كلـيـــــــــــة العلوم
السملالية - مراكش
FACULTE DES SCIENCES
SEMLALIA-MARRAKECH
MARRAKECH
جامعة القاضي عياض
UNIVERSITÉ CADI AYYAD

Département Informatique

Technologies de Web :
Chapitre 4 : Effets avancés en CSS

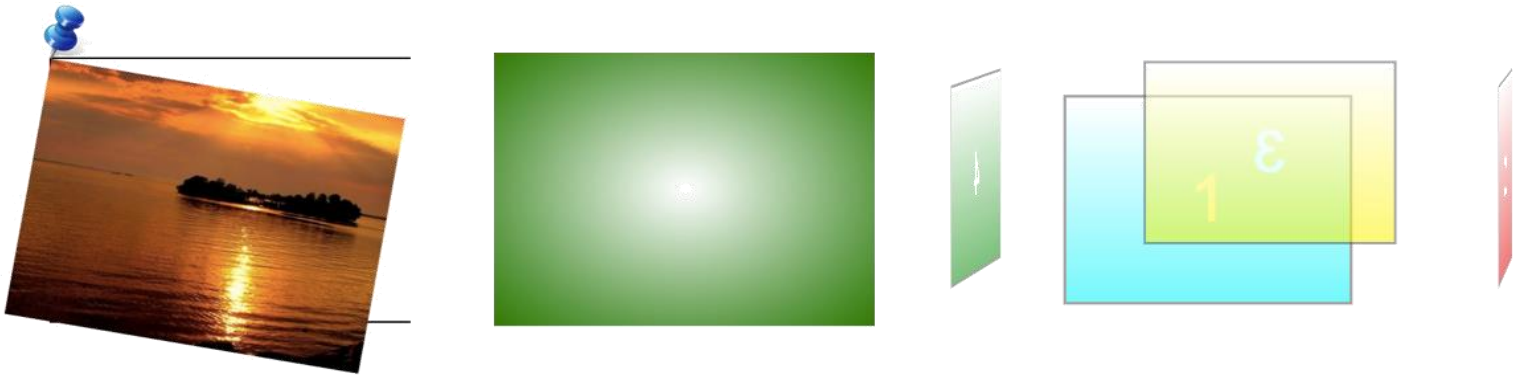
Aimad QAZDAR

a.qazdar@uca.ma

24 octobre 2019

Introduction

- Depuis CSS3, les développeurs web peuvent utiliser de nombreux effets dont ils ne pouvaient que rêver il y a encore quelques années :
  - Séparation du texte sur plusieurs colonnes
  - Dégradés
  - Transformations et animations d'éléments (en 2D et en 3D !)
  - etc.



Quelques effets en CSS

Plan

- Les multi-colonnes
- Les dégradés
- Les transformations 2D
- Les transitions CSS
- Les animations CSS
- Les transformations 3D

Les multi-colonnes

Les multi-colonnes

- Pour un meilleur confort de lecture, les spécialistes recommandent de limiter la largeur des lignes de texte ?
- Il n'y a pas de règle absolue, mais on dit que l'idéal se situe entre 50 et 70 caractères par ligne.
- Cela évite aux yeux d'avoir à faire de trop larges allers-retours d'une ligne à l'autre.
- C'est pour cela que, depuis longtemps, les journaux utilisent une mise en page sur plusieurs colonnes.
- Avant le Web ne permettait pas ce type de mise en page.
- Grâce à CSS3, on peut maintenant automatiquement créer une mise en page du texte sur plusieurs colonnes de façon très efficace

Multi-colonnes

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Bonum incolumis acies: misera caecitas. Idem iste, inquam, de voluptate quid sentit? Aufert enim sensus actionemque tollit omnem. Erat enim Polemonis. Potius inflammat, ut coercendi magis quam dedocendi esse videantur. Hoc enim constituto in philosophia constituta sunt omnia. Duo Reges: constructio interrete. Dicet pro me ipsa virtus nec dubitabit isti vestro beato M. Sed ut iis bonis erigimur, quae expectamus, sic laetamur iis, quae recordamur. Et quidem illud ipsum non nimium probo et tantum patior, philosophum loqui de cupiditatibus finiendis. At multis malis affectus.

Non igitur bene. Gloriosa ostentatio in constituendo summo bono. Maximus dolor, inquit, brevis est. Quid nunc honeste dicit? maximarum rerum timore nemo potest. Estne, quaeso, inquam, sitienti in bibendo voluptas? Quae quidem vel cum periculo est quaerenda vobis; Quis est, qui non oderit libidinosam, protervam adolescentiam? Hoc est dicere: Non reprehenderem asotos, si non essent asoti. Vestri haec verecundius, illi fortasse constantius.

Plus de texte

Videamus igitur sententias eorum, tum ad verba redeamus. At certe gravius. Hoc dixerit potius Ennius: Nimium boni est, cui nihil est mali. Qui autem esse poteris, nisi te amor ipse ceperit? Ego quoque, inquit, didicerim libentius si quid attuleris, quam te reprehenderim. Licet hic rursus ea commemores, quae optimis verbis ab Epicuro de laude amicitiae dicta sunt. Effluit igitur voluptas quae tueretur. Illa argumenta propria videamus, cur omnia sint paria peccata. Plane idem, inquit, et maxima quidem, qua fieri nulla maior potest. Dolere malum est: in crucem qui agitur, beatus esse non potest.

Cum id fugiunt, re eadem defendunt, quae Peripatetici, verba. Ea possunt paria non esse. Audeo dicere, inquit. Ita prorsus, inquam; Non igitur de improbo, sed de callido improbo quaerimus, qualis Q. Sed virtutem ipsam inchoavit, nihil amplius. Proclivi currit oratio. Recte, inquit, intellegis.

Quae fere omnia appellantur uno ingenii nomine, easque virtutes qui habent, ingeniosi vocantur. Nos autem non solum beatae vitae istam esse oblectationem videmus, sed etiam levamentum miseriarum. Si mala non sunt, iacet omnis ratio Peripateticorum. Si agere, ut a se dolores, morbos, debilitates repellant.

Ait enim se, si uratur, Quam hoc suave! dicturum. At modo dixeras nihil in istis rebus esse, quod interesset. Ne amores quidem sanctos a sapiente alienos esse arbitrantur. Tollitur beneficium, tollitur gratia, quae sunt vincla concordiae. Tertium autem omnibus aut maximis rebus iis, quae secundum naturam sint, fruentem vivere. Quid ergo attinet gloriose loqui, nisi constanter loquare?

Un conclusion

Quod cum dixissent, ille contra. Mihi vero, inquit, placet agi subtilius et, ut ipse dixisti, pressius. Frater et T. Tum ille timide vel potius verecunde: Facio, inquit. Videamus animi partes, quarum est conspectus illustrior; Num igitur eum postea censes anxio animo

Un texte étalé sur plusieurs colonnes en CSS3

Créer des colonnes

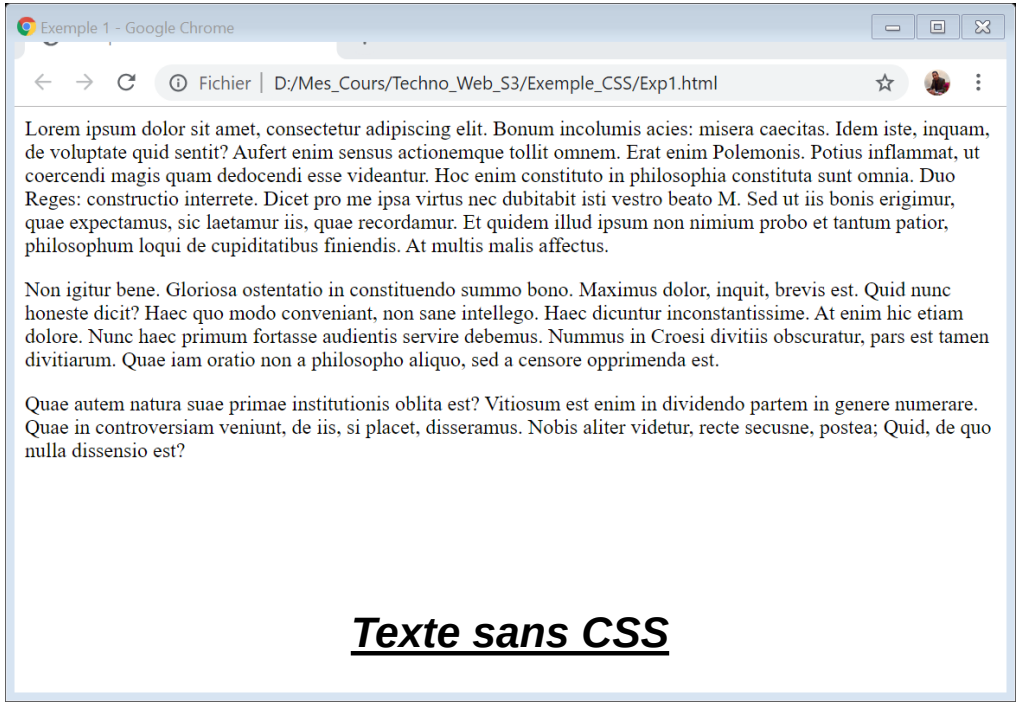
- Pour créer des colonnes, on peut utiliser ces 3 propriétés CSS :
  - **column-count** : indique sur combien de colonnes doit s'étaler le texte. Dans l'exemple précédent, il s'agissait de 4 colonnes.
  - **column-width** : indique la largeur souhaitée de chaque colonne (en pixels, cm... mais pas en pourcentage). Il est possible que le navigateur rende chaque colonne un peu plus large ou plus étroite en fonction de ses contraintes.
  - **columns** : c'est une super-propriété qui combine column-count et column-width (de la même manière que border combine border-width , border-style , border-color).

Créer des colonnes

- Les navigateurs supportent assez différemment ces propriétés.
- Seul Internet Explorer gère correctement ces propriétés.
- Pour les autres, il va falloir utiliser des préfixes :
  - Firefox : il faut commencer par -moz . Exemple :-moz-column-count , -moz-column-width ...
  - Opera, Safari, Google Chrome : il faut commencer par -webkit . Exemple : -webkit-column-count , -webkit-column-width ...
- Pour que votre code fonctionne sur tous les navigateurs, il va hélas falloir tripler les instructions CSS à chaque fois

Créer des colonnes

- **Exemple 1 :**



Texte sans CSS

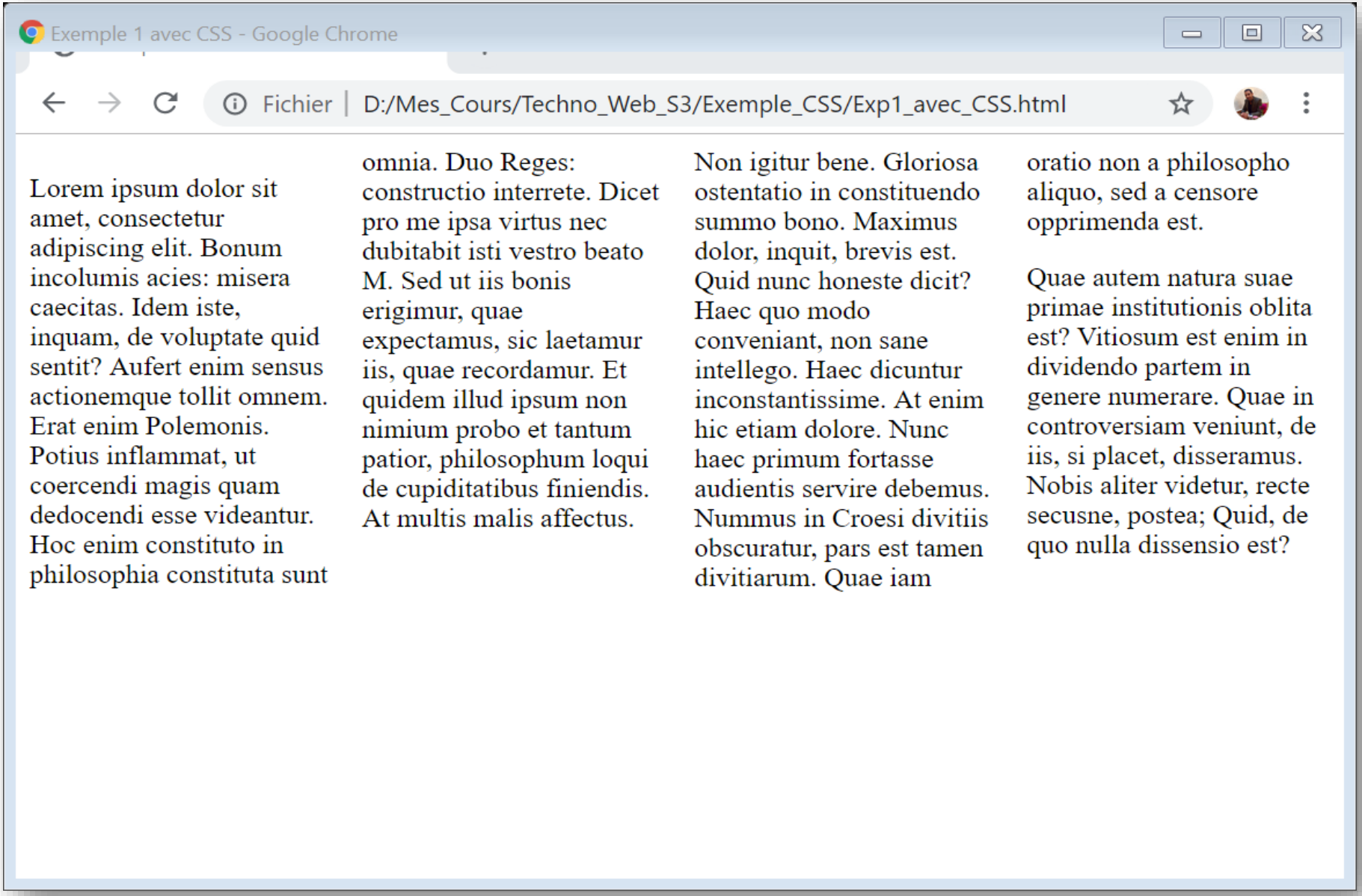
Version 1

```
section
{
    -webkit-column-count:4; /* Opera, Safari, Google Chrome */
    -moz-column-count: 4; /* Firefox */
    column-count: 4; /* Internet Explorer */
}
```

Version 2

```
section
{
    -webkit-columns: 4 100px;
    -moz-columns: 4 100px;
    columns: 4 100px;
}
```

Créer des colonnes



Séparations entre colonnes

- Nous avons la possibilité de modifier la séparation entre les colonnes de deux façons différentes :
  - **column-gap** : indique l'espace entre deux colonnes (en pixels, cm... mais pas en pourcentages).
  - **column-rule** : indique un liseré de séparation entre les colonnes. Il s'agit d'une super propriété qui ressemble beaucoup à border, et qui rassemble en fait les propriétés column-rule-color , column-rule-width et column-rule-style .

L'espace de séparation (column-gap)

- En plus du nombre de colonnes, si vous indiquez un "gap" (espace de séparation), vous pourrez espacer vos colonnes.
- Et là encore, il va falloir tripler vos propriétés pour que ça fonctionne sur tous les navigateurs !

```
-webkit-column-gap: 100px;
-moz-column-gap: 100px;
column-gap: 100px;
```

L'espace de séparation (column-gap)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Bonum incolumis acies: misera caecitas. Idem iste, inquam, de voluptate quid sentit? Aufert enim sensus actionemque tollit omnem. Erat enim Polemonis. Potius inflammat, ut coercendi magis quam dedocendi esse videantur. Hoc enim constituto in philosophia

constituta sunt omnia. Duo Reges: constructio interrete. Dicet pro me ipsa virtus nec dubitabit isti vestro beato M. Sed ut iis bonis erigimur, quae expectamus, sic laetamur iis, quae recordamur. Et quidem illud ipsum non nimium probo et tantum patior, philosophum loqui de cupiditatibus finiendis. At multis malis affectus.

Non igitur bene. Gloriosa ostentatio in constituendo summo bono. Maximus dolor, inquit, brevis est. Quid nunc honeste dicit? Haec quo modo conveniant, non sane intellego. Haec dicuntur inconstantissime. At enim hic etiam dolore. Nunc haec primum fortasse audientis servire debemus. Nummus in Croesi divitiis obscuratur, pars est tamen divitiarum. Quae

iam oratio non a philosopho aliquo, sed a censore opprimenda est.

Quae autem natura suae primae institutionis oblita est? Vitiosum est enim in dividendo partem in genere numerare. Quae in controversiam veniunt, de iis, si placet, disseramus. Nobis aliter videtur, recte secusne, postea; Quid, de quo nulla dissensio est?

Des colonnes séparées par un espace de 100px

Le liseré de séparation (column-rule)

- **column-rule** permet de liseré de séparation entre les colonnes :

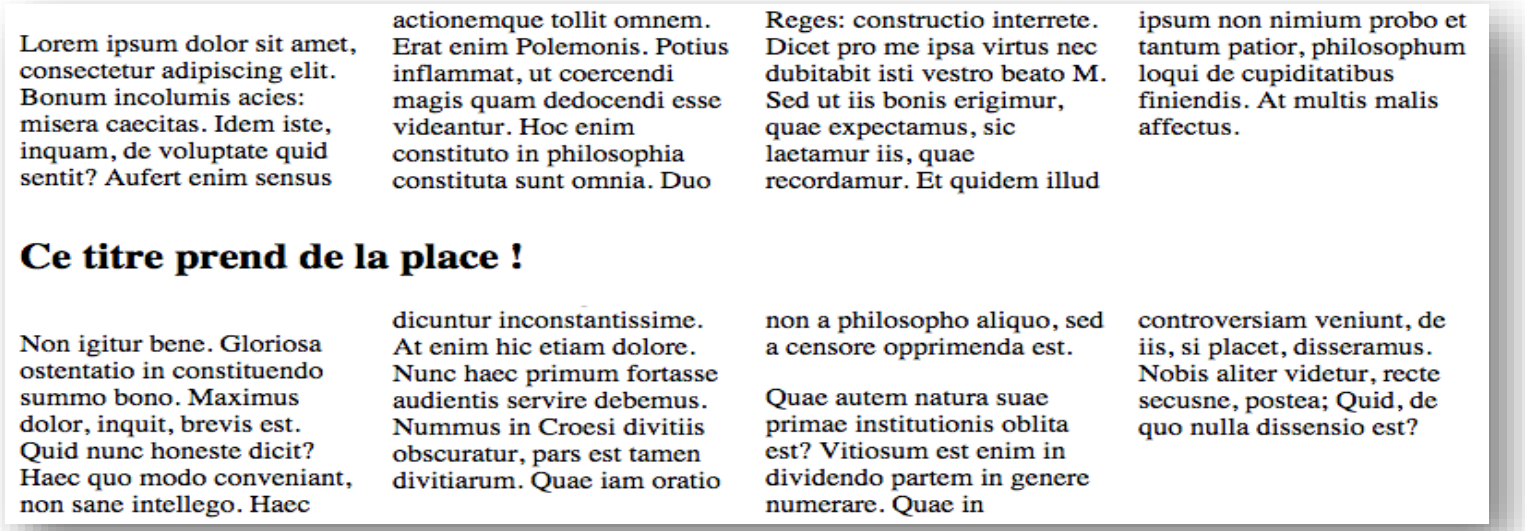
```
/* column-rule */
-webkit-column-rule: 2px dashed #F00;
-moz-column-rule: 2px dashed #F00;
column-rule: 2px dashed #F00;
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Bonum incolumis acies: misera caecitas. Idem iste, inquam, de voluptate quid sentit? Aufert enim sensus actionemque tollit omnem. Erat enim Polemonis. Potius inflammat, ut coercendi magis quam dedocendi esse videantur. Hoc enim constituto in philosophia constituta sunt omnia. Duo Reges: constructio interrete. Dicet pro me ipsa virtus nec dubitabit isti vestro beato M. Sed ut iis bonis erigimur, quae expectamus, sic laetamur iis, quae recordamur. Et quidem illud ipsum non nimium probo et tantum patior, philosophum loqui de cupiditatibus finiendis. At multis malis affectus. Non igitur bene. Gloriosa ostentatio in constituendo summo bono. Maximus dolor, inquit, brevis est. Quid nunc honeste dicit? Haec quo modo conveniant, non sane intellego. Haec dicuntur inconstantissime. At enim hic etiam dolore. Nunc haec primum fortasse audientis servire debemus. Nummus in Croesi divitiis obscuratur, pars est tamen divitiarum. Quae iam oratio non a philosopho aliquo, sed a censore opprimenda est.

Quae autem natura suae primae institutionis oblita est? Vitiosum est enim in dividendo partem in genere numerare. Quae in controversiam veniunt, de iis, si placet, disseramus. Nobis aliter videtur, recte secusne, postea; Quid, de quo nulla dissensio est?

Un texte sur plusieurs colonnes

- Si vous le souhaitez, vous pouvez faire en sorte qu'un élément s'étale sur plusieurs colonnes (par exemple un titre important).



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Bonum incolumis acies: misera caecitas. Idem iste, inquam, de voluptate quid sentit? Aufert enim sensus actionemque tollit omnem. Erat enim Polemonis. Potius inflammat, ut coercendi magis quam dedocendi esse videantur. Hoc enim constituto in philosophia constituta sunt omnia. Duo Reges: constructio interrete. Dicet pro me ipsa virtus nec dubitabit isti vestro beato M. Sed ut iis bonis erigimur, quae expectamus, sic laetamur iis, quae recordamur. Et quidem illud ipsum non nimium probo et tantum patior, philosophum loqui de cupiditatibus finiendis. At multis malis affectus.

Ce titre prend de la place !

Non igitur bene. Gloriosa ostentatio in constituendo summo bono. Maximus dolor, inquit, brevis est. Quid nunc honeste dicit? Haec quo modo conveniant, non sane intellego. Haec dicuntur inconstantissime. At enim hic etiam dolore. Nunc haec primum fortasse audientis servire debemus. Nummus in Croesi divitiis obscuratur, pars est tamen divitiarum. Quae iam oratio non a philosopho aliquo, sed a censore opprimenda est.

Quae autem natura suae primae institutionis oblita est? Vitiosum est enim in dividendo partem in genere numerare. Quae in controversiam veniunt, de iis, si placet, disseramus. Nobis aliter videtur, recte secusne, postea; Quid, de quo nulla dissensio est?

```
section h1
{
    /* Le titre s'étalera sur toutes les colonnes */
    -webkit-column-span: all;
    -moz-column-span: all;
    column-span: all;
}
```

Les sauts de colonne

- Parfois, vous voulez absolument éviter qu'un contenu soit "coupé en deux" parce qu'on arrive en bas de la colonne.
- Ou vous aimeriez qu'un texte commence obligatoirement en haut d'une colonne.
- C'est possible grâce à ces propriétés :
  - **break-before** : indique si on a le droit ou non de faire un saut de colonne avant le texte
  - **break-inside** : indique si on a le droit ou non de faire un saut de colonne à l'intérieur du texte
  - **break-after** : indiquer si on a le droit ou non de faire un saut de colonne après le texte

Les sauts de colonne

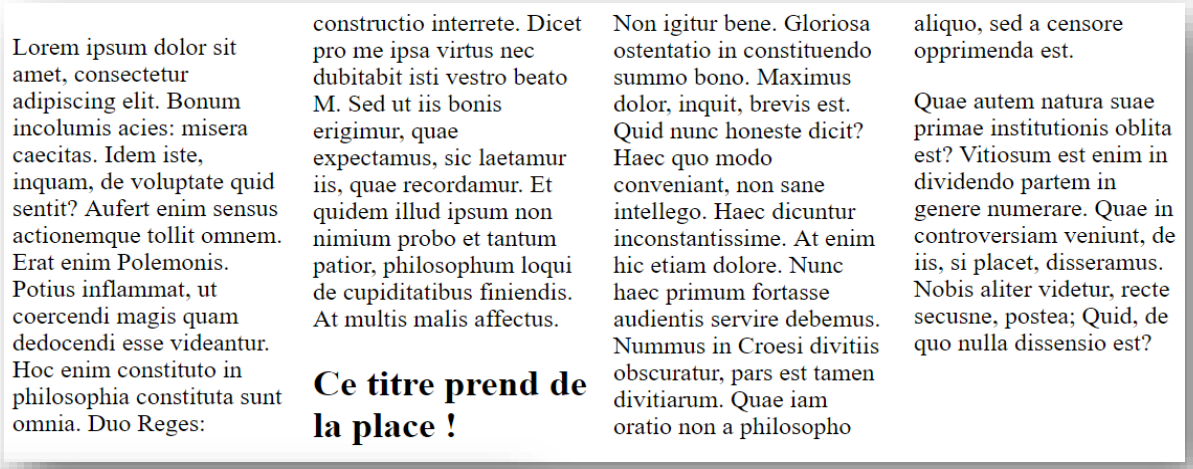
- Pour chacune de ces propriétés, plusieurs valeurs sont possibles. Les 3 que je vous recommande de retenir sont :
  - **auto** : le saut de colonne est autorisé mais n'est pas obligatoire (valeur par défaut).
  - **column** : force un saut de colonne.
  - **avoid-column** : interdit un saut de colonne.
- Quelques exemples :
  - Si vous écrivez **break-inside: avoid-column** , cela signifie qu'il est **interdit de faire un saut de colonne à l'intérieur du texte**.
  - Si vous écrivez **break-before: column** , cela **force un saut de colonne juste avant le texte**, pour que celui-ci se trouve en haut.

Les sauts de colonne

- Seuls Opera et Internet Explorer comprennent ces propriétés.
- Les navigateurs Google Chrome et Safari acceptent une syntaxe un peu différente :
  - **-webkit-column-break-before ,**
  - **-webkit-column-break-inside**
  - **-webkit-column-break-after.**
- Les valeurs sont aussi différentes : **always** pour forcer le saut de colonne, **avoid** pour l'éviter.

Les sauts de colonne

Exemple



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Bonum incolumis acies: misera caecitas. Idem iste, inquam, de voluptate quid sentit? Aufert enim sensus actionemque tollit omnem. Erat enim Polemonis. Potius inflammat, ut coercendi magis quam dedocendi esse videantur. Hoc enim constituto in philosophia constituta sunt omnia. Duo Reges: constructio interrete. Dicet pro me ipsa virtus nec dubitabit isti vestro beato M. Sed ut iis bonis erigimur, quae expectamus, sic laetamur iis, quae recordamur. Et quidem illud ipsum non nimium probo et tantum patior, philosophum loqui de cupiditatibus finiendis. At multis malis affectus.

Ce titre prend de la place !

Non igitur bene. Gloriosa ostentatio in constituendo summo bono. Maximus dolor, inquit, brevis est. Quid nunc honeste dicit? Haec quo modo conveniant, non sane intellego. Haec dicuntur inconstantissime. At enim hic etiam dolore. Nunc haec primum fortasse audientis servire debemus. Nummus in Croesi divitiis obscuratur, pars est tamen divitiarum. Quae iam oratio non a philosopho aliquo, sed a censore opprimenda est.

Quae autem natura suae primae institutionis oblita est? Vitiosum est enim in dividendo partem in genere numerare. Quae in controversiam veniunt, de iis, si placet, disseramus. Nobis aliter videtur, recte secusne, postea; Quid, de quo nulla dissensio est?

```
section h1
{
    break-before: column;
    -webkit-column-break-before: always;
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Bonum incolumis acies: misera caecitas. Idem iste, inquam, de voluptate quid sentit? Aufert enim sensus actionemque tollit omnem. Erat enim Polemonis. Potius inflammat, ut coercendi magis quam dedocendi esse videantur. Hoc enim constituto in philosophia constituta sunt omnia. Duo Reges: constructio interrete. Dicet pro me ipsa virtus nec dubitabit isti vestro beato M. Sed ut iis bonis erigimur, quae expectamus, sic laetamur iis, quae recordamur. Et quidem illud ipsum non nimium probo et tantum patior, philosophum loqui de cupiditatibus finiendis. At multis malis affectus.

Ce titre prend de la place !

Non igitur bene. Gloriosa ostentatio in constituendo summo bono. Maximus dolor, inquit, brevis est. Quid nunc honeste dicit? Haec quo modo conveniant, non sane intellego. Haec dicuntur inconstantissime. At enim hic etiam dolore. Nunc haec primum fortasse audientis servire debemus.

Nummus in Croesi divitiis obscuratur, pars est tamen divitiarum. Quae iam oratio non a philosopho aliquo, sed a censore opprimenda est.

Quae autem natura suae primae institutionis oblita est? Vitiosum est enim in dividendo partem in genere numerare. Quae in controversiam veniunt, de iis, si placet, disseramus. Nobis aliter videtur, recte secusne, postea; Quid, de quo nulla dissensio est?

Les dégradés

Introduction

- CSS permet de gérer les dégradés de couleur
- Différents type de degrés sont disponibles :
  - dégradés linéaires,
  - dégradés radiaux,
  - 2 couleurs,
  - 5 couleurs avec de la transparence
  - Etc ...
- Nous pouvons faire beaucoup de choses juste avec une simple ligne de CSS !
- Les dégradés peuvent s'appliquer sur le fond (background-image) de n'importe quel élément.

Introduction

- Les dégradés sont bien supportés par l'ensemble des navigateurs récents, pas besoin d'utiliser des préfixes (-moz, -webkit...).
- Voici un aperçu de ce qu'on peut faire (en une seule ligne de CSS !) :

Un dégradé réalisé en une seule ligne de CSS

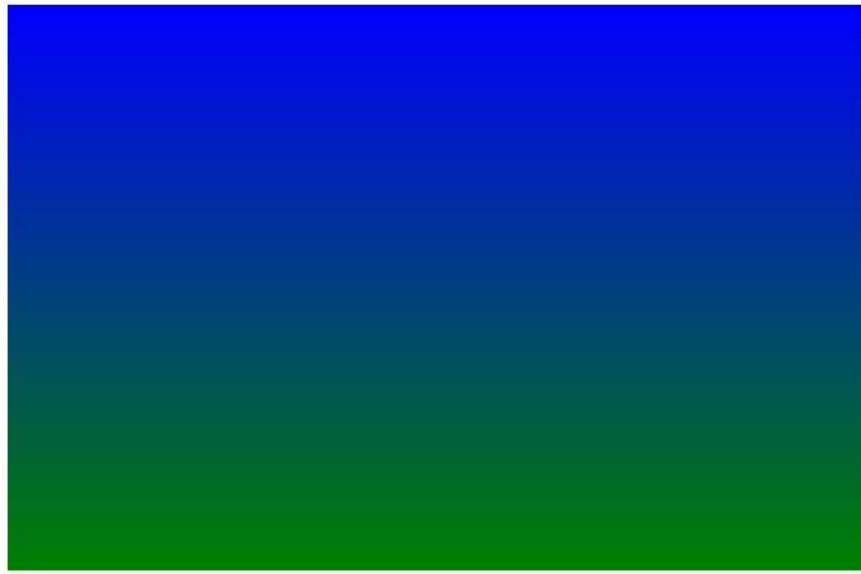
Les dégradés linéaires

- Un dégradé linéaire est un dégradé simple qui évolue sur une ligne.

```html
<html>
    <head>
        <meta charset="utf-8" />
        <title>Dégradés</title>
        <link rel="stylesheet" href="deg1.css" />
    </head>

    <body>
        <div></div>
    </body>
</html>
```

Page html



Résultat

```css
div
{
    background-image: linear-gradient(blue, green);

    /* Obligatoire si on veut "voir" le dégradé */
    width: 400px;
    height: 300px;
}
```

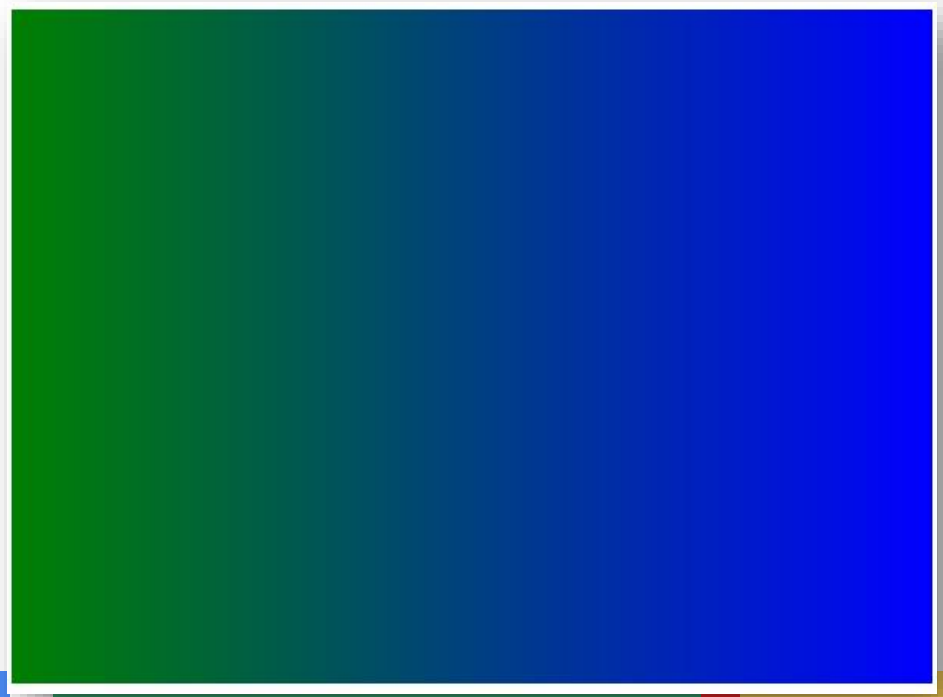
Code CSS

NB :
La valeur **linear-gradient** ne s'applique qu'à la propriété **background-image** (ou à la super-propriété **background**)

Modifier la direction du dégradé

- Par défaut, le dégradé va du haut vers le bas.
- Pour changer la direction il suffit d'ajouter un premier paramètre pour indiquer la direction du dégradé.
- Par exemple, "to left" signifie que le dégradé ira vers la gauche :

```
background-image: linear-gradient(to left,blue, green);
```



Modifier la direction du dégradé

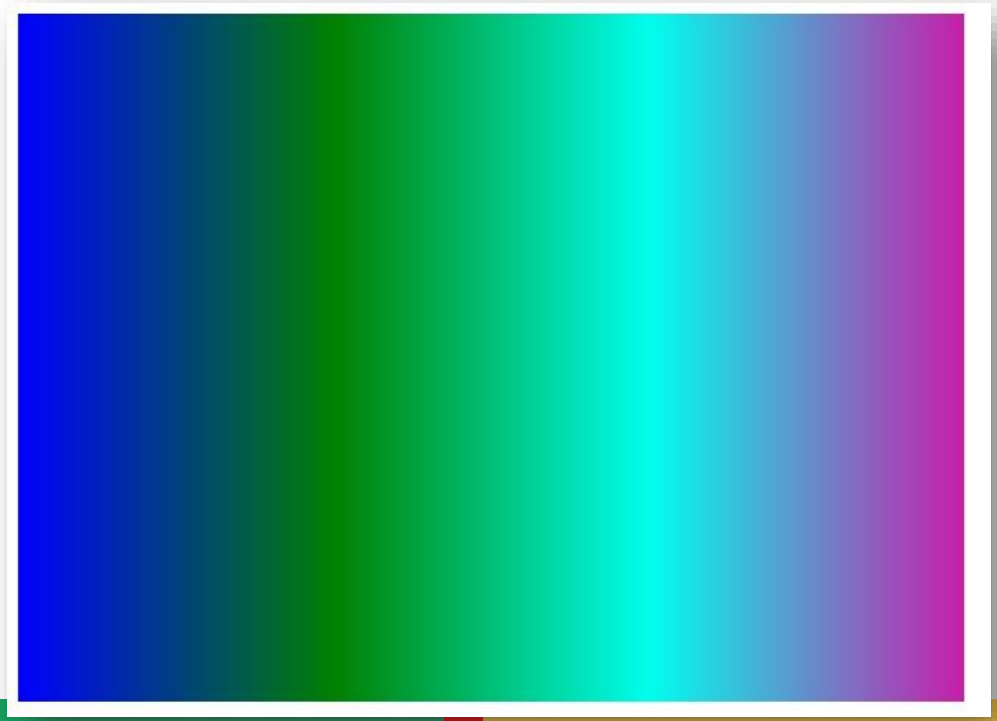
- Voici quelques syntaxes que vous pouvez utiliser pour le premier paramètre :
  - **to top** : vers le haut
  - **to bottom** : vers le bas (par défaut)
  - **to right** : vers la droite
  - **to left** : vers la gauche
  - **to top right** : vers en haut à droite
  - **to bottom left** : vers en bas à gauche
  - etc.

Augmenter le nombre de couleurs du dégradé

- CSS permet d'utiliser autant de couleurs que vous voulez dans votre dégradé
- Il suffit d'ajouter des couleurs à la suite.

```
background-image: linear-gradient(to right,
                  blue, green,
                  #0fe, rgb(200, 30, 170)
                  );
```

Résultat : un dégradé qui passe par 4 couleurs différentes



La transparence

- En CSS vous pouvez aussi utiliser la notation **RGBA** qui permet de définir de la transparence.
- La notation **RGBA** obéit aux mêmes règles de fonctionnement que la notation classique **RGB**, mis à part qu'une composante est ajoutée à la valeur :
- rgb(0,0,0) devient donc rgba(0,0,0,0). La dernière valeur (l'alpha) indiquant le degré d'opacité entre 0 et 1.
- Vous pouvez faire un dégradé qui va de l'opaque vers la transparence

La transparence

```
background-image: linear-gradient(to top,
                  rgba(0,0,0,0),
                  rgba(0,0,0,1));
```

Modifier l'espace occupé par chaque couleur

- Par défaut, les couleurs se répartissent équitablement dans le dégradé.
- Mais vous pouvez contrôler l’espace occupé par chaque couleur
- Ajouter un second paramètre à chaque couleur : une valeur en pourcentage ou pixels pour indiquer à quel endroit commence la nouvelle couleur par rapport à la couleur précédente.

Modifier l'espace occupé par chaque couleur

```
background-image: linear-gradient(
                  to right,
                  blue 0%,
                  green 30%);
```

La couleur bleue commence au début

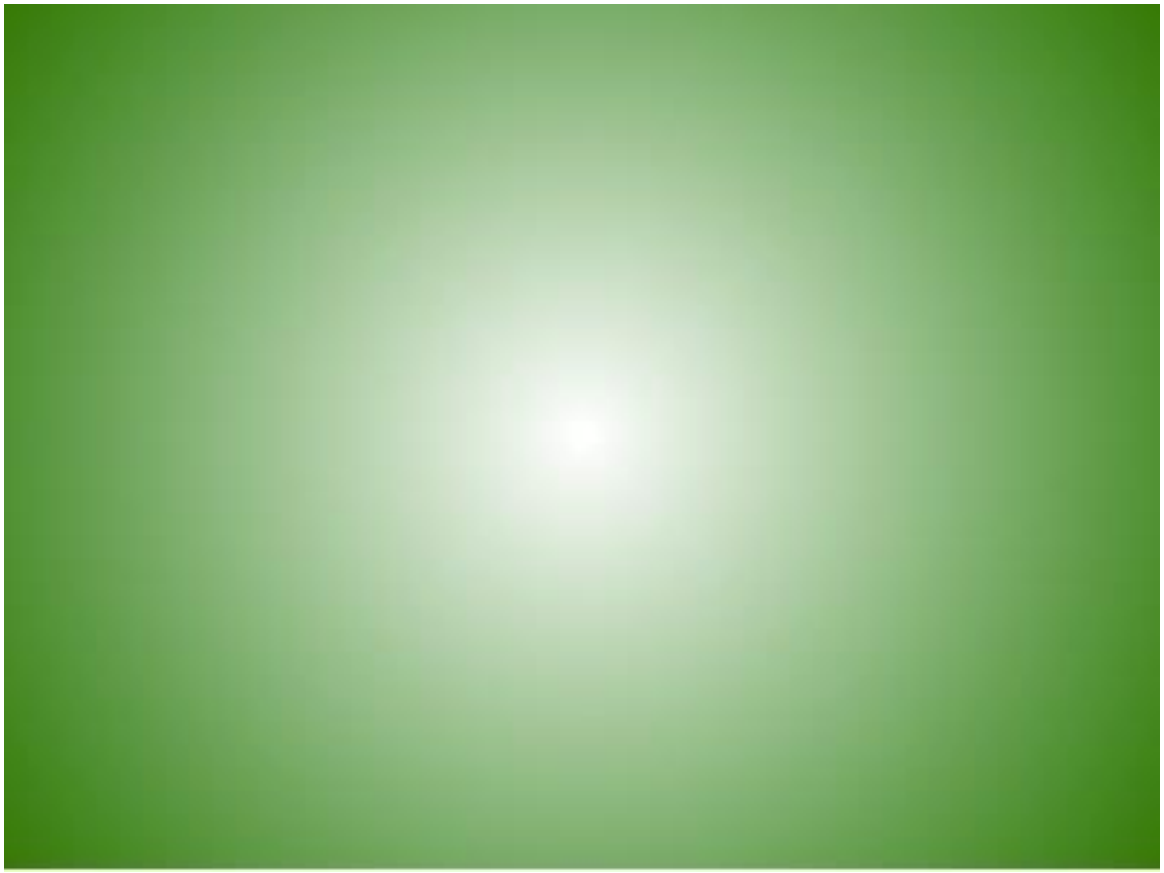
La couleur verte commence à 30% de l'espace occupé par défaut par la couleur bleue

```
background-image: linear-gradient(
                  to right,
                  blue,
                  green 70px);
```

Valeur absolue en pixels

Les dégradés radiaux

- Une image vaut mieux qu'un long discours

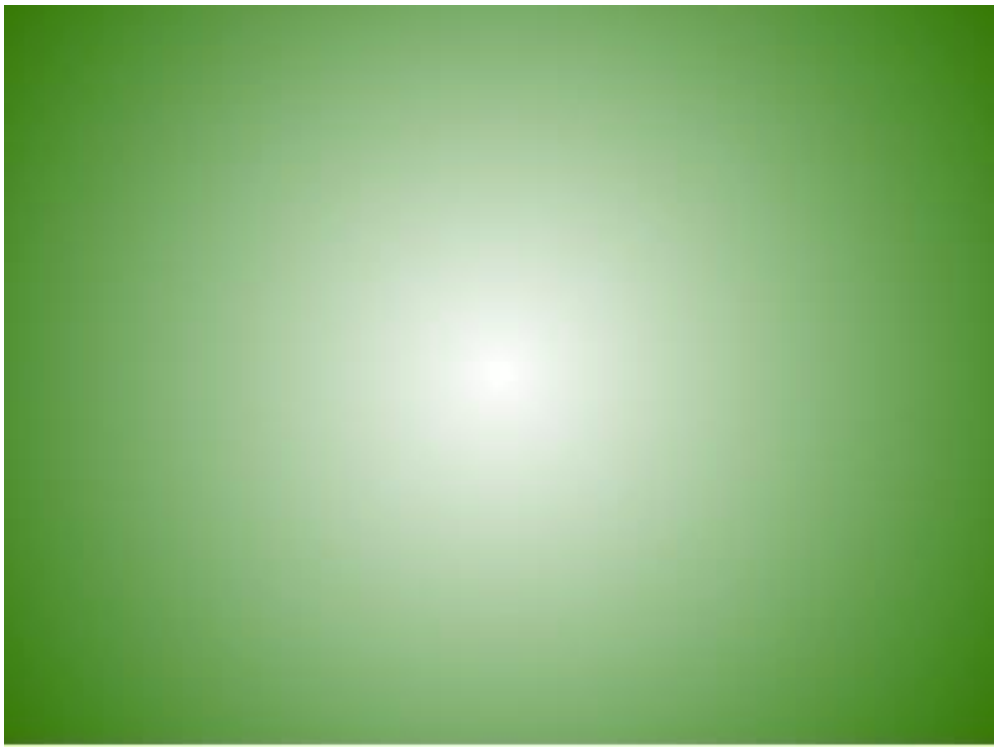


Un dégradé radial

Le dégradé part un peu dans toutes les directions

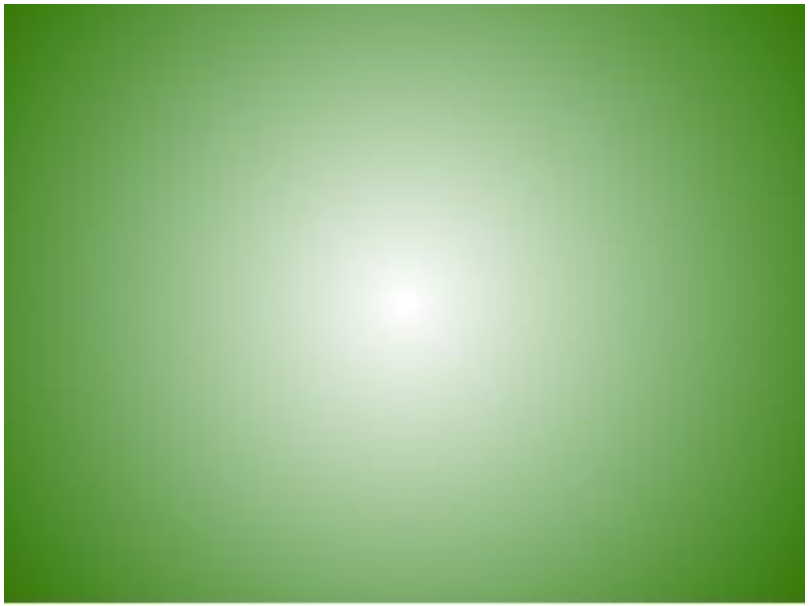
Les dégradés radiaux

```
background-image: radial-gradient(
                  circle at center,
                  white, green);
```

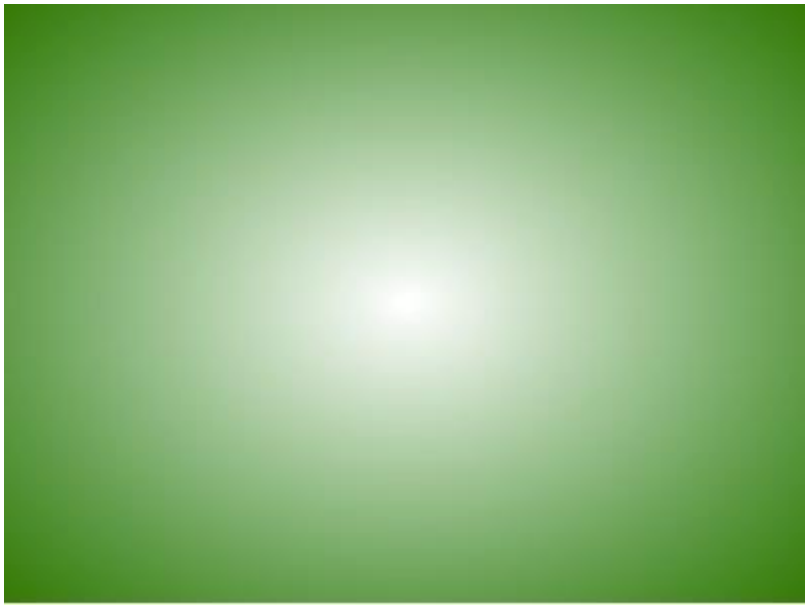


Les dégradés radiaux

- Il existe deux formes de dégradé radiaux :
  - **circulaire** : on utilise le mot –clé **circle** . On observe un véritable cercle dans le dégradé.
  - **elliptique** : on utilise le mot **–**clé **ellipse** . On observe un cercle "aplati" si la zone du dégradé est aplatie.



Un dégradé circulaire

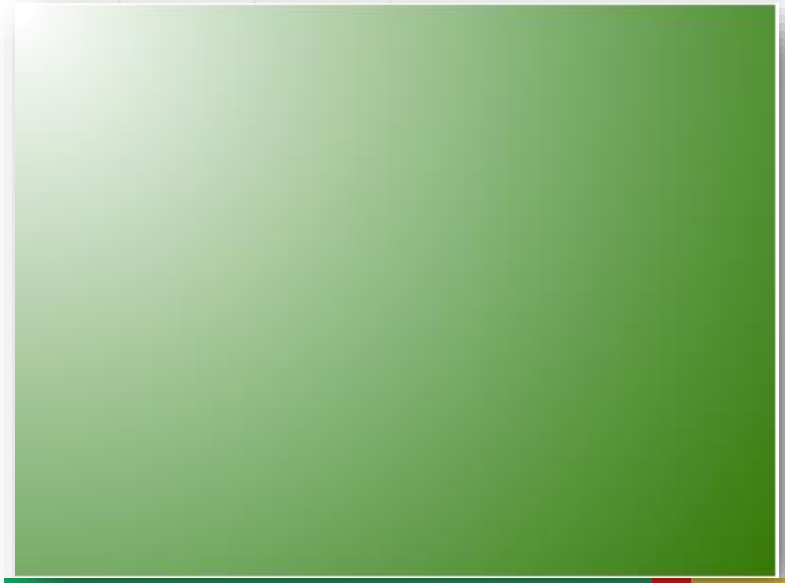


Un dégradé elliptique (ellipse)

Positionner le dégradé

- Vous devez aussi indiquer à quel endroit commence le dégradé.
- Vous pouvez indiquer "top left" (en haut à gauche), "bottom right" (en bas à droite)... mais aussi "center" (au centre).

```
background-image: radial-gradient(
                  circle at top left,
                  white, green);
```

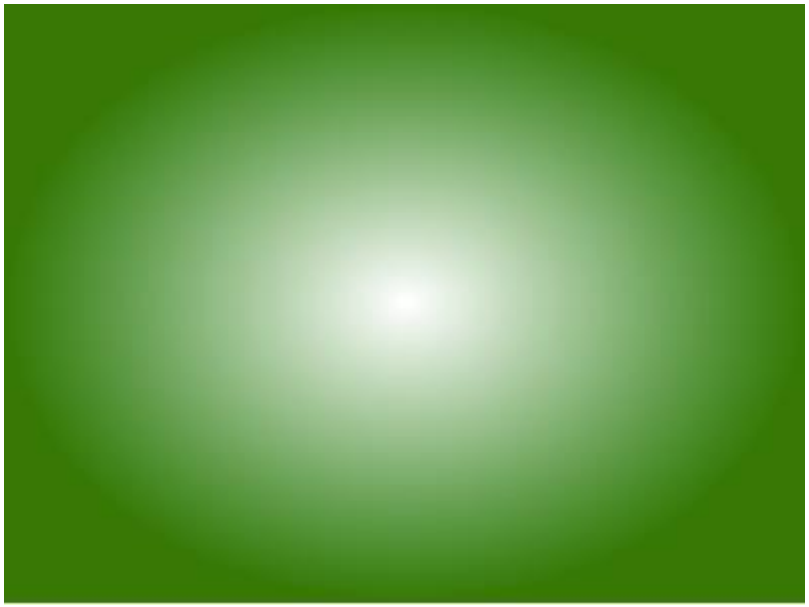


Indiquer jusqu'où va le dégradé

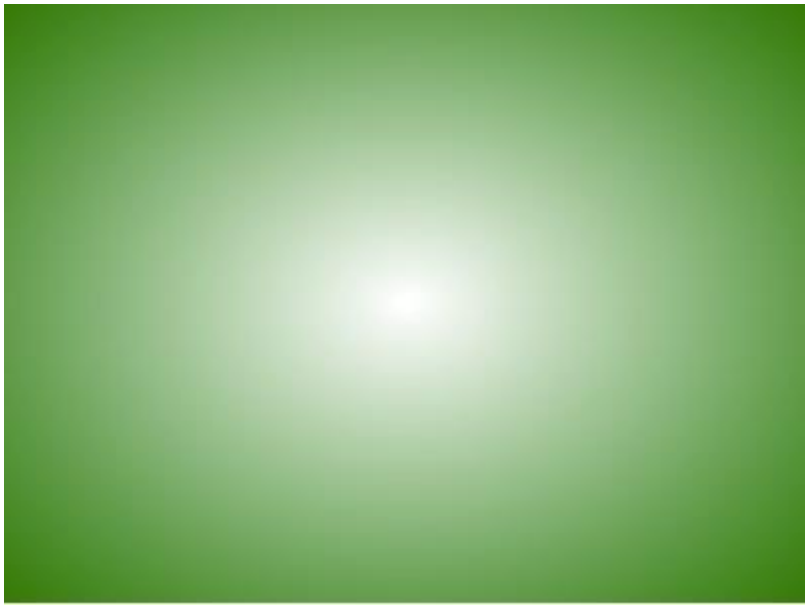
- Vous pouvez aussi indiquer jusqu'où se déroule l'effet de dégradé.
- Il faut rajouter un premier terme au tout début :
  - **closest-corner** : vers le coin le plus proche
  - **closest-side** : vers le côté le plus proche
  - **farthest-corner** : vers le coin le plus éloigné
  - **farthest-side** : vers le côté le plus éloigné

Indiquer jusqu'où va le dégradé

```
background-image: radial-gradient(
closest-side
ellipse at center,
white, green);
```



Un dégradé radial en closest-side



Un dégradé radial en closest-corner

Les dégradés répétitifs

- Il existe une autre technique qui permet d'avoir des dégradés qui se répètent.
- Pour cela, on peut utiliser les valeurs suivantes :
  - **repeating-linear-gradient** : un dégradé linéaire qui se répète
  - **repeating-radial-gradient** : un dégradé radial qui se répète

Les dégradés répétitifs

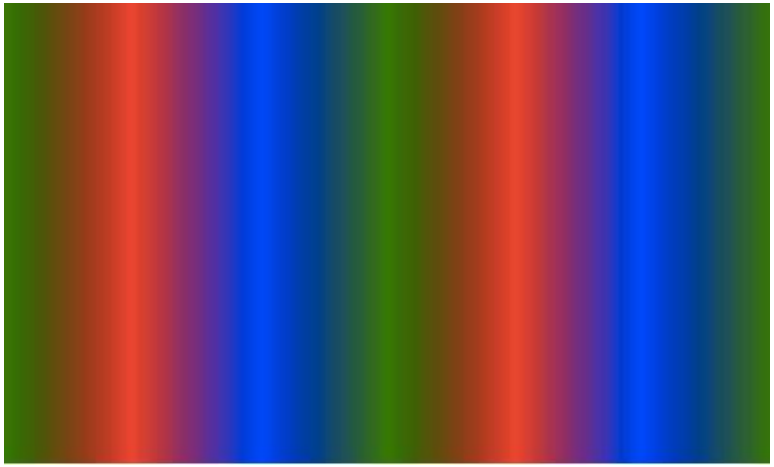
```
background-image: repeating-linear-gradient(
                  white, green 100px);
```

Un dégradé linéaire qui se répète, qui va du blanc vers le vert, et le vert s'arrête au bout de 100 pixels.

Les dégradés répétitifs

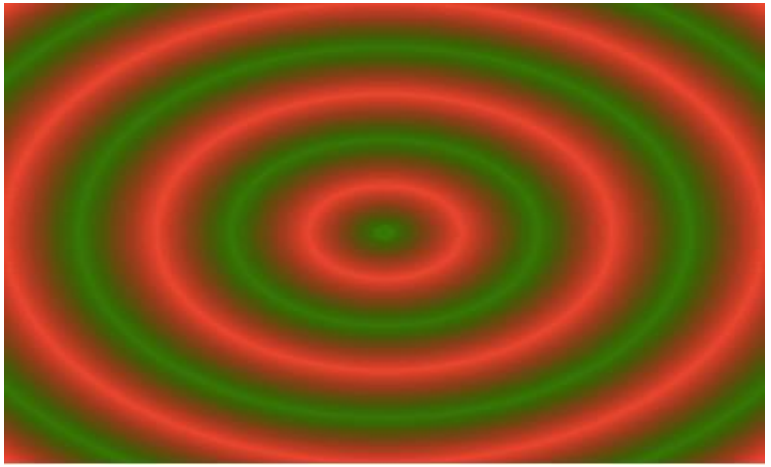
Dégradé linéaire répétitif avec plusieurs couleurs

```
background-image: repeating-linear-gradient(
                  to right,
                  green, red, blue,
                  green 200px);
```



Un dégradé radial elliptique répétitif

```
background: repeating-radial-gradient(
            farthest-side
            ellipse at center,
            green, red,
            green 80px);
```



Les transformations 2D

Introduction

- Les transformations 2D c'est l'ensemble des opérations qui permettent d'**étirer**, **déplacer**, **faire tourner** un objet de la page.
- Il existe plusieurs types de transformations :
  - L'agrandissement et le rétrécissement ;
  - Le déplacement (translation);
  - La rotation ;
  - La transformation oblique ;
- Applicable sur tous les objets de la page : textes, images, menus.

Introduction

Exemple d'une image qui a subi de multiples transformations en CSS

La rotation

```html
<html>
    <head>
        <meta charset="utf-8" />
        <title>Transformations</title>
        <link rel="stylesheet" href="trans.css" />
    </head>

    <body>
        <div><img src="nature.jpg" alt="" id="photo" /></div>
    </body>
</html>
```

```css
#photo
{
    /* Safari */
    -webkit-transform: rotate(10deg);
     /* Tous les autres navigateurs */
    transform: rotate(10deg);
}
```

L'origine de la transformation

- Toute transformation doit partir d'un point précis.
- Par défaut, la transformation se fait à partir du centre de l'élément.
- Par exemple, imaginez qu'on a placé une punaise au centre de la photo et qu'on la fait tourner autour de cette punaise :

L'origine de la transformation

- On peut modifier l'origine de la transformation (l'emplacement de la punaise dans notre cas).
- Par exemple, on peut partir du coin en haut à gauche :

L'origine de la transformation

- Pour modifier l'origine de la transformation, on a recours à **transform-origin**.
- Cette propriété est moins bien supportée par les navigateurs et il faudra ajouter tout un tas de préfixes :

```
#photo
{
    -webkit-transform-origin: 0 0;/* Safari, Chrome */
    -ms-transform-origin: 0 0; /* IE */
    -moz-transform-origin: 0 0; /* Firefox */
    -o-transform-origin: 0 0; /* Opera */
    transform-origin: 0 0; /* La transformation partira
                              du coin en haut à gauche */

    transform: rotate(10deg);
}
```

Deux valeurs en pixels ou pourcentages.

➔ **"0 0"** : la transformation part du point en haut à gauche de l'élément.

➔ **"50% 50%"** : valeur par défaut ce qui correspond au centre de l'élément

L'agrandissement (et le rétrécissement)

- La fonction **scale()** permet d'agrandir un élément (ou bien le rétrécir) :

```
#photo
{
     transform: scale(1.3);
}
```

Agrandir de 1,3 fois plus que la taille normale

```
#photo
{
     transform: scale(0.6);
}
```

Réduire de 0,6

```
#photo
{
    transform: scale(1.3 , 0.6);
}
```

Agrandira de 1,3 fois sur l'axe des X et réduira à 0,6 sur l'axe des Y

NB : Les fonctions **scaleX**() et **scaleY**() permettent d'appliquer la modification uniquement sur un des axes

La translation

- La translation permet de déplacer un objet comme bon vous semble sur la page

```
#photo
{
    transform: translate(30px, 90px);
}
```

Déplacera l'objet de 30px sur l'axe des X (de gauche à droite) et de 90px sur l'axe des Y (de haut en bas)

NB : Les fonctions **translateX**() et **translateY**() permettent déplacer l'élément uniquement sur un axe

La transformation oblique

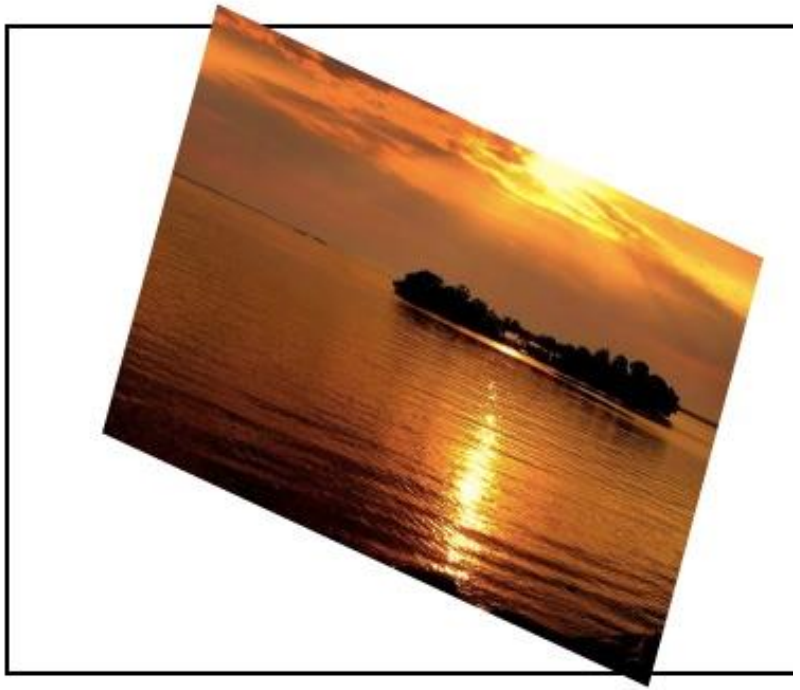
- Ce type de transformation "étire" un des côtés du bloc pour en faire un losange.
- On utilise pour cela les fonctions **skewX**() et **skewY**()

```
#photo
{
    transform: skewX(20deg);
}
```

Combinez les transformations !

- CSS permet de combiner plusieurs Les transformations.
- Il suffit d'écrire les fonctions les unes à la suite des autres sur la même ligne, séparées à chaque fois par un espace.

```
#photo
{
    transform: scale(0.7) rotate(15deg) skewY(10deg);
}
```



NB :

L'ordre des fonctions a de l'importance.

Les transitions CSS

Introduction

- Auparavant, il fallait programmer en JavaScript des fonctions parfois complexes pour pouvoir faire bouger des éléments sur la page.
- Aujourd'hui, grâce à CSS3, il est possible d'effectuer des animations avec quelques lignes de CSS.
- Les transitions nous permettent de modifier de façon progressive tout un ensemble de propriétés d'un objet.

Introduction

- Les **transitions** :
  - une animation basique qui fait passer un objet d'un état A à un état B.
  - Par exemple, on peut dire : "Déplace cet objet de la gauche vers la droite de l'écran tout en changeant sa couleur".
- Les **animations** :
  - Une combinaison de transitions les unes à la suite des autres.
  - Par exemple, on peut dire : "Déplace cet objet de la gauche vers la droite de l'écran puis déplace-le vers le bas de l'écran puis change sa couleur de fond vers le bleu."

Les bases des transitions CSS

- Le CSS dispose d'un nombre important de propriétés qui permettent de modifier l'apparence d'un élément :
  - **color** pour la couleur du texte,
  - **background-color** pour la couleur de fond,
  - **width** pour la largeur d'un bloc,
  - …etc.
- Deux propriétés essentielles de transitions en CSS3 :
  - **transition-property** : indique quelle propriété CSS est animée.
  - **transition-duration** : indique combien de temps dure l'animation

Les bases des transitions CSS

```html
<html>
    <head>
        <meta charset="utf-8" />
        <title>Transformations</title>
        <link rel="stylesheet" href="Transition.css" />
    </head>

    <body>
        <div></div>
    </body>
</html>
```

```css
div
{
    width: 200px;
    height: 150px;
    background-color: blue;
}

div:hover
{
    background-color: green;
}
```

Voir page

Les bases des transitions CSS

- Ajoutons une transition sur **background-color** pour indiquer qu'on doit animer l'objet progressivement du bleu vert le vert :

```css
div
{
    width: 200px;
    height: 150px;
    background-color: blue;
    /* Active la transition sur background-color */
    transition-property: background-color;
    /* La transition dure 1s */
    transition-duration: 1s;
}

div:hover
{
    background-color: green;
}
```

Voir page

Configuration de la transition

- Deux propriétés pour configure la transition :
  - **transition-delay** : indique au bout de combien de temps commence l'animation.
  - **transition-timing-function** : indique quel type de transition on veut voir. D'une certaine façon, cela détermine quelle est "l'accélération" de l'effet d'animation.

Configuration de la transition

- **transition-delay**
  - Délai avant le lancement de la transition
- Cela vous permet de retarder le début de la transition.

```css
div
{
    width: 200px;
    height: 150px;
    background-color: blue;
    transition-property: background-color;
    transition-duration: 1s;
    /* Démarre au bout de 0,4s */
    transition-delay: 0.4s;
}

div:hover
{
    background-color: green;
}
```

La transition commence au bout de 0,4s seulement

Configuration de la transition

- **transition-timing-function**
  - l'accélération de l'animation
- Cette propriété indique comment l'objet doit transiter d'un état à un autre.
- Les valeurs prédéfinies possibles sont :
  - **ease** : rapide au début et ralenti à la fin.
  - **linear** : vitesse constante.
  - **ease-in** : lent au début et accélère de plus en plus à la fin.
  - **ease-out** : rapide au début et de plus en plus lent à la fin.
  - **ease-in-out** : départ et fin lents.

Configuration de la transition

```css
div
{
    width: 200px;
    height: 150px;
    background-color: blue;
    transition-property: transform;
    transition-duration: 1s;
    /* Transition de type ease */
    transition-timing-function: ease;
}

div:hover
{
    /* On déplace l'objet */
    transform: translateX(200px);
}
```

Voir page

La même transition, de type "ease-in" Voir page

La super-propriété transition

- Il existe une super-propriété qui combine toutes ces autres propriétés : **transition**.

```
div
{
    /* Transition "ease" de 2s sur background-color */
    transition: background-color 2s ease;
}
```

La super-propriété transition

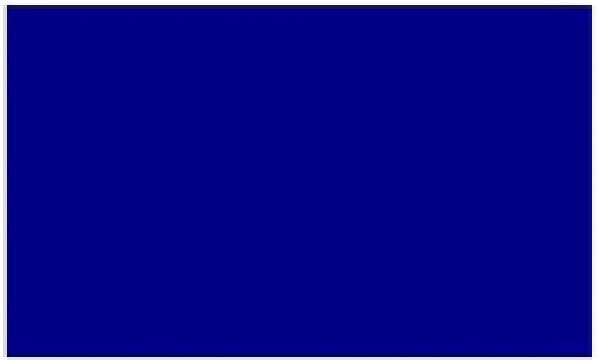
- Il est aussi possible d'avoir deux effets de transition en parallèle avec des paramètres différents.
- Il suffit de les séparer par des virgules

```
div
{
    /* Transition "ease" de 2s sur background-color
    et transition linéaire de 1s sur la largeur en parallèle */
    transition: background-color 2s ease, width 1s linear;
}
```

Les animations CSS

Introduction

- Les animations CSS sont en quelque sorte des "super-transitions".
- Là où la transition se contentait de transformer progressivement un objet d'un état A à un état B, l'animation permet de passer par autant d'états que vous voulez (A, B, C, D)



Les bases de l'animation CSS

- Pour faire une animation en CSS, vous devez :
  - Définir les étapes de l'animation à l'aide d'une directive CSS **@keyframes** qui est un peu particulière.
  - Appliquer la propriété **animation** sur l'élément que vous voulez animer (lors d'un :hover par exemple au survol).

Les bases de l'animation CSS

- **Définir les étapes de l'animation:**
  - La définition des étapes de l'animation se fait dans un coin à part dans le fichier CSS
- Exemple :

```
@keyframes masuperanimation {
    0% {
        transform: translateX(0px);
    }

    50% {
        transform: translateX(150px);
    }

    100% {
        transform: translateX(150px) rotate(30deg);
    }
}
```

Définition d'une animation appelé « masuperanimation »

A 0% (au début), l'objet n'aura pas bougé

A 50% (la moitié), il aura bougé de 150px

A 100% (à la fin), il aura bougé de 150px ET tourné de 30 degrés

Les bases de l'animation CSS

- **Appliquer l'animation à un objet**
  - Utiliser la propriété **animation** sur l'objet

```css
div
{
    margin: 100px;
    width: 200px;
    height: 150px;
    background-color: #008;
}

div:hover
{
    /* On utilise "ma super animation" définie plus tôt */
    animation: masuperanimation 2s;
}
```

Quelques propriétés CSS des animations

- **animation-name** : le nom de l'animation à utiliser.
- **animation-duration** : la durée de l'animation. Ex : 2s, 350ms...
- **animation-delay** : délai avant le démarrage de l'animation. Ex : 2s, 350ms...
  - Valeur négatif pour qu'on ait l'impression que l'animation a déjà démarré depuis un certain temps.
- **animation-timing-function** : l'accélération de l'animation.
  - Contrôler la vitesse de l'animation à différentes étapes (Partie transition) . Ex : ease, linear, ease-in, ease-out, ease-in-out, cubic-bezier(), etc.
- **animation-iteration-count** : nombre de répétitions de l'animation.
  - Avec "infinite", l'animation sera jouée en continu. Par défaut, la valeur est 1 (l'animation n'est jouée qu'une fois).
- **animation-direction** : permet de faire revenir l'animation en sens inverse avec la valeur "**alternate**", ou juste une fois à l'envers avec "**reverse**".
  - Par défaut avec "**normal**", une fois l'animation arrivée au bout, l'objet revient d'un coup à sa position initiale (ce qui peut être un peu surprenant).
- **animation-fill-mode** : permet d'indiquer à l'ordinateur comment l'objet doit s'afficher avant et après l'animation.
  - Doit-il revenir à la position initiale ? A la position finale ?
  - Les valeurs possibles sont notamment
  - "**none**" (par défaut), "**forwards**" (pour que l'élément reste à sa position finale) et "**backwards**" (pour que l'élément revienne à sa position initiale).

Une animation par étapes

- La propriété **animation-timing-function** fonctionne comme **transition-timing-function**
  - Elle permet de définir l'accélération de l'animation.
  - Admettre les même valeurs : ease, linear, ease-in, ease-out, ease-in-out et cubic-Bezier
- Autres proprietés :
  - **step-start** : l'objet prend son état final dès que l'animation commence. En clair, on "saute" directement à l'étape 100%.
  - **step-end** : l'objet prend son état final à la fin du délai de l'animation.
  - **steps(X)** : l'objet fait X paliers à chaque étape.

Une animation par étapes

```css
@keyframes movemydiv {
    from {
        transform: translateX(0px);
    }

    30% {
        transform: translateX(150px);
    }

    60% {
        transform: translateX(150px) rotate(30deg);
    }

    80% {
        transform: translate(100px, 80px) rotate(30deg);
    }

    to {
        transform: translate(100px, 80px) rotate(30deg);
        border-radius: 30px;
        background-color: red;
    }
}
```

```css
div
{
    margin: 100px;
    width: 200px;
    height: 150px;
    background-color: #008;
}

div:hover
{
    animation: movemydiv 2s steps(3); /* 3 étapes */
}
```

Animation à 3 images par étape.

Animation en continu

- Pour jouer l'animation en continu, on aura besoin de :
  - Définir **animation-iteration-count** à "**infinite**", pour que l'animation se joue à l'infini
  - Définir **animation-direction** à "**alternate**", pour faire en sorte que l'animation revienne sagement à sa position initiale sans "saut".

```
div:hover
{
    /* Boucle d'animation ! */
    animation: movemydiv 2s linear alternate infinite;
}
```

Animation en continu

```css
@keyframes movemydiv {
    from {
        transform: translateX(0px);
    }

    30% {
        transform: translateX(150px);
    }

    60% {
        transform: translateX(150px) rotate(30deg);
    }

    80% {
        transform: translate(100px, 80px) rotate(30deg);
    }

    to {
        transform: translate(100px, 80px) rotate(30deg);
        border-radius: 30px;
        background-color: red;
    }
}
```

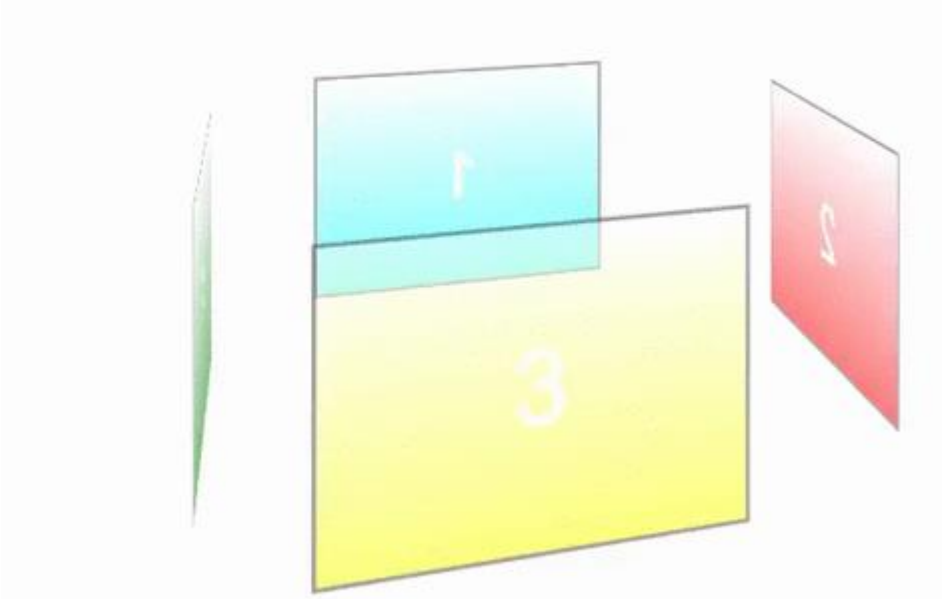
```css
div
{
    margin: 100px;
    width: 200px;
    height: 150px;
    background-color: #008;
}

div:hover
{
    animation: movemydiv 2s linear alternate infinite; /* Boucle d'animation ! */
}
```

Les transformations 3D

Introduction

- Rappel :
  - Vous savez modifier l'apparence des objets de la page en 2D en CSS : ce sont les **transformations**
  - Vous savez faire bouger un objet pour qu'il passe d'un état à un autre : ce sont les transitions
  - Vous savez combiner plusieurs transitions à la suite : ce sont les **animations**
- En CSS vous pouvez donner un effet 3D à vos objets dans la page.



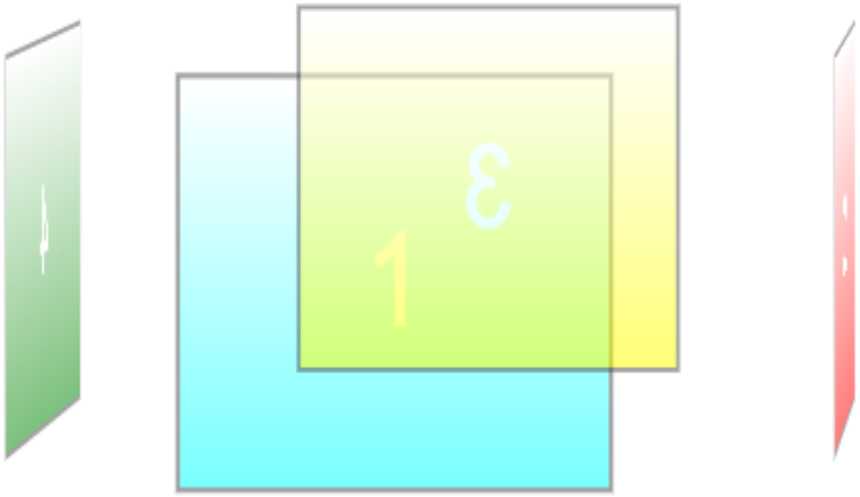
Définition la perspective

- La **perspective** est l’élément qui permet de donner l'impression que c'est en 3D.
- Elle permet de donner l'impression de profondeur à votre scène 3D.
- Sans perspective, pas de 3D sur votre écran.
- La perspective indique la distance de votre œil avec la scène.
  - Plus la perspective est faible, plus on est près.
  - Plus la perspective est élevée, plus on est loin.
-

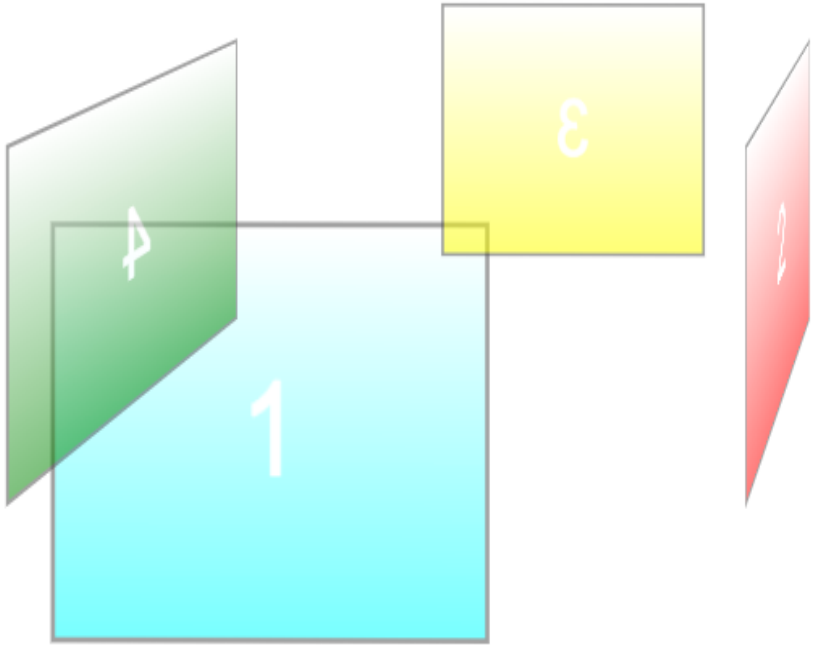
Définition la perspective

- La perspective se définit avec la propriété **perspective** dans un élément conteneur : une globale, ou même carrément <body>

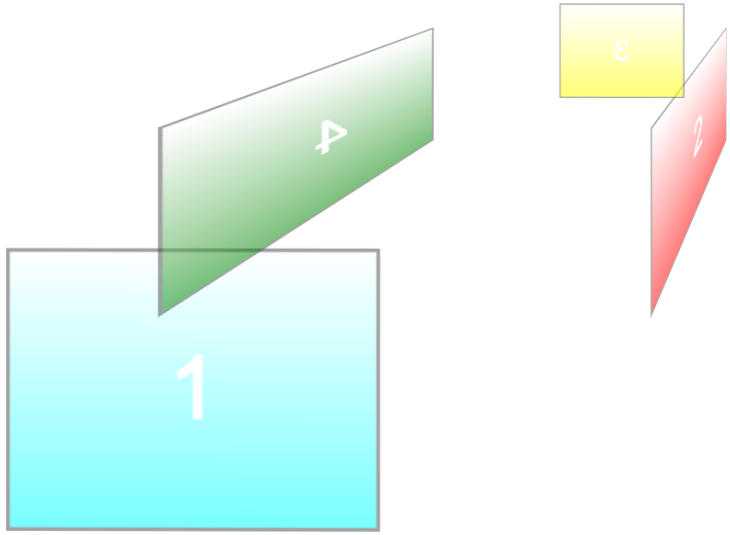
```
body
{
    perspective: 800px;
}
```



Une perspective de 3000 pixels : peu de profondeur



Une perspective de 800 pixels



Une perspective de 400 pixels : beaucoup de profondeur

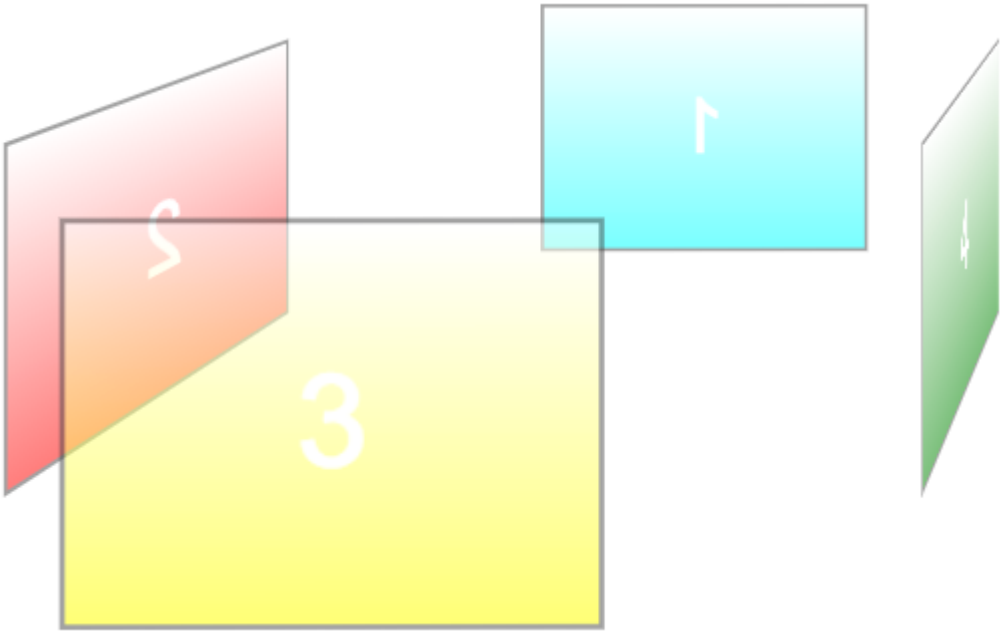
Définition l'origine du point de fuite

- Dans toute scène 3D avec perspective, il y a ce qu'on appelle un **point de fuite**.
- C'est le point d'ancrage de la scène, ce qui va donner l'impression qu'on regarde d'un certain côté la scène.
- L'origine du point de fuite se définit avec **perspective-origin** avec normalement deux valeurs : l'axe des **X** et l'axe des **Y**.

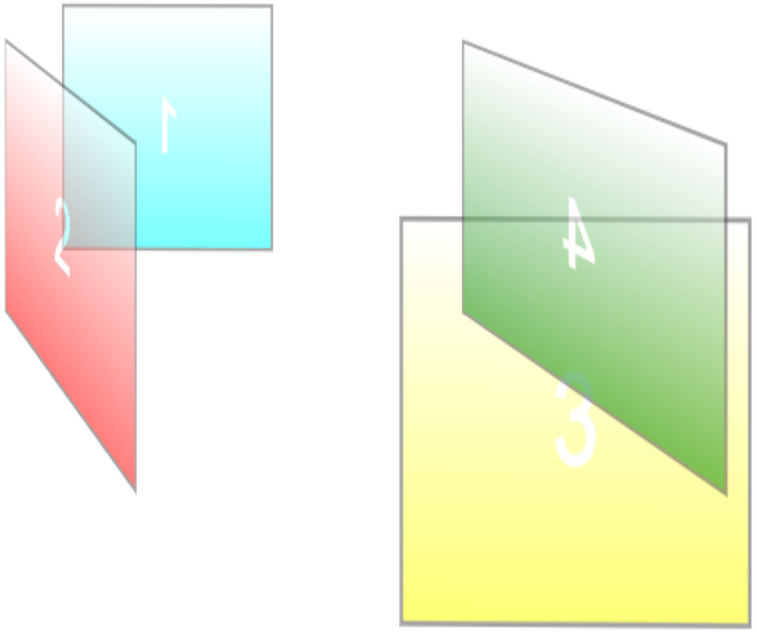
```
body
{
    perspective: 800px;
    /* Point de fuite placé aux coordonnées (20, 70) */
    perspective-origin: 20px 70px;
}
```

- Vous pouvez aussi employer les valeurs **top** (haut), **bottom** (bas), **center** (centre), **left** (gauche), **right** (droite).

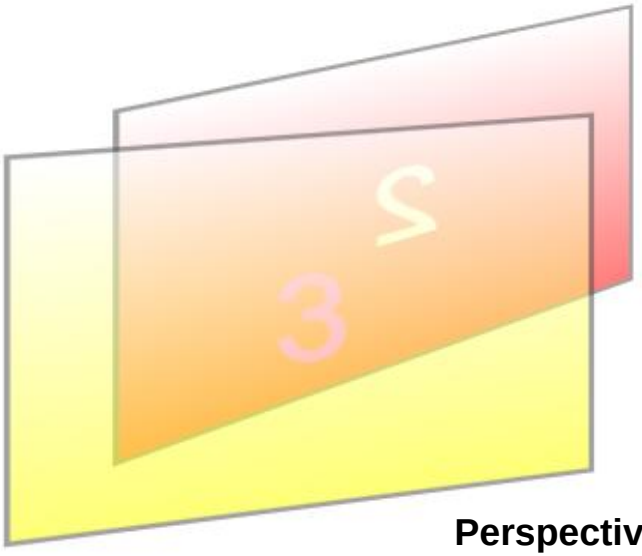
```
body
{
    perspective: 800px;
    perspective-origin: top left;
}
```

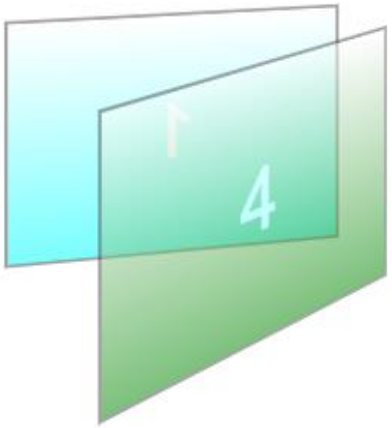


Perspective-origin: center



Perspective-origin: top left

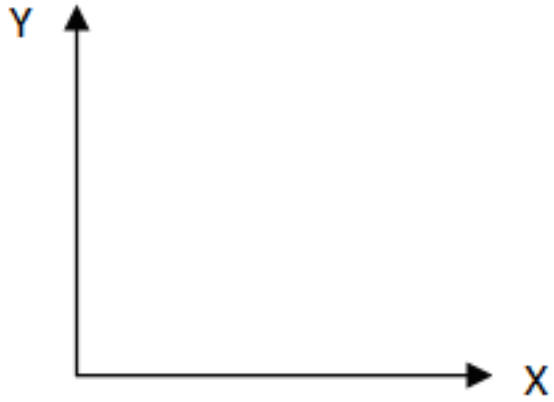




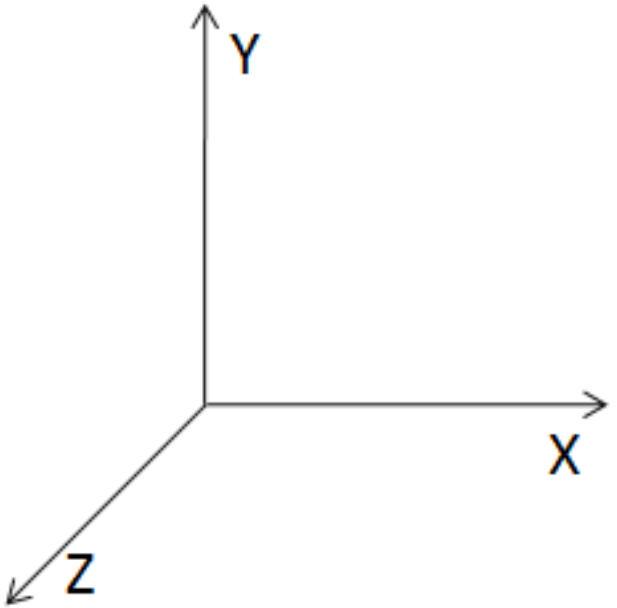
Perspective-origin: right

La base des transformations 3D

- En 2D, nous travaillons sur un plan avec 2 axes : X et Y.
- En 3D, nous avons un 3ieme axe Z : profondeur



Un plan en 2D a 2 axes : X et Y



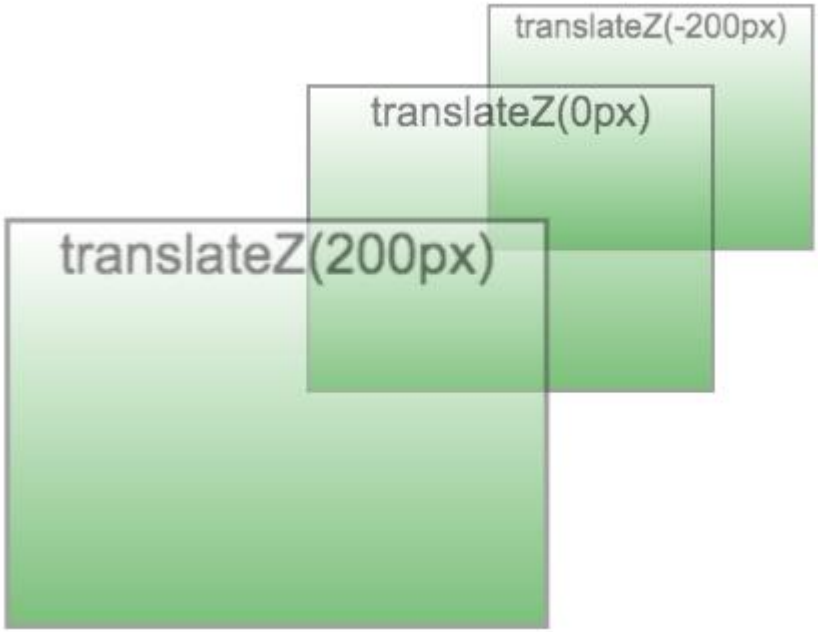
Un plan en 3D a 3 axes : X, Y et Z

La base des transformations 3D

- Rappel sur les fonctions 2D : translate(), rotate(), scale(), skew(), matrix()
- Leur version 3D :
  - **translate3d()** : déplacement de l'objet en 3D
  - **rotate3d()** : rotation de l'objet en 3D
  - **scale3d()** : mise à l'échelle de l'objet (agrandissement, rétrécissement) en 3D
  - **matrix3d()** : configuration personnalisée de la transformation en 3D.

Les translations 3D

- Il existe deux versions des translations 3D :
  - **translate3d(x, y, z)** : positionne l'objet dans l'espace en 3D aux coordonnées (X, Y, Z).
  - **translateZ(z)** : positionne l'objet dans l'espace en 3D aux coordonnées (0, 0, Z). En clair, vous ne jouerez ici que sur l'impression de profondeur, l'objet restant positionné au centre de la scène.

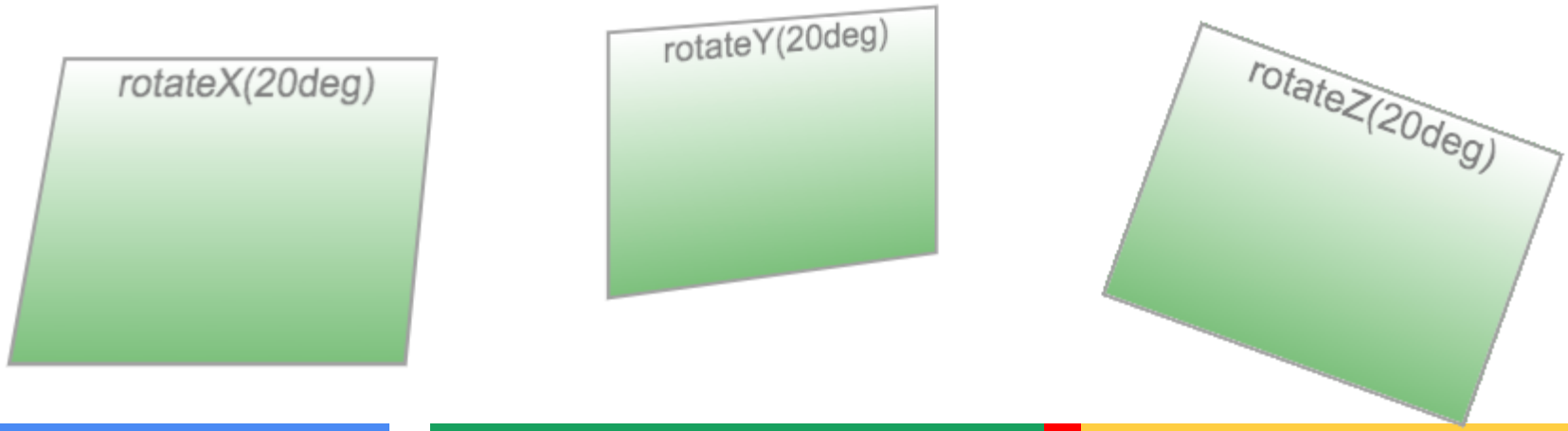


L'agrandissement 3D

- L'agrandissement 3D se gère avec l'une de ces fonctions :
  - **scale3d(sx, sy, sz)** : agrandit l'objet selon un facteur sx, sy, sz sur les différents axes. Avec un facteur de 2, la taille de l'objet est doublée. Avec un facteur de 0.5, la taille est divisée par deux.
  - **scaleZ(sz)** : agrandit l'objet selon un facteur de sz sur l'axe des Z (profondeur).

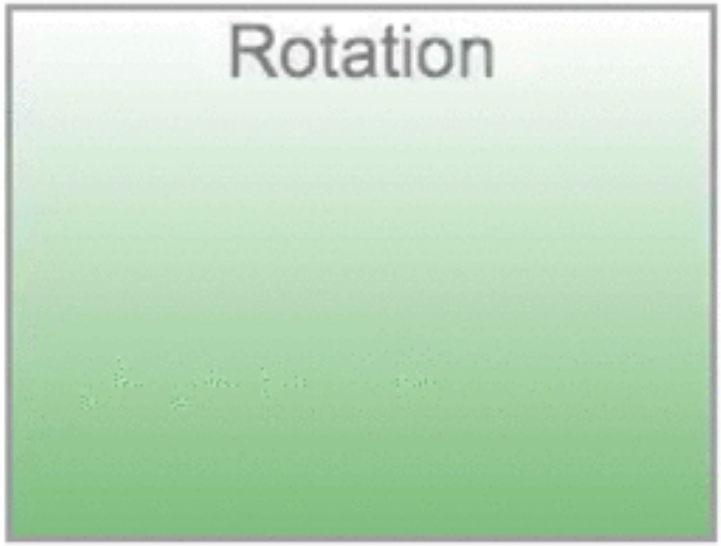
Les rotations 3D

- Les fonctions de rotations 3D :
  - **rotateX(x)** : une rotation sur l'axe X
  - **rotateY(y)** : une rotation sur l'axe Y
  - **rotateZ(z)** : une rotation sur l'axe Z. Cela correspond en fait à une rotation en 2D, donc c'est la même chose que la fonction rotate().
  - **rotate3d(x, y, z)** : une rotation sur plusieurs axes en même temps



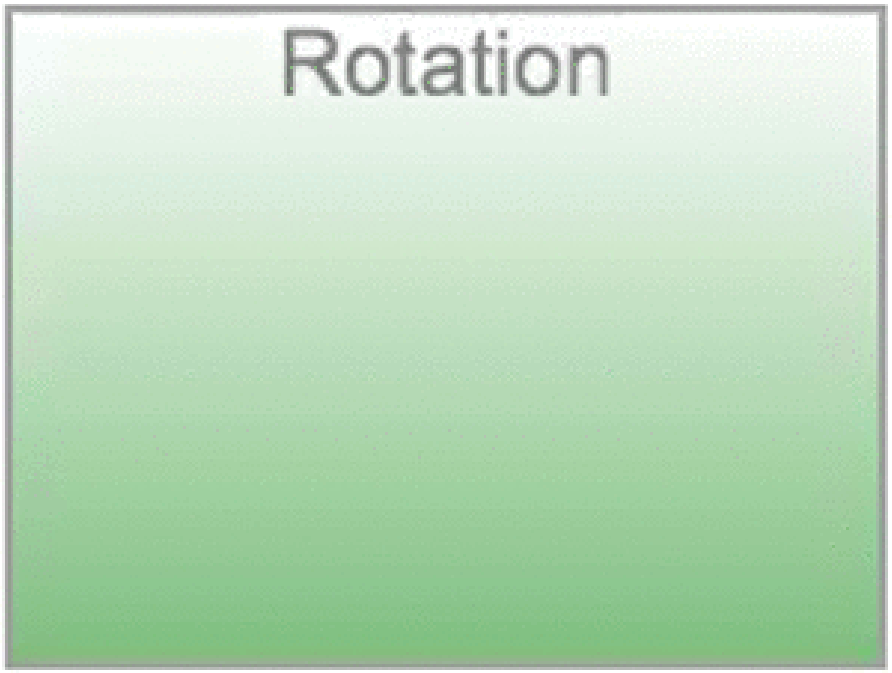
La face cachée des blocs

- Quand vous faites tourner suffisamment un bloc ?
- On voit sa face arrière.
- Par défaut, la face arrière est visible, ce qui nous donne :



La face cachée des blocs

- Si vous ne voulez pas que la face arrière soit visible, vous pouvez utiliser **backface-visibility**
- Passer de la valeur "visible" par défaut à la valeur "hidden" :



Jeux en CSS

https://keithclark.co.uk/labs/css-fps/desktop/
https://keithclark.co.uk/articles/creating-3d-worlds-with-html-and-css/