Les principales commandes CSS

1. Sélecteurs :

Les sélecteurs permettent de cibler des éléments HTML pour leur appliquer des styles.

- **Sélecteurs de base** :
  - * : sélectionne tous les éléments.
  - element : sélectionne un élément spécifique (ex. div, p).
  - #id : sélectionne un élément avec un id spécifique (ex. #header).
  - .class : sélectionne les éléments avec une classe spécifique (ex. .menu).
  - element.class : sélectionne un élément avec une classe spécifique (ex. div.menu).
  - [attribute] : sélectionne les éléments avec un attribut spécifique (ex. [type="text"]).
- **Sélecteurs combinés** :
  - element, element : applique le même style à plusieurs éléments.
  - element element : sélectionne un élément à l'intérieur d'un autre (descendant).
  - element > element : sélectionne un élément enfant direct.
  - element + element : sélectionne un élément suivant un autre élément.
  - element ~ element : sélectionne un élément qui suit un autre élément dans le même parent.

2. Propriétés de mise en page :

Ces propriétés permettent de contrôler la structure et l'alignement des éléments.

- **Positionnement** :
  - position : définit le mode de positionnement (static, relative, absolute, fixed, sticky).
  - top, right, bottom, left : positionne un élément selon sa position.
  - z-index : définit la superposition des éléments.
- **Affichage** :
  - display : définit le type d'affichage (block, inline, inline-block, flex, grid, none).
  - visibility : définit la visibilité d'un élément (visible, hidden, collapse).
- **Boîte de modèle (Box model)** :
  - width, height : définit la largeur et la hauteur d'un élément.
  - margin : espace extérieur d'un élément.
  - padding : espace intérieur d'un élément.
  - border : définit la bordure d'un élément (style, largeur, couleur).
  - box-sizing : définit le calcul de la taille de la boîte (content-box, border-box).
- **Flexbox** :
  - display: flex : active le mode flexbox.
  - flex-direction : définit l'orientation des éléments (row, column, row-reverse, column-reverse).
  - justify-content : aligne les éléments horizontalement (centre, espace entre, etc.).
  - align-items : aligne les éléments verticalement.
  - align-self : permet de définir l'alignement individuel d'un élément dans un conteneur flex.
- **Grid** :
  - display: grid : active le mode grid.
  - grid-template-columns, grid-template-rows : définit les colonnes et les rangées.
  - grid-gap : espace entre les éléments du grid.
  - grid-column, grid-row : définit où un élément doit apparaître dans le grid.

3. Propriétés de texte :

Permet de contrôler la typographie et l'apparence du texte.

- font-family : définit la police de caractère.
- font-size : définit la taille de la police.
- font-weight : définit l'épaisseur du texte (normal, bold, lighter, ou valeurs numériques).
- font-style : définit le style de la police (normal, italic, oblique).
- text-align : définit l'alignement du texte (left, center, right, justify).
- line-height : définit l'espacement entre les lignes de texte.
- letter-spacing : définit l'espacement entre les caractères.
- text-transform : définit la transformation du texte (uppercase, lowercase, capitalize).
- text-decoration : définit les décorations de texte (underline, line-through, none).
- text-shadow : applique une ombre au texte.

4. Couleurs et arrière-plan :

Contrôle la couleur des éléments et de leurs arrière-plans.

- color : définit la couleur du texte.
- background-color : définit la couleur d'arrière-plan.
- background-image : définit une image d'arrière-plan.
- background-position : définit la position de l'image d'arrière-plan.
- background-size : définit la taille de l'image d'arrière-plan.
- background-repeat : définit la répétition de l'image d'arrière-plan.

5. Effets de transition et animation :

Ajoute des animations et transitions à des éléments.

- **Transitions** :
  - transition : définit la durée, le type de transition et la propriété à animer.
  - transition-duration : définit la durée de la transition.
  - transition-timing-function : définit la courbe de vitesse de la transition (ex. ease, linear).
  - transition-delay : définit un délai avant de commencer la transition.
- **Animations** :
  - @keyframes : définit une animation avec des étapes (keyframes).
  - animation : applique une animation à un élément.
  - animation-name : spécifie le nom de l'animation.
  - animation-duration : définit la durée de l'animation.
  - animation-timing-function : définit la courbe de vitesse de l'animation.
  - animation-delay : définit un délai avant de commencer l'animation.

6. Propriétés d'affichage avancé :

Ces propriétés avancées permettent de créer des mises en page plus complexes.

- overflow : définit la gestion des débordements (visible, hidden, scroll, auto).
- clip : découpe un élément à une forme spécifique.
- opacity : définit la transparence d'un élément (de 0 à 1).
- box-shadow : applique une ombre portée à un élément.
- filter : applique des effets graphiques à un élément (ex. flou, luminosité).
- object-fit : définit comment une image ou un autre média s'ajuste dans son conteneur.

7. Propriétés pour les formulaires :

Contrôle l'apparence et le comportement des éléments de formulaire.

- input, textarea, button, select : spécifie des styles pour les éléments de formulaire.
- border-radius : arrondit les coins des éléments de formulaire.
- outline : définit le contour autour des éléments.
- box-shadow : ajoute une ombre autour des éléments de formulaire.
- appearance : modifie l'apparence des éléments de formulaire (par exemple, none pour supprimer l'apparence par défaut d'un bouton).

8. Autres propriétés utiles :

- cursor : définit le type de curseur lorsqu'il survole un élément (pointer, default, crosshair, etc.).
- visibility : définit si un élément est visible ou non (visible, hidden).
- user-select : définit si l'utilisateur peut sélectionner du texte (none, auto, text).

9. Propriétés pour la gestion des médias :

Ces propriétés sont utiles pour adapter les sites Web à différents types d'écrans, en particulier sur les appareils mobiles.

- **Media Queries** :
  - @media : permet d'appliquer des styles en fonction de conditions spécifiques comme la taille de l'écran ou l'orientation du dispositif.
    - Exemple :

```
@media (max-width: 600px) {
  body {
    background-color: lightblue;
  }
}
```

- **Propriétés** à utiliser dans les media queries :
  - width, height : taille de la fenêtre de l'affichage.
  - min-width, max-width : tailles minimales ou maximales de la fenêtre.
  - orientation : définit les styles en fonction de l'orientation (portrait ou paysage).
  - aspect-ratio : spécifie un ratio largeur/hauteur.
  - resolution : défini en dpi (dots per inch), utile pour les écrans haute résolution (ex. @media (min-resolution: 192dpi)).

10. Propriétés de transformation :

Les propriétés de transformation permettent de manipuler l'apparence d'un élément sans modifier sa structure.

- **Transformations 2D** :
  - transform : applique des transformations telles que la translation, la rotation, l'échelle et l'inclinaison.
    - rotate(deg) : fait pivoter un élément.
    - scale(x, y) : agrandit ou réduit un élément en fonction des axes x et y.
    - translate(x, y) : déplace un élément à une position spécifique sur l'axe x et y.
    - skew(x, y) : applique une distorsion (inclinaison) à un élément.
- **Transformations 3D** :
  - rotateX(deg), rotateY(deg), rotateZ(deg) : effectue une rotation sur les axes X, Y ou Z dans l'espace 3D.
  - perspective : définit la perspective appliquée aux éléments enfants d'un conteneur 3D.
  - transform-style : définit si un élément doit être rendu en 3D ou non (flat, preserve-3d).

11. Propriétés de la boîte de contenu :

- **box-sizing** : définit la manière dont la taille d'un élément est calculée. La valeur border-box inclut les bordures et les paddings dans la taille de l'élément.
  - Exemple :

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

12. Propriétés pour l'accessibilité :

Certaines propriétés CSS aident à améliorer l'accessibilité et l'interactivité des pages Web.

- :focus : permet de styliser les éléments lorsqu'ils sont focalisés (par exemple, lors de la navigation avec le clavier).
- outline : permet de personnaliser l'apparence du contour autour des éléments focalisés (très utilisé pour l'accessibilité, bien que outline ne soit pas visible par défaut).
  - Exemple :

```
input:focus {
  outline: 2px solid #00f;
}
```

- aria-* : bien que ce ne soit pas une propriété CSS, les attributs aria-* sont utilisés pour améliorer l'accessibilité en permettant aux technologies d'assistance de mieux comprendre et naviguer sur la page.

13. Propriétés avancées pour la création de transitions et d'animations :

Les animations CSS permettent de créer des effets visuels fluides.

- **@keyframes** :
  - Cette règle définit les étapes de l'animation en termes de pourcentages de progression.
  - Exemple de base d'animation :

```
@keyframes example {
  0% { background-color: red; }
  50% { background-color: yellow; }
  100% { background-color: green; }
}

div {
  animation: example 4s infinite;
}
```

- **Propriétés liées aux animations** :
  - animation-name : définit le nom de l'animation (utilise @keyframes).
  - animation-duration : définit la durée de l'animation.
  - animation-delay : délai avant de commencer l'animation.
  - animation-iteration-count : définit le nombre de fois que l'animation doit être répétée.
  - animation-timing-function : règle la vitesse de l'animation avec des courbes comme linear, ease, ease-in, ease-out.
  - animation-direction : définit la direction de l'animation (normal, reverse, alternate).

14. Propriétés pour la gestion des fontes et des glyphes :

- **@font-face** : permet d'importer des polices personnalisées depuis un fichier externe.
  - Exemple :

```
@font-face {
 font-family: "MyCustomFont";
 src: url("myfont.woff") format("woff");
}

body {
 font-family: "MyCustomFont", sans-serif;
}
```

- **font-variant** : modifie la variante typographique, comme les petites majuscules (small-caps), le style normal, etc.
- **font-feature-settings** : permet d'activer ou de désactiver des fonctionnalités OpenType (comme les ligatures, les fractions, etc.).

15. Propriétés de la bordure :

Les propriétés liées aux bordures permettent de personnaliser l'apparence des bordures d'un élément.

- border-radius : arrondit les coins d'un élément.
  - Exemple :

```
.box {
 border-radius: 10px;
}
```

- **Bordures complexes** :
  - border-style : définit le style de la bordure (solid, dashed, dotted, double, groove, ridge, etc.).
  - border-width : définit l'épaisseur de la bordure.
  - border-color : définit la couleur de la bordure.
  - border-image : permet d'utiliser une image comme bordure (par exemple, une bordure avec une image spécifique au lieu d'une couleur unie).

16. Propriétés des ombres :

- **Ombre portée** :
  - box-shadow : applique une ombre portée à un élément.
    - Exemple :

```
.shadow {
 box-shadow: 10px 10px 15px rgba(0, 0, 0, 0.3);
}
```

- **Ombre du texte** :
  - text-shadow : ajoute une ombre au texte.
    - Exemple :

```
h1 {
 text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);
}
```

17. Propriétés de positionnement avancé :

- clip-path : permet de découper un élément selon une forme géométrique (rectangle, cercle, polygone).
  - Exemple :

```
img {
 clip-path: circle(50%);
}
```

- contain : permet de contrôler la portée d'un élément (par exemple, pour optimiser les performances du rendu).

18. Propriétés pour les transformations de texte :

- writing-mode : modifie l'orientation du texte dans un élément (horizontal-tb, vertical-rl, vertical-lr).
- direction : détermine la direction du texte (généralement pour les langues de droite à gauche, comme l'arabe ou l'hébreu).

19. Propriétés liées aux tableaux :

- table-layout : permet de spécifier le mode de mise en page des tableaux (auto, fixed).
- border-collapse : définit si les bordures des cellules doivent se fusionner (collapse) ou rester séparées (separate).
- caption-side : définit la position de la légende d'un tableau (top, bottom).

Exemple HTML + CSS

index.html

```
<!DOCTYPE html>
    <html lang="fr">
    <head>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-width, initial-scale=1.0">
      <title>Exemple CSS Complet</title>
      <link rel="stylesheet" href="styles.css">
    </head>
    <body>

      <header>
        <h1>Exemple de Page avec CSS Avancé</h1>
      </header>

      <section class="content">
        <div class="box">
          <p>Cette boîte a une bordure arrondie, un ombre et une animation !</p>
        </div>
        <button class="btn">Cliquez-moi</button>
      </section>

      <footer>
        <p>&copy; 2024 Exemple de CSS</p>
      </footer>

    </body>
    </html>
```

styles.css

```
    /* Réinitialisation de la mise en page de base */
    * {
      margin: 0;
      padding: 0;
      box-sizing: border-box;
    }

    body {
      font-family: 'Arial', sans-serif;
      line-height: 1.6;
      background-color: #f0f0f0;
    }

    header {
      background-color: #333;
      color: white;
      text-align: center;
      padding: 20px;
    }
```

```
h1 {
  font-size: 2.5rem;
  text-transform: uppercase;
  letter-spacing: 2px;
  animation: fadeIn 2s ease-out;
}

/* Animation de fade-in pour le texte */
@keyframes fadeIn {
  from {
    opacity: 0;
  }
  to {
    opacity: 1;
  }
}

/* Mise en page de la section principale */
.content {
  display: flex;
  justify-content: space-around;
  align-items: center;
  min-height: 80vh;
  padding: 40px;
  flex-wrap: wrap;
}

.box {
  background-color: #fff;
  padding: 30px;
  width: 300px;
  border-radius: 15px;
  box-shadow: 5px 5px 15px rgba(0, 0, 0, 0.1);
  transition: transform 0.3s ease-in-out;
  text-align: center;
}

.box:hover {
  transform: scale(1.05); /* Transformation au survol */
}

.box p {
  font-size: 1.1rem;
}

/* Bouton avec animation de survol */
.btn {
  background-color: #007bff;
  color: white;
  padding: 15px 30px;
  font-size: 1.2rem;
  border: none;
  border-radius: 25px;
  cursor: pointer;
  transition: background-color 0.3s ease;
}

.btn:hover {
  background-color: #0056b3; /* Changement de couleur au survol */
}

footer {
  text-align: center;
  background-color: #333;
  color: white;
  padding: 10px;
  position: absolute;
  width: 100%;
  bottom: 0;
}

@media (max-width: 600px) {
```

```
  /* Media Query pour les petits écrans */
  .content {
    flex-direction: column;
  }

  .box {
    width: 80%;
    margin-bottom: 20px;
  }
}
```

Explication de l'exemple :

1. **Structure HTML** :
   - Un **`header`** avec un titre principal.
   - Une **`section.content`** contenant une boîte avec du texte et un bouton.
   - Un **`footer`** avec une mention de copyright.
2. **CSS** :
   - **Réinitialisation de la mise en page** : `* { margin: 0; padding: 0; box-sizing: border-box; }` pour un meilleur contrôle du style.
   - **Typographie** : Utilisation de la famille `Arial`, avec un espacement de lettres pour le titre.
   - **Animations** :
     - Une animation `fadeIn` pour faire apparaître le titre au chargement de la page.
     - Une transition de transformation pour la boîte qui s'agrandit légèrement lorsqu'on passe la souris dessus.
   - **Flexbox** : Utilisé pour aligner et espacer les éléments de manière fluide dans `.content`, avec un comportement adaptable sur les petits écrans grâce aux media queries.
   - **Bouton** : Le bouton change de couleur au survol grâce à une transition.
   - **Media Query** : Lorsque la largeur de l'écran est inférieure à 600px, la disposition des éléments passe de `row` à `column` et les boîtes prennent 80% de la largeur de l'écran.

Exemple HTML + CSS avec `Grid`, `Clip-path`, Animations et Transitions

`index.html`

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Exemple de Mise en Page Avancée</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>

  <header>
    <h1>Exemple de Design Moderne</h1>
  </header>

  <main class="main-content">
    <div class="card card-1">
      <h2>Carte 1</h2>
      <p>Voici un exemple d'une carte avec un effet de survol et une
animation.</p>
    </div>
    <div class="card card-2">
      <h2>Carte 2</h2>
```

```
            <p>Une autre carte avec un style différent et une transformation en 3D
      au survol.</p>
          </div>
          <div class="card card-3">
            <h2>Carte 3</h2>
            <p>Une carte avec une bordure en forme de cercle grâce à `clip-
      path`.</p>
          </div>
        </main>

        <footer>
          <p>&copy; 2024 Exemples de CSS</p>
        </footer>

      </body>
      </html>
```

styles.css

```
/* Réinitialisation des marges et du padding */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: 'Arial', sans-serif;
  background-color: #fafafa;
  color: #333;
}

/* Style du header */
header {
  background-color: #4CAF50;
  color: white;
  text-align: center;
  padding: 30px;
  text-transform: uppercase;
  font-size: 2rem;
  letter-spacing: 2px;
  animation: slideIn 1s ease-out;
}

/* Animation de l'en-tête */
@keyframes slideIn {
  from {
    transform: translateY(-100%);
  }
  to {
    transform: translateY(0);
  }
}

/* Style du contenu principal */
.main-content {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));
  gap: 20px;
  padding: 20px;
  justify-items: center;
}

/* Carte de base */
.card {
  background-color: white;
  border-radius: 15px;
  padding: 20px;
  text-align: center;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
  transition: transform 0.3s ease, box-shadow 0.3s ease;
```

```
  width: 280px;
}

/* Effet au survol pour la carte */
.card:hover {
  transform: translateY(-10px); /* Décalage vers le haut */
  box-shadow: 0 8px 15px rgba(0, 0, 0, 0.2); /* Ombre plus marquée */
}

/* Carte avec un effet 3D */
.card-2 {
  perspective: 1000px;
}

.card-2:hover h2 {
  transform: rotateY(360deg);
  transition: transform 1s;
}

.card h2 {
  font-size: 1.5rem;
  color: #4CAF50;
  margin-bottom: 10px;
  transition: color 0.3s ease;
}

.card p {
  font-size: 1rem;
  color: #666;
}

/* Carte avec clip-path pour une bordure arrondie en forme de cercle */
.card-3 {
  clip-path: circle(50% at 50% 50%);
  transition: transform 0.4s ease;
}

.card-3:hover {
  transform: scale(1.1); /* Agrandir la carte au survol */
}

/* Style du footer */
footer {
  text-align: center;
  background-color: #4CAF50;
  color: white;
  padding: 10px;
  position: fixed;
  bottom: 0;
  width: 100%;
}

/* Media Queries pour la réactivité */
@media (max-width: 768px) {
  header {
    font-size: 1.5rem;
    padding: 20px;
  }

  .main-content {
    grid-template-columns: 1fr; /* Affichage en une seule colonne */
  }

  .card {
    width: 100%; /* Les cartes prennent toute la largeur */
  }
}
```

Explication de l'exemple :

1. **Structure HTML** :
   - Un **`header`** avec un titre.
   - Un **`main`** contenant trois **cartes** (divs) avec des titres et des descriptions.
   - Un **`footer`** en bas de la page.
2. **CSS** :
   - **Animations et Transitions** :
     - L'animation `slideIn` fait glisser le titre du `header` depuis le haut de l'écran.
     - Les cartes utilisent une **transition** pour l'effet de survol : changement d'ombre et translation verticale.
     - Pour la **carte-2**, un effet 3D est appliqué en utilisant la **perspective** et une **rotation 360°** du titre au survol.
   - **Grid Layout** :
     - Utilisation de `display: grid` avec `grid-template-columns` pour créer une disposition fluide des cartes. Les cartes s'adaptent en fonction de l'espace disponible (grâce à `repeat(auto-fill, minmax(280px, 1fr))`).
   - **Clip-path** :
     - La **carte-3** utilise `clip-path: circle(50% at 50% 50%)` pour créer un bord arrondi circulaire.
   - **Réactivité (Media Queries)** :
     - Sur des écrans plus petits (moins de 768px de largeur), les cartes s'affichent en une seule colonne, et le texte de l'en-tête est réduit.