Manuel du programmeur

– Le jeu du Taquin –

Réalisé par Ilona Baran

Table des matières

1.Informations générales

Auteurs : Ilona Baran

Date : Décembre 2021 - Février 2022

URL du site : https://hugoreveneau2.alwaysdata.net

Hébergement du site et de la base de données : alwaysdata

Le projet et ses objectifs: L'objectif de ce mini-projet est de réaliser un «taquin» :

- son interface de jeu avec son web service associé
- de documenter un drag and drop efficace pour pousser les rangées de pions
- de développer l'interface de construction d'un jeu de quinze imagettes avec son web service associé.

Langages utilisés :

- HTML5, CSS3
- JavaScript, AJAX
- PHP, PostgreSQL

Responsive ? : Non, il n'est pas responsive.

2. Manuel du programmeur et organisation des fichiers du projet

Le projet contient tous les fichiers nécessaires au fonctionnement de la page web et des échanges avec la base de données. Les fichiers sont répartis dans des dossiers correspondant à leurs types.

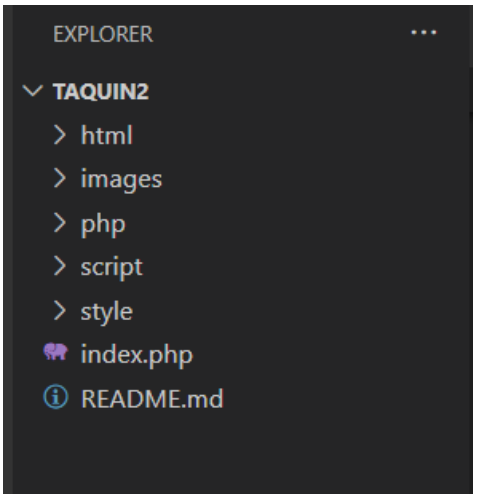

Figure 1 : Arborescence du Jeu du taquin

La page principale est ***index.php***, et sera celle lancée lors de l'appel de l'URL.

1. Dossier CSS

Il contient un unique fichier ***style.css***. C'est ce fichier qui implémente le style de la page. Il gère les couleurs, les animations des boutons mais également la mise en page des différentes divisions.

2. Dossier HTML

Il contient 3 fichiers qui présentent le plan du site, les mentions légales ainsi que les crédits. Ils ne servent pas directement au jeu, mais informe sur les créateurs du site, l'hébergement du site ou encore les sources utilisées.

3. Dossier IMG

Il contient les images utiles au projet. Utile pour le projet de Ziad.

4. Dossier JS

Il contient un unique fichier JavaScript ***taquin.js***, lancé dans ***index.php.***

- ***taquin.js*** a été codé par moi-même et Ziad. Les fonctions présentées seront celles que j'ai codées ou dont j'ai fortement contribuées. Il permet le lancement du jeu, le déplacement des imagettes.

```
fetch('php/taquin.php')
.then(response => response.json())
.then(result => {
    selectJeu.innerHTML = "";
    result.forEach((elem) => {
        selectJeu.innerHTML += elem;
    });

    var url = new URL(window.location.href);
    var jeu = url.searchParams.get("nJeu");

    if (Number.isInteger(parseInt(jeu))){
        recupNumeroJeu();
    }
})
```

Figure 2 : Affichage du jeu en fonction de la demande de l'utilisateur. Réalisé par Ilona

```
/**
 * Fonction qui permet d'afficher et de jouer le jeu choisi par le joueur
 */
var recupNumeroJeu = function() {
    var url = new URL(window.location.href);
    var jeu = url.searchParams.get("nJeu");
    var liste = listeAleatoire(1, 16);
    fetch("php/jeu.php?nJeu="+String(jeu))
    .then(r => r.json())
    .then(r => {
      myBD = r;
      let index = 0;
      for (let i = 0; i < 4; i++) {
        for (let k = 0; k < 4; k++) {
          var bulleImage = document.createElement('div');
          var img = document.createElement("img");
          bulleImage.setAttribute("class", "item");
          bulleImage.setAttribute("id", "item_" + i + "_" + k);
          img.setAttribute("draggable", "true");
          if (i != 3 || k != 3) {
            img.src = myBD[0]["img_"+NumToPos.get(liste[index])];
            tabImage.set(bulleImage, liste[index]);
          }
          else {
            img.src = "images/CarreBlanc.jpg";
            tabImage.set(bulleImage, "special");
          }
          bulleImage.appendChild(img);
          fondMap.appendChild(bulleImage);
          index++;
        }
      }
        items = document.querySelectorAll("#conteneur > div.item");
        checkScore(tabImage, items);
        nombre_de_coup = 0;
```

Figure 3 : Affichage du jeu en fonction de la demande de l'utilisateur

La fonction codée n'est pas montrée entièrement. La partie montrée a été réalisée par nous 2. Il y a un écouteur d'événement comportant une boucle qui a été réalisée entièrement par Ziad.

```
/**
 * Fonction qui permet de supprimer toutes les imagettes d'un jeu, et d'en ré-afficher un autre avec une autre disposition des imagettes
 */
btnAleatoire.onclick = function() {
    while (fondMap.firstChild) {
        fondMap.removeChild(fondMap.firstChild);
    }
    recupNumeroJeu();
};

/**
 * Fonction qui permet de générer une liste de nombre trié aléatoirement
 *
 * @param {int} min Minimum de la liste qu'on souhaite trier
 * @param {int} max Maximum de la liste qu'on souhaite trier
 * @returns Tableau de (max-min) valeur, trié aléatoirement
 */
function listeAleatoire(min, max){
    let tableauTrie = [];
    for (min; min < max; min ++){
        tableauTrie.push(min);
    }
    return randomize(tableauTrie);
}
```

Figure 4 : Fonctions permettant un nouvel affichage du même jeu de données mais triées dans un autre ordre aléatoire.

La fonction **randomize**, utilisée dans la fonction **listeAleatoire** provient de https://www.codegrepper.com/code-examples/javascript/javascript+randomly+shuffle+array.

5. Dossier PHP

Il contient tous les fichiers PHP utilisés par les fichiers ***index.php*** et ***taquin.js***.

- ***connexion.php*** est le fichier permettant la connexion à la base de données du projet hébergée sur les serveurs de AlwaysData.
- ***classement.php*** : contient la requête SQL permettant de sélectionner l'identifiant et le score des 10 meilleurs joueurs, et de l'afficher dans ***index.php.***
- ***jeu.php*** : contient la requête SQL qui va récupérer les adresses des images dans la base de données que le joueur aura sélectionné.
- ***taquin.php*** : contient la requête SQL qui va récupérer l'identifiant et le nom de 5 jeux parmis tout les jeux possibles

NB : Toutes les fonctions sont commentées directement dans le code, ainsi que les lignes que nous jugions utiles.