HTML5 en pratique

Multimédia

Plan

- Introduction
- Les balises
- CSS 3
- Javascript, le langage du web
- Vers des application plus interactives
- Gestion des données
- Multimédia
- Conclusion

Plan

- Intégration d'une piste audio
- Intégration d'une piste vidéo
- Utilisation du canvas

Intégration d'une piste audio

- HTML5 possède un tag natif pour embarquer des pistes audio
  - Selon le navigateur, le code nécessaire peut varier

```
<audio controls="controls" src="song.mp3"/>
```

ou

```
<audio controls="controls">
  <source src="music.mp3" type="audio/mp3" />
</audio>
```

- Rendu (spécifique à chaque navigateur)

Intégration d'une piste audio

- Plusieurs sources / codecs peuvent être spécifiés pour s'assurer de fournir un format supporté
  - Ogg
  - Mp3
  - Autre

- Dans le cas où le navigateur n'est pas capable de lire un des formats ou ne supporte pas la balise `audio`, il est possible de spécifier un message d'avertissement

```
<audio controls="controls">
    <source src="music.ogg" type="audio/ogg" />
    <source src="music.mp3" type="audio/mp3" />
    Ce navigateur ne supporte pas l'élément audio.
</audio>
```

Intération avec une piste audio

- La lecture de la piste peut être contrôlée par une API javascript

```
var player = document.getElementById('myPlayer') ;
player.play();
player.pause(),
player.load()
player.volume = ... ;
player.currentTime = ... ;
player.duration = ... ;
player.AddEventListener('play', function(){...}) ;
```

Intégration d'une vidéo

- HTML5 possède un tag natif pour la lecture de vidéos

```
<video controls src="myVideo.mp4"/>
// controls donne accès à la barre de contrôle
```

- Une application de démonstration est disponible sur le site du W3C

Intégration d'une vidéo

- Plusieurs sources ou codecs peuvent être spécifiés pour s'assurer de fournir un format supporté
  - Mp4
  - WebM
  - Ogg
- Un message et/ou composant alternatif peut être défini, si le navigateur ne peut lire aucune des sources proposées

```
<video ...>
    <source type="video/mp4" codecs="avc1.42E01E,mp4a.40.2">
    <source type="video/webm" codecs="vorbis,vp8">
    <source type="video/ogg" codecs="theora,vorbis">
    Ce navigateur ne supporte pas l'élément vidéo.
    <object width="…" height="…"
    type="application/x-shockwave-flash" data="player.swf">
    </object>
</video>
```

Intégration d'une vidéo

- Une API javascript permet d'agir sur la vidéo

```
var player = document.getElementById('myPlayer') ;
player.play();
player.pause(),
player.load()
player.volume = ... ;
player.currentTime = ... ;
player.duration = ... ;
player.AddEventListener('play', function() {...}) ;
```

Générer des images avec Canvas

- L'élément canvas est utilisé pour obtenir une zone de dessin que l'on contrôle avec une API spécifique

```
<canvas width="400" height="400"></canvas>
```

- Attention, définir la taille du canvas en CSS grossit la zone de dessin au lieu de l'agrandir

- Le canvas utilise un système de coordonnées pour pouvoir se repérer
  - L'origine correspond au point X=0, Y=0 (En haut à gauche)
  - La valeur maximale de X est sa largeur
  - La valeur maximale de Y est sa hauteur
- Par défaut, la largeur est de 300px et sa hauteur de 150px

Générer des images avec Canvas

- Pour manipuler l'élément, il faut dans un premier temps obtenir sa référence

```
<body onload="draw()">
<canvas id="canvas" width="400" height="400"></canvas>
<script type="text/javascript">
    function draw(){
        var c=document.getElementById("canvas");
        if(c.getContext){
            var ctx= c.getContext("2d");
            (...)
```

- Le contexte obtenu permet d'accéder à l'API de dessin

Générer des images avec Canvas
Formes géométriques

- Lignes en définissant un point d'origine et un point d'arrivée

```
ctx.moveTo(0,0);      //Aller au point (0,0)
ctx.lineTo(200,200); //Faire une ligne jusqu'au point (200,200)
ctx.stroke();         //Tracer
```

- Rectangles en définissant un point d'origine et une taille

```
ctx.fillRect(0,0,50,50);    // Dessine un rectangle plein
ctx.strokeRect(60,0,50,30); // Dessine un rectangle vide
ctx.clearRect(20,20,10,10); //Efface une zone
```

Générer des images avec Canvas
Formes géométriques

- Cercles ou arc de cercles

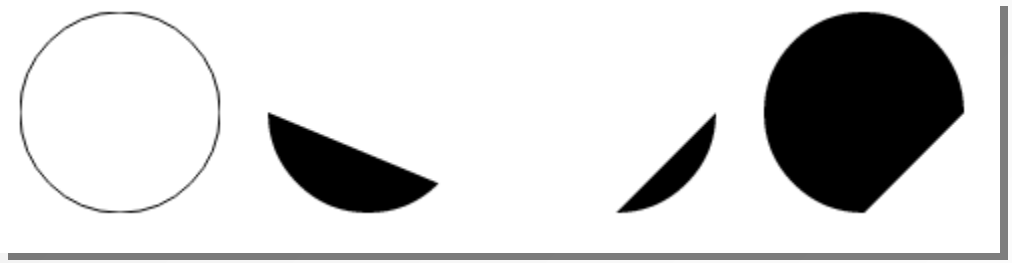
```
ctx.arc(x,y,                        // centre du cercle
        rayon,                      // rayon
        angleDepart, angleFin,      // angle
        sens) ;                     // sens inverse ou non
```

- Les angles sont des valeurs en radians. Pour passer de degrés à radians, on utilise la formule suivante

```
var radians = (Math.PI/180)*degres
```

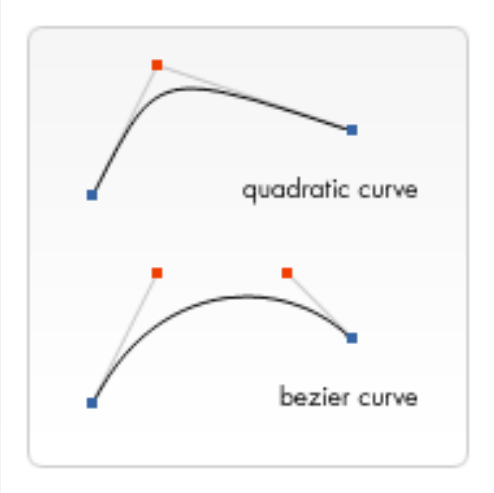
Générer des images avec Canvas
Formes géométriques

```
ctx.arc(50,50,50,0,Math.PI*2,false);
ctx.stroke();
ctx.arc(50,50,110,Math.PI*(1/4),(Math.PI),false);
ctx.fill();
ctx.arc(50,50,170,0,(Math.PI)*0.5,false);
ctx.fill();
ctx.arc(50,50,230,0,(Math.PI)*0.5,true);
ctx.fill();
```



Générer des images avec Canvas
Formes géométriques

- Pour des formes plus complexes, il existe les courbes de bézier et les courbes quadratiques qui sont un peu plus complexes d'utilisation



- Des arrondis se forment en fonction de points de contrôle, un seul pour les courbes quadratiques et deux pour les courbes de bézier

```
quadraticCurveTo(ctlX, ctlY, x, y);
bezierCurveTo(ctlX1, ctlY1, ctlX2, ctlY2, x, y)
```

Générer des images avec Canvas
Formes géométriques

```
ctx.moveTo(75,25);
ctx.quadraticCurveTo(25,25,25,62.5);
ctx.quadraticCurveTo(25,100,50,100);
ctx.quadraticCurveTo(50,120,30,125);
ctx.quadraticCurveTo(60,120,65,100);
ctx.quadraticCurveTo(125,100,125,62.5);
ctx.quadraticCurveTo(125,25,75,25);
ctx.stroke();
```

Générer des images avec Canvas
Texte

- Il est possible d'ajouter du texte, rempli ou non

```
fillText(Texte, x, y);  // texte plein
strokeText(Texte, x, y);// texte vide
```

- La police et la taille utilisée se définissent avec la propriété font

```
ctx.font = "30px Arial";
ctx.fillText("Mon texte",0,0);
```

Générer des images avec Canvas Images

- Pour charger des images, il existe plusieurs solutions
  - En chargeant les images en HTML

```
<img src="images/zenika.jpg" id="img"
style="display:none;" />
(...)
ctx.drawImage(document.getElementById("img"),x,y);
```

  - Ou entièrement en Javascript

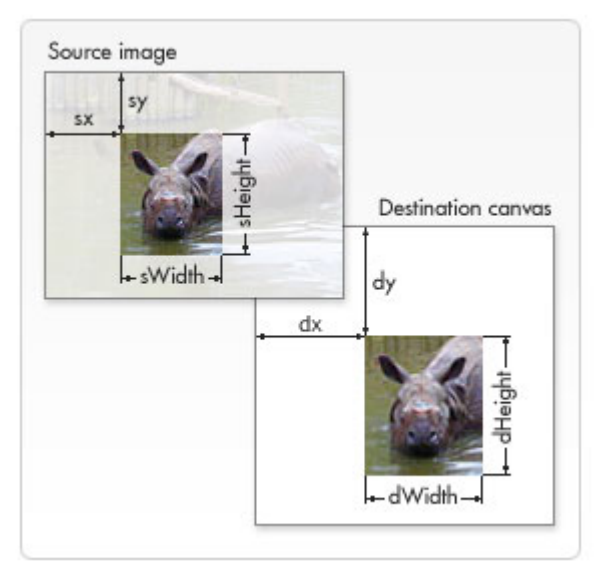
```
var img=new Image();
img.src='images/zenika.jpg';
ctx.drawImage(img,30,30);
```

- La méthode drawImage() permet aussi de redimensionner ou de couper une partie de l'image

```
drawImage(src, x, y, largeur, hauteur)
```

Générer des images avec Canvas Images

```
drawImage(src, sX, sY, sLargeur, sHauteur, dX, dY, dLargeur,
dHauteur);
```



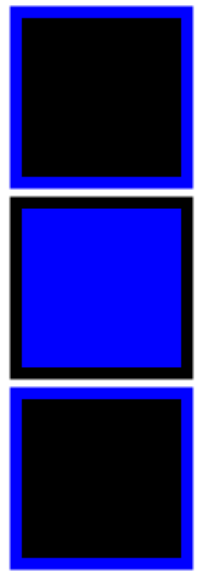
Générer des images avec Canvas Style

- Plusieurs propriétés permettent de personnaliser le style du canvas

```
strokeStyle="#FF0098"    //Couleur des contours
fillStyle="black"        //Couleur de remplissage
GlobalAlpha="0,6"        //Transparence
LineWidth=10             //Epaisseur de ligne
```

- Les méthodes `save()` et `restore()` sont particulièrement utiles, elles permettent de sauver puis de restaurer un style courant

```
ctx.strokestyle="blue"; ctx.fillstyle="black";
//Rectangle
ctx.save();
ctx.strokestyle="black" ; ctx.fillstyle="blue";
//Rectangle
ctx.restore();
//Rectangle
```



Générer des images avec Canvas Animation

- Animer le canvas, c'est possible !
- Le principe est de redessiner tout ou une partie du canvas à intervalles réguliers

```
setInterval(update,500); // Appelle la méthode update toutes les
                         // 500 Millisecondes
```

- Pour arrêter l'animation, `clearInterval()` est utilisé

```
var anim = setInterval(update, 500);
(...)
clearInterval(anim) ;
```

- Animer le canvas en fonction d'événements souris ou clavier du javascript permet de faire des jeux


TP5