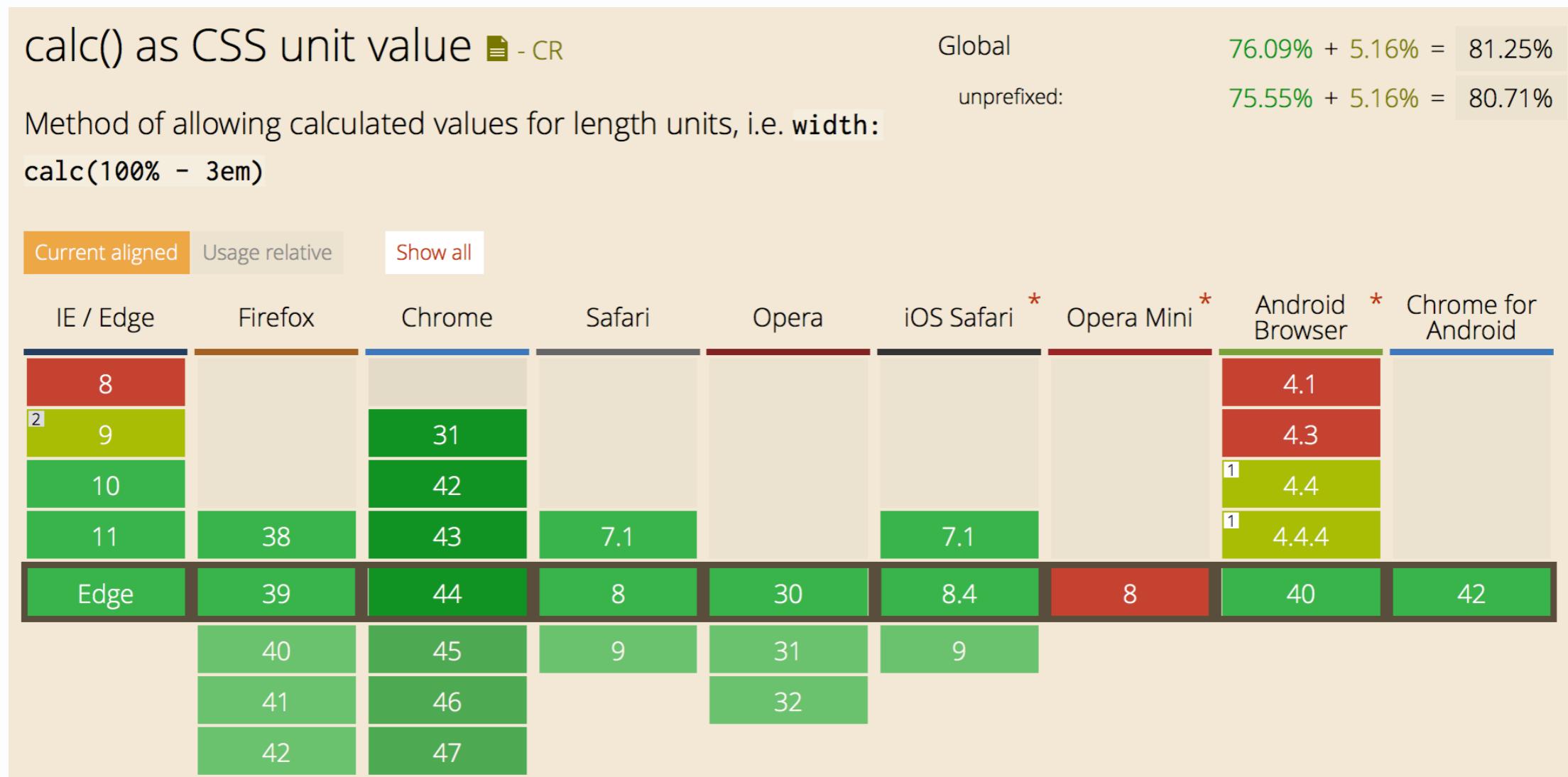
a.button
  -webkit-border-radius 5px
  -moz-border-radius 5px
  border-radius 5px
```

The Stylus logo features the word "stylus" in a lowercase, sans-serif font. The letter "s" is dark grey, while the letters "t", "y", "l", "u", and "s" are a vibrant lime green color. A thick, curved lime green line starts from the bottom left, goes up and over the "s", then down and under the "t", then back up and over the "y", "l", and "u", finally ending with a small loop at the top right.



# Préprocesseurs CSS

- Less et Sass ont inspiré le W3C  
Des normes sur les variables, opérateurs existe désormais nativement mais sont mal supportées





# Préprocesseurs CSS

- Apparu fin 2013  
<http://www.myth.io>
- Variables

```
:root {  
--purple: #847AD1;  
--large: 10px;  
}  
  
a {  
color: var(--purple);  
}  
  
pre {  
padding: var(--large);  
}
```

- Opérateurs

```
pre {  
margin: calc(var(--large) * 2);  
}
```



# Préprocesseurs CSS

- Apparu mi-2014  
<http://cssnext.io>
- Supporte plus de nouveautés CSS que Myth
  - automatic vendor prefixes, custom properties & `var()`, reduced `calc()`, custom media queries, media queries ranges, custom selectors, `color()`, `hwb()`, `gray()`, #rrggbbaa, `rebeccapurple`, font-variant, filter, `rem` units, `:any-link` pseudo-class, `:matches` pseudo-class, `:not` pseudo-class, pseudo-elements, Alpha colors, Bonus features, `@import`, minification, @todo

**cssnext {|||}**



**formation.tech**

# BEM



- ▶ BEM : Block Element Modifier
- ▶ Approche orientée composant
- ▶ Méthodologie créée par Yandex en 2010
- ▶ Documentation  
<https://en.bem.info>

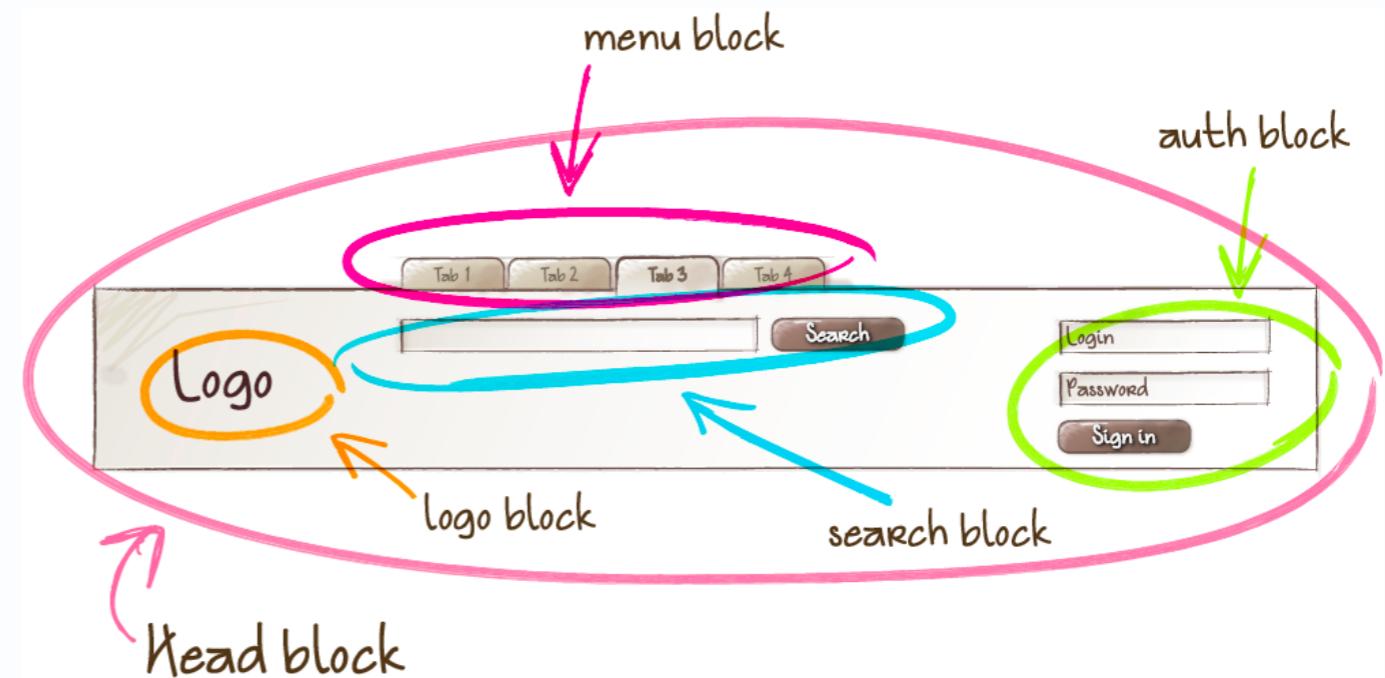


## ▶ Block

Un block est un composant réutilisable sur un site web.

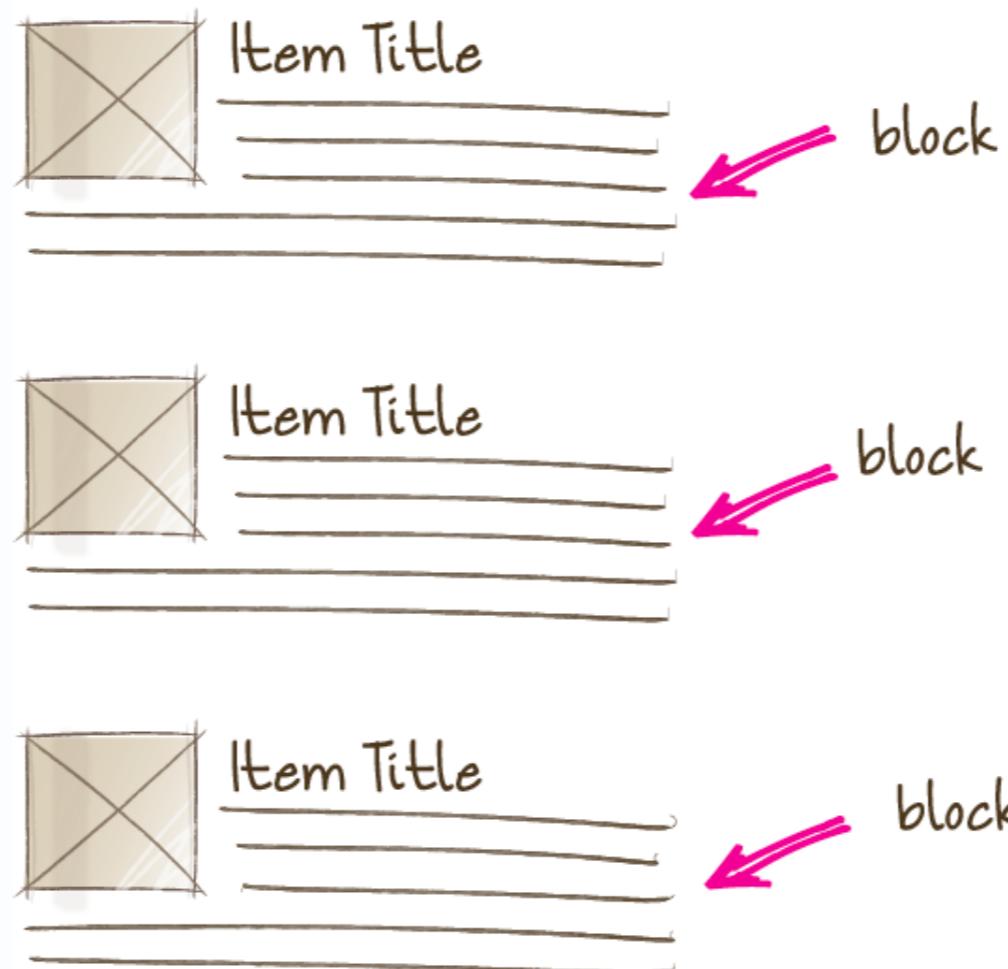
## ▶ Exemple :

- Menu
- Bouton
- Barre de recherche
- Formulaire de login
- Footer





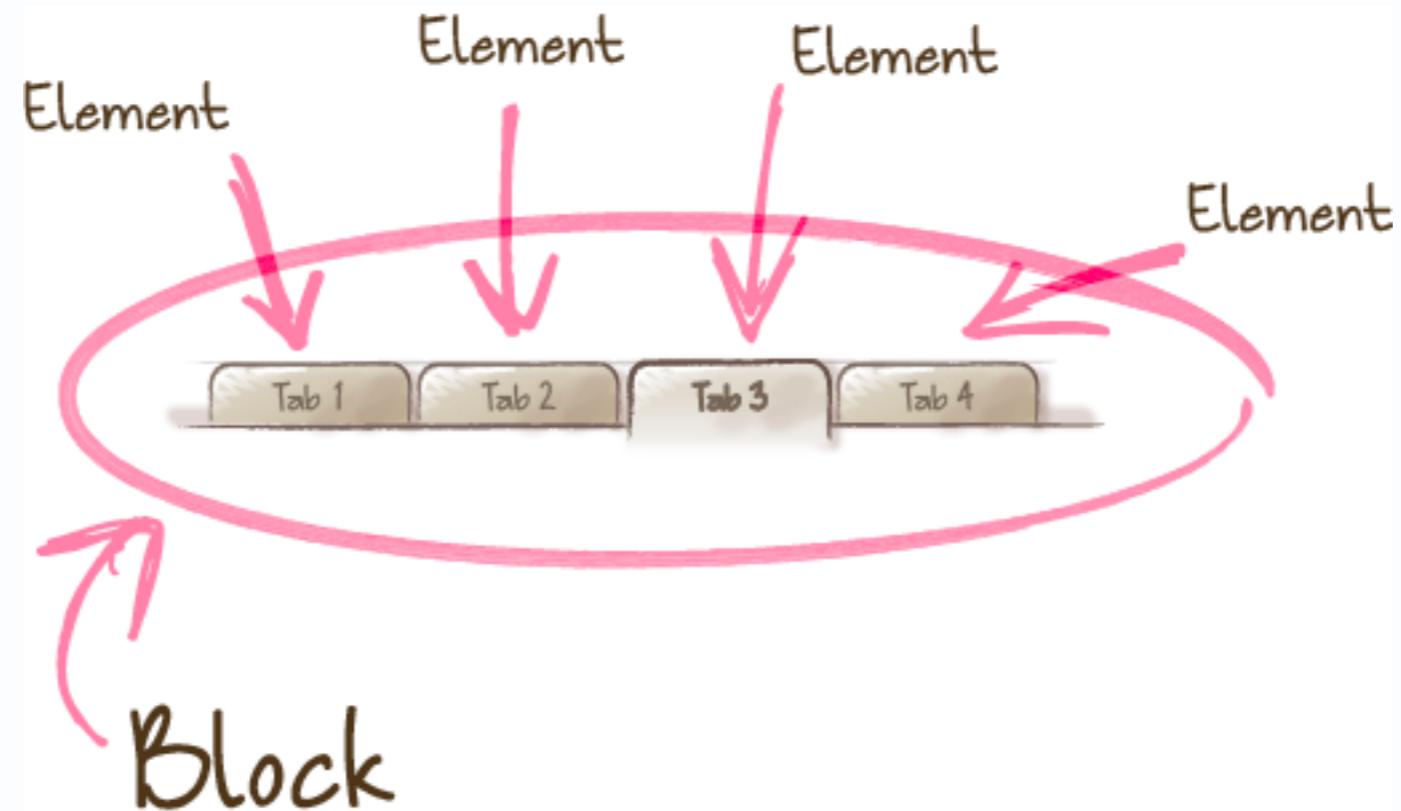
- ▶ Une page doit pouvoir contenir des instances multiples d'un bloc





## ▶ Element

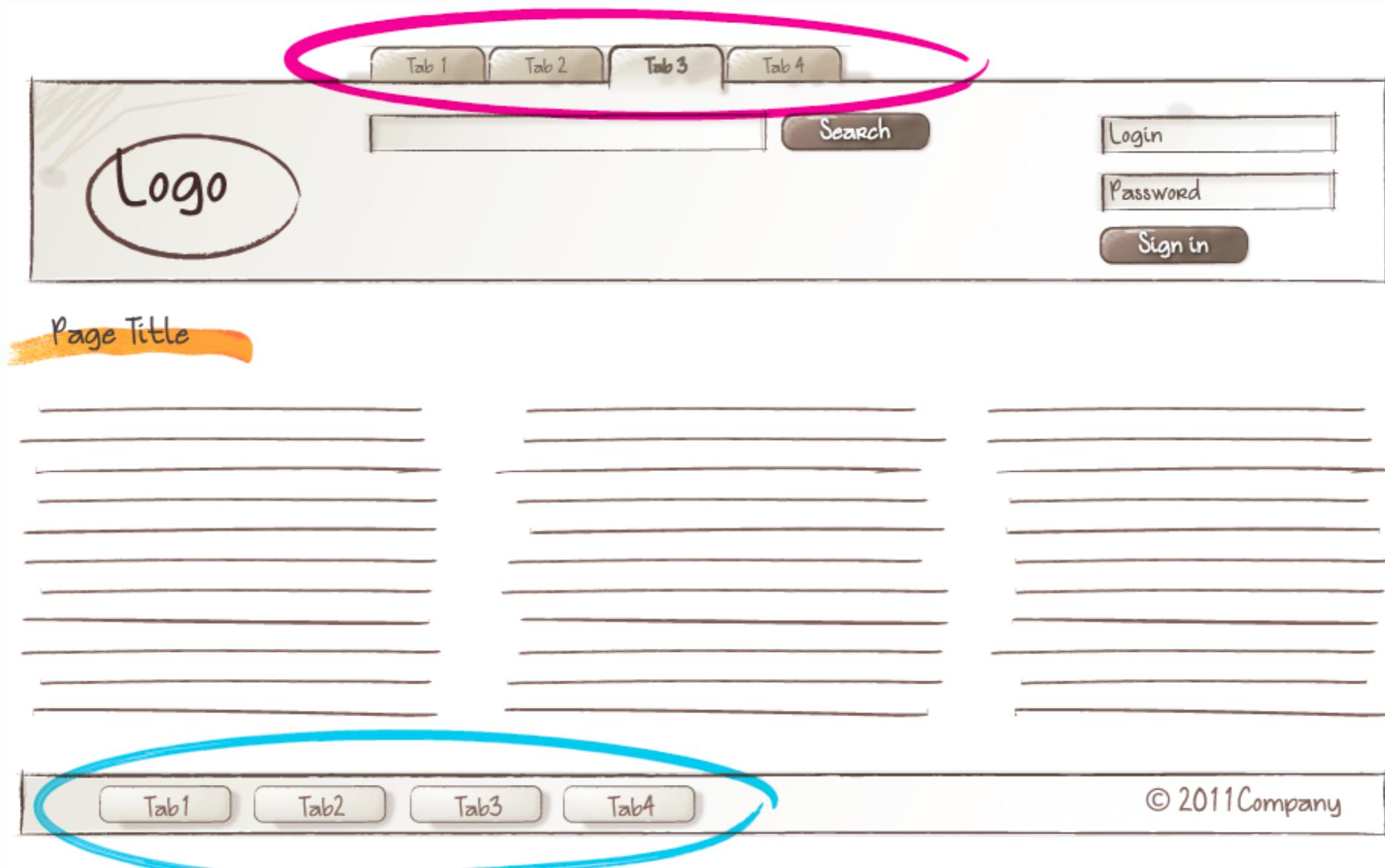
Un élément est un composant d'un block qui n'a pas pour vocation à être utilisé en dehors de celui-ci.





## ▶ Modifier

Un modifier permet de modifier l'apparence ou le comportement d'un bloc ou d'un élément.





## ▶ Conventions de nom

L'approche BEM suggère les conventions de nom suivante.

- ▶ Les noms des entités BEM (blocks, elements, modifiers) utilisent des chiffres et lettres non accentuées.
  - ▶ Les mots sont séparés par des traits d'union (-).
  - ▶ Les noms utilisent des classes CSS
- ## ▶ Pour les Entités :
- ▶ Blocks (parfois CamelCase), exemples : menu, form-login
  - ▶ Elements, reprend le nom du block suivi d'un double underscore (\_\_), exemple : menu\_\_item, form-login\_\_input-email
  - ▶ Modifiers, reprend le nom du block suivi d'un simple underscore (\_) (parfois double trait d'union (-) ), exemple : menu\_active, selecteur-langue\_theme\_windows



## ▶ Organisation des fichiers

```
blocks/
input/
  _type/  # type modifier directory
  input_type_search.css # Implementation of modifier type
    # with value search in CSS technology
  __box/  # box element directory
  input_box.css
  __clear/ # clear element directory
  __visible/ # visible modifier directory
  input_clear_visible.css # Implementation of boolean modifier visible
    # with value true in CSS technology
  __size/ # size modifier directory
  input_clear_size_large.css# Implementation of modifier size
    # with value large in CSS technology
  input_clear.css
  input_clear.js
    input.css
    input.js
button/
  button.css
  button.js
  button.png
```

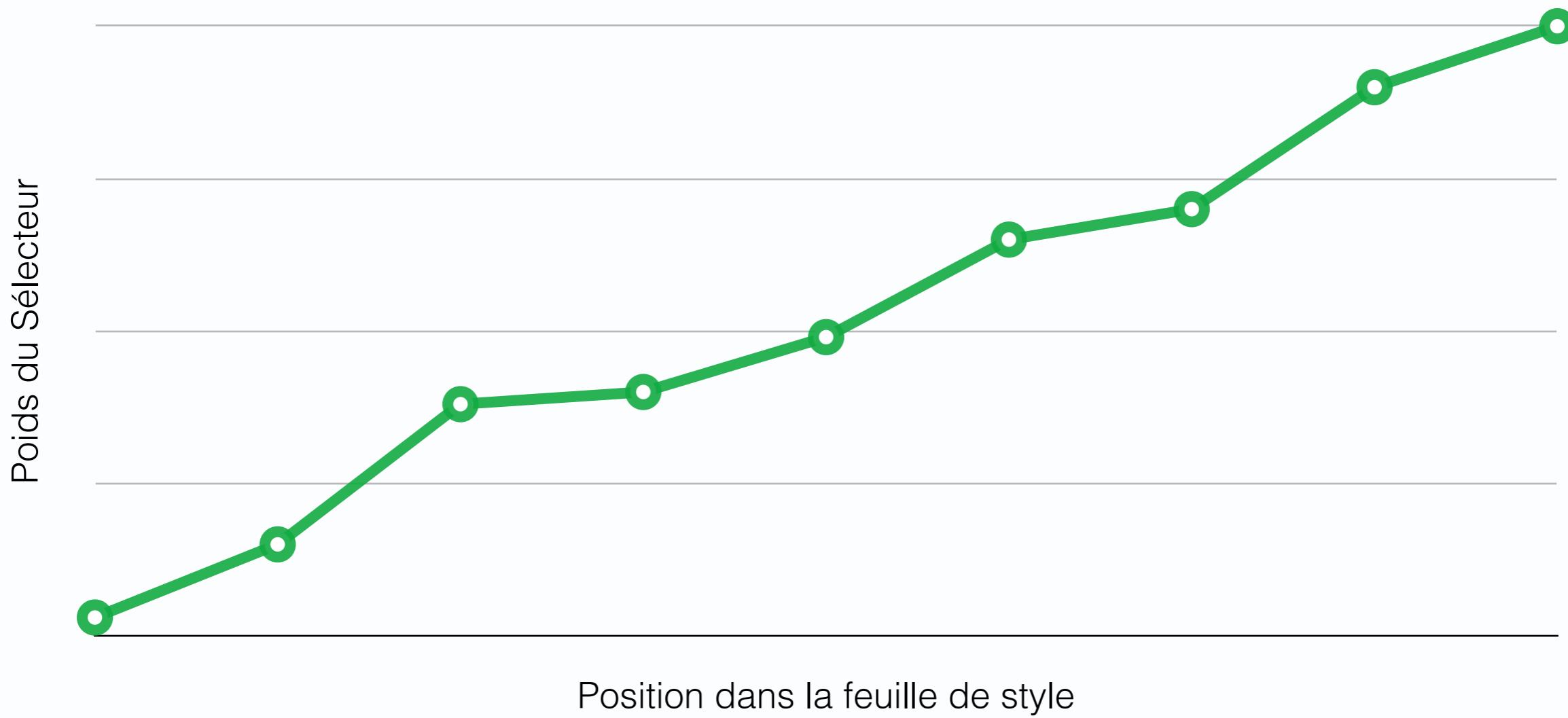


**formation.tech**

# ITCSS

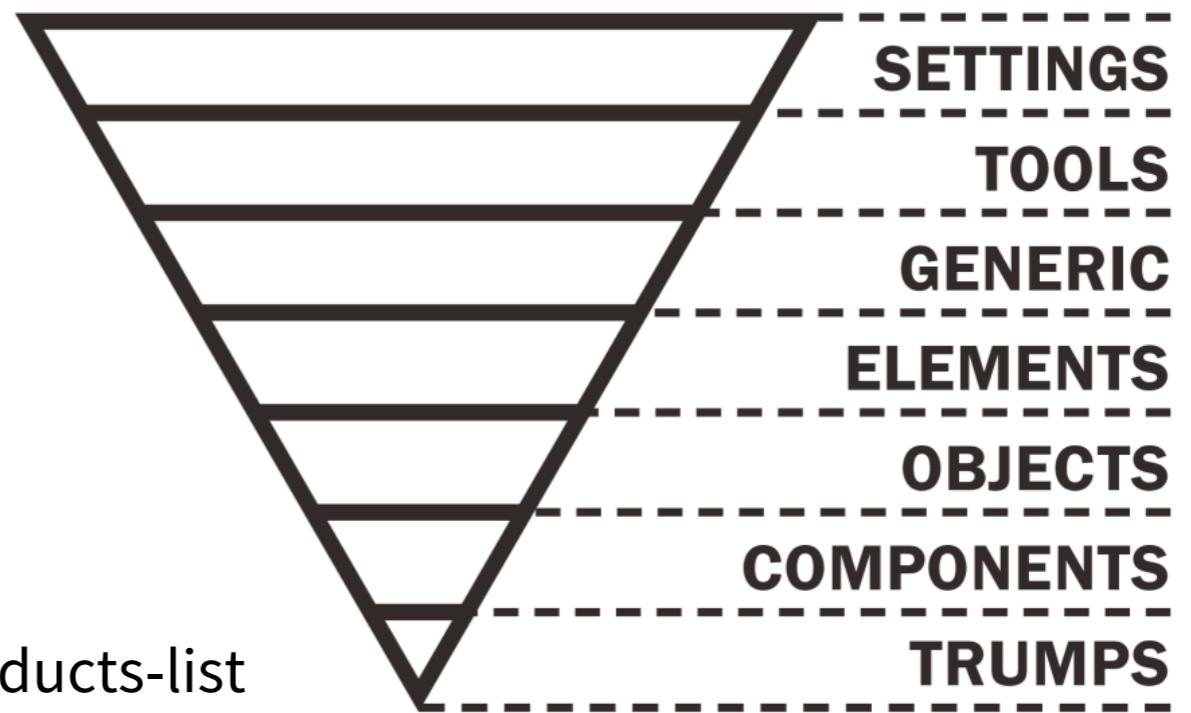


- ▶ Organisation du CSS par couche
- ▶ Ecriture de la priorité de sélecteur la plus faible à plus forte





- Settings : variables
- Tools : mixins
- Generic : Resets, Normalize.css
- Elements : style de base, h1-h6, body, lists
- Objects : composants génériques .ui-list
- Components : composants spécialisés .products-list
- Trumps : écrasement de propriétés, !important



Certaines couches peuvent être ajoutées  
(themes...) ou retirés (pas de  
Préprocesseur = pas de settings ou tools)



```
@import "settings.colors";
@import "settings.global";

@import "tools.mixins";

@import "normalize-scss/normalize.scss";
@import "generic.reset";
@import "generic.box-sizing";
@import "generic.shared";

@import "elements.headings";
@import "elements.forms";
@import "elements.links";
@import "elements.quotes";
@import "elements.tables";

@import "objects.animations";
@import "objects.drawer";
@import "objects.layout";
@import "objects.overlays";

@import "components.404";
@import "components.about";
@import "components.archive";
@import "components.avatars";
@import "components.blog-post";
@import "components.topbar";
@import "components.work-list";
@import "components.work-detail";

@import "trumps.clearfix";
```