

Project: Game PICO PARK

-La création d'un jeu a partir de l'application cocos2d-x.

- Préparer par :
- Oumaima boughdad
- Hajar benyaich



■ C'est quoi cocos2d-x ?

Cocos2d-x est le moteur de jeu open source le plus populaire au monde. Cocos2d-x intègre deux langages de programmation principaux, C++ et Lua. Ce rapport se concentrera sur l'implémentation C++. Il existe également une version JavaScript appelée Cocos2d-JS, qui prend également en charge le développement Web. Pour mettre les choses en perspective, quiconque a joué à un jeu mobile en aura joué un qui a très probablement été construit à l'aide Cocos2d-x. Les capacités du moteur de jeu s'étendent au-delà du développement de jeux, avec fonctionnalités pour le développement d'applications générales. Cependant, l'aspect le plus important qui rend Cocos2d-x phénoménal est sa nature multiplateforme, permettant le développement pour toutes les principales plates-formes mobiles et de bureau.

Avant de configurer un projet, certains fichiers doivent être téléchargés afin pour poursuivre le projet.

Cependant, le développement Android peut se faire sur un Mac ou une machine Windows, et aucun compte spécial n'est nécessaire pour exécuter l'application sur un appareil Android. La liste suivante fournit les prérequis qui doivent être téléchargés pour configurer Cocos2d-x :

■ Téléchargement de cocos2d-x :

- Cocos2d-x : on l'a télécharger depuis <http://www.cocos2d-x.org/download> (au moment de la rédaction de ce livre, la v3.0 est la dernière version stable). C'est le moteur de jeu utilisé pour développer le jeu .
- Outils de développement Android (ADT) : cela n'est nécessaire que pour Android développement. On l'a télécharger depuis <http://developer.android.com/sdk/index.html>. Ces outils sont utilisés pour développer des applications Android. Les outils comprennent Android SDK et Eclipse IDE.

- Kit de développement natif (NDK) : nécessaire uniquement pour Android développement et peut être téléchargé à partir de <https://developer.android.com/tools/sdk/ndk/index.html>. NDK active l'application Android développement à l'aide de langages de programmation tels que C et C++.
- Apache ANT : ceci n'est nécessaire que pour le développement d'Android et peut être téléchargé depuis <http://ant.apache.org/bindownload.cgi>. C'est un Bibliothèque Java qui aide à créer des logiciels.

Ces étapes qui nous ont guider tout au long du

processus de configuration de Cocos2d-x :

1. Extrayez/décompressez tous les fichiers téléchargés.
2. Ouvrir le terminal.
3. Modifiez le répertoire dans le terminal vers le e répertoire racine Cocos2d-x, par exemple, `cd /répertoire/emplacement`.
4. Run `setup.py` using the `python ./setup.py` command.
5. Le terminal nous a demander `NDK_ROOT`, qui est le kit de développement natif ; faites glisser et

déposez le dossier racine NDK sur le terminal, puis on a appuyer sur Entrée.

6. Maintenant, le terminal nous a demander ANDROID_SDK_ROOT, qui fait partie de l'ADT emballer. Après le glissement et on a déposer le dossier SDK qui se trouve dans le dossier racine ADT sur le terminal, puis on a appuyer sur Entrée .

7. Ensuite, le terminal nous a demander ANT_ROOT. Après le glissement et on a déposer le dossier bin qui se trouve dans le dossier racine Apache sur le terminal, puis on a appuyer sur Entrée .

8. Enfin, .bash_profile doit être exécuté pour ajouter les variables système.

Maintenant après avoir terminé avec succès le processus d'installation de Cocos2d-x.

Les prochaines étapes généreront un nouveau projet Cocos2d-x à utiliser comme base pour créer Jeux. Les étapes suivantes nous a guider tout au long du processus de génération d'un nouveau Projet Cocos2d-x :

1. Ouvrir le terminal.
2. Exécuter la commande cocos avec les paramètres suivants :
 - ° nouveau nom de projet
 - ° -p nom du package (le nom de l'application dans le entreprise/organisation, qui doit être unique)
 - ° -l langage de programmation (cpp ou lua)
 - ° -d emplacement pour générer le projet .

Refactorisation de HelloWorldScene.h :

1. Renommer HelloWorldScene.h to MainMenuScene.h:
2. Ouvrir MainMenuScene.h. 3. Renommer the # commands at the top of the document, as follows: ° From #ifndef
__HELLOWORLD_SCENE_H__ to #ifndef
__MAINMENU_SCENE_H__ ° From #define
__HELLOWORLD_SCENE_H__ to #define
__MAINMENU_SCENE_H__

```

1  #ifndef __MAINMENU_SCENE_H__
2  #define __MAINMENU_SCENE_H__

```

3. Renommer the HelloWorld class a MainMenu:

```

6  class MainMenu : public cocos2d::Layer

```

4. supprimer le void

menuCloseCallback(cocos2d::Ref* pSender);
function.

5. Replacement de HelloWorld in CREATE_FUNC with MainMenu

Le code est comme suit :

```

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

```

Refactorisation de **HelloWorldScene.cpp:**

**1. Renomer HelloWorldScene.cpp to
MainMenuScene.cpp**

2. Ouvrir MainMenuScene.cpp.

**3. Include MainMenuScene.h instead of
HelloWorldScene.h**

**4. Avant la correction des erreurs, on a supprimer
le menu, label, et sprite de init() function**

**5. On a a supprimer le void
HelloWorld::menuCloseCallback(Ref* pSender)
function.**

**6. Finalement on renomer HelloWorld vers
MainMenu; ° Scene* HelloWorld::createScene()
vers Scene* MainMenu::createScene(): ° auto
layer = HelloWorld::create(); to auto layer =
MainMenu::create();: ° bool HelloWorld::init()
vers bool MainMenu::init(): www.it-ebooks.info
Setting Up .**

Le MainMenuScene.cpp file est comme suit :

```
#include "MainMenuScene.h"
#include "GameScene.h"
USING_NS_CC;

Scene* MainMenu::createScene()
{
    auto scene = Scene::create();

    auto layer = MainMenu::create();

    scene->addChild(layer);

    return scene;
    //return HelloWorld::create();
}

// on "init" you need to initialize your instance
bool MainMenu::init()
{
    //////////////////////////////////////
    // 1. super init first
    if (!Layer::init())
    {
        return false;
    }

    Size visibleSize = Director::getInstance()->getVisibleSize();
    Point origin = Director::getInstance()->getVisibleOrigin();
}
```

L'addition des scenes :

1. Duplicate le MainMenuScene.h et MainMenuScene.cpp fichiers et on les a renommer comme suit : GameScene.h et GameScene.cpp respectivement
2. Puis, right-click sur la Classes folder dans XCode et on a selectioner Add Files vers Pico Park, et on a Selectioner le fichires qui sera additionne et on a a cliquer sur Add.

Refactorisation de GameScene.h :

```
#ifndef _GAME_SCENE_H_
#define _GAME_SCENE_H_

#include "cocos2d.h"

class GameScreen : public cocos2d::Layer
{
public:
    static cocos2d::Scene* createScene();

    virtual bool init();

    // a selector callback

    // implement the "static create()" method manually
    CREATE_FUNC(GameScreen);

    void GoToPauseScene(Ref* pSender);
    void GoToGameOverScene(Ref* pSender);

    cocos2d::Sprite* mySprite;
};

#endif // _GAME_SCENE_H_
```

Refactorisation de

GameScene.cpp :

1. On a ouvert GameScene.cpp file.
2. Include GameScene.h a la place de MainMenuScene.h.
3. On a Changer toutes instance de MainMenu vers GameScreen.

```
1  #include "GameScene.h"
2
3  USING_NS_CC;
4
5  Scene* GameScreen::createScene()
6  {
7      // 'scene' is an autorelease object
8      auto scene = Scene::create();
9
10     // 'layer' is an autorelease object
11     auto layer = GameScreen::create();
12
13     // add layer as a child to scene
14     scene->addChild(layer);
15
16     // return the scene
17     return scene;
18 }
19
20 // on "init" you need to initialize your instance
21 bool GameScreen::init()
22 {
23     ///////////////////////////////////
24     // 1. super init first
25     if ( !Layer::init() )
26     {
27         return false;
28     }
29
30     Size visibleSize = Director::getInstance()->getVisibleSize();
31     Point origin = Director::getInstance()->getVisibleOrigin();
32
33     return true;
34 }
```

Code de Main Menu scene:

```
3
4  USING_NS_CC;
5
6  Scene* MainMenu::createScene()
7  {
8      // 'scene' is an autorelease object
9      auto scene = Scene::create();
10
11     // 'layer' is an autorelease object
12     auto layer = MainMenu::create();
13
14     // add layer as a child to scene
15     scene->addChild(layer);
16
17     // return the scene
18     return scene;
19 }
20
21 // on "init" you need to initialize your instance
22 bool MainMenu::init()
23 {
24     //////////////////////////////////////
25     // 1. super init first
26     if ( !Layer::init() )
27     {
28         return false;
29     }
30
31     Size visibleSize = Director::getInstance()->getVisibleSize();
32     Point origin = Director::getInstance()->getVisibleOrigin();
33
34     return true;
35 }
36
37 void MainMenu::GoToGameScene(cocos2d::Ref *pSender)
38 {
39     auto scene = GameScreen::createScene();
40
41     Director::getInstance()->replaceScene(scene);
42 }
```

Code for the Game scene:

On a ajouter le code suivant a
GameScene.h:

```
void GoToPauseScene(Ref *pSender);  
void GoToGameOverScene(Ref *pSender);
```

Les deux premières lignes de l'extrait de code précédent sont utilisées pour inclure la pause et Game Over scènes afin que les scènes soient accessibles. La fonction **GoToPauseScene** crée d'abord une instance de scène locale de la scène Pause, puis la pousse sur la empiler. La fonction **GoToGameOverScene** crée d'abord une instance de scène locale du Game Over et remplace ensuite la scène Game par celle-ci. Le fichier **GameScene.cpp** devrait ressembler à la capture d'écran suivante :

```

1  #include "GameOverScene.h"
2  #include "GameScene.h"
3  #include "MainMenuScene.h"
4
5  USING_NS_CC;
6
7  Scene* GameOver::createScene()
8  {
9      // 'scene' is an autorelease object
10     auto scene = Scene::create();
11
12     // 'layer' is an autorelease object
13     auto layer = GameOver::create();
14
15     // add layer as a child to scene
16     scene->addChild(layer);
17
18     // return the scene
19     return scene;
20 }
21
22 // on "init" you need to initialize your instance
23 bool GameOver::init()
24 {
25     //////////////////////////////////////
26     // 1. super init first
27     if ( !Layer::init() )
28     {
29         return false;
30     }
31
32     Size visibleSize = Director::getInstance()->getVisibleSize();
33     Point origin = Director::getInstance()->getVisibleOrigin();
34
35     return true;
36 }
37
38 void GameOver::GoToGameScene(cocos2d::Ref *pSender)
39 {
40     auto scene = GameScreen::createScene();
41
42     Director::getInstance()->replaceScene(scene);
43 }
44
45 void GameOver::GoToMainMenuScene(cocos2d::Ref *pSender)
46 {
47     auto scene = MainMenu::createScene();
48
49     Director::getInstance()->replaceScene(scene);
50 }

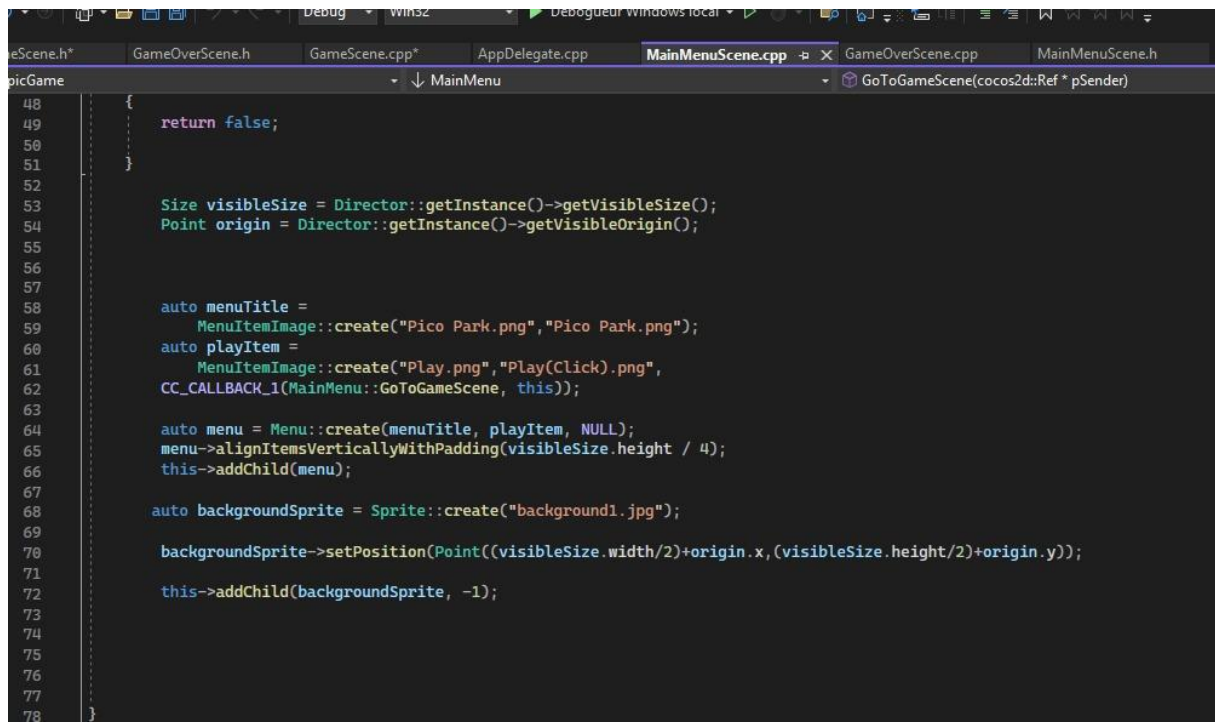
```


Code for the Pause scene :

```
1 #include "PauseScene.h"
2 #include "GameScene.h"
3 #include "MainMenuScene.h"
4
5 USING_NS_CC;
6
7 Scene* PauseMenu::createScene()
8 {
9     // 'scene' is an autorelease object
10     auto scene = Scene::create();
11
12     // 'layer' is an autorelease object
13     auto layer = PauseMenu::create();
14
15     // add layer as a child to scene
16     scene->addChild(layer);
17
18     // return the scene
19     return scene;
20 }
21
22 // on "init" you need to initialize your instance
23 bool PauseMenu::init()
24 {
25     ///////////////////////////////////
26     // 1. super init first
27     if ( !Layer::init() )
28     {
29         return false;
30     }
31
32     Size visibleSize = Director::getInstance()->getVisibleSize();
33     Point origin = Director::getInstance()->getVisibleOrigin();
34
35     return true;
36 }
37
38 void PauseMenu::Resume(cocos2d::Ref *pSender)
39 {
40     Director::getInstance()->popScene();
41 }
42
43 void PauseMenu::GoToMainMenuScene(cocos2d::Ref *pSender)
44 {
45     auto scene = MainMenu::createScene();
46
47     Director::getInstance()->popScene();
48     Director::getInstance()->replaceScene(scene);
49 }
50
51 void PauseMenu::Retry(cocos2d::Ref *pSender)
52 {
53     auto scene = GameScreen::createScene();
54
55     Director::getInstance()->popScene();
56     Director::getInstance()->replaceScene(scene);
57 }
```

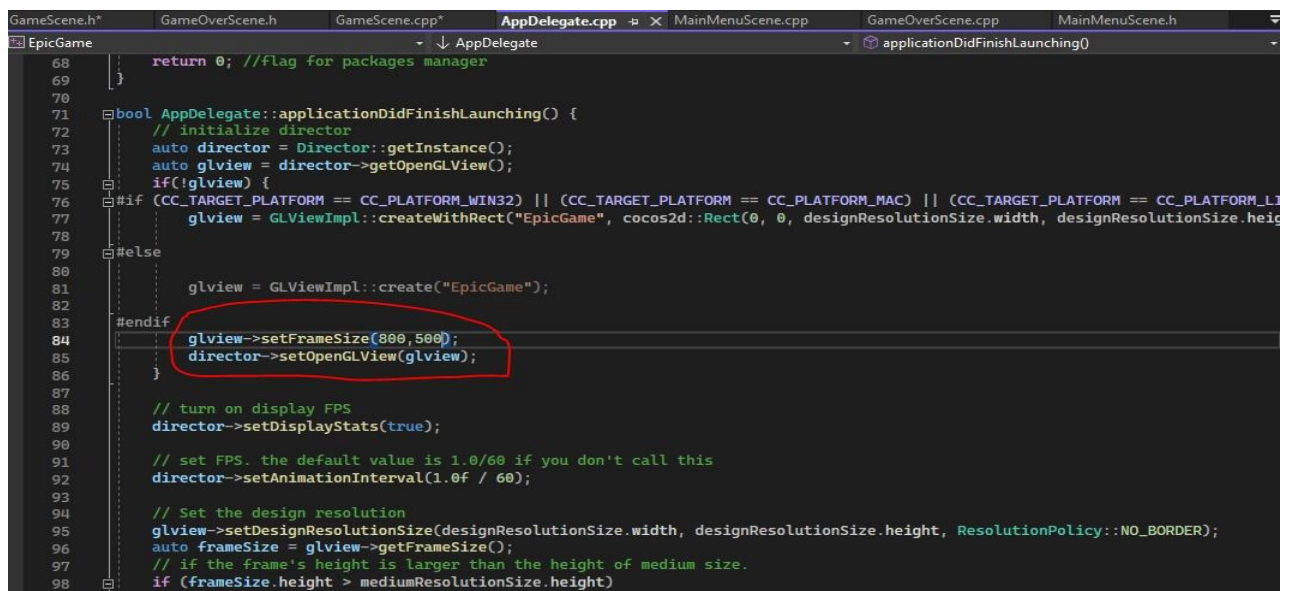
Le background du jeu :

```
auto backgroundSprite = Sprite::create("backgroundTEST.jpg");
backgroundSprite->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
this->addChild(backgroundSprite, -1);
```



```
48     return false;
49
50 }
51
52
53 Size visibleSize = Director::getInstance()->getVisibleSize();
54 Point origin = Director::getInstance()->getVisibleOrigin();
55
56
57 auto menuItem = MenuItemImage::create("Pico Park.png", "Pico Park.png");
58 auto playItem = MenuItemImage::create("Play.png", "Play(Click).png", CC_CALLBACK_1(MainMenu::GoToGameScene, this));
59 auto menu = Menu::create(menuItem, playItem, NULL);
60 menu->alignItemsVerticallyWithPadding(visibleSize.height / 4);
61 this->addChild(menu);
62
63 auto backgroundSprite = Sprite::create("background1.jpg");
64
65 backgroundSprite->setPosition(Point((visibleSize.width/2)+origin.x, (visibleSize.height/2)+origin.y));
66
67 this->addChild(backgroundSprite, -1);
68
69
70
71
72
73
74
75
76
77
78 }
```

La résolution de la fenêtre du jeu :



```
68     return 0; //flag for packages manager
69 }
70
71 bool AppDelegate::applicationDidFinishLaunching() {
72     // initialize director
73     auto director = Director::getInstance();
74     auto glview = director->getOpenGLView();
75     if(!glview) {
76         #if (CC_TARGET_PLATFORM == CC_PLATFORM_WIN32) || (CC_TARGET_PLATFORM == CC_PLATFORM_MAC) || (CC_TARGET_PLATFORM == CC_PLATFORM_LINUX)
77             glview = GLViewImpl::createWithRect("EpicGame", cocos2d::Rect(0, 0, designResolutionSize.width, designResolutionSize.height));
78         #else
79             glview = GLViewImpl::create("EpicGame");
80         #endif
81         // Set the frame size
82         glview->setFrameSize(800, 500);
83         director->setOpenGLView(glview);
84     }
85
86     // turn on display FPS
87     director->setDisplayStats(true);
88
89     // set FPS. the default value is 1.0/60 if you don't call this
90     director->setAnimationInterval(1.0f / 60);
91
92     // Set the design resolution
93     glview->setDesignResolutionSize(designResolutionSize.width, designResolutionSize.height, ResolutionPolicy::NO_BORDER);
94     auto frameSize = glview->getFrameSize();
95     // if the frame's height is larger than the height of medium size.
96     if (frameSize.height > mediumResolutionSize.height)
97     {
98         //
99     }
```


Additon de Game

Menu:

Les menus sont une collection d'éléments qui sont organisés pour former les boutons structurés et peut être utilisé pour les fonctionnalités d'un jeu, telles que la navigation. Notre jeu utilisera des menus pour les cas d'utilisation suivants :

- Dans la scène Menu principal :**
 - Le bouton de lecture, qui lance le jeu**
- Dans la scène Jeu :**
 - Le bouton pause, qui met le jeu en pause**
- Dans la scène Pause :**
 - Le bouton de reprise, qui reprend le jeu**
 - Le bouton réessayer, qui relance le jeu**
 - Le bouton du menu principal, qui vous amène au menu principal**

- Dans la scène Game Over :
 - Le bouton réessayer, qui relance le jeu
 - Le bouton du menu principal, qui vous amène au menu principal

```
#include "MainMenuScene.h"
#include "GameScene.h"
USING_NS_CC;

Scene* MainMenu::createScene()
{
    auto scene = Scene::create();

    auto layer = MainMenu::create();

    scene->addChild(layer);

    return scene;
    //return HelloWorld::create();
}

// on "init" you need to initialize your instance
bool MainMenu::init()
{
    ///////////////////////////////////////////////////
    // 1. super init first
    if (!Layer::init())
    {
        return false;
    }

    Size visibleSize = Director::getInstance()->getVisibleSize();
    Point origin = Director::getInstance()->getVisibleOrigin();

    auto menuTitle =
        MenuItemImage::create("Pico Park.png", "Pico Park.png");
    auto playItem =
        MenuItemImage::create("play.png", "play(Click).png",
            CC_CALLBACK_1(MainMenu::GoToGameScene, this));

    auto menu = Menu::create(menuTitle, playItem, NULL);
    menu->alignItemsVerticallyWithPadding(visibleSize.height / 4);
    this->addChild(menu);

    auto backgroundSprite = Sprite::create("Background1.jpg");
    backgroundSprite->setPosition(Point((visibleSize.width/2)+origin.x, (visibleSize.height/2)+origin.y));
    this->addChild(backgroundSprite);
}
```

```

    auto menuItemImage = MenuItemImage::create("Pico Park.png", "Pico Park.png");
    auto playItem = MenuItemImage::create("play.png", "play(click).png",
        CC_CALLBACK_1(MainMenu::GoToGameScene, this));

    auto menu = Menu::create(menuTitle, playItem, NULL);
    menu->alignItemsVerticallyWithPadding(visibleSize.height / 4);
    this->addChild(menu);

    auto backgroundSprite = Sprite::create("Background1.jpg");
    backgroundSprite->setPosition(Point((visibleSize.width/2)+origin.x, (visibleSize.height/2)+origin.y));
    this->addChild(backgroundSprite, -1);
}

void MainMenu::GoToGameScene(cocos2d::Ref* pSender) {
    auto scene = GameScreen::createScene();
    Director::getInstance()->replaceScene(scene);
}

```

Coding the menus in the Game Over scene:

```

#include "GameOverScene.h"
#include "GameScene.h"
#include "MainMenuScene.h"

USING_NS_CC;

Scene* GameOver::createScene()
{
    auto scene = Scene::create();
    auto layer = GameOver::create();
    scene->addChild(layer);

    return scene;
    //return HelloWorld::create();
}

// on "init" you need to initialize your instance
bool GameOver::init()
{
    ///////////////////////////////////////////////////
    // 1. super init first
    if (!Layer::init())
    {
        return false;
    }

    auto visibleSize = Director::getInstance()->getVisibleSize();
    Point origin = Director::getInstance()->getVisibleOrigin();

    auto menuTitle = MenuItemImage::create("GameOver.png", "GameOver.png");
    auto retryItem = MenuItemImage::create("Retry.png",
        "Retry(click).png",
        CC_CALLBACK_1(GameOver::GoToGameScene, this));
    auto mainMenuItem = MenuItemImage::create("Menu.png",
        "Menu(click).png",
        CC_CALLBACK_1(GameOver::GoToMainMenuScene, this));

    auto menu = Menu::create(menuTitle, retryItem, mainMenuItem, NULL);
    menu->alignItemsVerticallyWithPadding(visibleSize.height / 4);
}

```

% No issues found

```

// 1. super init first
if (!Layer::init())
{
    return false;
}

auto visibleSize = Director::getInstance()->getVisibleSize();
Point origin = Director::getInstance()->getVisibleOrigin();

auto menuTitle = MenuItemImage::create("GameOver.png", "GameOver.png");
auto retryItem = MenuItemImage::create("Retry.png",
    "Retry(Click).png",
    CC_CALLBACK_1(GameOver::GoToGameScene, this));
auto mainMenuItem = MenuItemImage::create("Menu.png",
    "Menu(Click).png",
    CC_CALLBACK_1(GameOver::GoToMainMenuScene, this));

auto menu = Menu::create(menuTitle, retryItem, mainMenuItem, NULL);

menu->alignItemsVerticallyWithPadding(visibleSize.height / 4);

this->addChild(menu);

return true;
}

void GameOver::GoToGameScene(cocos2d::Ref* pSender)
{
    auto scene = GameScreen::createScene();

    Director::getInstance()->replaceScene(scene);
}

void GameOver::GoToMainMenuScene(cocos2d::Ref* pSender)
{
    auto scene = MainMenu::createScene();

    Director::getInstance()->replaceScene(scene);
}

```

The Game Sprites

Adding the Main Menu sprites :

L'addition du code suivant :

```
auto backgroundSprite = Sprite::create  
("MainMenuScreen/Background.png");  
backgroundSprite-  
>setPosition(Point((visibleSize.width / 2) +  
origin.x, (visibleSize.height / 2) + origin.y)); this-  
>addChild(backgroundSprite, -1);
```

Adding the Game Over sprites:

L'addition du code suivant :

```
auto backgroundSprite = Sprite::create  
("GameOverScreen/Game_Over_Screen_Backgro  
und.png"); backgroundSprite-  
>setPosition(Point((visibleSize.width / 2) +  
origin.x, (visibleSize.height / 2) + origin.y)); this-  
>addChild(backgroundSprite, -1);
```

Adding the Game sprites :

```
#include "GameScene.h"
#include "PauseScene.h"
#include "GameOverScene.h"

USING_NS_CC;

Scene* GameScreen::createScene()
{
    auto scene = Scene::create();

    auto layer = GameScreen::create();

    scene->addChild(layer);



    return scene;
    //return HelloWorld::create();
}

// on "init" you need to initialize your instance
bool GameScreen::init()
{
    ///////////////////////////////////////////////////
    // 1. super init first
    if (!Layer::init())
    {
        return false;
    }

    Size visibleSize = Director::getInstance()->getVisibleSize();
    Point origin = Director::getInstance()->getVisibleOrigin();

    auto pauseItem = MenuItemImage::create("Pause.png", "Pause(Click).png", CC_CALLBACK_1(GameScreen::GoToPauseScene, this));
    pauseItem->setPosition(Point(pauseItem->getContentSize().width - (pauseItem->getContentSize().width / 4) +
        origin.x, visibleSize.height - pauseItem->getContentSize().height +
        (pauseItem->getContentSize().width / 4) + origin.y));
    auto menu = Menu::create(pauseItem, NULL);
    menu->setPosition(Point::ZERO);
    this->addChild(menu);

    auto backgroundSprite = Sprite::create("Background1.jpg");
    backgroundSprite->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
    this->addChild(backgroundSprite, -1);
}
```

%   No issues found

```
auto backgroundSprite = Sprite::create("Background1.jpg");
backgroundSprite->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
this->addChild(backgroundSprite, -1);

mySprite = Sprite::create("WASHA.png");
mySprite->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
mySprite->setScale(0.3);
this->addChild(mySprite);
```

Actions :

La structure générale du code pour exécuter une action est la suivante :

```
Autoaction=actionName::create(actionPr  
operties); nodeName-  
>runAction(actionName);
```

Moving:

```
auto action = MoveBy::create(duration,  
Point (xPosition, yPosition)); playerSprite-  
>runAction(action);
```

```
auto action = MoveBy::create(2.0,  
Point(30, 40)); playerSprite-  
>runAction(action);
```

```
60 auto action = MoveBy::create(2.0, Point(30, 40));  
61 playerSprite->runAction(action);
```

Moving the spirite:

```
//*****  
mySprite = Sprite::create("WASHA.png");  
mySprite->setAnchorPoint(Vec2(0.5, 0.5));  
mySprite->setPosition(Vec2(50, 60));  
// mySprite->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));  
mySprite->setScale(0.3);  
mySprite->setName("mySprite");  
this->addChild(mySprite, 2);
```

Deplacer le spirite avec les touches : A -W- S- D

```
me GameScreen init()  
//*****  
auto eventListener = EventListenerKeyboard::create();  
eventListener->onKeyPressed = [](EventKeyboard::KeyCode keyCode, Event* event) {  
    int offsetX = 0, offsetY = 0;  
  
    switch (keyCode) {  
        case EventKeyboard::KeyCode::KEY_LEFT_ARROW:  
        case EventKeyboard::KeyCode::KEY_A:  
            offsetX = -30;  
            break;  
        case EventKeyboard::KeyCode::KEY_RIGHT_ARROW:  
        case EventKeyboard::KeyCode::KEY_D:  
            offsetX = 30;  
            break;  
        case EventKeyboard::KeyCode::KEY_UP_ARROW:  
        case EventKeyboard::KeyCode::KEY_W:  
            offsetY = 30;  
            break;  
        case EventKeyboard::KeyCode::KEY_DOWN_ARROW:  
        case EventKeyboard::KeyCode::KEY_S:  
            offsetY = -30;  
            break;  
    }  
    auto moveTo = MoveTo::create(0.3, Vec2(event->getCurrentTarget()->getPositionX() + offsetX, event->getCurrentTarget()->getPositionY() + offsetY));  
    event->getCurrentTarget()->runAction(moveTo);  
};  
this->_eventDispatcher->addEventListenerWithSceneGraphPriority(eventListener, mySprite);  
//*****
```

Les nuages se sont les emplacements ou le spirite peut se déplacé en dessus .


```
auto* floor2 = Sprite::create("CLOUD.png");
floor2->setAnchorPoint(Vec2(0, 0));
floor2->setPosition(Vec2(30, 25));
floor2->setScale(0.2);
this->addChild(floor2, 1);

auto physicsBody_floor2 = PhysicsBody::createBox(floor2->getContentSize(), PhysicsMaterial(500.0f, 0.1f, 0.9f));
physicsBody_floor2->setDynamic(false);
physicsBody_floor2->setCollisionBitmask(1);
physicsBody_floor2->setCategoryBitmask(1);

floor2->setPhysicsBody(physicsBody_floor2);
```

Le code suivant va permettre le sprite de ce déplacé sur les nuages sont craindre de tomber.

```
auto physicsBody1 = PhysicsBody::createBox(mySprite->getContentSize(), PhysicsMaterial(1000.0f, 0.55f, 0.55f));
physicsBody1->setDynamic(true);
physicsBody1->setContactTestBitmask(1);
physicsBody1->setRotationEnable(false);
physicsBody1->setCollisionBitmask(1);
mySprite->setPhysicsBody(physicsBody1);
Vec2 force = Vec2(0, -physicsBody1->getMass() * 11.8f);
physicsBody1->applyForce(force);
```

Pause scene :

```
#include "PauseScene.h"
#include "GameScene.h"
#include "MainMenuScene.h"

USING_NS_CC;

Scene* PauseMenu::createScene()
{
    auto scene = Scene::create();
    auto layer = PauseMenu::create();
    scene->addChild(layer);
    return scene;
    //return HelloWorld::create();
}

// on "init" you need to initialize your instance
bool PauseMenu::init()
{
    ///////////////////////////////////////////////////
    // 1. super init first
    if (!Layer::init())
    {
        return false;
    }

    Size visibleSize = Director::getInstance()->getVisibleSize();
    Point origin = Director::getInstance()->getVisibleOrigin();

    auto resumeItem = MenuItemImage::create("Resume.png",
        "Resume(Click).png",
        CC_CALLBACK_1(PauseMenu::Resume, this));
    auto retryItem = MenuItemImage::create("Retry.png",
        "Retry(Click).png",
        CC_CALLBACK_1(PauseMenu::Retry, this));
    auto mainMenuItem = MenuItemImage::create("Menu.png",
        "Menu(Click).png",
        CC_CALLBACK_1(PauseMenu::GoToMainMenuScene, this));
    auto menu = Menu::create(resumeItem, retryItem, mainMenuItem, NULL);
    menu->alignItemsVerticallyWithPadding(visibleSize.height / 4);
    this->addChild(menu);

    auto backgroundSprite = Sprite::create("Background1.jpg");
    backgroundSprite->setPosition(Direct((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
}
```

%



No issues found

```

        auto resumeItem = MenuItemImage::create("Resume.png",
            "Resume(Click).png",
            CC_CALLBACK_1(PauseMenu::Resume, this));
        auto retryItem = MenuItemImage::create("Retry.png",
            "Retry(Click).png",
            CC_CALLBACK_1(PauseMenu::Retry, this));
        auto mainMenuItem = MenuItemImage::create("Menu.png",
            "Menu(Click).png",
            CC_CALLBACK_1(PauseMenu::GoToMainMenuScene, this));
        auto menu = Menu::create(resumeItem, retryItem, mainMenuItem, NULL);
        menu->alignItemsVerticallyWithPadding(visibleSize.height / 4);
        this->addChild(menu);

        auto backgroundSprite = Sprite::create("Background1.jpg");
        backgroundSprite->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
        this->addChild(backgroundSprite, -1);

        return true;
    }

void PauseMenu::Resume(cocos2d::Ref* pSender)
{
    Director::getInstance()->popScene();
}

void PauseMenu::GoToMainMenuScene(cocos2d::Ref* pSender)
{
    auto scene = MainMenu::createScene();

    Director::getInstance()->popScene();

    Director::getInstance()->replaceScene(scene);
}

void PauseMenu::Retry(cocos2d::Ref* pSender)
{
    auto scene = GameScreen::createScene();

    Director::getInstance()->popScene();

    Director::getInstance()->replaceScene(scene);
}

```

Les difficultés rencontrées:

Dans un certain moment , a la dernière phase de la programmation de notre projet (jeu) l'application cocos2d-x ne pouvait plus exécuter notre programme, du coup on ne pouvait plus l'exécuter.

Mais heureusement on avait déjà pris quelques photos et une vidéo mais cela était avant la finition de notre projet, il était encore incomplet.

La répartition du travaille de notre binôme :

Oumaima boughdad: a pris on charge l'installation de l'application de cocos2d-x et aussi la programmation des codes de classes : MainMenuScene.cpp et MainMenuScene.h aussi de GameScene.cpp et GameScene.h .

Benyaich hajar : a pris on charge la description du code du jeu a partir de la rédaction de ce rapport et aussi la programmation des codes des classes : GameOverScene.cpp et GameOverScene.h aussi PauseScene.cpp et PauseScee.h.