Fise-INFO4.3

Panorama des langages de script

Javascript

TD3 - Le Canvas

AMATHIEU Valentin

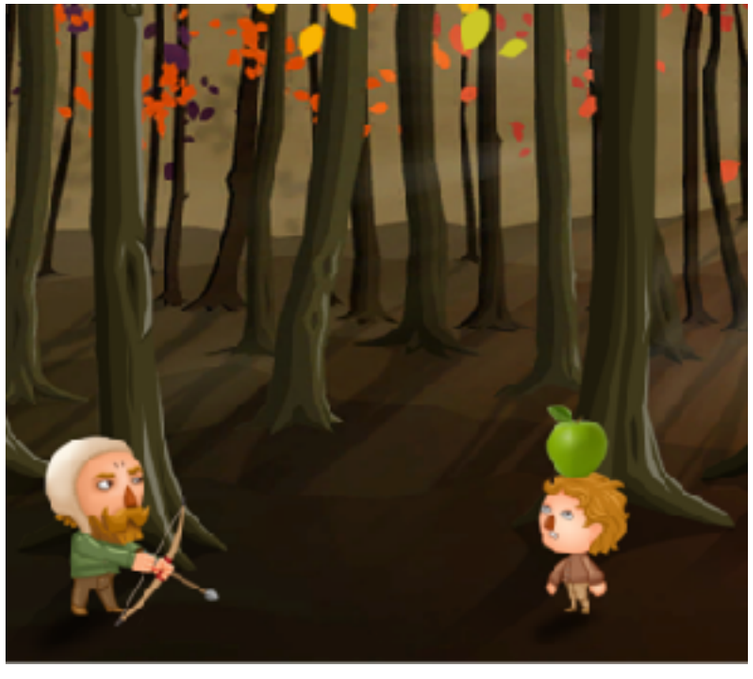
QU’EST-CE QUE LE CANVAS ?

Description :

- **Élément HTML**
- **Dessin**
- **2D/3D (WebGL)**
- **Formes géométriques**
- **Écriture**
- **Manipulation d'images**

Applications :

- **API graphiques (ex : ChartJs)**
- **Jeux vidéos (ex: MasterArcher)**



CANVAS - MISE EN PLACE

- On crée un élément de type canvas en lui donnant un id, une largeur et une hauteur (ces deux derniers sont obligatoires pour que le canvas s'affiche.

- On récupère le canvas en JS (on préfèrera ne pas utiliser jQuery qui ne renvoie pas véritablement un objet du DOM mais une surcouche qui gênera pour la récupération du contexte).

- La fonction getContext nous renvoie un objet contenant toutes les méthodes d'interaction avec le canvas. Le paramètre passé à cette fonction renseigne sur le mode d'utilisation du canvas (2D, 3D)

```
<canvas id="cv" width="500" height="500"></canvas>
<script>
   var canvas = document.getElementById("cv");
   var context = canvas.getContext("2d");
</script>
```

CANVAS - LIGNES ET CHEMINS

- **moveTo (re-)définit la position du curseur virtuel. C'est à partir de ce curseur que se tracent les lignes dessinées via lineTo.**

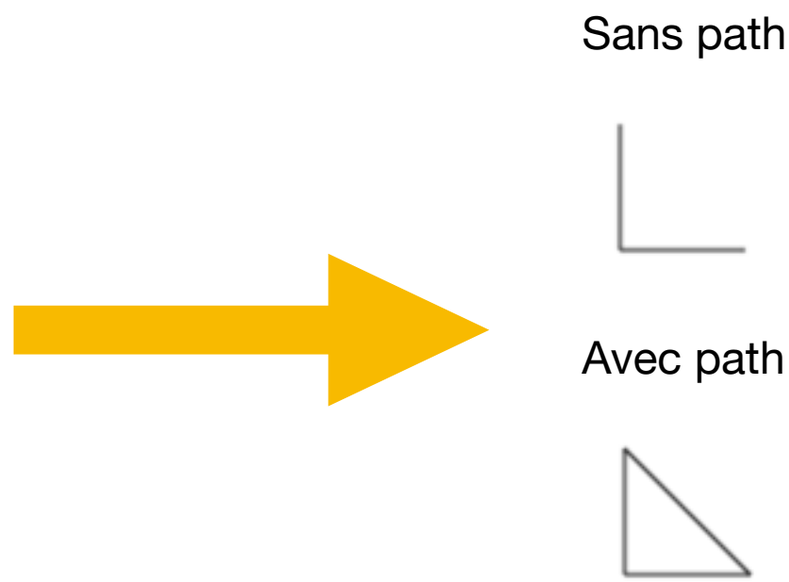
- **lineTo permet de tracer une ligne du curseur virtuel au point dont les coordonnées sont passées en paramètre.**

- **La fonction lineTo trace virtuellement une ligne. Pour afficher cette ligne, il faut l'indiquer via une fonction comme stroke (dessin des contours) ou fill (remplissage) ou les deux.**

- **Il est possible de définir ce que l'on appelle un chemin (path). L'intérêt de tracer un chemin est la possibilité, lors de sa fermeture, de relier le point final au point de départ via une ligne.**

- **L'exemple suivant, sans chemin, tracerait un angle droit (un L). Mais ici, en délimitant un chemin (avec beginPath et closePath), on obtient un triangle.**

```
<canvas id="cv" width="500" height="500"></canvas>
<script>
    var canvas = document.getElementById("cv");
    var context = canvas.getContext("2d");
        context.beginPath();
        context.moveTo(10, 10);
        context.lineTo(10, 60);
        context.lineTo(60, 60);
        context.closePath();
        context.stroke();
</script>
```



CANVAS - RECTANGLES

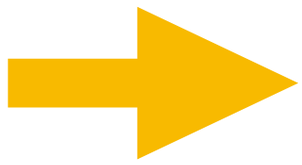
- **rect dessine un rectangle virtuel. Pour l'afficher, utiliser stroke ou fill.**
- **clearRect efface une portion du canvas sous la forme d'un rectangle.**
- **fillRect ou strokeRect font sensiblement la même chose que rect, à ceci prêt que l'affichage est instantané et le rectangle est soit rempli (fillRect), soit contourné (strokeRect).**

```
<canvas id="cv" width="500" height="500"></canvas>
<script>
    var canvas = document.getElementById("cv");
    var context = canvas.getContext("2d");
        context.rect(10, 10, 100, 100);
        context.fill();
        context.fillRect(50, 50, 200, 200);
        context.clearRect(50, 50, 60, 60);
</script>
```

CANVAS - TEXTES

- L'attribut font permet de définir la police utilisée pour le texte à dessiner
- strokeText et fillText servent à afficher du texte
- measureText permet de récupérer la largeur en pixel d'un texte (très pratique pour des calculs de position)

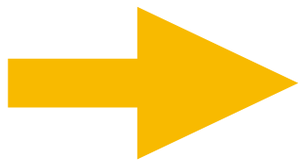
```
<canvas id="cv" width="500" height="500"></canvas>
<script>
    var canvas = document.getElementById("cv");
    var context = canvas.getContext("2d");
        context.font = "30px Verdana";
        context.strokeText("Canvas in da place", 10, 30);
    var textWidth = context.measureText("Canvas in da place").width;
        context.strokeText("Width : " + textWidth, 10, 70);
</script>
```



CANVAS - STYLES

- **Et les couleurs dans tout ça ?**
- **Pour redéfinir la couleur des contours, on utilise la propriété strokeStyle.**
- **Pour redéfinir la couleur de remplissage, on utilise la propriété fillStyle.**
- **Il existe de nombreuses manières de styliser vos tracés (ombres, dégradés, …)**

```
<canvas id="cv" width="500" height="500"></canvas>
<script>
    var canvas = document.getElementById("cv");
    var context = canvas.getContext("2d");
        context.font = "30px Verdana";
        context.strokeStyle = "#FFAABB";
        context.strokeText("Canvas in da place", 10, 30);
        context.strokeStyle = "#00FF00";
        context.fillStyle = "#0000FF";
    var textWidth = context.measureText("Canvas in da place").width;
        context.fillText("Width : " + textWidth, 10, 70);
        context.strokeText("Width : " + textWidth, 10, 70);
</script>
```



CANVAS - IMAGES

- Deux manières de manipuler une image :
  - Récupérer une image dans le dom (balise img)
  - Créer une instance de la classe Image et lui fournir une source (attribut src). Un événement onload sera enclenché dés que l'image sera chargé.

- Afficher l'image sur le canvas avec la fonction drawImage !

- Plusieurs façons d'utiliser cette fonction. Voir la slide suivante pour les variantes

```
<canvas id="cv" width="500" height="500"></canvas>
<script>
    var cs = document.getElementById("cv");
    var ctx = cs.getContext("2d");

    var drawPikachu = function () {
        ctx.drawImage(pikachu, 0, 0);
    };

    var pikachu = new Image();
        pikachu.src = "pikachu.png";
        pikachu.onload = drawPikachu;
</script>
```

CANVAS - SPRITING

- **Variante 1 : drawImage(image, x, y)**
- **Variante 2 : drawImage(image, x, y, width, height)**
- **Variante 3 : drawImage(image, sx, sy, swidth, sheight, x, y, width, height)**

```
ctx.drawImage(pikachu, 10, 20);
```

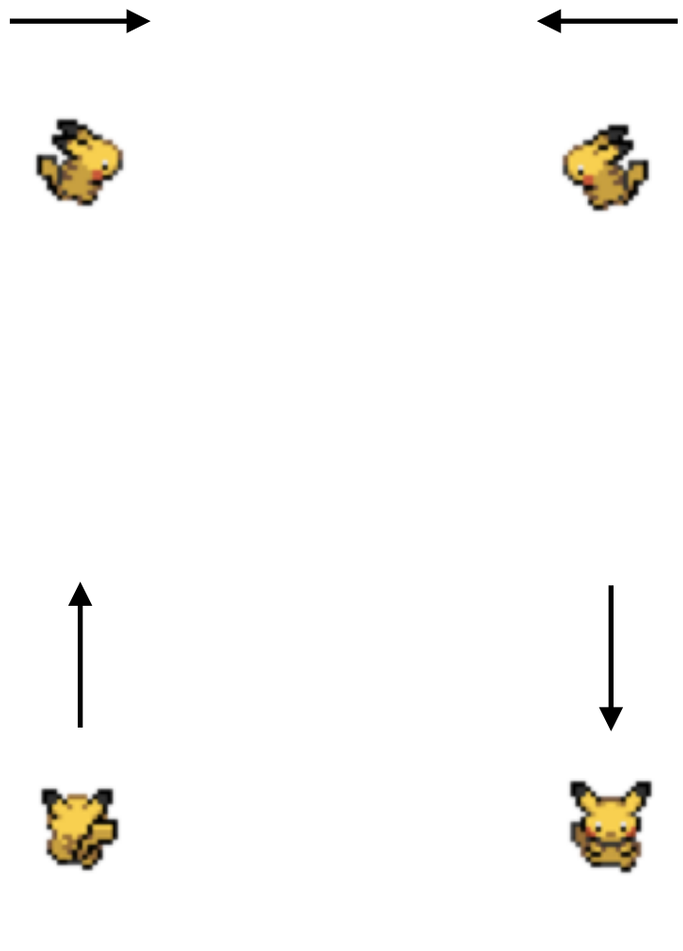
```
ctx.drawImage(pikachu, 0, 0, 75, 150);
```

```
ctx.drawImage(pikachu, 0, 0, 64, 64, 0, 0, 64, 64);
```

CANVAS - ANIMATION

```
<canvas id="cv" width="500" height="500"></canvas>
<script>
    var cs = document.getElementById("cv");
    var ctx = cs.getContext("2d");

    var direction = {
        "ArrowRight": 128,
        "ArrowLeft": 64,
        "ArrowUp": 192,
        "ArrowDown": 0
    };

    var sy = direction["ArrowDown"];

    var drawPikachu = function () {
        ctx.drawImage(pikachu, 0, sy, 64, 64, 0, 0, 64, 64);
    };

    var pikachu = new Image();
        pikachu.src = "pikachu.png";
        pikachu.onload = drawPikachu;

    document.onkeydown = function (e) {
        if (direction[e.key] === undefined) {
            return;
        }
        sy = direction[e.key];
        ctx.clearRect(0, 0, 500, 500);
        drawPikachu();
    };
</script>
```

- **Spriting + gestion des events = animation !**



CANVAS - ET D'AUTRES…

- **Transformations matricielles : changements d'échelle, rotations, translations, …**
- **Manipulation de pixels**
- **Sauvegarde et chargement de contextes**
- **Manipulation de formes : arcs, courbes de Bézier, …**
- **Références :**
  - **https://www.w3schools.com/tags/ref_canvas.asp**
  - **https://developer.mozilla.org/fr/docs/Tutoriel_canvas**