TP n°2 – Affichage d'images en Javascript

Rappel : pour simplifier la récupération des fichiers HTML sous moodle, nous changeons l'extension en .txt (penser à remettre la bonne extension .html).

1 – Création dynamique de balises

- Récupérer sur Moodle le fichier tp2_ex1.html et le tester
- Avant de passer à la suite, vous devez vérifier que vous comprenez toutes les instructions du script
- Modifier le script pour afficher une balise <img>, d'id ="01_carreau" et de nom de classe "carte"
  Le script doit créer la balise : `<img id="01_carreau" class="carte"/>`

2 – Chargement d'une image

- Récupérer sur Moodle les images correspondant aux 52 cartes d'un jeu standard :

À chaque carte du jeu correspond un fichier qui a un nom composé du numéro de la carte dans l'ordre (01 → as, 02 → 2, …, 11 → valet, 12 → dame, 13→ roi) et de la couleur.
Par exemple :
- le fichier « 01_carreau.gif » correspond à l'« as de carreau »
- le fichier « 12_coeur.gif » correspond à la « dame de cœur ».

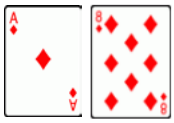
Important : enregistrer les images dans un répertoire de nom « cartes »

- Reprendre le script de la question précédente (conseil : travailler sur un nouveau fichier tp2_ex2.html) et compléter l'attribut `src` pour afficher l'« as de carreau ».

Le script doit créer la balise :

```
<img id="01_carreau" class="carte" src="cartes/01_carreau.gif"/>
```

- Écrire la fonction **`creerCarte`** qui crée une carte à partir de son nom passé en paramètre puis qui l'ajoute dans la

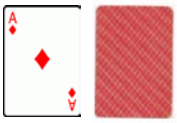


Ajouter l'instruction **`creerCarte("08_carreau");`**

- Écrire la fonction **`retournerVerso`** qui retourne sur le verso une carte dont le nom est passé en paramètre (pour le verso, utiliser le fichier **`verso.gif`**)

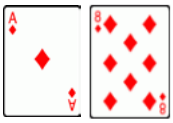
- Vous noterez que la balise **`<img>`** existe déjà et que la fonction ne fait que modifier l'attribut **`src`**
Il ne se passe rien si la carte n'existe pas (objet de valeur **`null`**)

Ajouter l'instruction **`retournerVerso("08_carreau");`**



- Écrire la fonction **`retournerRecto`** qui retourne côté recto une carte dont on connaît le nom
  Comme pour **`retournerVerso,`** la fonction ne fait que modifier l'attribut **`src`** et rien ne se passe si la carte n'existe pas

  Ajouter l'instruction **`retournerRecto("08_carreau");`**



Avant de passer au paragraphe suivant : travaillez sur un nouveau fichier tp2_ex3.html dans lequel vous ne conservez que la déclaration du tableau et les 3 fonctions **`creerCarte`**, **`retournerVerso`** et **`retournerRecto`**

3 – Affichage dynamique de plusieurs images

- Écrire la fonction **`choisirHasard`** qui choisit une carte au hasard dans un jeu de carte ; la fonction prend le nom du tableau en paramètre et retourne le nom de la carte (c'est-à-dire le contenu de la case choisie au hasard).
  L'instruction ci-dessous permet de choisir au hasard un indice du tableau de nom **`tab`** :

```
let i = Math.round(Math.random()*(tab.length-1));
```

  Tester la fonction avec l'instruction **`console.log(choisirHasard(jeu));`**

- Utiliser les fonctions **`choisirHasard`** et **`creerCarte`** pour créer et afficher une carte choisie au hasard

- Compléter le code pour afficher 5 cartes

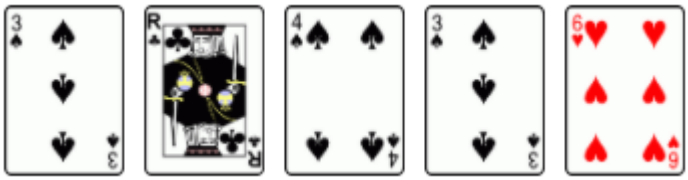


Figure 1 : les 5 cartes affichées dans

4 – Amélioration de l'affichage des images : éviter les doublons

- Écrire la fonction **`verifierDejaChoisie`** qui cherche si une carte, dont le nom est passé en paramètre, est déjà affichée.
  Par exemple, avec le tirage de la Figure 1 :

  `verifierDejaChoisie ("03_pique")` ➔ **`true`**
  `verifierDejaChoisie ("4_pique")` ➔ **`false`**

- Modifier le code pour afficher 5 cartes différentes.

5 – Pour aller plus loin : coder un Jeu de poker

- Écrire la fonction **`mainIsCouleur`** qui vérifie si la suite des cartes affichées est une couleur (toutes les cartes sont de la même couleur, quel que soit le nombre de cartes).

Remarque 1 : dans le nom de la carte, les caractères d'indice 3 et suivants, permettent de connaître la couleur. Pour récupérer une partie de la chaîne de caractères, regarder la méthode substring dans String (https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Objets_globaux/String).
Remarque 2 : proposition d'algo : récupérer la couleur de la première carte ; passer en revue les cartes restantes et regarder si elles sont de la même couleur que la première.
Remarque 3 : pour tester facilement la fonction, il est préférable de choisir soi-même les 5 cartes, par exemple :

```
creerCarte("01_carreau");
creerCarte("02_carreau");
creerCarte("03_carreau");
creerCarte("04_carreau");
creerCarte("05_carreau");
```

- Écrire la fonction **`mainIsQuinte`** qui vérifie si la suite de cartes est une quinte (les cartes forment une suite ordonnée sans trou).

Remarque 1 : proposition d'algo :
a) créer un tableau contenant les numéros des cartes affichées (les 2 premiers caractères du nom de la carte)
b) trier ce tableau avec la méthode sort (voir la doc en ligne de Array)
c) prendre la première valeur du tableau ; parcourir le reste du tableau et vérifier que le numéro de la carte est égal à 1+ le numéro de la carte précédente.

- Écrire la fonction **`mainIsQuinteFlush`** qui vérifie si la suite de cartes est à la fois une quinte et une couleur.

- Écrire les fonctions **`mainIsCarre`** (4 cartes identiques), **`mainIsBrelan`** (3 cartes identiques), **`mainIsPaire`** (2 cartes identiques), **`mainIsDoublePaire`**, **`mainIsFull`** (1 brelan + 1 paire).

Remarque : pour toute cette série de fonctions, on a besoin de compter le nombre de cartes par numéros (combien d'As, combien de 1, …, combien de rois). Il faut donc écrire une première fonction **`compterCartes`** *qui remplit un tableau de 13 compteurs avec le nombre de cartes par numéro (par exemple la case d'indice 0 du tableau contient le nombre d'As, la case d'indice 1 contient le nombre de 1 etc).*