CoursInfo

CSS3 & Design Responsive

Maîtrisez les styles, animations, Flexbox, Grid et le responsive design pour créer des interfaces modernes et attractives.

Niveau : Débutant • Durée : ~18h • 10 chapitres • Certificat inclus

---

1. Introduction au CSS3

CSS (Cascading Style Sheets) est le langage qui contrôle l'apparence visuelle des pages HTML. Sans CSS, toutes les pages web seraient du texte brut sans mise en forme. CSS3 est la version moderne, avec des fonctionnalités puissantes comme les animations, les dégradés et les mises en page avancées.

Comment lier CSS à HTML ?

Il existe 3 façons d'intégrer du CSS, mais une seule est recommandée pour des projets sérieux.

```
<!-- ■ Méthode 1 : Feuille de style externe (recommandée) -->

<link
rel
=
"stylesheet"

href
=
"assets/css/theme.css"
>


<!-- ■■ Méthode 2 : Style interne dans <head> (usage limité) -->

<style>

body
{
background
:
#f8f9fa
; }
</style>


<!-- ■ Méthode 3 : Style inline (éviter !) -->

<p
style
=
"color: red;"
>Texte</p>
```

Syntaxe CSS de base

```
/* Sélecteur { propriété: valeur; } */
```

```
h1
{

color
:
#264de4
;

font-size
:
2.5rem
;

font-weight
:
700
;

text-align
:
center
;
}


p
{

color
:
#374151
;

line-height
:
1.8
;

margin-bottom
:
1rem
;
}
```

CSS vs HTML — rôles distincts

HTML	CSS
Structure le contenu	Définit l'apparence
Titre	h1 { color: blue; }
Texte	p { font-size: 16px; }
Squelette de la page	Habillage visuel

■ Règle d'or : Gardez toujours HTML et CSS séparés. Évitez les styles inline — ils rendent la maintenance cauchemardesque sur un grand projet.

2. Sélecteurs CSS & Cascade

Les sélecteurs CSS permettent de cibler précisément les éléments HTML à styler. La maîtrise des sélecteurs est essentielle pour écrire du CSS propre et efficace.

Les sélecteurs principaux

```
/* Sélecteur de balise */


p
{
color
:
gray
; }


/* Sélecteur de classe (. avant le nom) */


.btn-primary
{
background
:
#264de4
; }


/* Sélecteur d'ID (# avant le nom) */


#navbar
{
position
:
sticky
;
top
:
0
; }


/* Sélecteur universel */


*
{
box-sizing
:
border-box
; }


/* Descendant (h2 à l'intérieur d'un .card) */


.card h2
```

```
{
font-size
:
1.5rem
; }


/* Pseudo-classes */


a:hover
{
color
:
#264de4
; }

li:first-child
{
font-weight
:
700
; }

input:focus
{
border-color
:
#264de4
; }


/* Pseudo-éléments */


p::first-line
{
font-weight
:
700
; }

.btn::after
{
content
:
" →"
; }
```

La spécificité CSS

Quand plusieurs règles s'appliquent au même élément, CSS utilise la spécificité pour décider laquelle gagne.

Type de sélecteur	Spécificité	Exemple
Style inline	1,0,0,0 (le plus fort)	style="..."
ID	0,1,0,0	#navbar

Classe / Pseudo-classe	0,0,1,0	.btn, :hover
Balise	0,0,0,1	h1, p
Universel	0,0,0,0	*

■■ !important : Évitez !important. C'est une béquille qui rend le CSS incontrôlable. Mieux vaut augmenter la spécificité de votre sélecteur.

3. Couleurs & Typographie

Les couleurs et la typographie sont les deux piliers de l'identité visuelle d'un site. CSS3 offre de nombreuses façons de les définir avec précision.

Formats de couleur

```
/* Hexadécimal (le plus courant) */


color
:
#264de4
;

color
:
#fff
;
/* raccourci */


/* RGB */


color
:
rgb(38, 77, 228)
;


/* RGBA — avec transparence (0 = transparent, 1 = opaque) */


background
:
rgba(38, 77, 228, 0.15)
;


/* HSL (teinte, saturation, luminosité) */


color
:
hsl(228, 75%, 52%)
;


/* Dégradés */


background
:
linear-gradient(135deg, #264de4, #2965f1)
;

background
:
```

```
radial-gradient(circle, #264de4, #1a3a9e)
;
```

Typographie CSS

```
/* Importer une police Google Fonts dans HTML */


/* <link href="https://fonts.googleapis.com/css2?family=Rubik:wght@400;700&display=swap">
*/


body
{

font-family
:
'Rubik'
,
sans-serif
;

font-size
:
16px
;
/* base */


line-height
:
1.6
;
}


h1
{

font-size
:
3rem
;

font-weight
:
700
;

letter-spacing
:
-0.02em
;

text-transform
:
uppercase
;
}
```

```
p
{

font-size
:
1rem
;

text-align
:
justify
;

color
:
#374151
;
}
```

Unités CSS

Unité	Type	Usage recommandé
px	Absolue	Bordures, ombres fines
rem	Relative à la racine	Tailles de texte (préféré)
em	Relative au parent	Marges relatives au texte
%	Relative au parent	Largeurs de colonnes
vw / vh	Relative au viewport	Sections plein écran

4. Le Modèle de Boîte (Box Model)

Chaque élément HTML est une boîte rectangulaire. Comprendre le Box Model est indispensable pour maîtriser l'espacement et la taille des éléments.

```
.carte
{

/* CONTENU */


width
:
300px
;

height
:
200px
;


/* PADDING — espace intérieur */


padding
:
1.5rem
;
/* tous côtés */


padding
:
1rem

2rem
;
/* haut/bas gauche/droite */


padding-top
:
1rem
;

padding-left
:
2rem
;


/* BORDER — bordure */


border
:
2px
```

```
solid

#264de4
;

border-radius
:
8px
;


/* MARGIN — espace extérieur */


margin
:
2rem

auto
;
/* centrage horizontal */


margin-bottom
:
1.5rem
;


/* IMPORTANT : inclut padding+border dans width */


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

Propriétés d'affichage (display)

```
display
:
block
;
/* Prend toute la largeur (div, p, h1) */


display
:
inline
;
/* Suit le flux du texte (span, a) */


display
:
inline-block
;
/* Inline + accepte width/height */
```

```
display
:
flex
;
/* Flexbox (chapitre 5) */


display
:
grid
;
/* CSS Grid (chapitre 6) */


display
:
none
;
/* Cache l'élément */
```

■ Astuce universelle : Mettez toujours * { box-sizing: border-box; } en début de CSS. Ça évite des surprises de calcul de taille.

5. Flexbox

Flexbox est un système de mise en page unidimensionnel (une ligne ou une colonne) qui rend l'alignement et la distribution d'espace entre éléments extrêmement simple.

Principe de base

```
.conteneur
{

display
:
flex
;


/* Direction */


flex-direction
:
row
;
/* défaut: horizontal */


flex-direction
:
column
;
/* vertical */


/* Alignement horizontal */


justify-content
:
flex-start
;
/* gauche */


justify-content
:
center
;
/* centré */


justify-content
:
space-between
;
/* espace entre */


justify-content
:
```

```
space-around
;
/* espace autour */


/* Alignement vertical */


align-items
:
stretch
;
/* défaut */


align-items
:
center
;
/* centré verticalement */


align-items
:
flex-end
;
/* en bas */


/* Retour à la ligne */


flex-wrap
:
wrap
;

gap
:
1rem
;
}


.enfant
{

flex
:
1
;
/* grandit pour remplir l'espace */


flex
:
0 0 300px
;
/* largeur fixe : ne grandit pas, ne rétrécit pas */
```

```
}
```

Cas d'usage courants

```
/* Centrage parfait horizontal + vertical */


.hero
{

display
:
flex
;

align-items
:
center
;

justify-content
:
center
;

min-height
:
100vh
;
}


/* Navbar avec logo à gauche, liens à droite */


nav
{

display
:
flex
;

align-items
:
center
;

justify-content
:
space-between
;

padding
:
1rem 2rem
;
}
```

■ Flexbox = 1 dimension. Utilisez Flexbox pour les navbars, les rangées de cartes, les boutons alignés. Pour des mises en page en 2 dimensions, utilisez CSS Grid (chapitre 6).

6. CSS Grid

CSS Grid is un système de mise en page bidimensionnel (lignes ET colonnes). C'est l'outil idéal pour créer des mises en page complexes comme des tableaux de bord, des galeries photo ou des layouts de page entière.

Syntaxe de base

```
.grille
{

display
:
grid
;


/* 3 colonnes égales */


grid-template-columns
:
1fr

1fr

1fr
;


/* Même chose avec repeat() */


grid-template-columns
:
repeat
(
3
,
1fr
);


/* Colonnes auto-adaptées (responsive natif) */


grid-template-columns
:
repeat
(
auto-fit
,
minmax
(
280px
,
1fr
));
```

```
/* Espace entre les cellules */


gap
:
1.5rem
;
}


/* Élément qui s'étend sur 2 colonnes */


.carte-large
{

grid-column
:
span

2
;
}


/* Layout classique header/sidebar/content/footer */


.page
{

display
:
grid
;

grid-template-areas
:

"header header"


"sidebar content"


"footer footer"
;

grid-template-columns
:
250px

1fr
;
}

.header
{
```

```
grid-area
:
header
; }

.sidebar
{
grid-area
:
sidebar
; }

.content
{
grid-area
:
content
; }

.footer
{
grid-area
:
footer
; }
```

■■ Flexbox vs Grid : Utilisez Flexbox pour aligner des éléments dans une direction (navbar, boutons). Utilisez Grid pour des mises en page 2D (cartes, dashboard, galeries).

7. Responsive Design & Media Queries

Le responsive design permet à une page de s'adapter automatiquement à toutes les tailles d'écran : mobile, tablette, desktop. En 2024, plus de 60% du trafic web vient des mobiles.

Meta viewport — indispensable

```
<!-- Toujours dans <head> — active le responsive -->

<meta name=
"viewport"
content=
"width=device-width, initial-scale=1.0"
>
```

Media Queries

```
/* Mobile first : on commence par le mobile, on ajoute pour les grands écrans */


/* Styles de base = mobile */


.cartes
{

display
:
grid
;

grid-template-columns
:
1fr
;
/* 1 colonne sur mobile */


gap
:
1rem
;
}


/* Tablette (≥ 768px) */


@media (min-width: 768px)
{

.cartes
{

grid-template-columns
:
repeat
(
2
```

```
,
1fr
);
}
}


/* Desktop (≥ 1024px) */


@media (min-width: 1024px)
{

.cartes
{

grid-template-columns
:
repeat
(
3
,
1fr
);
}
}


/* Masquer sur mobile */


@media (max-width: 767px)
{

.sidebar
{
display
:
none
; }

.hero h1
{
font-size
:
1.8rem
; }
}
```

Breakpoints standard

Nom	Largeur	Appareils ciblés
Mobile	< 576px	Petits smartphones
Mobile L	576px – 767px	Grands smartphones
Tablette	768px – 1023px	Tablettes, iPad

Desktop	1024px – 1279px	Petits laptops
Large	≥ 1280px	Grands écrans

■ Mobile First : Écrivez vos styles pour mobile en premier, puis utilisez min-width pour enrichir progressivement sur les grands écrans. C'est la méthode recommandée en 2024.

8. Transitions & Animations CSS3

CSS3 permet de créer des animations fluides sans JavaScript, directement avec les propriétés transition et animation.

Transitions

Une transition est un changement progressif d'une propriété CSS d'un état à un autre (souvent sur :hover).

```
.btn
{

background
:
#264de4
;

color
:
white
;

padding
:
0.75rem 1.5rem
;

border-radius
:
6px
;


/* Transition sur toutes les propriétés, 0.3s, effet ease */


transition
:
all

0.3s

ease
;
}


.btn:hover
{

background
:
#1a3a9e
;

transform
:
translateY
(
-3px
```

```
);
/* monte légèrement */


box-shadow
:
0 8px 25px

rgba
(
38,77,228,0.4
);
}
```

Animations @keyframes

```
/* Définir l'animation */


@keyframes

glisser-bas
{

from
{

opacity
:
0
;

transform
:
translateY
(
-30px
);
}

to
{

opacity
:
1
;

transform
:
translateY
(
0
);
}
}


/* Appliquer l'animation */
```

```
.hero-titre
{

animation
:
glisser-bas

0.8s

ease

forwards
;
}


/* Animation infinie — spinner */


@keyframes

rotation
{

from
{
transform
:
rotate
(
0deg
); }

to
{
transform
:
rotate
(
360deg
); }
}

.spinner
{

animation
:
rotation

1s

linear

infinite
;
}
```

■■ Performance : Animez uniquement transform et opacity. Animer width, height, margin force le navigateur à recalculer la mise en page — c'est lent.

9. Variables CSS & Bonnes pratiques

Les variables CSS (custom properties) permettent de stocker des valeurs réutilisables. Elles rendent votre CSS plus maintenable et cohérent.

Déclarer et utiliser des variables

```
/* Variables globales dans :root */


:root
{

/* Couleurs */


--primary
:
#264de4
;

--secondary
:
#1a3a9e
;

--dark
:
#1f2937
;

--gray
:
#6b7280
;

--white
:
#ffffff
;

--light-bg
:
#f8f9fa
;


/* Typographie */


--font-main
:
'Rubik'
,
sans-serif
;

--font-size-base
:
```

```
1rem
;


/* Ombres */


--shadow
:
0 4px 15px

rgba(0,0,0,0.1)
;

--shadow-hover
:
0 8px 25px

rgba(0,0,0,0.15)
;


/* Rayons */


--radius
:
8px
;

--radius-lg
:
16px
;
}


/* Utilisation avec var() */


.btn-primary
{

background
:
var(--primary)
;

border-radius
:
var(--radius)
;

box-shadow
:
var(--shadow)
;

font-family
```

```
:
var(--font-main)
;
}

.btn-primary:hover
{

background
:
var(--secondary)
;

box-shadow
:
var(--shadow-hover)
;
}
```

Bonnes pratiques CSS

■ À faire	■ À éviter
Utiliser des classes pour styler	Utiliser des IDs pour styler
Mobile First avec min-width	Desktop First avec max-width
Variables CSS pour les couleurs	Répéter les codes hex partout
box-sizing: border-box sur *	Mélanger px et % sans logique
Organiser le CSS par sections	Tout mettre dans un seul bloc
Commentaires clairs	Code CSS non documenté

■ Organisation du CSS : Structurez votre fichier CSS dans cet ordre : Variables → Reset → Base → Composants → Pages → Utilitaires → Responsive. Vous retrouverez tout facilement.

10. Projet Final — Page d'accueil complète

Mettons en pratique tout ce que nous avons appris. Voici le CSS complet d'une page d'accueil moderne avec navbar sticky, hero, cartes de cours, et footer.

```
/* ============================
VARIABLES
============================ */


:root
{

--primary
:
#6366f1
;

--secondary
:
#4f46e5
;

--dark
:
#1f2937
;

--shadow
:
0 4px 15px

rgba(0,0,0,0.1)
;
}


/* ============================
RESET
============================ */


*
{
margin
:
0
;
padding
:
0
;
box-sizing
:
border-box
; }

html
{
```

```
scroll-behavior
:
smooth
; }


/* ============================
NAVBAR
============================ */


.navbar
{

position
:
sticky
;
top
:
0
;
z-index
:
1000
;

background
:
white
;

box-shadow
:
var(--shadow)
;

padding
:
1rem 2rem
;

display
:
flex
;

align-items
:
center
;

justify-content
:
space-between
;
}
```

```
/* ============================
HERO
============================ */


.hero
{

background
:
linear-gradient
(
135deg
,
var(--primary)
,
var(--secondary)
);

color
:
white
;

min-height
:
600px
;

display
:
flex
;

align-items
:
center
;

justify-content
:
center
;

text-align
:
center
;
}


/* ============================
CARTES COURS
============================ */


.cours-grid
{
```

```
display
:
grid
;

grid-template-columns
:
repeat
(
auto-fit
,
minmax
(
280px
,
1fr
));

gap
:
1.5rem
;
}

.card
{

border-radius
:
8px
;

box-shadow
:
var(--shadow)
;

transition
:
transform

0.3s

ease
,
box-shadow

0.3s

ease
;

overflow
:
hidden
;
}

.card:hover
```

```
{

transform
:
translateY
(
-10px
);

box-shadow
:
0 8px 25px

rgba(0,0,0,0.15)
;
}

.card img
{

width
:
100%
;

height
:
200px
;

object-fit
:
cover
;
}


/* ============================
RESPONSIVE
============================ */


@media (max-width: 768px)
{

.hero h1
{
font-size
:
2rem
; }

.navbar
{
padding
:
0.75rem 1rem
; }
}
```

■ Félicitations ! Vous avez terminé le cours CSS3. Vous maîtrisez maintenant : les sélecteurs et la cascade, les couleurs et la typographie, le Box Model, Flexbox, Grid, le Responsive Design, les animations et les variables CSS. La prochaine étape : PHP pour rendre vos pages dynamiques !

Outils recommandés pour continuer

- CSS Tricks : css-tricks.com — Guides et astuces CSS avancées
- Flexbox Froggy : flexboxfroggy.com — Jeu pour apprendre Flexbox
- Grid Garden : cssgridgarden.com — Jeu pour apprendre Grid
- MDN CSS : developer.mozilla.org/fr/docs/Web/CSS — Référence complète

Exercices & Quiz de vérification

Testez vos connaissances avant de passer au cours suivant. Chaque question a une correction détaillée à dérouler.

■ Exercice pratique : Créez une carte (.card) avec une image, un titre et un bouton. Centrez-la horizontalement avec Flexbox, ajoutez un border-radius, une ombre douce, puis une transition qui la soulève légèrement au survol (:hover). Voir une correction possible .card { max-width:320px; margin:2rem auto; border-radius:10px; box-shadow:0 4px 15px rgba(0,0,0,0.1); overflow:hidden; transition:transform .3s ease; } .card:hover { transform:translateY(-8px); }

1. Que signifie l'acronyme CSS ?

- A. Computer Style Sheets
- B. Cascading Style Sheets
- C. Creative Style System

Correction : Réponse : B. CSS signifie « Cascading Style Sheets » (feuilles de style en cascade).

2. Quel sélecteur a la plus forte spécificité ?

- A. Une classe (.btn)
- B. Une balise (p)
- C. Un style inline (style="...")

Correction : Réponse : C. Le style inline a la spécificité la plus élevée, avant l'ID, la classe et la balise.

3. Quelle propriété permet de créer une mise en page unidimensionnelle flexible ?

- A. display: grid
- B. display: flex
- C. display: block

Correction : Réponse : B. Flexbox (display: flex) organise les éléments sur un seul axe, ligne ou colonne.

4. Quelle unité est relative à la taille de police de l'élément racine (html) ?

- A. em
- B. rem
- C. vh

Correction : Réponse : B. rem se calcule toujours par rapport à la taille de police de <html>, contrairement à em qui dépend du parent direct.

5. Quelles propriétés faut-il privilégier pour des animations fluides et performantes ?

- A. width et height
- B. margin et padding
- C. transform et opacity

Correction : Réponse : C. Animer transform et opacity évite de recalculer la mise en page, contrairement à width/height/margin.

6. Que fait la règle box-sizing: border-box ?

- A. Elle cache la boîte
- B. Elle inclut le padding et la bordure dans la largeur/hauteur totale

- C. Elle supprime les marges

Correction : Réponse : B. Sans elle, padding et border s'ajoutent à la largeur définie, ce qui complique les calculs de mise en page.

7. Quel outil CSS est le plus adapté pour un tableau de bord en 2 dimensions (lignes ET colonnes) ?

- A. CSS Grid
- B. Flexbox
- C. Position absolute uniquement

Correction : Réponse : A. CSS Grid gère nativement lignes et colonnes ; Flexbox est conçu pour une seule dimension.

8. Quelle approche est recommandée pour écrire des media queries en 2024 ?

- A. Desktop first avec max-width
- B. Mobile first avec min-width
- C. Aucune media query n'est nécessaire

Correction : Réponse : B. On écrit les styles de base pour mobile, puis on enrichit progressivement avec min-width pour les écrans plus grands.

Ressources supplémentaires

- MDN Web Docs — Référence CSS complète
- MDN — Apprendre les bases du CSS
- W3C — Spécification officielle CSS
- Can I Use — Compatibilité des propriétés CSS par navigateur