Développement Hybride

Canvas, SVG, Drag & Drop, WebGl

Martinez Nicolas, Caltot Stephan, Guillon Julien

TP Développement Hybride

Canvas, SVG, Drag & Drop et WebGl

Ce TP se décompose en plusieurs partie correspondant aux différents éléments présentés durant le cours. Le but est de prendre en main ces éléments avec des exemples présentant leurs différentes fonctionnalités.
À la fin de ce TP on aura donc une application web / mobile présentant les exemples.

Les exemples fournis pourront être déployés sur mobile avec cordova, pour cela il faut donc un environnement fonctionnel que ce soit avec une installation locale ou avec docker (lien vers le cours précédent)

On souhaite obtenir à la fin du TP une application simple mettant en pratique les différents outils présentés. Cette application sera composée de pages accessibles avec une NavBar, chaque page correspondant à une partie ci-dessous.

Création de l'application

Exécuter les commandes suivantes pour créer le projet de base:

```
git clone jdjdjjdjdj.git
cordova create App
```

Il vous suffit de remplacer le dossier “www” du projet cordova créé “App” par le dossier “www” que vous venez de récupérer.

Le projet contient les différentes dépendances pour les jquery, jquery-mobile et les plugins ainsi que le menu de navigation pour accéder aux différentes pages.

Les différentes dépendances nécessaire à l’application sont::

- http://code.jquery.com/mobile/1.4.3/jquery.mobile-1.4.3.min.css
- http://code.jquery.com/jquery-1.11.1.min.js
- http://code.jquery.com/mobile/1.4.3/jquery.mobile-1.4.3.min.js

- https://raw.githubusercontent.com/caleb531/jcanvas/master/jcanvas.min.js
- https://raw.githubusercontent.com/furf/jquery-ui-touch-punch/master/jquery.ui.touch-punch.min.js

- L’ensemble des dépendances javascript nécessaire pour svg disponible ici

Le projet contient les fichiers html complets ainsi que les fichiers js associes correspondant

aux noms suivants:

- canvas
- svg
- canvas-svg
- dragAndDrop
- webgl
- tictactoecanvas
- tictactoesvg

Le but est de compléter les fichiers javascript pour arriver à une application fonctionnelle en suivant les indications du tp et les commentaires dans les fichiers.

Important:

Pour adapter l'application pour cordova, il faut que les événements tactiles mobiles soit reconnus comme des événements souris, pour cela lier le script dans le <head> de la page:

<script src="jquery.ui.touch-punch.min.js"></script>

Canvas

La balise canvas va nous permettre de réaliser une espace de dessin sur lequel on va pouvoir créer des formes : cercle, rectangle et ligne droite, mais également afficher une image et effectuer des animations sur ces objets.

La première page doit donc comporter un élément canvas ainsi que des boutons radio permettant de sélectionner l'objet à créer.
La création se fera lors d'un clic droit sur le canvas.

Pour cela on utilisera le plugin de jQuery pour Canvas : jCanvas

1. La création de l'objet se fera avec une taille fixe et positionné à l'emplacement du clic souris.

   Utiliser pour cela la méthode suivante pour récupérer la position d'un clic:

```
function getCursorPosition(canvas, event) {
        var rect = canvas.getBoundingClientRect();
        var x = event.clientX - rect.left;
        var y = event.clientY - rect.top;
        console.log("x: " + x + " y: " + y);
                return [x, y];
}
```

   Ou pour la position d'un événement tactile:

```
function getTouchPos(canvas,, touchEvent) {
        var loc =
touchEvent.originalEvent.changedTouches[0];
        var rect =
canvas.getBoundingClientRect();
        var x = loc.clientX - rect.left;
        var y = loc.clientY - rect.top;
        return [x, y];
```

2. Mettre en place la possibilité de déplacer les objets graphiques créés
   Utiliser l'API layer de jCanvas, plus précisément l'option "draggable"
3. Mettre en place un agrandissement des formes lors du clic droit sur celles-ci
   Utiliser pour cela l'API Event de jCanvas, "mousedown".

SVG

La balise svg va nous permettre de réaliser une espace de dessin sur lequel on va pouvoir créer des formes : cercle, rectangle et ligne droite, mais également afficher une image et effectuer des animations sur ces objets.

Cette page doit donc comporter un élément svg ainsi que des boutons radio permettant de sélectionner l'objet à créer.
La création se fera lors d'un clic droit sur le svg.

Pour cela on utilisera le plugin de jQuery pour SVG : svg jQuery

1. Mettre en place la possibilité de déplacer les objets graphiques créés
   Utiliser la méthode jQuery draggable, en redéfinissant la méthode drag, par exemple pour les rectangles:

```
$('rect').draggable({
    drag: function(event, ui) {
        var position = ui.position;
        var x = $(this).attr('x').val;
        var y = $(this).attr('y').val;
        $(this).attr('x', position.left);
        $(this).attr('y', position.top);
    }
});
```

2. Mettre en place un agrandissement des formes lors du clic droit sur celles-ci:
   a. 1er moyen, modifier les attributs de l'objet cliqué passant par le DOM et jQuery

Pour récupérer l'élément cliqué en jQuery:

```
$("#svg").click(function(event) {
        var clicked = event.target;
}
```

b. 2eme moyen, mettre en place un script svg et l'associer lors de la création de la forme avec la méthode onclick:

```
svg.circle(300, 150, 100, {onclick: clickFunction(evt)', fill: 'red'});
```

Le script pour augmenter la taille d'un cercle par exemple:

```
svg.script('function circleClick(evt) {
\n' + ' var circle = evt.target;\n'
 + ' var currentRadius = circle.getAttribute("r");\n'
+ ' if (currentRadius == 100)\n'
+ ' circle.setAttribute("r", currentRadius * 2);\n'
+ ' else\n'
+ ' circle.setAttribute("r", currentRadius * 0.5);\n' + '}',
'text/ecmascript');
```

SVG / Canvas

Cette troisième page a pour but de montrer les différences entre SVG et Canvas.
Pour cela la page doit comporter côte à côte un élément canvas et un élément svg sur lesquels on crée un cercle.

1. Récupérer et afficher le DOM de chaque élément canvas ou svg dans la partie correspondante.
2. Mettre en place pour chaque partie la possibilité de déplacer les objets graphiques créés et mettre à jour dynamiquement l'affichage du DOM.

On peut noter que comme vu dans le cours, canvas ne produit pas de DOM contrairement à SVG.

Drag & Drop

Vous trouverez tous les exemples nécessaires sur le site de jquery aux adresses suivantes :

- Pour le plugin draggable : https://jqueryui.com/draggable/
- Pour le plugin droppable : https://jqueryui.com/droppable/
- Pour le plugin sortable : https://jqueryui.com/sortable/

WebGl

Cette page a pour but de mettre en place un environnement graphique 3D en utilisant webGl, pour cela on va utiliser la librairie threeJs

1. Mettre en place une scène permettant d'afficher le rendu. https://threejs.org/docs/index.html#Manual/Introduction/Creating_a_scene
2. Créer un cube ou bien une sphère (les deux) en utilisant l'option wireframe à true pour visualiser le cube en 3D.
3. Ajouter à cet objet une texture ou bien une couleur en utilisant "THREE.MeshPhongMaterial" pour appliquer la texture à l'objet créé.
4. Ajouter une lumière directionnel à la scène afin de visualiser les ombres sur l'objet.
5. Ajouter la possibilité de faire tourner l'objet en modifiant la fonction render créée lors de l'initialisation de la scène.

TicTacToe

Cette partie se présente sous la forme de deux pages du projet, ces deux pages correspondant chacune à la mise en place d'un jeu Tic Tac Toe en utilisant les éléments Canvas et SVG vu précédemment. Pour ces deux parties, on n'utilisera pas de plugins, il s'agit ici de mettre en place les deux éléments en utilisant les fonctions et méthodes de bases utilisables sur ces éléments.

Le but est ici de mettre en évidence les différences d'utilisation qu'il peut y avoir avec l'utilisation des plugins mais également vous permettre d'utiliser et de mettre en pratique ces éléments apparu avec HTML5.