#7 : DOM, Events, Canvas

1) Introduction

Dans cette partie du parcours de formation nous allons apprendre à utiliser le javascript pour animer les pages web, interagir avec l'utilisateur et faire des animations. Ainsi nous serons prêts pour réaliser en projet de véritables applications web ou des jeux vidéo par exemple.

2) Le DOM (Document Object Model)

- Une page HTML5 est en réalité organisée sous forme hiérarchique et tous les objets qui la composent sont reliés par des relations d'héritages (parents – enfants). Nous n'allons pas rentrer dans le détail du DOM mais uniquement utiliser les commandes de base pour interagir avec les objets de la page.

- Commencez par créer une page HTML standard nommée **dom.html** qui lance le programme javascript **dom.js** et dans laquelle vous ajoutez le contenu suivant dans la balise **body** :

```
<body>
	<p id="para_1">Ceci est un paragraphe avec un texte qui ne sert à rien</p>
</body>
```

- Enregistrez le programme javascript suivant sous le nom : **dom.js**

```
window.onload = function ()
{
	alert("Votre page va être modifiée après avoir appuyé sur OK");
	var monPara = window.document.getElementById("para_1");
	monPara.innerHTML = "Voilà le texte a bien été modifié";
}
```

- Lancez la page HTML dans Firefox et expliquons les nouveautés que vous venez de mettre en œuvre :

- A la première ligne du programme javascript : **window.onload = function () { … }**, nous disons au navigateur de lancer le contenu de la fonction pour répondre à l'évènement « la page est chargée ». **Cette méthode est très souvent utilisée pour lancer votre programme javascript automatiquement quand la page est chargée.**

- Ensuite, **window.document.getElementById("para_1")** recherche dans l'arborescence de la page, l'objet qui porte **l'id para_1** et renvoie cet objet. On crée une variable **monPara** pour récupérer cet objet.

- Cet objet **monPara** contient plusieurs attributs dont l'attribut **innerHTML** qui permet de **lire et d'écrire le contenu du paragraphe.**

- Modifiez votre programme js pour changer la couleur du paragraphe avec **monPara.style.color = "blue"**.

- **Exercice** : Modifiez le programme pour afficher dans l'alert « Votre texte était : + le contenu du paragraphe », avant de réaliser les modifications.

3) Interface et évènements

- Ajoutez à la fin de votre page HTML un bouton à l'aide de la balise **<button type="button" id="bouton_1">Click Me!</button>**. Modifiez la page pour qu'elle lance le programme **bouton.js** et enregistrez la sous le nom **bouton.html**.

- Puis enregistrez le programme **bouton.js** suivant :

```
window.onload = function () // Fonction lancée juste après le chargement
{
	var monPara = window.document.getElementById("para_1"); // Récup de l'objet paragraphe
	var monBouton = window.document.getElementById("bouton_1"); // Récup de l'objet bouton
	monBouton.onclick = function () // Fonction à lancer en cas de clic sur l'objet monBouton
	{
		monPara.innerHTML = "Voilà le texte a bien été modifié";
		monPara.style.color = "blue";
	}
}
```

- Essayez ce programme en lançant la page HTML bouton.html.

❑Vous constatez que nous avons défini une fonction à lancer en cas de clic sur le bouton avec **monBouton.onclick = function () {… }**. En informatique, une fonction qui répond à un **évènement** s'appelle un **callback**.

❑Il existe de nombreux évènements que l'on peut utiliser sur les objets des pages HTML :

onmouseover	Faire entrer le curseur sur l'élément
onmouseout	Faire sortir le curseur de l'élément
onmousedown	Appuyer (sans relâcher) sur le bouton gauche de la souris sur l'élément
onmouseup	Relâcher le bouton gauche de la souris sur l'élément
onmousemove	Faire déplacer le curseur sur l'élément
onkeydown	Appuyer (sans relâcher) sur une touche clavier sur l'élément
onkeyup	Relâcher une touche clavier sur l'élément
onkeypress	Frapper (appuyer puis relâcher) sur une touche clavier sur l'élément

❑**Exercice** : Créez 3 fichiers **div.html, div.css et div.js** pour créer une div bleue carrée de 100 pixels de coté qui devient rouge quand on clique dessus ! Appelez le professeur pour montrer votre exploit !

❑**Exercice** : Créez 3 fichiers **div2.html, div2.css et div2.js** qui vont vous permettre de créer une div bleue qui devient rouge quand le curseur la survole et redevient bleu quand le curseur quitte la div !

❑Voyons comment utiliser les évènements pour traiter les touches du clavier (fichier **clavier.html**) :

```
<!doctype html>
<html lang="fr">
<head>
	<meta charset="utf-8">
	<link rel="stylesheet" href="clavier.css" />
	<title>Utilisation du clavier</title>
	<script src="clavier.js"></script>
</head>
<body>
	<h1>Appuyer sur une flèche</h1>
	<div id="maDiv">
		Aucune touche
	</div>
</body>
</html>
```

❑Faisons un petit fichier CSS pour formater tout ça : **clavier.css**.

```
#maDiv
{
	background-color : red;
	width : 250px;
	height : 100px;
}
```

❑Et je programme javascript qui va gérer les évènements : **clavier.js**.

```
window.onload = function ()
{
	var maDiv = window.document.getElementById("maDiv");
	window.onkeydown = function(event)
	{
		if (event.keyCode == 39) maDiv.innerHTML = "flèche droite";
		else if (event.keyCode == 37) maDiv.innerHTML = "flèche gauche";
		else if (event.keyCode == 38) maDiv.innerHTML = "flèche haut";
		else if (event.keyCode == 40) maDiv.innerHTML = "flèche bas";
	}
}
```

❑Lancez cet exemple. Vous constatez que le callback de l'évènement **window.onkeydown** doit trouver quelle touche a été appuyée en comparant le code de la touche donné par **event.keyCode** avec les codes standards des touches du clavier afin de réaliser l'action adaptée.

❑**Exercice** : Modifiez cet exemple pour changer la couleur de la div quand on appuie sur **espace** et sa taille quand on appuie sur **t**. Montrez votre résultat !

4) Le Canvas

❑Le canvas est un objet HTML qui permet d'avoir une zone vide dans une page HTML dans laquelle il est très facile de dessiner, voici comment faire. Commençons par le fichier HTML **canvas.html** :

```
<!doctype html>
<html lang="fr">
<head>
      <meta charset="utf-8">
      <title>Utilisation du Canvas</title>
      <script src="canvas.js"></script>
</head>
<body>
      <h1>Le beau Canvas !</h1>
      <canvas id="mon_canvas" width="800" height="600" style="border:1px black solid"></canvas>
</body>
</html>
```

❑Et bien sûr le fichier javascript pour dessiner dedans et interagir avec : **canvas.js**

```
window.onload = function ()
{
      var canvas=document.getElementById('mon_canvas');     // Récup de l'objet canvas
      var context=canvas.getContext('2d');      // Récup du context graphique du canvas
      var a=0;
      var b=100;
      var c=100;
      var d=100;
      context.fillRect(a,b,c,d);    // utilisation des fonctions du contexte graphique
}
```

❑Lancez cet exemple ! Modifiez les variables a, b, c et d pour expliquer leur rôle dans le programme :

_ _

_ _

_ _

❑A quoi sert la fonction **fillrect**() du context graphique ?

_ _

_ _

❑Ajoutez juste avant **fillRect** la fonction : **context.fillStyle="rgb(125,255,0)"**. Testez le résultat en modifiant les valeurs entre parenthèses. A quoi sert **fillStyle** ?

_ _

_ _

❑**Exercice** : Ajoutez au programme précédent une boucle pour que le canvas affiche des rectangles décalés les uns des autres de 10 pixels sur la droite sans que le dernier rectangle soit coupé par le bord du canvas ! (la largeur du canvas s'obtient avec canvas.width).

❑**Exercice** : Modifiez votre programme pour que le dessin ne s'effectue qu'après avoir cliqué sur le canvas ! Montrez votre résultat !

❑Vous pouvez dessiner ce que vous voulez en combinant les primitives graphiques référencées ici : http://www.w3schools.com/tags/ref_canvas.asp