I. Introduction au CSS

Le **CSS** (Cascading Style Sheets) est un langage utilisé pour décrire la présentation visuelle des pages web. Il permet de contrôler la mise en forme des éléments HTML, comme les couleurs, les polices, les marges, et la disposition. CSS sépare le contenu (HTML) de la présentation, facilitant la maintenance et la modification de l'apparence d'un site web.

1. Définition d'une règle CSS

Une **règle CSS** se compose de deux parties principales :

1. **Sélecteur** : Il identifie les éléments HTML à styliser.
2. **Déclaration** : Elle contient les propriétés CSS et leurs valeurs à appliquer aux éléments sélectionnés. Par exemple :

```
p {
  color: blue;
  font-size: 16px;
}
```

Ici, le sélecteur `p` cible tous les éléments de paragraphe `<p>`, et la déclaration modifie leur couleur et la taille de la police.

2. Types de Sélecteurs CSS

a) **Sélecteur de type** : Cible un élément HTML par son nom. Ex. : `p` pour tous les paragraphes.

```
p { color: red; }
```

b) **Sélecteur de classe** : Cible un ou plusieurs éléments avec une classe spécifique, précédé d'un point (.). Ex. : `.important`.

```
.important { font-weight: bold; }
```

c) **Sélecteur d'ID** : Cible un élément unique avec un identifiant spécifique, précédé d'un dièse (#). Ex. : `#header`.

```
#header { background-color: yellow; }
```

d) **Sélecteur universel** : Cible tous les éléments (`*`).

```
* { margin: 0; }
```

e) **Sélecteur d'attribut** : Cible les éléments avec un attribut spécifique. Ex. : `[type="text"]`.

```
input[type="text"] { border: 1px solid black; }
```

f) **Sélecteurs combinés** : Combinaison de sélecteurs pour plus de précision.

Cible les éléments descendant d'un autre. Ex. : div p (tous les <p> à l'intérieur d'un).

```
div p { color: green; }
```

II. Propriétés de mise en forme du texte en CSS

Le CSS offre un large éventail de propriétés permettant de styliser le texte dans les pages web. Voici quelques-unes des plus courantes, avec des exemples :

1. font-family :

Cette propriété permet de définir la police de caractères utilisée pour le texte.

Exemple :

```
p {
  font-family: Arial, sans-serif;
}
```

Ici, le texte des paragraphes utilisera la police **Arial** avec une alternative **sans-serif** si Arial n'est pas disponible. Voici les 5 famille de polices avec des exemples des fonts :

Generic Font Family	Examples of Font Names
Serif	Times New Roman Georgia Garamond
Sans-serif	Arial Verdana Helvetica
Monospace	Courier New Lucida Console Monaco
Cursive	Brush Script MT Lucida Handwriting
Fantasy	Copperplate Papyrus

2. font-weight :

Elle permet de définir l'épaisseur du texte. Les valeurs possibles incluent normal, bold, ou des valeurs numériques comme 100 à 900.

Exemple :

```
h1 {
  font-weight: bold;
}
p {
  font-weight: 300;
}
```

Ici, les titres <h1> seront en gras, tandis que les paragraphes auront un texte plus léger.

3. font-style :

Utilisée pour mettre le texte en italique. Les valeurs possibles sont normal, italic, ou oblique.

Exemple :

```
em {
  font-style: italic;
}
```

Le texte encadré par sera en italique.

4. font-size :

Elle permet de définir la taille du texte. Les valeurs peuvent être en pixels (px), en pourcentage (%), en em.

Exemple :

```
p {
  font-size: 16px;
}
h1 {
  font-size: 2em;
}
```

Le texte des paragraphes sera de **16px**, tandis que les titres <h1> seront deux fois plus grands que la taille de police par défaut.

5. font :

Cette propriété regroupe plusieurs propriétés liées aux polices, comme font-style, font-weight, font-size, et font-family dans une seule déclaration.

Exemple :

```
p {
  font: italic bold 16px Arial, sans-serif;
}
```

Cela rendra le texte des paragraphes **italique**, **en gras**, avec une taille de **16px** et la police **Arial**.

6. text-align :

Elle contrôle l'alignement horizontal du texte. Les valeurs incluent left, right, center, et justify.

Exemple :

```
h1 {
  text-align: center;
}
p {
  text-align: justify;
}
```

Ici, les titres seront centrés, tandis que les paragraphes seront justifiés (alignés des deux côtés).

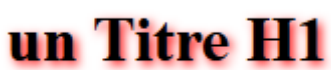
7. text-shadow :

Cette propriété permet d'ajouter une ombre au texte. Elle prend des valeurs pour l'offset horizontal et vertical, le flou, et la couleur.

Exemple :

```
h1 {
  text-shadow: 2px 2px 5px red;
}
```

Cela appliquera une ombre rouge au titre <h1> avec un décalage de **2px** dans les deux directions et un flou de **5px**.



8. text-transform :

Elle permet de modifier la casse du texte. Les valeurs incluent uppercase, lowercase, et capitalize.

Exemple :

```
h2 {
  text-transform: uppercase;
}
```

Cela transformera tout le texte des titres <h2> en **majuscules**.

9. color :

Cette propriété définit la couleur du texte. Les couleurs peuvent être spécifiées en nom (comme red), en hexadécimal (#ff0000), en RGB (rgb(255, 0, 0)), ou en.

Exemple :

```
p {
  color: #333333;
}
```

Le texte des paragraphes sera de couleur **gris foncé** (#333333).

QCM :

QCM 1 : Quel sélecteur CSS est utilisé pour cibler un élément avec un identifiant spécifique ?

1. .class
2. #id
3. element
4. [attribute]

QCM 2 : Quelle propriété CSS est utilisée pour définir la police de caractères d'un élément ?

1. font-weight
2. font-family
3. font-size
4. text-align

QCM 3 : Quel sélecteur CSS cible tous les éléments de type paragraphe (<p>) dans un document HTML ?

1. .p
2. #p
3. p
4. [p]

QCM 4 : Quelle propriété CSS permet de définir l'épaisseur d'une police de caractères ?

1. font-style
2. font-weight
3. font-size
4. font-family

QCM 5 : Quel sélecteur CSS permet de cibler tous les éléments d'une page ?

1. *
2. #
3. .
4. div

QCM 6 : Quelle propriété CSS est utilisée pour mettre un texte en italique ?

1. font-style
2. font-family
3. font-weight
4. text-transform

QCM 7 : Quelle propriété CSS définit l'alignement du texte à gauche, à droite, au centre, ou justifié ?

1. font-weight
2. text-align
3. font-family
4. color

QCM 8 : Quelle propriété CSS permet de changer la casse du texte pour être tout en majuscules, minuscules, ou capitalisé ?

1. font-weight
2. text-transform
3. text-align
4. font-size

QCM 9 : Quelle propriété CSS permet d'ajouter une ombre au texte ?

1. text-align
2. text-shadow
3. font-family
4. font-size

QCM 10 : Quelle propriété CSS permet de changer la couleur du texte d'un élément ?

1. color
2. text-transform
3. font-size
4. font-style

III. Propriétés de couleur et de fond :

1. background-color

La propriété **background-color** définit la couleur de fond d'un élément. Elle peut être spécifiée à l'aide de noms de couleurs, de valeurs hexadécimales, RGB, ...

Exemple :

```
div {
  background-color: lightblue;
}
```

Dans cet exemple, l'élément aura un fond de couleur **bleu clair**.

2. background-image

La propriété **background-image** permet d'appliquer une image en tant qu'arrière-plan d'un élément. L'image peut être spécifiée par une URL.

Exemple :

```
body {
  background-image: url('background.jpg');
}
```

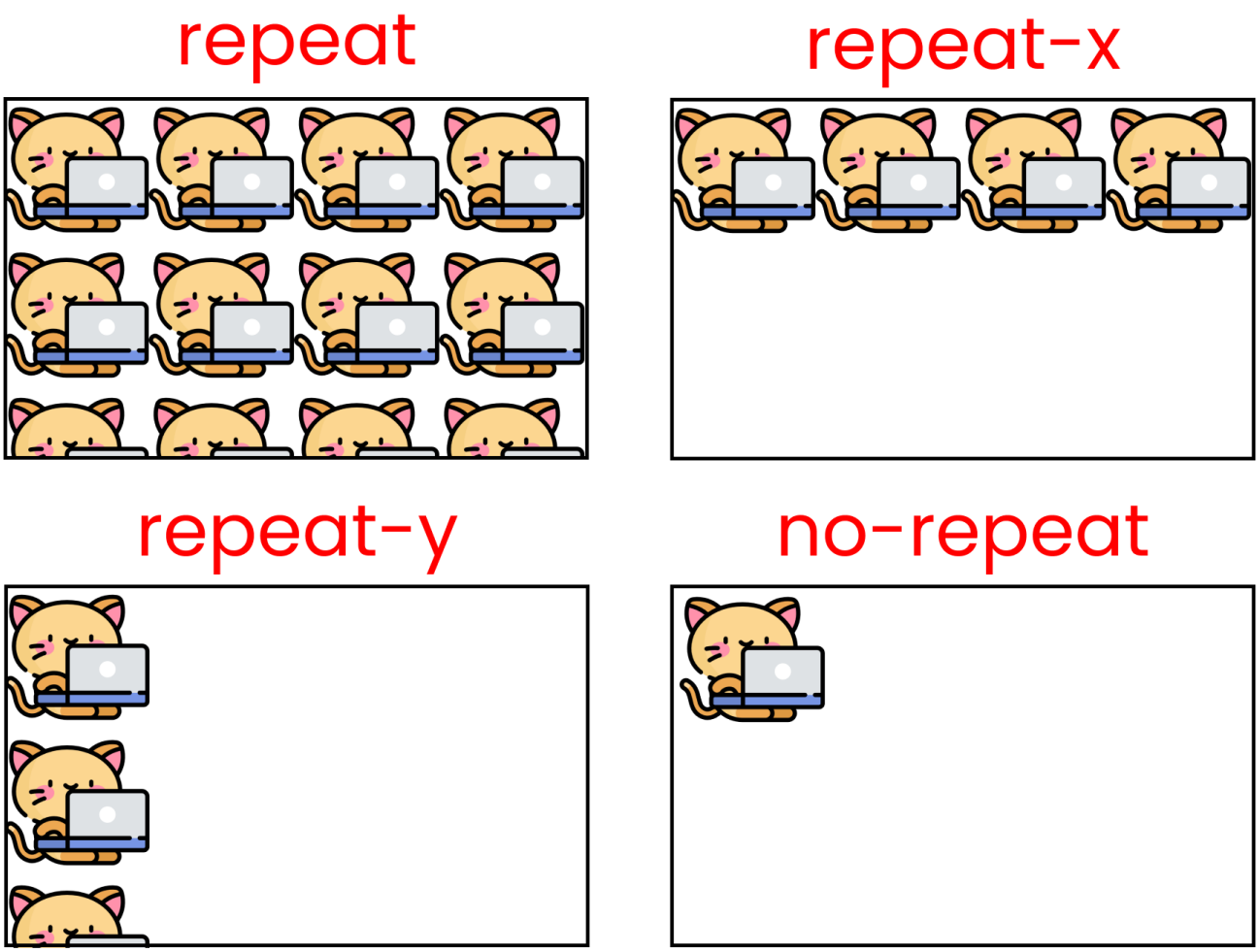
Dans cet exemple, l'image **background.jpg** sera utilisée comme arrière-plan de la page entière.

3. background-repeat :

La propriété **background-repeat** contrôle la répétition de l'image de fond. Par défaut, une image de fond se répète pour couvrir l'élément. Vous pouvez choisir de désactiver la répétition ou de répéter l'image horizontalement ou verticalement.

Valeurs :

- repeat (par défaut) : L'image de fond se répète dans les deux directions.
- repeat-x : L'image se répète uniquement horizontalement.
- repeat-y : L'image se répète uniquement verticalement.
- no-repeat : L'image ne se répète pas.



Exemple :

```
div {
  background-image: url('pattern.png');
  background-repeat: repeat-x;
}
```

Ici, l'image **pattern.png** se répète seulement horizontalement.

4. background-size :

La propriété **background-size** définit la taille de l'image d'arrière-plan. Vous pouvez définir des dimensions précises, utiliser des pourcentages, ou employer des mots-clés comme **cover** ou **contain**.

Valeurs :

- auto (par défaut) : La taille de l'image de fond reste à sa taille d'origine.
- cover : L'image couvre entièrement l'élément tout en gardant ses proportions.
- contain : L'image est redimensionnée pour être complètement visible à l'intérieur de l'élément.
- Valeurs spécifiques comme 100px 50px, pour des largeurs et hauteurs spécifiques.

Exemple :

```
section {
  background-image: url('large-image.jpg');
  background-size: cover;
}
```

Dans cet exemple, l'image **large-image.jpg** couvrira toute la section, sans déformation, mais une partie de l'image pourrait être coupée.

5. background

La propriété **background** est une propriété **raccourcie** qui permet de définir plusieurs propriétés de fond en une seule déclaration, comme background-color, background-image, background-repeat, background-position, et background-size.

Syntaxe :

```
background: [background-color] [background-image] [background-repeat]
[background-size];
```

Exemple :

```
div {
  background: lightblue url('background.jpg') no-repeat cover;
}
```

Dans cet exemple :

- La couleur de fond est **bleu clair**.
- L'image de fond est **background.jpg**.
- L'image ne se répète pas (**no-repeat**) et est centrée.
- L'image couvre entièrement l'élément (**cover**).

QCM :

QCM 1 : Quelle propriété CSS est utilisée pour définir la couleur de fond d'un élément ?

1. color
2. background-color
3. background-image
4. background

QCM 2 : Quelle propriété CSS permet d'ajouter une image en tant que fond ?

1. background-repeat
2. background-color
3. background-image
4. background-position

QCM 3 : Quelle valeur de la propriété background-repeat permet d'empêcher la répétition de l'image de fond ?

1. repeat
2. repeat-x
3. repeat-y
4. no-repeat

QCM 4 : Quelle propriété CSS permet de définir la taille de l'image de fond ?

1. background-position
2. background-size
3. background-repeat
4. background-image

QCM 5 : Quelle valeur de la propriété `background-size` fait en sorte que l'image de fond couvre toute la surface de l'élément, même si une partie de l'image peut être coupée ?

1. auto
2. contain
3. cover
4. repeat

QCM 6 : Quelle propriété CSS est une propriété raccourcie permettant de définir plusieurs aspects du fond en une seule ligne ?

1. background-image
2. background-repeat
3. background-color
4. background

QCM 7 : Quelle syntaxe est correcte pour définir un fond bleu clair avec une image qui ne se répète pas?

1. background: lightblue url('image.jpg') repeat;
2. background: url('image.jpg') lightblue no-repeat;
3. background: lightblue url('image.jpg') no-repeat;
4. background: no-repeat center lightblue url('image.jpg');

QCM 8 : Quelle valeur de la propriété background-repeat permet de répéter une image uniquement horizontalement ?

1. no-repeat
2. repeat
3. repeat-x
4. repeat-y

QCM 9 : Quelle propriété permet de spécifier la couleur, l'image, la répétition et la position du fond dans une seule déclaration ?

1. background-position
2. background
3. background-image
4. background-size

QCM 10 : Quelle est la valeur par défaut de la propriété background-size ?

1. cover
2. contain
3. auto
4. 100%

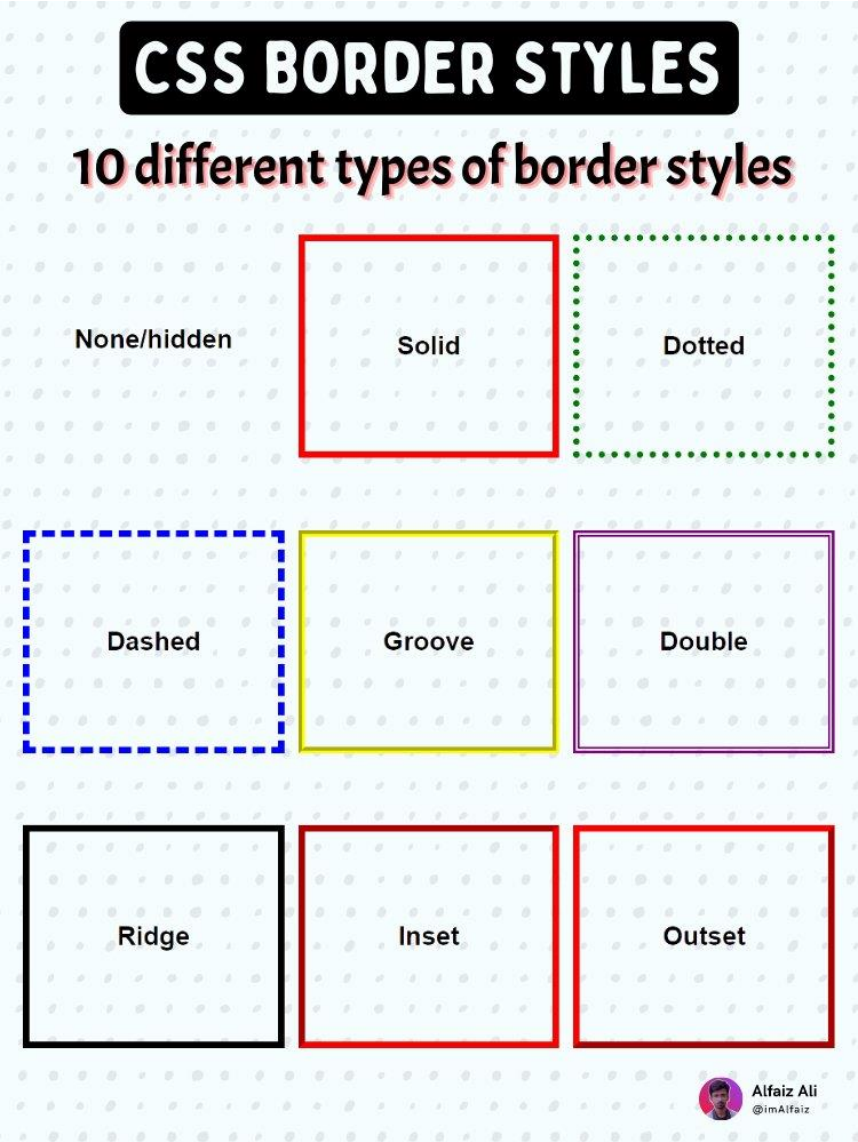
IV. Propriétés des bordures :

1. border-style

La propriété **border-style** définit le style de la bordure d'un élément. Elle peut prendre différentes valeurs pour spécifier l'apparence de la bordure (solide, en pointillés, en tirets, etc.).

Valeurs possibles :

- none : Pas de bordure.
- solid : Bordure pleine.
- dashed : Bordure en tirets.
- dotted : Bordure en pointillés.
- double : Bordure double ligne.
- groove, ridge, inset, outset : Effets en 3D.



Exemple :

```
div {
  border-style: solid;
}
```

Dans cet exemple, l'élément aura une bordure pleine.

2. border-color

La propriété **border-color** définit la couleur de la bordure. Elle peut être spécifiée en utilisant un nom de couleur, une valeur hexadécimale, RGB, ...

Exemple :

```
div {
  border-style: solid;
  border-color: red;
}
```

Dans cet exemple, la bordure de l'élément sera de couleur rouge.

3. border-width

La propriété **border-width** contrôle l'épaisseur de la bordure. Vous pouvez définir une épaisseur spécifique en pixels ou utiliser des valeurs prédéfinies telles que thin, medium, ou thick.

Exemple :

```
div {
  border-style: solid;
  border-width: 5px;
}
```

Ici, l'élément aura une bordure pleine de **5 pixels** d'épaisseur.

4. border-radius

La propriété **border-radius** permet de créer des coins arrondis pour les bordures. Vous pouvez définir un rayon spécifique pour les coins en pixels ou en pourcentage.

Exemple :

```
div {
  border-style: solid;
  border-width: 2px;
  border-radius: 20px;
}
```

Dans cet exemple, la bordure de l'élément aura des **coins arrondis** avec un rayon de **20 pixels**.

5. border

La propriété **border** est une **propriété raccourcie** qui permet de définir en une seule déclaration le style, la couleur et l'épaisseur de la bordure.

Exemple :

```
div {
  border: 3px solid blue;
}
```

Ici, l'élément aura une bordure :

- **épaisseur** : 3 pixels
- **style** : pleine (**solid**)
- **couleur** : bleue

QCM 1 : Quelle propriété CSS est utilisée pour définir le style de la bordure ?

1. border-width
2. border-style
3. border-radius
4. border-color

QCM 2 : Quelle est la bonne syntaxe pour définir une bordure pleine de 2 pixels de couleur rouge ?

1. border: red solid 2px;
2. border: 2px solid red;
3. border: solid red 2px;
4. border: 2px red solid;

QCM 3 : Quelle propriété permet d'arrondir les coins d'une bordure ?

1. border-width
2. border-radius
3. border-style
4. border-color

QCM 4 : Quelle propriété CSS est utilisée pour définir l'épaisseur de la bordure ?

1. border-style
2. border-radius
3. border-width
4. border-color

QCM 5 : Quelle est la propriété raccourcie qui permet de définir le style, la couleur et l'épaisseur d'une bordure en une seule ligne ?

1. border-style
2. border-color
3. border-width
4. border

V. Propriétés des listes

1. list-style-type

La propriété **list-style-type** définit le type de puce ou de numérotation à utiliser pour les éléments d'une liste. Elle peut être appliquée à des listes ordonnées (<ol>) et non ordonnées (<ul>).

Valeurs possibles :

- disc (par défaut pour <ul>) : un point plein.
- circle : un cercle vide.
- square : un carré plein.
- decimal (par défaut pour <ol>) : nombres 1, 2, 3, ...
- lower-alpha : a, b, c, ...
- upper-alpha : A, B, C, ...
- none : aucun style de puce ou de numérotation.
- ...

armenian	circle	decimal	georgian	decimal-leading-zero	lower-alpha	lower-greek	lower-roman	square	upper-alpha	upper-roman
ա.	◦	1.	ა.	01.	a.	α.	i.	■	A.	I.
բ.	◦	2.	ბ.	02.	b.	β.	ii.	■	B.	II.
գ.	◦	3.	გ.	03.	c.	γ.	iii.	■	C.	III.
դ.	◦	4.	დ.	04.	d.	δ.	iv.	■	D.	IV.
ե.	◦	5.	ე.	05.	e.	ε.	v.	■	E.	V.

Exemple :

```
ul {
  list-style-type: square;
}

ol {
  list-style-type: upper-alpha;
}
```

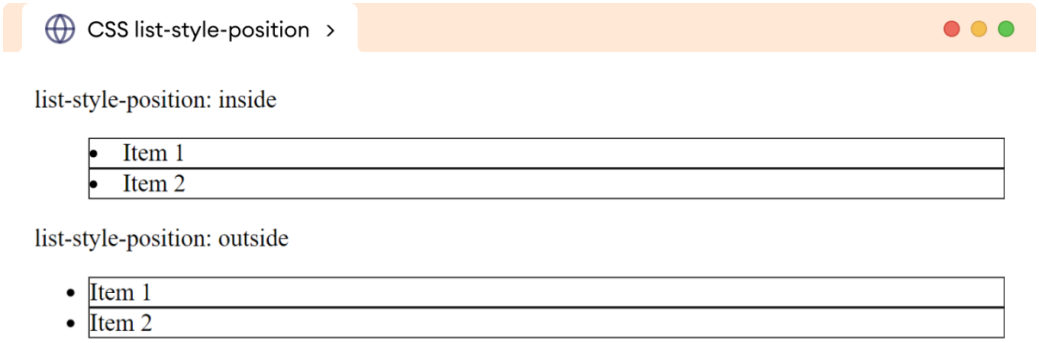
Dans cet exemple, les éléments de la liste <ul> auront des puces carrées, tandis que la liste <ol> utilisera des lettres majuscules pour la numérotation.

2. list-style-position

La propriété **list-style-position** détermine la position de la puce ou du numéro par rapport au contenu de l'élément de la liste.

Valeurs possibles :

- inside : la puce ou le numéro est à l'intérieur du bloc, donc aligné avec le texte de l'élément.
- outside (par défaut) : la puce ou le numéro est à l'extérieur du bloc, décalé à gauche du texte.



Exemple :

```
ul {
  list-style-position: inside;
}
```

Dans cet exemple, les puces des éléments de la liste seront alignées à l'intérieur du bloc de texte.

3. list-style-image

La propriété **list-style-image** permet d'utiliser une image en tant que puce dans une liste.

Exemple :

```
ul {
  list-style-image: url('custom-bullet.png');
}
```

Dans cet exemple, l'image **custom-bullet.png** sera utilisée comme puce pour les éléments de la liste.

4. list-style

La propriété **list-style** est une propriété **raccourcie** qui permet de définir simultanément **list-style-type**, **list-style-position**, et **list-style-image** en une seule déclaration.

Exemple :

```
ul {
  list-style: square inside url('custom-bullet.png');
}
```

Dans cet exemple, la liste utilise une image **custom-bullet.png**, avec un style de puce carré comme secours, et les puces sont alignées à l'intérieur du texte.

QCM :

QCM 1 : Quelle propriété est utilisée pour changer le type de puce dans une liste non ordonnée (<ul>) ?

1. list-style-position
2. list-style-image
3. list-style-type
4. list-style

QCM 2 : Quelle valeur de la propriété list-style-type permet de supprimer les puces ?

1. none
2. circle
3. square
4. decimal

QCM 3 : Quelle propriété permet d'utiliser une image en tant que puce dans une liste ?

1. list-style-type
2. list-style-position
3. list-style-image
4. list-style

QCM 4 : Quelle valeur de la propriété list-style-position place la puce à l'intérieur du texte de l'élément de liste ?

1. outside
2. inside
3. left
4. center

VI. Propriétés des tableaux

1. table-layout

La propriété **table-layout** détermine la manière dont la largeur des colonnes d'un tableau est calculée. Elle influence la rapidité du rendu du tableau et la manière dont les colonnes sont redimensionnées.

Valeurs possibles :

- **auto** (valeur par défaut) : La largeur des colonnes est calculée en fonction de leur contenu. Si une colonne contient beaucoup de texte ou de grandes images, elle sera élargie.
- **fixed** : La largeur des colonnes est déterminée par la largeur explicite définie (ou la largeur du tableau si aucune largeur n'est spécifiée pour les colonnes). Cela permet au tableau d'être plus rapidement rendu, car la largeur est calculée une seule fois.

auto

Name	Location
Lion	Africa
Norwegian Lemming	Europe
Seal	Antarctica
Tiger	Asia

fixed

Name	Location
Lion	Africa
Norwegian Lemming	Europe
Seal	Antarctica
Tiger	Asia

Exemple :

```
table {
  table-layout: fixed;
  width: 100%;
}
```

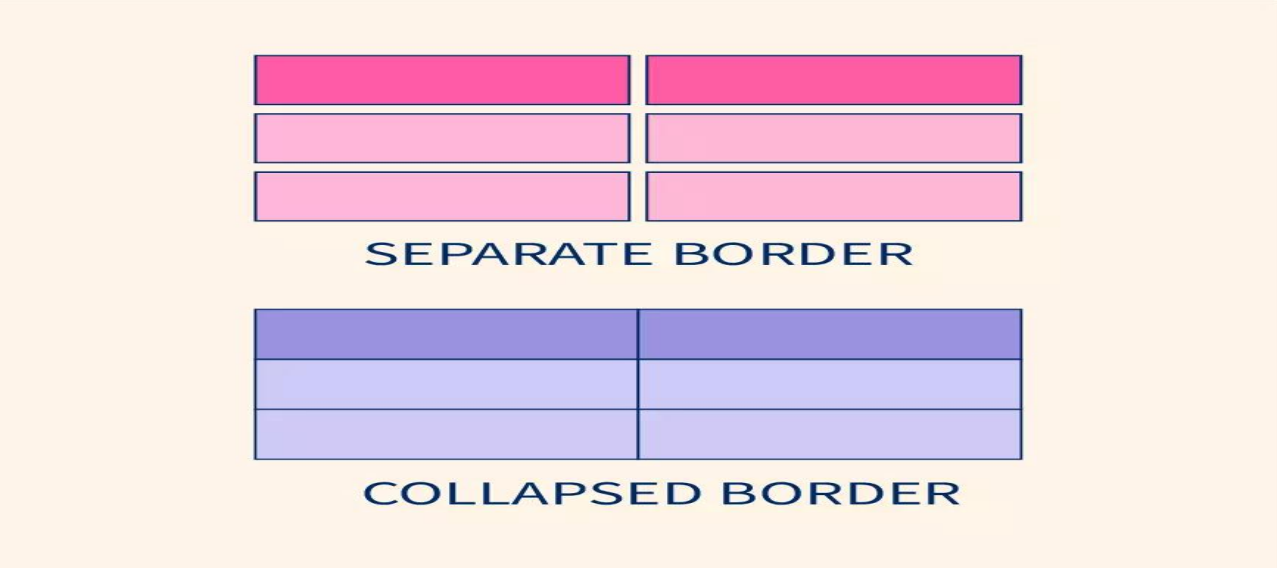
Dans cet exemple, la largeur des colonnes du tableau est fixe, ce qui signifie que chaque colonne prendra une proportion égale de l'espace total du tableau (si aucune autre règle de largeur n'est définie).

2. border-collapse

La propriété **border-collapse** détermine si les bordures des cellules adjacentes dans un tableau sont fusionnées en une seule bordure ou séparées.

Valeurs possibles :

- **separate** (valeur par défaut) : Les bordures des cellules sont séparées, et l'espacement entre elles peut être contrôlé avec la propriété **border-spacing**.
- **collapse** : Les bordures adjacentes des cellules fusionnent en une seule bordure. Cela donne une apparence plus propre au tableau avec des bordures continues.



Exemple :

```
table {
  border-collapse: collapse;
  width: 100%;
}
td, th {
  border: 1px solid black;
}
```

Dans cet exemple, les bordures des cellules sont fusionnées en une seule bordure de **1 pixel** d'épaisseur autour du tableau.

QCM 1 : Quelle est la valeur par défaut de la propriété table-layout ?

1. auto
2. fixed
3. collapse
4. separate

QCM 2 : Quelle propriété CSS contrôle la manière dont la largeur des colonnes est calculée dans un tableau ?

1. border-collapse
2. table-layout
3. border-spacing
4. width

QCM 3 : Quelle valeur de border-collapse fusionne les bordures adjacentes des cellules en une seule ?

1. separate
2. auto
3. collapse
4. fixed

QCM 4 : Quelle propriété permet de séparer ou fusionner les bordures des cellules d'un tableau ?

1. border-spacing
2. table-layout
3. border-collapse
4. width

VII. Propriétés des images :

La propriété **filter** en CSS permet d'appliquer des effets graphiques (comme un flou ou un changement de couleurs) aux éléments tels que des images, du texte, ou même des blocs. C'est un moyen simple de modifier l'apparence visuelle d'un élément sans avoir à modifier l'image ou l'élément lui-même.

1. invert()

La fonction **invert()** inverse les couleurs d'un élément. Cela signifie que les couleurs claires deviennent sombres et les couleurs sombres deviennent claires.

- La valeur est un pourcentage entre **0%** (aucune inversion) et **100%** (couleurs complètement inversées).

Exemple :

```
img {
  filter: invert(100%);
}
```

Dans cet exemple, l'image aura ses couleurs complètement inversées.

2. grayscale()

La fonction **grayscale()** applique un effet de désaturation, transformant une image en niveaux de gris.

- La valeur est un pourcentage entre **0%** (pas de changement) et **100%** (image entièrement en niveaux de gris).

Exemple :

```
img {
  filter: grayscale(100%);
}
```

Ici, l'image sera entièrement en noir et blanc.

3. blur()

La fonction **blur()** applique un flou à l'élément. La valeur spécifie le rayon du flou, exprimé en pixels.

Exemple :

```
img {
  filter: blur(5px);
}
```

Dans cet exemple, l'image aura un effet de flou avec un rayon de 5 pixels.

QCM 1 : Quelle fonction CSS est utilisée pour transformer une image en noir et blanc ?

1. blur()
2. invert()
3. grayscale()
4. brightness()

QCM 2 : Quelle est la valeur maximale que vous pouvez utiliser avec invert() pour inverser complètement les couleurs d'un élément ?

1. 50%
2. 100%
3. 200%
4. 0%

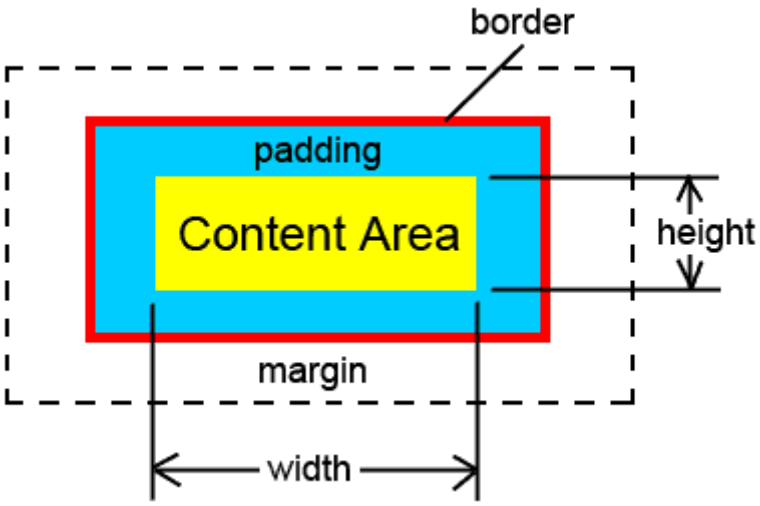
QCM 3 : Quelle fonction CSS applique un effet de flou sur un élément ?

1. blur()
2. invert()
3. grayscale()
4. sepia()

QCM 4 : Comment appliquer un flou de 5 pixels à une image en CSS ?

1. filter: blur(5%);
2. filter: blur(5px);
3. filter: blur(5em);
4. filter: blur(5pt);

VIII. Propriétés des boîtes :



Les propriétés CSS des boîtes permettent de gérer la taille, la position, les marges, les bordures, le remplissage et d'autres aspects des éléments HTML. Voici une liste de ces propriétés avec des exemples :

1. width et height

- **width** : définit la largeur d'un élément.
- **height** : définit la hauteur d'un élément.

Exemple :

```
div {
  width: 200px;
  height: 150px;
}
```

Dans cet exemple, la boîte aura une largeur de **200 pixels** et une hauteur de **150 pixels**.

2. margin

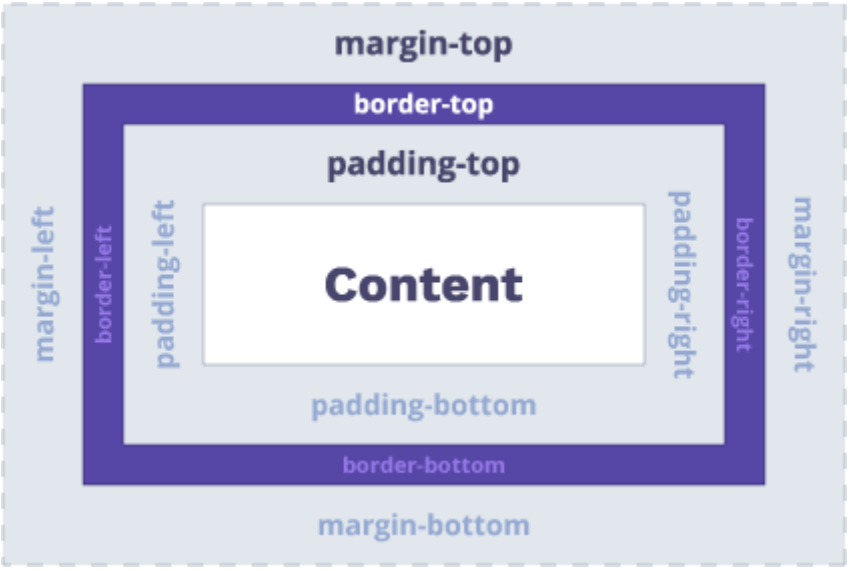
La propriété **margin** ajoute de l'espace autour d'un élément. Elle peut être définie pour les quatre côtés individuellement ou globalement.

Exemple :

```
div {
  margin: 20px; /* marge de 20px sur tous les côtés */
}

div.special {
  margin-top: 10px;
  margin-right: 15px;
  margin-bottom: 20px;
  margin-left: 25px;
}
```

3. padding



La propriété **padding** ajoute de l'espace à l'intérieur de l'élément, entre son contenu et sa bordure.

Exemple :

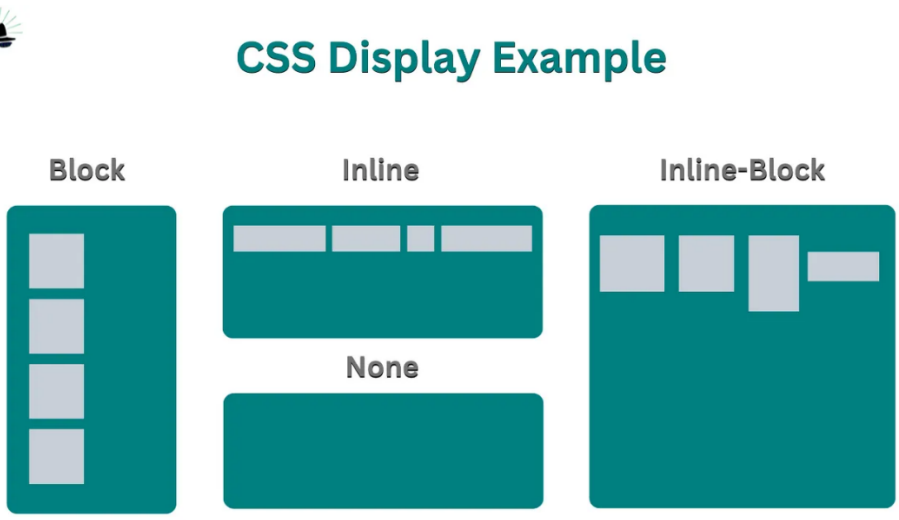
```
div {
  padding: 15px; /* 15px de padding pour tous les côtés */
}
```

4. display

La propriété **display** contrôle la manière dont un élément est rendu dans le flux du document.

Valeurs communes :

- **block** : l'élément prend toute la largeur disponible.
- **inline** : l'élément ne prend que la largeur nécessaire à son contenu.
- **inline-block** : combine des caractéristiques de block et inline.
- **none** : l'élément est supprimé du flux visuel.

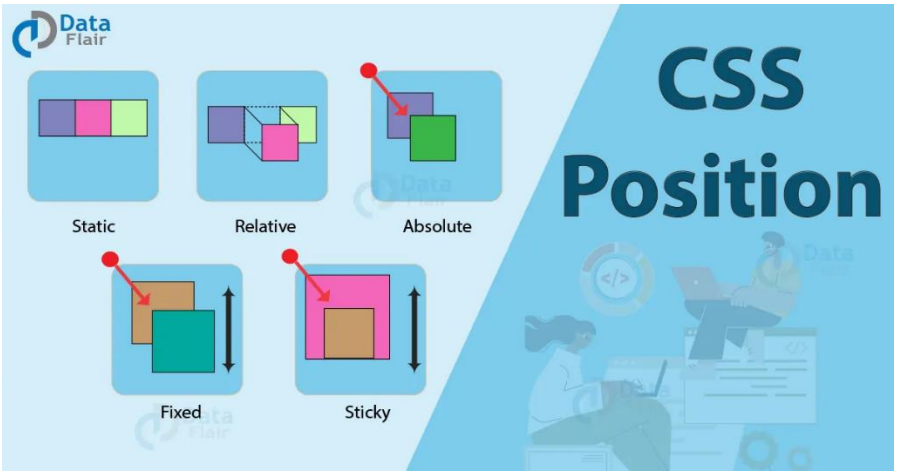


Exemple :

```
div {
  display: block;
}
span {
  display: inline;
}
```

5. position (avec top, bottom, left, right)

- La propriété **position** définit comment un élément est positionné dans le document.



Valeurs communes :

- **static** (par défaut) : l'élément suit le flux normal du document.
- **relative** : l'élément est positionné par rapport à sa position normale.
- **absolute** : l'élément est retiré du flux et positionné par rapport à son conteneur positionné le plus proche.
- **fixed** : l'élément est positionné par rapport à la fenêtre et ne bouge pas lors du défilement.

- **sticky** :permet à un élément de rester positionné normalement dans la page, mais de devenir fixe (collant) lorsqu'on fait défiler la page et qu'il atteint une certaine position définie

Exemple :

```
div {
  position: absolute;
  top: 50px;
  left: 100px;
}
```

6. overflow

- La propriété **overflow** contrôle comment le contenu débordant d'un élément est géré.

Valeurs :

- **visible** (par défaut) : le contenu déborde de la boîte.
- **hidden** : le contenu qui dépasse est masqué.
- **scroll** : des barres de défilement sont ajoutées si nécessaire.
- **auto** : des barres de défilement apparaissent seulement si le contenu déborde.
- **clip**: semble fonctionner exactement comme overflow: hidden. La différence est que vous pouvez utiliser JavaScript pour faire défiler quelque chose avec overflow: hidden, mais pas si l'élément a overflow: clip.

Exemple :

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

7. box-shadow

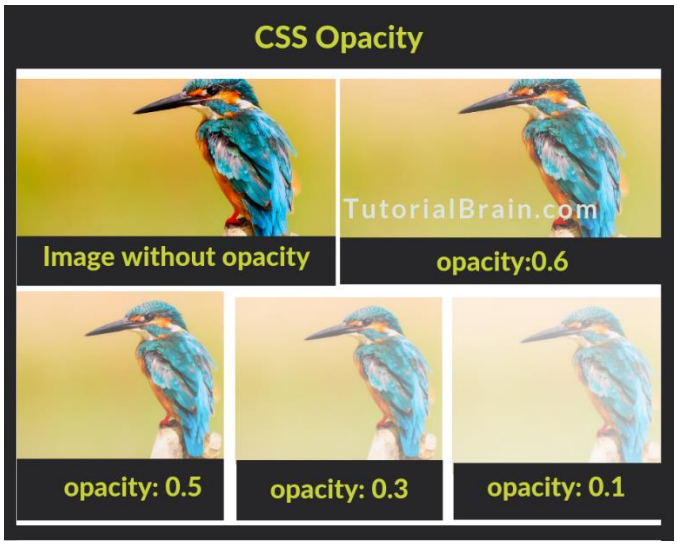
La propriété **box-shadow** ajoute une ombre autour d'un élément.

Exemple :

```
div {
  box-shadow: 10px 10px 5px red;
}
```

Dans cet exemple, l'ombre rouge a un décalage horizontal de 10px, un décalage vertical de 10px, et un flou de 5px avec une opacité de 50%.

8. Opacity



La propriété **opacity** définit la transparence d'un élément. La valeur varie de **0** (entièrement transparent) à **1** (entièrement opaque).

Exemple :

```
div {
  opacity: 0.5; /* 50% de transparence */
}
```

QCM sur les propriétés des boîtes en CSS

QCM 1 : Quelle propriété CSS est utilisée pour définir la largeur d'un élément ?

1. width
2. height
3. padding
4. display

QCM 2 : Quelle propriété contrôle l'espace entre le contenu de l'élément et sa bordure ?

1. margin
2. border
3. padding
4. width

QCM 3 : Quelle valeur de la propriété display supprime un élément du flux visuel sans le rendre invisible ?

1. block
2. inline
3. none
4. inline-block

QCM 4 : Quelle propriété permet de gérer ce qu'il se passe lorsque le contenu dépasse la taille définie d'une boîte ?

1. padding
2. margin
3. display
4. overflow

QCM 5 : Quelle propriété CSS définit la transparence d’un élément ?

1. opacity
2. box-shadow
3. visibility
4. background-color

QCM 6 : Quelle valeur de la propriété position permet à un élément d'être fixé à une position donnée, même pendant le défilement ?

1. static
2. absolute
3. relative
4. fixed

QCM 7 : Que fait la propriété box-shadow ?

1. Change la couleur de fond.
2. Ajoute une ombre autour de l'élément.
3. Modifie la transparence de l'élément.
4. Ajoute une bordure.

QCM 8 : Quelle propriété contrôle l'espace à l'extérieur d'un élément (entre cet élément et les autres) ?

1. padding
2. margin
3. width
4. border

QCM 9 : Quelle propriété permet de définir une ombre autour d'une boîte en CSS ?

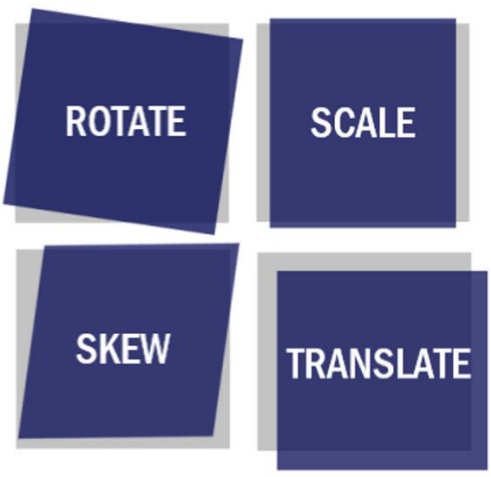
1. box-shadow
2. text-shadow
3. border
4. opacity

QCM 10 : Quelle propriété CSS est utilisée pour définir la hauteur d'un élément ?

1. width
2. padding
3. height
4. display

IX. Transformation

Les **transformations CSS** permettent de modifier l'apparence des éléments en les faisant pivoter, déplacer, redimensionner ou déformer. Les fonctions principales utilisées pour ces transformations sont : **rotate, skew, scale,** et **translate.** Voici les définitions et exemples pour chacune d'elles.



1. rotate() : Rotation d'un élément

La fonction **rotate()** permet de faire pivoter un élément autour de son point d'origine. La valeur est exprimée en degrés (**deg**).

Exemple :

```
div {
  transform: rotate(45deg);
}
```

Dans cet exemple, l'élément est tourné de **45 degrés** dans le sens des aiguilles d'une montre.

2. skew() : Inclinaison d'un élément

La fonction **skew()** permet d'incliner un élément le long de l'axe X ou Y. Les valeurs sont exprimées en degrés.

Exemple :

```
div {
  transform: skew(20deg, 10deg);
}
```

Cet exemple inclut une inclinaison de **20 degrés** sur l'axe X et de **10 degrés** sur l'axe Y.

3. scale() : Redimensionnement d'un élément

La fonction **scale()** modifie la taille d'un élément en fonction des facteurs de redimensionnement donnés. Un facteur de **1** signifie que l'élément conserve sa taille d'origine.

Exemple :

```
div {
  transform: scale(1.5);
}
```

Dans cet exemple, l'élément est agrandi de **1.5 fois** (150% de sa taille d'origine).

4. translate() : Déplacement d'un élément

La fonction **translate()** déplace un élément sur l'axe X ou Y. Les valeurs sont exprimées en pixels (**px**) ou en pourcentages (**%**).

Exemple :

```
div {
  transform: translate(50px, 100px);
}
```

Cet exemple déplace l'élément de **50 pixels** vers la droite et de **100 pixels** vers le bas.

QCM 1 : Quelle fonction CSS permet de faire pivoter un élément ?

1. translate()
2. scale()
3. rotate()
4. skew()

QCM 2 : Quelle valeur de scale(2) va-t-elle produire sur un élément ?

1. L'élément sera deux fois plus petit.
2. L'élément sera deux fois plus large.
3. L'élément sera deux fois plus grand.
4. L'élément sera deux fois plus haut.

QCM 3 : Quelle fonction CSS est utilisée pour déplacer un élément horizontalement et verticalement ?

1. rotate()
2. translate()
3. scale()
4. skew()

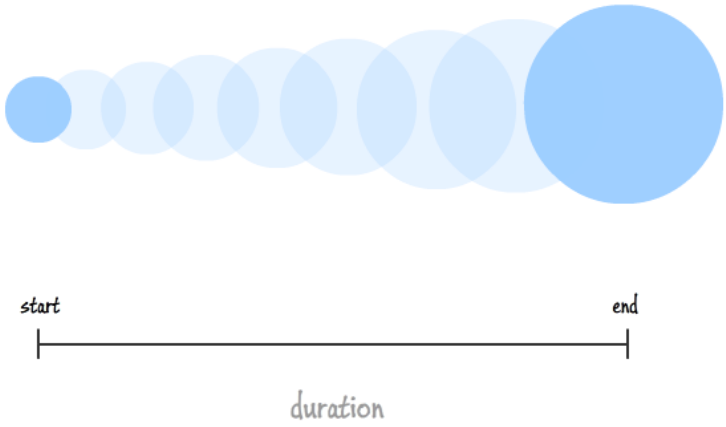
QCM 4 : Quelle fonction CSS inclinerait un élément selon les axes X et Y ?

1. rotate()
2. translate()
3. skew()
4. scale()

QCM 5 : Quel est l'effet de transform: translate(50px, 100px); sur un élément ?

1. L'élément est déplacé de 50 pixels vers le haut et 100 pixels vers la gauche.
2. L'élément est déplacé de 50 pixels vers la gauche et 100 pixels vers le bas.
3. L'élément est déplacé de 50 pixels vers la droite et 100 pixels vers le bas.
4. L'élément est déplacé de 50 pixels vers le bas et 100 pixels vers la droite.

X. Les transitions



Les **transitions en CSS** permettent d'ajouter des animations simples et fluides lorsque des propriétés d'un élément changent d'état, par exemple lorsqu'un élément est survolé, cliqué, ou lorsqu'une classe lui est ajoutée. Les transitions facilitent le passage progressif d'une propriété CSS à une autre.

Il existe plusieurs propriétés spécifiques pour contrôler les transitions : **transition**, **transition-property**, **transition-duration**, et **transition-delay**.

1. transition-property : Propriété(s) à animer

transition-property spécifie la ou les propriétés CSS à animer. On peut animer plusieurs propriétés séparées par des virgules.

Exemple :

```
div {
  width: 100px;
  height: 100px;
  background-color: skyblue;
  transition-property: background-color, width;
  transition-duration: 1s;
}

div:hover {
  background-color: coral;
  width: 200px;
}
```

Explication : Ici, seules les propriétés **background-color** et **width** sont animées.

2. transition-duration : Durée de la transition

La propriété **transition-duration** définit la durée pendant laquelle la transition s'effectue. La valeur est exprimée en secondes (**s**) ou millisecondes (**ms**).

Exemple :

```
div {
  width: 100px;
  height: 100px;
  background-color: skyblue;
  transition-property: background-color, width;
  transition-duration: 2s, 1s; }
div:hover {
  background-color: coral;
  width: 200px;
}
```

Explication : La couleur change sur **2 secondes**, tandis que la largeur change sur **1 seconde**.

3. transition-delay : Délai avant le début de la transition

transition-delay spécifie le temps à attendre avant de démarrer la transition. La valeur est exprimée en secondes (**s**) ou millisecondes (**ms**).

Exemple :

```
div {
  width: 100px;
  height: 100px;
  background-color: skyblue;
  transition-property: background-color, width;
  transition-duration: 1s;
  transition-delay: 0.5s, 1s; /* Délai de 0,5 seconde pour la couleur et
de 1 seconde pour la largeur */
}

div:hover {
  background-color: coral;
  width: 200px;
}
```

Explication : Il y a un délai de **0.5 seconde** avant que la couleur change, et un délai de **1 seconde** avant que la largeur ne commence à s'agrandir.

4. transition : Propriété raccourcie

La propriété **transition** est une propriété raccourcie qui permet de définir toutes les autres propriétés de transition en une seule déclaration : la propriété à animer, la durée de l'animation, le moment où elle commence et le type de courbe (fonction de synchronisation).

Syntaxe :

```
transition: [transition-property] [transition-duration] [transition-timing-function] [transition-delay];
```

Exemple :

```
div {
  width: 100px;
  height: 100px;
  background-color: skyblue;
  transition: background-color 0.5s ease-in-out, width 1s;
}

div:hover {
  background-color: coral;
  width: 200px;
}
```

Explication : Lorsque l'élément est survolé, sa couleur de fond change progressivement sur **0.5 seconde** et sa largeur s'agrandit sur **1 seconde**.

QCM :

QCM 1 : Quelle propriété CSS permet de définir toutes les propriétés de transition en une seule ligne ?

1. transition-duration
2. transition-property
3. transition
4. transition-delay

QCM 2 : Si vous souhaitez qu'une transition démarre après 2 secondes, quelle propriété devez-vous utiliser ?

1. transition-duration
2. transition-property
3. transition-delay
4. transition-timing-function

QCM 3 : Quelle propriété CSS est utilisée pour spécifier quelles propriétés CSS doivent être animées ?

1. transition-duration
2. transition-property
3. transition-delay
4. transition-timing-function

QCM 4 : Que fait la règle suivante ?

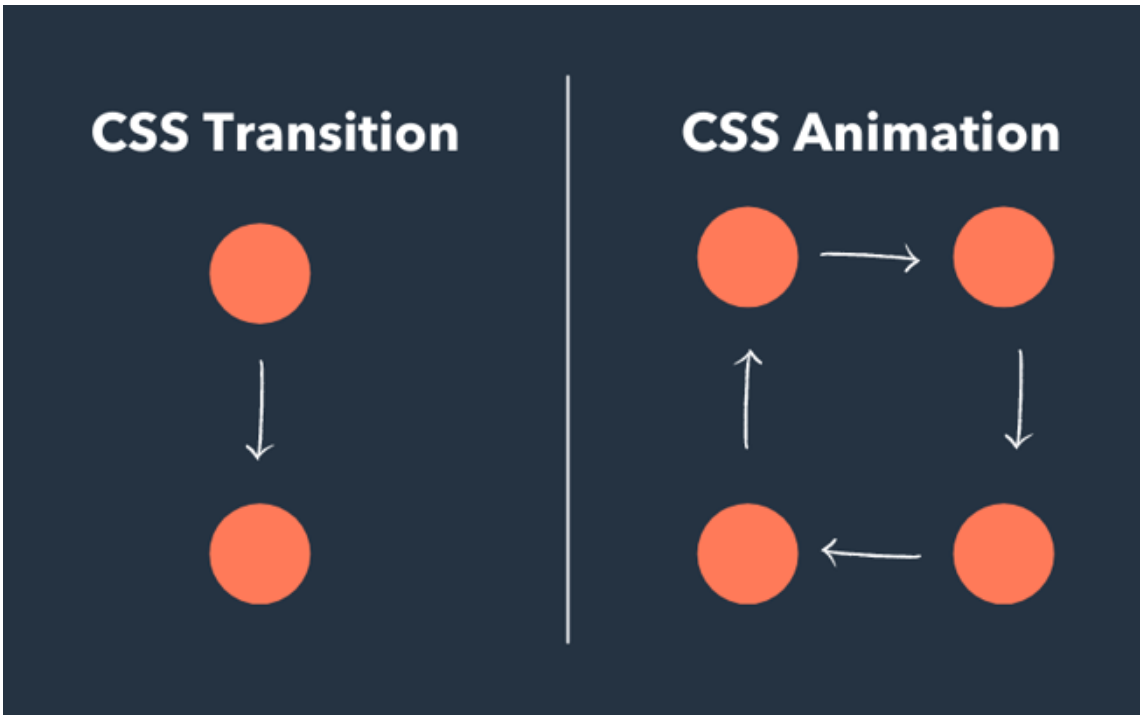
```
div {
  transition-duration: 3s;
}
```

1. Définit un délai de 3 secondes avant le démarrage de la transition.
2. Définit la durée de la transition sur 3 secondes.
3. Change la propriété background-color en 3 secondes.
4. Aucun effet.

QCM 5 : Quelle est la fonction de la propriété transition-delay ?

1. Spécifie les propriétés à animer.
2. Définit combien de temps une transition dure.
3. Spécifie combien de temps attendre avant que la transition commence.
4. Détermine la courbe de la transition.

XI. Les animations



Les **animations en CSS** permettent de créer des mouvements complexes et dynamiques en spécifiant une série d'étapes à travers les valeurs des propriétés CSS. Contrairement aux transitions

qui ne se produisent que lorsque l'état d'un élément change (comme lors d'un survol), les animations peuvent être exécutées en continu ou selon un cycle défini.

Voici une description détaillée des principales propriétés d'animations CSS :

1. @keyframes : Définir les étapes de l'animation

La règle **@keyframes** permet de définir les étapes d'une animation, c'est-à-dire comment une propriété doit évoluer au fil du temps. Chaque étape correspond à un pourcentage ou à des mots-clés (from, to).

Syntaxe :

```
@keyframes animation-name {
  0% {
    /* état initial */
  }
  100% {
    /* état final */
  }
}
```

Exemple :

```
@keyframes slide {
  0% {
    transform: translateX(0);
  }
  100% {
    transform: translateX(200px);
  }
}
```

Explication : L'élément se déplacera de **0px à 200px** sur l'axe X.

2. animation-name : Nom de l'animation

La propriété **animation-name** associe une animation définie par **@keyframes** à un élément.

Exemple :

```
div {
  animation-name: slide;
  animation-duration: 2s;
}
```

Explication : L'animation nommée **slide** est appliquée à l'élément, et dure **2 secondes**.

3. animation-duration : Durée de l'animation

La propriété **animation-duration** spécifie la durée totale de l'animation. Elle peut être définie en secondes (**s**) ou millisecondes (**ms**).

Exemple :

```
div {
  animation-name: slide;
  animation-duration: 2s;
}
```

Explication : L'animation dure **2 secondes**.

4. animation-delay : Délai avant le début de l'animation

animation-delay définit le temps à attendre avant que l'animation commence. Elle peut être définie en secondes (**s**) ou millisecondes (**ms**).

Exemple :

```
div {
  animation-name: slide;
  animation-duration: 2s;
  animation-delay: 1s;
}
```

Explication : Il y a un délai de **1 seconde** avant que l'animation ne commence.

5. animation-iteration-count : Nombre de répétitions

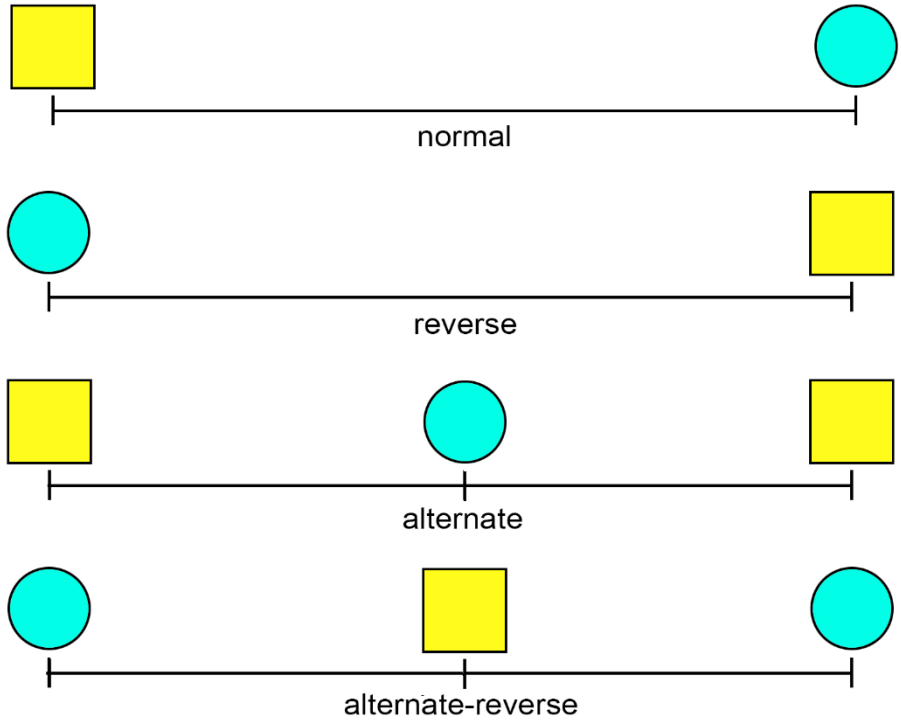
La propriété **animation-iteration-count** définit combien de fois l'animation doit être répétée. Elle peut être un nombre ou la valeur spéciale **infinite** pour des répétitions infinies.

Exemple :

```
div {
  animation-name: slide;
  animation-duration: 2s;
  animation-iteration-count: 3;
}
```

Explication : L'animation se répétera **3 fois**.

6. animation-direction : Direction de l'animation



La propriété **animation-direction** spécifie si l'animation doit se jouer dans le sens normal, inverse ou en alternant. Les valeurs possibles sont :

- normal : L'animation se déroule de manière normale (par défaut).
- reverse : L'animation se joue à l'envers.
- alternate : L'animation alterne entre sens normal et sens inverse à chaque cycle.
- alternate-reverse : Alterne, en commençant par le sens inverse.

Exemple :

```
div {
  animation-name: slide;
  animation-duration: 2s;
  animation-iteration-count: infinite;
  animation-direction: alternate;
}
```

Explication : L'animation se joue d'abord en avant, puis en arrière, indéfiniment.

7. animation : Propriété raccourcie

La propriété **animation** est une propriété raccourcie pour définir toutes les propriétés d'animation en une seule déclaration.

Syntaxe :

```
animation: [animation-name] [animation-duration] [animation-delay]
[animation-iteration-count] [animation-direction];
```

Exemple :

```
div {
  animation: slide 2s 1s 3 alternate;
}
```

Explication : L'animation nommée **slide** dure **2 secondes**, commence après un délai de **1 seconde**, se joue **3 fois** en alternant les directions.

QCM :

QCM 1 : Que permet de définir la règle @keyframes ?

1. Le nom de l'animation.
2. Les étapes d'une animation.
3. La durée de l'animation.
4. Le nombre de répétitions de l'animation.

QCM 2 : Quelle propriété associe un nom d'animation à un élément ?

1. animation-name
2. animation-duration
3. @keyframes
4. animation-fill-mode

QCM 3 : Quelle propriété CSS permet de spécifier combien de temps une animation doit durer ?

1. animation-iteration-count
2. animation-delay
3. animation-duration
4. animation-direction

QCM 4 : Que fait la propriété animation-delay ?

1. Définit la durée de l'animation.
2. Spécifie le temps avant que l'animation ne commence.
3. Contrôle la direction de l'animation.
4. Indique combien de fois l'animation doit être répétée.

QCM 5 : Quelle valeur permet de jouer une animation de manière infinie ?

1. none
2. once
3. infinite
4. forever

QCM 6 : Que permet de contrôler la propriété `animation-direction` ?

1. Le sens de lecture de l'animation.
2. La durée de l'animation.
3. Le moment où l'animation commence.
4. La propriété CSS qui doit être animée.

QCM 7 : Quelle propriété détermine comment la vitesse de l'animation évolue au fil du temps ?

1. animation-fill-mode
2. animation-direction
3. animation-play-state
4. animation-timing-function

QCM 8 : Quelle est la propriété raccourcie permettant de définir plusieurs propriétés d'animation en une seule ligne ?

1. animation
2. @keyframes
3. animation-timing-function
4. animation-iteration-count