HTML5

HTML Tutorial

CSS AVANCE

HTML Tutorial

NOTATIONS LONG HAND VS SHORT HAND

Short hand et long hand

- Il s'avère qu'il arrive qu'une même propriété CSS puisse être écrite de différentes façons.
- En effet, en CSS, certaines propriétés peuvent être « regroupées » en une seule qui constitue leur notation **short hand**.
- Par exemple, pour définir une bordure, il y a deux façons de faire :
  - soit on utilise les trois propriétés CSS **border-width, border-style** et **border-color** (notation long hand),
  - soit on utilise la propriété **border** (qui correspond donc à la notation short hand) avec les valeurs des trois propriétés précédentes.
- Les notations short hand ont l'avantage d'être plus rapides à écrire, de consommer moins de ressources et d'être également finalement plus lisibles.
- Dans la suite de cette partie, nous allons voir quelques unes des notations short hand les plus utilisées.

La propriété CSS font

- La propriété **font** est la notation short hand des propriétés CSS **font-style, font-weight, font-size, line-heightet font-family.**

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Short hand</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Long hand vs short hand</h1>

        <p>Un paragraphe</p>
    </body>
</html>
```

```css
/*Notre titre h1 et notre paragraphe recevrons exactement
 *la même mise en forme CSS*/

h1{
    font-style: italic;
    font-weight: normal;
    font-size: 16px;
    line-height: 24px;
    font-family: Verdana, sans-serif;
}

p{
    font: italic normal 16px/24px Verdana, sans-serif;
}
```

La propriété CSS font

- Attention cependant à l'utilisation des notations short hand : il faut toujours suivre un ordre précis d'écriture des différentes valeurs.
- Noter également que l'on doit ici séparer les deux valeurs en pixels par un slash.

- Pour tester le code : http://jsbin.com/gopega/110/edit?html,css,output

Les propriétés CSS relatives au modèle des boîtes

Parmi les propriétés étudiées dans la partie « modèle des boîtes », il y a trois notations short hand à connaître :

- **border** qui regroupe **border-width, border-style et border-color** ;
- **padding** qui est la notation short hand de **padding-top, padding-right, padding-bottom** et **padding-left** ;
- **margin** qui regroupe **margin-top, margin-right, margin-bottom** et **margin-left.**

Les propriétés CSS relatives au modèle des boîtes

- (Suite)

```css
/*Notre titre h1 et notre paragraphe recevrons exactement
 *la même mise en forme CSS*/

h1{
    padding-top: 20px;
    padding-right: 15px;
    padding-bottom: 0px;
    padding-left: 50px;

    border-width: 5px;
    border-style: solid;
    border-color: blue;

    margin-top: 20px;
    margin-right: 15px;
    margin-bottom: 10px;
    margin-left: 5px;
}

p{
    padding: 20px 15px 0px 50px;
    border: 5px solid blue;
    margin: 20px 15px 10px 5px;
}
```

Les propriétés CSS relatives au modèle des boîtes

- (Suite)

La propriété CSS background

- La dernière notation short hand qu'il faut connaître est la propriété **background**.
- Cette propriété regroupe les valeurs des propriétés **background-color, background-image, background-repeat, background-attachment** et **background-position.**

```
/*Notre titre h1 et notre paragraphe recevrons exactement
 *la même mise en forme CSS*/

h1{
    background-color: #FFF;
    background-image: url("coucher-soleil.png");
    background-repeat: no-repeat;
    background-attachment: scroll;
    background-position: top left;
}

p{
    background: #FFF url("coucher-soleil.png") no-repeat scroll top left;
}
```

La propriété CSS background

- (Suite)

HTML Tutorial

SELECTEURS CSS AVANCES

Les sélecteurs CSS « avancés »

- Ce qui fait la grande force du CSS, c'est avant tout sa capacité à cibler précisément du contenu HTML pour lui appliquer des styles.
- Le CSS dispose de très nombreux sélecteurs permettant de sélectionner très précisément le ou les éléments à traiter graphiquement (surtout depuis CSS3).
- Certains de ces sélecteurs proposent des critères de sélection très ajustés. On les désigne par « sélecteurs avancés ».
- Ces sélecteurs permettent par exemple de sélectionner tous les éléments d'une page HTML, de sélectionner un élément par rapport à son parent ou encore de sélectionner un élément selon la valeur de son attribut.
- On en présentera ici quelques-uns à titre d'exemple.

Le sélecteur CSS universel « * »

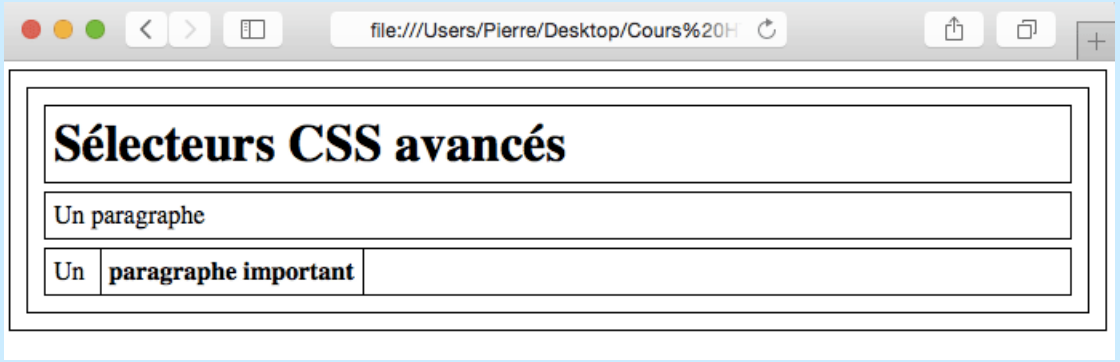
- Le sélecteur « * » (étoile) permet de sélectionner tous les éléments HTML d'une page d'un coup ; il est appelé **sélecteur CSS universel.**
- Grâce à ce sélecteur, on peut par exemple appliquer une bordure et des marges identiques à tous les éléments HTML de la page :

```
<!DOCTYPE html>
<html>
    <head>
        <title>Sélecteurs avancés</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Sélecteurs CSS avancés</h1>

        <p>Un paragraphe</p>
        <p>Un <strong>paragraphe important</strong></p>
    </body>
</html>
```

Le sélecteur CSS universel « * »

- (Suite)

```css
*{
    border: 1px solid black;
    padding: 5px;
    margin: 5px;
}
```



- Pour tester le code :
  http://jsbin.com/gopega/112/edit?html,css,output

Grouper les sélecteurs

- Grouper les sélecteurs, ce qui se fait à l'aide du symbole « , » (virgule) permet d'appliquer un même style aggrégé à différents éléments.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Sélecteurs avancés</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Sélecteurs CSS avancés</h1>

        <p>Un paragraphe</p>
        <p>Un <strong>paragraphe important</strong></p>
    </body>
</html>
```

```css
h1, strong{
    font-size: 20px;
    color: blue;
}
```

Grouper les sélecteurs

- (Suite)

- Pour tester le code : http://jsbin.com/gopega/113/edit?html,css,output

Sélectionner un élément par rapport à un autre

- Le CSS permet également de sélectionner un élément relativement à un autre.
- Ainsi, le sélecteur « **A B** » (où « A » et « B » désignent n'importe quel élément) permet de sélectionner un élément B contenu dans un élément A.

Sélectionner un élément par rapport à un autre

- Soit par exemple à sélectionner tous les paragraphes contenus dans un élément div.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Sélecteurs avancés</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Sélecteurs CSS avancés</h1>

        <div class="test">
            <p>Un premier paragraphe dans un div</p>
            <p>Un <strong>paragraphe important</strong></p>
        </div>

        <div>
            <p>Un paragraphe dans un deuxième div</p>
        </div>
        <p>Un paragraphe hors div</p>
    </body>
</html>
```

```css
.test p{
    font-size: 20px;
    color: blue;
}
```

Sélectionner un élément par rapport à un autre

- (Suite)

- http://jsbin.com/gopega/115/edit?html,css,output

Sélectionner un élément par rapport à un autre

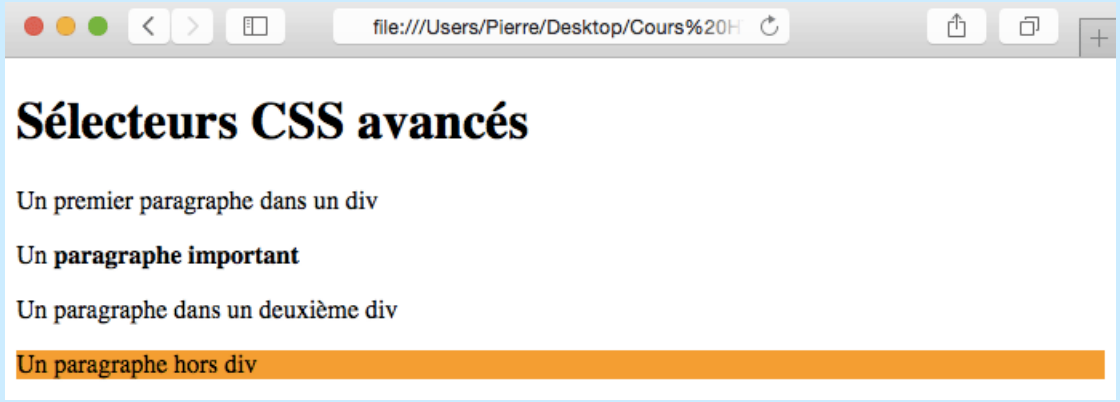
- Sous la forme « **A + B** » on peut sélectionner tous les éléments B de même niveau et suivant immédiatement un élément A.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Sélecteurs avancés</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Sélecteurs CSS avancés</h1>

        <div class="test">
            <p>Un premier paragraphe dans un div</p>
            <p>Un <strong>paragraphe important</strong></p>
        </div>

        <div>
            <p>Un paragraphe dans un deuxième div</p>
        </div>
        <p>Un paragraphe hors div</p>
    </body>
</html>
```

```css
div + p{
    background-color: orange;
}
```

Sélectionner un élément par rapport à un autre

- (Suite)



- Ici, seul le dernier paragraphe possède une couleur de fond car c'est bien le seul élément p de même niveau qu'un élément div et suivant directement un élément div
- Pour tester le code : http://jsbin.com/gopega/114/edit?html,css,output

Sélectionner un élément par rapport à un autre

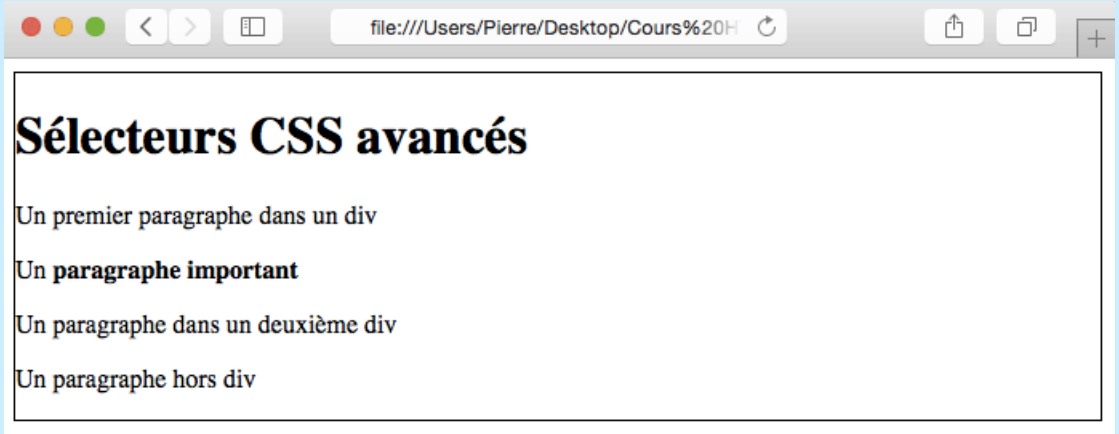
- Finalement, le sélecteur « **A > B** » sélectionne tous les éléments B qui sont des éléments enfants directs de A.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Sélecteurs avancés</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <div class="page-container">
            <h1>Sélecteurs CSS avancés</h1>

            <div class="test">
                <p>Un premier paragraphe dans un div</p>
                <p>Un <strong>paragraphe important</strong></p>
            </div>

            <div>
                <p>Un paragraphe dans un deuxième div</p>
            </div>
            <p>Un paragraphe hors div</p>
        </div>
    </body>
</html>
```

```css
body > div{
    border: 1px solid black;
}
```

Sélectionner un élément par rapport à un autre

- (Suite)



- Ici, seul l'élément div class="page-container" est un enfant direct de l'élément body, ce sera donc le seul à posséder une bordure.
- Noter bien qu'ici l'attribut class ne sert à rien, il n'est précisé ici que pour bien savoir de quel div on parle.
- Pour tester le code : http://jsbin.com/gopega/116/edit?html,css,output

Les sélecteurs CSS d'attributs

- Le CSS permet de sélectionner un élément HTML selon le fait qu'il possède ou non un attribut et selon la valeur de l'attribut.
- Ici encore, nous ne verrons que les sélecteurs assez courants.

Les sélecteurs CSS d'attributs

- Le sélecteur « **A[attribut]** » (où "A" désigne n'importe quel élément HTML et "attribut« le nom d'un attribut HTML), sélectionne tous les éléments A possédant cet attribut en particulier.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Sélecteurs avancés</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <div class="page-container">
            <h1>Sélecteurs CSS avancés</h1>

            <div class="test">
                <p>Un <a href="http://pierre-giraud.com" target="_blank">lien</a></p>
                <p>Un <strong>paragraphe important</strong></p>
            </div>

            <div>
                <p>Un autre <a href="http://pierre-giraud.com">lien</a></p>
            </div>
            <p>Un paragraphe hors div</p>
        </div>
    </body>
</html>
```

```css
a[target]{
    color: orange;
    font-weight: bold;
}
```

Les sélecteurs CSS d'attributs

- (Suite)

- Pour tester le code :
  http://jsbin.com/gopega/117/edit?html,css,output

Les sélecteurs CSS d'attributs

- Le sélecteur CSS « **A[attribut="valeur"]** » sélectionne tous les éléments A possédant un attribut particulier avec une valeur précise.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Sélecteurs avancés</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <div class="page-container">
            <h1>Sélecteurs CSS avancés</h1>

            <div class="test">
                <p>Un <a href="http://pierre-giraud.com" target="_blank">lien</a></p>
                <p>Un <strong>paragraphe important</strong></p>
            </div>

            <div>
                <p>Un autre <a href="http://pierre-giraud.com" target="_top">lien</a></p>
            </div>
            <p>Un paragraphe hors div</p>
        </div>
    </body>
</html>
```

```css
a[target]{
    color: orange;
    font-weight: bold;
}

a[target="_top"]{
    font-size: 30px;
}
```

Les sélecteurs CSS d'attributs

- (Suite)

- Pour tester le code :
  http://jsbin.com/gopega/118/edit?html,css,output

HTML Tutorial

LES PSEUDO CLASSES CSS

Définition des pseudo classes CSS

- Les **pseudo classes** permettent de changer le style d'un élément HTML selon son état, c'est-à-dire de façon dynamique.
- Par exemple, cela permettra d'afficher un paragraphe en gras lorsque l'utilisateur passe sa souris dessus ou changer la couleur d'un lien une fois celui-ci cliqué.
- Les pseudo classes sont reconnaissables au fait qu'elles commencent tous par le symbole « : » (deux-points).

Les pseudo classes CSS :link, :visited, :hover et :active

La pseudo classe

- **:link** permet de styliser un lien non visité;
- **:visited** permet de styliser un lien une fois celui-ci visité**.** Cependant, comme cet état a souvent été utilisé dans le passé par des utilisateurs malveillants pour exploiter des failles, les navigateurs modernes ont tendance à ignorer la plupart des règles associées en CSS ;
- **:hover** va permet de changer l'aspect d'un élément lorsque le curseur passe dessus ;
- **:active** va permet de modifier l'aspect d'un lien lors d'un clic**.**

Les pseudo classes CSS :link, :visited, :hover et :active

- Dans le cas où l'on fait appel à plusieurs des pseudo classes ci-dessus sur un même élément, il faut impérativement les utiliser dans l'ordre donné ici.
- Les pseudo classes **:link, :visited, :hover** et **:active** sont le plus souvent utilisées sur des liens HTML. Mais rien n'empêche d'utiliser **:hover** sur des paragraphes par exemple.

Les pseudo classes CSS :link, :visited, :hover et :active

- Exemple :
- HTML :

```
<!DOCTYPE html>
<html>
    <head>
        <title>Pseudo classes</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Pseudo classes CSS</h1>

        <div class="test">
            <p>Un <a href="http://www.pierre-giraud.com">lien</a> non visité</p>
            <p>Un paragraphe</p>
        </div>

        <div>
            <p>Un <a href="http://pierre-giraud.com">lien</a> déjà visité</p>
        </div>
        <p>Un paragraphe hors div</p>
    </body>
</html>
```

Les pseudo classes CSS :link, :visited, :hover et :active

- CSS :

```css
h1:hover{
    color: orange;
    font-family: Verdana, sans-serif;
}

a:link{
    color: blue;
    text-decoration: underline;
}

/*Font-weight et text-decoration ne fonctionneront pas car les
 *navigateurs modernes ont désactivé  la plupart des propriétés
 *CSS liées à cet état pour réduire le risque de faille de sécurité*/
a:visited{
    color: green;
    font-weight: bold;
    text-decoration: none;
}
```

Les pseudo classes CSS :link, :visited, :hover et :active

- Résultat :

- Dans l'exemple ci-dessus, le curseur de la souris se trouve sur le titre h1; celui-ci change donc de couleur et de police. De plus, on remarque que le deuxième lien a été visité, mais pas le premier.
- Pour tester le code : http://jsbin.com/gopega/120/edit?html,css,output

Les pseudo classes :first-child et :last-child

- Les pseudo classes CSS **:first-child** et **:last-child** permettent respectivement de sélectionner le premier élément et le dernier élément enfant HTML d'un type déterminé par rapport à un élément parent.
- Par exemple, dans le fichier HTML ci-après, le paragraphe contenant le texte « un premier paragraphe » est le premier élément enfant de l'élément div class="test". Et le troisième paragraphe est son dernier enfant.
- De même, le paragraphe « paragraphe hors div » est le dernier enfant de l'élément body.

Les pseudo classes :first-child et :last-child

- (Suite)

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Pseudo classes</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Pseudo classes CSS</h1>

        <div class="test">
            <p>Un premier paragraphe</p>
            <p>Un deuxième paragraphe</p>
            <p>Un troisième paragraphe</p>
        </div>
        <p>Un paragraphe hors div</p>
    </body>
</html>
```

```css
p:first-child{
    color: orange;
}

p:last-child{
    color: green;
}
```

Les pseudo classes :first-child et :last-child

- (Suite)

- Pour tester le code : http://jsbin.com/gopega/121/edit?html,css,output

La pseudo classe CSS :nth-child()

- La pseudo classe CSS **:nth-child()** permet de cibler certains enfants précis d'un élément HTML. On indique entre les parenthèses de cette pseudo classe soit un nombre, soit un mot clef.
- Quelques exemples pour mieux en appréhender le fonctionnement.

La pseudo classe CSS :nth-child()

- (Suite)

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Pseudo classes CSS</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Un titre de niveau 1</h1>
        <h2>Un titre de niveau 2</h2>
        <p>Un premier paragraphe</p>
        <p>Un deuxième paragraphe</p>
        <div>
            <h2>Un autre titre de niveau 2</h2>
            <p>Un troisième paragraphe</p>
            <p>Un quatrième paragraphe</p>
        </div>
    </body>
</html>
```

```css
/*Ce sélecteur va cibler tous les éléments p qui sont des enfants
 *pairs (2è enfant, 4è enfant, etc.) d'un élément HTML dans notre page*/
p:nth-child(even){
    color: purple;
}

/*Ce sélecteur va cibler tous les éléments h2 qui sont le deuxième
 *enfant d'un élément parent*/
h2:nth-child(2){
    color: green
}
```

La pseudo classe CSS :nth-child()

- (Suite)

- Pour tester le code : http://jsbin.com/sipujup/27/edit?html,css,output

La pseudo classe CSS :nth-child()

- Lorsque l'on utilise la pseudo classe :nth-child(), il ne faut pas faire de distinction entre les éléments HTML dans le comptage.
  - Par exemple, dans l'exemple précédent, l'élément body possède en tout cinq enfants : un titre h1, un titre h2, deux éléments p et un élément div.
  - L'élément div possède à son tour trois enfants : un titre h2 et deux éléments p.
- Dans un premier temps, on utilise la pseudo classe pour mettre en violet tous les paragraphes qui sont des enfants pairs d'un élément HTML donné, quel qu'il soit (grâce au mot clef "even" qui signifie "pair" en anglais).
- Le premier paragraphe est le troisième enfant de l'élément body, il ne sera donc pas ciblé. Le deuxième paragraphe, en revanche, est le quatrième enfant. Il sera donc ciblé.
- De même, le troisième paragraphe est le deuxième enfant de l'élément div, il s'affichera donc également en violet.
- Ensuite, le code indique que tous les éléments h2 qui sont le deuxième enfant d'un élément HTML doivent s'afficher en vert. Ce n'est le cas que de mon premier titre h2.

Autres pseudo classes CSS

- Il existe de nombreuses autres classes CSS, moins utilisées que celles présentées jusqu'ici. La majorité d'entre elles servent à cibler des types d'éléments présents dans les formulaires HTML.

HTML Tutorial

LES PSEUDO ELEMENTS CSS

Présentation des pseudo éléments CSS

- Les pseudo éléments CSS permettent de modifier l'apparence d'une partie seulement d'un ou de plusieurs éléments HTML, ou encore d'ajouter du contenu au début ou à la fin d'un certain élément HTML.
- Les pseudo éléments sont reconnaissables en CSS à leur écriture : ils commencent par « **::** » (un double deux-points).
- Note : Il est possible de rencontrer encore des pseudo éléments CSS écrits avec seulement « **:** » (un deux-points). Ceci est une ancienne syntaxe, vouée à disparaître désormais.

Présentation des pseudo éléments CSS

Il n'existe que cinq pseudo éléments en CSS, dans l'ordre :

- **::first-letter** permet de ne sélectionner que la première lettre d'un texte contenu dans un élément ;
- **::first-line** permet de ne sélectionner que la première ligne d'un texte contenu dans un élément ;
- **::selection** permet de sélectionner la partie d'un élément sélectionnée (par un double clic) par l'utilisateur ;
- **::before** permet d'insérer du contenu au début d'un élément HTML ;
- **::after** permet d'insérer du contenu à la fin d'un élément HTML.

Le pseudo élément CSS ::first-letter

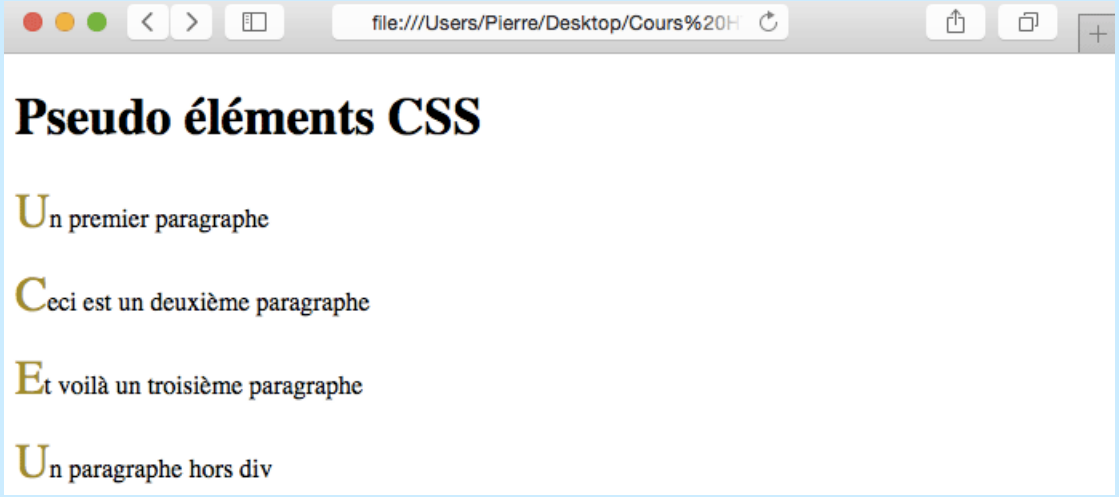
- Le pseudo élément CSS **::first-letter** permet de modifier l'apparence de la première lettre du texte d'un élément.
- On peut par exemple agrandir et changer la couleur de la première lettre de tous les paragraphes :

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Pseudo éléments</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Pseudo éléments CSS</h1>

        <div>
            <p>Un premier paragraphe</p>
            <p>Ceci est un deuxième paragraphe</p>
            <p>Et voilà un troisième paragraphe</p>
        </div>
        <p>Un paragraphe hors div</p>
    </body>
</html>
```

```css
p::first-letter{
    font-size: 30px;
    color: #A92;
}
```

Le pseudo élément CSS ::first-letter

- (Suite)



- Pour tester le code :
  http://jsbin.com/gopega/123/edit?html,css,output

Le pseudo élément CSS ::first-letter

- A noter que le pseudo élément **::first-letter** ne peut être appliqué qu'à des éléments HTML de type block.
- Ce pseudo élément ne supporte pas toutes les propriétés CSS ; voici une liste non exhaustive des propriétés compatibles :
  - Les propriétés liées à la police (font) ;
  - Les propriétés liées à la couleur ;
  - Les propriétés liées au fond (background) ;
  - La propriété border ;
  - La propriété margin ;
  - La propriété padding ;
  - Les propriétés text-decoration et text-transform ;
  - La propriété vertical-align ;
  - La propriété line-height ;
  - Les propriétés float et clear.

Le pseudo élément CSS ::first-line

- Le pseudo élément CSS **::first-line** permet de cibler et de modifier le style de la première ligne de texte d'un élément.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Pseudo éléments</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Pseudo éléments CSS</h1>

        <div>
            <p>Un premier paragraphe qui contient beaucoup de texte afin
            que celui-ci occupe plusieurs lignes au sein de notre page web</p>
            <p>On<br>peut<br>aussi<br>ajouter des break...</p>
        </div>
        <p>Un paragraphe hors div</p>
    </body>
</html>
```

```css
p::first-line{
    font-size: 30px;
    color: #A92;
}
```

Le pseudo élément CSS ::first-line

- (Suite)

- Pour tester le code : http://jsbin.com/gopega/124/edit?html,css,output

Le pseudo élément CSS ::first-line

- Ce pseudo élément ne s'applique également qu'aux éléments HTML de type block.
- De la même façon que pour :: first-letter, une liste non exhaustive des propriétés compatibles:
  - Les propriétés liées à la couleur ;
  - Les propriétés liées au fond (background) ;
  - La propriété border ;
  - Les propriétés letter-spacing et word-spacing ;
  - Les propriétés text-decoration et text-transform ;
  - La propriété vertical-align ;
  - La propriété line-height ;
  - La propriété clear.

Le pseudo élément CSS ::selection

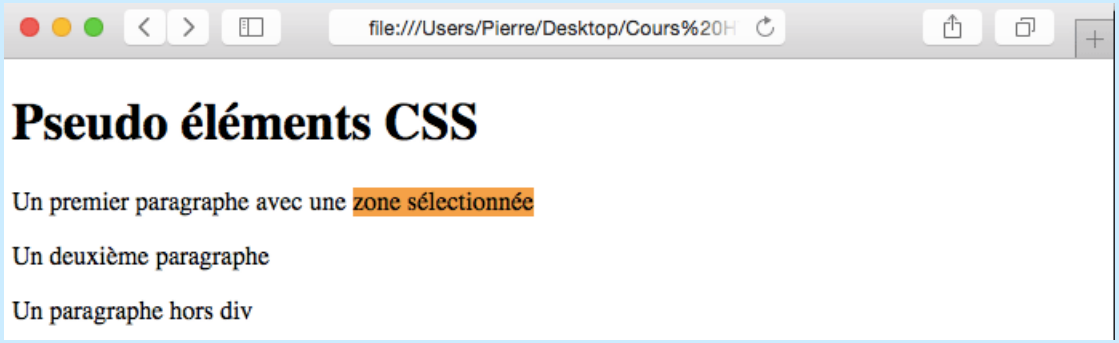
- Le pseudo élément CSS **::selection** permet de cibler et de mettre en forme la partie d'un élément sélectionné (double cliquée ou sélectionnée directement avec le pointeur de la souris) par un utilisateur.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Pseudo éléments</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Pseudo éléments CSS</h1>

        <div>
            <p>Un premier paragraphe avec une zone sélectionnée</p>
            <p>Un deuxième paragraphe</p>
        </div>
        <p>Un paragraphe hors div</p>
    </body>
</html>
```

```css
/*Alternative pour Mozilla (qui supporte mal ::selection)*/
p::-moz-selection{
    background-color: orange;
}

/*Pour les autres navigateurs*/
p::selection{
    background-color: orange;
}
```

Le pseudo élément CSS ::selection

- Ce pseudo élément supporte les propriétés CSS color et background ainsi que cursor et outline.
- Dans cet exemple, il est clair que l'on a sélectionné les mots "zone sélectionnée" avec la souris au moment de prendre la capture d'écran.



- Pour tester le code : http://jsbin.com/gopega/125/edit?html,css,output

Les pseudo éléments CSS ::before et ::after

- Les pseudo éléments CSS **::before** et **::after** permettent respectivement d'ajouter du contenu HTML au début et à la fin d'un certain élément HTML.
- Ces pseudo éléments devront être utilisés avec la propriété CSS **content** qui permet d'ajouter des contenus divers (du texte, des images, etc.).

Les pseudo éléments CSS ::before et ::after

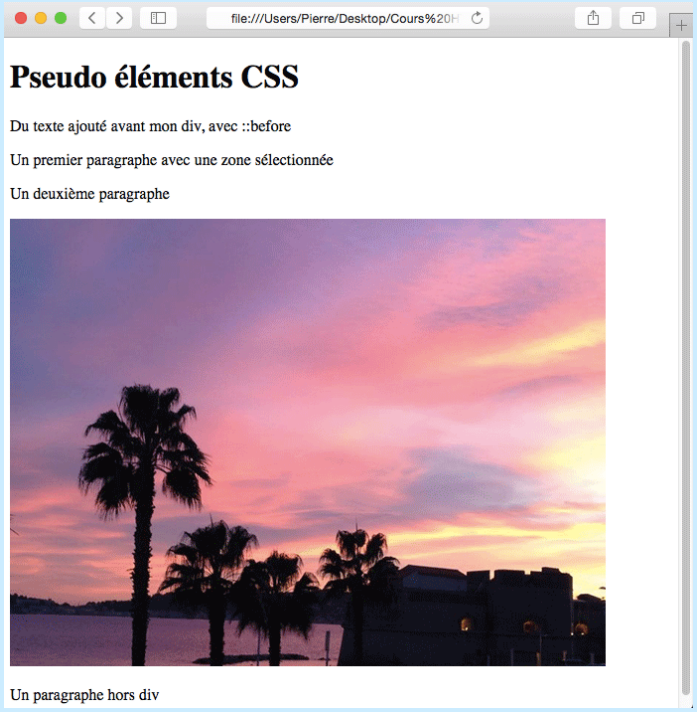
- Exemple :

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Pseudo éléments</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Pseudo éléments CSS</h1>

        <div class="div-un">
            <p>Un premier paragraphe avec une zone sélectionnée</p>
            <p>Un deuxième paragraphe</p>
        </div>
        <p>Un paragraphe hors div</p>
    </body>
</html>
```

```css
.div-un::before{
    content: "Du texte ajouté avant mon div, avec ::before";
}

.div-un::after{
    content: url("coucher-soleil.png");
}
```

Les pseudo éléments CSS ::before et ::after

- Dans cet exemple, on a ajouté du texte au début du div class="div-un" et une image à la fin de celui-ci grâce à ::before et ::after.



- Pour tester le code : http://jsbin.com/gopega/126/edit?html,css,output

HTML Tutorial

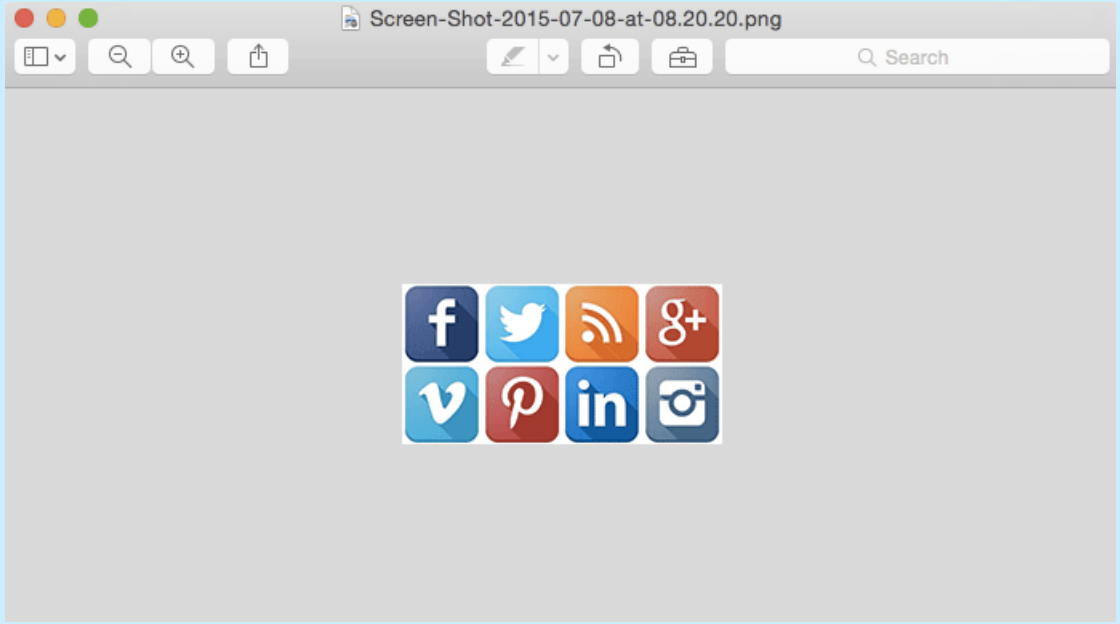
LES SPRITES D'IMAGES EN CSS

Définition des sprites et utilisation

- Un sprite d'images correspond à une collection d'images rassemblées en une seule grande image.
- L'idée sous-jacente à l'utilisation de sprites est de réduire le nombre de requêtes serveur.
- En effet, en cas d'images séparées, une requête différente va être envoyée au serveur pour récupérer chaque image présente dans une page web. Dans le cas d'une page contenant de nombreuses images, cela peut diminuer sensiblement la vitesse d'affichage de la page.
- En utilisant des sprites, une seule requête sera envoyée au serveur et cela optimisera les performances du site.

Exemple d'utilisation de sprites CSS

- Supposons par exemple que la page comprenne des liens vers des réseaux sociaux (situation fréquente …) . Ces liens sont représentés par les logos de ces réseaux.
- Plutôt que d'enregistrer séparément les logos des réseaux sociaux qui nous intéressent, on les enregistre côte-à-côte, sur la même image, comme ceci :



Exemple d'utilisation de sprites CSS

- Puis on place l'image correspondante dans le même dossier que les fichiers HTML et CSS.
- Ensuite, on crée simplement les liens en HTML et leur appliquer la propriété background afin d'y ajouter l'image cliquable.
- L'idée sera alors de ne récupérer que le logo qui nous intéresse dans notre sprite d'image pour chaque lien, ce qui se fait en CSS.
- Soit donc une image contenant différents logos de réseaux sociaux, dénommée par exemple « logos-sociaux.png » .
- Ensuite, on crée les liens en HTML en donnant un id à chaque lien afin de pouvoir ensuite le cibler précisément en CSS.
- Dans l'exemple qui suit, on se limite aux réseaux sociaux Facebook, Twitter et LinkedIn.

Exemple d'utilisation de sprites CSS

- HTML :

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Sprites</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Sprites CSS</h1>

        <div class="social">
            <a href="http://facebook.com" id="fb"><a>
            <a href="http://twitter.com" id="tw"></a>
            <a href="http://linkedin.com" id="lk"></a>
        </div>
    </body>
</html>
```

Exemple d'utilisation de sprites CSS

- On utilise le CSS pour mettre seulement le bon « bout » d'image en fond de chaque lien. Pour cela, la solution la plus simple est déclarer les liens comme des éléments block afin de pouvoir leur donner une taille en pixels puis de positionner l'image de fond.

```css
#fb, #tw, #lk{
    position: absolute;
    display: block;
    width: 50px;
    height: 50px;
}

#fb{
    left: 200px;
    top: 100px;
    background: url("logos-sociaux.png") 0px 0px;
}

#tw{
    left: 300px;
    top: 100px;
    background: url("logos-sociaux.png") -50px 0px;
}

#lk{
    left: 400px;
    top: 100px;
    background: url("logos-sociaux.png") -100px -50px;
}
```

Exemple d'utilisation de sprites CSS

- Sachant que les petits logos ont une taille d'environ 50*50 pixels, on donnera la même taille à tous les liens afin de ne récupérer qu'un bout de 50*50 pixels du sprite CSS à chaque fois.
- On précise en outre un positionnement absolu pour pouvoir par la suite séparer les images de fond les unes des autres.
- La propriété background permet d'insérer notre image de fond et on y précise deux coordonnées qui vont correspondre au déplacement de l'image, c'est-à-dire à partir d'où elle s'affiche.

Exemple d'utilisation de sprites CSS

- Par exemple, pour afficher le logo de Twitter, on sait qu'il faut se déplacer de 50px vers la droite sur l'image, ou encore faire bouger l'image de 50px vers la gauche pour la faire « démarrer » à 50px de son bord gauche. On utilise donc la valeur -50px qui fera commencer à 50px du début « théorique » de mon image.
- La deuxième valeur correspond au déplacement vertical. Le logo Twitter est sur la première ligne, donc à 0px du haut de notre image. On précise donc 0px.
- Pour le logo LinkedIn, en revanche, on précise -50px pour décaler l'image de 50px vers le haut et donc commencer 50px plus bas.
- Au final, voici le résultat :

HTML Tutorial

LES TRANSITIONS EN CSS

Présentation des transitions

- Les transitions permettent de modifier la valeur d'une propriété CSS de façon fluide, dans le temps, créant ainsi une transition entre les différentes valeurs de cette propriété. On pourra par exemple changer la couleur de textes de façon progressive.
- Sachant que le CSS ne gère les transitions que depuis relativement peu de temps, il faut donc utiliser les préfixes vendeurs afin de « forcer » la compatibilité des propriétés pour les anciennes versions des navigateurs.
- Ajoutons qu'Internet Explorer dans ses versions inférieures à 10 ne gère tout simplement pas du tout les transitions, que ce soit sans ou avec son préfixe vendeur.

La propriété CSS transition

La propriété CSS transition est une écriture short hand regroupant en fait quatre propriétés :

- **Transition-delay** : quand doit commencer la transition ;
- **Transition-duration :** durée de la transition ;
- **Transition-property :** propriété à laquelle la transition doit s'appliquer;
- **Transition-timing-function :** la courbe de vitesse de la transition.

La propriété CSS transition

- Parmi ces quatre propriétés, seules les valeurs de **transition-duration** et **transition-property** doivent être obligatoirement précisées.préciser pour utiliser la propriété transition.
- Par ces deux valeurs, on désigne la durée de la transition en secondes (« s ») ainsi que le nom de la propriété CSS concernée.
- Pour qu'une transition démarre, la propriété ciblée doit connaître un changement de valeur. Ceci sera réalisé par un changement d'état de l'élément auquel elle est associée. Par exemple, en utilisant la pseudo-classe :hover.

Création d'une transition simple en CSS

- Dans cet exemple, on effectue un effet de transition sur la taille et la couleur de fond d'un div, lorsqu'un utilisateur passe la souris dessus.
- HTML :

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Transitions !</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Les transitions en CSS</h1>

        <div class="div-un">
            <p>Un premier paragraphe</p>
            <p>Un deuxième paragraphe</p>
            <p>Un troisième paragraphe</p>
        </div>
    </body>
</html>
```

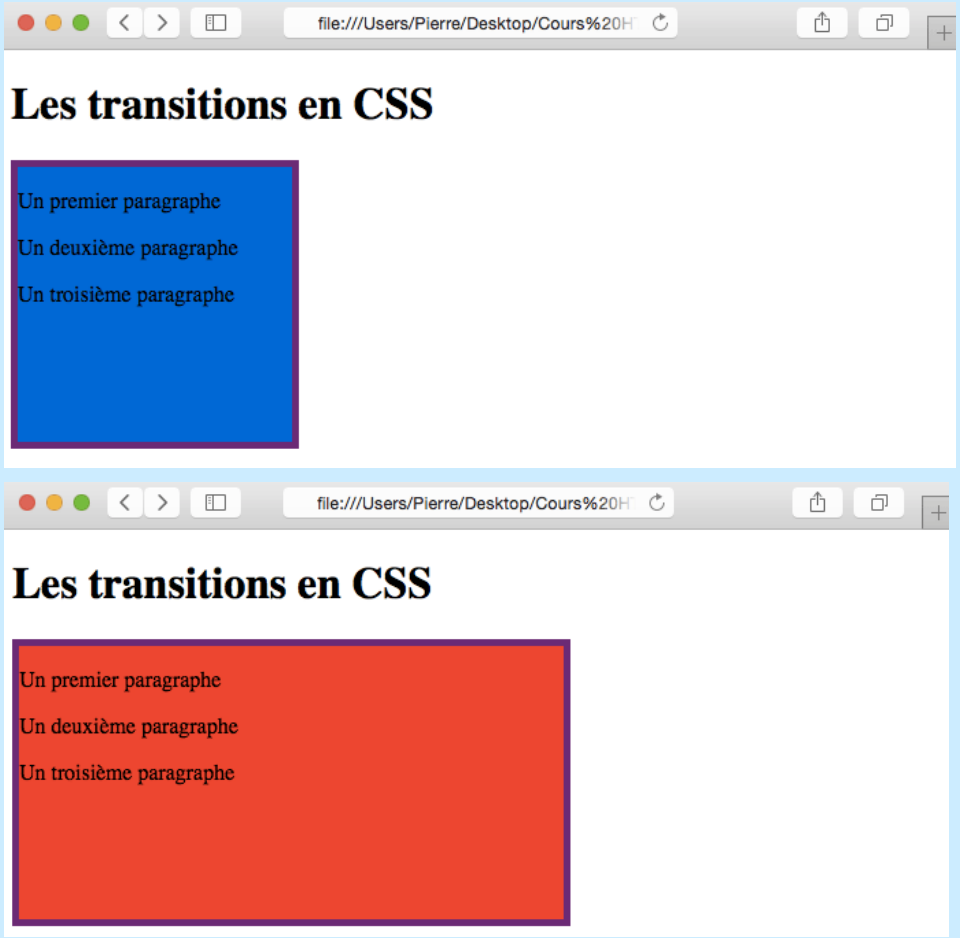
Création d'une transition simple en CSS

- CSS :

```css
.div-un{
    width: 200px;
    height: 200px;
    background-color: #06D;
    border: 5px solid purple;
    -webkit-transition: width 2s, background-color 5s;
    -moz-transition: width 2s, background-color 5s;
    -o-transition: width 2s, background-color 5s;
    transition: width 2s, background-color 5s;
}

.div-un:hover{
    width: 400px;
    background-color: red;
}
```

Création d'une transition simple en CSS

- Résultat :



- Pour tester le code : http://jsbin.com/gopega/127/edit?html,css,output

Création d'une transition CSS en utilisant toutes les valeurs

Indiquons ici comment on doit préciser les valeurs de **transition-delay** et de **transition-timing-function.**

- La propriété **transition-delay** spécifie le temps d'attente en secondes avant que la transition démarre.
- La propriété **transition-timing-function** définit la courbe de vitesse de notre transition, en choisissant parmi les valeurs suivantes :
  - **Ease** : La transition commence lentement puis accélère au milieu pour finir lentement
  - **Ease-in** : la transition commence lentement ;
  - **Ease-out** : la transition se termine lentement ;
  - **Ease-in-out** : la transition commence et se finit lentement ;
  - **Linear** : transition linéaire.

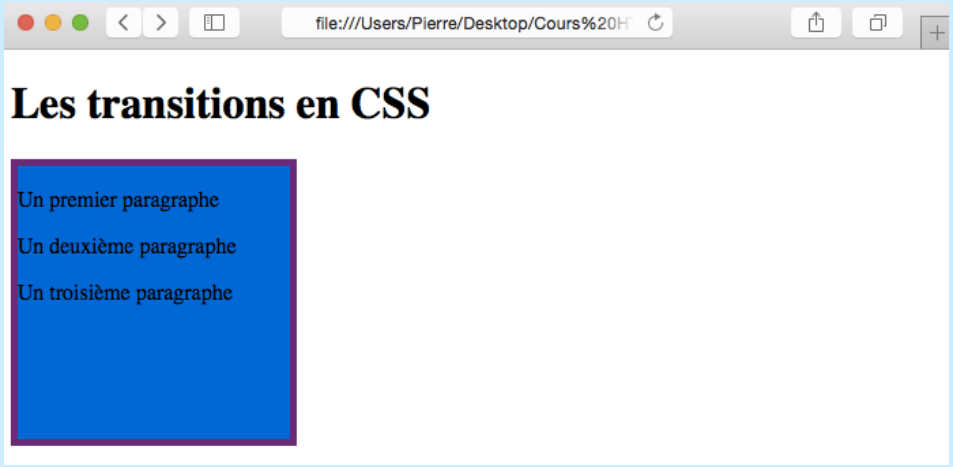
Création d'une transition CSS en utilisant toutes les valeurs

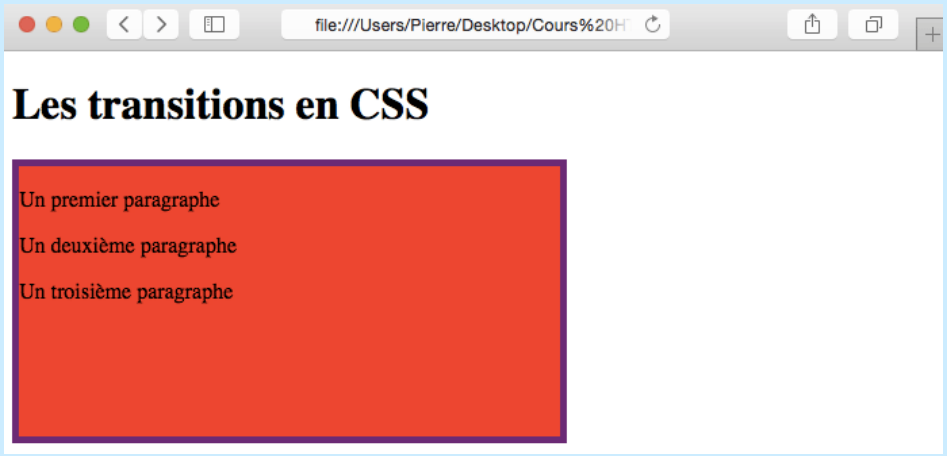
- (Suite)

```css
.div-un{
    width: 200px;
    height: 200px;
    background-color: #06D;
    border: 5px solid purple;
    -webkit-transition: 3s width 2s ease, background-color 5s ease;
    -moz-transition: 3s width 2s ease, background-color 5s ease;
    -o-transition: 3s width 2s ease, background-color 5s ease;
    transition: 3s width 2s ease, background-color 5s ease;
}

.div-un:hover{
    width: 400px;
    background-color: red;
}
```

Création d'une transition CSS en utilisant toutes les valeurs

- (Suite)





- Pour tester le code :
  http://jsbin.com/gopega/128/edit?html,css,output

HTML Tutorial

LES ANIMATIONS EN CSS

Présentation des animations en CSS

- Les animations en CSS permettent de changer le style d'un élément HTML.
- Contrairement aux transitions, on peut grâce aux animations modifier le style d'un élément HTML sans changement d'état de celui-ci et autant de fois que désiré.
- Sachant que le CSS ne gère les animations que depuis relativement peu de temps, il faut donc utiliser les préfixes vendeurs afin de « forcer » la compatibilité des propriétés pour les anciennes versions des navigateurs.
- A noter qu'Internet Explorer, dans ses versions antérieures à la version 10, ne supporte tout simplement pas les animations.
- Pour créer une animation en CSS, on utilise la propriété **animation** ainsi que la règle CSS **@keyframes.**

La règle CSS @keyframes et la propriété animation

- On appelle **@keyframes** une « règle » en CSS qui permet de modifier progressivement le style d'un élément.
- La propriété CSS animation est en fait la notation short-hand des propriétés suivantes :
  - **animation-name :** nom de l'animation, qu'on réutilisera dans la déclaration du @keyframes ;
  - **animation-duration :** durée de l'animation, en secondes ;
  - **animation-timing-function :** courbe de vitesse de l'animation ;
  - **animation-delay :** délai de l'animation ;
  - **animation-iteration-count :** nombre de fois que l'animation doit être jouée ;
  - **animation-direction :** spécifie si l'animation doit se jouer à l'envers ou selon des cycles alternés, c'est-à-dire en changeant de sens à chaque fois.

La règle CSS @keyframes et la propriété animation

- Parmi toutes ces propriétés, seules les valeurs relatives aux deux premières doivent obligatoirement être précisées
- Il faut noter cependant que la notation short-hand n'est pas toujours bien supportée par les navigateurs, et qu'il est donc prudent de programmer plutôt à l'aide des propriétés animation-xx.

Créer une animation simple en CSS

- Soit à créer une animation simple en CSS : changer la couleur d'un div, du bleu vers l'orange.
- HTML :

```
<!DOCTYPE html>
<html>
    <head>
        <title>Animations CSS !</title>
        <meta charset= "utf-8">
        <link rel="stylesheet" href="styles.css">
    </head>

    <body>
        <h1>Les animations en CSS</h1>

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

Créer une animation simple en CSS

- CSS :

```css
div {
    width: 100px;
    height: 100px;
    background-color: blue;
    -webkit-animation-name: chgt-couleur; /* Chrome, Safari, Opera */
    -webkit-animation-duration: 5s;
    -moz-animation-name: chgt-couleur; /* Mozilla */
    -moz-animation-duration: 5s;
    animation-name: chgt-couleur;
    animation-duration: 5s;
}

/*Pour Chrome, Safari et Opéra*/
@-webkit-keyframes chgt-couleur {
    from {background-color: blue;}
    to {background-color: orange;}
}

/*Pour Mozilla*/
@moz-keyframes chgt-couleur{
    from {background-color: blue;}
    to {background-color: orange;}
}

/*Syntaxe standard*/
@keyframes chgt-couleur {
    from {background-color: blue;}
    to {background-color: orange;}
}
```

- Pour tester le code : http://jsbin.com/gopega/129/edit?html,css,output

Créer une animation simple en CSS

- Dans cet exemple, on applique l'animation à l'élément div, en se limitant aux propriétés animation-name et animation-duration (version standard à la fin : rappel)
- Ensuite, le nom de notre animation est réutilisée dans le @keyframes qu'il faut également préfixer pour s'assurer un fonctionnement correct sur différents navigateurs.
- A l'intérieur du @keyframes, on indique par les mots clefs «from» et «to» la couleur de départ et de fin d'animation.
- Alternativement, on peut préciser des pourcentages plutôt que «from» et «to» afin de gérer plusieurs changements d'état et préciser quand ceux-ci doivent survenir durant notre animation.
- Notez qu'une fois l'animation terminée l'élément reprend son style d'origine.

Créer une animation complète en CSS

- Soit maintenant à créer une animation plus complète pour le div, exploitant les différentes propriétés animation-xx.
- Il s'agit de faire changer le div de couleur plusieurs fois dans notre @keyframe en précisant des valeurs en pourcentage.
- En outre, on fera bouger div dans la page en lui attribuant une position relative puis en modifiant celle-ci dans @keyframes.
- Pour tester le code de la page suivante : http://jsbin.com/gopega/130/edit?html,css,output

Créer une animation complète en CSS

- CSS :

```css
div {
    width: 200px;
    height: 200px;
    background-color: green;
    position: relative;
    -webkit-animation-name: chgt-couleur; /* Chrome, Safari, Opera */
    -webkit-animation-duration: 4s;
    -webkit-animation-iteration-count: infinite;
    -webkit-animation-direction: alternate;
    -moz-animation-name: chgt-couleur; /* Mozilla */
    -moz-animation-duration: 4s;
    -moz-animation-iteration-count: infinite;
    -moz-animation-direction: alternate;
    animation-name: chgt-couleur;
    animation-duration: 4s;
    animation-iteration-count: infinite;
    animation-direction: alternate;
}

/*Pour Chrome, Safari et Opéra*/
@-webkit-keyframes chgt-couleur {
    0% {background-color: green; top: 0px; left: 0px}
    25% {background-color: yellow; top: 0px; left: 400px}
    50%{background-color: red; top: 400px; left: 400px}
    75%{background-color: blue; top: 400px; left: 0px}
    100% {background-color: green; top: 0px; left: 0px}
}

/*Pour Mozilla*/
@moz-keyframes chgt-couleur{
    0% {background-color: green; top: 0px; left: 0px}
    25% {background-color: yellow; top: 0px; left: 400px}
    50%{background-color: red; top: 400px; left: 400px}
    75%{background-color: blue; top: 400px; left: 0px}
    100% {background-color: green; top: 0px; left: 0px}
}

/*Syntaxe standard*/
@keyframes chgt-couleur {
    0% {background-color: green; top: 0px; left: 0px}
    25% {background-color: yellow; top: 0px; left: 400px}
    50%{background-color: red; top: 400px; left: 400px}
    75%{background-color: blue; top: 400px; left: 0px}
    100% {background-color: green; top: 0px; left: 0px}
}
```

Créer une animation complète en CSS

- Pour commencer, on définit la taille, une couleur de fond par défaut et le positionnement du div.
- Puis on applique les propriétés animation-name, animation-duration, animation-iteration-count et animation-direction.
- La propriété animation-iteration-count définit combien de fois l'animation sera jouée. Elle prend en valeur soit un nombre, soit un mot clef comme « infinite » (l'animation sera répétée indéfiniment).
- La propriété animation-direction définit le sens de l'animation. Celle – ci peut aller en sens inverse (valeur reverse), en cycles alternatifs (une fois à l'endroit, une fois à l'envers avec la valeur alternate) ou encore en cycles alternatifs inversés.

Créer une animation complète en CSS

- Finalement, on précise dans @keyframes la nature de notre transition et comment lui donner un effet fluide. On utilise donc des valeurs en pourcentage pour définir à quel moment dans l'animation doit commencer chaque nouvel effet et on attribue les pourcentages souhaités à chaque effet d'animation.
- Dans le cas présent, on change la couleur de fond du div à chaque itération ainsi que sa place relativement à sa place de départ.

- (Suite)