PROGRAMMATION WEB AVANCEE

TP4 suite

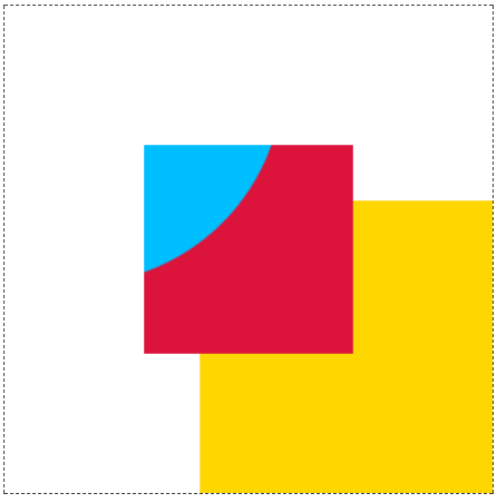
Canvas

Exercice 1

Créer le canvas ci-dessou :

Exercice 2

Créer le canvas ci-dessous :



Exercice 3

Écrire une page HTML 5 :
· contenant un **canvas**
Écrire un texte rempli de 30px de taille et la police « Arial » à l'aide de fillText() sur le Canvas
· charger cette image dans le **canvas**
· le canvas aura pour dimension 1306x720 pixels
· placer un cercle rouge semi-transparent de 50 pixels de rayon aux coordonnées 330 (horizontal) et 240 (vertical)
· éffacer la zone rectangulaire de 100x100 pixels placée aux coordonnées 670 (horizontal) et 100 (vertical)
Il faudra faire attention au temps pris par le chargement de l'image, et donc ne commencer le traitement qu'une fois que l'image aura été complètement chargée. Dans le cas contraire on aura un comportement erratique dépendant des conditions de courses des processus chargement/traitement.
L'appel du traitement peut se faire par un handler positionné sur la fin du chargement de l'image par sa méthode **addEventListener('load', *fonction_de_traitement*, false)**.

Exercice 4

Ajouter un titre au canvas avec un remplissage dégradé linéaire de couleur

A votre imagination : créez votre propre effet sonore et vidéo en utlisant les canvas

Exercice 5

1. Afficher un carré dans un *canvas*.
2. Permettre son déplacement avec les touches du clavier en veillant à ne pas dépasser les bords du *canvas*.
3. Ajouter un bouton pour lancer le déplacement aléatoire du carré, un autre bouton pour l'arrêter.
4. Comment faire disparaître la trace pour seulement déplacer un carré ?

Exercice 6

Une cible apparaît à une position aléatoire, le joueur a quelques instants pour cliquer dessus.

1. Définir l'aspect du jeu à l'aide d'un *canvas* html5.
2. Faire apparaître un carré en guise de cible, à une position aléatoire du canvas. La cible doit rester apparente deux secondes, puis disparaître. Une autre cible est alors générée aléatoirement.

La production de cibles doit débuter dès le chargement de la page.

3. Capturer les clics de l'utilisateur.

Tester si le clic a touché la cible. Si c'est le cas, effacer la cible et en produire une autre.

4. Lors d'un clic, afficher un message indiquant si la cible est touchée ou non. Afficher également le nombre de cibles détruites.
5. Ajouter un bouton pour faire une pause dans la partie et un autre pour démarrer une nouvelle partie.
6. (bonus) Faire varier taille, forme et couleur des cibles.

Exercice 7 : Editeur de dessin

On explore le fonctionnement d'un `canvas` (html5) : comme y dessiner et comment y capturer un clic de souris.

1. Créer le canvas en html en spécifiant ses dimensions.
2. Écrire une fonction initialisations() qui récupère le *contexte 2d* du canvas et associer cette fonction à l'événement onload du document.
3. Définir une fonction surclic(e) qui lance une alerte affichant les coordonnées du clic contenu dans e. Compléter la fonction initialisations() pour associer un clic sur le canvas à l'appel de la fonction surclic(e).
4. Remplacer le lancement de l'alerte par l'affichage d'un carré à l'endroit du clic. Régler la taille de ce carré.
5. Ajouter des boutons, chaque bouton permettant de choisir une couleur différente.
6. Ajouter deux boutons pour choisir la forme de la prochaine figure, soit un carré soit un disque.
7. Ajouter un bouton qui va lancer la composition aléatoire d'un dessin, par exemple en produisant une figure toutes les demi-secondes, figure dont la forme et la couleur sont choisies au hasard. Le dessin prend fin après 100 figures ou par un clic sur un dernier bouton.