

# Mémento

## CSS

---

Version 2.0 (créé le 20/04/2024, modifié le 26/08/2024)

**CSS**



CSS (Cascading Style Sheets) est un langage utilisé pour mettre en forme une page web.

# Table des matières

1. Prise en main.....	4
1.1. Outils nécessaires .....	4
1.2. Où écrire le code CSS ? .....	4
2. Bases .....	4
2.1. Syntaxe .....	4
2.2. Commentaires .....	5
2.3. Les variables .....	5
2.3.1. Déclarer une variable .....	5
2.3.2. Utiliser une variable.....	5
2.4. Les couleurs .....	5
3. Les différents sélecteurs .....	6
3.1. Sélecteurs simples.....	6
3.2. Les pseudo-classes.....	7
3.3. Les pseudo-éléments .....	8
3.4. Les media queries .....	9
4. Les différentes propriétés .....	9
4.1. Propriétés sur les espacements (margin et padding) .....	9
4.2. Propriétés sur l'arrière-plan .....	10
4.3. Propriétés sur le texte .....	10
4.4. Propriétés sur la police .....	11
4.4.1. Propriétés prédéfinies .....	11
4.4.2. Importer une police personnalisée .....	12
4.1. Propriétés sur la bordure .....	12
4.2. Propriétés sur l'affichage (display) .....	13
4.2.1. block.....	13

4.2.2. inline.....	13
4.2.3. inline-block.....	13
4.2.4. none.....	13
4.2.5. grid.....	14
4.2.6. inline-grid.....	15
4.2.7. flex.....	15
4.2.8. inline-flex.....	17
4.3. Propriétés sur le positionnement.....	18
4.3.1. static.....	18
4.3.2. relative.....	18
4.3.3. absolute.....	18
4.3.4. fixed.....	19
4.4. Propriétés sur les transformations.....	19
4.4.1. Les différentes transformations.....	19
4.4.2. Déplacements par rapport à l'origine.....	20
4.5. Propriétés sur les transitions.....	20
4.5.1. Manière simple et propre.....	20
4.5.2. Manière rapide.....	21
4.5.3. Les fonctions temporelles.....	21
4.6. Propriétés sur les animations.....	22
4.6.1. Créer une animation.....	22
4.6.2. Utiliser les animations.....	23

# 1. Prise en main

## 1.1. Outils nécessaires

- Navigateur Internet à jour (ex : Firefox ou Chrome)
- Un logiciel de codage (Visual Studio Code (recommandé) ou Notepad++)
- Connaissances en HTML
- Valideur CSS : <https://jigsaw.w3.org/css-validator/>



Le mémento HTML est disponible en ligne sur le site :

[https://loricaudin.github.io/loric-informatique/langages/html/langage\\_html.html](https://loricaudin.github.io/loric-informatique/langages/html/langage_html.html)

## 1.2. Où écrire le code CSS ?

Soit dans un fichier (ex : style.css), mais entre les balises <head></head>, il faut rajouter : <link rel="stylesheet" href="style.css"/>

Soit entre les balises <style></style>, placées dans <body></body> dans le fichier HTML (à éviter).

# 2. Bases

## 2.1. Syntaxe

```
selecteur1 {  
    attribut1: valeur1;  
    attribut3: valeur2;  
}  
selecteur2 {  
    attribut3: valeur3;  
}
```

## 2.2. Commentaires

```
/*  
Commentaire pouvant être sur une ou plusieurs lignes  
*/
```

## 2.3. Les variables

### 2.3.1. Déclarer une variable

Les variables sont à déclarer dans le sélecteur html et peuvent prendre n'importe quel type de valeurs.

Syntaxe :

```
html {  
    --ma-variable: valeur;  
}
```

### 2.3.2. Utiliser une variable

```
attribut: var(--ma-variable);
```

## 2.4. Les couleurs

Format des couleurs	Exemples de valeurs possibles
Couleurs prédéfinies	blue green red
Code hexadécimal #rrggbb	#3a3aff
Système RGB	rgb(58, 58, 255);
Système RGBA (avec transparence (1 : opaque, 0 : transparent))	rgb(58, 58, 255, 1);

## 3. Les différents sélecteurs

### 3.1. Sélecteurs simples

Sélecteur	Description
*	Sélectionner toutes les balises
<i>balise1</i>	Sélectionner toutes les balises <i>balise1</i>
<i>balise1</i> , <i>balise2</i>	Sélectionner toutes les balises <i>balise1</i> et <i>balise2</i>
<i>balise1</i> > <i>balise2</i>	Sélectionner toutes les balises <i>balise2</i> filles directes de <i>balise1</i>
<i>balise1</i> <i>balise2</i>	Sélectionner toutes les balises <i>balise2</i> qui sont dans <i>balise1</i>
<i>balise1</i> + <i>balise2</i>	Sélectionner la première balise <i>balise2</i> qui suit la balise <i>balise1</i>
<i>balise1</i> ~ <i>balise2</i>	Sélectionner toutes les balises <i>balise2</i> qui sont après <i>balise1</i> et au même niveau
<i>balise1</i> [ <i>attribut1</i> ]	Sélectionner toutes les balises <i>balise1</i> qui ont un attribut <i>attribut1</i>
<i>balise1</i> [ <i>attribut1</i> ="valeur1"]	Sélectionner toutes les balises <i>balise1</i> qui ont un attribut <i>attribut1</i> avec pour valeur <i>valeur1</i>
<i>balise1</i> . <i>classe1</i>	Sélectionner les balises <i>balise1</i> qui ont pour classe <i>classe1</i>
. <i>classe1</i>	Sélectionner toutes les balises qui ont pour classe <i>classe1</i>
<i>balise1</i> # <i>id1</i>	Sélectionner une balise <i>balise1</i> qui a pour identifiant <i>id1</i>
# <i>id1</i>	Sélectionner une balise qui a pour identifiant <i>id1</i>

### 3.2. Les pseudo-classes

Les pseudo-classes sont similaires aux classes HTML, mais elles sont attachées automatiquement aux éléments qui correspondent à leur définition.

Syntaxe : *selecteur:pseudo-classe*

Pseudo-classe	Description
:link	Lien pas encore visité
:visited	Lien déjà visité
:active	Élément en train d'être cliqué
:focus	Élément ayant le focus clavier (pour un élément dans lequel on peut taper du texte) ou ayant été cliqué
:hover	Élément survolé par la souris
:target	Élément ciblé par l'URL courante (pour les ancres)
:first-child	Premier élément du parent
:last-child	Dernier élément du parent
:nth-child(3)	3ème élément du parent
:nth-child(2n)	Élément du parent de rang pair
:nth-child(2n+1)	Élément du parent de rang impair
:only-child	Élément unique dans le parent
:empty	Élément sans enfant

### 3.3. Les pseudo-éléments

Les pseudo-éléments sont des éléments virtuels contenant un sous-ensemble d'un élément existant.

Syntaxe : *selecteur::pseudo-element*

Pseudo-élément	Description
::first-line	Première ligne de l'élément (souvent un paragraphe)
::first-letter	Première lettre de l'élément (souvent un paragraphe)
::before	Emplacement avant l'élément (vide par défaut)
::after	Emplacement après l'élément (vide pas défaut)

Pour remplir un élément vide, on peut ajouter la propriété : `content` : *contenu*; avec pour contenu un texte entre guillemets ou une image dans la fonction `url()`.



### 3.4. Les media queries

Media query	Description
<pre>media screen and (max-width: taille) {   selecteur1 {     attribut1: valeur1;     attribut2: valeur2;   }   selecteur2 {     attribut3: valeur3;   }   ... }</pre>	Appliquer un style différent pour les petits écrans et les mobiles
<pre>media print {   selecteur1 {     attribut1: valeur1;     attribut2: valeur2;   }   selecteur2 {     attribut3: valeur3;   }   ... }</pre>	Appliquer un style différent pour les impressions

## 4. Les différentes propriétés

### 4.1. Propriétés sur les espacements (margin et padding)

Propriété	Description
<pre>margin: taille;  margin-top: taille; margin-bottom: taille; margin-left: taille; margin-right: taille;</pre>	Changer l'espacement entre l'élément actuel et l'élément parent (extérieur)
<pre>padding: taille;  padding-top: taille; padding-bottom: taille; padding-left: taille; padding-right: taille;</pre>	Changer l'espacement entre l'élément actuel et les éléments enfants (intérieur)

## 4.2. Propriétés sur l'arrière-plan

Propriété	Description
background-color: <i>couleur</i> ;	Appliquer un arrière-plan uni
background-image: linear-gradient(90deg, <i>couleur1</i> , <i>couleur2</i> ...);	Appliquer un arrière-plan dégradé (des pourcentages peuvent être ajoutées à chaque couleur et les fonctions peuvent être ajoutée à la suite, séparées par des virgules)
background-image: url( <i>Lien</i> );	Appliquer une image en arrière-plan
background-position: left; background-position: top;	Fixer un arrière-plan en haut ou à gauche (si l'image n'est pas répétée)
background-attachment: fixed; background-attachment: scroll;	Bloquer ou non le défilement de l'arrière-plan (scroll par défaut)
background-repeat: no-repeat; background-repeat: repeat-x; background-repeat: repeat-y; background-repeat: repeat;	Faire répéter l'arrière-plan (repeat par défaut)

## 4.3. Propriétés sur le texte

Propriété	Description	Valeurs possibles
color: <i>couleur</i> ;	Changer la couleur du texte	Exemple : blue
text-decoration: <i>valeur</i> ;	Changer la décoration du texte (à utiliser pour retirer la décoration d'un lien si besoin)	none underline
text-transform: <i>valeur</i> ;	Transformer le texte	none capitalize uppercase lowercase
text-align: <i>valeur</i> ;	Changer l'alignement du texte	left right center justify

## 4.4. Propriétés sur la police

### 4.4.1. Propriétés prédéfinies

Propriété	Description	Valeurs possibles
<code>font-style: valeur;</code>	Changer le style de la police	normal italic oblique
<code>font-weight: valeur;</code>	Changer l'épaisseur de la police (pour mettre en gras)	normal bold bolder
<code>font-variant: valeur;</code>	Changer l'apparence de la police (small-caps met tout en capitale avec les lettres en majuscule plus grandes)	normal small-caps
<code>font-family: valeur1, valeur2...;</code>	Remplacer la police par la première disponible de la liste (si la première n'existe pas, c'est la deuxième qui est chargée...)	Calibri Arial serif sans-serif ...
<code>font-size: taille;</code>	Changer la taille du texte	20px 20pt 0.2in 0.5cm 5mm 2pc

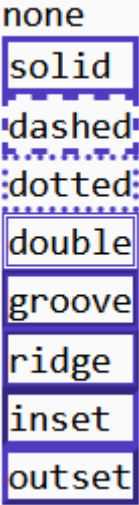
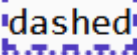
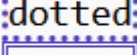
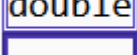
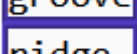
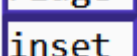
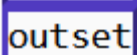
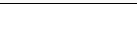
#### 4.4.2. Importer une police personnalisée

Pour importer une police personnalisée, vous devez récupérer un fichier .ttf ou .eot. Pour l'importer, entrer les lignes suivantes :

```
@font-face {  
    font-family: "maPolice";  
    src: url("LienDeLaPolice.ttf");  
}
```

La police pourra ensuite être utilisée avec : `font-family: "maPolice";` dans les sélecteurs à appliquer.

#### 4.1. Propriétés sur la bordure

Propriété	Description	Valeurs possibles
<code>border: <i>taille style couleur</i>;</code>  <code>border-top: <i>taille style couleur</i>;</code> <code>border-bottom: <i>taille style couleur</i>;</code> <code>border-left: <i>taille style couleur</i>;</code> <code>border-right: <i>taille style couleur</i>;</code>	Changer la bordure (épaisseur, style et couleur)	Pour le style : <div><div>none</div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>
<code>border-radius: <i>valeur</i>;</code>  <code>border-top-left-radius: <i>valeur</i>;</code> <code>border-top-right-radius: <i>valeur</i>;</code> <code>border-bottom-left-radius: <i>valeur</i>;</code> <code>border-bottom-right-radius: <i>valeur</i>;</code>	Changer l'arrondi des bords	<i>taille</i>

## 4.2. Propriétés sur l'affichage (display)

Syntaxe : `display: methode;`

### 4.2.1. block

Il s'agit de la valeur par défaut pour les balises `<p>`, `<h1>`, `<section>`, `<nav>`, `<article>`... Elle permet d'afficher un élément (utile si le `display` était en `none`), impose un retour à la ligne après l'élément et permet de modifier sa taille (elle prend la longueur maximale et la hauteur minimale par défaut).

### 4.2.2. inline

Il s'agit de la valeur par défaut pour les balises `<a>`, `<em>`, `<strong>`, `<span>`, `<i>`.... Elle permet d'afficher un élément (utile si le `display` était en `none`), n'impose pas de retour à la ligne après l'élément et prend la taille minimale nécessaire (non modifiable).

### 4.2.3. inline-block

Il s'agit d'un mélange des méthodes `block` et `inline`, c'est-à-dire qu'il n'y a pas de retour à la ligne après l'élément, mais on peut modifier sa taille.

### 4.2.4. none

Avec cette valeur, l'élément n'apparaît pas à l'écran.

#### 4.2.5. grid

Il s'agit d'un positionnement dans une grille.

Propriété	Description	Exemple
<code>grid-template-rows: valeur;</code> <code>grid-template-columns: valeur;</code>	Définir la taille des lignes et des colonnes (unités possibles : fr (pour les fractions de taille de contenu encore à prendre), % (pour des pourcentages), auto (pour prendre toute la place restante), px (pour des tailles fixes))	Exemple : On souhaite une grille de 3 lignes et de 2 colonnes. On a donc : <code>grid-template-rows: 1fr 2fr 1fr;</code> <code>grid-template-columns: 30% 70%;</code>
<code>grid-row-start: valeur;</code> <code>grid-row-end: valeur;</code> <code>grid-column-start: valeur;</code> <code>grid-column-end: valeur;</code>	Définir la position d'un élément dans la grille (par défaut dans la première case vide)	Exemple : Considérons une grille de 3 lignes et de 2 colonnes. Pour placer 1 élément dans la 2ème et 3ème ligne de la 1ère colonne, on a donc : <code>grid-row-start: 2;</code> <code>grid-row-end: 4;</code> <code>grid-column-start: 1;</code> <code>grid-column-end: 2;</code>
<code>grid-template-areas: "valeur1 valeur2..."</code> <code>"valeur3 valeur4..."...</code>	Définir des noms aux zones pour faciliter le positionnement	Exemple : Considérons une grille de 3 lignes et de 2 colonnes. Pour placer 1 élément dans la 2ème et 3ème ligne de la 1ère colonne, on peut faire : <code>grid-template-areas: "zone1 zone2"</code> <code>"zone3 zone4"</code> <code>"zone3 zone5";</code> Il restera plus qu'à positionner l'élément

		avec : grid-area: zone3;
gap: <i>taille</i> ; row-gap: <i>taille</i> ; column-gap: <i>taille</i> ;	Changer l'espacement des éléments enfants	(Aucun exemple)

#### 4.2.6. inline-grid

Il s'agit d'un mélange des méthodes grid et inline-block, c'est-à-dire qu'il n'y a pas de retour à la ligne après l'élément, mais on peut modifier sa taille.

#### 4.2.7. flex

Il s'agit d'un positionnement dans une boîte flexible.

Propriété	Description	Valeurs possibles
flex-direction: <i>direction</i> ;	Définir la direction de l'axe principal pour positionner les éléments enfants	row (par défaut) : de gauche à droite column : de haut en bas row-reverse : de droite à gauche column-reverse : de bas en haut
flex-wrap: <i>valeur</i> ;	Définir si les éléments enfants peuvent changer de ligne (si la direction est row ou row-reverse)	nowrap (par défaut) : pas de retour à la ligne wrap : retour à la ligne s'il n'y a pas assez de place wrap-reverse : retour à la ligne s'il n'y a pas assez de place (remonté au-dessus)
flex: <i>valeur</i> ;	Définir si un élément doit être flexible ou non (à mettre dans l'élément enfant)	1 : flexible (prend la place disponible) 0 (par défaut) : non

		flexible (prend la place minimale)
<code>justify-content: valeur;</code>	Répartir les éléments sur l'axe principal	flex-start (par défaut) : au début flex-end : à la fin center : Centré (avec uniquement de l'espace autour) space-around : avec de l'espace entre les éléments et autour space-between : avec de l'espace uniquement entre les éléments space-evenly : avec de l'espace entre les éléments et autour
<code>align-content: valeur;</code>	Gérer la répartition des lignes (si le retour à la ligne est autorisé)	stretch (par défaut) : toute la place disponible flex-start : au début flex-end : à la fin center : centré verticalement space-around : avec de l'espace entre les lignes et autour space-between : avec de l'espace uniquement entre les lignes
<code>align-items: valeur;</code>	Gérer l'alignement des items sur l'axe secondaire (inverse de l'axe principale)	stretch (par défaut) : toute la place disponible flex-start : au début flex-end : à la fin



		center : centré sur l'axe secondaire
<code>align-self: valeur;</code>	Gérer l'alignement d'un seul item sur l'axe secondaire (inverse de l'axe principale)	stretch (par défaut) : toute la place disponible flex-start : au début flex-end : à la fin center : centré sur l'axe secondaire
<code>order: position;</code>	Changer l'emplacement d'un élément	-1 : décale d'une place vers le début 1 : décale d'une place vers la fin

#### 4.2.8. inline-flex

Il s'agit d'un mélange des méthodes flex et inline-block, c'est-à-dire qu'il n'y a pas de retour à la ligne après l'élément, mais on peut modifier sa taille.

## 4.3. Propriétés sur le positionnement

Syntaxe : `position: methode;`

### 4.3.1. static

Il s'agit du positionnement par défaut. L'élément est placé à sa position par défaut et ne peut pas être déplacé.

### 4.3.2. relative

Cette méthode décale l'élément par rapport à sa position par défaut.

Propriété	Description
<code>top: Longueur;</code>	Déplace l'élément vers le bas
<code>bottom: Longueur;</code>	Déplace l'élément vers le haut
<code>left: Longueur;</code>	Déplace l'élément vers la droite
<code>right: Longueur;</code>	Déplace l'élément vers la gauche

### 4.3.3. absolute

Cette méthode positionne l'élément par rapport à la page et non par rapport à sa position par défaut.

Propriété	Description
<code>top: Longueur;</code>	Écart entre l'élément et le haut de la page
<code>bottom: Longueur;</code>	Écart entre l'élément et le bas de la page
<code>left: Longueur;</code>	Écart entre l'élément et le bord gauche de la page
<code>right: Longueur;</code>	Écart entre l'élément et le bord droit de la page

#### 4.3.4. fixed

Cette méthode positionne l'élément par rapport à l'écran et non par rapport à sa position par défaut. L'élément est insensible au défilement de la page (avec le media print, l'élément apparaît sur toutes les pages).

Propriété	Description
top: <i>Longueur</i> ;	Écart entre l'élément et le haut de l'écran
bottom: <i>Longueur</i> ;	Écart entre l'élément et le bas de l'écran
left: <i>Longueur</i> ;	Écart entre l'élément et le bord gauche de l'écran
right: <i>Longueur</i> ;	Écart entre l'élément et le bord droit de l'écran

#### 4.4. Propriétés sur les transformations

Syntaxe : `transform: transformation1 transformation2...;`

##### 4.4.1. Les différentes transformations

Transformation	Description
rotate(90deg)	Tourner de 90 degrés vers la droite
skewX(10deg)	Appliquer une distorsion de 10 degrés (avec le haut vers la gauche et le bas vers la droite)
skewY(20deg)	Appliquer une distorsion de 20 degrés (avec la gauche vers le haut et la droite vers le bas)
skew(10deg, 20deg)	Appliquer une distorsion sur les 2 axes en même temps
translateX(100px)	Déplacer une image de 100 pixels vers la droite
translateY(200px)	Déplacer une image de 200 pixels vers le bas

<code>translate(100px, 200px)</code>	Déplacer une image selon les 2 axes
--------------------------------------	-------------------------------------

#### 4.4.2. Déplacements par rapport à l'origine.

Commande	Description
<code>transform-origin: center;</code>	Transformer par rapport au centre de l'élément
<code>transform-origin: top left;</code> <code>transform-origin: bottom right;</code>	Transformer par rapport au coin de l'élément
<code>transform-origin: 10px 20px;</code> <code>transform-origin: bottom right;</code>	Transformer par rapport au coin supérieur gauche de l'élément avec un décalage de 10 pixels à droite et de 20 pixels en bas

### 4.5. Propriétés sur les transitions

Les transitions permettent d'animer les éléments qui ont des pseudo-classes (par exemple lors du survol de la souris).

#### 4.5.1. Manière simple et propre

Propriété	Description
<code>transition-property: attribut1, attribut2...;</code>	Sélectionner les propriétés à animer (Pour animer toutes les propriétés, il est possible de remplacer la propriété par <code>all</code> )
<code>transition-duration: duree1, duree2...;</code>	Appliquer la durée de la transition à la propriété positionnée au même emplacement (avec la durée sous la forme : 1s par exemple pour 1 seconde)
<code>transition-timing-function: fonction-temporelle1, fonction-temporelle2...;</code>	Attribuer une fonction temporelle à la propriété positionnée au même

	emplacement (voir les fonctions temporelles)
<code>transition-delay: <i>delai1</i>, <i>delai2</i>...;</code>	Appliquer un délai à la propriété positionnée au même emplacement (avec le délai sous la forme : 1s par exemple pour 1 seconde)

#### 4.5.2. Manière rapide

Syntaxe : `transition: attribut1 duree1 fonction-temporelle1 delai1, attribut2 duree2 fonction-temporelle2 delai2...;`

Note : Le délai et la fonction ne sont pas à renseigner obligatoirement.

#### 4.5.3. Les fonctions temporelles

Fonction temporelle	Description du mouvement
linear	Même vitesse du début jusqu'à la fin
ease	Ralentissement à la fin (valeur par défaut)
ease-in	Ralentissement au début et accélération à la fin
ease-out	Accélération au début et ralentissement à la fin
ease-in-out	Accélération au début et à la fin uniquement
cubic-bezier( <i>c1</i> , <i>c2</i> , <i>c3</i> , <i>c4</i> )	Courbe de Bézier paramétrable à l'aide de quatre coefficients compris entre 0 et 1
steps( <i>n</i> , jump-start) ou steps( <i>n</i> , start) ou step-start (si <i>n</i> vaut 1)	Nombre d'étapes à partir du début

<code>steps(n, jump-start)</code> ou <code>steps(n, start)</code> ou <code>step-start</code> (si <i>n</i> vaut 1)	Nombre d'étapes à partir de la fin
<code>steps(n, jump-none)</code>	Nombre d'étapes entre le début et la fin compris
<code>steps(n, jump-both)</code>	Nombre d'étapes entre le début et la fin non compris

## 4.6. Propriétés sur les animations

### 4.6.1. Créer une animation

#### 4.6.1.1. Animation simple avec un début et une fin

```
@keyframe mon-animation {
  from {
    attribut1: valeur1;
    attribut2: valeur2;
  }
  to {
    attribut1: valeur3;
    attribut2: valeur4;
  }
}
```

#### 4.6.1.2. Animation complexe avec des pourcentages

```
@keyframe mon-animation {
  0% {
    attribut1: valeur1;
    attribut2: valeur2;
  }
  25% {
    attribut1: valeur3;
    attribut2: valeur4;
  }
}
```

```

50% {
    attribut1: valeur5;
    attribut2: valeur6;
}
75% {
    attribut1: valeur7;
    attribut2: valeur8;
}
100% {
    attribut1: valeur9;
    attribut2: valeur10;
}
}

```

## 4.6.2. Utiliser les animations

### 4.6.2.1. Manière simple et propre

Propriété	Description
<code>animation-name: mon-animation1, mon-animation2...;</code>	Appliquer des animations
<code>animation-duration: duree1, duree2...;</code>	Appliquer la durée à l'animation positionnée au même emplacement (avec la durée sous la forme : 1s par exemple pour 1 seconde)
<code>animation-timing-function: fonction-temporelle1, fonction-temporelle2...;</code>	Attribuer une fonction temporelle à l'animation au même emplacement (voir les fonctions temporelles (dans les transitions))
<code>animation-delay: delai1, delai2...;</code>	Appliquer un délai à l'animation positionnée au même emplacement (avec le délai sous la forme : 1s par exemple pour 1 seconde)
<code>animation-direction: direction1, direction2...;</code>	Appliquer une direction à l'animation positionnée au même emplacement (normal, reverse,

	alternate (dans les 2 sens), alternate-reverse)
animation-play-state: <i>etat1</i> , <i>etat2</i> ...;	Choisir l'état de l'animation positionnée au même emplacement entre running et paused
animation-iteration-count: <i>iteration1</i> , <i>iteration2</i> ...;	Appliquer un nombre d'itération de l'animation positionnée au même emplacement (les nombres décimaux sont autorisés et infinite permet une boucle infinie)
animation-fill-mode: <i>fill-mode1</i> , <i>fill-mode2</i> ...;	Choisir un style que doit prendre avant et après l'animation positionnée au même emplacement (none (aucun style avant et après l'animation), forwards (garder le style de l'animation après son exécution), backwards (récupérer le style de l'animation avant son exécution), both (prendre le style de l'animation avant et après son exécution))

#### 4.6.2.2. Manière rapide

Syntaxe: animation: *duree1 fonction-temporelle1 delai1 iteration1 direction1 fill-mode1 etat1 mon-animation1*, *duree2 fonction-temporelle2 delai2 iteration2 direction2 fill-mode2 etat2 mon-animation2*...;

Note : La durée et le nom de l'animation sont à renseigner obligatoirement.