



## HTML 5 - CSS3



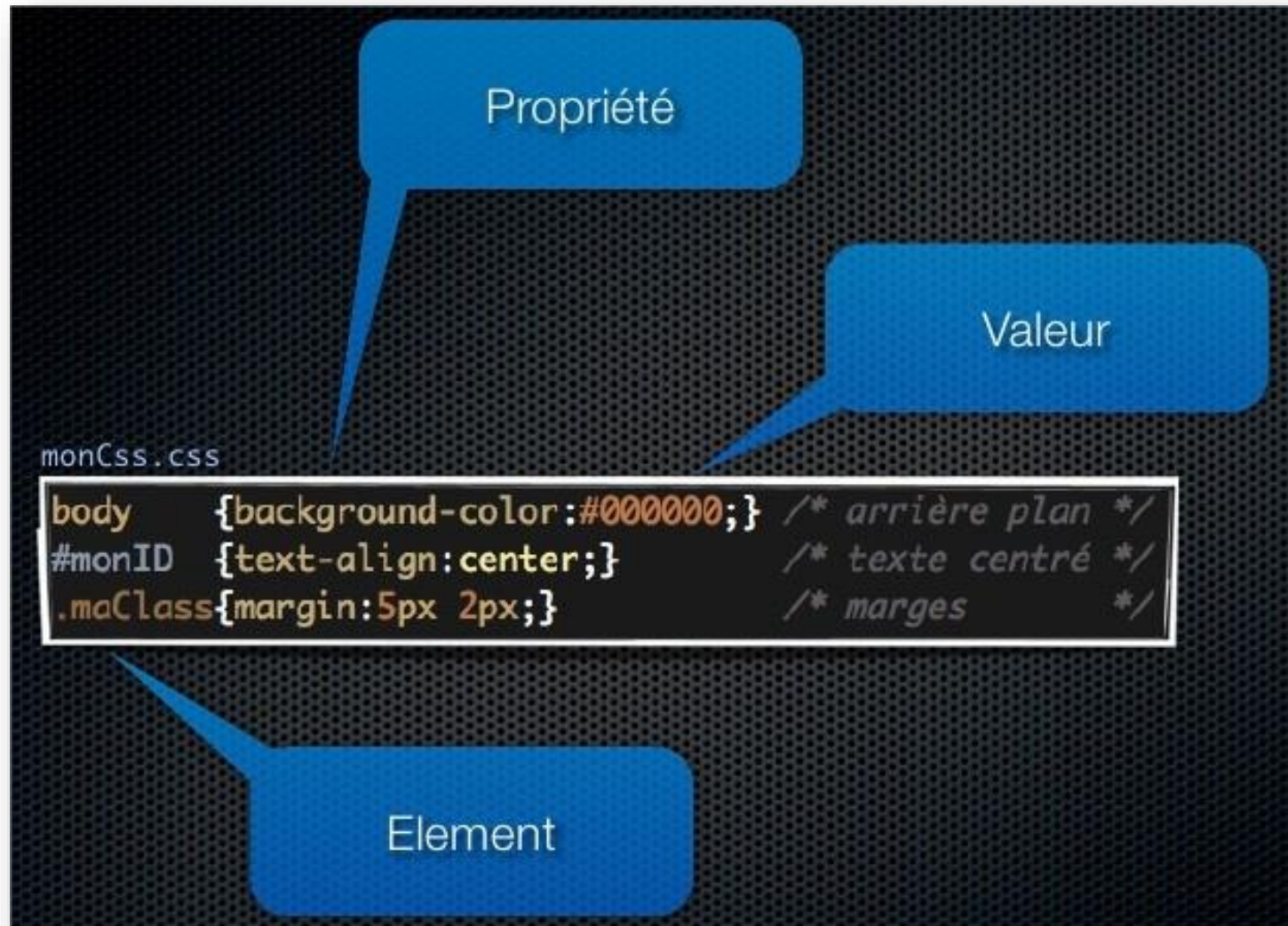
# HTML5 – CSS3

- Rappels
  - **CSS** = « *Cascading Style Sheets* »
  - Le CSS permet de décrire la **présentation** d'un **document**
  - On réalise donc via le Css une séparation entre le **fond** et la **forme**



# HTML5 – CSS3

- Rappels





# HTML5 – CSS3

- Rappels



# HTML5 – CSS3

- Rappels
  - Règle #1 : **Pas de CSS/JS inline**
  - **Règle #2** : Utilisez un **CSS Reset**

# HTML5 – CSS3

- Pourquoi CSS3 ?
  - HTML4 -> HTML5
  - CSS 2.1 -> CSS3
- Comment?
  - CSS3 bénéficie d'une rétrocompatibilité complète avec CSS 2.1 😊
  - Possibilité d'utiliser des sélecteurs et de nouvelles propriétés
  - CSS3 est divisé en 'modules'
    - Le modèle de boîtes ;
    - Le module de listes ;
    - La présentation des hyperliens ;
    - Le module vocal ;
    - Les arrières plan et les Bords ;
    - Les effets de texte ;
    - La mise en page multi-colonnes.

# HTML5 – CSS3

- Un peu d'histoire ...
  - CSS1 est né en 1996
  - En 2000, Internet Explorer 5 implémente le premier 99% de CSS1
  - CSS2 débute en 1998 avec beaucoup d'ambition (affichage en braille ou des rendus vocaux)
  - Le W3C revoit ses objectifs à la baisse avec CSS2.1
  - 2007 que CSS2.1 devient une recommandation candidate !  
Neuf ans après sa création, on n'avait toujours pas une version 100% stable de CSS2 !
  - Ce n'est que le 7 Juin 2011 que le W3C a officialisé ses recommandations : ce n'est que depuis cette date que nous sommes censés utiliser les propriétés CSS2.1 sans risque de bogues !
  - Le développement de CSS3 s'est fait en parallèle de CSS2.1, dès 1999 !
  - CSS3 sera le prochain objectif du W3C ... mais, ce n'est pas parce qu'ils sont désormais focalisés dessus qu'il sortira prochainement...  
Encore une fois, reprenons l'exemple de CSS2.1...

# HTML5 – CSS3

- Un peu d'histoire ...
  - Heureusement pour nous, les navigateurs Web modernes préfèrent prendre le risque d'interpréter les propriétés avant l'officialisation de leurs recommandations : imaginez un peu si nous avions dû attendre plus d'une dizaine d'années avant de pouvoir utiliser CSS2.1 !
  - Ainsi, même s'ils ne gèrent pas encore le CSS2.1 à 100%, les navigateurs modernes ont un point d'honneur à interpréter le plus de propriétés CSS3 possible : tout comme en 1996, ils ne souhaitent pas que leurs concurrents puissent se dire être les plus conformes.
  - Seulement, comme tout n'est pas normé, il va nous falloir gérer quelques soucis de compatibilité...



# HTML5 – CSS3

- Problèmes de compatibilité
  - Comme nous venons de le voir, toutes les propriétés CSS3 ne sont pas gérées par tous les navigateurs
  - à vouloir aller plus vite que ceux qui établissent les normes, on se casse parfois un peu les dents !
  - Le W3C a bien remarqué que les navigateurs veulent aller plus vite que les normes.
  - Puisqu'on ne peut pas empêcher les navigateurs d'implémenter des propriétés, le W3C leur recommande de préfixer celles qui ne sont pas 100% officialisées.
  - C'est une façon de rendre "propriétaire" une propriété. En préfixant le nom d'une propriété par un tiret et un petit code correspondant au navigateur, seul ce dernier sera capable de l'interpréter.
  - Cela permet aux navigateurs qui le souhaitent (c'est-à-dire la plupart ) de proposer aux développeurs Web un support des futurs modules CSS.

# HTML5 – CSS3

- Tableau de compatibilité

CSS3 Properties														
	MAC						WIN							
	CHROME	FIREFOX	OPERA	SAFARI	CHROME	FIREFOX	OPERA	IE	6	7	8	9	10	
RGBA	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	91%
HSLA	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	91%
Box Sizing	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓	35%
Background Size	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓	90%
Multiple Backgrounds	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓	90%
Border Image	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	85%
Border Radius	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓	91%
Box Shadow	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓	91%
Text Shadow	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	76%
Opacity	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓	91%
CSS Animations	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	66%
CSS Columns	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	85%
CSS Gradients	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	84%
CSS Reflections	✓	✗	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	46%
CSS Transforms	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓	90%
CSS Transforms 3D	✓	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗	✓	✓	41%
CSS Transitions	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	72%
CSS FontFace	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	98%
FlexBox	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	72%
Generated Content	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓	✓	✓	35%
DataURI	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓	✓	✓	35%
Pointer Events	✓	✓	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	29%
Display: table	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓	✓	✓	35%
Overflow Scrolling	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	3%
Media Queries	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓	✓	34%

- Ressources

- <http://caniuse.com/#cats=CSS>
- [http://www.w3schools.com/cssref/css3\\_browsersupport.asp](http://www.w3schools.com/cssref/css3_browsersupport.asp)
- <http://css3test.com/>

# HTML5 – CSS3

- Préfixes
  - Ces préfixes sont à placer devant chaque propriété non finalisée.
  - Lorsqu'une propriété est finalisée, le préfixe peut être supprimé, car elle est considérée comme officialisée et donc prête à l'emploi !
  - Préfixes
    - **-moz-** pour le moteur Gecko (Mozilla Firefox) ;
    - **-webkit-** pour le moteur Webkit (Google Chrome, Safari...) ;
    - **-o-** pour Opera ;
    - **-ms-** pour Internet Explorer 8+ ;

# HTML5 – CSS3

- Aide-mémoire
  - Propriétés CSS3 qui ne nécessitent plus de préfixes
    - [border-radius](#)
    - [box-shadow](#)
    - [text-shadow](#)
    - [opacity](#)
    - [background-size](#)
  - Linear-gradient
    - La syntaxe la plus "compatible" possible

```
background: #59b8b5;
```

```
background: -webkit-linear-gradient(top, #63c1be 0%, #59b8b5 100%);
```

```
background: linear-gradient(to bottom, #63c1be 0%, #59b8b5 100%);
```

# HTML5 – CSS3

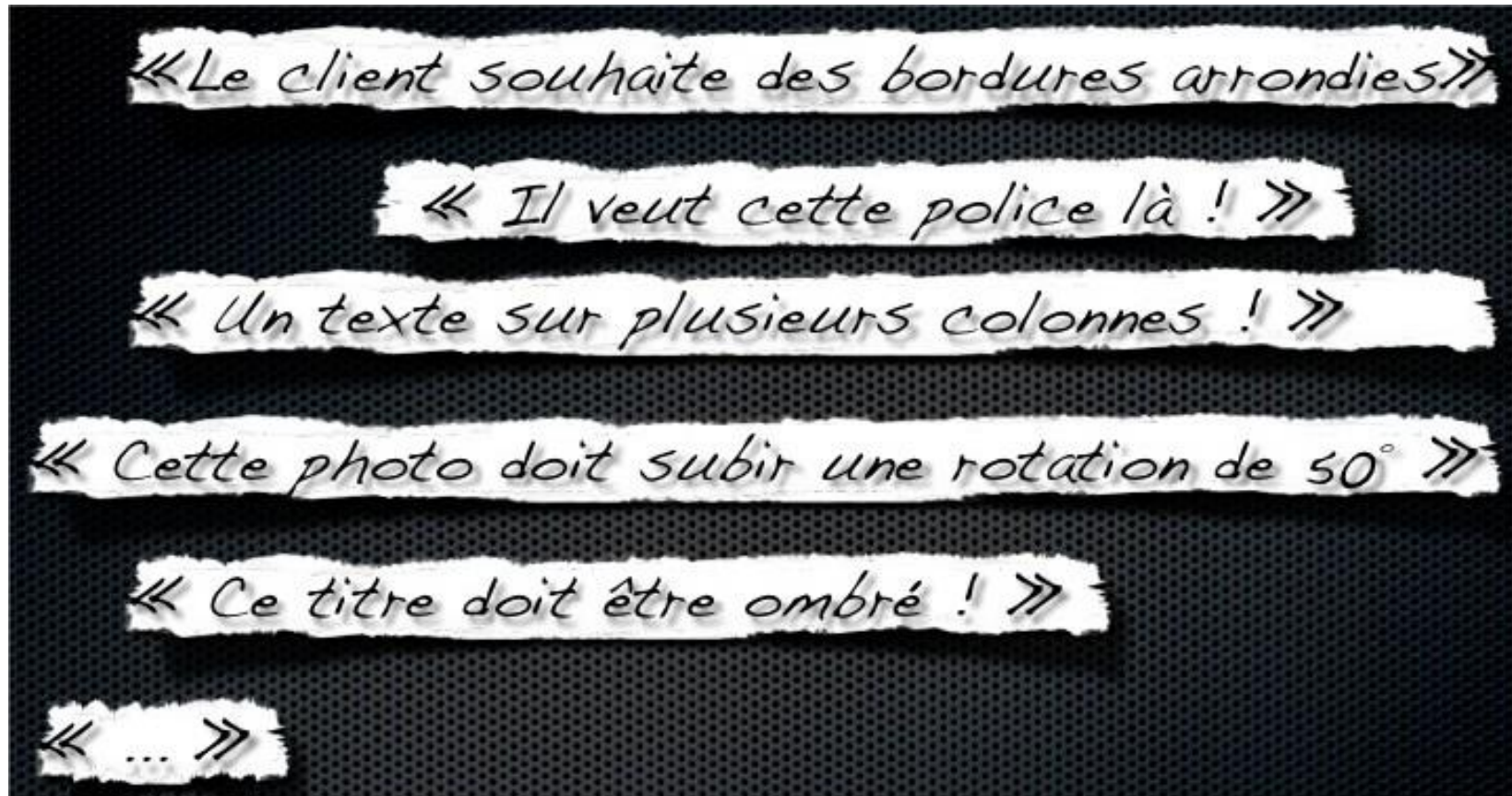
- Le Webdesigner conçoit le site sous Photoshop !





# HTML5 – CSS3

- Puis intégration du design ...





# HTML5 – CSS3



L'intégrateur  
CSS/HTML...

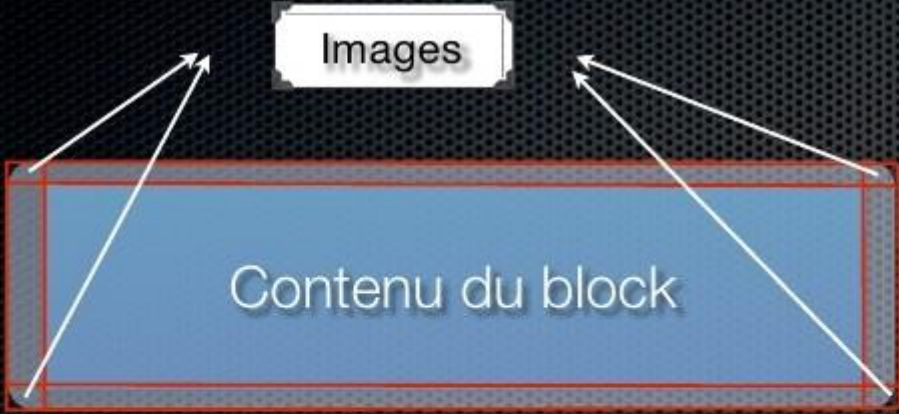
# HTML5 – CSS3

☹️ (quasi) impossible en pur CSS 2.1

😊 CSS3 le permet

# HTML5 – CSS3

- Bordures arrondies **avant CSS3**



The diagram shows a blue rectangular block with rounded corners. A red border is drawn around the block, with arrows pointing to the corners from a label 'Images' above. The text 'Contenu du block' is centered inside the blue area. Below the block, the text 'De 4 à 8 éléments pour réaliser la bordure' is displayed.

**Images**

Contenu du block

De 4 à 8 éléments  
pour réaliser la bordure

*Equivalent HTML\*:*

```
<div class="block">  
  <div>  
    <div></div>  
    <div></div>  
  </div>  
<div></div>  
<p>Contenu du block</p>  
<div></div>  
<div>  
  <div></div>  
  <div></div>  
</div>  
</div>
```

Trop de CSS [...]

# HTML5 – CSS3

- Bordures arrondies avec CSS3



# HTML5 – CSS3

- Les ombres avant CSS3



The diagram shows a visual representation of a block with a shadow and a flash effect. On the left, a blue rectangle labeled "Contenu du block" is shown with a red border and a drop shadow. To its right, the text "Texte avec Ombre" is displayed with a drop shadow. A red starburst labeled "Flash" is positioned to the right of the text. Below this visual, the text "Equivalent HTML\*:" is followed by the corresponding HTML code.

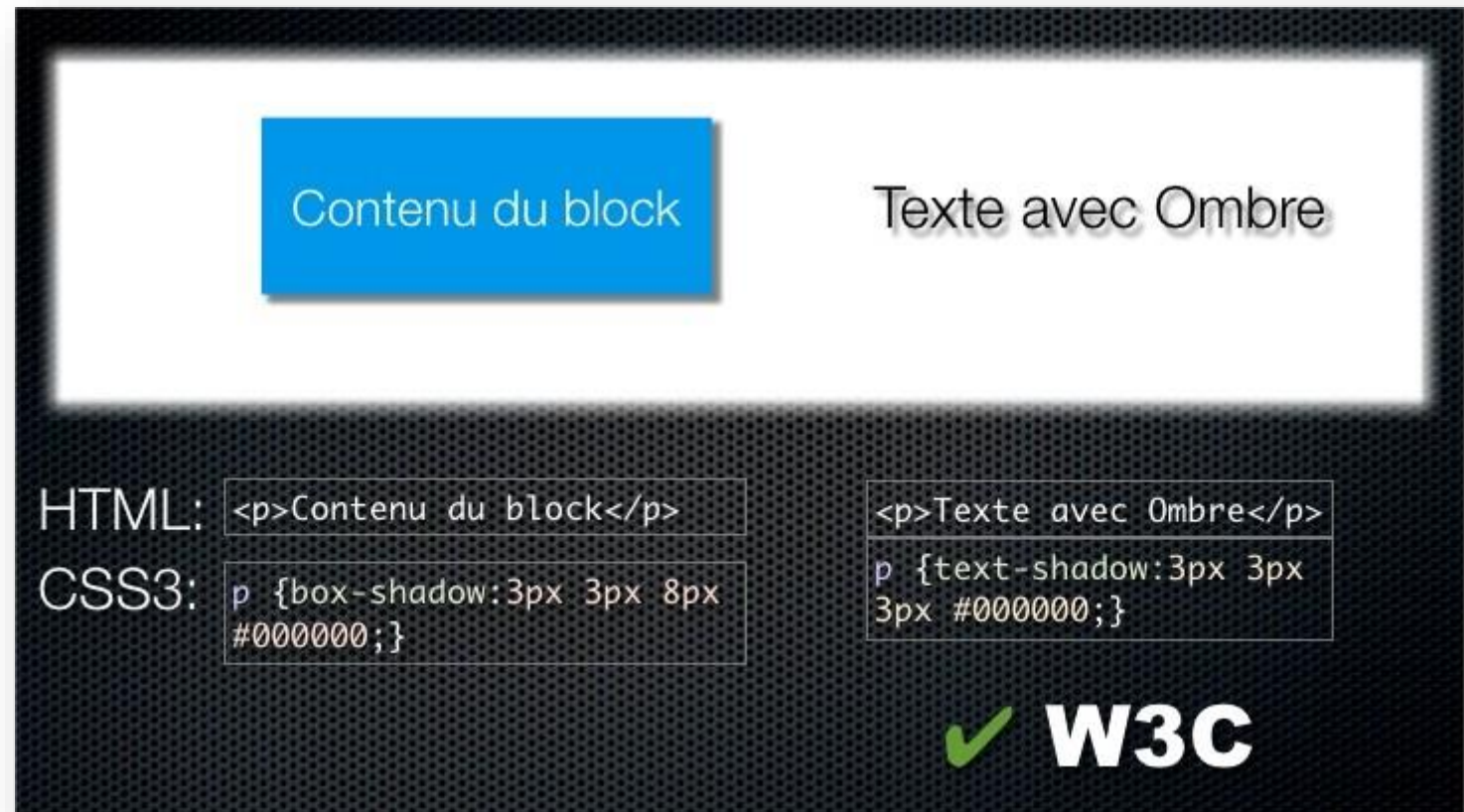
*Equivalent HTML\*:*

```
<div class="block">
  <div>
    <p>Contenu du block</p>
  </div>
  <div>
    <div></div>
    <div></div>
  </div>
</div>
```



# HTML5 – CSS3

- Les ombres avec **CSS3**



Contenu du block

Texte avec Ombre

HTML: `<p>Contenu du block</p>`

CSS3: `p {box-shadow:3px 3px 8px #000000;}`

`<p>Texte avec Ombre</p>`

`p {text-shadow:3px 3px 3px #000000;}`

✓ W3C

- Accessibles sur les textes et boîtes

`text-shadow: decalX, decalY, force, couleur;`

`box-shadow: decalX, decalY, force, couleur;`



# HTML5 – CSS3



Un intégrateur  
utilisant CSS3

# HTML5 – CSS3

- Composition des couleurs

- Couleurs nommées

- ```
color: purple;
```

- Couleur HTML

- ```
color: #ffffff;
```

- Composition **RGBA**

- ```
color: rgba(127,127,127,0.5);
```

- Composition **hsla : Teinte Saturation Valeur**

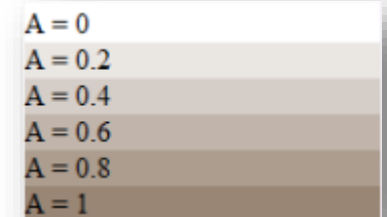
- ```
color: hsla(128,75%,33%,1);
```

- [http://fr.wikipedia.org/wiki/Teinte Saturation Valeur](http://fr.wikipedia.org/wiki/Teinte_Saturation_Valeur)

# HTML5 – CSS3

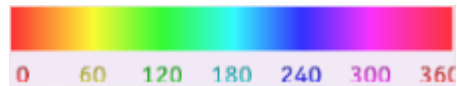
- Composition des couleurs en **RGBA**
  - RGB est le format de codage des couleurs utilisé dans les versions précédentes du CSS.
  - CSS3 ajoute l'opacité (ou transparence) au RGB et cela donne le RGBA :
    - **Red** (Rouge) : valeur entre 0 et 255 ou en %
    - **Green** (Vert) : valeur entre 0 et 255 ou en %
    - **Bleu** (Bleu) : valeur entre 0 et 255 ou en %
    - **Alpha** (Opacité) : valeur entre 0.0 et 1.0
  - Exemple :

```
div.rgba1 { background-color:rgba(153, 134, 117, 0); }  
div.rgba2 { background-color:rgba(153, 134, 117, 0.2); }  
div.rgba3 { background-color:rgba(153, 134, 117, 0.4); }  
div.rgba4 { background-color:rgba(153, 134, 117, 0.6); }  
div.rgba5 { background-color:rgba(153, 134, 117, 0.8); }  
div.rgba6 { background-color:rgba(153, 134, 117, 1); }
```



# HTML5 – CSS3

- HSL : CSS3 intègre une autre manière que RGB de définir des couleurs.
  - Le système HSL (hue, saturation, lightness)
  - Décrit les couleurs en fonction de la teinte, de la saturation, et de la luminosité :
    - **H** : teinte, valeur entre 0 et 360, définit la couleur (pas de blanc, ni de noir, ni de gris)



- **S** : saturation (ou pureté), valeur en pourcentage, définit l'intensité de la teinte
  - une teinte hautement saturée a une couleur vive et une teinte moins saturée paraît plus fade et g
- **L** : luminosité, valeur en pourcentage, ajout d'une intensité de blanc ou de noir
  - 0% est noir (sombre), 100% est le blanc (lumière)



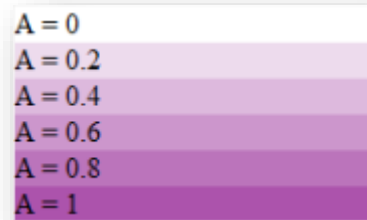
```
div.hs11 { background-color:hsl(300, 100%, 0%); color:white; }
div.hs12 { background-color:hsl(300, 100%, 25%); }
div.hs13 { background-color:hsl(300, 100%, 50%); }
div.hs14 { background-color:hsl(300, 100%, 75%); }
div.hs15 { background-color:hsl(300, 100%, 100%); }
div.hs16 { background-color:hsl(300, 0%, 50%); }
div.hs17 { background-color:hsl(300, 25%, 50%); }
div.hs18 { background-color:hsl(300, 50%, 50%); }
div.hs19 { background-color:hsl(300, 75%, 50%); }
div.hs110 { background-color:hsl(300, 100%, 50%); }
```

S = 100 % - L = 0 %
S = 100 % - L = 25 %
S = 100 % - L = 50 %
S = 100 % - L = 75 %
S = 100 % - L = 100 %
S = 0 % - L = 50 %
S = 25 % - L = 50 %
S = 50 % - L = 50 %
S = 75 % - L = 50 %
S = 100 % - L = 50 %

# HTML5 – CSS3

- HSLA

- Comme HSL avec la gestion de l'opacité (ou transparence).
- Comme pour RGBA, l'opacité est définie avec une flottante entre 0.0 et 1.0
- Exemple :



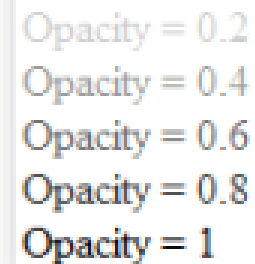
```
div.hs1a1 { background-color:hsla(300, 35%, 50%, 0); }  
div.hs1a2 { background-color:hsla(300, 35%, 50%, 0.2); }  
div.hs1a3 { background-color:hsla(300, 35%, 50%, 0.4); }  
div.hs1a4 { background-color:hsla(300, 35%, 50%, 0.6); }  
div.hs1a5 { background-color:hsla(300, 35%, 50%, 0.8); }  
div.hs1a6 { background-color:hsla(300, 35%, 50%, 1); }
```

# HTML5 – CSS3

- Opacity

- Permet d'appliquer une transparence à un élément et à tous ses enfants.
- Prend pour valeur un flottant (entre 0.0 et 1.0)
- Exemple :

```
div.opacity1 { opacity:0; }  
div.opacity2 { opacity:0.2; }  
div.opacity3 { opacity:0.4; }  
div.opacity4 { opacity:0.6; }  
div.opacity5 { opacity:0.8; }  
div.opacity6 { opacity:1; }
```



Opacity = 0.2  
Opacity = 0.4  
Opacity = 0.6  
Opacity = 0.8  
Opacity = 1



# HTML5 – CSS3

- Différence entre la transparence RGBA ou HSL et la propriété Opacity
  - RGBA : Une valeur qui s'applique à une propriété de l'élément sélectionné uniquement.
    - RGBa est susceptible de s'appliquer à toutes les propriétés dont la valeur peut être une couleur : background-color, color, border-color, box-shadow, text-shadow, etc.

**Ici un texte**

(parent alpha = 1)

**Ici un texte**

(parent alpha = 0.5)

- Opacity : Une propriété qui s'applique à l'élément dans son intégralité (ainsi qu'à tous ses descendants)

**Ici un texte**

(parent opacity: 1)

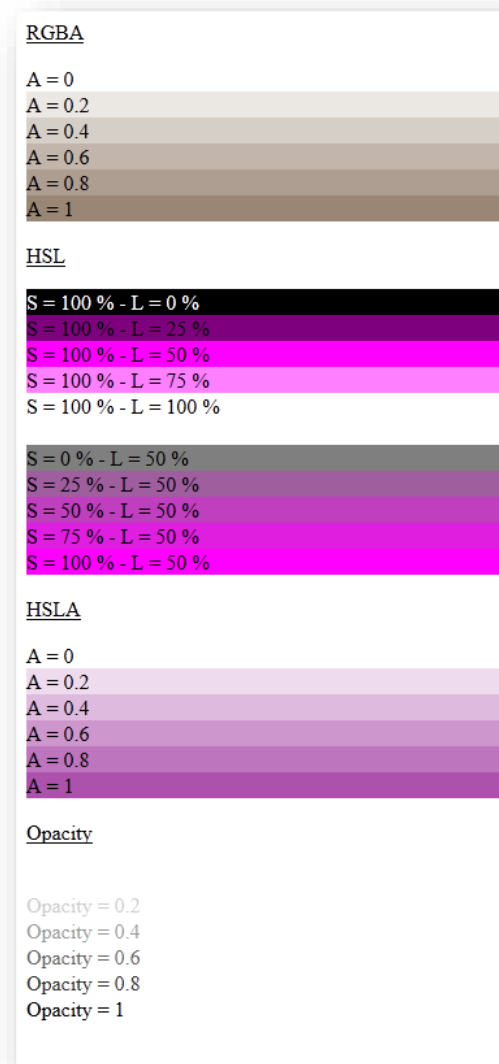
**Ici un texte**

(parent opacity: 0.5)

# HTML5 – CSS3

- Exercice 5.0 – Couleur

*Reproduire le Printscreen de droite.*



# HTML5 – CSS3

- Bordures

- Avec le CSS3, on atteint un niveau supérieur avec la possibilité d'utiliser des dégradés, des coins arrondis, des ombres et des bordures d'images.
- Bordures arrondies : border-radius

```
border-radius: 5px;
```

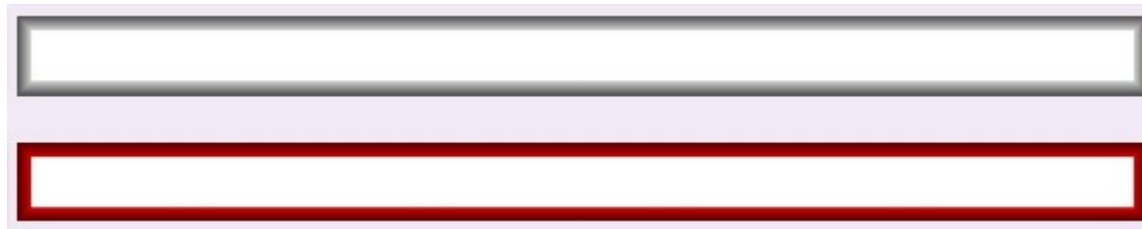
- Exemple



```
.my_border_radius {  
    background-color: #f3de89;  
    border: 1px solid #1120ed;  
    border-radius: 5px;  
}
```

# HTML5 – CSS3

- Bordures
  - Bordures dégradées



```
.my_gradient_border_color {  
    border: 8px solid #000;  
    border-bottom-colors: #555 #666 #777 #888 #999 #aaa #bbb #ccc;  
    border-top-colors:    #555 #666 #777 #888 #999 #aaa #bbb #ccc;  
    border-left-colors:   #555 #666 #777 #888 #999 #aaa #bbb #ccc;  
    border-right-colors:  #555 #666 #777 #888 #999 #aaa #bbb #ccc;  
    padding: 5px 5px 5px 15px;  
}
```

# HTML5 – CSS3

- Bordures

- boîtes à ombre portée



Exemple de boîte avec une ombre portée.

```
.my_box_shadow {  
    -moz-box-shadow: 6px 20px 10px #811819;  
    -webkit-box-shadow: 6px 20px 10px #811819;  
    box-shadow: 6px 20px 10px #811819;  
}
```

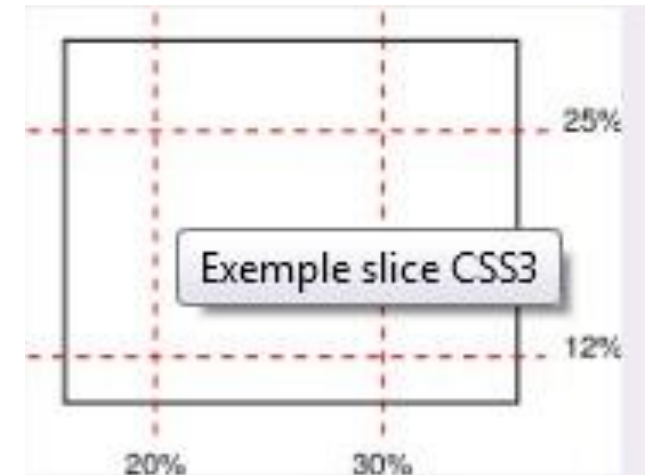
- Voici la signification des paramètres :

- Horizontal length : définit le décalage horizontal de l'ombre
      - valeur positive : l'ombre sera sur la droite de la boîte
      - valeur négative : l'ombre sera sur la gauche de la boîte
      - 0px : l'ombre sera au même niveau que la boîte
    - Vertical length : comme ci-dessus mais verticalement
      - valeur positive : l'ombre sera en dessous de la boîte
      - valeur négative : l'ombre au dessus de la boîte
      - 0px : l'ombre sera au même niveau que la boîte
    - Blur radius : permet d'affecter un effet de flou à l'ombre (0px : l'ombre sera nette)
    - Il s'agit de la couleur de l'ombre

# HTML5 – CSS3

## Bordures en images

- Cette propriété prend 5 attributs en paramètres :
  - source : url de l'image à utiliser comme bordure
  - slice : valeur qui sépare l'image en la divisant en 9 régions
    - 4 pourcentages
    - 4 nombres représentant des pixels
    - fill : mot clé qui force l'utilisation du milieu de l'image (par défaut, le milieu est considéré comme vide)
    - exemple de "slice" ayant pour valeurs 25% 30% 12% 20% :
  - width (optionnel) : 4 valeurs (nombres ou pourcentages) pour définir la largeur des côtés
  - outset(optionnel) : permet de renseigner 4 nombres permettant le rendu de l'image à l'extérieur de son conteneur
  - repeat : spécifie comment les côtés et le centre de l'image sont redimensionnés et répétés. Il faut donc renseigner cet attribut 2 fois (1 pour les côtés, 1 pour le centre). S'il n'est présent qu'une fois, on considère que les 2 sont identiques :
    - stretch : l'image est étirée pour remplir la zone
    - repeat : l'image est répétée pour remplir la zone
    - round : l'image est répétée. Si un nombre entier ne remplit pas la zone, l'image est adaptée.
    - space : l'image est répétée. Si un nombre entier ne remplit pas la zone, l'espace supplémentaire est réparti.





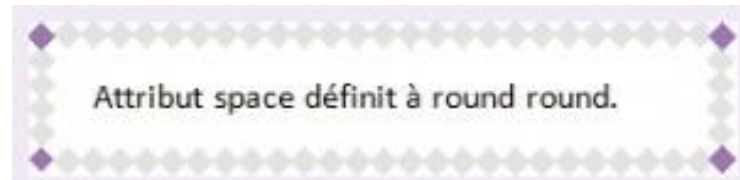
# HTML5 – CSS3

- Bordures en images

- Exemple1 avec comme base cette image (attribut round)

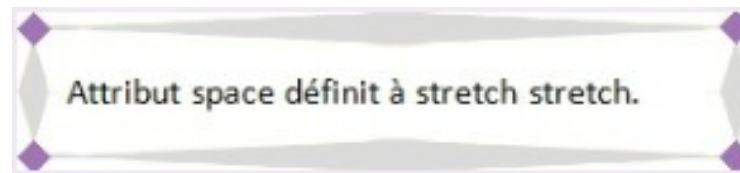


```
.my_border_image {  
    border-image: url(border.png) 27 27 27 27 round round;  
}
```



- Exemple2 avec la même image : attribut stretch

```
.my_border_image {  
    border-image: url(border.png) 27 27 27 27 stretch stretch;  
}
```



# HTML5 – CSS3

- Exercice 5.1 – Bordures

*Reproduire le Printscreen de droite.*

## Les bordures CSS3

### Les dégradés

Ceci est un exemple de boîte avec une bordure dégradée

### Coins arrondis

Ceci est un exemple de boîte à coins arrondis

### Les bordures à ombre portée

Ceci est un exemple de boîte à ombre portée

### Les bordures à images

Ceci est un exemple de boîte avec une image comme bordure

# HTML5 – CSS3

- Effets de texte
  - Ombre portée : text-shadow
    - Les ombres pourraient certainement être mises à profit dans les entêtes et les titres.

```
.text_shadow {  
    color: #897048;  
    background-color: #fff;  
    text-shadow: 2px 2px 2px #ddccb5;  
    font-size: 15px;  
}
```

Les effets de texte CSS3

Les ombres

Ceci est un exemple de texte avec une ombre

# HTML5 – CSS3

- Effets de texte

- Retour à la ligne : word-wrap

- À l'heure actuelle, si un mot est trop long pour tenir sur une ligne d'un bloc, il va déborder.
    - Les nouvelles possibilités de *retour à la ligne* permettent de forcer le texte sur une nouvelle ligne et ce même s'il faut couper un mot.

```
.text_wrap {  
    word-wrap: break-word;  
}
```

*Retour à la ligne*

Ceci est un paragraphe contenant de  
trèstrèslongsetpossible mots et encore  
un  
longmotpourillustrerletutoriel dedesign  
shack

# HTML5 – CSS3

- Effets de texte

- Les points de suspension : text-overflow

- Afin de ne pas dénaturer une mise en page, on est parfois amené à masquer certain contenu à l'aide de overflow.

Les contenus qui débordent de ce bloc sont alors rognés et invisibles. Il peut être utile de laisser un indice visuel pour indiquer la présence de ce contenu masqué.

- C'est là qu'intervient la propriété text-overflow associé à la valeur ellipsis. Des points de suspension (...) sont générés

```
.text_ellipsis {  
    width: 200px;  
    overflow: hidden;  
    -o-text-overflow: ellipsis;  
    text-overflow: ellipsis;  
}
```

points de suspension

Ceci est un paragraphe  
contenant de  
très très long set possible mots et  
encore un  
long mot pour illustrer le tutori...



# HTML5 – CSS3

- Effets de texte

- Les césures (hyphénation) : hyphens

- Cette propriété va bien plus loin que word-wrap puisque la césure est "intelligente" (elle s'adapte aux règles typographiques de la langue employée) et affiche des traits d'union.

```
.text_cesure{  
  width: 200px;  
  -webkit-hyphens: auto;  
  -moz-hyphens: auto;  
  -ms-hyphens: auto;  
  -o-hyphens: auto;  
  hyphens: auto;  
}
```

*Texte avec césure*

Ceci est un paragraphe contenant de très très long set possible mots et encore un long mot pour illustrer le tutoriel de design-shack

- Combiner hyphens et word-wrap

En attendant un bon support navigateur, il est parfaitement possible de combiner les deux propriétés pour le même élément : les navigateurs reconnaissant hyphens l'appliqueront en priorité; les autres disposeront de l'alternative word-wrap.

# HTML5 – CSS3

- Importer des polices dans une page : web Fonts
  - Cette fonctionnalité permet d'afficher une police dite « exotique » aux design mais attention
    - Comme toute œuvre artistique, une fonte est soumise au droits d'auteurs
    - L'ensemble des fichiers sont téléchargé même si vous n'utilisez que quelques caractères => temps d'affichage

- Incorporer depuis le serveur

```
@font-face {  
  font-family: 'Coda Caption';  
  src: url('Coda-Caption-Heavy.eot?#iefix') format('embedded-opentype');  
  url('Coda-Caption-Heavy.woff') format('woff');  
  url('Coda-Caption-Heavy.ttf') format('truetype');  
}
```

- Types supportés (vous pouvez convertir vos fichier grâce à ce type de site : [Font-Face Generator](http://www.font-face-generator.com/)

- EOT : Embedded OpenType (IE)
- TTF : TrueType
- OTF : OpenType
- SVG : Scalable Vector Graphics
- WOFF : Web Open Font Format

- Utilisation de librairie externes

<http://www.google.com/webfonts>

<link href='http://fonts.googleapis.com/css?family=Lobster' type='text/css'>



# HTML5 – CSS3

- Exercice 5.2 – Texte

*Reproduire le Printscreen de droite.*

*La police utilisée est une googleWebFont nommée « Reenie Beanie »*

## Les effets de texte CSS3

### Les ombres

Ceci est un exemple de texte avec une ombre

### Retour à la ligne

Ceci est un paragraphe contenant de très très long set possible mots et encore un long mot pour illustrer le tutoriel de design shack

### points de suspension

Ceci est un paragraphe contenant de très très long set possible mots et encore un long mot pour illustrer le tutoriel...

### Texte avec césure

Ceci est un paragraphe contenant de très très long set possible mots et encore un long mot pour illustrer le tutoriel de design shack

# HTML5 – CSS3

- Multiples colonnes

- Permet de spécifier en combien de colonnes le texte doit être divisé
- Permet de spécifier comment celles-ci doivent apparaître
- 4 propriétés
  - column-count : nombre de colonnes,
  - column-width : largeur,
  - column-gap : espace entre chaque colonne
  - column-rule : Trait de séparation affiché entre les colonnes (comparable a des bordures CSS classiques)
    - Column-rule-color : Couleur du séparateur
    - Column-rule-style : Style du séparateur (solid, dotted, dashed, groove, ridge, etc.)
    - Column-rule-width : Largeur du séparateur
      - » Propriété raccourcie pour l'ensemble : Column-rule

Div{ column-rule : 3px solid rgba(0,0,0,.4);}

# HTML5 – CSS3

```
.multiplecolumns {  
  -moz-column-width: 130px;  
  -webkit-column-width: 130px;  
  -moz-column-gap: 20px;  
  -webkit-column-gap: 20px;  
  -moz-column-rule: 1px solid #ddccb5;  
  -webkit-column-rule: 1px solid #ddccb5;  
}
```

- Multiples colonnes

- Enjambrer une colonne : column-span

```
h2 { column-span: all }
```

- Pas encore supporté sur les navigateurs !

- Exercice 5.3 – Multi-colonnes

*Reproduisez le Printscreen ci-dessus (excepté le column-span, pas encore supporté!).*





# HTML5 – CSS3

- Arrières-plans

- Avant CSS3, la taille de fond était déterminée par la taille de l'image utilisée.

- Taille du fond : background-size

- Cette propriété permet de réutiliser des images dans plusieurs contextes et aussi d'agrandir un fond pour remplir une zone de façon plus précise.

- Propriété background-size : spécifie la taille de l'image dans l'arrière plan

Valeurs possibles :

- » Dimensions (en px ou %) précise la hauteur ou la largeur de l'image
        - » Cover : Forcera à couvrir toute la surface sans déformer l'image. Qui à la rogner.
        - » Contain : Forcera l'image à ne pas dépasser de l'élément sans la déformer.

# HTML5 – CSS3

- Arrières-plans

- Exemple

```
.backgroundsize {  
    background: url(logo.gif);  
    -webkit-background-size: 137px 50px;  
    -o-background-size: 137px 50px;  
    background-size: 137px 50px;  
    background-repeat: no-repeat;  
    padding: 60px 5px 5px 10px;  
    border: 3px solid #ddccb5;  
}
```



# HTML5 – CSS3

- Fonds multiples

- Permet de réaliser des effets qui exigeaient plus d'une *div*.
- L'exemple ci-dessous utilise un fond pour la bordure supérieure, un qui se répète verticalement pour les bordures gauches et droites et un troisième pour le bas.

```
.multiplebackgrounds {  
    height: 150px;  
    width: 270px;  
    padding: 40px 20px 20px 20px;  
    background:  
        url(top.gif) top left no-repeat,  
        url(bottom.gif) bottom left no-repeat,  
        url(middle.gif) left repeat-y; }
```

Lorem ipsum dolor sit amet, consectetur  
adipiscing elit. Nulla commodo. Pellentesque  
mattis velit et nunc. Donec sed sem rhoncus  
ligula vestibulum tincidunt.

Nam gravida. Praesent leo. Etiam rhoncus,  
erat eu iaculis gravida, magna dui tincidunt  
dolor, id sodales eros pede ac risus. Quisque  
aliquet arcu sed sapien. Vivamus vulputate.  
Duis venenatis quam eget quam.

- Les différentes valeurs doivent être ajustées en fonction du nombre d'images à charger et de leurs positions respectives. => l'ordre de déclaration est important

# HTML5 – CSS3

- Exercice 5.4 - Arrieres plans  
*Reproduire le Printscreen ci-contre.*

## Les arrière-plans

### Plusieurs arrière-plans

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla commodo. Pellentesque mattis velit et nunc. Donec sed sem rhoncus ligula vestibulum tincidunt.

Nam gravida. Praesent leo. Etiam rhoncus, erat eu iaculis gravida, magna dui tincidunt dolor, id sodales eros pede ac risus. Quisque aliquet arcu sed sapien. Vivamus vulputate. Duis venenatis quam eget quam.

# HTML5 – CSS3

- Dégradés : gradient
  - Type : linear OU radial
  - Pas de prise en charge native ...
    - Webkit

```
-webkit-linear-gradient();  
-webkit-radial-gradient();
```
    - Gecko

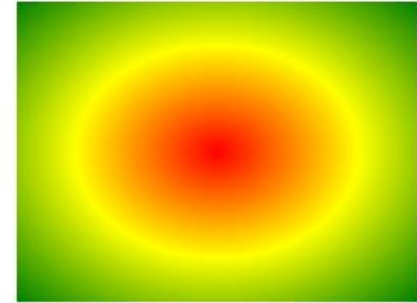
```
-moz-linear-gradient();  
-moz-radial-gradient();
```
    - Opera

```
-o-linear-gradient();  
-o-radial-gradient();
```
    - Syntaxe standard

```
linear-gradient();  
radial-gradient();
```

# HTML5 – CSS3

- Exercice 5.5 -Dégradé  
*Reproduire le Printscreen ci-contre.*





# HTML5 CSS3

- RAPPEL : syntaxe des sélecteurs en CSS2

Motif	Signification
*	Correspond à tout élément.
E	Correspond à tout élément E (c.à.d., un élément de type E).
E F	Correspond à tout élément F qui est un descendant de l'élément E.
E > F	Correspond à tout élément F aussi un enfant de l'élément E.
E:first-child	Correspond à un élément E aussi le premier enfant de son élément parent.
E:link, E:visited	Correspond à un élément E qui est une ancre dans la source dont le lien n'a pas été visité (:link) ou bien l'a déjà été (:visited).
E:active, E:hover, E:focus	Correspond à l'élément E au cours de certaines actions de l'utilisateur.
E:lang(c)	Correspond à l'élément de type E qui emploie une langue c (la détermination de cette langue est spécifique au langage du document).
E + F	Correspond à tout élément F immédiatement précédé par un élément E.
E[foo]	Correspond à tout élément E avec l'attribut "foo" (quelles qu'en soient les valeurs).
E[foo="warning"]	Correspond à tout élément E dont l'attribut "foo" a exactement la valeur "warning".
E[foo~="warning"]	Correspond à tout élément E dont l'attribut "foo" a pour valeur une liste de valeurs séparées par des caractères blancs et dont une de celles-ci est "warning".
E[lang = "en"]	Correspond à tout élément E dont l'attribut "lang" a pour valeur une liste de valeurs séparées par des tirets, cette liste commençant (à gauche) par "en".
DIV.warning	Seulement en HTML. Identique à DIV[class~="warning"].
E#myid	Correspond à tout élément E dont l'ID est "myid".

# HTML5 – CSS3

- Combinateur d'adjacence directe
  - Composés du caractère "~" séparant deux séquences de sélecteurs simples
  - Permet d'ajouter un style à tous les éléments qui suivent un élément particulier.

```
.example4 div{
    margin:0;
    padding:10px;
    color:#000;
}
.example4 div~p{
    color:#fff;
    margin:20px;
    width:200px;
    padding:5px;
    border:1px solid #333;
    background:#006644;
}
```

```
<div class="example4">

    <div>je suis l'élément particulier div</div>
    <p> je suis un p qui suit le div (l'élément
particulier)</p>
    <p>je suis un p qui suit le div (l'élément
particulier)</p>
    <span>je suis un span</span>
    <p>je suis un p qui ne suit pas le div (l'élément
particulier)</p>

</div>
```

je suis l'élément particulier div

je suis un p qui suit le div  
(l'élément particulier)

je suis un p qui suit le div  
(l'élément particulier)

je suis un span

je suis un p qui ne suit pas le div (l'élément particulier)

# HTML5 – CSS3

- Pseudo-classes
  - Pseudo classe **:root**
    - Représente un élément qui est la racine d'un document
  - Pseudo classe **:nth-child(*expression*)**
    - Permet de cibler les éléments en se basant sur leur position dans la liste des enfants de leur parent.
    - ***expression*** peut être un nombre, une expression numérique ou un mot clé tel que "odd" ou "even" (impair et pair), parfait pour faire un style alternatif pour vos tableaux.

```
tr:nth-child(3n) /* tous les 3 enfants */
```

```
tr:nth-child(odd) /* tous les enfants aux numéros impairs */
```

# HTML5 – CSS3

- Pseudo-classes

- pseudo-classe **:nth-last-child(*expression*)**

- Accepte les mêmes arguments que **:nth-child()** et correspond au dernier enfant d'un élément parent.
    - Même principe que **:nth-child()** sauf que l'on part de la fin.

`tr:nth-last-child(-n+2) /*les 2 derniers enfants.*/*`

- **:last-child**

- Correspond à **:nth-last-child(1)**

- **:first-child**

- Correspond à **:nth-child(1)**

# HTML5 – CSS3

- Pseudo-classes

- pseudo-classe **:nth-of-type(*expression*)**

- Représente un élément qui a *expression* frère du même type **devant** lui dans l'arbre DOM.
    - Exemple : alternance de la position des images en flottant

```
img:nth-of-type(2n+1) { float: right; }  
img:nth-of-type(2n) { float: left; }
```

- pseudo-classe **:nth-last-of-type(*expression*)**

- Représente un élément qui a *expression* - 1 frère du même type **après** lui dans l'arbre DOM.
    - Exemple : représente tous les <h2> fils du <body> sauf le premier et le dernier

```
body > h2:nth-of-type(n+2):nth-last-of-type(n+2)
```

# HTML5 - CSS3

- Pseudo-classes
  - **:first-of-type**
    - Correspond à **:nth-of-type(1)**.
    - Représente un élément qui est le premier enfant de son type dans la liste des enfants de son élément parent.
  - **:last-of-type**
    - Correspond à **:nth-last-of-type(1)**.
    - Représente un élément qui est le dernier enfant de son type dans la liste des enfants de son élément parent.
  - **:only-child**
    - Correspond à un élément qui n'a aucun frère.
    - Correspond à **:first-of-type:last-of-type** ou **:nth-of-type(1):nth-last-of-type(1)**



# HTML5 – CSS3

- Pseudo-classes

- **:checked**

- Correspond aux éléments qui sont cochés.

- **:contains(value)**

- Correspond aux éléments dont le contenu textuel contient la sous-chaine donnée en argument.
    - Usage de :contain est restreint aux médias statiques.
    - Exemple :

```
p:contains('essai') {  
    background:#900;  
}
```

- Signifie que tous les éléments "p" contenant la sous chaine "essai" auront pour couleur d'arrière plan, la valeur '#900'

- **:empty**

- Correspond aux éléments qui n'ont pas d'enfants ou qui sont vides.

# HTML5 – CSS3

- Pseudo-classe

- **:not([*expression*])**

- Représente un élément qui n'est pas représenté par l'expression donnée en argument.
    - Exemple:

```
button:not ( [DISABLED] ) {  
    ...  
}  
a:not (:visited) {  
    ...  
}
```

- **:target**

- Représente un élément qui est la cible de l'URI.
    - Exemple :
- Tout élément p qui sera la cible de l'URI (via l'ID # en tant que ancre) aura pour couleur d'arrière plan la valeur "#900".

# HTML5 – CSS3

- Pseudo-éléments : fragments d'éléments d'interface

- **::first-line**

- Applique la règle de style à la première ligne du texte de l'élément.
    - Exemple : la 1ère ligne de ou des éléments "p" est mise en majuscule.

```
p::first-line { text-transform: uppercase }
```

- **::first-letter**

- Applique la règle de style à la première lettre du texte de l'élément.
    - Exemple : La 1ère lettre de ou des éléments "p" a une taille de police de 2em.

```
p::first-letter { font-size: 2em }
```

- **::selection**

- Applique la règle de style à la sélection du texte de l'élément faite par l'utilisateur.
    - L'exemple suivant permet de modifier la couleur de fond du texte en cours de sélection.

```
::selection { background-color: blue; }
```

- **::before** et **::after**

- Génère un contenu avant ou après un contenu d'un élément
    - Ces pseudo-éléments existent en CSS2.1 sous forme **:before** et **:after**.

# HTML5 – CSS3

- Exercice 5.6 – Sélecteurs Lignes tableau

*:nth-child(expression)*

1ere ligne
2eme ligne
3eme ligne
4eme ligne
5ere ligne
6eme ligne
7eme ligne
8eme ligne

*:nth-last-child(expression)*

1ere ligne
2eme ligne
3eme ligne
4eme ligne
5ere ligne
6eme ligne
7eme ligne
8eme ligne

# HTML5 – CSS3

- Sélecteurs CSS3

- Sélecteurs d'attributs

- [attr^="val"] : sélectionner un élément DOM dont l'attribut "attr" commence exactement par la valeur "val".
    - [attr\$="val"] : sélectionner un élément DOM dont l'attribut dont la valeur finit exactement par le suffixe "val".
    - [attr\*="val"] : sélectionner un élément DOM dont l'attribut dont la valeur contient au moins une fois la sous-chaîne "val".

# HTML5 – CSS3

## Exercice 5.7 – Sélecteurs attributs

### *[attr^="stringValue"]*

je n'ai pas d'attribut title

j'ai un attribut title mais il ne commence pas par "ess"

j'ai un attribut title commençant par "ess"

j'ai un attribut title commençant par "ess" également

### *[attr\$="stringValue"]*

je n'ai pas d'attribut title

j'ai un attribut title mais il ne commence pas par "ess"

j'ai un attribut title commençant par "ess"

j'ai un attribut title commençant par "ess" également

### *[attr\*="stringValue"]*

je n'ai pas d'attribut title

j'ai un attribut title mais il ne contient pas "val"

j'ai un attribut title contenant au moins "val"

j'ai un attribut title contenant au moins "val" également

j'ai un attribut title contenant au moins "val" également

# HTML5 – CSS3

- Transformations 2D : transform2d
  - Centre de la déformation : `transform-origin : x y`
    - Point d'origine de la déformation
    - Prend deux types de valeurs
      - soit **numérique** : 100 px, 50 % ...
      - soit **alphabétique** : left, top, right, bottom

`transform-origin: 0 0;`

ou

`transform-origin: top left;`

- Combinaison de transformations :
  - La propriété **transform** autorise plusieurs valeurs de transformation à la suite.
  - Ainsi vous pourrez coupler les différents types de transformation entre eux :

`transform: skew(30deg, 15deg) translate(10px, 0px) rotate(-30deg);`



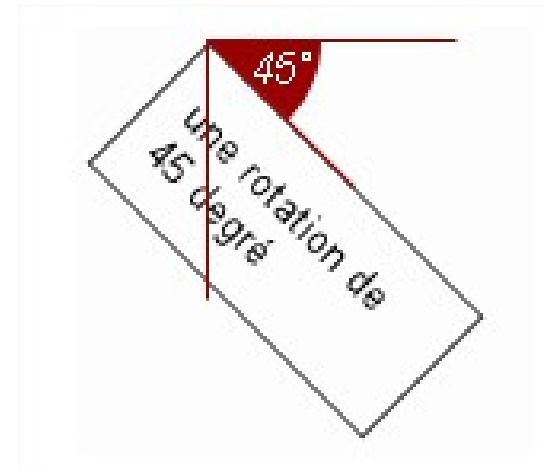
# HTML5 – CSS3

- Transformations 2D

- La rotation :

- La rotation est possible via la propriété **transform** prenant pour valeur **rotate(x)**.
    - L'argument de **rotate** correspond à la valeur de l'angle de rotation et peut être négatif.
    - **Attention** : Sous les navigateurs Webkit (Chrome et Safari), on ne peut pas appliquer la transformation à n'importe quel tag HTML ( par exemple, le span ne permet pas de transformation de type rotation).
    - Exemple :

```
transform: rotate(45deg);
```



# HTML5 – CSS3

- Transformations 2D

- Redimensionnement

- **scale** de la propriété **transform** permet de dilater/redimensionner un élément. Cette valeur peut prendre **un ou deux arguments** : **scale(x)** ou **scale(x, y)**.
    - **scale(x)** : **x** correspond au coefficient de dilatation en **largeur (width)** et en **hauteur (height)**.
    - **scale(x, y)** : **x** correspond à la **largeur (width)** et **y** à la **hauteur (height)**.
    - On peut utiliser les sous-valeurs de **scale** : **scaleX** et **scaleY**.
    - Exemples :

```
transform: scale(2);
```

ou

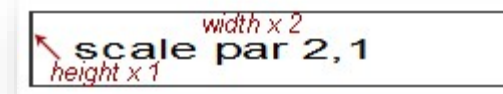
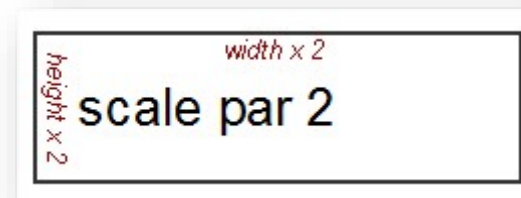
```
transform: scale(2, 1);
```

ou

```
transform: scaleX(2) scaleY(1);
```

ou

```
transform: scaleX(2);
```



# HTML5 – CSS3

- Transformations 2D

- Distorsion

- **skew** de la propriété **transform** permet de tordre un élément.
    - Cette valeur peut prendre **un ou deux arguments** (unité en degré).
    - Un argument équivaut à une distorsion horizontale.
    - Le deuxième argument, contrôle la distorsion verticale.

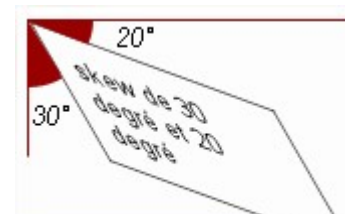
- Exemple : glissement horizontal

```
transform: skew(30deg);
```



- Exemple : glissement horizontal et vertical

```
transform: skew(30deg, 20deg);
```



- Exemple : avec des arguments négatifs

```
transform: skew(30deg, -15deg);
```



- comme **scale**, **skew** peut être utilisé sous cette forme

```
transform: skewY(30deg) skewX(deg);
```

# HTML5 – CSS3

- Transformations 2D

- translate** de la propriété **transform** permet de déplacer un élément en **x** et **y** par rapport à sa position d'origine. Cette valeur peut prendre un ou deux arguments.
- Le premier argument correspond à une translation en **x** (horizontale) et le deuxième argument à une translation en **y** (verticale).
- Ces arguments peuvent être négatifs.
- translate** peut être utilisé sous cette forme :

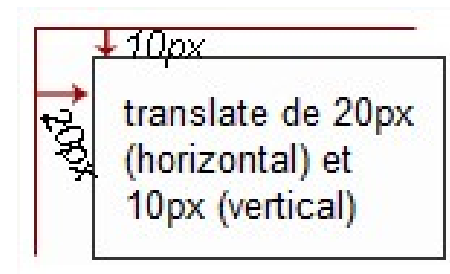
```
transform: translateY(20px) translateX(10px);
```

- Exemple : translation horizontale de 20px et translation verticale de 10px

```
transform: translate(20px, 10px);
```

ou

```
transform: translate(20px);
```



# HTML5 – CSS3

- Exercice 5.8 - Transformations  
*Reproduire le Printscreen ci-contre.*

**Rotate sur texte**

**HTML5**

**Skew et vidéo**



**Translate sur rollover (avec transition)**

translate de  
50px sur la  
droite et 20px  
vers le bas sur  
un roll over

**Scale sur rollover (avec transition)**

scale par 2 sur  
un roll over

# HTML5 – CSS3

- Animations : Enchaînement d'effets
  - Animations = Enchaînement d'effets = Transitions
  - Un évènement CSS fait appel à une animation

```
a:hover{  
  -webkit-animation: pulse 0.7s ease;  
}
```

- Déclaration de l'animation  
Gestion d'une 'timeline' pour les étapes (10%, 20%...)

```
@-webkit-keyframes pulse {  
  from {...}  
  0% {...}  
  10{...}  
  to {...}  
}
```

# HTML5 – CSS3

- Transitions : effets de transitions de propriétés CSS

- transition-property : la propriété modifiée par la transition

All ou ls propriétés CSS

- Transition-timing-function : fonction de transition utilisée

ease, linear, ease-in, ease out, ease-in-out

- Transition-duration : durée de la transition

secondes

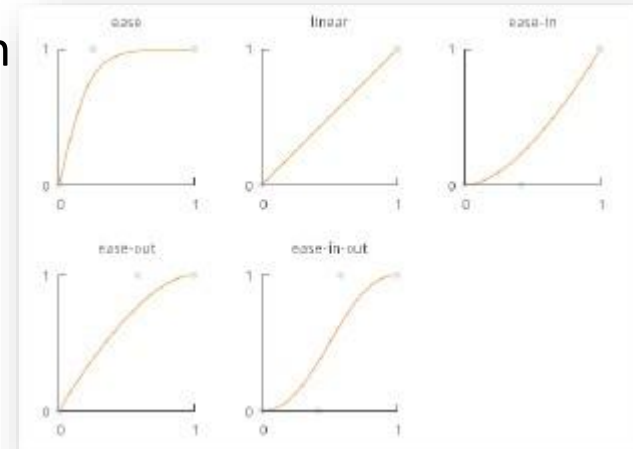
- Transition-delay : temps avant le déclenchement de la transition

secondes

- Exemple :

```
<style>
  video {
    width: 100px;
    transition: all 0.5s ease-in-out;
  }

  video:hover, video:focus {
    width: 600px;
    transform: rotate(45deg);
  }
</style>
```





# HTML5 – CSS3

- Ressources:
  - <http://www.css3.info/>
  - <http://www.w3schools.com/css/default.asp>

# HTML5 – CSS3

- Les Media Queries

- Définition

- Technique pour l'application de feuilles de styles en fonction des périphériques utilisé pour visualiser un site internet => Responsive Web Design
    - But est de satisfaire des contraintes de dimensions, résolutions pour améliorer l'apparence graphique et la lisibilité d'un site web

- Syntaxe

- Les Media Queries utilisent des opérateurs logiques
      - And (et)
      - Only (uniquement)
      - Not (non)
      - Pour obtenir l'équivalent du « ou », on énumère les différentes média queries à la suite séparée par des virgules.

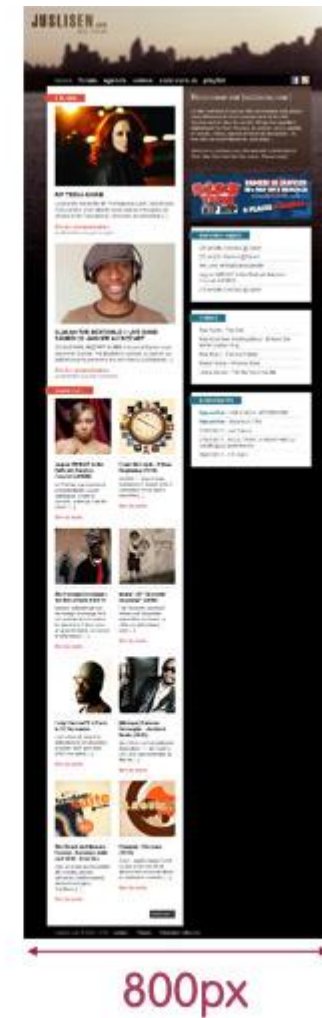
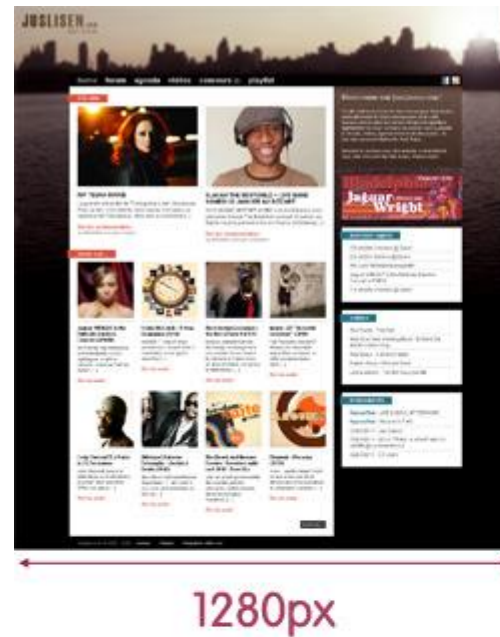
- Exemple

```
<link rel="stylesheet" media="screen and (max-width: 640px)" href="smallscreen.css" type="text/css" /> <!--HTML-->
```

```
@media screen and (min-width: 200px) and (max-width: 640px) {  
  .bloc { display:block; clear:both; }  
} /*CSS*/
```

# HTML5 – CSS3

- Les Media Queries
  - Voici un exemple de design pour un site en Responsive Web Design



Création : <http://www.juslisten.com/>

# HTML5 – CSS3

- Les Media Queries
  - Fonctionnalités (la plupart peuvent être préfixée par min et max (si les valeurs sont numériques))
    - Color, color-index : couleur
    - Aspect-ratio : ratio du périphérique (ex: 16/9)
    - Device-aspect-ratio : ratio de la zone d'affichage (ex : fractions)
    - Device-height , device-width : dimension en hauteur/largeur du périphérique (ex : px ou em)
    - Height, width: dimension de la zone d'affichage (ex : px ou em)
    - Orientation : portrait ou paysage
    - Resolution : (ex : en dpi ou dpcm)
  - Support des navigateurs
    - Mozilla Firefox : 3.5+
    - Internet Explorer : 9+
    - Google Chrome : 5+
    - Opera : 10.5+, Opera Mobile : 10+, Opera Mini : 5+
    - Apple Safari : 4+ et iOS (mobile) 3.2+
    - Android : 2.1+
    - WebKit en général

# HTML5 – CSS3

- Exercice 5. – Responsive Web Design

Créez le visuel suivant et appliquez une couleur de fond en fonction des éléments répondant aux règles media ci-dessous

- Règle(s) à appliquer :
  - @media screen and (max-width: 640px)
  - @media screen and (min-width: 800px)
  - @media screen and (min-width: 1024px) and (max-width: 1280px)
  - @media screen and (max-device-width: 480px)
  - @media screen and (orientation:portrait)
  - @media screen and (orientation:landscape)

# HTML5 – CSS3

- Exercice 5.10 – Responsive Web Design

Voici ce que vous devriez obtenir en fonction du redimensionnement de votre navigateur

La page est affichée avec une largeur < 640px
La page est affichée avec une largeur > 800px
La page est affichée avec une largeur > 1024px et < 1280px
La page est affichée sur un périphérique de maximum 480px (ex: iPhone)
La page est affichée en portrait
La page est affichée en paysage

La page est affichée avec une largeur < 640px
La page est affichée avec une largeur > 800px
La page est affichée avec une largeur > 1024px et < 1280px
La page est affichée sur un périphérique de maximum 480px (ex: iPhone)
La page est affichée en portrait
La page est affichée en paysage

La page est affichée avec une largeur < 640px
La page est affichée avec une largeur > 800px
La page est affichée avec une largeur > 1024px et < 1280px
La page est affichée sur un périphérique de maximum 480px (ex: iPhone)
La page est affichée en portrait
La page est affichée en paysage

La page est affichée avec une largeur < 640px
La page est affichée avec une largeur > 800px
La page est affichée avec une largeur > 1024px et < 1280px
La page est affichée sur un périphérique de maximum 480px (ex: iPhone)
La page est affichée en portrait
La page est affichée en paysage

# HTML5 – CSS3

- Polices, quelle taille choisir ?
    - Les unités (RELATIVES) de la propriété font-size
      - Em : cadratin qui se base sur la hauteur de la police
      - Rem : Equivalent de em mais utilisant comme base de calcul la valeur de :root
      - Ex : qui se base sur la hauteur de la lettre « x » de la police (diffère selon les navigateurs)
      - Px : le pixel
    - Les unités (ABSOLUES) de la propriété font-size
      - Pt : le point
      - Pc : le pica (12 points)
      - Cm & mm
      - In : le pouce
- 1 in = 2.54cm = 25.4 mm = 72 pt = 6pc

**Attention** : l’affichage de ce type d’unité dépend de la résolution et de la dimension de l’écran => plutôt destiné à l’impression



# HTML5 – CSS3

- Polices, quelle taille choisir ?
  - L'idéal est d'attribuer des valeurs proportionnelles au corps de base (unité em, rem ou %)
  - *Pour obtenir un texte correctement lisible sur toutes les configurations, il est préférable de ne pas imposer de taille de texte « de base » fixe (px), mais d'utiliser la taille du texte par défaut du navigateur (em ou rem).*
- Le EM : exemple `P { font-size: 2em; }`  
la taille de la police du paragraphe sera 2 fois plus grande que le corps de base (défini du body ou du parent)

```
html {font-size: 100%;}/* hack IE */  
body {font-size: .8em;}
```

Dans cet exemple, on détermine une taille de texte "globale".

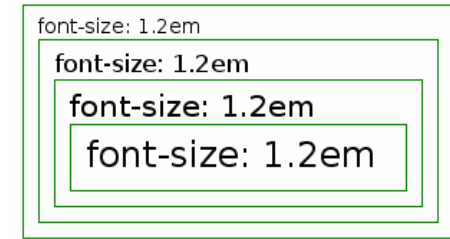
Tous les enfants de l'élément "body" auront comme taille 80% de la taille du texte par défaut

# HTML5 – CSS3

- Polices, quelle taille choisir ?

Attention, pensez à la cascade CSS !

Attention à ne pas déclarer de « font-size: x em ; » à tout bout de champ ! Sinon les valeurs se multiplieront entre elles.



- Nouveauté CSS3 : le REM ce qui signifie « root em »

- Cette unité est relative à l'élément racine : la balise HTML

On ne définit qu'une seule taille de police sur l'élément html puis on compose toutes les tailles en REM comme pourcentage de cette valeur initiales.

Il n'y a donc plus de soucis de cascade ! Moins de calcul tout en garantissant l'accessibilité de la page.

- Que faire pour les navigateurs qui n'utilise pas le Rem ? On spécifie une unité de secours en pixel puis une seconde fois en utilisant l'unité REM.

```
html { /* =10px */ font-size: 62.5%; }  
body { /* =14px */ font-size: 14px; font-size: 1.4rem;}  
h1    { /* =24px */ font-size: 24px; font-size: 2.4rem;}
```

# HTML5 – CSS3

- Exercice 5.11 – Unité de mesure

# HTML5 – CSS3

- Flexbox Layout module

- Positionner correctement des éléments est la base du rôle du CSS. Il y a 4 type de rendus d'éléments

- Le rendu « bloc »
- Le rendu « inline »
- Le rendu « tabulaire »
- Le rendu « positionné »

Chacune des valeurs de la propriété display, float ou position renvoie à l'un de ces quatre types de rendus

- En CSS3, il y a un nouveau paramètre avec un nouveau modèle de boîte « flexible » introduit via la propriété display. Cette propriété permet de créer un contexte général sur un parent et d'en faire hériter ses enfants

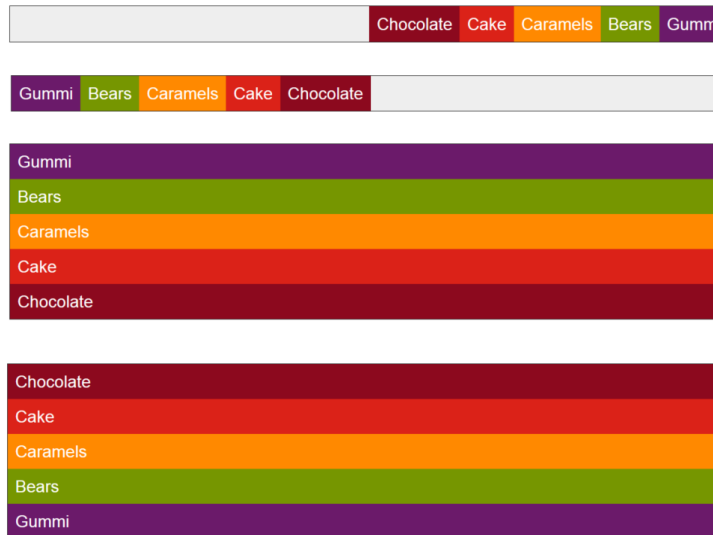
- Distribution en bloc ou ligne
- Alignement horizontaux et verticaux
- Gestion des espaces disponibles (fuidité)
- Réorganisation des éléments indépendant de l'ordre du flux

# HTML5 – CSS3

- Flexbox Layout module, ses propriétés

- Flex-direction

- Row
    - Row-reverse
    - Column
    - Column-reverse



- Flex-wrap

- Nowarp : les éléments demeurent toujours sur une ligne et peuvent déborder
    - Wrap : les éléments passent à la ligne
    - Wrap-reverse : les éléments passent à la ligne en inversant leur direction

# HTML5 – CSS3

- Flexbox Layout module, les alignements horizontaux et verticaux
  - Les alignements sur l'axe horizontal sont traité avec la propriété « justify-content »

- Flex-start



- Flex-end



- Center



- Space-between

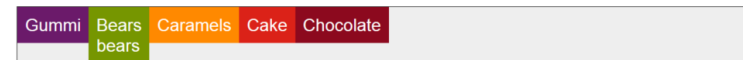


- Space-around

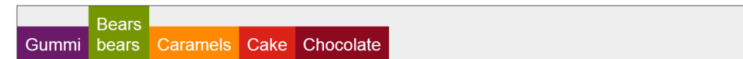


- Les alignements sur l'axe vertical sont traité avec la propriété « align-items »

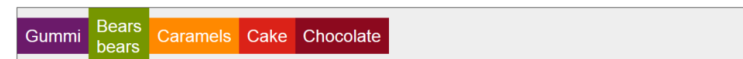
- Flex-start



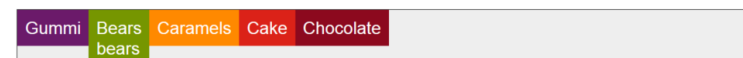
- Flex-end



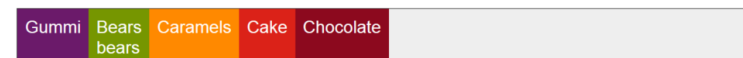
- Center



- Baseline



- Stretch



# HTML5 – CSS3

- Flexbox Layout module, les alignements gérer les exceptions
  - La propriété align-self offre la possibilité de distinguer un élément particulier de ses frères en adoptant un alignement différent.

```
div> li:last-child {align-self: flex-end;}
```

- La propriété margin et sa valeur auto permettent de positionner un élément en dehors de la distribution de ses frères => possible de placer un élément à droite alors que les autres sont aligné à gauche

```
div{justify-content: flex-start;}
```

```
div > li:last-child { margin-left: auto;}
```



# HTML5 – CSS3

- Flexbox Layout module, les alignements gérer les exceptions
  - La propriété flex est un raccourci de trois propriétés
    - flex-grow : s'étirer dans l'espace restant
    - flex-shrink : se contracter si besoin
    - flex-basis : taille initiale de l'élément avant que l'espace restant ne lui soit distribué
  - La propriété order permet de réordonner à sa guise chacun des éléments indépendamment.

Les valeurs de order agissent telles des pondérations : les éléments dont la valeur est la plus forte se trouveront en base de la pile.

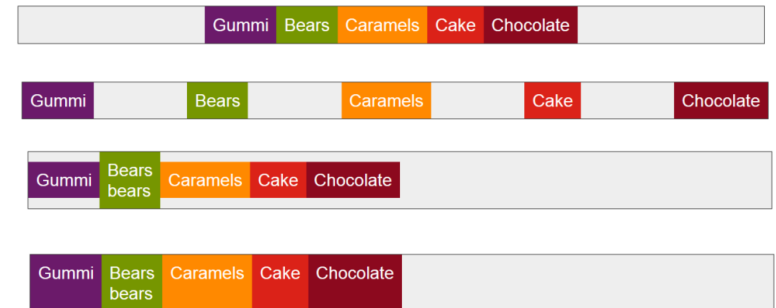
La valeur initiale de order est 0

```
/* le premier de la liste s'affichera en bas de pile */  
li:first-child {  
    order: 1;  
}
```

# HTML5 – CSS3

- Exercice 5.12 – Flex box

Reproduisez la capture d'écran suivante



# HTML5 – CSS3

- Flexbox Layout module, pour aller plus loin
  - Flexbox est une spécification complexe, nous n'en avons démontré que quelques exemples de ce qui est possible  
Pour aller plus loin, voici quelques liens intéressants
    - [css-tricks.com](http://css-tricks.com) : "Old" Flexbox and "New" Flexbox
    - [smashingmagazine.com](http://smashingmagazine.com) : CSS3 Flexible Box Layout Explained
    - [net.tutsplus.com](http://net.tutsplus.com) : An Introduction to the CSS Flexbox Module

# HTML5 – CSS3

- Bootstrap

- Historique

- Bootstrap a été créé en 2011 chez Twitter comme une solution d'usage interne afin de minimiser les incohérences de code => création du framework
    - Bootstrap est « le plus populaire des frameworks front-office pour développer des projets responsive et mobile-first sur le web »

- Qu'est-ce que Bootstrap ?

- Bootstrap est une compilation de plusieurs de plusieurs éléments et fonctions web-design personnalisables
    - Une combinaison d'HTML, CSS et Javascript
    - Open Source => amélioration permanente comme 100% mobile responsive, plugins jQuery
    - Août 2013 nouvelle version Bootstrap 3

