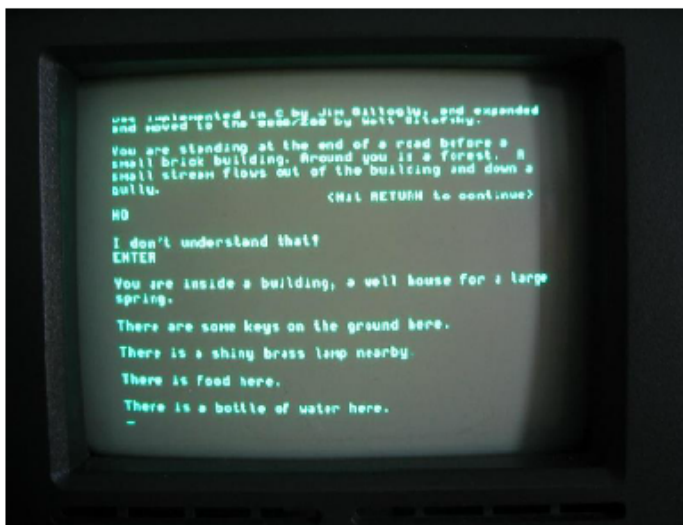


Programmer un jeu d'aventure textuel



Un jeu d'aventure textuel ?

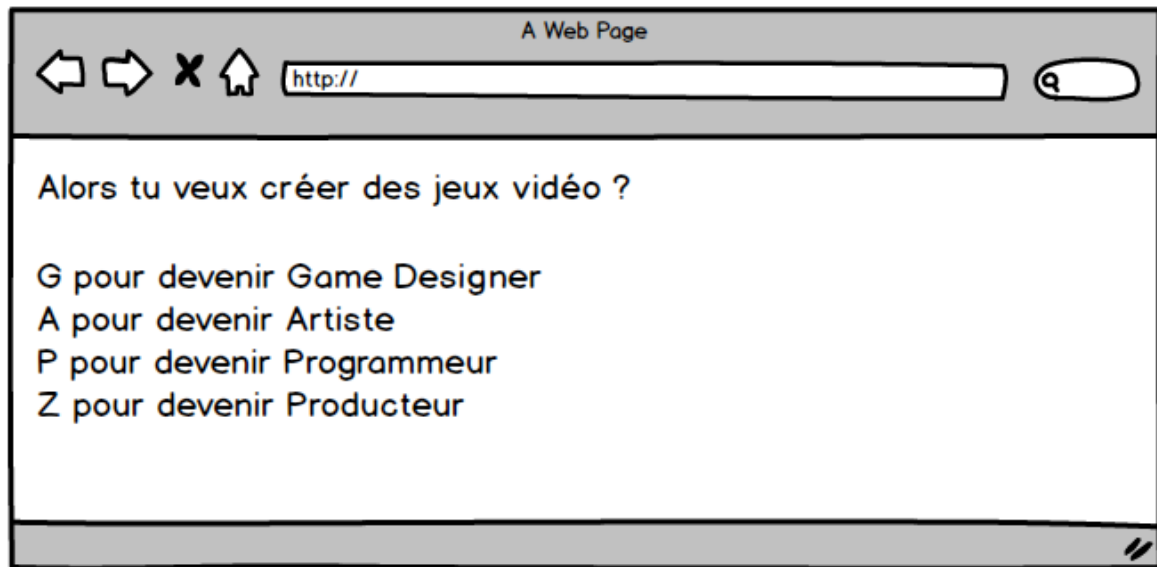


La BASE d'un jeu vidéo

```
fenêtre {  
    load();  
  
    boucle {  
        update();  
        draw();  
    };  
}
```

COMPLETED

Tranche verticale



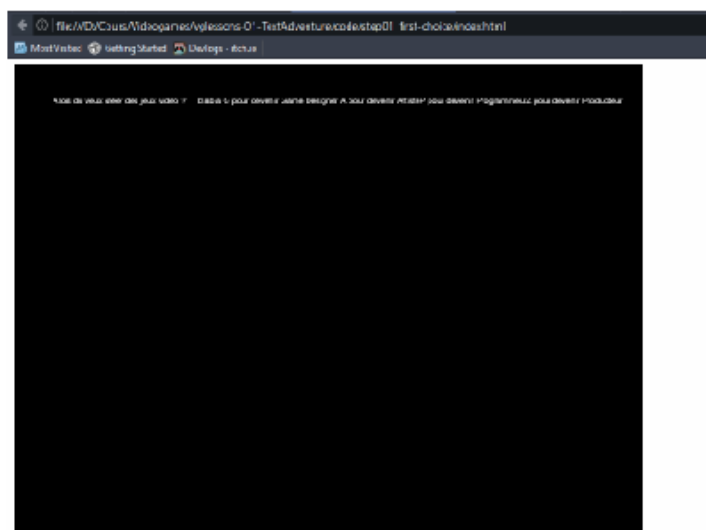
- Appuyer sur une touche du clavier nous envoie vers un autre choix.

Ecrire le texte sur plusieurs lignes : tentative 1

- On peut essayer d'écrire le texte en concaténant des chaînes de caractères :

```
function load() {  
  ...  
  text = 'Alors du veux créer des jeux vidéo ?' +  
        'G pour devenir Game Designer\ ' +  
        'A pour devenir Artiste' +  
        'P pour devenir Programmeur' +  
        'Z pour devenir Producteur';  
}
```

- Mais :



Ecrire le texte sur plusieurs lignes : tentative 2

- On va utiliser un tableau. On déclare et on remplit un tableau ainsi :

```
function load() {  
  ...  
  text = [];  
  text.push('Alors tu veux créer des jeux vidéo ?');  
  ...  
}
```

- Et on l'affiche ainsi :

```
function draw() {  
  ...  
  // Draw  
  canvasContext.fillStyle = 'white';  
  for (let i = 0; i < text.length; i++) {  
    canvasContext.fillText(text[i], 50, 50 + i * 20);  
  }  
}
```




file:///D:/Cours/Videogames/vglessons-01-TextAdventure/code/step02_multiline-array/index.html



Most Visited



Getting Started



Devlogs - itch.io

Alors du veux créer des jeux vidéo ?

G pour devenir Game Designer

A pour devenir Artiste

P pour devenir Programmeur

Z pour devenir Producteur

Implémenter les options

- Créer les différents textes pour les options
- Créer les interactions quand on appuie sur le bonnes touches. Les codes sont :
 - A : 65
 - G : 71
 - P : 80
 - Z : 90
- Ne pas oublier de créer les variables keyA, keyG etc.
- Ne pas oublier de remettre le texte à 0 avec :

```
text = [];
```



file:///D:/Cours/Videogames/vglessons-01-TextAdventure/code/step03_create-options/index.html



Most Visited



Getting Started



Devlogs - itch.io

Plutôt programmeur Gameplay ou Moteur ?

G pour Gameplay

M pour Moteur



Deux problèmes



- Premier problème : Si on appuie sur A alors qu'on est sur le texte du Programmeur, on va sur la page Artiste
- Second problème : la touche G ne peut pas servir à la fois pour Game Designer, programmeur Gameplay ou Grosse production

Deux problèmes, une solution : la machine à état


- Une machine à état, c'est une machine qui se comporte de telle manière dans un état...
- ...et de telle autre quand elle est dans un autre état
- Pour implémenter notre machine à état, on va créer une variable state qui contiendra le nom de l'état
- On testera l'appui des touches en fonctions de l'état dans lequel on est
- On peut tester un état (par exemple l'état 'start') avec la condition :

```
if(state == 'start') {  
    ...  
}
```

Deux problèmes, une solution : la machine à état

• main.js

```
function update() {  
  if(state == 'start') {  
    ...  
    if (keyG) {  
      text = [];  
      text.push('Tu préfères les petits jeux Smartphones ou les  
Grosses productions ?');  
      text.push('S pour Smartphone');  
      text.push('G pour Grosses production');  
      state = 'gd';  
    }  
    ....  
  }  
  if(state == 'gd') {  
    if (keyG) {  
      text = [];  
      text.push('Il te faudra te spécialiser au sein d\'une équipe.');
```



```
      state = 'gd grosses productions';  
    }  
    if (keyS) {  
      text = [];  
      text.push('Tu vas devoir travailler seul sur plusieurs  
compétences.');
```

Caractère d'échappement
pour l'apostrophe



Problème



- Si on appuie sur G (game designer) on se retrouve dans l'état Game designer grosses productions
- C'est comme si on avait appuyé deux fois sur G
- D'où vient le souci ?

Réinitialiser les contrôles : design

- Le problème vient du fait que `keyG` reste à `true` une fois appuyée, donc le choix Grosses production est sélectionné immédiatement
- La solutions c'est de remettre à `false` les variables `keyXXX` à chaque fois que l'on change d'état
- Créer une fonction `resetKeys()` pour cela

Réinitialiser les contrôles : code

- main.js

```
function update() {  
  ...  
  if(state == 'gd') {  
    if (keyG) {  
      text = [];  
      text.push('Il te faudra te spécialiser au sein d\'une équipe.');      state = 'gd grosses productions';  
      resetKeys();  
    }  
    if (keyS) {  
      text = [];  
      text.push('Tu vas devoir travailler seul sur plusieurs compétences.');      state = 'gd smartphones';  
      resetKeys();  
    }  
  }  
  ...  
}  
  
function resetKeys() {  
  keySpace = keyA = keyG = keyM = keyP = keyS = keyZ = false;  
}
```


Améliorer le code : éviter les copier/coller

- Dans le code précédent, on écrit plusieurs fois :

```
state = 'gd grosses productions';  
resetKeys();
```

- On peut remplacer cela par une fonction qui prendra le nouvel état en argument.
- Cette fonction aura le prototype :

```
function changeState(newState)
```

On passe un argument,
qui sera utilisé dans la
fonction

- Et sera appelée grace à :

```
changeState('gd grosses production');
```

Améliorer le code : éviter les copier/coller

• main.js

```
function update() {  
  ...  
  if(state == 'gd') {  
    if (keyG) {  
      text = [];  
      text.push('Il te faudra te spécialiser au sein d\'une équipe.');      changeState('gd grosses productions');    }  
    if (keyS) {  
      text = [];  
      text.push('Tu vas devoir travailler seul sur plusieurs compétences.');      changeState('gd smartphones');    }  
  }  
  ...  
}  
  
function changeState(newState) {  
  state = newState;  
  resetKeys();  
}
```

Améliorer le code : séparer données et gameplay

- En l'état du code, on mélange l'attribution de la variable texte et le changement d'état :

```
function update() {  
  if(state == 'start') {  
    if (keyA) {  
      text = [];  
      text.push('Tu te spécialises en graphisme 2D ou en 3D ?');  
      text.push('D pour 2D');  
      text.push('T pour 3D');  
      changeState('artiste');  
    }  
    ...  
  }  
}
```

- Ce n'est pas très pratique quand on veut ajouter un état ou changer le texte d'un état
- On va séparer cela en créer une méthode `changeText(state)` prend en entrée un état et modifie le texte en fonction

Améliorer le code : séparer données et gameplay

• main.js

```
function changeState(newState) {
  state = newState;
  resetKeys();
  changeText();
}

function update() {
  if (state == 'start') {
    if (keyA) {
      changeState('artiste');
    }
    if (keyG) {
      changeState('gd');
    }
    ...
  }
  if (state == 'gd') {
    if (keyG) {
      changeState('gd grosses productions');
    }
    ...
  }
}

function changeText() {
  if (state == 'start') {
    text = [];
    text.push('Alors du veux créer des jeux vidéo ?');
    ...
  }
  if (state == 'artiste') {
    text = [];
    text.push('Tu te spécialises en graphisme 2D ou en 3D ?');
    ....
  }
  ...
}
```

Maintenant que tout est bien rangé et que vous êtes un développeur content...



Creez votre propre aventure !

(Et pour rendre les choses faciles, permettez au joueur de réinitialiser le jeu avec Espace.)