

# CSS 3 avancé et Responsive Design

---

16/11/2023 – 17/11/2023

**Glodie Tshimini**



# Glodie Tshimini

Consultant Formateur et développeur depuis 2017

Certifié 2AICONCEPT Expert Trainer

Certifié 2AICONCEPT IT Expert Trainer At POE

Email : [contact@tshimini.fr](mailto:contact@tshimini.fr)

# Horaires et pauses

## Jeudi

- 9h00 - 12h30
- 13h30 - 17h30

## Vendredi

- 09h00 - 12h30
- 13h30 - 16h30

## Les pauses

- Matin : 15 minutes
- Déjeuner : 1 heure
- Après-midi : 15 minutes

# Émargement et évaluation formateur

Chaque matin et chaque après-midi, vous devrez signer les feuilles d’émargement. Les “X” et initiales ne sont pas autorisés, merci de bien vouloir émarger avec une signature.

Le dernier jour de la formation, avant 14h00, vous aurez à saisir une évaluation de la formation. Cette évaluation est obligatoire, il doit être rempli consciencieusement.

# Objectifs pédagogiques

- Concevoir des layouts de page en CSS
- Structurer efficacement vos CSS
- Définir le Responsive Design
- Utiliser les nouveautés du CSS 3

# Niveau requis et public concerné

## Niveau requis

- Avoir des connaissances de base HTML / CSS
- Avoir suivi la formation HTM-FND (HTML 5, CSS 3, Responsive – Création de pages Web) ou posséder les connaissances équivalentes

## Public concerné

- Webmasters
- Développeurs
- Concepteurs Web

# Déroulement de la formation

- 30 % théorie 70% pratique
- Daily meeting
- Exercices immédiats d'application du cours
- Les énoncés des exercices, TP et corrections seront accessibles depuis un [dépôt public GitHub](#)

# GitHub de la formation



<https://github.com/glo10/css-adavance-16112023>



# Un formateur à votre écoute



Google Forms

<https://docs.google.com/forms/d/e/1FAIpQLSdl7puiBiBdNPKCJR4C8EuY-EIWIg7qvtCkvGoRqMwv2b8-yA/viewform>



# Présentations

**Merci de bien vouloir vous présenter individuellement.**

Merci de bien vouloir m'indiquer quelles sont vos attentes suite à cette formation.

# PLAN DU COURS

---



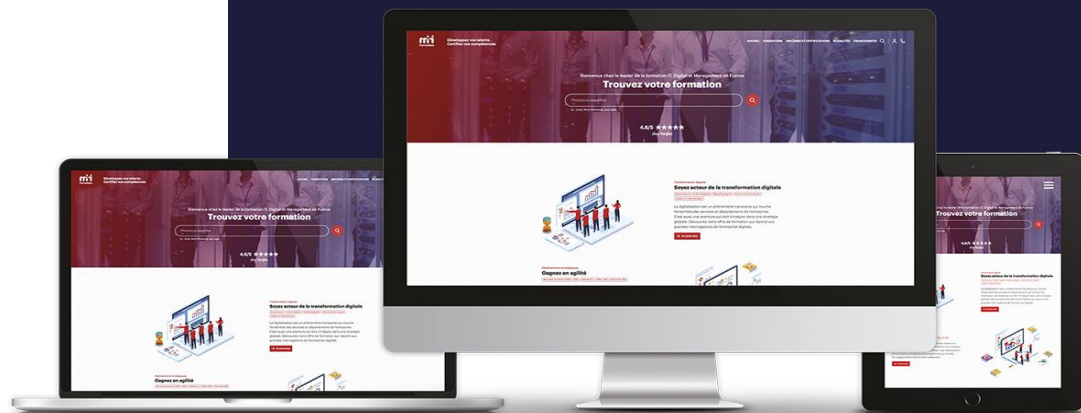
# Plan

- I. Rappels
- II. Box Model
- III. FlexBox
- IV. CSS Grid
- V. Positionnement (flottement)
- VI. Mobile First et Responsive Design
- VII. Animations et Transitions



# I. RAPPELS

---



m2iinformation.fr

# HISTORIQUE

1996

- Naissance du **CSS1**

2001

- **CSS 2.1**

1998

- **CSS2**

Version  
actuelle  
**CSS3**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="main.css">
  <title>Document</title>
  <style>
    p {
      color: orange;
    }
  </style>
</head>
<body>
  <h1 style="color: red; font-weight:bold;">Title</h1>
  <p>Paragraph</p>
</body>
</html>
```

- Le *CSS* permet de gérer
  - Couleurs
  - Polices
  - Mise en page
  - Etc
- Est Complémentaire du *HTML*
- Comment l'utiliser avec le *HTML* ?
  1. Soit dans un **fichier externe** *filename.css* avec une liaison dans le document HTML au niveau du *<head></head>* grâce à la balise link.
 

```
<link rel="stylesheet" href="main.css"/>
```
  2. Ou directement dans le document HTML avec l'utilisation de la balise *<style></style>*.
  3. Ou directement au sein d'une balise HTML avec l'utilisation de l'attribut *style*.

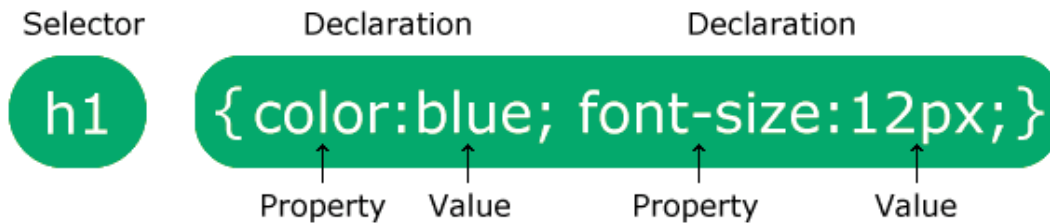
## Source image MDN

```
@import url("fineprint.css") print;  
@import url("bluish.css") speech;  
@import "custom.css";  
@import url("chrome://communicator/skin/");  
@import "common.css" screen;  
@import url("landscape.css") screen and (orientation: landscape);
```

- Permet d'importer
  - D'autres feuilles de style (fichiers CSS) dans un fichier CSS
- Doit être en en-tête du document CSS
- A combiner éventuellement avec les *media queries* que l'on verra un peu plus tard pour spécifier l'import du fichier uniquement lorsque la règle sur les *media queries* est précisé ou le type d'écran



## Source image W3Schools



- Sélecteur
  - Élément *HTML* ciblé
- Propriété
  - Élément de style ciblé
- Valeur
  - Valeur à appliquer à l'élément de style ciblé
- /\*Commentaires en CSS\*/

# Attribut viewport dans les métadonnées du HTML

## Source image MDN

### Un viewport de base

Un site type, optimisé pour les mobiles, contient quelque chose comme ce qui suit :

HTML



```
<meta name="viewport" content="width=device-width, initial-scale=1" />
```

- « Le *viewport* est la zone de la fenêtre dans laquelle le contenu d'une page web peut être vu »
- Cette zone varie selon les appareils (pc, tablette, mobile, tv, etc.)
- Dans le *head* du HTML avec la balise *meta*, on peut spécifier le comportement du *viewport* à travers des attributs et valeurs à définir pour contrôler la taille et l'échelle du zoom.
- Pour voir toutes les possibilités des valeurs possibles, consulter cette [documentation](#)

# LES SELECTEURS

---





# JEU CSS DINER

Réalisez le jeu [CSS DINER](#)

# Sélecteurs simples

SÉLECTEURS	EXEMPLE(S)
Balise HTML	<i>p { color : red; } h1 { text-align : center; }</i>
À partir de l'attribut class	<i>.p-r { padding-right : 2em; } .bold { font-weight : bold; }</i>
À partir de l'attribut identifiant (id)	<i>#id { border : 1 px solid black; } #product-5 { text-align : justify; }</i>
Universel	<i>* { margin : 10px auto ; }</i>

## Documentation

- [W3schools sur les sélecteurs](#)
- [Test w3schools sur les sélecteurs](#)

SÉLECTEURS	EXEMPLE(S)
Sélection des enfants directs	<i>div &gt; p { background-color : orange ; }</i>
Sélecteur des enfants à n'importe quel niveau	<i>div p { display : none ; }</i> <i>div span { color : green ; }</i>
Voisin direct	<i>div + a { cursor : pointer ; }</i>

# Sélecteurs : pseudo-classes

- Ajouter un style spécifique en fonction de l'état dans lequel se trouve l'élément ciblé ou en tenant compte des autres éléments présents dans le document HTML
- [Documentation MDN sur les pseudo-classes](#)

PSEUDO-CLASSE	
:hover	État de survol de l'élément
:active	État lors du clique sur l'élément
:visited	État après avoir cliqué sur l'élément
:checked	État lorsque l'élément ciblé est coché
:only-child	Enfant unique d'un élément parent

# Sélecteurs : pseudo-elements

Styliser une partie de l'élément sélectionné



PSEUDO-ELEMENT		
::first-line		Styliser la première lettre de l'élément ciblé.
::before		Ajouter du contenu avant l'élément cible à l'aide la propriété <i>content</i> et de sa valeur (première enfant de la cible)
::after		Ajouter du contenu après l'élément cible à l'aide la propriété <i>content</i> et de sa valeur (dernière enfant de la cible)
PROPRIÉTÉ	VALEUR	
content	«Contenu entre guillemets »	Associé aux pseudo-elements ::before et ::after dont la valeur est générée par le CSS et affichée par le navigateur



# Cheat Sheet Sélecteurs

## All css selectors

CSS {selectors: cheat-sheet}

Name	CSS	Description	Results
<b>Basic</b>			
<b>Universal Selector</b>	<code>*</code>	Select all elements	<code>a</code> <code>b</code> <code>c</code> <code>d</code>
<b>Type Selector</b>	<code>div</code>	Select elements of that type Select div elements	<code>a</code> <code>div</code> <code>c</code> <code>div</code>
<b>Class Selector</b>	<code>.c</code>	Select elements with that class Select elements with the c class	<code>a</code> <code>b</code> <code>c</code> <code>d</code>
<b>Id Selector</b>	<code>#i</code>	Select elements with that id Select elements with the id *It is best practice to not use ids in CSS	<code>#a</code> <code>#b</code> <code>#i</code> <code>#d</code>
<b>Combination</b>			
<b>Descendant Selector</b>	<code>div a</code>	Select elements that are descendants of the first element Select anchors that are inside a div	
<b>Direct Child Selector</b>	<code>div &gt; a</code>	Select elements that are direct children of the first element Select anchors that are direct children of a div	
<b>General Sibling Selector</b>	<code>div ~ a</code>	Select elements that are siblings of the first element and come after the first element Select all anchors that are siblings of a div and come after the div	<code>a</code> <code>div</code> <code>a</code> <code>a</code>
<b>Adjacent Sibling Selector</b>	<code>div + a</code>	Select elements that are siblings of the first element and come directly after the first element Select all anchors that are siblings of a div and come directly after the div	<code>a</code> <code>div</code> <code>a</code> <code>a</code>
<b>Or Selector</b>	<code>div, a</code>	Select elements that match any selector in the list Selects all anchors and all divs	<code>a</code> <code>div</code> <code>a</code> <code>b</code>
<b>And Selector</b>	<code>div.c</code>	Select elements that match all the selector combinations Selects all divs with the class c	<code>a</code> <code>div.c</code> <code>c</code> <code>div</code>

Name	CSS	Description	Results
<b>Attribute</b>			
<b>Has Attribute</b>	<code>[a]</code>	Select elements that have that attribute Select elements with the a attribute	<code>a</code> <code>a="1"</code> <code>c</code> <code>d</code>
<b>Exact Attribute</b>	<code>[a="1"]</code>	Select elements that have that attribute with exactly that value Select elements with the a attribute with a value of 1	<code>a</code> <code>a="1"</code> <code>c</code> <code>d</code>
<b>Begins With Attribute</b>	<code>[a^="1"]</code>	Select elements that have that attribute which start with that value Select elements with the a attribute with a value that starts with 1	<code>a="12"</code> <code>a="21"</code>
<b>Ends With Attribute</b>	<code>[a\$="1"]</code>	Select elements that have that attribute which end with that value Select elements with the a attribute with a value that ends with 1	<code>a="12"</code> <code>a="21"</code>
<b>Substring Attribute</b>	<code>[a*="1"]</code>	Select elements that have that attribute which contain that value anywhere Select elements with the a attribute with a value that contains a 1	<code>a="12"</code> <code>a="21"</code>
<b>Pseudo Element</b>			
<b>Before Selector</b>	<code>div:before</code>	Creates an empty element directly before the children of selected element	<code>div</code> <code>before</code> <code>c</code> <code>after</code>
<b>After Selector</b>	<code>div:after</code>	Creates an empty element directly after the children of selected element	<code>div</code> <code>before</code> <code>c</code> <code>after</code>
<b>Pseudo Class State</b>			
<b>Hover Selector</b>	<code>button:hover</code>	Select elements that are hovered by the mouse Select buttons that are being hovered	
<b>Focus Selector</b>	<code>button:focus</code>	Select elements that are focused. Select buttons that are being focused. *Focus is set by either tabbing to an element or clicking an element such as a button or anchor tag	
<b>Required Selector</b>	<code>input:required</code>	Select inputs that are required Select inputs with the required attribute	
<b>Checked Selector</b>	<code>input:checked</code>	Select checkboxes/radio buttons that are checked Select inputs that are checked	
<b>Disabled Selector</b>	<code>input:disabled</code>	Select inputs that are disabled Select inputs with the disabled attribute	

# Cheat Sheet Pseudo-Classes

Source image [dev.to/areedev](https://dev.to/areedev)

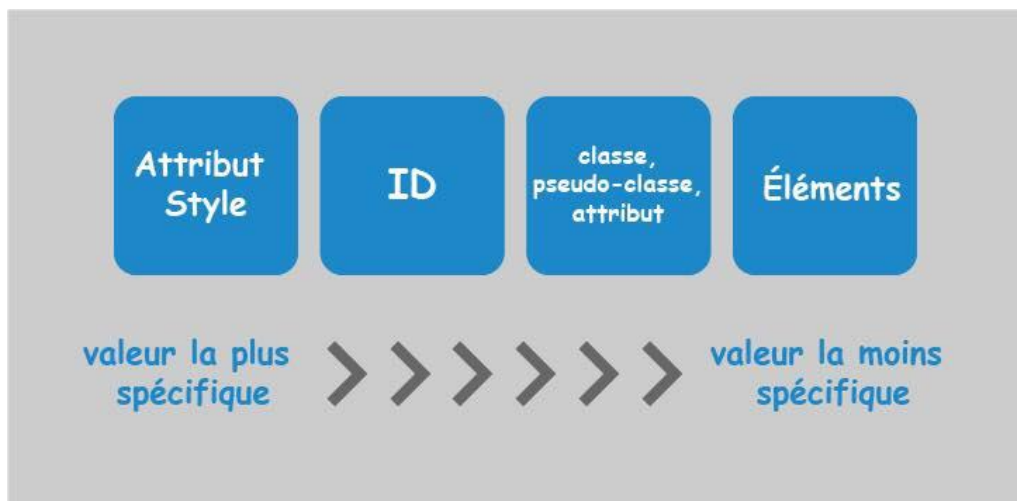
Pseudo Class Position/Other			
Name	CSS	Description	Results
First Child Selector	<code>a:first-child</code>	Select elements that are the first child inside a container Select anchors that are the first child	
Last Child Selector	<code>a:last-child</code>	Select elements that are the last child inside a container Select anchors that are the last child	
Nth Child Selector	<code>a:nth-child(2n)</code>	Select elements that are the nth child inside a container based on the formula Select anchors that are even numbered children	
Nth Last Child Selector	<code>a:nth-last-child(3)</code>	Select elements that are the nth child inside a container based on the formula counting from the end Select anchors that are the third to last child	
Only Child Selector	<code>a:only-child</code>	Select elements that are the only child inside a container Select anchors that are the only child	
First Of Type Selector	<code>a:first-of-type</code>	Select elements that are the first of a type inside a container Select the first anchor in a container	
Last Of Type Selector	<code>a:last-of-type</code>	Select elements that are the last of a type inside a container Select the last anchor in a container	
Nth Of Type Selector	<code>a:nth-of-type(2n)</code>	Select elements that are the nth of a type inside a container based on the formula Select every second anchor	
Nth Last Of Type Selector	<code>a:nth-last-of-type(2)</code>	Select elements that are the nth of a type inside a container based on the formula counting from the end Select the second to last anchor	
Only Of Type Selector	<code>a:only-of-type</code>	Select elements that are the only of a type inside a container Select anchors that are the only anchor in a container	
Not Selector	<code>a:not(.c)</code>	Select all elements that do not match the selector inside the not selector Select all anchor tags that do not have the 'c' class	

- Ajouter un style spécifique en fonction de l'état dans lequel se trouve l'élément ciblé ou en tenant compte des autres éléments présents dans le document HTML
- [Documentation MDN pseudo-classes](#)
- [Documentation W3Schools sélecteurs](#)
- [Test sélecteurs W3Schools](#)

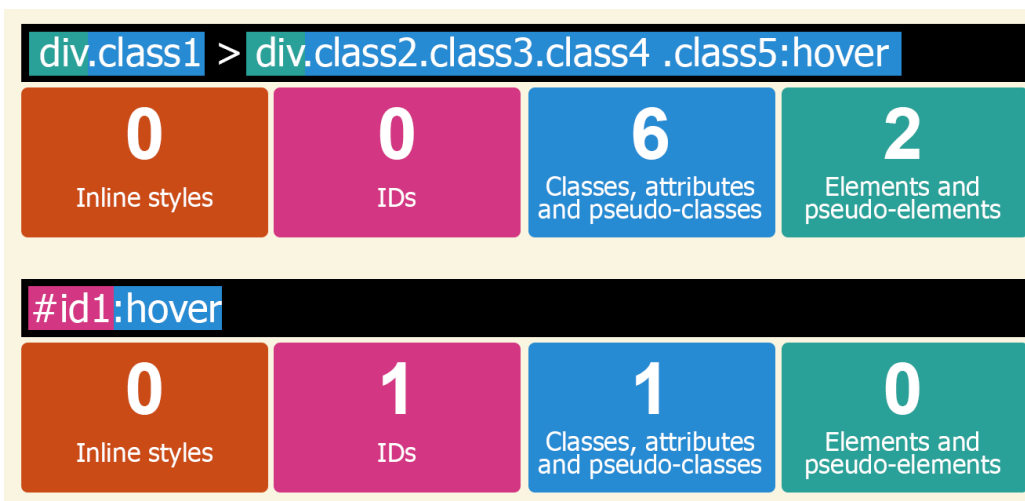
# SPÉCIFICITÉS (priorité)

**Source image gauche w3docs**

**Source image droite groupe Sii**



- Lorsque plusieurs règles (propriétés et valeurs) sont appliquées à un élément avec des sélecteurs différents qui ciblent ce même élément, le navigateur applique en premier les règles les plus spécifiques.



# LES UNITES

---



# NUMÉRIQUES ET ABSOLUES

Unité num. et absolue	
<i>cm</i>	Réservé aux feuilles de style dédiées à l'impression
<i>mm</i>	Réservé aux feuilles de style dédiées à l'impression
<i>in</i>	Réservé aux feuilles de style dédiées à l'impression
<i>px</i>	Unité de référence pour les écrans, un écran est défini par sa résolution en pixels

[Documentation des unités](#)

# NUMÉRIQUES ET RELATIVES

Unités num. et relatives	
%	Dimension est proportionnelle à la dimension du conteneur (l'élément parent), à utiliser principalement pour définir une largeur ( <i>width</i> )
<i>em</i>	Relative par rapport à la taille de la police de l'élément ou de l'élément parent
<i>rem</i>	Relative par rapport à la taille de la police de l'élément racine html ou body
<i>vh</i>	vh pour viewport height (hauteur visible) 1 vh = 1 % de la hauteur visible de la fenêtre
<i>vw</i>	vw pour viewport width (largeur visible) 15 vw = 15 % de la largeur visible de la fenêtre



# Ce qu'il faut retenir



SAVE

- Il y a une multitude de façons de sélectionner un élément HTML à partir du CSS, il faut privilégier les sélecteurs directs
- Les unités relatives sont à privilégier par rapport aux unités absolues, à part dans le cadre d'un media de type *print* pour l'impression d'un document de type A4, A3, etc

# Conseils du formateur

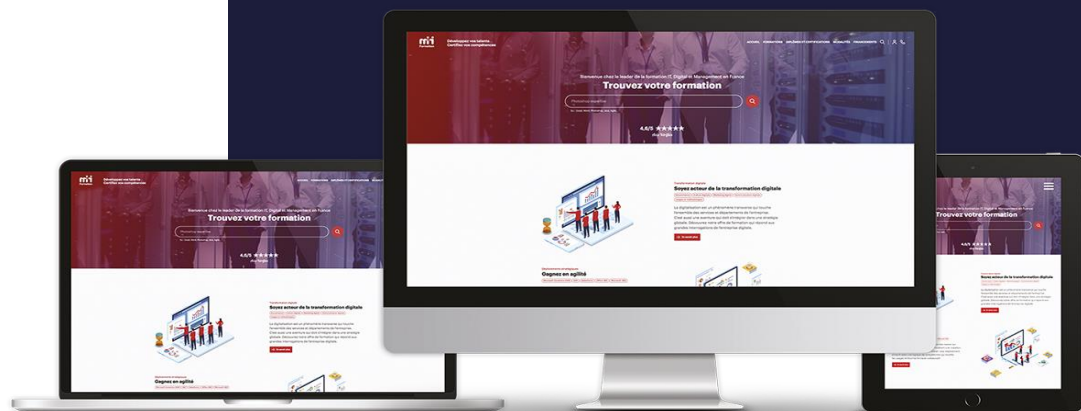


- Privilégiez les Id et classes, plus rapides et moins coûteux en ressource pour trouver les éléments concernés
- Séparez les CSS en découpant en plusieurs fichiers selon une certaine logique (thème, layout, module, etc.) et importez les css qui concernent une page ou une fonctionnalité dans un seul fichier. Encore mieux, si vous êtes à l'aise avec JavaScript, vous pouvez faire appel à des bundlers comme WebPack, Vite, Esbuild, etc. pour compiler un seul fichier qui contient tout



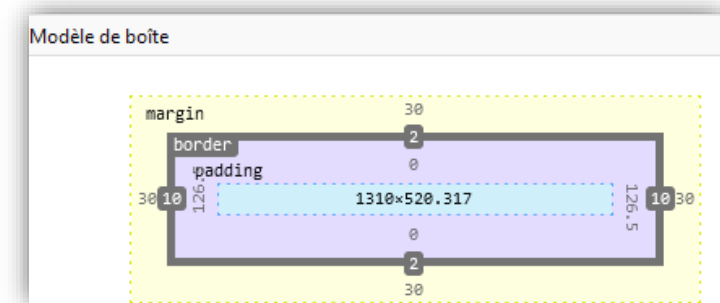
## II. BOX MODEL

---



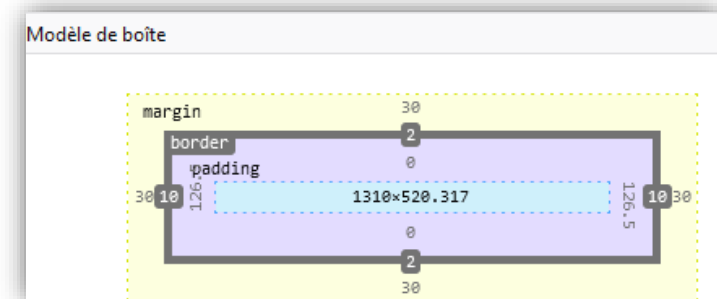
# BOX MODEL

- Block
  - Occupe toute la largeur disponible
  - Élément suivant se positionne en bas (retour à la ligne)
  - Propriétés *width* & *height* modifiables
  
- Inline
  - Occupe la place nécessaire qu'il lui faut
  - Élément suivant se positionne à la suite
  - Propriétés *width* et *height* non-modifiables
  
- Inline-block
  - Se comporte comme un *inline*
  - Propriétés *width* & *height* modifiables



# BOX SIZING

- Définis la façon dont la hauteur et la largeur totale d'un élément sont calculées à l'aide de la propriété *box-sizing*
- Valeur par défaut *content-box*
  - À la largeur (*width*) et à la hauteur (*height*) de base de l'élément, viennent s'ajouter les marges intérieures (*padding*), les marges extérieures (*margin*) et la bordure (*border*)
- Valeur *border-box*
  - Prends en compte le *padding*, *margin* et *border* dans le calcul du *width* et *height*
  - Valeurs du *width* et *height* incluent le *margin*, *padding* et *border*





# EXERCICE 1

0-exercices/ex1.md

# Ce qu'il faut retenir



SAVE

- Tous les éléments d'un document HTML sont représentés sous la forme d'une boîte qui est composée des dimensions (largeur et hauteur) de l'élément, des marges intérieurs (*padding*), des bordures et des marges extérieures (*margin*)
- La propriété ***box-sizing*** permet de définir la manière d'intégrer ou non les dimensions des marges et bordures dans le calcul de la taille finale de l'élément

# Conseils du formateur



- Écrasez les propriétés CSS ajoutées par les navigateurs pour homogénéiser vos mises en page sur différents navigateurs, pour cela, vous pouvez utiliser les fichiers [normalisés de css](#)
- Utilisez l'inspecteur d'élément pour voir les propriétés appliquées à l'élément et déterminer s'il s'agit de base d'un élément de type *block*, *inline-block* ou *inline*.



# III. FLEXBOX

---



m2iinformation.fr



# JEU FLEXBOX FROGGY

Réalisez le jeu [FLEXBOX FROGGY](#)



# SYNTHÈSE FLEXBOX

[Source image Samanthaming](#)

## Flexbox Cheat Sheet

@simonpaix

LearnPine

### Parent properties

**display:** enables flex context for all direct children.

```
.container {
  display: flex; // or inline-flex
}
```

**flex-direction:** sets the main-axis.

row  
row-reverse  
column  
column-reverse

**flex-wrap:** allows the items to wrap as needed.

no-wrap  
wrap  
wrap-reverse

**justify-content:** defines alignment along the main axis.

flex-start  
flex-end  
center  
space-between  
space-around  
space-evenly

**align-items:** defines alignment along the cross axis.

flex-start  
flex-end  
center  
stretch  
baseline

**align-content:** aligns multiple lines, like justify-content does with individual items.

flex-start  
flex-end  
center  
stretch

space-between  
space-around

### Children properties

**order:** changes the order of flex items.

```
.item {
  order: 3 // the default is 0
}
```

**flex-grow:** allows item to grow using remaining space.

```
.item-1 { flex-grow: 0; } //default
.item-2 { flex-grow: 1; }
```

**Tip:** If all items have flex-grow: 1, the remaining space is distributed equally.

**flex-shrink:** defines the ability for a flex item to shrink.

```
.one { flex-shrink: 1; }
.two { flex-shrink: 2; }
.three { flex-shrink: 3; }
.four { flex-shrink: 4; }
```

**Tip:** Defaults to 1. The highest the value the more it shrinks compared to siblings.

**flex-basis:** sets the default size of a flex item. It accepts:

- specific values : pixels, rm, %
- auto : defaults to width or height property
- content : automatic sizing, based on its content
- global values : inherit, initial, unset

**align-self:** overrides default alignment (or the one specified by align-items) for a specific item.

flex-start  
center  
flex-end  
baseline  
stretch

- Boites flexibles
- Un *conteneur* (l'élément parent)
  - Rendu flexible grâce à la déclaration *display : flex*
  - Les éléments vont se positionner de manière flexible à l'intérieur selon une direction
- 2 axes
  - *Axe principale* ou main axis
  - *Axe secondaire* ou cross axis ou axe transversal

# FLEXBOX CHEAT SHEET source image Mariana Simon

## Flexbox Cheat Sheet

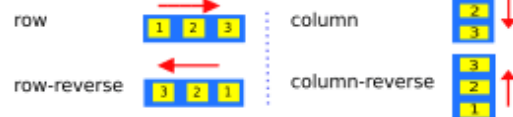
@simonpaix

### Parent properties

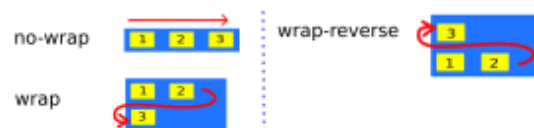
**display:** enables flex context for all direct children.

```
.container{
  display: flex; // or inline-flex
}
```

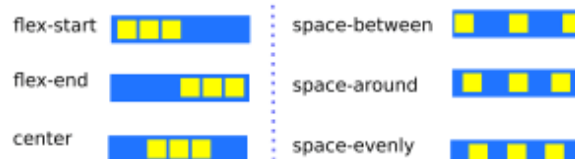
**flex-direction:** sets the main-axis.



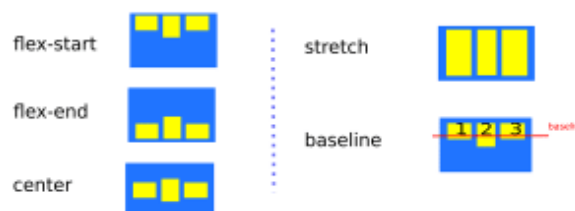
**flex-wrap:** allows the items to wrap as needed.



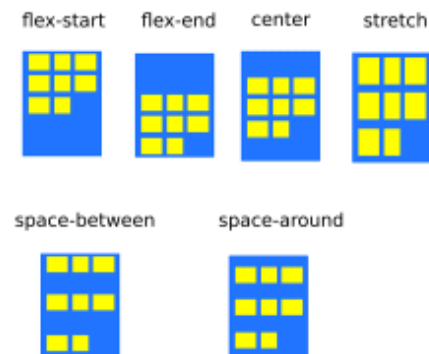
**justify-content:** defines alignment along the main axis.



**align-items:** defines alignment along the cross axis.



**align-content:** aligns multiple lines, like justify-content does with individual items.



### Children properties

**order:** changes the order of flex items.

```
.item {
  order: 3 // the default is 0
}
```



**flex-grow:** allows item to grow using remaining space.

```
.item-1 { flex-grow: 0; } //default
.item-1 { flex-grow: 1; }
```



**Tip:** If all items have flex-grow: 1, the remaining space is distributed equally.

**flex-shrink:** defines the ability for a flex item to shrink.

```
.one { flex-shrink: 1; }
.two { flex-shrink: 2; }
.three { flex-shrink: 3; }
.four { flex-shrink: 4; }
```

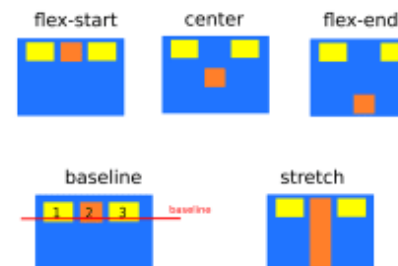


**Tip:** Defaults to 1. The highest the value the more it shrinks compared to siblings.

**flex-basis:** sets the default size of a flex item. It accepts:

- specific values : pixels, rm, %
- auto : defaults to width or height property
- content : automatic sizing, based on its content
- global values : inherit, initial, unset

**align-self:** overrides default alignment (or the one specified by align-items) for a specific item.





## EXERCICE 2

0-exercices/ex2.md

# Ce qu'il faut retenir



SAVE

- La mise en page avec les FlexBox est l'un des plus populaires aujourd'hui
- Les FlexBox permettent de faire des mises en page simple et facilement adaptable pour les différentes tailles d'écran

# Conseils du formateur



- Privilégiez les mises en page FlexBox à la place des mises en page avec les float, les display table, etc. qui ont servi par le passé à faire des mises en page



## IV. CSS GRID

---



m2iinformation.fr

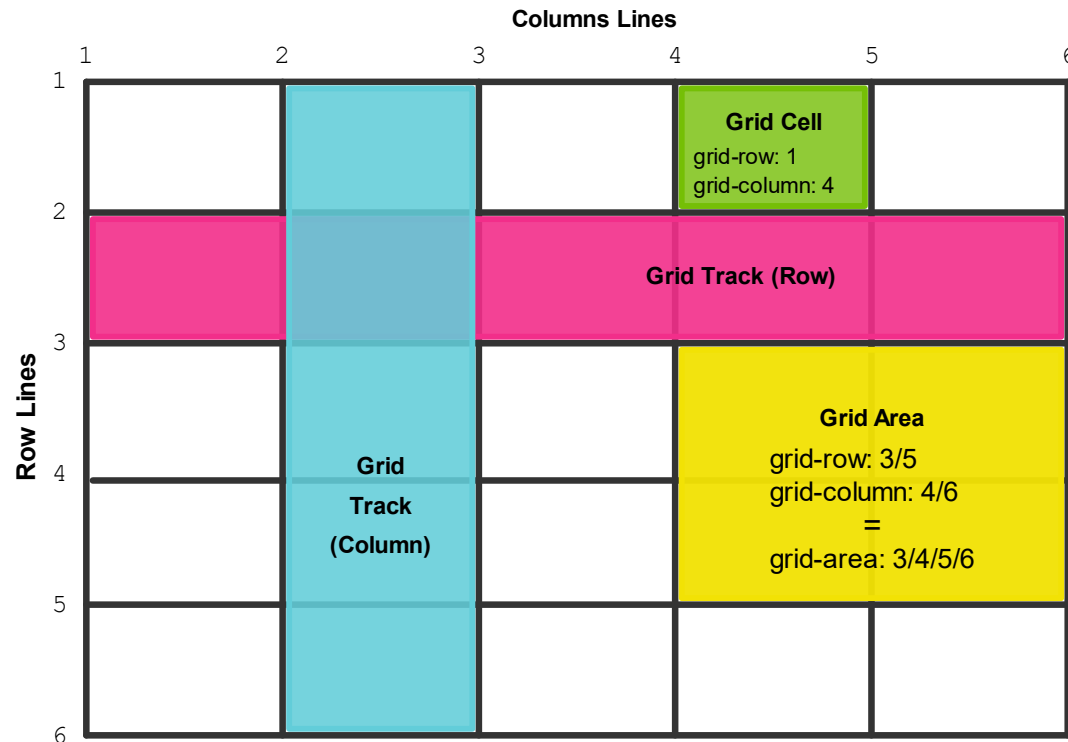


# JEU GRID GARDEN

Réalisez le jeu [GRID GARDEN](#)

# SYNTHÈSE CSS GRID

[Source image Kustavo](#)



- Grille
- Un *conteneur* (l'élément parent)
  - Transformer en grille grâce à la déclaration *display: grid*
  - Les éléments vont se positionner selon 2 dimensions (colonne et ligne).
- Unité *fr unit* (fraction unit) correspond à une fraction de l'espace disponible dans le conteneur de la grille ([Source Developer Mozilla](#))



# CSS GRID CHEAT SHEET source image Simon Paix

## Grid Cheat Sheet

@simonpaix

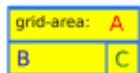
### Parent properties

**display:** enables grid context for all direct children.  
`.container{ display: grid; } // or inline-grid`

**grid-template:** defines the rows and columns of the grid. Set track size values and line-names(optional).

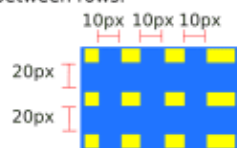
`grid-template-columns: 10px 30px auto 20%;`  
`grid-template-rows: repeat(3, 20px);`

`grid-template-areas:`  
`"A A A A"`  
`"B B B C"`



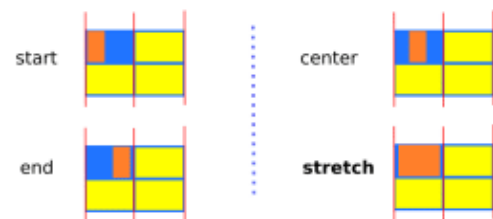
**grid-gap:** sets the size of the grid lines, the gutters between columns and between rows.

`column-gap: 10px;`  
`row-gap: 20px;`

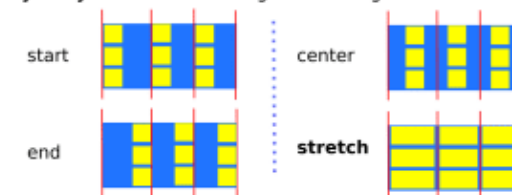


### Children properties

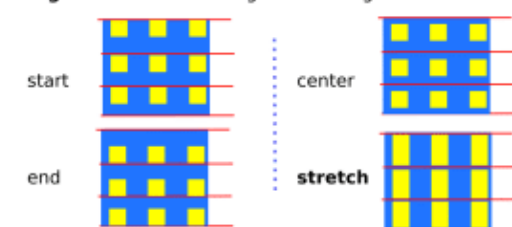
**justify-self:** aligns an item inside a single cell along the row axis.



**justify-items:** defines alignment along the row axis.



**align-items:** defines alignment along the column axis.



**grid-auto-flow:** defines how to automatically place grid items that aren't explicitly placed.



LearnPine ©

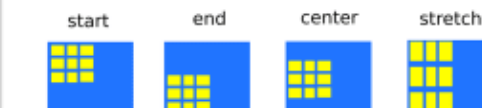
**justify-content:** justifies all grid content on row axis if total grid size is smaller than container.



**space-between** **space-around** **space-evenly**



**align-content:** justifies all grid content on column axis if total grid size is smaller than container.



**space-between** **space-around** **space-evenly**



LearnPine ©

**grid-column:** determines the item's location based on a start and an end column lines (or a span).

`grid-column-start: 2;`  
`grid-column-end: 4;`  
`// or grid-column: 2 / 4`



**grid-row:** same but for the row location.

`grid-row-start: 1;`  
`grid-row-end: 3;`  
`// or grid-row: 1 / span 2`



©



# EXERCICE 3

0-exercices/ex3.md

# Ce qu'il faut retenir



SAVE

- Le CSS Grid permet d'effectuer tout type de mise en page, simple à complexe
- Le CSS Grid est utilisé par les framework CSS tels que Bootstrap pour ne citer qu'eux parmi tant d'autres

# Conseils du formateur

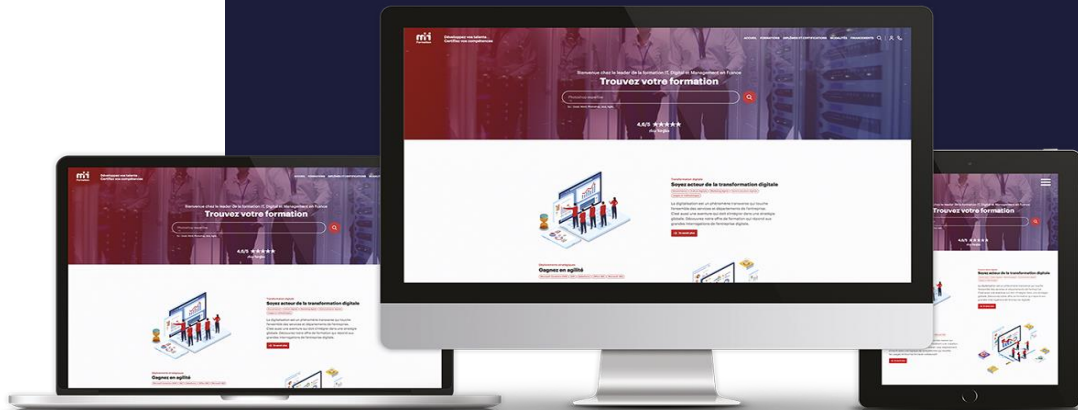


- Le CSS GRID permet d'intégrer plus facilement des maquettes graphiques, en effet, les graphistes raisonnent et découpent souvent une page en grille
- Comme le FlexBox, le CSS GRID facilite la mise en page adaptable sur plusieurs tailles d'écran
- Il y a plusieurs propriétés au niveau de CSS GRID qui font la même chose, choisissez celles avec lesquelles vous êtes le plus à l'aise. Pour le reste, vous pourrez toujours aller consulter la documentation si vous êtes confronté à elles



# V. POSITIONNEMENT

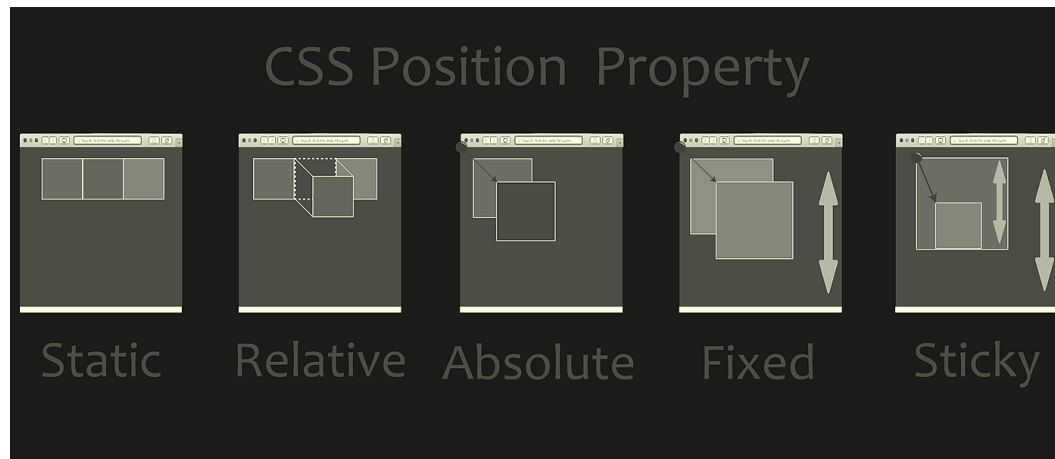
---



m2iinformation.fr

# POSITIONNEMENT

[Source image velog Danubi21](#)

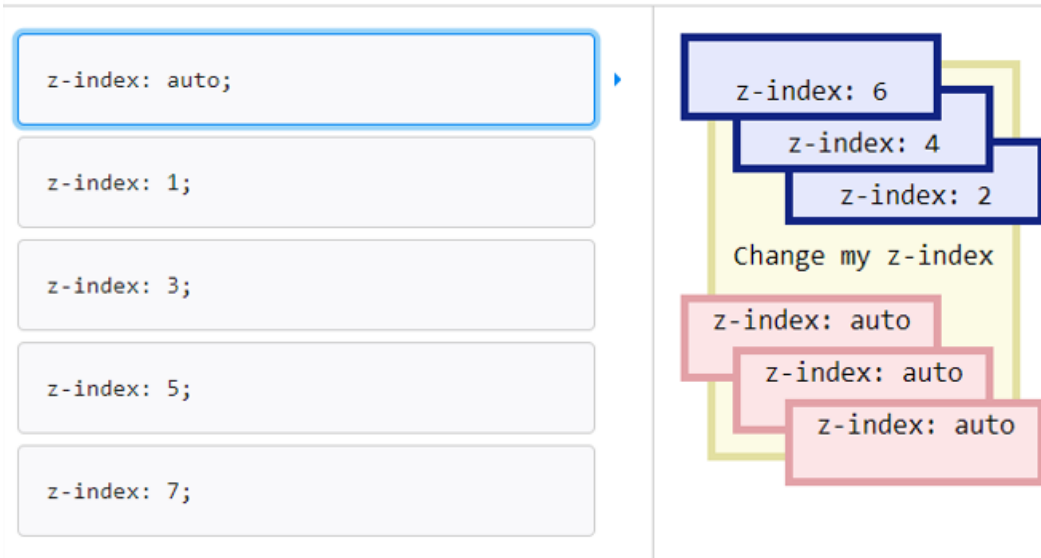


- Le positionnement permet de gérer la mise en page des éléments à l'aide de la propriété CSS *position*.
- Elle peut prendre 5 valeurs :
  1. *static*: valeur par défaut.
  2. *relative*: positionnement ajusté par rapport à la position initiale de l'élément.
  3. *absolute*: positionnement ajusté par rapport au parent le plus proche qui n'est pas en position *static*.
  4. *fixed*: semblable à *absolute*, cependant l'élément est toujours visible sur la fenêtre en scrollant (navigation sur la page).
  5. *sticky*: positionnement *relative* puis se transforme en *fixed* par rapport à son parent lorsqu'on défile la page.

# POSITIONNEMENT

[Source image MDN](#)

CSS Demo: z-index



The image shows a CSS demo for z-index. On the left, there is a list of five CSS rules in light blue boxes: `z-index: auto;`, `z-index: 1;`, `z-index: 3;`, `z-index: 5;`, and `z-index: 7;`. On the right, a visual representation shows three overlapping boxes. The topmost box is light blue with a dark blue border and contains `z-index: 6`. Below it is a light blue box with a dark blue border containing `z-index: 4`. The bottommost box is light blue with a dark blue border containing `z-index: 2`. These three boxes are stacked on top of a larger yellow box that contains the text "Change my z-index". Below the yellow box are three overlapping light red boxes, each containing `z-index: auto`.

- La propriété `position` est complétée par les propriétés *top*, *left*, *bottom* et *right* pour positionner l'élément à l'endroit souhaité. Les valeurs sont numériques avec une unité absolue ou relative. Elles peuvent être négatives.
- Les éléments qui ne sont pas positionnés de manière statique « n'appartiennent plus » au document HTML, ils sont en quelque sorte ajoutés « par dessus » du HTML
- La propriété **z-index** permet de gérer l'affichage des éléments lorsqu'ils se chevauchent.



# EXERCICE 4

0-exercices/ex4.md



# Ce qu'il faut retenir



SAVE

- Le positionnement avec une autre propriété que `static` fait sortir votre élément du document. Cet élément vient se rajouter sur le document par rapport aux « coordonnées » qu'on peut lui donner grâce aux propriétés `top`, `left`, `right` et `bottom`
- Pour gérer les chevauchements des éléments, il faut utiliser la propriété `z-index`

# Conseils du formateur



- Utile pour modifier le positionnement de certains éléments par rapport à leurs parents
- S'en servir principalement pour rendre les éléments flottants tels que les menus, les contenus additionnels sur les côtés, etc.

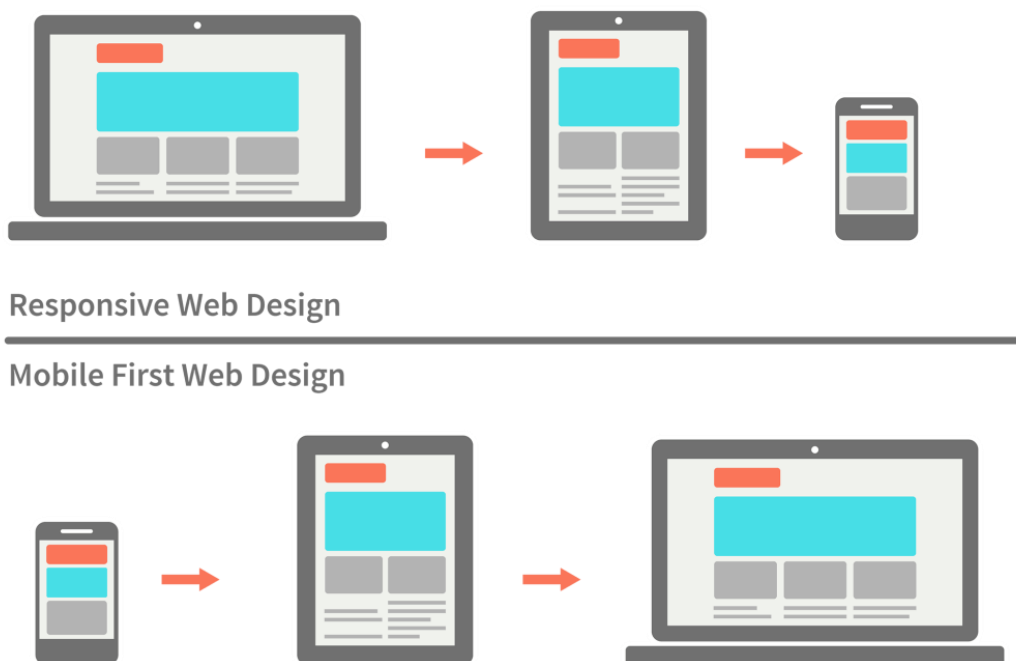
## VI. MOBILE FIRST ET RESPONSIVE DESIGN

---



# MOBILE FIRST ET RESPONSIVE DESIGN

[Source image Frederic Gonzalo](#)



- Le **Mobile First** est la méthodologie de conception inventée par *Luke Wroblewski* en 2011.

**Concevoir** son application d'abord sur **mobile** puis adapter le contenu pour les écrans plus grands.

- Le **Responsive Web Design** est une méthode qui permet de créer une page Web qui **s'adapte automatiquement** par rapport à la **taille de l'écran d'affichage**.

- Ces méthodes sont possible grâce à la mise en place des *medias queries* dans le style **CSS**.

1 seul code **HTML** et des règles **CSS** différentes selon les tailles d'écran.

# MISE EN PLACE DES MEDIA QUERIES

[Source image Christian Lisangola](#)

```
@media screen and (min-width: 480px) {  
    /* Votre code CSS */  
}  
  
@media screen and (min-width: 768px) {  
    /* Votre code CSS */  
}  
  
@media screen and (min-width: 992px) {  
    /* Votre code CSS */  
}  
  
@media screen and (min-width: 1200px) {  
    /* Votre code CSS */  
}
```

[Source image Megawebdesign](#)



# Types de media

Types de média et opérateurs logiques cf. <a href="#">documentation</a>	
<i>all</i>	Caractéristique pour désigner tout type de media ( <i>screen, print, speech</i> )
<i>only</i>	Caractéristique pour indiquer les règles à appliquer uniquement pour un type de media spécifique
<i>min-width/width/max-witdh</i>	Largeur minimal (au moins) ou fixe ou maximal (jusqu'à)
<i>orientation</i>	Orientation du périphérique, <i>portrait</i> ou <i>landscape</i> (paysage)
<i>and</i>	Opérateur logique permettant de combiner les caractéristiques
<i>not</i>	Opérateur logique permettant de spécifier ce qu'on ne veut pas (l'opposé)

# TAILLES DES ÉCRANS

Tailles	
Maximum <b>600px</b> de large	<b>Smartphone, montre</b>
Minimum 600px de large	Orientation portrait tablette et phablette (entre smartphone et <b>tablette</b> )
Minimum <b>768px</b> de large	Orientation paysage tablette
Minimum <b>992px</b> de large	<b>Ordinateurs portable et de bureau</b>
Minimum <b>1200px</b> de large	Écrans de <b>télévision</b> , ordinateurs portables, <b>ordinateurs de bureau très large</b>

- Les tailles des écrans peuvent varier en fonction des librairies CSS utilisées plus tard. Ici, les tailles sont celles définies par le site [W3schools](https://www.w3schools.com/css/css3_media_queries.asp)

# Avantages du Responsive Design et du Mobile First

Source image Webhoppers

## 10 benefits of a responsive website



1  
Functions  
on all  
Devices



2  
Clean,  
Flexible  
Grids &  
Layouts



3  
Adapts to  
all Screen  
Sizes



4  
One  
Universal  
URL



5  
Decreased  
load time



6  
Increased  
site speed



7  
Future-  
Proof



8  
Higher  
search  
rankings  
(SEO)



9  
Highest-  
quality



10  
One  
Codebase

- Développement rapide
- Cout faible
- Bon pour le référencement naturel (SEO) des moteurs de recherche
- Transition automatique et fluide de l'interface utilisateur en fonction de la taille de l'écran
- Couverture de toutes les tailles d'écran très facile et aisé à l'aide des media queries





# EXERCICE 5

0-exercices/ex5.md

# CHALLENGE ([source image edflex](#))



# Ce qu'il faut retenir



SAVE

- Le Mobile First permet de penser sa mise en page sur mobile et ensuite l'adapter pour les autres tailles d'écran
- À l'inverse, le responsive design permet d'adapter une mise en page prévue initialement pour la version desktop aux autres tailles d'écran plus petites
- Dans les deux cas, la mise en page sera différente selon les tailles d'écran, seul la stratégie diffère. D'un point de vue technique, dans les deux cas, on utilise les media queries

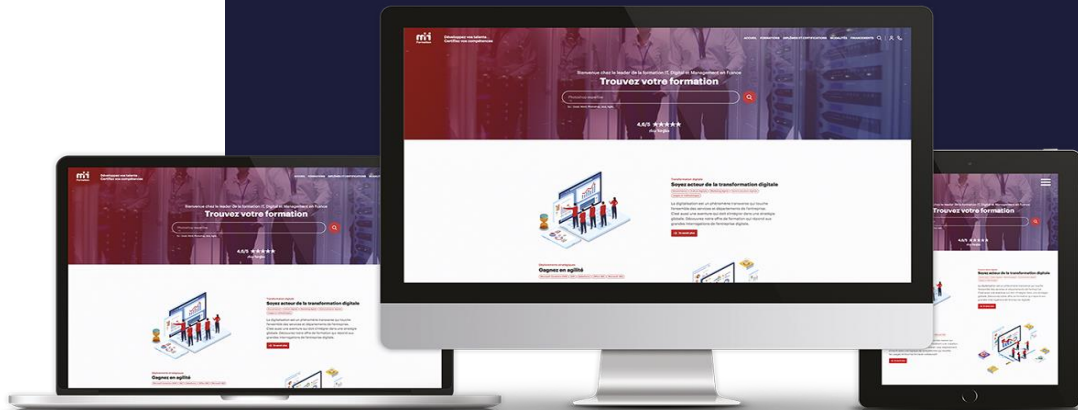
# Conseils du formateur



- Concevoir ses mises en page avec le Mobile First et en s'appuyant notamment sur les grilles pour faciliter l'intégration des maquettes par la suite. Autrement dit réaliser ces maquettes avec des grilles

# VII. ANIMATIONS ET TRANSITIONS

---

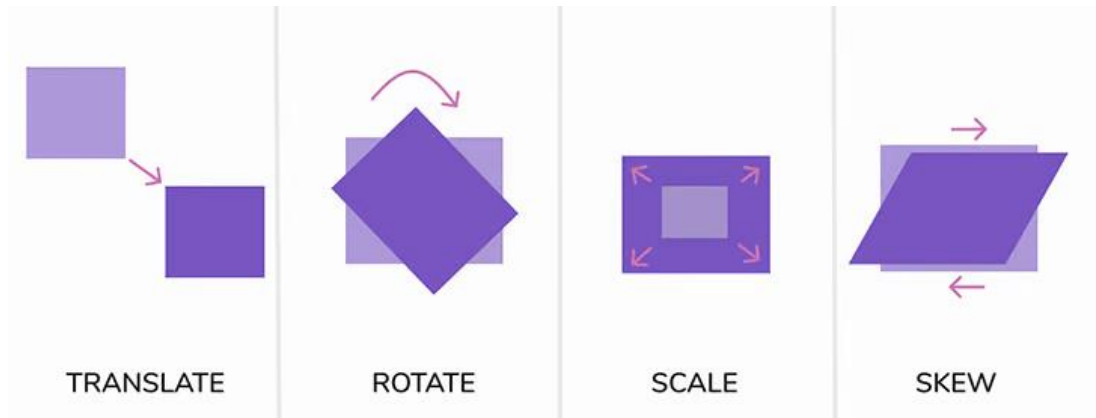


# TRANSFORMATION

---



[Source image logrocket.com](https://logrocket.com)



- Est possible grâce à la propriété *transform*, cf. [documentation](#)
- Permet de modifier la position d'un élément
  - La taille d'un élément
  - L'orientation d'un élément
  - L'inclinaison de l'élément
  - Etc.

PROPRIÉTÉ	VALEURS	
<i>transform-origin</i>	Unités et mots-clés : %, px, rem, right, top, center, etc.  Nombre : 1 ou 2 ( X et/ Y)	Par défaut, toutes les modifications partent du centre de l'élément, la propriété <i>transform-origin</i> permet de modifier le point de départ (point d'ancrage).

# Fonctions de la propriété *transform*

FONCTION	
<i>translate()</i> , <i>translateX()</i> , <i>translateY()</i> , <i>translate3d()</i>	Déplacer l'élément dans une ou plusieurs directions (axe)
<i>scale()</i> , <i>scaleX()</i> , <i>scaleY()</i> , <i>scale3d()</i>	Agrandir, rétrécir l'élément
<i>rotate()</i> , <i>rotateX()</i> , <i>rotateY()</i> , <i>rotate3d()</i>	Effectuer une rotation
<i>skew()</i> , <i>skewX()</i> , <i>skewY()</i>	Définir la courbe d'accélération (accélération et décélération) de la transition
<i>perspective()</i>	Indiquer la distance du spectateur. Mouvement amplifié lorsque le spectateur est proche et moins marqué lorsque la distance est plus longue.



# Démo : transformations

---

**TRANSITION**

# Transition

- Petite animation permettant à un élément de passer d'un état A à un état B lors d'un événement CSS, cf. documentation
- Les événements sont les pseudo-classes CSS
  - Survol d'un élément (*:hover*)
  - Clique d'un élément (*:active*)
  - Focus sur un élément (*:focus*)
  - Etc.

## Pour effectuer une transition il faut

1. Propriété CSS à modifier
2. Valeur initiale de la propriété
3. Valeur finale de la propriété
4. Durée
5. Événement

# Propriétés liées aux *transitions*

PROPRIÉTÉ	SIGNIFICATION
<i>transition-property</i>	Propriété dont la valeur sera modifiée
<i>transition-duration</i>	Durée du changement d'état en millisecondes ( <i>ms</i> ) ou secondes ( <i>s</i> )
<i>transition-delay</i>	Retarde le début de l'animation de X ms ou s
<i>Transition-timing-function</i>	Définis la <a href="#">courbe d'accélération</a> (accélération et décélération) de la transition
<i>transition</i>	Notation courte qui combine les propriétés transition-property, transition-duration, transition-timing-function dans cet ordre

PROPRIÉTÉ	VALEUR	SIGNIFICATION
<i>transition-property</i>	<i>all</i>	Toutes les propriétés présentes dans le pseudo-class vont avoir un changement d'état

# Démo : transitions

---

# ANIMATIONS

---





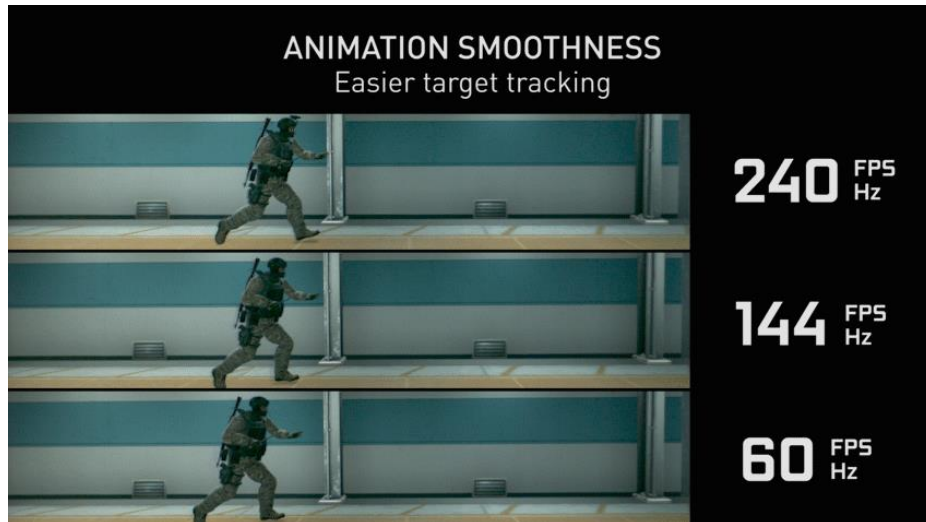
# Animations

- Animations plus élaborées
- Adapté pour UX Design d'émotion
  - Plus longue
  - Plus complexe
- Associé aux *@keyframes*, pour gérer l'évolution de l'animation (les valeurs des propriétés durant le temps d'exécution)
- [Documentation sur les animations](#)

# Explication fonctionnement des animations et de l'affichage du contenu par le navigateur ([source image Nvidia](#))

## Animation

- Succession d'images rapides perçues comme un mouvement par notre cerveau
- Frames Per Second (FPS)
  - Nombre d'images affichées par seconde
  - Plus il est élevé plus l'animation est fluide



## Navigateur

- 3 étapes après le chargement du style CSS
  1. **Layout**: détermine la mise en page et la taille des éléments définis dans le style CSS et les placent dans le DOM
  2. **Paint**: transforme les éléments en pixels
  3. **Composition**: Layout + Paint et affichage
- Pour avoir une animation fluide et optimisée, utilisez par exemple, les propriétés ***transform*** et ***opacity*** qui couvrent une bonne partie des besoins et sont moins gourmands aux niveaux des performances.
- Pour voir l'impact de l'utilisation des propriétés CSS dans les animations, vous pouvez consulter [csstriggers](#)



# Propriétés de l'animation

PROPRIÉTÉ	SIGNIFICATION
<i>animation-name</i>	Nom ou noms séparés par des virgules des animations attribuées sur l'élément cible (sélecteur)
<i>animation-duration</i>	Durée totale de l'animation en millisecondes (ms) ou secondes (s)
<i>animation-timing-function</i>	Définis la courbe d'accélération de l'animation
<i>animation-delay</i>	Latence (durée d'attente) avant de démarrer l'animation
<i>animation-iteration-count</i>	Nombre de cycles de répétition d'une animation avant sa fin
<i>animation-direction</i>	Direction des cycles d'animation
<i>animation-fill-mode</i>	Détermine quelles valeurs des propriétés seront gardés entre les valeurs initiales et finales à la fin de l'animation

VALEURS	SIGNIFICATION
<i>from</i> {...} <i>to</i> {...}	Dans le <i>from</i> , on indique les valeurs initiales et dans <i>to</i> les valeurs finales, les valeurs vont évoluer de <i>from</i> à <i>to</i>
<i>0 %</i> {} <i>25 %</i> {} <i>33 %</i> {} ... <i>100%</i> {}	Modification des propriétés à X % de la durée totale

# Démo : animations

---

# Bonnes pratiques de l'usage du CSS

- Approche *Mobile First*
  - CSS *Modulaire* (fichiers séparés en fonction des besoins)
  - Regroupez dans un seul fichier
  - *Compressez, minifiez les fichiers*
  - *Validez* votre code CSS
- Images aux formats *SVG*
  - Choisissez les sélecteurs les plus spécifiques comme l'ID au lieu de sélectionner les éléments en parcourant le DOM dans son intégralité
  - Utilisez les notations de propriétés courtes ou abrégées
  - Ayez les **bonnes tailles des images** (évitez des redimensionnement avec le CSS)
  - Utilisez les animations que lorsqu'elles sont rapides, pertinentes et respectent les critères UX Design d'homogénéité, universalité, etc.

# CHALLENGE ([source image edflex](#))



# Ce qu'il faut retenir



SAVE

- La propriété *transform* s'accompagne d'une multitude des fonctions qui permettent de créer des effets.
- *Transform*, *transition* et les événements CSS permettent de réaliser des animations simples et peu coûteuse au navigateur
- La propriété *animation* permet d'effectuer des animations beaucoup plus poussées en complexe en gérant les différentes phases et plusieurs autres propriétés qui tournent autour

# Conseils du formateur



- Effectuez les animations sur les couleurs et l'opacité de préférence, car elles permettent d'attirer l'œil de votre utilisateur sans être trop gourmand en performance
- Inspectez avec les outils de la *devtools* pour suivre et calibrer au mieux vos effets ainsi que le rendu et le coût en termes de performance
- N'abusez pas trop des animations, les utiliser que lorsque c'est vraiment pertinent
- Utilisez les [bibliothèques dédiées](#) pour intégrer les animations dans vos projets

# Évaluations formateur

N'oubliez pas les évaluations formateur avant de partir.





# MERCI D'AVOIR SUIVI CETTE FORMATION

---

