

Cascading Style Sheets



- Événements
- Bordures
- Dégradés
- Transformations
- Animations
- Type d'écran

Les événements

- Souris au-dessus de l'élément
⇒ événement : **hover**, s'écrit :

```
nomBalise: hover{  
  // style à appliquer pour cet état  
}
```

```
<div class="survol">div avec gestion de passage de souris</div>  
<div class="clique">div avec gestion de clique de souris</div>
```

```
div.survol: hover{  
  background-color: greenyellow;  
}  
  
div.clique: active{  
  background-color: red;  
}
```

- Élément actif (clic gauche sur l'élément)
□ événement : **active**, s'écrit :

```
nomBalise: active{  
  // style à appliquer pour cet état  
}
```

```
div avec gestion de passage de souris  
div avec gestion de clique de souris
```


Bordures

Il est possible de mettre une image en tant que bordure d'une boîte. Ceci s'écrira :

border-image : url(n) s w r

- n : nom de l'image
- s : taille des "coins" de l'image
- w : largeur du contour
- r : répétition ou pas de l'image, hors "coins" (round ou stretch)



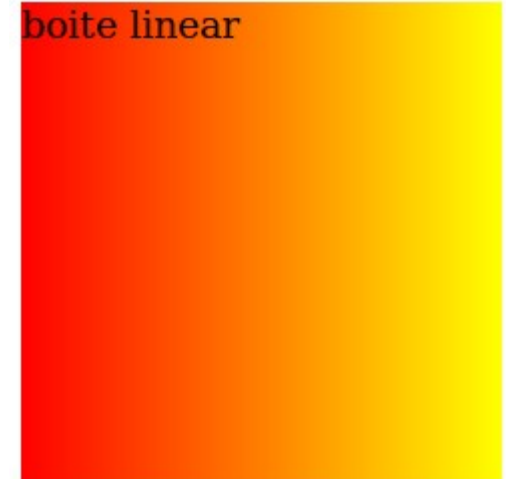
Dégradés

Il est possible de créer un dégradé linéaire.

Ceci s'écrira : **background : linear-gradient(dir,cd,cf) ;**

- dir : direction du dégradé
- cd : couleur début
- cf : couleur fin

```
div.boiteLinear{  
  background: linear-gradient(to left,yellow,red)  
  width: 200px;  
  height: 200px;  
}
```

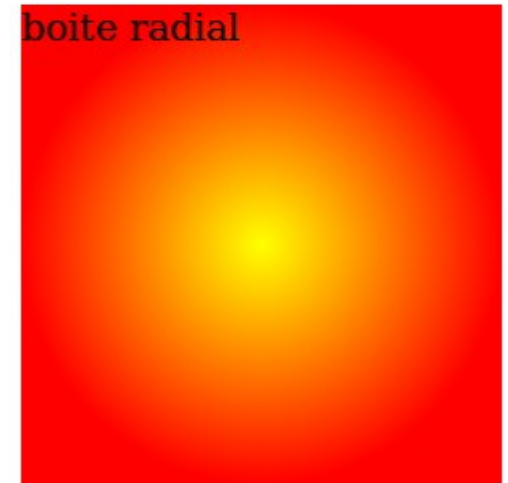


Dégradés

Il est possible de créer un dégradé radial. Ceci s'écrira : **background : radial-gradient(r,cd,cf) ;**

- r : rayon du dégradé
- cd : couleur début
- cf : couleur fin

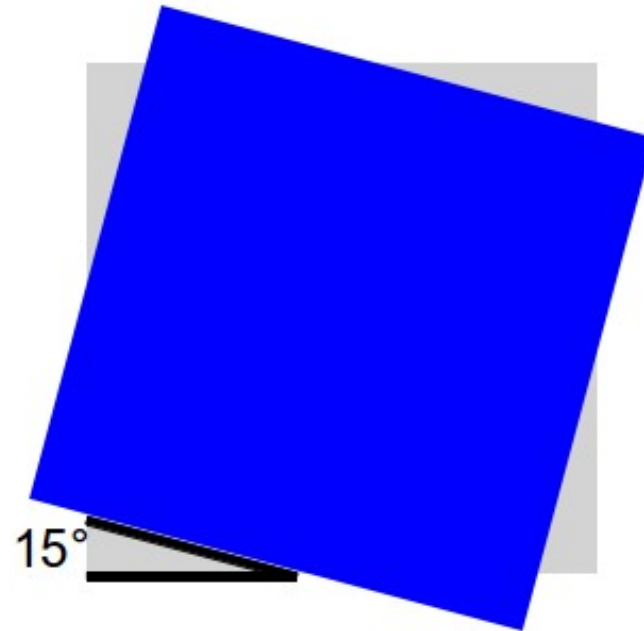
```
div.boiteRadiale{  
  background: radial-gradient(100px,yellow,red);  
  width: 200px;  
  height: 200px;  
}
```



Transformations

Il est possible de réaliser une rotation d'un élément.
Ceci s'écrira : **transform : rotate(angleEnDegre) ;**

```
div.pasrot{  
  background: lightgray;  
  width: 200px;  
  height: 200px;  
  position: absolute;  
  left: 300px;  
  top: 300px;  
}  
div.rot{  
  background: blue;  
  width: 200px;  
  height: 200px;  
  position: absolute;  
  left: 300px;  
  top: 300px;  
  transform: rotate(15deg);  
}
```

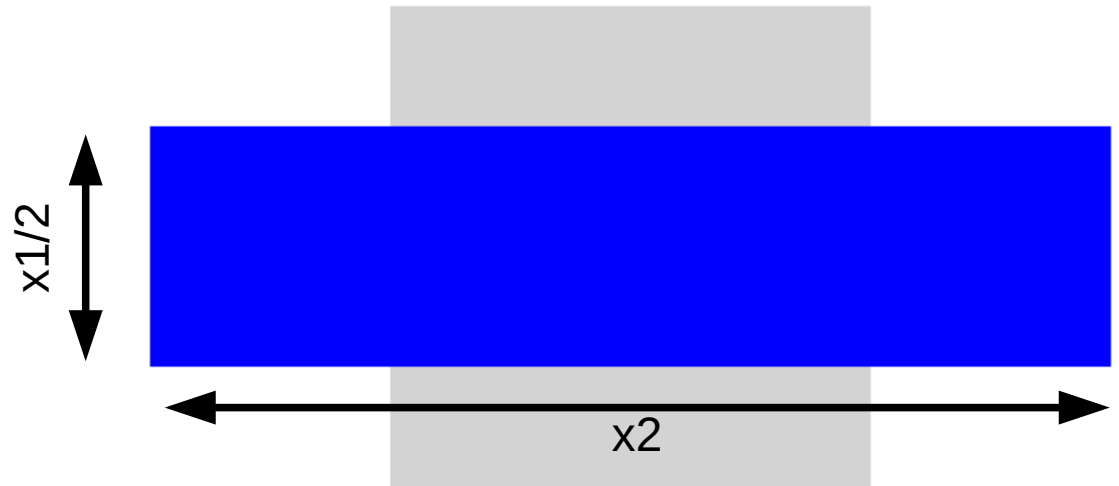


Transformations

Il est possible de réaliser une mise à l'échelle d'un élément (agrandissement / réduction). Ceci s'écrira :

transform : scale(ajustementLargeur,ajustementHauteur) ;

```
div.pascal{  
  background: lightgray;  
  width: 200px;  
  height: 200px;  
  position: absolute;  
  left: 300px;  
  top: 300px;  
}  
div.scal{  
  background: blue;  
  width: 200px;  
  height: 200px;  
  position: absolute;  
  left: 300px;  
  top: 300px;  
  transform: scale(2,0.5);  
}
```

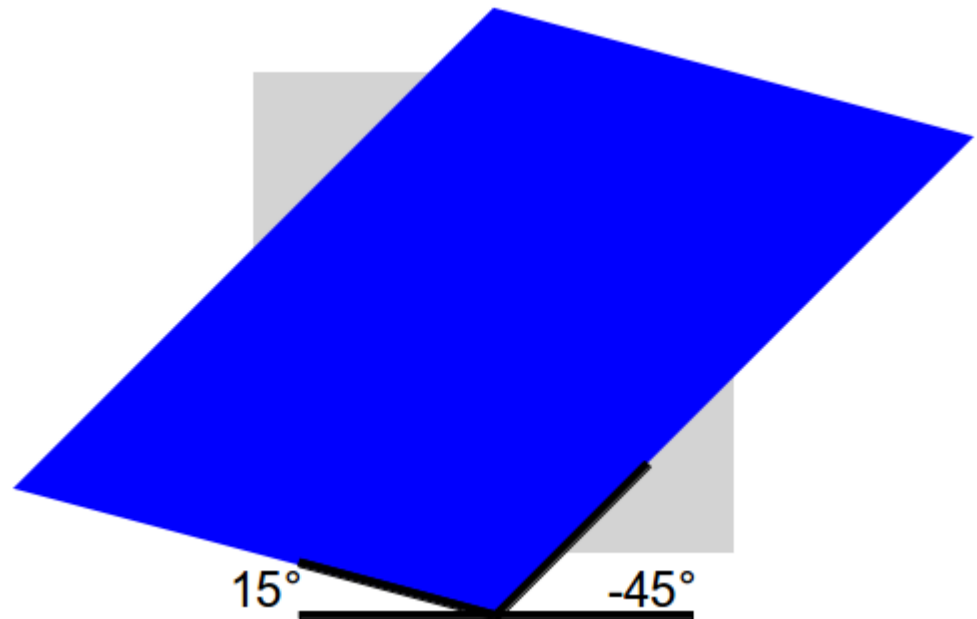


Transformations

Il est possible de réaliser une déformation oblique d'un élément. Ceci s'écrira :

transform : skew(angleVertical,angleHorizontal) ;

```
div.passkew{  
  background: lightgray;  
  width: 200px;  
  height: 200px;  
  position: absolute;  
  left: 300px;  
  top: 300px;  
}  
div.avecskew{  
  background: blue;  
  width: 200px;  
  height: 200px;  
  position: absolute;  
  left: 300px;  
  top: 300px;  
  transform: skew(-45deg,15deg);  
}
```



Animations

Il est possible d'animer des éléments.

Ceci s'écrira : **animation : ani dur type tempo nb re ;**

- ani : nom de la liste des points clefs pour les déplacements
- dur : durée de l'animation
- type : vitesse de l'animation (constante ou avec des variations)
- tempo : temporisation avant le début de l'animation
- nb : nombre de fois où l'animation devra être faite (peut être infini)
- re : refaire l'animation en sens inverse ou pas.

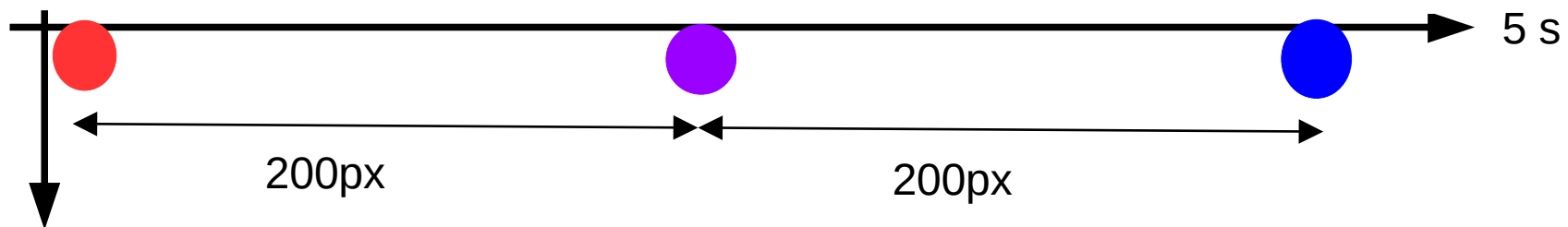
Animations

Exemple d'une animation :

```
div.bouleAnim
{
    width: 30px;
    height: 30px;
    border-radius: 15px;
    background: red;
    position: relative;
    animation: bougeHorizontal 5s linear 0s infinite alternate;
}
```

`<div class="bouleAnim"></div>`

```
@keyframes bougeHorizontal
{
    0% {background: red; left: 0px; top: 0px}
    50% {background: violet; left: 200px; top: 0px}
    100% {background: blue; left: 400px; top: 0px}
}
```



Type d'écran

Il est possible d'appliquer un certain style en fonction des dimensions de l'écran.

Ceci s'écrira : `@media screen and|not|only (media feature)
{
 code css ;
}`

- media feature : conditions d'application du css
(voir : http://www.w3schools.com/cssref/css3_pr_mediaquery.asp)

Type d'écran

```
<!DOCTYPE html>
<html>
  <head>
    <title>TODO supply a title</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" type="text/css" href="tstmedia.css" />
  </head>
  <body>
    <div class="bigCarre">carre qui change de couleur et de dimension en e
fonction de la taille de l'écran</div>
  </body>
</html>
```

```
@media screen and (min-width:640px)
{
  div.bigCarre{
    position: absolute;
    top: 10px;
    left: 10px;
    width: 620px;
    background-color: yellowgreen;
    height: 320px;
  }
}
```

```
@media screen and (min-width:800px)
{
  div.bigCarre{
    position: absolute;
    top: 10px;
    left: 10px;
    width: 780px;
    background-color: blue;
    height: 320px;
  }
}
```


Type d'écran

- Attention, toujours définir de la plus petite résolution à la plus grande.
- L'adaptation des pages en fonctions des écrans est désignée sous l'appellation RWD (Responsive Web Design).
- Il existe des bibliothèques css permettant de simplifier la pratique du RWD comme Bootstrap
- Sites de référence :



http://www.w3schools.com/css/css_rwd_intro.asp