Exemples d’applications de l’élément <canvas>

1. Exemple 1 : Tracé d’un simple carré

Dans ce premier exemple nous allons nous contenter d’un tracé basique d’un carré. Ce sera la base future de la création d’une grille de TicTacToe.

Section HTML <head>

Cette section ne présente pas de vraies spécificités. Vous y trouverez une simple balise `meta` et le titre du script.

```
<!-- Début en-tête script HTML -->
<head>

    <!-- Balise meta -->
    <meta HTTP-equiv="Content-Type" content="text/html; charset=utf-8" />

    <!-- Titre du script HTML -->
    <title>MORPION_01/title>

</head>
```

Section HTML <body>

Le code de la section `<body>` est intégralement reproduit ci-après :

```
<!-- Mise en place de l’élément HTML canvas -->
<!-- NB : - Largeur du canvas : 1000 -->
<!--      - Hauteur du canvas : 800 -->
<canvas id="grilleMorpion" width="1000" height="800">
    Attention votre navigateur ne supporte pas l’élément Canvas
</canvas>

<!-- Code JavaScript associé au canvas -->
<script type="text/javascript">

    /* Définition de l’emplacement du canvas */
    var emplacementCanvas = document.getElementById("grilleMorpion");

    /* Définition du contexte du canvas (2D) */
    var contexteCanvas = emplacementCanvas.getContext("2d");

    /*
    Tracé dans le canvas
    */

    /* Définition des propriétés des traits */
    contexteCanvas.strokeStyle = "black";
    contexteCanvas.lineWidth = 1;
```

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

```
    /* Début du parcours en ligne brisée */
    contexteCanvas.beginPath();

    /* Point de départ du tracé (x, y) */
    contexteCanvas.moveTo(10, 10);

    /* Tracé du trait haut horizontal */
    contexteCanvas.lineTo(110, 10);

    /* Tracé du trait droit vertical */
    contexteCanvas.lineTo(110, 110);

    /* Tracé du trait bas horizontal */
    contexteCanvas.lineTo(10, 110);

    /* Tracé du trait gauche vertical */
    contexteCanvas.lineTo(10, 10);

    /* Fin de parcours en ligne brisée (facultatif) */
    contexteCanvas.closePath();

    /* Tracé effectif */
    contexteCanvas.stroke();

</script>
```

Étudions maintenant les différentes séquences de ce script (le niveau de détail sera moindre dans les exemples suivants).

Le code de la section `<body>` est intégralement reproduit ci-après :

Le script débute par la mise en place du canvas :

```
<!-- Mise en en place de l'élément HTML canvas -->
<!-- NB : - Largeur du canvas : 1000 -->
<!--      - Hauteur du canvas : 800 -->
<canvas id="grilleMorpion" width="1000" height="800">
    Attention votre navigateur ne supporte pas l'élément Canvas
</canvas>
```

L'élément `<canvas>` est défini par un identifiant (`id="grilleMorpion"`) et est dimensionné en largeur (`width="1000"`) et en hauteur (`height="800"`). Les valeurs sont exprimées en pixels.

La mention texte (`"Attention votre navigateur ..."`) présente entre la balise `<canvas>` et la balise `</canvas>` sera affichée dans le cas où le navigateur ne supporte pas l'élément `<canvas>`.

La séquence suivante :

```
/* Définition de l'emplacement du canvas */
var emplacementCanvas = document.getElementById("grilleMorpion");
```


© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

définit l'emplacement occupé par le <canvas>, c'est-à-dire un rectangle de 1000 pixels de largeur sur 800 pixels de hauteur.

Le contexte du <canvas> est ensuite annoncé :

```
/* Définition du contexte du canvas (2D) */
var contexteCanvas = emplacementCanvas.getContext("2d");
```

Rappelons qu'un contexte 2D fournit un jeu d'objets, de méthodes et de propriétés permettant la manipulation des dessins en 2D.

Nous pouvons maintenant passer à l'essentiel, le tracé d'un simple rectangle de couleur noire (sans remplissage). Ce carré sera ensuite répliqué de multiples fois dans le cadre de notre deuxième exemple pour constituer une grille de TicTacToe.

Avant d'envisager le tracé d'un trait de pourtour pour notre carré, il est indispensable de choisir une couleur (`black`) et une épaisseur (`1`) pour ce trait :

```
/* Définition des propriétés des traits */
contexteCanvas.strokeStyle = "black";
contexteCanvas.lineWidth = 1;
```

Il faut ensuite annoncer le début d'un tracé de dessin sur la base d'une ligne brisée :

```
/* Début du parcours en ligne brisée */
contexteCanvas.beginPath();
```

La suite demande un peu de patience, il s'agit de l'annonce des coordonnées des quatre côtés de notre carré :

```
/* Point de départ du tracé (x, y) */
contexteCanvas.moveTo(10, 10);

/* Tracé du trait haut horizontal */
contexteCanvas.lineTo(110, 10);

/* Tracé du trait droit vertical */
contexteCanvas.lineTo(110, 110);

/* Tracé du trait bas horizontal */
contexteCanvas.lineTo(10, 110);

/* Tracé du trait gauche vertical */
contexteCanvas.lineTo(10, 10);
```

À partir d'un point de départ (10, 10), les quatre côtés sont tracés.

La fermeture du tracé n'est pas obligatoire :

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

```
/* Fin de parcours en ligne brisée (facultatif) */
contexteCanvas.closePath();
```

Le tracé effectif du carré est déclenché par :

```
/* Tracé effectif */
contexteCanvas.stroke();
```

Exécution du script

L'exécution du script `MORPION_01.htm` donne l'affichage suivant :

2. Exemple 2 : Tracé d'une grille de TicTacToe

Il reste maintenant à reproduire le traitement vu au travers du premier exemple pour obtenir une véritable de grille de TicTacToe.

Section HTML <body>

Débutons l'explication cette fois-ci par l'étude de la section `<body>`.

Après la définition du `<canvas>` (déjà étudiée au travers de l'exemple précédent), le tracé de la grille (5 lignes * 5 colonnes) s'obtient comme suit :

```
<!-- Code JavaScript associé au canvas -->
<script type='text/javascript'>

    /* Définition de l'emplacement du canvas */
    var emplacementCanvas = document.getElementById("grilleMorpion");

    /* Définition du contexte du canvas (2D) */
    var contexteCanvas = emplacementCanvas.getContext("2d");
```


© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

```
    /* Tracé de la grille */
    /* NB : La fonction tracerGrille possède 3 paramètres : */
    /*      - Paramètre 1 : xDepart */
    /*      - Paramètre 2 : yDepart */
    /*      - Paramètre 3 : longueurCote */
    for (ligne = 1; ligne <=5; ligne++)
    {
            /* Tracé des 5 carrés de la ligne n°ligne de la grille */
            for (colonne = 1; colonne <=5; colonne++)
            {
                   tracerGrille(colonne*100, ligne*100, 100);
            }
    }

</script>
```

Deux boucles `for` imbriquées sont mises en place pour assurer le tracé de la grille du TicTacToe, la plus externe gérant les cinq lignes de la matrice, la plus interne assurant le tracé des cinq cellules de la ligne en cours.

La fonction `tracerGrille` est appelée pour assurer le tracé effectif des cellules. Trois paramètres (exprimés en pixels) sont passés à cette fonction :

- paramètre n°1 (`colonne*100`) : coordonnée x de la cellule à tracer,
- paramètre n°2 (`ligne*100`) : coordonnée y de la cellule à tracer,
- paramètre n°3 (`100`) : longueur du côté.

Section HTML <head>

Cette section contient principalement le code de la fonction `tracerGrille` :

```
/* Fonction tracerGrille */
function tracerGrille(x, y, cote)
{

    /*
    Tracé dans le canvas
    */

    /* Définition des propriétés des traits */
    contexteCanvas.strokeStyle = "black";
    contexteCanvas.lineWidth = 1;

    /* Définition des propriétés de la grille */
    var xDepart = x;
    var yDepart = y;
    var longueurCote = cote;

    /* Début du parcours en ligne brisée */
    contexteCanvas.beginPath();
```

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

```
    /* Point de départ du tracé (x, y) */
    contexteCanvas.moveTo(xDepart, yDepart);

    /* Tracé du trait haut horizontal */
    contexteCanvas.lineTo(xDepart + longueurCote, yDepart);

    /* Tracé du trait droit vertical */
    contexteCanvas.lineTo(xDepart + longueurCote, yDepart +
    longueurCote);

    /* Tracé du trait bas horizontal */
    contexteCanvas.lineTo(xDepart, yDepart + longueurCote);

    /* Tracé du trait gauche vertical */
    contexteCanvas.lineTo(xDepart, yDepart);

    /* Fin de parcours en ligne brisée (facultatif) */
    contexteCanvas.closePath();

    /* Tracé effectif */
    contexteCanvas.stroke();

}
```

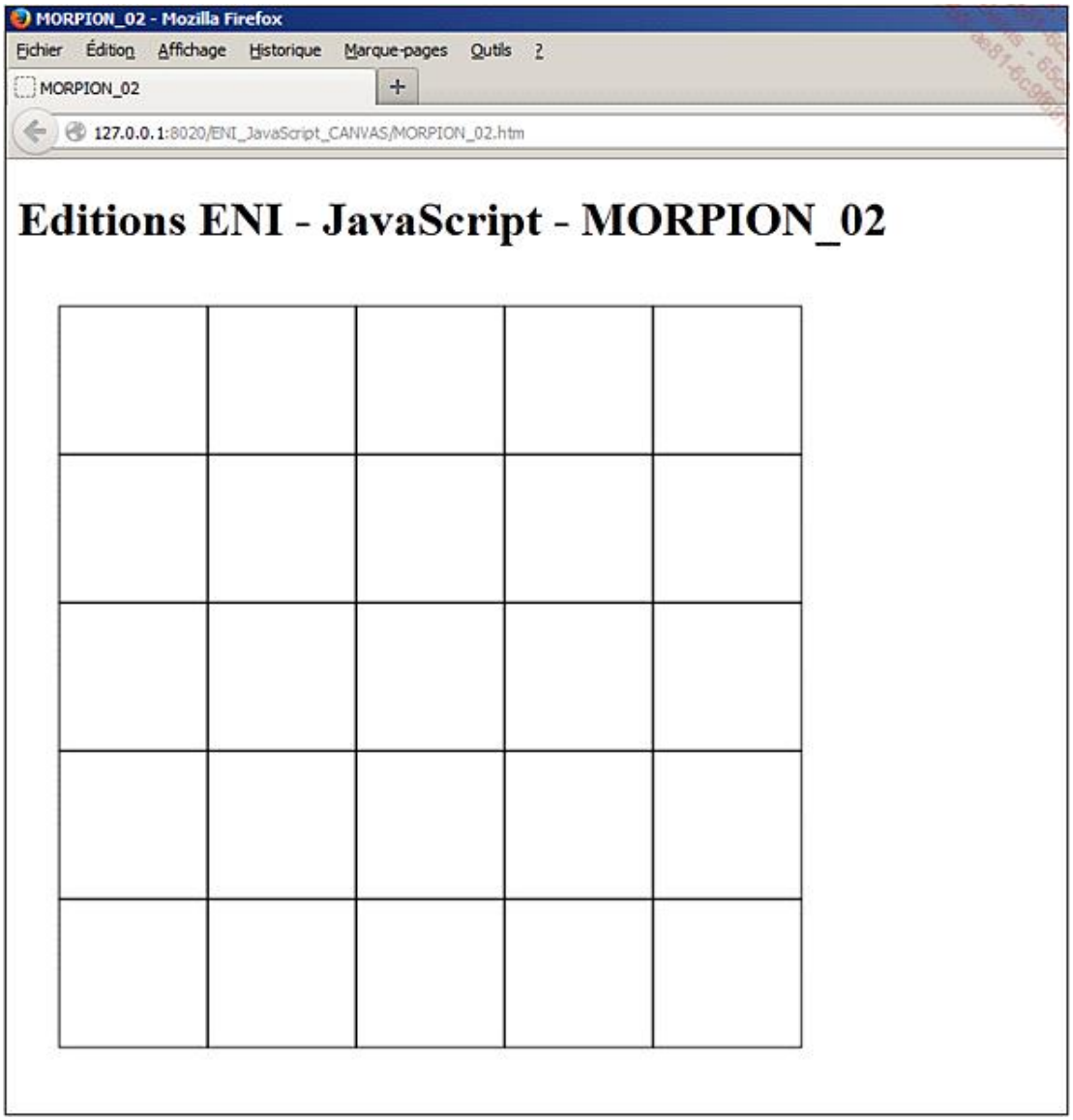
Le code est assez similaire à celui vu dans le cadre de l'exemple 1.

Les trois variables `xDepart`, `yDepart` et `longueurCote` sont déclarées et recoivent les valeurs passées en paramètres par le script appelant (cf. section `<body>`).

Exécution du script

À l'exécution, le script `MORPION_02.htm` donne ceci :

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare



3. Exemple 3 : Positionnement de deux marques dans la grille du TicTacToe

L'objectif est ici de vous donner les bases techniques pour que vous puissiez positionner des marques dans la grille du TicTacToe.

Section HTML <body>

Il a été choisi ici de placer deux marques sur la grille du TicTacToe :

- un carré vert en ligne 1, colonne 2 pour le joueur n°1 ;
- un cercle rouge en ligne 2, colonne 2 pour le joueur n°2.

Le code placé à la fin de la section <body> est bien sûr minimaliste. Il ne s'agit pas pour l'instant du jeu définitif qui permettrait aux deux joueurs d'agir successivement et de placer leurs marques (pions) sur la grille avec l'objectif d'obtenir des alignements de quatre marques par exemple.

Le code est le suivant :

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

```
/* Tracé d'un carré de couleur verte en ligne 1 & colonne 2 */
var x, y, cote;
contexteCanvas.fillStyle = "green";
ligne = 1;
colonne = 2 ;
x = (colonne * 100) + 20;
y = (ligne * 100) + 20;
largeur = 60;
hauteur = 60;
// alert("x = " + x);
// alert("y = " + y);
// alert("cote = " + cote);
contexteCanvas.fillRect(x, y, cote,cote);

/* Tracé d'un cercle de couleur rouge en ligne 2 & colonne 2 */
var xCentreCercle, yCentreCercle, rayonCercle;
contexteCanvas.fillStyle = "red";
ligne = 2;
colonne = 2 ;
xCentreCercle = (colonne * 100) + 50;;
yCentreCercle = (ligne * 100) + 50;
rayonCercle = 30;
// alert("xCentreCercle = " + xCentreCercle);
// alert("yCentreCercle = " + yCentreCercle);
// alert("rayonCercle = " + rayonCercle);
contexteCanvas.beginPath();
contexteCanvas.arc(xCentreCercle, yCentreCercle, rayonCercle, 0, 2
* Math.PI, false);
contexteCanvas.fill();
```

Étudions ce code pas à pas.

Pour le tracé du carré vert (correspond par exemple à la marque du joueur n°1), nous trouvons tout d'abord la déclaration de trois variables locales :

- `x` : position horizontale (en pixels) dans le `<canvas>`,
- `y` : position verticale (en pixels) dans le `<canvas>`,
- `cote` : longueur du côté en pixels du carré.

Il ne vous aura pas non plus échappé que les valeurs de `x` et de `y` ont été déterminées (pour faciliter le positionnement des marques)) à partir de variables `ligne` et `colonne` correspondant à la position souhaitée de la marque dans la grille.

La méthode `fillRect` assure le tracé du carré :

```
contexteCanvas.fillRect(x, y, cote,cote);
```

dans la couleur définie par :

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

```
contexteCanvas.fillStyle = "green";
```

Pour le tracé du cercle rouge (correspond par exemple à la marque du joueur n°2), nous trouvons aussi la déclaration de trois variables locales :

- `xCentreCercle` : position horizontale (en pixels) du centre du cercle dans le `<canvas>`,
- `yCentreCercle` : position verticale (en pixels) du centre du cercle dans le `<canvas>`,
- `rayonCercle` : rayon du cercle (en pixels).

Le tracé du cercle débute par la méthode déjà rencontrée dans le cadre du tracé de la grille du TicTacToe :

```
contexteCanvas.beginPath();
```

Le tracé du cercle se fait par l'intermédiaire de la méthode `arc`, comme suit :

```
contexteCanvas.arc(xCentreCercle, yCentreCercle, rayonCercle, 0, 2 * Math.PI, false);
```

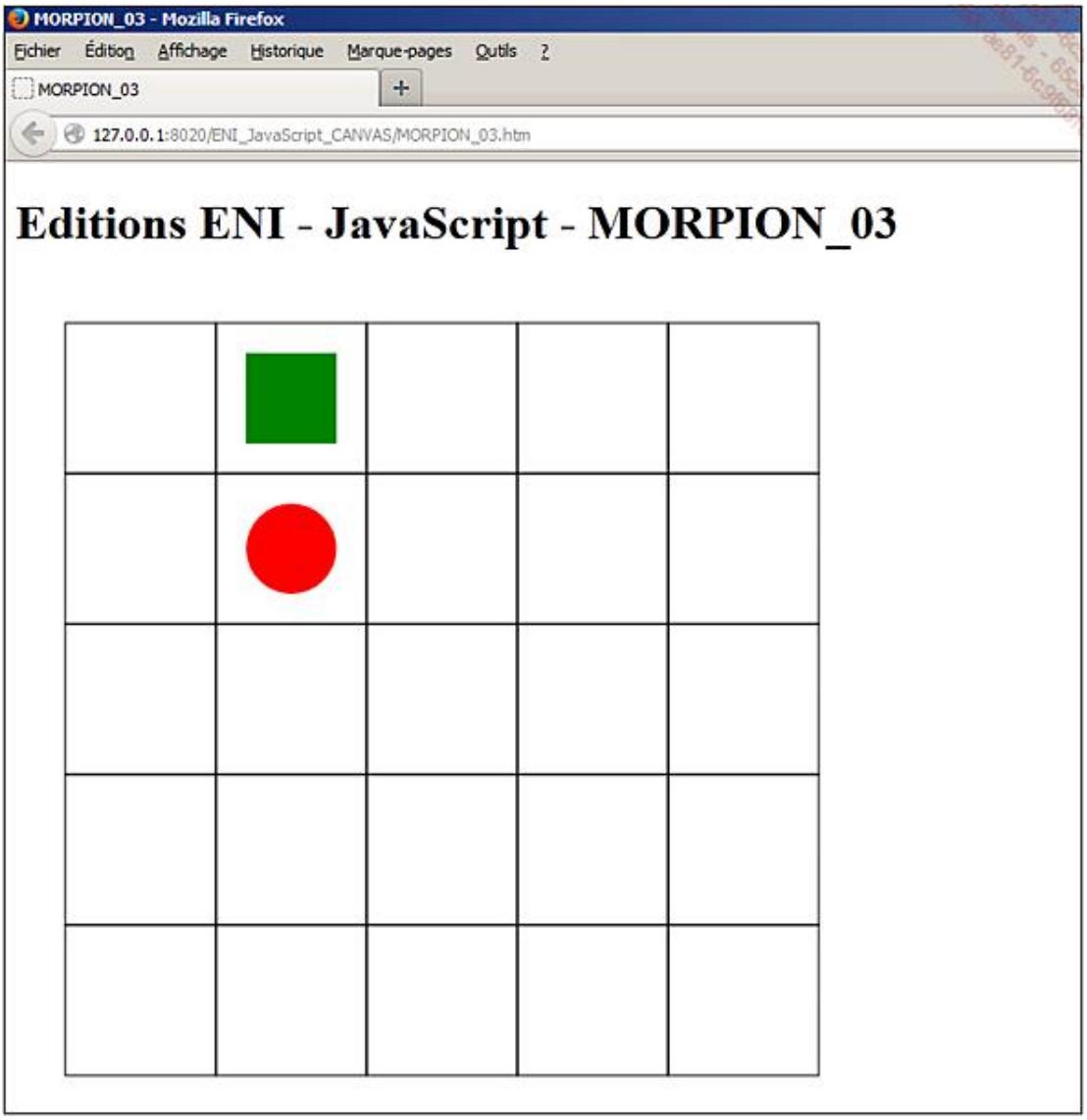
Le remplissage du cercle (en rouge) est effectué comme suit :

```
contexteCanvas.fillStyle = "red";
contexteCanvas.fill();
```

Exécution du script

L'exécution du script `MORPION_03.htm` donne l'affichage suivant :

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare



4. Améliorations possibles sur le jeu du TicTacToe

Le jeu du TicTacToe n'a pas été programmé entièrement dans le cadre de ce chapitre. Il mériterait (pourquoi pas de votre part) de nombreuses améliorations pour en faire une application aboutie, notamment :

- la possibilité pour les joueurs de sélectionner les cellules validées par l'intermédiaire de la souris,
- un algorithme permettant l'octroi de points, par exemple quand quatre marques identiques sont alignées verticalement, horizontalement ou en diagonale,
- un système de comptage des points,
- la mémorisation des scores,
- l'interruption et le redémarrage d'une partie,
- ...

Ces optimisations sont aisément réalisables et seraient l'occasion de revoir des techniques étudiées dans les précédents chapitres de ce livre (algorithmique, gestion des événements, stockage local ou distant en base de données...).

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare

Vous l'avez sans doute perçu au travers des trois exemples étudiés, nous ne sommes pas allés très loin dans l'étude des nombreuses possibilités de <canvas>.

Nous vous recommandons vivement de consulter les multiples exemples disponibles sur Internet. Vous apprendrez au travers de ceux-ci à utiliser les fonctionnalités avancées de <canvas>, en particulier pour gérer :

- les tracés et les chemins, les lignes et les styles associés,
- les formes (carrés, rectangles, arcs, cercles, courbes (y compris de Bézier ou quadratiques)),
- les images,
- les textes (polices, couleurs, alignements...),
- les effets (ombrage, dégradé, transparence, répétition...),

et vous trouverez des critères de choix d'un framework (librairie) pour vous faciliter le travail lors de réalisations plus ambitieuses.

© Editions ENI – Tous droits réservés – Copie personnelle de Denis Gaudare