GAMECODEUR
École de jeux vidéo en ligne

Atelier Gamecodeur :

Initiation à JavaScript et HTML5

La programmation de jeux web à la portée de tous

Sommaire

Introduction

Imaginez programmer des jeux que vos joueurs n'ont pas besoin de télécharger. Un simple navigateur Internet suffit pour y jouer.

Imaginez pouvoir programmer et tester ces jeux avec un simple éditeur de code, sur n'importe quelle plateforme : Windows, Mac, Linux (et même sur Raspberry Pi).

Dans ce grand atelier vous allez apprendre JavaScript en moins d'une heure. Et ce langage va vous ouvrir les portes de la programmation de jeux pour le web grâce à la technologie HTML5.

Dans les années 90 et 2000, des centaines de millions de joueurs à travers le monde jouaient à des jeux Flash. Des jeux Flash existaient par centaines de milliers.

Oui Flash a dominé le marché des jeux Web. Mais Flash est mort, victime de sa vulnérabilité et de la décision d'Apple de le supprimer de ses plateformes.

Aujourd'hui, une nouvelle ère est en train de naître pour les jeux Web grâce au langage JavaScript et la technologie HTML5.

Avec moi, pas de librairie externe, pas de moteur usine à gaz : quelques lignes de code et vous voilà capable de programmer vos premiers jeux, aussi facilement que si vous le faisiez avec un framework comme Love2D.

Suivez-moi dans cette merveilleuse aventure !

JavaScript et HTML5… C'est quoi en fait ?

Vous êtes peut-être un grand débutant, donc vous ne connaissez pas ces termes.

Je vais tenter de vous les expliquer.

Commençons par **JavaScript**, c'est tout simplement **un langage de développement**.

> Note : Il n'a pas grande chose à voir avec Java. Il se nomme ainsi car Sun (les créateurs de Java) avaient prévu au départ qu'il soit un langage complémentaire à Java et sa syntaxe s'inspire pas mal de Java (mais Java a une syntaxe elle même inspirée du C et du C++). Lire https://stackoverflow.com/questions/2018731/why-is-javascript-called-javascript-since-it-has-nothing-to-do-with-java)

C'est un langage de "script" (comprenez : non compilé) comme l'est le langage Lua.

Il est utilisé pour rendre "dynamique" les sites internet, et aujourd'hui pour créer des applications, des jeux web, etc.

Le saviez-vous ? Visual Studio Code est programmé en JavaScript.

Mais HTML5 c'est quoi alors ?

HTML5 est tout simplement **la toute dernière version de HTML**, le langage de description des pages Web que tout le monde utilise.

Cette version a la particularité de proposer de nombreuses nouvelles fonctionnalités dont certaines spécifiquement pour le multimédia et donc pour les jeux vidéo.

Nous allons par exemple utiliser une balise "canvas" pour afficher notre jeu dans une page, et c'est une balise qui n'existe que depuis HTML5.

Rien de magique donc derrière ce terme :). Mais utilisez le, ça impressionne !

Votre environnement de développement

Pour travailler en JavaScript et HTML5, nul besoin d'installer quoi que ce soit. Tout est déjà préinstallé sur n'importe quel système d'exploitation moderne :

Il s'agit de votre navigateur Internet !

Note : Je vous conseille de travailler avec Chrome qui est de loin le meilleur pour HTML5. *(Le pire étant Safari, pas seulement pour ses performances mais pour les efforts qu'Apple déploie pour empêcher les développeurs de faire tourner des jeux sur navigateur, car cela concurrence leur store d'applications).*

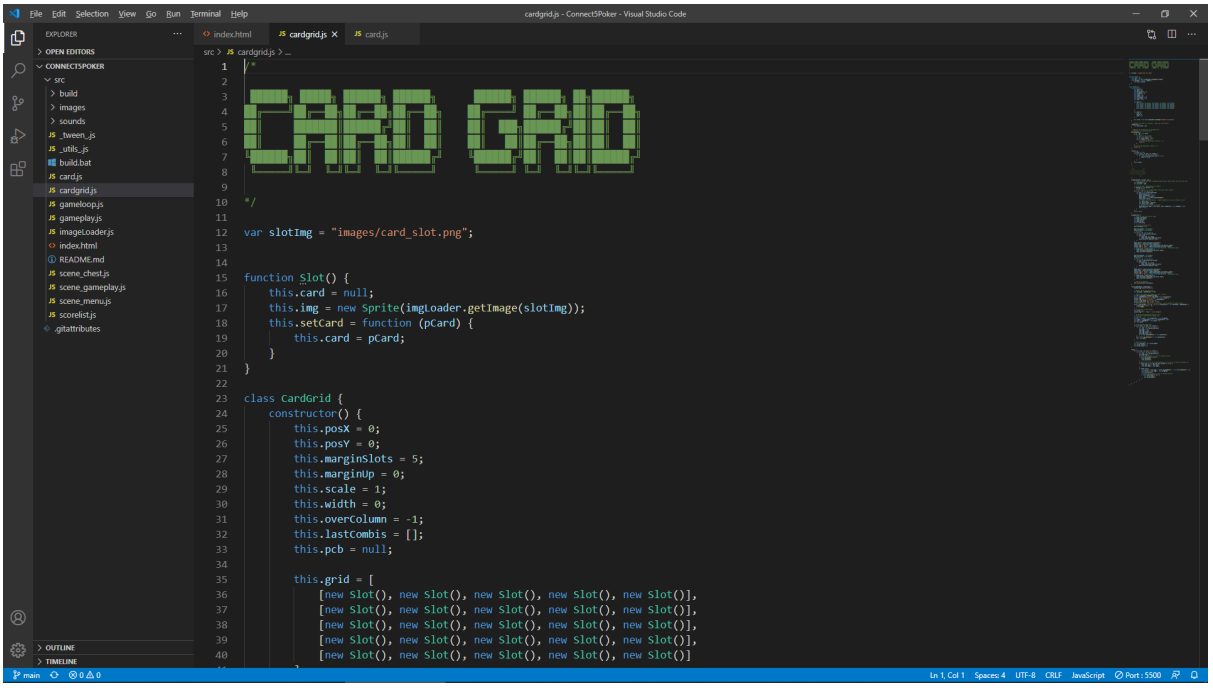
Donc, votre navigateur fait office de plateforme d'exécution, incluant le moteur.

Théoriquement, Notepad pourrait suffir à taper votre code, mais nous aimons un peu de confort donc nous allons vous préparer un petit environnement de développement à la fois simple et pro.

Tout d'abord, installez Visual Studio Code.

https://code.visualstudio.com/

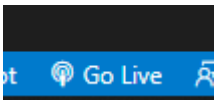
Cet éditeur de code est aujourd'hui un des plus utilisé et il est pratique, open-source, et fonctionne sur toutes les plateformes (même un Raspberry Pi).



En complément de Visual Studio Code, nous avons besoin d'une extension qui va nous aider à lancer notre jeu dans le navigateur, comme si il était hébergé sur un serveur :

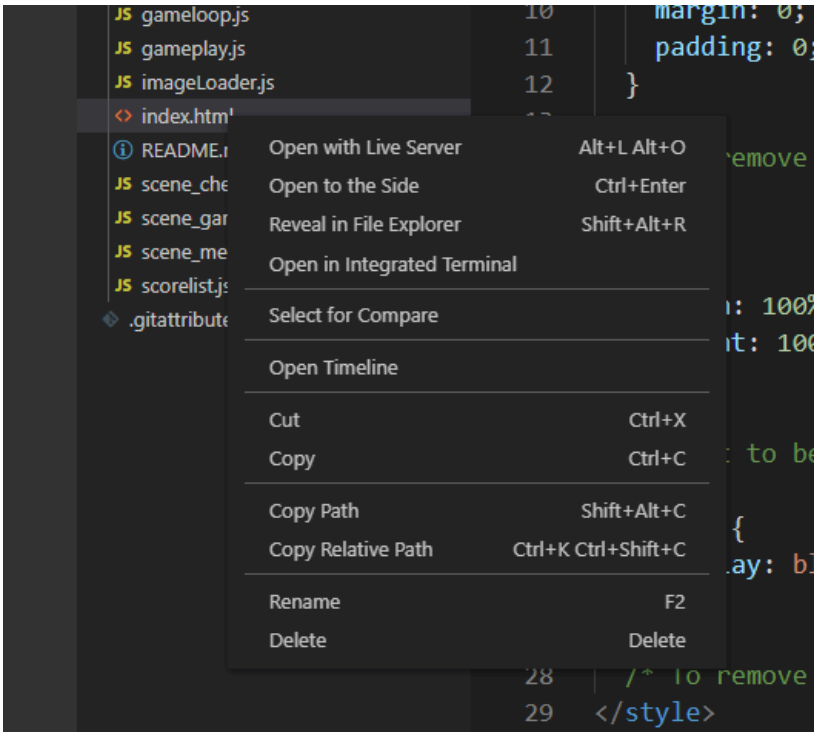
Live Server (suivez les instructions dans ma vidéo pour l'installer)

Pour tester votre jeu il vous suffira ensuite de cliquer sur le bouton :



Il est situé en bas à droite de l'éditeur une fois l'extension activée.

Autre méthode : un clic-droit sur votre fichier .html puis l'option : "Open with Live Server".



Mais tout cela vous est expliqué et démontré dans ma vidéo :

▸ Voir la vidéo correspondante : 0 - Installation de votre environnement de développement

Note : une autre alternative est l'éditeur Brackets : http://brackets.io/. Je le présente dans la vidéo. Brackets inclut une fonction "Live preview" équivalente à l'extension Live Server.

Voilà vous êtes parés pour coder !

Votre premier code

▸ Voir la vidéo correspondante : 1 - Votre premier code

Dans cette première étape, et en suivant pas à pas les instructions de ma vidéo, vous allez construire la structure de base nécessaire pour démarrer un jeu Web en JavaScript.

Il s'agit tout simplement d'une page HTML.

Cette page contient des "scripts" en JavaScript, qui vont lui donner vie.

C'est le même principe que pour créer un site web, donc vous allez aussi apprendre quelques bases de HTML pour y arriver.

Pour référence, voici le code minimum de votre page HTML :

```
<html>
    <body>

        <canvas id="canvas" width="1280" height="720"
                  style="background-color: cornflowerblue;"></canvas>

        <script src="jeu.js"></script>

    </body>
</html>
```

Le fichier jeu.js contenant le code JavaScript :

```
console.log("Bonjour tout fonctionne !");
```

Quelques conseils et astuces :

Habituez-vous à ajouter un point-virgule à la fin de chacune de vos lignes de code

C'est optionnel je sais, mais pourquoi le faire ?

- C'est une bonne habitude pour passer facilement à des langages qui ont une syntaxe quasi identique : C, C++, C#, Java.
- Dans Visual Studio Code, cela donne le signal à l'éditeur de mettre en forme la ligne que vous venez de taper. VS Code va ajouter des espaces là où il faut et tout sera propre.

Commentez votre code

Pour ajouter un commentaire dans du JavaScript c'est comme en C++ :

```
// Commentaire sur une ligne
/* Commentaire sur
   plusieurs lignes /*
```

Tracez votre code

Il est très utile, voire indispensable, de savoir ce qui se passe dans votre code et dans vos variables. La meilleure façon de le faire est de "tracer" votre code, en affichant des messages dans la console de votre éditeur.

La console sera visible au sein de votre navigateur, tout simplement en ouvrant les outils pour développeurs (Ctrl + Maj + i) ou (F12). Je vous montre ça dans la vidéo.

Pour afficher quelque chose dans la console :

```
console.log(....)
```

Entre les parenthèse vous pouvez y mettre une variable ou bien construire une chaîne de caractères :

```
let vies = 5;
console.log("Nombre de vie " + vies);
```

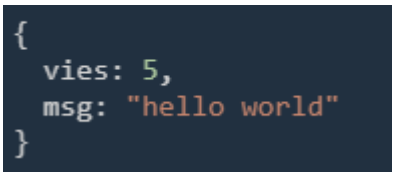
ou afficher plusieurs valeurs séparées par des virgules :

```
console.log(vies, score);
```

Autre syntaxe plus élaborée :

```
let vies = 5;
console.log({vies});
```

Cette dernière version va afficher dans la console un tableau, façon JSON avec l'association "nom: valeur" pour chacune des variables :



C'est très pratique !

Ou encore un affichage façon printf du C :

```
let joueur = "david";
let score = 100;
console.log("%s a un score de %i", joueur, score);
```

Voici les différents spécificateurs possibles :

%s	La valeur est convertie en chaîne de caractères
%d or %i	La valeur est convertie en numérique entier
%f	La valeur est convertie en numérique flottant (à virgules)
%o	La valeur est convertie dans le format le plus adapté

Remarquez le dernier spécificateur %o (c'est la lettre "o") vous pouvez l'utiliser systématiquement car il se débrouille et est même capable de vous afficher un objet avec tous ses membres !

Voilà !

Note : Il y a aussi une autre méthode avec des accolades dans une chaîne de caractères mais je ne l'aime pas, elle ne permet pas d'encadrer avec des guillemets.

Le langage JavaScript

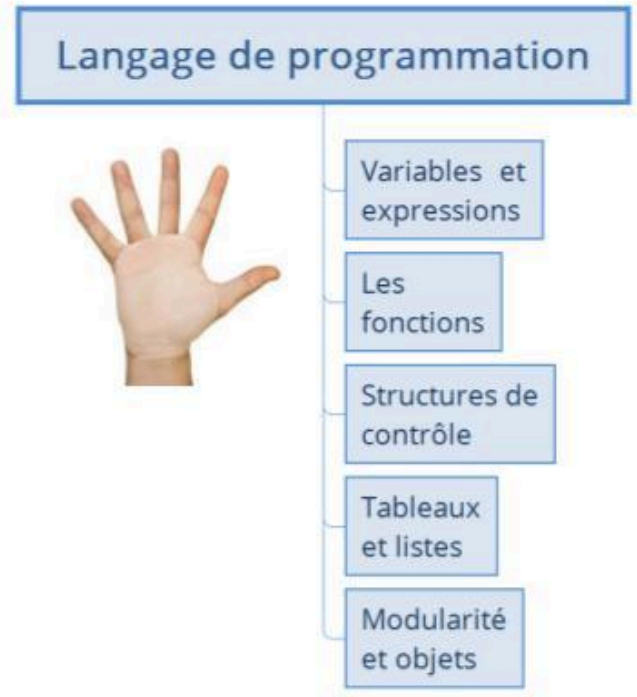
▸ Voir la vidéo correspondante : 2 - Apprenez JavaScript en 35 minutes

Dans cette vidéo, vous allez suivre le principe des fondamentaux Gamecodeur pour apprendre rapidement (en 35 mn) les bases du langage JavaScript.

Je vous conseille d'avoir des bases en programmation (suivre ma formation ici).

Tout est dans la vidéo mais en guise de pense bête et référence, voici une version texte résumée.

Je vous donne ici les 5 fondamentaux à apprendre, même sans rentrer dans les détails, pour commencer à travailler avec un langage :



Pour chaque sous chapitre, après vous avoir donné la syntaxe de base et un exemple, je vous donne le lien de la documentation officielle pour approfondir.

Je vous conseille de lire cette documentation, même en diagonale. Habituez vous à lire de la doc ! Je ne peux pas vous donner TOUS les détails sinon cet atelier s'appellerait : La programmation de Jeux Web pour les experts maniaques et compulsifs qui ne veulent avancer que quand ils savent tout dans le moindre détail.

Les variables

Rien de plus simple avec JavaScript, c'est comme en Lua : il n'y a pas de typage (inutile de préciser si la variable est de type entier, flottant, etc).

Vous devez simplement commencer votre ligne par "let ". Si on devait traduire ce terme, on le traduirait pas "permet" ou "autorise".

Exemple :

```
let score = 100;
let vies = 5;
```

Il existe aussi le mot clé "const" qui permet de déclarer une variable, de lui assigner une valeur et de la verrouiller (on ne peut plus changer sa valeur ensuite, elle est "constante") :

```
const marge = 5;
marge = 2; // Erreur, c'est interdit de changer la valeur de "marge"
```

Voici les principaux types reconnus par JavaScript :

```
// Entiers
let score = 100;
// Flottants
let marge = 5.5;
// Booléens (valeur : true ou false):
let porteOuverte = true;
// Chaîne de caractères
let nom = "gamecodeur";
```

💡 Référence utile :
https://developer.mozilla.org/fr/docs/Web/JavaScript/Structures_de_donn%C3%A9es
(à noter que le mot clé "var" est déconseillé et à remplacer par let/const aujourd'hui).

Les expressions

Rien de bien compliqué ici.

Le plus direct est d'apprendre à construire des expressions avec :

- Les calculs mathématiques de bases
- Les incréments / décréments
- La concaténation de chaînes

```
// Calcul simple
vies = vies - 1

// ou bien façon C
vies--;     // Décrémente de 1
vies++; // Incrémente de 1
vies *= 2; // Multiplie par 2
vies /=2; // Divise par 2
```

Pour les chaînes de caractères :

```
nom = "david";
nom += " de Gamecodeur";
// Le résultat est "david de Gamecodeur"
```

💡 Référence utile :
https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Expressions_et_Op%C3%A9rateurs

Les structures de contrôle

Voir https://www.w3schools.com/js/js_if_else.asp.

Les conditions :

```
if (vies >= 3) {
      // Le joueur a 3 vies ou plus
}
else if (vies <= 3) {
      // Le joueur a moins de 3 vies
}
else {
      // Autres cas
}
```

Pour les "et" et "ou" la syntaxe JavaScript est encore une fois celle du C :

&& = ET
|| = OU

Pour les 2 barres, il faut faire Alt GR + 6.

Exemple :

```
if (vies >=3 && vies <= 5) {
      console.log("Il reste plus de 3 vies et moins de 5 vies");
}
```

Les boucles :

La boucle For :

```
for (let index = 0; index <= 10; index++) {
    console.log(index);
}
```

Ici le bloc de code contenu entre les 2 accolades sera exécuté 11 fois. La console affiche les valeurs de 0 à 10.

Le For c'est un peu complexe quand on débute en programmation.

Entre parenthèses il y a 3 blocs séparés par des points-virgules :


```
for (let index = 0; index <= 10; index++) {
    console.log(index);
}
```

1. Ce bloc initialise la variable qui va servir de compteur. Elle y est traditionnellement déclarée et assignée. Ici on déclare une variable "index" et on lui donne la valeur 0.

2. Ce bloc est la condition qui détermine si la boucle continue de s'exécuter ou pas.

3. Ce bloc modifie la variable à chaque itération de la boucle, dans notre cas on incrémente la variable index.

Le do - while :

```
let index = 0;
do {
    console.log(index);
    index++;
} while (index < 10);
```

Il y a également un "while" simple :

```
var i = 0;
while (i < 5) {
      // Faire quelque chose tant que i est inférieur à 5
      i++;
}
```

💡 Référence utile :
https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Boucles_et_it%C3%A9ration

Les fonctions

Voici une fonction simple qui reçois 2 arguments (ou paramètres), les additionne et renvoie le total :

```
function ajoute(px, py) {
      let total = px + py;
      return total;
}
```

Pour appeler la fonction :

```
let resultat = ajoute(10, 20);
```

Notez les accolades pour marquer le début et la fin de la fonction, comme en C.

Note sur la portée de variables :

Les variables "let" sont locales au module si elles sont déclarées à la racine du code, ou locales à la fonction si elles sont déclarées dans une fonction. Je vous en fais la démo dans la vidéo.

Il existe aussi un mot clé "var" à la place de "let" qui rend la variable globale entre modules si elle est déclarée à la racine du code. Attention, les variables "var" sont sensibles aux bugs car on peut les déclarer plusieurs fois sans que JavaScript ne râle.

💡 Référence utile :
https://developer.mozilla.org/fr/docs/Web/JavaScript/Guide/Fonctions

Les listes et tableaux

En JavaScript les listes et les tableaux c'est la même chose.

Pour créer un tableau :

```
let jours = ["lundi", "mardi", "mercredi"];
```

A noter qu'en JavaScript, comme en C, C++, C#, etc. les tableaux sont indexés à partir de 0.

Donc, voici les index pour le tableau qu'on vient de créer :

```
                0         1          2
let jours = ["lundi", "mardi", "mercredi"];
```

Donc le code :

```
console.log(jours[0]);
```

Va afficher : "lundi".

Pour gérer votre tableau façon liste :

```
let jours= [];
jours.push("lundi");
jours.push("mardi");
jours.push("mercredi");
```

La méthode push va ajouter à la fin de la liste. Utilisez "pop" si vous voulez supprimer le dernier élément, "shift" pour supprimer le premier, et "unshift" pour ajouter au début.

Pour supprimer un élément au milieu du tableau, utilisez "splice" :

```
jours.splice(1,1);

console.log({jours});
```

Splice reçoit en 1er paramètre la position où supprimer dans la liste/le tableau, et en 2ème paramètre le nombre d'éléments à supprimer. Dans mon cas : 1 élément.

Le résultat affiché est donc :

```
jours:(2) [
"lundi",
"mercredi"
]
```

Car nous avons supprimé l'élément à la position 1 (donc le 2ème élément puisque le premier élément est l'élément à la position 0).

Pour parcourir une liste/un tableau, voici les 2 méthodes classiques :

1) La plus illisible pour un débutant, celle avec **foreach** (2 versions) :

```
jours.forEach(function(item, index) {
  console.log(item, index);
});
```

ou la version avec la fonction fléchée :

```
jours.forEach(element => {
    console.log(element);
});
```

2) La méthode pour débutant avec un for :

```
for (index = 0; index < jours.length; index++) {
    console.log(jours[index]);
}
```

💡 Référence utile :
https://developer.mozilla.org/fr/docs/Learn/JavaScript/First_steps/tableaux

Les variables complexes et les objets

Voici comment créer une variable complexe comme les tables en Lua.
(C'est une sorte de tableau associatif pour JavaScript)

```
let hero = {
      x: 10;
      y: 50;
      speed: 2;
}
```

Et pour un objet :

```
class ennemi {
    constructor(px, py) {
        this.x = px;
        this.y = py;
        this.energie = 100;
    }

    hit(pNbrPoints) {
        this.energie -= pNbrPoints;
        console.log("L'ennemi a maintenant :", this.energie);
    }
}
```

Créer une instance d'objet et appeler une de ses méthodes :

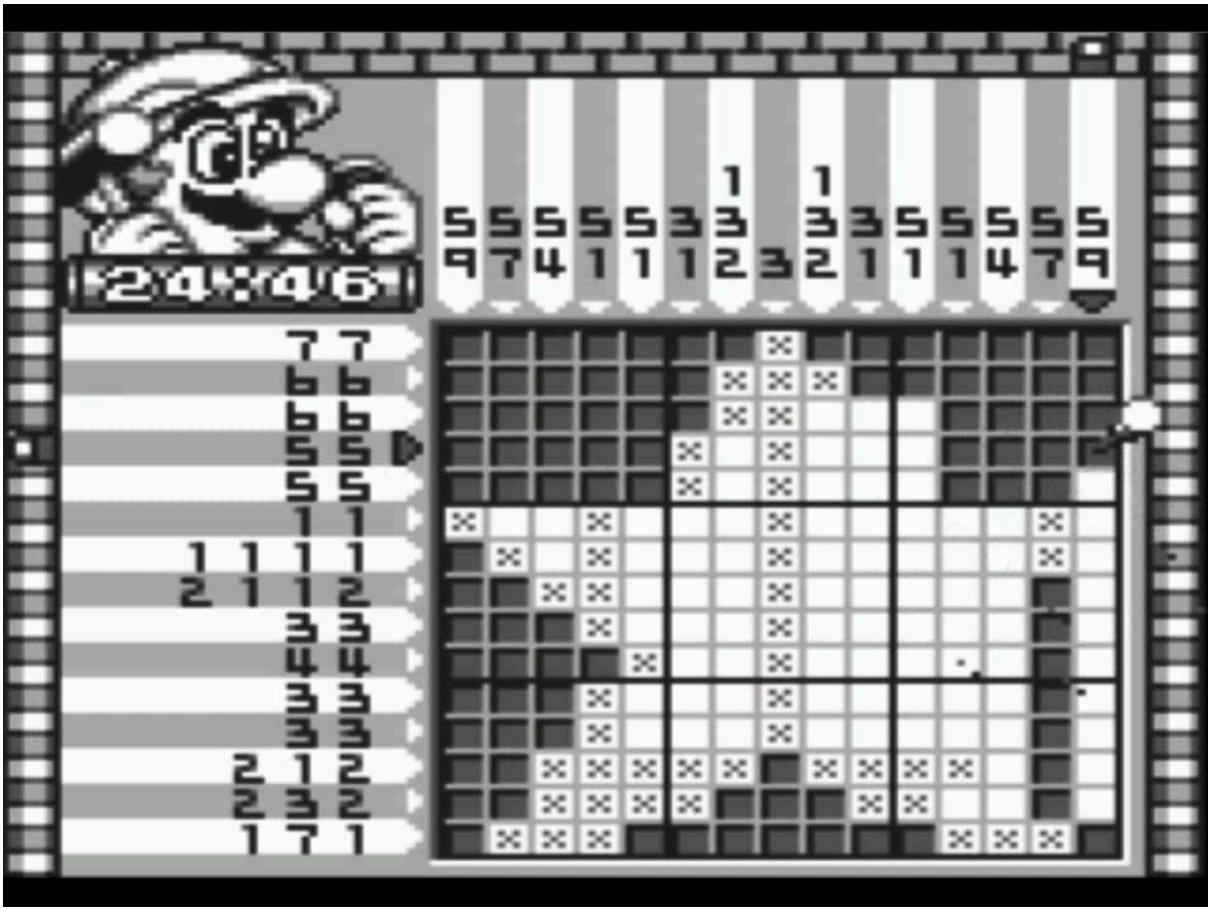
```
let monEnnemi = new ennemi(10, 10);

monEnnemi.hit(10);
```

Après l'exécution de ce code, notre variable "monEnnemi" aura une valeur de 90 à son membre "energie".

Référence utile :
https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Classes

Exercice de programmation



Dans Picross, le joueur doit cocher des cases afin de respecter la série affichée dans la ligne ou la colonne.

Par exemple sur cette image, sur la première ligne, on a 7 cases noires, une case blanche, et à nouveau 7 cases noires. Il est donc affiché "7 7" au début de la ligne : 7 cases noires contiguës, un écart, puis 7 cases noires contiguës. Au joueur de deviner comment ces séries contiguës de cases noires sont réparties.

Voir le jeu en action : https://www.youtube.com/watch?v=GN8Fydge0nw.

Sur la 8eme ligne, la ligne affiche "2 1 1 2". Cela signifie qu'il y a 2 cases noires, puis 1 case noire, puis 1 case noire, puis 2 cases noires. La difficulté est qu'on ne sait pas combien il y a de cases blanches qui séparent ces séries !

C'est tout le fun du jeu. L'astuce c'est de commencer par les lignes faciles. La 1ère par exemple, car il y a 15 cases sur la ligne, la solution ne peut être que d'avoir 7 cases noires, une case blanche, puis 7 cases noires.

Voici maintenant un exercice de programmation :

Note : Je vous demande de vous y casser les dents ! Si vous débutez alors c'est un exercice compliqué, vous n'y arriverez vraisemblablement pas… MAIS VOUS APPRENDREZ À ÉCHOUER !

Avec les connaissances en programmation que vous avez, et les bases de JavaScript que je vous ai enseignées, résolvez cet exercice :

Compter les séries de 1 dans un tableau, considérant qu'elles sont séparées par un ou plusieurs 0.

- Partant par exemple d'un tableau de valeur [1,1,1,0,1,0,1,1,1,0,1,1]
- Compter les séries de 1, espacées par un ou plusieurs 0
- Dans cet exemple le résultat est : 3,1,3,2
- Il y a en effet une série de trois 1, puis un 1 tout seul, puis encore une série de trois 1, puis une série de deux 1
- Stockez le résultat dans un tableau et afficher le

C'est le principe de Picross !

Je vous donne la 1ère ligne de votre code :

```
let ligne = [1,1,1,0,1,0,1,1,1,0,1,1];
```

Voici quelques pistes de travail :

- Il faut parcourir ce tableau (je vous ai appris à le faire, revoir la leçon)
- Il faut une variable, une sorte de compteur, pour compter chaque fois qu'un élément du tableau est égal à 1. Vous pourriez l'appeler "compteur" ?
- Chaque fois que vous tombez sur un "un", vous augmentez le compteur
- Chaque fois que vous tombez sur un zéro, c'est là qu'il faut faire quelque chose…
    - Si le compteur est supérieur à 0, alors vous avez terminé une série de 1
    - Ajoutez votre compteur (avec "push") à un tableau qui vous servira de résultat
    - Remettez le compteur à zéro
- Il y a un cas particulier quand vous atteignez la fin de la ligne
    - Il n'y a pas de zéro à la fin, donc vous risquez d'abandonner une série de 1 en route, prenez cela en compte

Il y a 1000 façons de faire tout cela, inventez la vôtre !

▸ **Une solution vous est donnée en vidéo.** Ne la regardez qu'après avoir passé au moins 30 minutes sur l'exercice ou plus si vous êtes motivé(e) !

Apprendre à afficher un sprite et le déplacer au clavier, en 25 minutes !

▸ Voir la vidéo correspondante : 3 - It's alive ! Affichez une image et donnez vie à votre jeu

Dans cette partie on va aller vite, car je veux que vous preniez rapidement du plaisir après avoir galéré à apprendre le langage.

Tout ceci est très abstrait non ?

Alors affichons une image :

Et déplaçons là à l'écran, avec les touches du clavier, comme dans un jeu vidéo.

C'est parti ?

👉 Suivez la vidéo pas à pas, et utilisez ce support de cours comme référence.