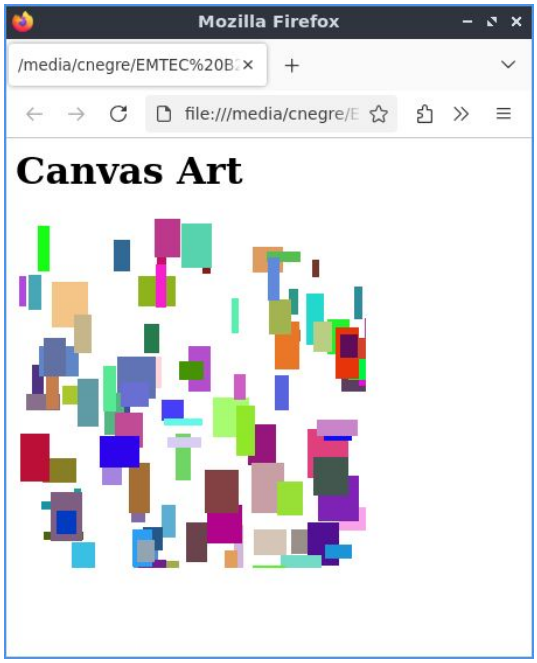
Université de Perpignan Année 2025
Licence Info 3 C. Negre

TD6 - Programmation Web

—

canvas, animation

Exercice 1 (prise en main : canvas-art)**.** Creez une page web avec un canvas de taille 500 par 500 pixels. Le canvas a pour identifiant `"canvas"`. L'objectif est de faire un script JavaScript qui positionne des rectangles aléatoirement dans le canvas comme ci-dessous:

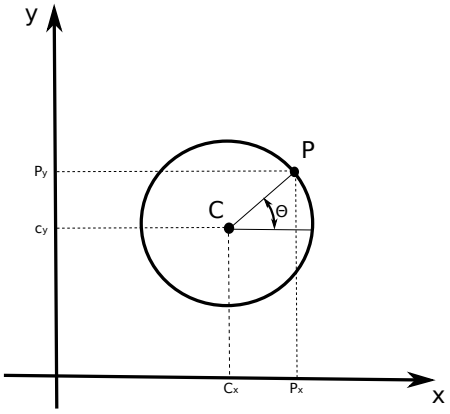


1. Faire une fonction `rand_color()` qui renvoie une chaîne de caractère de type "rgb(nr,ng,nb)" où nr, ng et nb sont des valeurs entières dans [0,255] de red,green et blue choisie aléatoirement.
2. Définir en variable globale n le nombre de rectangle à dessiner. Récupérez aussi l'élément canvas et le contexte 2D. Faire ensuite une boucle qui choisit aléatoirement pour chaque rectangle
   - largeur dans [5,30] et la hauteur dans [10,50].
   - la position du coin en haut à gauche du rectangle, aléatoirement dans le canvas,
   - une couleur.

   Dessinez ensuite le rectangle.

Exercice 2 (animation : horloge)**.** L'objectif est de faire un script JavaScript pour la page `horloge.html` afin de dessinez l'horloge et d'animer le mouvement des aiguilles. Les coordonnées du centre de l'horloge sont deux variables globales Cx et Cy. L'élément canvas et le contexte sont aussi deux variables globales.

1. Faire une fonction `dessine_aiguille(longueur,angle,thick)` qui a comme paramètre sa longueur en pixel, l'angle en radian avec l'horizontale où est dessinée l'aiguille et l'épaisseur. La fonction dessine ensuite l'aiguille: pour rappel si on considère un point P d'un cercle de centre C et de rayon r, situé à l'angle θ de l'axe des x:

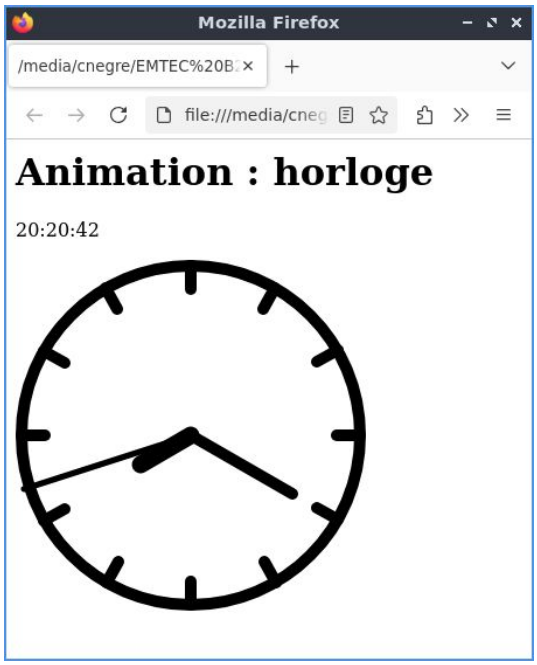


Les coordonnées de P sont données par

$$\begin{aligned} P_x &= C_x + r \times cos(\theta) \\ P_y &= C_y + r \times sin(\theta) \end{aligned}$$

Vous ferez attention que les coordonnées dans le canvas sont renversées (l'axe y est vers le bas) par rapport au schéma du dessus.

2. Faire une fonction `dessine_horloge()` sans paramètre qui dessine le tour de l'horloge et les traits pour les heures et les aiguilles comme ci-dessous.



Indications:

- Cercle : Notez que c'est un cercle complet dessiné avec un `stroke`, mais avec une épaisseur de 10 (`contexte.lineWidth = 10;`).
- Traits. Pour dessiner les 12 traits des heures, les coordonnées des deux extrémités des traits se calculent comme ce qui a été fait dans la question 1).
- Aiguille: On récupère les heures, minutes et secondes avec les méthodes `getMinutes`, `getSeconds` et `getHours` de l' objet `Date`. On dessine ensuite les aiguilles correspondantes de l'horloge avec la fonction `dessine_aiguille`.

3. Faire une fonction `init` qui est lancée lorsque le document est chargée. Cette fonction récupère le contexte et le canvas et le contexte et avec la fonction `setInterval` définit un appel à `dessine_horloge` toutes les secondes.

Exercice 3 (**Facultatif** : balle rebondissante dans un carré)**.** L'objectif est d'animer une balle qui rebondit à l'intérieur d'un carré. Faites une page web qui contient un canvas carré de taille 300 par 300. Le bord du carré est dessiné en noir.

L'animation consiste à redessiner à intervalle régulier le carré avec la balle qui a avancé en appelant une fonction. C'est la fonction `update` qui doit faire cela. Il y a quatre variables globales qui vont servir à animer la balle

- `posx,posy` les coordonnées du centre de la balle.
- `dx,dy` la direction, $\vec{d} = (dx, dy)$ peuvent prendre comme valeur (correspondant à 4 directions de déplacement) :
  - $(-1, 1)$ direction en bas à gauche.
  - $(1, 1)$ direction en bas à droite.
  - $(-1, -1)$ direction en haut à gauche.
  - $(1, -1)$ direction en haut à droite.

Vous devez coder la fonction `update` qui :

- Détermine la nouvelle position après un déplacement dans la direction $\vec{d} = (dx, dy)$:

```
newposx=posx+dx;
newposy=posx+dy;
```

- Ensuite détermine si la balle doit rebondir (changer de direction):
  - Si `newposx+5` (le bord droit de la balle) déborde à droite du cadre. On change alors `dx` en `-dx`.
  - Si `newposx-5` (le bord gauche de la balle) déborde à gauche du cadre. On change alors `dx` en `-dx`.
  - Si `newposy+5` (le bord bas de la balle) déborde en bas du cadre. On change alors `dy` en `-dy`.
  - Si `newposy-5` (le bord haut de la balle) déborde en haut du cadre. On change alors `dy` en `-dy`.

- Ensuite met à jour `posx` et `posy` avec

```
posx=posx+dx;
posy=posx+dy;
```

- Enfin on redessine le tout: on remplit (`fill`) en blanc tout le canvas, on dessine (`stroke`) en noir les bords du canvas et on dessine un disque (`fill`) en rouge pour la position du centre `posx`,`posy` et un rayon de 5.

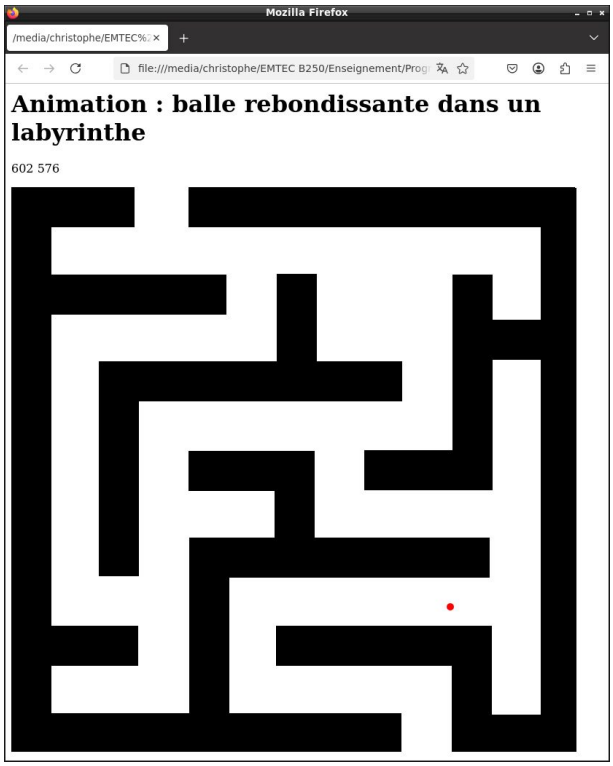
Faites une fonction `init` qui est appelée après le chargement du document. Cette fonction définit `posx` et `posy` aléatoirement dans le canvas (prendre des valeurs aléatoire dans [5,295]). Elle dessine le cadre et la balle. Ensuite elle appelle `setInterval` pour exécuter `update` toute les 10 milisecondes.

Exercice 4 (**Facultatif** : une balle qui rebondit dans un labyrinthe)**.** La page `labyrinthe.html` contient un canvas et un code JavaScript pour dessiner un labyrinthe. Le labyrinthe est dessiné dans la fonction `draw` grâce à des rectangles horizontaux et verticaux dont les coordonnées sont données dans les tableaux `rect_v` et `rect_h`.

Voici la position initiale de la balle et la direction initiale de la balle.

```
var posx=568;
var posy=744;
var dx=-1;
var dy=-1;
```

La fonction `draw` dessine aussi la balle.



L'objectif est de compléter le code javascript afin de faire avancer la balle et qui devra rebondir sur les parois du labyrinthes:

- Faire une fonction is `is_in_rect(px,py,rect)` qui étant donné un rectangle `rect` et deux coordonnées `px,py` elle renvoie 1 si le point px,py est dans rect et 0 sinon. `rec` est l'un des sous-tableaux des tableaux `rect_v` et `rect_h`.

- Faire une fonction `update`: qui met à jour `posx`,`posy`,`dx`,`dy`:

- Elle détermine `newposx` et `newposy`.
- Elle regarde si le bord droit de la balle `newposx+5` rentre dans un des rectangles. Si c'est le cas, elle changera alors `dx` en `-dx`.
- Même chose pour `newposx-5` elle changera alors `dx` en `-dx`.
- Même chose pour `newposy+5` elle changera alors `dy` en `-dy`.
- Même chose pour `newposy-5` elle change alors `dy` en `-dy`.

La fonction update appelle ensuite la fonction `draw` pour redessiner le labyrinthe et la balle.

- Faire une fonction `init` qui appelle `draw` puis `setInterval` pour dessiner l'horloge à interval régulier. La fonction `init` doit être exécutée juste après le chargement du document. La fonction `setInterval(interval,fct)` permet d'exécuter une fonction `fct` tous les `interval` milisecondes.