



- **PROJET REALISE PAR :**

MOHCINE BOUDENJAL | YASSINE ANNAIMI

- **ENCADRE PAR :**

Ikram Ben ABDELOUAHAB | Lotfi ELAACHAK

- **L'OBJECTIF DE PROJET :**

L'objectif principal de ce projet (ROLLER SPLAT) est de maîtriser la programmation orientée objet par la mise en place d'un jeu vidéo 2D



Les outils et logiciels utilisés :



Lien GitHub: <https://github.com/boudenjal-mohcine/RollerSplat>

SOMMAIRE

- PRESENTATION D'ENVIRONNEMENT
- OBJECTIF DU PROJET.....
- LA CONCEPTION D'APPLICATION
- 1. CODE SOURCE.....
- 2. INTERFACE GRAPHIQUE.....
- CONCLUSION

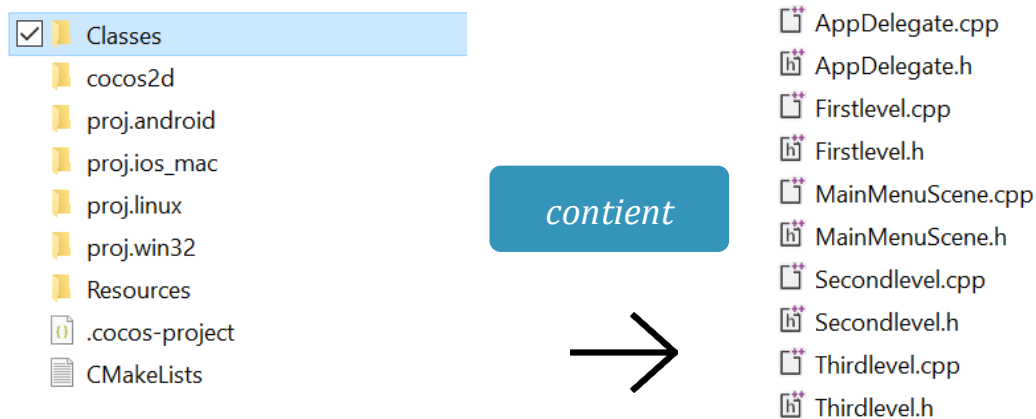
PRESENTATION D'ENVIRONNEMENT :

Cocos 2d permettant de développer des applications ou des jeux vidéo gratuitement.

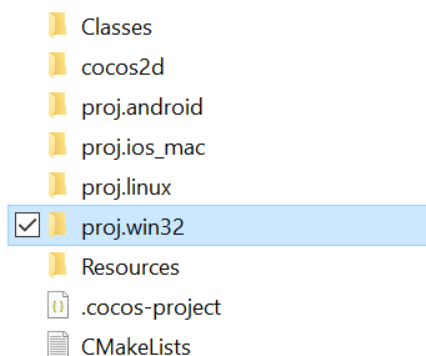
Cocos2d-x est un moteur de jeux très populaire au monde du développeurs, il utilise principalement deux langages de programmations (C++ et LUA)

Comme on utilise aussi le " Visual studio " comme un compilateur et débogueur de code pour lancer la visualisation des scènes développés avec cocos 2dx

- ***Pour lancer le jeux (suivez-vous les étapes suivants)***



1



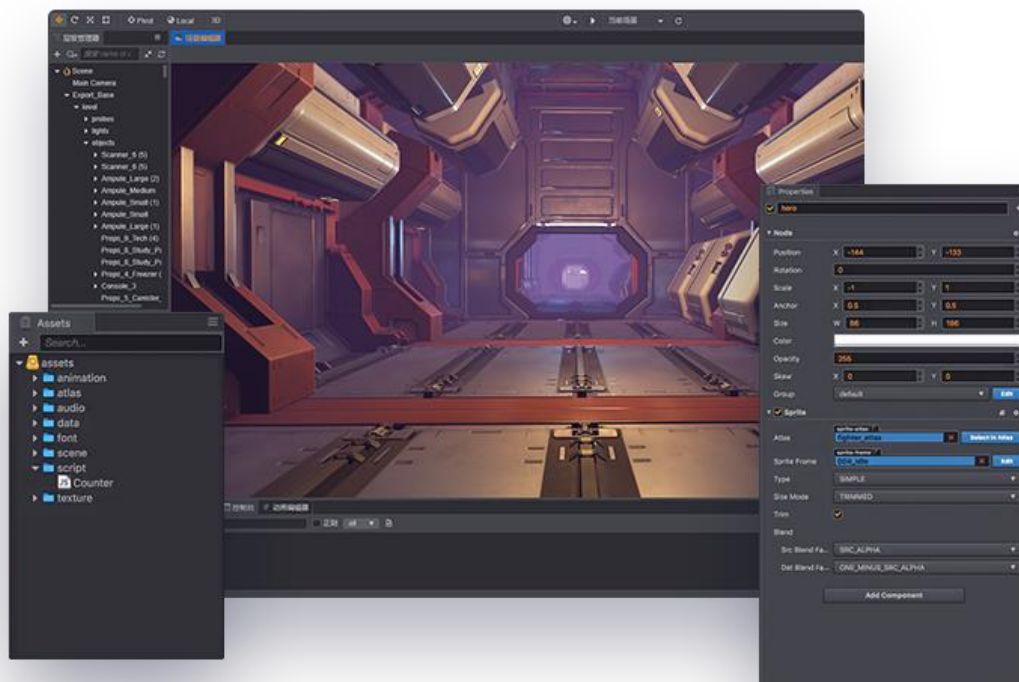
2

OBJECTIF DU PROJET :

L'objectif principal de ce projet (ROLLER SPLAT) est de maîtriser la programmation orientée objet par la mise en place d'un jeu vidéo 2D

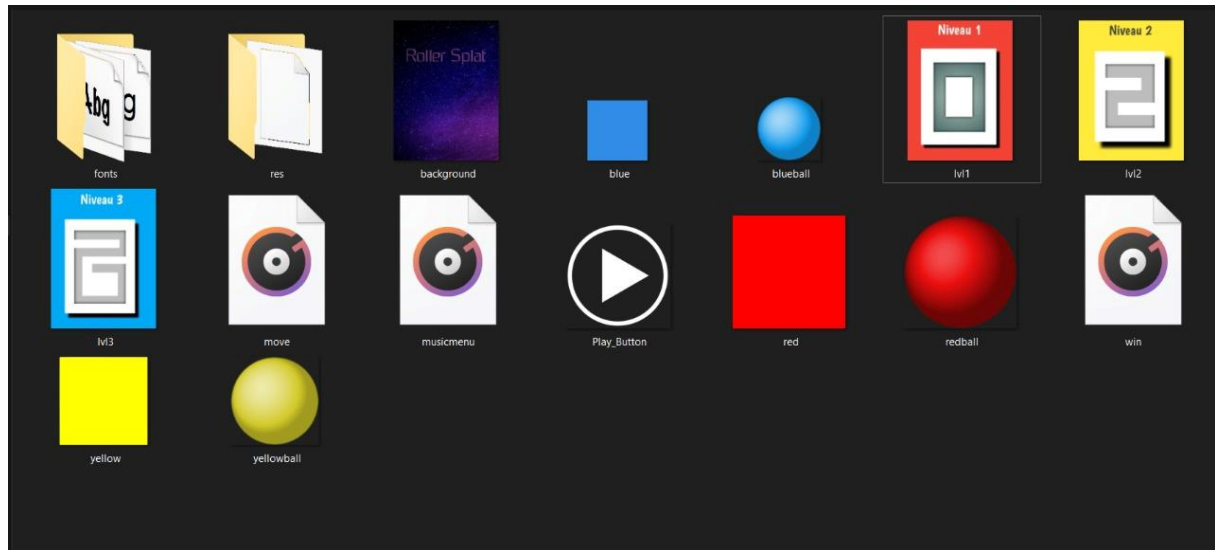
L'avantage de développer les **jeux vidéo** stimulent l'attention, la motivation, la concentration, la mémoire, la résolution de problèmes, la reconnaissance visuelle des personnages et des objets, la rapidité, un début de logique et une bonne coordination œil-main.

Thomas Guest : « Une sérieuse Game est un ensemble de mécanique de jeux qu'on assemble pour faire passer des messages, provoquer des émotions et changer les comportements des personnes ou d'un groupe de personnes de façon pérenne dans le temps.
















LA CONCEPTION D'APPLICATION :

Les ressources qu'on est utilisé durant la création de notre projet :



Notre projet contient les classes suivantes :

Nom	Modifié le	Type	Taille
 AppDelegate	05/01/2022 14:45	C++ source file	5 Ko
 AppDelegate	29/12/2021 21:33	Header file	1 Ko
 Definitions	05/01/2022 15:34	Header file	1 Ko
 Firstlevel	05/01/2022 15:36	C++ source file	8 Ko
 Firstlevel	05/01/2022 01:36	Header file	1 Ko
 MainMenuScene	05/01/2022 15:31	C++ source file	2 Ko
 MainMenuScene	05/01/2022 01:34	Header file	1 Ko
 Secondlevel	05/01/2022 16:42	C++ source file	7 Ko
 Secondlevel	05/01/2022 16:08	Header file	1 Ko
 SplashScreen	05/01/2022 15:16	C++ source file	1 Ko
 SplashScreen	05/01/2022 15:03	Header file	1 Ko
 Thirdlevel	05/01/2022 15:38	C++ source file	8 Ko
 Thirdlevel	05/01/2022 15:07	Header file	1 Ko

(Le code source) :

SplashScene.h

```
1  #ifndef __SPLASH_SCENE_H__
2  #define __SPLASH_SCENE_H__
3
4  #include "cocos2d.h"
5
6  class SplashScene : public cocos2d::Scene
7  {
8  public:
9      static cocos2d::Scene* createScene();
10
11      virtual bool init();
12
13      CREATE_FUNC(SplashScene);
14
15      void GoToMainMenu(float dt);
16
17 };
18
19 #endif
20
21
```

SplashScene.cpp

```
1  #include "MainMenuScene.h"
2  #include "SimpleAudioEngine.h"
3  #include "SplashScene.h"
4
5
6
7  USING_NS_CC;
8
9  Scene* SplashScene::createScene()
10 {
11     return SplashScene::create();
12 }
13
14
15
16 // on "init" you need to initialize your instance
17 bool SplashScene::init()
18 {
19     ///////////////////////////////////////////////////
20     // 1. super init first
21     if (!Scene::init())
22     {
23         return false;
24     }
25
26     auto visibleSize = Director::getInstance()->getVisibleSize();
27     Vec2 origin = Director::getInstance()->getVisibleOrigin();
28
29     this->scheduleOnce(schedule_selector(SplashScene::GoToMainMenu), 2.0);
30
31     auto background = Sprite::create("splash.png");
32     background->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
33     this->addChild(background, -1);
34
35     return true;
36 }
37
38
39 void SplashScene::GoToMainMenu(float dt)
40 {
41     //go to a MainPage
42     auto scene = MainMenu::createScene();
43     Director::getInstance()->pushScene(TransitionFade::create(2.0, scene));
44 }
45
46
```

SplashScene est créer une première scène contient un background avec nos nom et le logo de FSTT avant le lancement de game avec une dure de 8seconds et un effet nomme TransitionFade.

MainMenuScene.h

```
1  #ifndef __MAINMENU_SCENE_H__
2  #define __MAINMENU_SCENE_H__
3
4  #include "cocos2d.h"
5
6  class MainMenu : public cocos2d::Scene
7  {
8  public:
9      static cocos2d::Scene* createScene();
10
11      virtual bool init();
12
13      CREATE_FUNC(MainMenu);
14
15      void GoToLevel1(Ref *pSender);
16
17 };
18
19 #endif // __MAINMENU_SCENE_H__
20
21
```

```
1  #include "MainMenuScene.h"
2  #include "SimpleAudioEngine.h"
3  #include "Firstlevel.h"
4
5
6
7  USING_NS_CC;
8
9  Scene* MainMenu::createScene()
10 {
11     return MainMenu::create();
12 }
13
14
15 // on "init" you need to initialize your instance
16 bool MainMenu::init()
17 {
18     ////////////////
19     // 1. super init first
20     if ( !Scene::init() )
21     {
22         return false;
23     }
24
25     auto visibleSize = Director::getInstance()->getVisibleSize();
26     Vec2 origin = Director::getInstance()->getVisibleOrigin();
27
28     auto background = Sprite::create("background.png");
29     background->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
30     this->addChild(background, -1);
31
32     CocosDenshion::SimpleAudioEngine::getInstance()->preloadBackgroundMusic("musicmenu.mp3");
33     CocosDenshion::SimpleAudioEngine::getInstance()->playBackgroundMusic("musicmenu.mp3");
34
35
36     auto playItem =
37     MenuItemImage::create("Play_Button.png",
38                           "Play_Button.png",
39                           CC_CALLBACK_1(MainMenu::GoToLevel1, this));
40
41     auto menu = Menu::create(playItem, NULL);
42     menu->alignItemsVerticallyWithPadding(visibleSize.height);
43     this->addChild(menu);
44
45     return true;
46 }
47
48 void MainMenu::GoToLevel1(Ref* pSender)
49 {
50     //go to a level1
51     auto scene = level1::createScene();
52     Director::getInstance()->pushScene(TransitionFade::create(6.0, scene));
53 }
54
```


MainMenuScene créer une scène qui contient un background, le titre et bouton PLAY qui nous ramené au FirstLevel par CC_CALLBACK_1 qui fait appelle a la fonction GoToLevel1 , cette fonction qui crée une scène level1 et il le remplace avec un effet de transition.

firstlevel.h /secondlevel.h/thirdlevel.h

Tous les header des niveaux contient généralement la même chose ,

- La méthode init () qui initialise la scène
- Le prototype de la méthode Paint
- Les booliens isColored
- Les 3 sprites (background level, ball, color)

```
1  #ifndef __FIRSTLEVEL_H__
2  #define __FIRSTLEVEL_H__
3
4  #include "cocos2d.h"
5
6  using namespace cocos2d;
7
8  class level1 : public cocos2d::Scene
9  {
10 public:
11
12     static cocos2d::Scene* createScene();
13
14     virtual bool init();
15
16     void paint(EventKeyboard::KeyCode KeyCode, Event* event);
17
18     //variable depend du coloration des parties
19     bool isColored1 = false;
20     bool isColored2 = false;
21     bool isColored3 = false;
22     bool isColored4 = false;
23
24
25
26     // implement the "static create()" method manually
27     CREATE_FUNC(level1);
28
29     cocos2d::Sprite* Level1_background;
30     cocos2d::Sprite* ball1;
31     cocos2d::Sprite* color1;
32 };
33
34 #endif
35
```



```

1  #ifndef __SECONDLEVEL_H__
2  #define __SECONDLEVEL_H__
3
4  #include "cocos2d.h"
5
6  using namespace cocos2d;
7
8  class level2 : public cocos2d::Scene
9  {
10 public:
11
12     static cocos2d::Scene* createScene();
13
14     virtual bool init();
15
16     void paint(EventKeyboard::KeyCode KeyCode, Event* event);
17
18     bool isColored1 = false;
19     bool isColored2 = false;
20     bool isColored3 = false;
21     bool isColored4 = false;
22     bool isColored5 = false;
23
24     // implement the "static create()" method manually
25     CREATE_FUNC(level2);
26
27     cocos2d::Sprite* Level2_background;
28     cocos2d::Sprite* ball2;
29     cocos2d::Sprite* color2;
30
31 };
32
33 #endif
34
35
36

```

```

1  #ifndef __THIRDLEVEL_H__
2  #define __THIRDLEVEL_H__
3
4  #include "cocos2d.h"
5
6  using namespace cocos2d;
7
8  class level3 : public cocos2d::Scene
9  {
10 public:
11
12     static cocos2d::Scene* createScene();
13
14     virtual bool init();
15
16     void paint(EventKeyboard::KeyCode KeyCode, Event* event);
17
18     bool isColored1 = false;
19     bool isColored2 = false;
20     bool isColored3 = false;
21     bool isColored4 = false;
22     bool isColored5 = false;
23     bool isColored6 = false;
24     bool isColored7 = false;
25
26     // implement the "static create()" method manually
27     CREATE_FUNC(level3);
28
29     cocos2d::Sprite* Level3_background;
30     cocos2d::Sprite* ball3;
31     cocos2d::Sprite* color3;
32     cocos2d::DrawNode* color;
33
34 };
35
36 #endif
37
38
39
40

```

Firstlevel.cpp

```
1 #include "MainMenuScene.h"
2 #include "SimpleAudioEngine.h"
3 #include "Firstlevel.h"
4 #include "Secondlevel.h"
5
6 USING_NS_CC;
7
8 Scene* level1::createScene()
9 {
10     return level1::create();
11 }
12
13
14 // on "init" you need to initialize your instance
15 bool level1::init()
16 {
17     ///////////////////////////////////////////////////
18     // 1. super init first
19     if (!Scene::init())
20     {
21         return false;
22     }
23     //stopper la music du menu
24     CocosDenshion::SimpleAudioEngine::getInstance()->pauseBackgroundMusic();
25
26     //Create The Level ( Ball and Background )
27     Size visibleSize = Director::getInstance()->getVisibleSize();
28     Vec2 origin = Director::getInstance()->getVisibleOrigin();
29
30     //set background level
31     Level1_background = Sprite::create("lvl1.jpg");
32     Level1_background->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
33     this->addChild(Level1_background, -1);
34
35     //set ball in the level
36     ball1 = Sprite::create("redball.png");
37     ball1->setPosition(Point(190, 194));
38     this->addChild(ball1, 0);
39     ball1->setName("ball");
40
41     //Create an event when we press a key
42     auto Listener = EventListenerKeyboard::create();
43     Listener->onKeyPressed = CC_CALLBACK_2(level1::paint, this);
44     ball1->getEventDispatcher()->addEventListenerWithSceneGraphPriority(Listener, ball1);
45
46     return true;
47 }
48
49
```

```
50 //fonction qui traite le mouvement en chaque position , coulerer et partie au niveau suivant si on gangne :
51 void level1::paint(EventKeyboard::KeyCode KeyCode, Event* event) {
52
53     auto ballVect = ball1->getPosition();
54     //Les mouvements possible , il depend de x ou y .
55     auto actionMoveRight = MoveTo::create(0.09f, Point(410, ballVect.y));
56     auto actionMoveUp = MoveTo::create(0.09f, Point(ballVect.x, 524));
57     auto actionMoveLeft = MoveTo::create(0.09f, Point(190, ballVect.y));
58     auto actionMoveDown = MoveTo::create(0.09f, Point(ballVect.x, 194));
59
60     //faire appelle a chaque mouvement le son d'effet
61     CocosDenshion::SimpleAudioEngine::getInstance()->preloadEffect("move.mp3");
62
63     switch (KeyCode) {
64     case EventKeyboard::KeyCode::KEY_LEFT_ARROW:
65     case EventKeyboard::KeyCode::KEY_A:
66         if (ballVect == Point(410, 194)) {
67             //repositionnement du ball et faire des tests si la partie est deja coulerer ou non
68             ball1->runAction(actionMoveLeft);
69             if (!isColored1) {
70                 //si non on les coulerer
71                 for (int j = 410; j >= 190; j = j - 20) {
72                     auto colorred = Sprite::create("red.png");
73                     colorred->setPosition(Point(j, ballVect.y));
74                     this->addChild(colorred, -1);
75                     isColored1 = true;
76                 }
77             }
78             //allumer l'effet
79             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
80         }
81         if (ballVect == Point(410, 524)) {
82             ball1->runAction(actionMoveLeft);
83             if (!isColored3) {
84                 for (int j = 410; j >= 190; j = j - 20) {
85                     auto colorred = Sprite::create("red.png");
86                     colorred->setPosition(Point(j, ballVect.y));
87                     this->addChild(colorred, -1);
88                     isColored3 = true;
89                 }
90             }
91             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
92         }
93     }
94     break;
95 }
96
97
```

```

99 case EventKeyboard::KeyCode::KEY_RIGHT_ARROW:
100 case EventKeyboard::KeyCode::KEY_D:
101     if (ballVect == Point(190, 194)) {
102         ball1->runAction(actionMoveright);
103         if (!isColored1) {
104             for (int j = 190; j <= 410; j = j + 20) {
105                 auto coloredred = Sprite::create("red.png");
106                 coloredred->setPosition(Point(j, ballVect.y));
107                 this->addChild(coloredred, -1);
108                 isColored1 = true;
109             }
110         }
111         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
112     }
113 }
114 if (ballVect == Point(190, 524)) {
115     ball1->runAction(actionMoveright);
116     if (!isColored3) {
117         for (int j = 190; j <= 410; j = j + 20) {
118             auto coloredred = Sprite::create("red.png");
119             coloredred->setPosition(Point(j, ballVect.y));
120             this->addChild(coloredred, -1);
121             isColored3 = true;
122         }
123     }
124     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
125 }
126 }
127 break;
128
129 case EventKeyboard::KeyCode::KEY_UP_ARROW:
130 case EventKeyboard::KeyCode::KEY_W:
131     if (ballVect == Point(190, 194)) {
132         ball1->runAction(actionMoveup);
133         if (!isColored2) {
134             for (int j = 194; j <= 524; j = j + 30) {
135                 auto coloredred = Sprite::create("red.png");
136                 coloredred->setPosition(Point(ballVect.x, j));
137                 this->addChild(coloredred, -1);
138             }
139             isColored2 = true;
140         }
141         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
142     }
143 }

```

```

50
51 void level2::paint(EventKeyboard::KeyCode KeyCode, Event* event) {
52
53     //Positionnement actuelle du balle
54     auto ballVect = ball2->getPosition();
55
56     //les mouvements possible
57     auto actionMoveright = MoveTo::create(0.0f, Point(405, ballVect.y));
58     auto actionMoveup1 = MoveTo::create(0.0f, Point(ballVect.x, 362));
59     auto actionMoveup2 = MoveTo::create(0.0f, Point(ballVect.x, 515));
60     auto actionMoveleft = MoveTo::create(0.0f, Point(185, ballVect.y));
61     auto actionMovedown1 = MoveTo::create(0.0f, Point(ballVect.x, 362));
62     auto actionMovedown2 = MoveTo::create(0.0f, Point(ballVect.x, 210));
63
64     //Effet music
65     CocosDenshion::SimpleAudioEngine::getInstance()->preloadEffect("move.mp3");
66
67
68     //run action en fonction du positionnement et d'action correspondant
69     switch (KeyCode) {
70
71     case EventKeyboard::KeyCode::KEY_LEFT_ARROW:
72     case EventKeyboard::KeyCode::KEY_A:
73         if (ballVect == Point(405, 210)) {
74             ball2->runAction(actionMoveleft);
75             if (!isColored1) {
76                 for (int j = 405; j >= 183; j = j - 37) {
77                     auto coloryellow = Sprite::create("yellow.png");
78                     coloryellow->setPosition(Point(j, ballVect.y+5));
79                     this->addChild(coloryellow, -1);
80                     isColored1 = true;
81                 }
82             }
83             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
84         }
85     }
86
87     if (ballVect == Point(405, 362)) {
88         ball2->runAction(actionMoveleft);
89         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
90     }
91
92     if (ballVect == Point(405, 515)) {
93         ball2->runAction(actionMoveleft);
94         if (!isColored5) {
95             for (int j = 400; j >= 185; j = j - 5) {
96                 auto coloryellow = Sprite::create("yellow.png");
97                 coloryellow->setPosition(Point(j, ballVect.y));
98                 this->addChild(coloryellow, -1);
99                 isColored5 = true;
100             }
101         }
102         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
103     }
104     break;
105 }

```



```

193
194 //tester si tout les partie est coulerer
195 if (isColored1 && isColored2 && isColored3 && isColored4)
196 {
197     //l'effet du win
198     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("win.mp3");
199     CocosDenshion::SimpleAudioEngine::getInstance()->preloadEffect("win.mp3");
200
201     //la balle jump
202     auto actionWIN = JumpBy::create(8, Point(0, 0), 80, 20);
203     ball1->runAction(actionWIN);
204
205     //allez au niveau prochain
206     auto scene = level2::createScene();
207     Director::getInstance()->pushScene(TransitionFade::create(8.0, scene));
208
209 }
210
211

```

Second level.cpp

```

1  #include "MainMenuScene.h"
2  #include "SimpleAudioEngine.h"
3  #include "Secondlevel.h"
4  #include "Thirdlevel.h"
5
6
7
8  USING_NS_CC;
9
10 Scene* level2::createScene()
11 {
12     return level2::create();
13 }
14
15
16
17 // on "init" you need to initialize your instance
18 bool level2::init()
19 {
20     //////////////////////////////////////
21     // 1. super init first
22     if (!Scene::init())
23     {
24         return false;
25     }
26
27     Size visibleSize = Director::getInstance()->getVisibleSize();
28     Vec2 origin = Director::getInstance()->getVisibleOrigin();
29
30     //set background of level2 from Resources
31     Level2_background = Sprite::create("lvl2.png");
32     Level2_background->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
33     this->addChild(Level2_background, -1);
34
35     //set yellowbal from Resources
36     ball2 = Sprite::create("yellowball.png");
37     ball2->setPosition(Point(405, 210));
38     this->addChild(ball2, 1);
39
40     //Create an event when we press a key
41     auto Listener = EventListenerKeyboard::create();
42     Listener->onKeyPressed = CC_CALLBACK_2(level2::paint, this);
43     ball2->getEventDispatcher()->addEventListenerWithSceneGraphPriority(Listener, ball2);
44
45     return true;
46 }
47
48
49

```

```

50
51 void level2::paint(EventKeyboard::KeyCode KeyCode, Event* event) {
52
53     //Positionnement actuelle du balle
54     auto ballVect = ball2->getPosition();
55
56     //les mouvements possible
57     auto actionMoveright = MoveTo::create(0.0f, Point(405, ballVect.y));
58     auto actionMoveup1 = MoveTo::create(0.0f, Point(ballVect.x, 362));
59     auto actionMoveup2 = MoveTo::create(0.0f, Point(ballVect.x, 515));
60     auto actionMoveleft = MoveTo::create(0.0f, Point(185, ballVect.y));
61     auto actionMovedown1 = MoveTo::create(0.0f, Point(ballVect.x, 362));
62     auto actionMovedown2 = MoveTo::create(0.0f, Point(ballVect.x, 210));
63
64     //Effet music
65     CocosDenshion::SimpleAudioEngine::getInstance()->preloadEffect("move.mp3");
66
67
68     //run action en fonction du positionnement et d'action correspondant
69     switch (KeyCode) {
70
71     case EventKeyboard::KeyCode::KEY_LEFT_ARROW:
72     case EventKeyboard::KeyCode::KEY_A:
73         if (ballVect == Point(405, 210)) {
74             ball2->runAction(actionMoveleft);
75             if (!isColored1) {
76                 for (int j = 405; j >= 183; j = j - 37) {
77                     auto coloryellow = Sprite::create("yellow.png");
78                     coloryellow->setPosition(Point(j, ballVect.y+5));
79                     this->addChild(coloryellow, -1);
80                     isColored1 = true;
81                 }
82             }
83             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
84         }
85
86         if (ballVect == Point(405, 362)) {
87             ball2->runAction(actionMoveleft);
88             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
89         }
90
91         if (ballVect == Point(405, 515)) {
92             ball2->runAction(actionMoveleft);
93             if (!isColored5) {
94                 for (int j = 400; j >= 185; j = j - 5) {
95                     auto coloryellow = Sprite::create("yellow.png");
96                     coloryellow->setPosition(Point(j, ballVect.y));
97                     this->addChild(coloryellow, -1);
98                     isColored5 = true;
99                 }
100             }
101             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
102         }
103     }
104     break;
105

```

```

106 case EventKeyboard::KeyCode::KEY_RIGHT_ARROW:
107 case EventKeyboard::KeyCode::KEY_D:
108
109     if (ballVect == Point(185, 362)) {
110         ball2->runAction(actionMoveright);
111         if (!isColored3) {
112             for (int j = 185; j <= 400; j = j + 1) {
113                 auto coloryellow = Sprite::create("yellow.png");
114                 coloryellow->setPosition(Point(j, ballVect.y+2));
115                 this->addChild(coloryellow, -1);
116                 isColored3 = true;
117             }
118         }
119         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
120     }
121
122     if (ballVect == Point(185, 210)) {
123         ball2->runAction(actionMoveright);
124         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
125     }
126
127     if (ballVect == Point(185, 515)) {
128         ball2->runAction(actionMoveright);
129         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
130     }
131     break;
132
133 case EventKeyboard::KeyCode::KEY_UP_ARROW:
134 case EventKeyboard::KeyCode::KEY_W:
135
136     if (ballVect == Point(185, 210)) {
137         ball2->runAction(actionMoveup1);
138         if (!isColored2) {
139             for (int j = 220; j <= 362; j++) {
140                 auto coloryellow = Sprite::create("yellow.png");
141                 coloryellow->setPosition(Point(ballVect.x-1, j));
142                 this->addChild(coloryellow, -1);
143             }
144             isColored2 = true;
145         }
146     }
147
148     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
149 }
150
151
152 if (ballVect == Point(405, 362)) {
153
154     ball2->runAction(actionMoveup2);
155     if (!isColored4) {
156         for (int j = 365; j <= 520; j = j + 10) {
157             auto coloryellow = Sprite::create("yellow.png");
158             coloryellow->setPosition(Point(ballVect.x-3, j));
159             this->addChild(coloryellow, -1);
160         }
161         isColored4 = true;
162     }
163     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");

```

```

151
152     if (ballVect == Point(405, 362)) {
153
154         ball2->runAction(actionMoveup2);
155         if (!isColored4) {
156             for (int j = 365; j <= 520; j = j + 10) {
157                 auto coloryellow = Sprite::create("yellow.png");
158                 coloryellow->setPosition(Point(ballVect.x-3, j));
159                 this->addChild(coloryellow, -1);
160             }
161             isColored4 = true;
162         }
163         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
164     }
165     break;
166
167 case EventKeyboard::KeyCode::KEY_DOWN_ARROW:
168 case EventKeyboard::KeyCode::KEY_S:
169
170     if (ballVect == Point(405, 515)) {
171
172         ball2->runAction(actionMovedown1);
173         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
174     }
175
176     if (ballVect == Point(185, 362)) {
177         ball2->runAction(actionMovedown2);
178         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
179     }
180
181     break;
182
183
184
185
186
187 }
188 //tester si tout les partie est coulerer
189 if (isColored1 && isColored2 && isColored3 && isColored4 && isColored5)
190 {
191     //l'effet du win
192     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("win.mp3");
193     CocosDenshion::SimpleAudioEngine::getInstance()->preloadEffect("win.mp3");
194
195     //ball jump
196     auto actionWIN = JumpBy::create(8, Point(0, 0), 80, 20);
197     ball2->runAction(actionWIN);
198
199     //go to level3
200     auto scene = level3::createScene();
201     Director::getInstance()->pushScene(TransitionFade::create(8.0, scene));
202 }
203
204

```


Third level.cpp

```
1  #include "MainMenuScene.h"
2  #include "SimpleAudioEngine.h"
3  #include "Thirdlevel.h"
4
5  USING_NS_CC;
6
7  Scene* Level3::createScene()
8  {
9      return Level3::create();
10 }
11
12
13
14 // on "init" you need to initialize your instance
15 bool Level3::init()
16 {
17     ///////////////////////////////////
18     // 1. super init first
19     if (!Scene::init())
20     {
21         return false;
22     }
23
24     Size visibleSize = Director::getInstance()->getVisibleSize();
25     Vec2 origin = Director::getInstance()->getVisibleOrigin();
26
27     //set level3 background
28     Level3_background = Sprite::create("lvl3.jpeg");
29     Level3_background->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
30     this->addChild(Level3_background, -1);
31
32     //set blue ball of level3
33     ball3 = Sprite::create("blueball.png");
34     ball3->setPosition(Point(260,340));
35     this->addChild(ball3, 0);
36
37     //le capteur du touche et qui fait appelle au fonction paint
38     //Create an event when we press a key
39     auto listener = EventListenerKeyboard::create();
40     listener->onKeyPressed = CC_CALLBACK_2(Level3::paint, this);
41     ball3->getEventDispatcher()->addEventListenerWithSceneGraphPriority(listener, ball3);
42
43
44
45     return true;
46 }
47
48
```

```

49
50 void level3::paint(EventKeyboard::KeyCode KeyCode, Event* event) {
51
52     //Positionnement du balle et les mouvements possibles
53     auto ballVect = ball3->getPosition();
54     auto actionMoveright = MoveTo::create(0.0f, Point(410, ballVect.y));
55     auto actionMoveup1 = MoveTo::create(0.0f, Point(ballVect.x, 437));
56     auto actionMoveup2 = MoveTo::create(0.0f, Point(ballVect.x, 539));
57     auto actionMoveup3 = MoveTo::create(0.0f, Point(ballVect.x, 340));
58     auto actionMoveleft = MoveTo::create(0.0f, Point(150, ballVect.y));
59     auto actionMoveleft2 = MoveTo::create(0.0f, Point(260, ballVect.y));
60     auto actionMovedown = MoveTo::create(0.0f, Point(ballVect.x, 155));
61     auto actionMovedown2 = MoveTo::create(0.0f, Point(ballVect.x, 210));
62
63
64     //Effet music
65     CocosDenshion::SimpleAudioEngine::getInstance()->preloadEffect("move.mp3");
66
67     //En fonction du position , RunAction correspondant
68     switch (KeyCode) {
69
70
71     case EventKeyboard::KeyCode::KEY_RIGHT_ARROW:
72     case EventKeyboard::KeyCode::KEY_D:
73         if (ballVect == Point(260, 340)) {
74             ball3->runAction(actionMoveright);
75             if (!isColored1) {
76                 for (int i = 260; i < 415; i = i + 5) {
77                     color3 = Sprite::create("blue.jpg");
78                     color3->setPosition(Point(i, ballVect.y));
79                     this->addChild(color3, -1);
80                 }
81                 isColored1 = true;
82             }
83             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
84         }
85
86
87         if (ballVect == Point(150, 437)) {
88             ball3->runAction(actionMoveright);
89             if (!isColored5) {
90                 for (int i = 150; i < 420; i = i + 20) {
91                     color3 = Sprite::create("blue.jpg");
92                     color3->setPosition(Point(i, ballVect.y));
93                     this->addChild(color3, -1);
94                 }
95                 isColored5 = true;
96             }
97             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
98         }
99     }

```

```

100
101     if (ballVect == Point(150, 155)) {
102         ball3->runAction(actionMoveRight);
103         CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
104     }
105
106
107     break;
108
109     case EventKeyboard::KeyCode::KEY_DOWN_ARROW:
110     case EventKeyboard::KeyCode::KEY_S:
111
112         if (ballVect == Point(410, 340)) {
113             ball3->runAction(actionMovedown);
114             if (!isColored2) {
115                 for (int i = 340; i >= 155; i = i - 5) {
116                     color3 = Sprite::create("blue.jpg");
117                     color3->setPosition(Point(ballVect.x, i));
118                     this->addChild(color3, -1);
119                 }
120                 isColored2 = true;
121             }
122             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
123         }
124
125
126         if (ballVect == Point(410, 539)) {
127             ball3->runAction(actionMoveup1);
128             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
129         }
130
131
132         if (ballVect == Point(150, 437)) {
133             ball3->runAction(actionMovedown);
134             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
135         }
136
137     }
138
139     break;
140
141     case EventKeyboard::KeyCode::KEY_LEFT_ARROW:
142     case EventKeyboard::KeyCode::KEY_A:
143
144         if (ballVect == Point(410, 155)) {
145             ball3->runAction(actionMoveLeft);
146             if (!isColored3) {
147                 for (int i = 410; i > 150; i = i - 5) {
148                     color3 = Sprite::create("blue1.jpg");
149                     color3->setPosition(Point(i, ballVect.y));
150                     this->addChild(color3, -1);
151                 }
152                 isColored3 = true;
153             }
154             CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
155         }

```

```

155
156 if (ballVect == Point(410, 539)) {
157     ball3->runAction(actionMoveleft);
158     if (!isColored7) {
159         for (int i = 410; i > 150; i = i - 10) {
160             color3 = Sprite::create("blue1.jpg");
161             color3->setPosition(Point(i, ballVect.y));
162             this->addChild(color3, -1);
163         }
164         isColored7 = true;
165     }
166     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
167 }
168
169 if (ballVect == Point(410, 437)) {
170     ball3->runAction(actionMoveleft);
171     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
172 }
173
174
175 if (ballVect == Point(410, 340)) {
176     ball3->runAction(actionMoveleft2);
177     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
178 }
179
180 }
181
182 break;
183
184 case EventKeyboard::KeyCode::KEY_UP_ARROW:
185 case EventKeyboard::KeyCode::KEY_W:
186
187 if (ballVect == Point(150, 155)) {
188     ball3->runAction(actionMoveup1);
189     if (!isColored6) {
190         for (int i = 155; i < 440; i = i + 5) {
191             color3 = Sprite::create("blue.jpg");
192             color3->setPosition(Point(ballVect.x, i));
193             this->addChild(color3, -1);
194         }
195         isColored6 = true;
196     }
197     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
198 }
199
200 }

```

```

200
201 if (ballVect == Point(410, 437)) {
202     ball3->runAction(actionMoveup2);
203     if (!isColored4) {
204         for (int i = 437; i <= 547; i = i + 20) {
205             color3 = Sprite::create("blue.jpg");
206             color3->setPosition(Point(ballVect.x, i));
207             this->addChild(color3, -1);
208         }
209         isColored4 = true;
210     }
211     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
212 }
213
214
215 if (ballVect == Point(410, 155)) {
216     ball3->runAction(actionMoveup3);
217     CocosDenshion::SimpleAudioEngine::getInstance()->playEffect("move.mp3");
218 }
219
220 }
221 break;
222
223
224 //verifier si tout les partie sont colorer
225
226 if (isColored1 && isColored2 && isColored3 && isColored4 && isColored5 && isColored6 && isColored7)
227 {
228
229     //ball jump
230     auto actionJUMP = JumpBy::create(8, Point(0, 0), 80, 20);
231     ball3->runAction(actionJUMP);
232
233     //Retourner a la page menu
234     auto scene = MainMenu::createScene();
235     Director::getInstance()->pushScene(TransitionFade::create(8.0, scene));
236 }
237
238 }

```

FirstLevel.cpp/SecondLevel.cpp/ThirdLevel.cpp

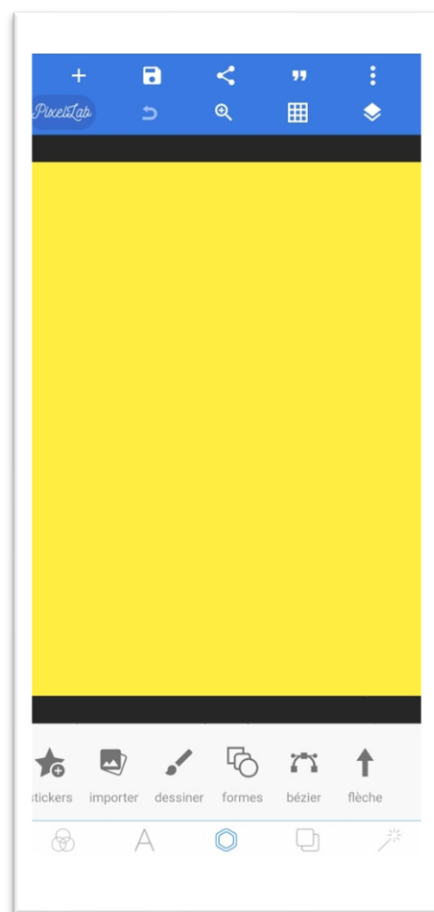
- Au début on appelle la méthode createScene () qui va créer une nouvelle scène .
- La fonction init() teste si la scène est créée si oui il va ajouter les sprites qui ont déjà été déclarés dans les classes header de chaque niveau , on les positionne par la méthode setPosition et avec addChild on les ajoute dans la scène.
- On déclare un détecteur « Listener » des touches de clavier qui fait appel à la fonction Paint avec CC_CALLBACK_2.
- Cette méthode Paint prend comme argument la touche cliquée, il prend le positionnement actuel de la Ball avec la fonction getPosition () et il cite les mouvements possibles en fonction de son positionnement.
- On a divisé la partie non colorée en plusieurs parties par exemple dans le niveau 1 on a 4 parties et chaque partie on lui donne un variable booléen isColored initialiser par false c'est-à-dire non coloré . la fonction Paint va appliquer l'action du mouvement correspondant par la méthode runAction et il va tester si la partie parcourue colore ou non si non il va entrer la boucle for qui parcourue aussi la même partie en ajoutant un Sprite de couleur puis il donne au booléen correspondant à la partie la valeur TRUE .
- Si toutes les parties sont colorées c'est-à-dire leurs booléens sont TRUE , la Ball saute à sa place avec la class JumpBy , on va lancer une musique avec la classe SimpleAudioEngine qui appartient à la bibliothèque SimpleAudioEngine.h , il va créer une scène de niveau suivante d'une autre façon tu es gagnier le niveau.
- Et on va passer au niveau suivant par la méthode pushScene ().

INTERFACE GRAPHIQUE :

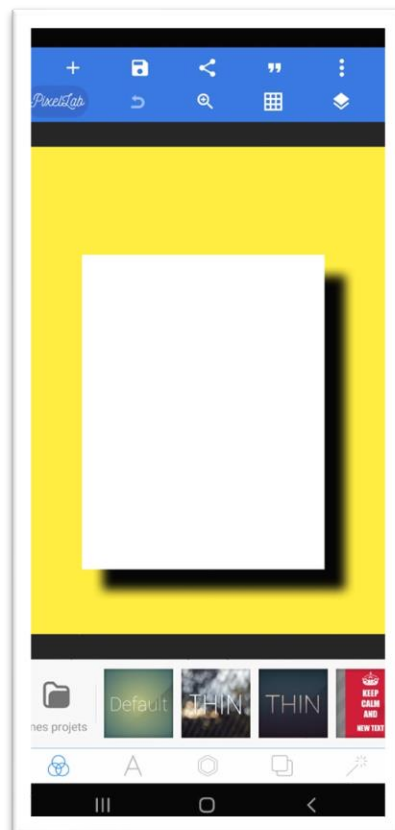
La construction des Level background fait avec l'application PixelLab , application mobile qui facilite les choses .

Premièrement on choisit La taille de l'image, dans notre cas 1600x1200

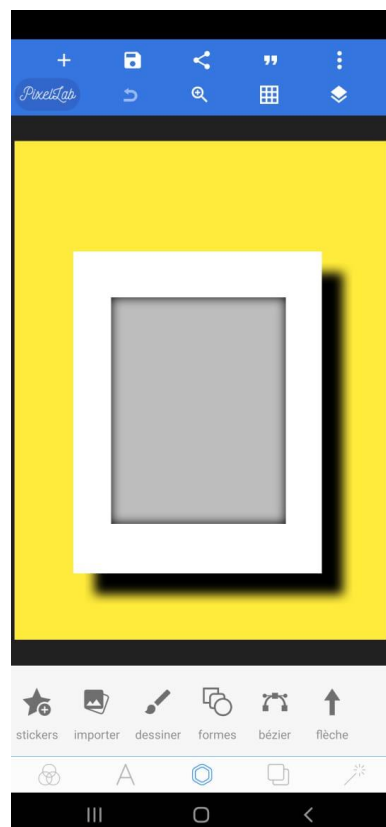
On ajoute un arrière-plan couleur :



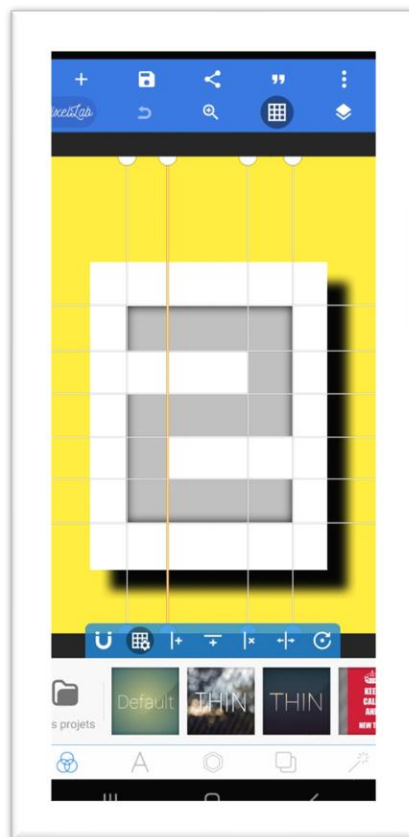
On ajoute un rectangle qui est notre patte avec un effet d'ombre :



On ajoute la partie grise à l'intérieur de notre patte qui va être colorer :



On ajoute des obstacles on mettant en compte le même mesure du parcours :



On ajoute un titre du numéro du niveau :



Généralement tous les niveaux qu'on a créés passe par les mêmes étapes.

CONCLUSION :

Pour conclure, nous avons réalisé notre projet fin de module avec Visual Studio basé à la bibliothèque cocos 2dx qui est une plateforme simple et facile à utiliser pour les développeurs débutants. Aussi sachant que sa flexibilité et utilité.

Le point fort de ce travail est d'apprendre comment réagir avec les nouveaux Framework et d'être capable de pratiquer le cursus académique au monde réel, plus de bien maîtriser la programmation orientes objets C++.

Les références :

<https://www.raywenderlich.com/1848-cocos2d-x-tutorial-for-beginners#toc-anchor-001>

https://www.youtube.com/watch?v=qXggSNUf9Cc&list=PLRtjMdoYXLf4od_bOKN3WjAPr7snPXzoe

<https://gamefromscratch.com/cocos2d-x-c-game-programming-tutorial-series/>