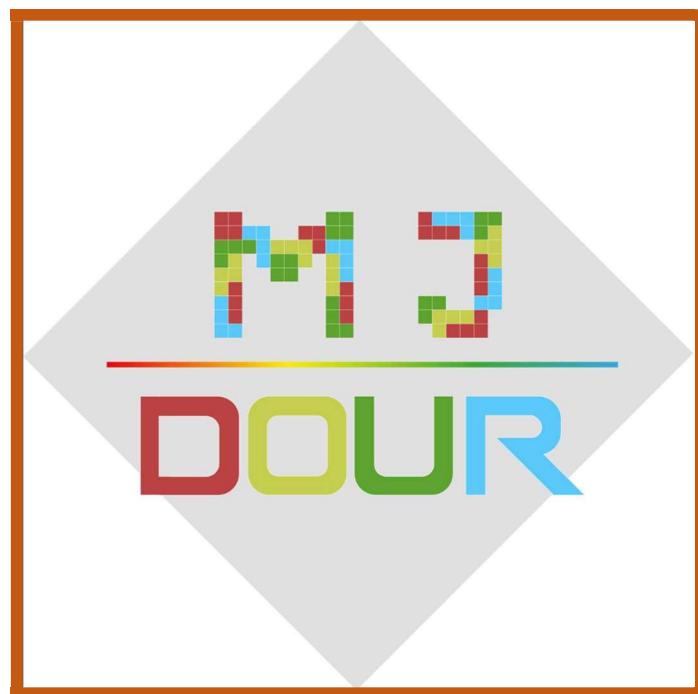


Stage de programmation

Codeur Commando



: Liolabs



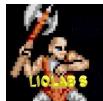
: Liolabs

CODEUR COMMANDO BY LIOLABS (LIOLABS1973@GMAIL.COM)

1

Table des matières

Installation des logiciels nécessaires :	3
Installation du moteur de jeu Love2D.....	3
Installation sur Windows.....	3
Installation sur macOS.....	3
Installation sur Linux	3
Installation de Zerobrane Studio :.....	5
Installation sur Windows.....	5
Installation sur macOS.....	5
Installation sur Linux	6
Tutoriel de base pour l'apprentissage du Moteur Love2D	8
Découvrir la structure d'un jeu Love2D.....	8
Écrire notre premier programme	8
Ajouter une image.....	9
Ajouter un personnage.....	10
Déplacer le personnage.....	12
Ajouter un ennemi	14
Ajouter un ennemi	15
Ajouter un objet bonus	17
Afficher un message de victoire	19
Ajouter de la musique	21
Ajouter des effets sonores	22
Jouer des effets sonores.....	23
Ajouter un score	24
Ajouter un écran de fin de jeu.....	25



Installation des logiciels nécessaires :

Installation du moteur de jeu Love2D.

Installation sur Windows

1. Allez sur le site officiel de LOVE2D : <https://love2d.org/>
2. Cliquez sur le bouton "Download" pour accéder à la page de téléchargement.
3. Cliquez sur le lien de téléchargement correspondant à votre version de Windows (32 bits ou 64 bits).
4. Enregistrez le fichier d'installation sur votre ordinateur.
5. Double-cliquez sur le fichier d'installation pour lancer l'assistant d'installation.
6. Suivez les instructions de l'assistant d'installation pour installer LOVE2D sur votre ordinateur.
7. Une fois l'installation terminée, vous pouvez utiliser LOVE2D en double-cliquant sur l'icône de l'application.

Installation sur macOS

1. Allez sur le site officiel de LOVE2D : <https://love2d.org/>
2. Cliquez sur le bouton "Download" pour accéder à la page de téléchargement.
3. Cliquez sur le lien de téléchargement correspondant à votre version de macOS.
4. Enregistrez le fichier d'installation sur votre ordinateur.
5. Double-cliquez sur le fichier d'installation pour ouvrir l'image disque.
6. Faites glisser l'icône de l'application LOVE2D dans le dossier Applications.
7. Une fois l'installation terminée, vous pouvez utiliser LOVE2D en ouvrant le dossier Applications et en double-cliquant sur l'icône de l'application.

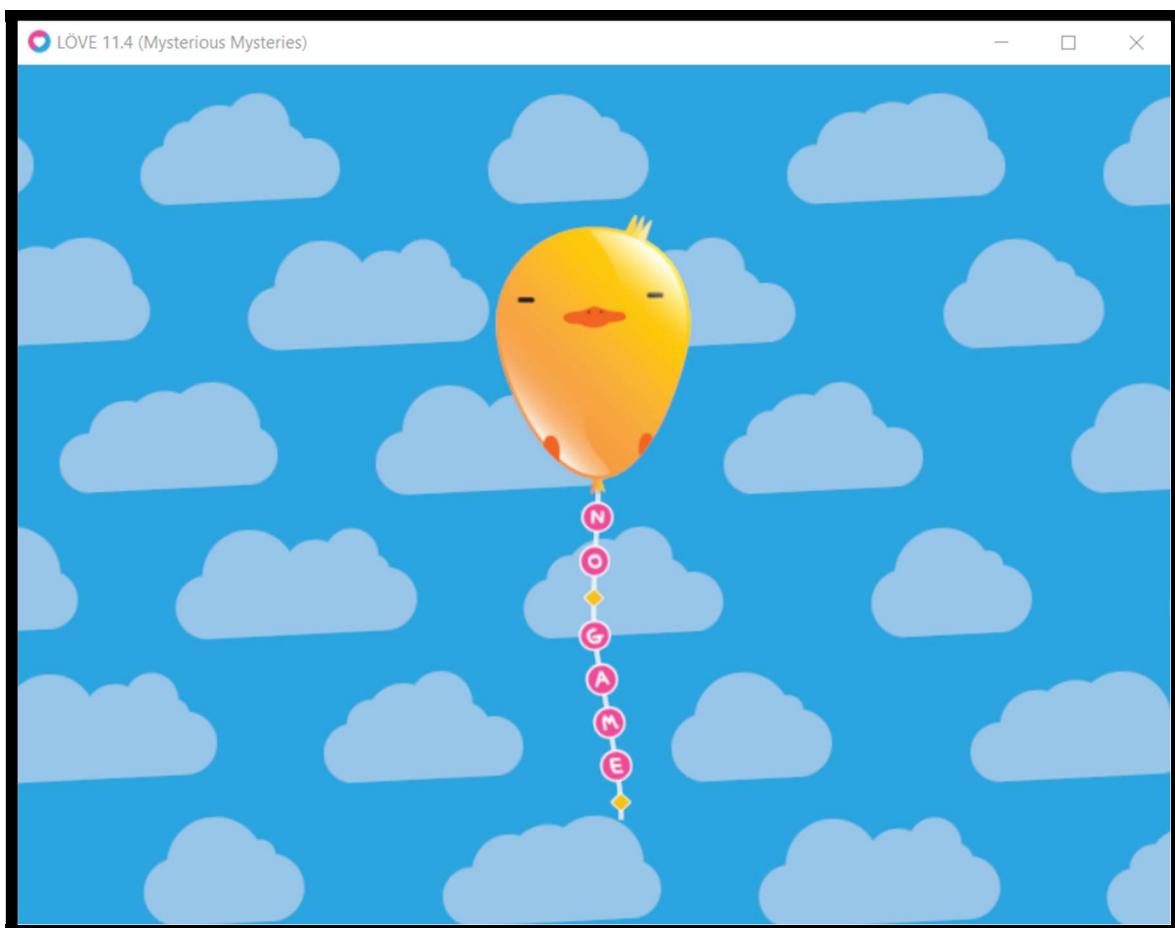
Installation sur Linux

1. Ouvrez un terminal et tapez la commande suivante pour installer les dépendances requises :
sudo apt-get install liblove0 liblove-dev love-doc
2. Téléchargez le fichier d'installation de LOVE2D depuis le site officiel :
<https://love2d.org/>
3. Décompressez le fichier d'installation en utilisant la commande suivante :
unzip love-11.3-linux-x86_64.zip
4. Copiez l'application LOVE2D dans le dossier /usr/local/bin avec la commande suivante :
sudo cp love-11.3-linux-x86_64/love /usr/local/bin



5. Vous pouvez maintenant utiliser LOVE2D en exécutant la commande suivante dans le terminal :
love chemin/vers/votre/jeu
6. Remplacez "chemin/vers/votre/jeu" par le chemin d'accès vers votre jeu LOVE2D.

Si l'installation s'est bien déroulée, lorsque vous exécutez la commande love2D, vous devriez obtenir ceci :



: Liolabs



: Liolabs

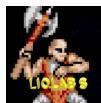
Installation de Zerobrane Studio :

Installation sur Windows

1. Allez sur le site officiel de Zerobrane Studio : <https://studio.zerobrane.com/>
2. Cliquez sur le bouton "Download" pour accéder à la page de téléchargement.
3. Cliquez sur le lien de téléchargement correspondant à votre version de Windows (32 bits ou 64 bits).
4. Enregistrez le fichier d'installation sur votre ordinateur.
5. Double-cliquez sur le fichier d'installation pour lancer l'assistant d'installation.
6. Suivez les instructions de l'assistant d'installation pour installer Zerobrane Studio sur votre ordinateur.
7. Une fois l'installation terminée, vous pouvez utiliser Zerobrane Studio en double-cliquant sur l'icône de l'application.

Installation sur macOS

1. Allez sur le site officiel de Zerobrane Studio : <https://studio.zerobrane.com/>
2. Cliquez sur le bouton "Download" pour accéder à la page de téléchargement.
3. Cliquez sur le lien de téléchargement correspondant à votre version de macOS.
4. Enregistrez le fichier d'installation sur votre ordinateur.
5. Double-cliquez sur le fichier d'installation pour ouvrir l'image disque.
6. Faites glisser l'icône de l'application Zerobrane Studio dans le dossier Applications.
7. Une fois l'installation terminée, vous pouvez utiliser Zerobrane Studio en ouvrant le dossier Applications et en double-cliquant sur l'icône de l'application.



: Liolabs

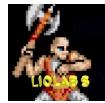


: Liolabs

Installation sur Linux

1. Ouvrez un terminal et tapez la commande suivante pour télécharger le fichier d'installation :
wget https://download.zerobrane.com/ZeroBraneStudioEduPack-1.90-linux.sh
2. Rendez le fichier d'installation exécutable en utilisant la commande suivante :
chmod +x ZeroBraneStudioEduPack-1.90-linux.sh
3. Exécutez le fichier d'installation en utilisant la commande suivante :
./ZeroBraneStudioEduPack-1.90-linux.sh
4. Suivez les instructions de l'assistant d'installation pour installer Zerobrane Studio sur votre ordinateur.
5. Une fois l'installation terminée, vous pouvez utiliser Zerobrane Studio en exécutant la commande suivante dans le terminal :
zerobrane-studio

Voilà, j'espère que ces instructions vous aideront à installer Zerobrane Studio sur votre ordinateur !

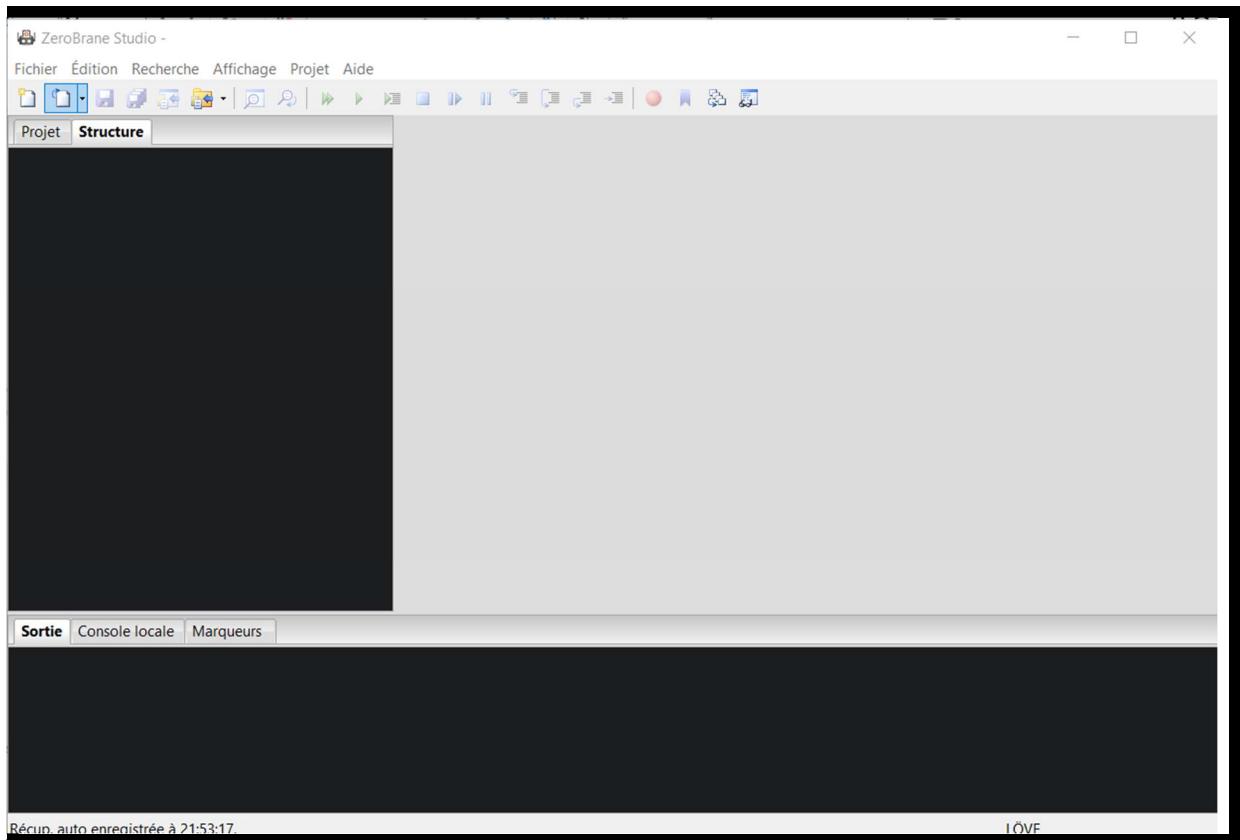


: Liolabs



: Liolabs

Si l'installation s'est bien déroulée, lorsque vous cliquez sur l'icône ZeroBrane Studio, vous devriez obtenir quelque chose de similaire à ceci :



Tutoriel de base pour l'apprentissage du Moteur Love2D

Découvrir la structure d'un jeu Love2D

Avant de commencer à créer notre jeu, examinons la structure d'un jeu Love2D. Chaque jeu Love2D est constitué de deux parties principales :

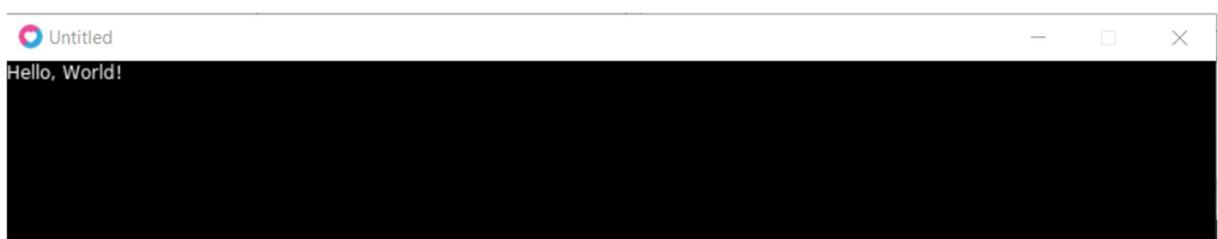
- Le fichier main.lua : c'est le point d'entrée de votre jeu. C'est ici que vous allez écrire le code qui va être exécuté lorsque le jeu démarre.
- Le dossier assets : c'est l'endroit où vous stockez tous les fichiers nécessaires à votre jeu, tels que les images, les sons, les polices, etc.

Écrire notre premier programme

Créez un nouveau dossier pour votre jeu et créez un nouveau fichier main.lua à l'intérieur. Ouvrez ce fichier dans un éditeur de texte et entrez le code suivant :

```
1  function love.draw()
2      love.graphics.print("Hello, World!")
3  end
4
```

Sauvegardez le fichier et exéutez-le en double-cliquant sur l'icône Love2D. Si tout fonctionne correctement, vous devriez voir la chaîne de caractères "Hello, World!" s'afficher à l'écran.



: Liolabs



: Liolabs

Ajouter une image

Créez un nouveau fichier dans le dossier assets appelé "background.png". C'est une image qui sera utilisée comme fond pour notre jeu. Téléchargez ou créez votre propre image pour cette étape.

Ajoutez le code suivant à votre fichier main.lua :

```
1  function love.load()
2      background = love.graphics.newImage("assets/background.png")
3  end
4
5
6  function love.draw()
7      love.graphics.draw(background, 0, 0)
8  end
9
```

Dans cette étape, nous avons créé un objet Image à partir de notre fichier d'image et nous l'avons affiché à l'écran en utilisant la fonction love.graphics.draw.



L'image étant plus petite que l'affichage nous allons apprendre comment :

- Soit redimensionner notre fenêtre de jeu (Dimensionner la fenêtre ainsi que pleins d'autres options hyper intéressantes).
- Soit redimensionner notre image pour qu'elle s'adapte à notre fenêtre de jeu (Agrandir, Rétrécir ou déformer l'image).

Ajouter un personnage

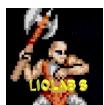
Maintenant que nous avons ajouté un arrière-plan, nous allons ajouter un personnage.

Ajoutez une nouvelle image dans le dossier assets appelée "character.png". C'est une image qui représente notre personnage. Téléchargez ou créez votre propre image pour cette étape.

Ajoutez le code suivant à votre fichier main.lua :

```
1  function love.load()
2      background = love.graphics.newImage("assets/background.png")
3      character = love.graphics.newImage("assets/character.png")
4  end
5
6  function love.draw()
7      love.graphics.draw(background, 0, 0)
8      love.graphics.draw(character, 100, 100)
9  end
10 
```

Dans cette étape, nous avons chargé l'image du personnage dans notre programme et l'avons dessiné sur l'écran à une position spécifique à l'aide de la fonction `love.graphics.draw()`. Vous pouvez modifier les coordonnées de la fonction `love.graphics.draw()` pour ajuster la position du personnage sur l'écran.





: Liolabs



: Liolabs

CODEUR COMMANDO BY LIOLABS (LIOLABS1973@GMAIL.COM)

11

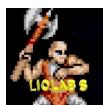
Déplacer le personnage

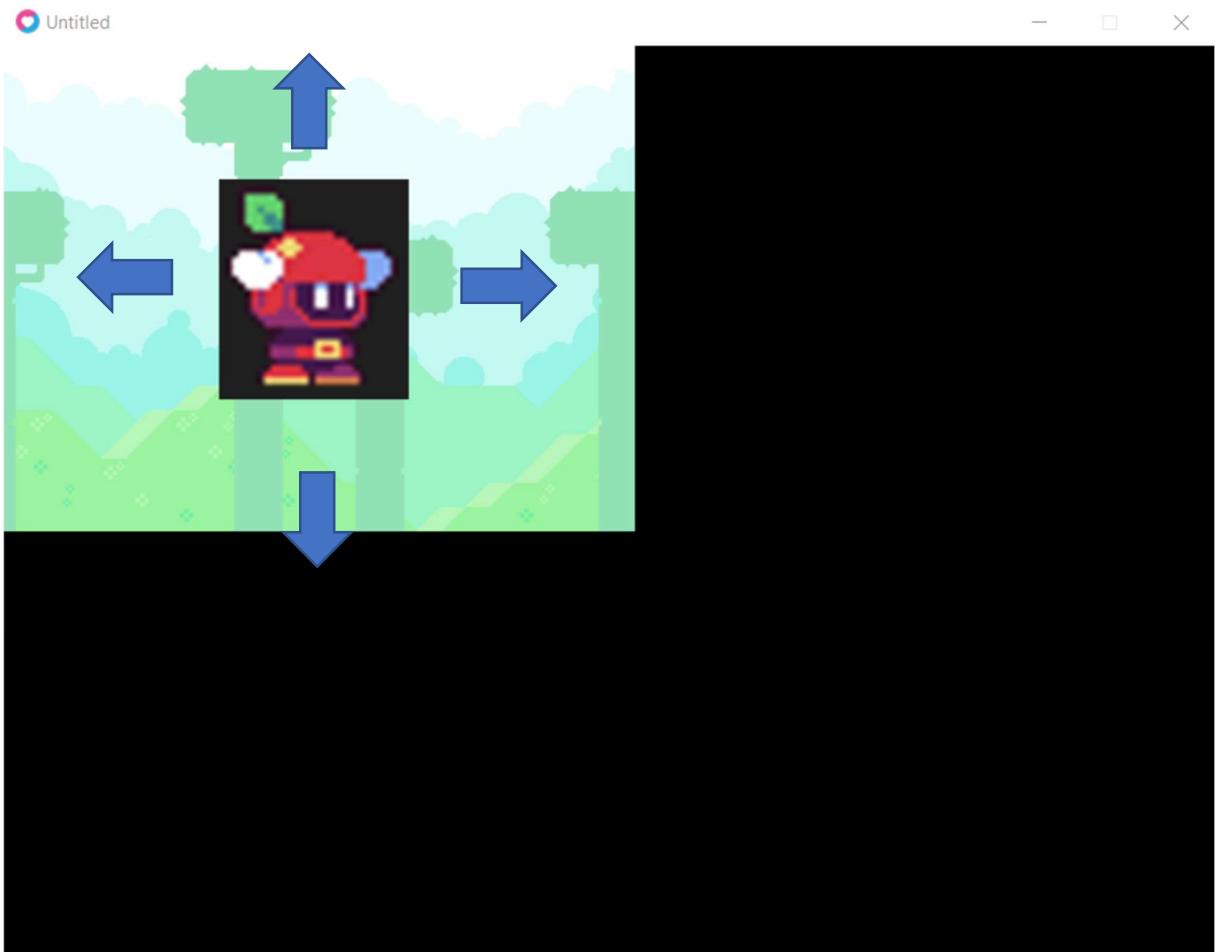
Maintenant que nous avons ajouté un personnage, nous allons le déplacer en réponse aux touches du clavier.

Ajoutez le code suivant à votre fichier main.lua :

```
1  function love.load()
2      background = love.graphics.newImage("assets/background.png")
3      character = love.graphics.newImage("assets/character.png")
4      characterX = 100
5      characterY = 100
6  end
7
8  function love.update(dt)
9      if love.keyboard.isDown("right") then
10         characterX = characterX + 100 * dt
11     elseif love.keyboard.isDown("left") then
12         characterX = characterX - 100 * dt
13     end
14
15     if love.keyboard.isDown("down") then
16         characterY = characterY + 100 * dt
17     elseif love.keyboard.isDown("up") then
18         characterY = characterY - 100 * dt
19     end
20   end
21
22 function love.draw()
23     love.graphics.draw(background, 0, 0)
24     love.graphics.draw(character, characterX, characterY)
25   end
```

Dans cette étape, nous avons ajouté deux variables characterX et characterY pour stocker la position du personnage. Nous avons également ajouté une fonction love.update(dt) qui est appelée à chaque frame et qui met à jour la position du personnage en fonction des touches du clavier.





CODEUR COMMANDO BY LIOLABS (LIOLABS1973@GMAIL.COM)

13



: Liolabs



: Liolabs

Ajouter un ennemi

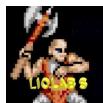
Maintenant que nous avons ajouté un personnage et que nous pouvons le déplacer avec les touches du clavier, nous allons ajouter un ennemi qui suit le personnage.

Ajoutez une nouvelle image dans le dossier assets appelée "enemy.png". C'est une image qui représente notre ennemi. Téléchargez ou créez votre propre image pour cette étape.

Ajoutez le code suivant à votre fichier main.lua :

```
1  function love.load()
2      background = love.graphics.newImage("assets/background.png")
3      character = love.graphics.newImage("assets/character.png")
4      enemy = love.graphics.newImage("assets/enemy.png")
5      characterX = 100
6      characterY = 100
7      enemyX = 500
8      enemyY = 300
9  end
10 
```

```
11 function love.update(dt)
12     if love.keyboard.isDown("right") then
13         characterX = characterX + 100 * dt
14     elseif love.keyboard.isDown("left") then
15         characterX = characterX - 100 * dt
16     end
17     if love.keyboard.isDown("down") then
18         characterY = characterY + 100 * dt
19     elseif love.keyboard.isDown("up") then
20         characterY = characterY - 100 * dt
21     end
22     if enemyX > characterX then
23         enemyX = enemyX - 50 * dt
24     elseif enemyX < characterX then
25         enemyX = enemyX + 50 * dt
26     end
27     if enemyY > characterY then
28         enemyY = enemyY - 50 * dt
29     elseif enemyY < characterY then
30         enemyY = enemyY + 50 * dt
31     end
32 end
```



```

34  function love.draw()
35      love.graphics.draw(background, 0, 0)
36      love.graphics.draw(enemy, enemyX, enemyY)
37      love.graphics.draw(character, characterX, characterY)
38  end
39

```

Dans cette étape, nous avons chargé l'image de l'ennemi dans notre programme et l'avons dessiné sur l'écran à une position spécifique à l'aide de la fonction `love.graphics.draw()`. Nous avons également ajouté deux variables `enemyX` et `enemyY` pour stocker la position de l'ennemi et une fonction `love.update(dt)` qui est appelée à chaque frame et qui met à jour la position de l'ennemi en fonction de la position du personnage.

Maintenant, lorsque vous déplacez le personnage avec les touches du clavier, l'ennemi le suit.

Ajouter un ennemi

Maintenant que nous avons ajouté un personnage et que nous pouvons le déplacer avec les touches du clavier, nous allons ajouter un ennemi qui suit le personnage.

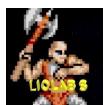
Ajoutez une nouvelle image dans le dossier assets appelée "enemy.png". C'est une image qui représente notre ennemi. Téléchargez ou créez votre propre image pour cette étape.

Ajoutez le code suivant à votre fichier main.lua :

```

1  function love.load()
2      background = love.graphics.newImage("assets/background.png")
3      character = love.graphics.newImage("assets/character.png")
4      enemy = love.graphics.newImage("assets/enemy.png")
5      characterX = 100
6      characterY = 100
7      enemyX = 500
8      enemyY = 300
9  end
10

```



```

11  function love.update(dt)
12  if love.keyboard.isDown("right") then
13      characterX = characterX + 100 * dt
14  elseif love.keyboard.isDown("left") then
15      characterX = characterX - 100 * dt
16  end
17  if love.keyboard.isDown("down") then
18      characterY = characterY + 100 * dt
19  elseif love.keyboard.isDown("up") then
20      characterY = characterY - 100 * dt
21  end
22  if enemyX > characterX then
23      enemyX = enemyX - 50 * dt
24  elseif enemyX < characterX then
25      enemyX = enemyX + 50 * dt
26  end
27  if enemyY > characterY then
28      enemyY = enemyY - 50 * dt
29  elseif enemyY < characterY then
30      enemyY = enemyY + 50 * dt
31  end
32 end
33

```

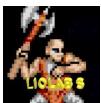
```

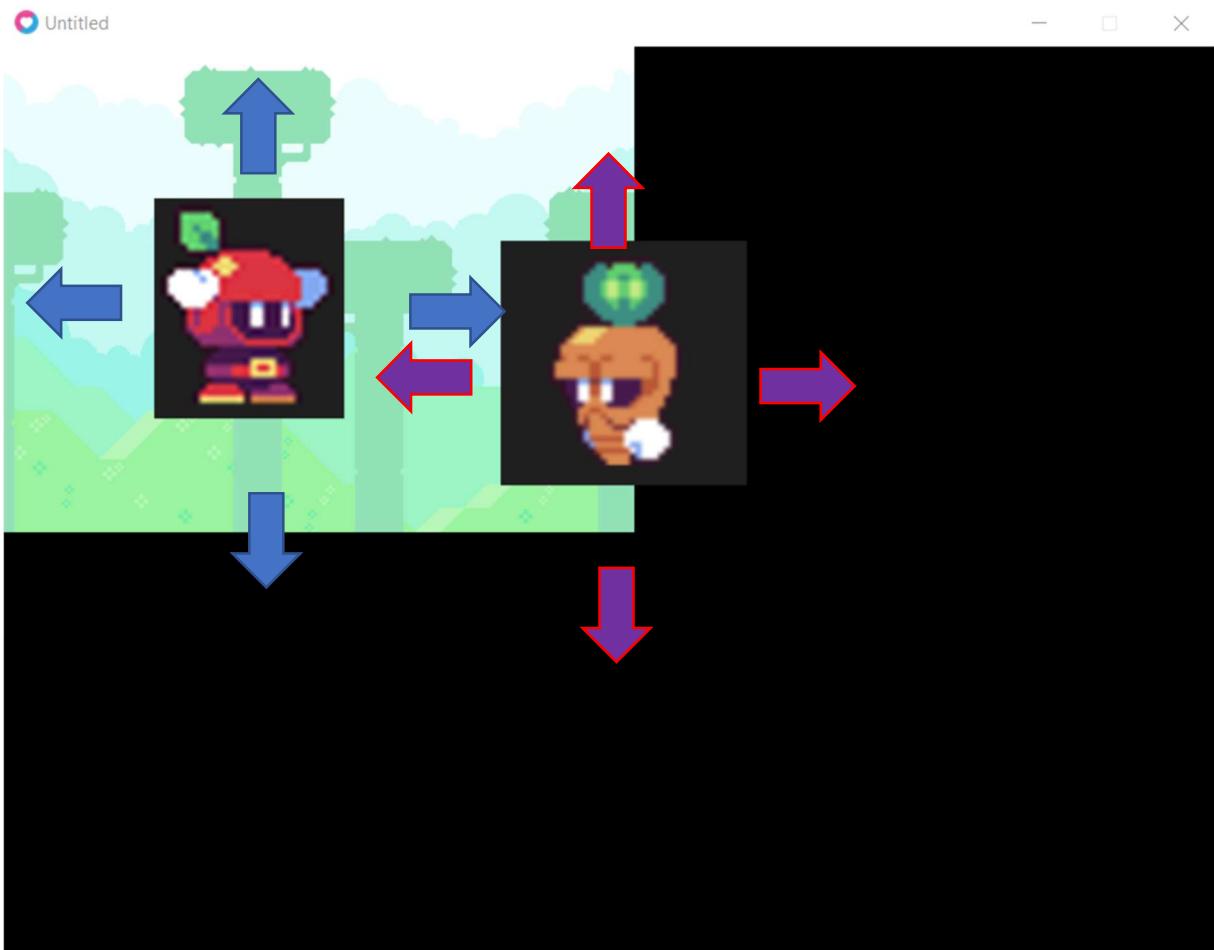
34  function love.draw()
35      love.graphics.draw(background, 0, 0)
36      love.graphics.draw(enemy, enemyX, enemyY)
37      love.graphics.draw(character, characterX, characterY)
38  end
39
40

```

Dans cette étape, nous avons chargé l'image de l'ennemi dans notre programme et l'avons dessiné sur l'écran à une position spécifique à l'aide de la fonction `love.graphics.draw()`. Nous avons également ajouté deux variables `enemyX` et `enemyY` pour stocker la position de l'ennemi et une fonction `love.update(dt)` qui est appelée à chaque frame et qui met à jour la position de l'ennemi en fonction de la position du personnage.

Maintenant, lorsque vous déplacez le personnage avec les touches du clavier, l'ennemi le suit.



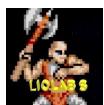


Ajouter un objet bonus

Maintenant que nous avons ajouté un personnage, un ennemi et une collision, nous allons ajouter un objet bonus qui peut être collectés par le personnage.

Ajoutez une nouvelle image dans le dossier assets appelée "bonus.png". C'est une image qui représente notre objet bonus. Téléchargez ou créez votre propre image pour cette étape.

Ajoutez le code suivant à votre fichier main.lua :



: Liolabs



: Liolabs

```

1  function checkCollision(x1,y1,w1,h1,x2,y2,w2,h2)
2      return x1 < x2 + w2 and
3          x2 < x1 + w1 and
4          y1 < y2 + h2 and
5          y2 < y1 + h1
6  end
7

```

```

8  function love.load()
9      background = love.graphics.newImage("assets/background.png")
10     character = love.graphics.newImage("assets/character.png")
11     enemy = love.graphics.newImage("assets/enemy.png")
12     bonus = love.graphics.newImage("assets/bonus.png")
13     characterX = 100
14     characterY = 100
15     enemyX = 500
16     enemyY = 300
17     bonusX = 200
18     bonusY = 400
19     bonusCollected = false
20  end
21

```

```

22  function love.update(dt)
23      if love.keyboard.isDown("right") then
24          characterX = characterX + 100 * dt
25      elseif love.keyboard.isDown("left") then
26          characterX = characterX - 100 * dt
27      end
28
29      if love.keyboard.isDown("down") then
30          characterY = characterY + 100 * dt
31      elseif love.keyboard.isDown("up") then
32          characterY = characterY - 100 * dt
33      end
34
35      if enemyX > characterX then
36          enemyX = enemyX - 50 * dt
37      elseif enemyX < characterX then
38          enemyX = enemyX + 50 * dt
39      end
40
41      if enemyY > characterY then
42          enemyY = enemyY - 50 * dt
43      elseif enemyY < characterY then
44          enemyY = enemyY + 50 * dt
45      end
46
47      if checkCollision(characterX, characterY, character:getWidth(), character:getHeight(),
48          enemyX, enemyY, enemy:getWidth(), enemy:getHeight()) then
49          love.load()
50      end
51
52      function love.draw()
53          love.graphics.draw(background, 0, 0)
54          love.graphics.draw(enemy, enemyX, enemyY)
55          if not bonusCollected then
56              love.graphics.draw(bonus, bonusX, bonusY)
57          end
58          love.graphics.draw(character, characterX, characterY)
59      end
60

```



```

52   function love.draw()
53     love.graphics.draw(background, 0, 0)
54     love.graphics.draw(enemy, enemyX, enemyY)
55     if not bonusCollected then
56       love.graphics.draw(bonus, bonusX, bonusY)
57     end
58     love.graphics.draw(character, characterX, characterY)
59   end
60

```

Afficher un message de victoire

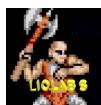
Maintenant que nous avons ajouté un objet bonus, nous allons ajouter un message de victoire pour indiquer que le joueur a réussi à collecter l'objet bonus.

Ajoutez le code suivant à votre fichier main.lua :

```

1  function checkCollision(x1,y1,w1,h1,x2,y2,w2,h2)
2    return x1 < x2 + w2 and
3           x2 < x1 + w1 and
4           y1 < y2 + h2 and
5           y2 < y1 + h1
6  end
7
8  function love.load()
9    background = love.graphics.newImage("assets/background.png")
10   character = love.graphics.newImage("assets/character.png")
11   enemy = love.graphics.newImage("assets/enemy.png")
12   bonus = love.graphics.newImage("assets/bonus.png")
13   characterX = 100
14   characterY = 100
15   enemyX = 500
16   enemyY = 300
17   bonusX = 200
18   bonusY = 400
19   bonusCollected = false
20 end
21

```



```

34
35  if bonusCollected == false then
36    if enemyX > characterX then
37      enemyX = enemyX - 50 * dt
38    elseif enemyX < characterX then
39      enemyX = enemyX + 50 * dt
40    end
41
42    if enemyY > characterY then
43      enemyY = enemyY - 50 * dt
44    elseif enemyY < characterY then
45      enemyY = enemyY + 50 * dt
46    end
47  end
48
49  if checkCollision(characterX, characterY, character:getWidth(), character:getHeight(),
enemyX, enemyY, enemy:getWidth(), enemy:getHeight()) then
50    love.load()
51  end
52
53  if checkCollision(characterX, characterY, character:getWidth(), character:getHeight(),
bonusX, bonusY, bonus:getWidth(), bonus:getHeight()) then
54    bonusCollected = true
55  end
56
57

```

```

58  function love.draw()
59    love.graphics.draw(background, 0, 0)
60    love.graphics.draw(enemy, enemyX, enemyY)
61    if not bonusCollected then
62      love.graphics.draw(bonus, bonusX, bonusY)
63    else
64      love.graphics.setColor(0,0,0)
65      love.graphics.rectangle("fill",0,0,love.graphics.getWidth(),love.graphics.getHeight())
66      love.graphics.setColor(1,0,0)
67      love.graphics.print("You win!", love.graphics.getWidth()/2-40, love.graphics.getHeight()/
2-10)
68      love.graphics.setColor(1,1,1)
69    end
70    love.graphics.draw(character, characterX, characterY)
71  end
72

```

Dans cette étape, nous avons modifié `love.update()` et ajouté une condition à la fonction `love.draw()` pour afficher le message "You win!" si la variable `bonusCollected` est true. Nous avons utilisé la fonction `love.graphics.print()` pour dessiner le texte centré à l'écran.



Ajouter de la musique

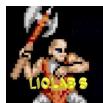
Maintenant que nous avons ajouté des objets bonus et un message de victoire, nous allons ajouter de la musique pour ajouter une ambiance sonore à notre jeu.

Ajoutez un nouveau fichier audio dans le dossier assets appelé "music.mp3". Téléchargez ou créez votre propre fichier audio pour cette étape.

Ajoutez le code suivant à votre fichier main.lua :

```
8  function love.load()
9      background = love.graphics.newImage("assets/background.png")
10     character = love.graphics.newImage("assets/character.png")
11     enemy = love.graphics.newImage("assets/enemy.png")
12     bonus = love.graphics.newImage("assets/bonus.png")
13     characterX = 100
14     characterY = 100
15     enemyX = 500
16     enemyY = 300
17     bonusX = 200
18     bonusY = 400
19     bonusCollected = false
20     music = love.audio.newSource("assets/music.mp3", "stream")
21     music:setLooping(true)
22     music:play()
23 end
24
```

Dans cette étape, nous avons ajouté une nouvelle variable music qui est initialisée avec le fichier audio "music.mp3". Nous avons également utilisé la fonction love.audio.newSource() pour créer un objet de source audio et nous avons utilisé les fonctions setLooping() et play() pour faire jouer la musique en boucle.



Ajouter des effets sonores

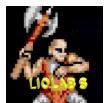
Maintenant que nous avons ajouté de la musique, nous allons ajouter des effets sonores pour ajouter des sons à notre jeu.

Ajoutez deux nouveaux fichiers audios dans le dossier assets appelés "collect.wav" et "hit.wav". Téléchargez ou créez vos propres fichiers audios pour cette étape.

Ajoutez le code suivant à votre fichier main.lua :

```
8  function love.load()
9      background = love.graphics.newImage("assets/background.png")
10     character = love.graphics.newImage("assets/character.png")
11     enemy = love.graphics.newImage("assets/enemy.png")
12     bonus = love.graphics.newImage("assets/bonus.png")
13     characterX = 100
14     characterY = 100
15     enemyX = 500
16     enemyY = 300
17     bonusX = 200
18     bonusY = 400
19     bonusCollected = false
20     music = love.audio.newSource("assets/music.mp3", "stream")
21     music:setLooping(true)
22     music:play()
23     collectSound = love.audio.newSource("assets/collect.wav", "static")
24     hitSound = love.audio.newSource("assets/hit.wav", "static")
25
26 end
```

Dans cette étape, nous avons créé deux nouvelles variables collectSound et hitSound qui sont initialisées avec les fichiers audio "collect.wav" et "hit.wav". Nous avons également utilisé la fonction love.audio.newSource() avec le deuxième argument "static" pour créer des objets de source audio qui ne sont pas en streaming.



: Liolabs : Liolabs

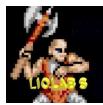
Jouer des effets sonores

Maintenant que nous avons ajouté des effets sonores, nous allons les jouer lorsque le joueur collecte un objet bonus ou touche l'ennemi.

Ajoutez le code suivant à votre fichier main.lua :

```
54  if checkCollision(characterX, characterY, character:getWidth(), character:getHeight(),
55      enemyX, enemyY, enemy:getWidth(), enemy:getHeight()) then
56          love.load()
57          music:stop()
58          hitSound:play()
59      end
60
60  if checkCollision(characterX, characterY, character:getWidth(), character:getHeight(),
61      bonusX, bonusY, bonus:getWidth(), bonus:getHeight()) then
62      bonusCollected = true
63      music:stop()
64      collectSound:play()
65  end
66
```

Dans cette étape, nous avons ajouté du code à la fonction `love.update()` pour jouer les effets sonores `collectSound` et `hitSound` lorsque le joueur collecte un objet bonus ou touche l'ennemi. Nous avons utilisé la fonction `checkCollision()` pour détecter les collisions entre les objets.

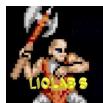


Ajouter un score

Maintenant que nous avons ajouté des effets sonores, nous allons ajouter un score pour suivre combien d'objets bonus le joueur a collecté.

Ajoutez le code suivant à votre fichier main.lua :

```
8  function love.load()
9      background = love.graphics.newImage("assets/background.png")
10     character = love.graphics.newImage("assets/character.png")
11     enemy = love.graphics.newImage("assets/enemy.png")
12     bonus = love.graphics.newImage("assets/bonus.png")
13     characterX = 100
14     characterY = 100
15     enemyX = 500
16     enemyY = 300
17     bonusX = 200
18     bonusY = 400
19     bonusCollected = false
20     music = love.audio.newSource("assets/music.mp3", "stream")
21     music:setLooping(true)
22     music:play()
23     collectSound = love.audio.newSource("assets/collect.mp3", "static")
24     hitSound = love.audio.newSource("assets/hit.mp3", "static")
25     score = 0
26     scoreFont = love.graphics.newFont(24)
27 end
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70  function love.draw()
71      love.graphics.draw(background, 0, 0)
72      love.graphics.draw(enemy, enemyX, enemyY)
73
74
75      if not bonusCollected then
76          love.graphics.draw(bonus, bonusX, bonusY)
77
78      else
79          love.graphics.setColor(0,0,0)
80          love.graphics.rectangle("fill",0,0,love.graphics:getWidth(),love.graphics:getHeight())
81          love.graphics.setColor(1,0,0)
82          love.graphics.setNewFont(40)
83          love.graphics.print("You win!", love.graphics:getWidth()/2-40, love.graphics:getHeight()/
2-10)
84          love.graphics.setColor(1,1,1)
85      end
86      love.graphics.draw(character, characterX, characterY)
87      love.graphics.setFont(scoreFont)
88      love.graphics.print("Score: " .. score, 10, 400)
89 end
90
```



```

62   if checkCollision(characterX, characterY, character:getWidth(), character:getHeight(),
63     bonusX, bonusY, bonus:getWidth(), bonus:getHeight()) then
64     bonusCollected = true
65     music:stop()
66     collectSound:play()
67     score = score + 10
68   end
69 end

```

Dans cette étape, nous avons créé deux nouvelles variables `score` et `scoreFont` pour suivre le score du joueur et afficher le score à l'écran. Nous avons également ajouté du code à la fonction `love.draw()` pour afficher le score à l'écran.

Dans la fonction `love.update()`, nous avons augmenté le score de 1 chaque fois que le joueur collecte un objet bonus. Le score est affiché à l'écran avec la fonction `love.graphics.print()`.

Ajouter un écran de fin de jeu

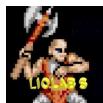
Maintenant que nous avons ajouté un score, nous allons ajouter un écran de fin de jeu qui s'affiche lorsque le joueur est touché par l'ennemi.

Ajoutez le code suivant à votre fichier `main.lua` :

```

8  function love.load()
9    background = love.graphics.newImage("assets/background.png")
10   character = love.graphics.newImage("assets/character.png")
11   enemy = love.graphics.newImage("assets/enemy.png")
12   bonus = love.graphics.newImage("assets/bonus.png")
13   characterX = 100
14   characterY = 100
15   enemyX = 500
16   enemyY = 300
17   bonusX = 200
18   bonusY = 400
19   bonusCollected = false
20   music = love.audio.newSource("assets/music.mp3", "stream")
21   music:setLooping(true)
22   music:play()
23   collectSound = love.audio.newSource("assets/collect.mp3", "static")
24   hitSound = love.audio.newSource("assets/hit.mp3", "static")
25   score = 0
26   scoreFont = love.graphics.newFont(24)
27   gameOverFont = love.graphics.newFont(36)
28   gameOver = false
29 end
30

```



```

72  function love.draw()
73      love.graphics.draw(background, 0, 0)
74      love.graphics.draw(enemy, enemyX, enemyY)
75
76
77  if not bonusCollected then
78      love.graphics.draw(bonus, bonusX, bonusY)
79
80  else
81      love.graphics.setColor(0,0,0)
82      love.graphics.rectangle("fill",0,0,love.graphics.getWidth(),love.graphics.getHeight())
83      love.graphics.setColor(1,0,0)
84      love.graphics.setNewFont(40)
85      love.graphics.print("You win!", love.graphics.getWidth()/2-40, love.graphics.getHeight()/
2-10)
86      love.graphics.setColor(1,1,1)
87  end
88  love.graphics.draw(character, characterX, characterY)
89  love.graphics.setFont(scoreFont)
90  love.graphics.print("Score: " .. score, 10, 400)
91
92  if gameOver then
93      love.graphics.setFont(gameOverFont)
94      love.graphics.print("Game Over", 250, 200)
95      love.graphics.print("Press R to restart", 200, 250)
96  end
97
98 end
99

```

```

31  function love.update(dt)
32
33  if love.keyboard.isDown("right") then
34      characterX = characterX + 100 * dt
35  elseif love.keyboard.isDown("left") then
36      characterX = characterX - 100 * dt
37  end
38
39  if love.keyboard.isDown("down") then
40      characterY = characterY + 100 * dt
41  elseif love.keyboard.isDown("up") then
42      characterY = characterY - 100 * dt
43  end
44
45  if bonusCollected == false then
46      if enemyX > characterX then
47          enemyX = enemyX - 50 * dt
48      elseif enemyX < characterX then
49          enemyX = enemyX + 50 * dt
50      end
51
52      if enemyY > characterY then
53          enemyY = enemyY - 50 * dt
54      elseif enemyY < characterY then
55          enemyY = enemyY + 50 * dt
56      end
57  end
58

```



```

59   if checkCollision(characterX, characterY, character:getWidth(), character:getHeight(),
60     enemyX, enemyY, enemy:getWidth(), enemy:getHeight()) then
61     music:stop()
62     hitSound:play()
63     gameOver = true
64   end
65   if checkCollision(characterX, characterY, character:getWidth(), character:getHeight(),
66     bonusX, bonusY, bonus:getWidth(), bonus:getHeight()) then
67     bonusCollected = true
68     music:stop()
69     collectSound:play()
70     score = score + 10
71   end
72   if gameOver and love.keyboard.isDown("r") then
73     love.load()
74   end
75 end
76
77

```

Dans cette étape, nous avons ajouté une nouvelle variable gameOver et une nouvelle police de caractères gameOverFont pour afficher l'écran de fin de jeu. Dans la fonction love.draw(), nous avons ajouté du code pour afficher l'écran de fin de jeu lorsque la variable gameOver est définie sur true.

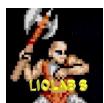
Dans la fonction love.update(), nous avons ajouté du code pour définir la variable gameOver sur true lorsque le joueur est touché par l'ennemi. Nous avons également ajouté du code pour redémarrer le jeu lorsque le joueur appuie sur la touche "r" après la fin de partie.

Félicitations, vous avez maintenant créé un jeu complet en utilisant Love2D ! N'hésitez pas à expérimenter avec le code et ajouter des fonctionnalités supplémentaires pour améliorer le jeu d'autant que notre jeu contient plusieurs bugs que je vous laisserai corriger à votre aise chez vous...

Ceci clôture, le tutoriel de base. Grâce à lui, nous avons les outils nécessaires pour démarrer notre projet de jeux de plateforme.

Il est important de noter que ce guide est purement pédagogique et n'a pour but que de vous apprendre les notions de base du framework Love2D.

Nous verrons durant ce stage, beaucoup de notions complémentaires comme l'animation des personnages, l'affichage d'un tilemap, la notion de camera, et



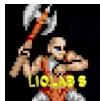
bien d'autres effets graphiques amusant pour que vous puissiez rendre vos futurs jeux des plus amusants et graphiquement sublime....

Bon courage les Codeurs Commandos.

Remerciements :

Liolabs tiens particulièrement à remercier la maison des jeunes de Dour sans qui ce stage n'aurait pas avoir eu lieu.

Si d'aventure vous souhaitiez participer à d'autres stages de programmations de jeux Vidéos. N'hésitez pas à en faire part à la maison des jeunes ou en me laissant un petit message sur l'email : liolabs1973@gmail.com



: Liolabs : Liolabs