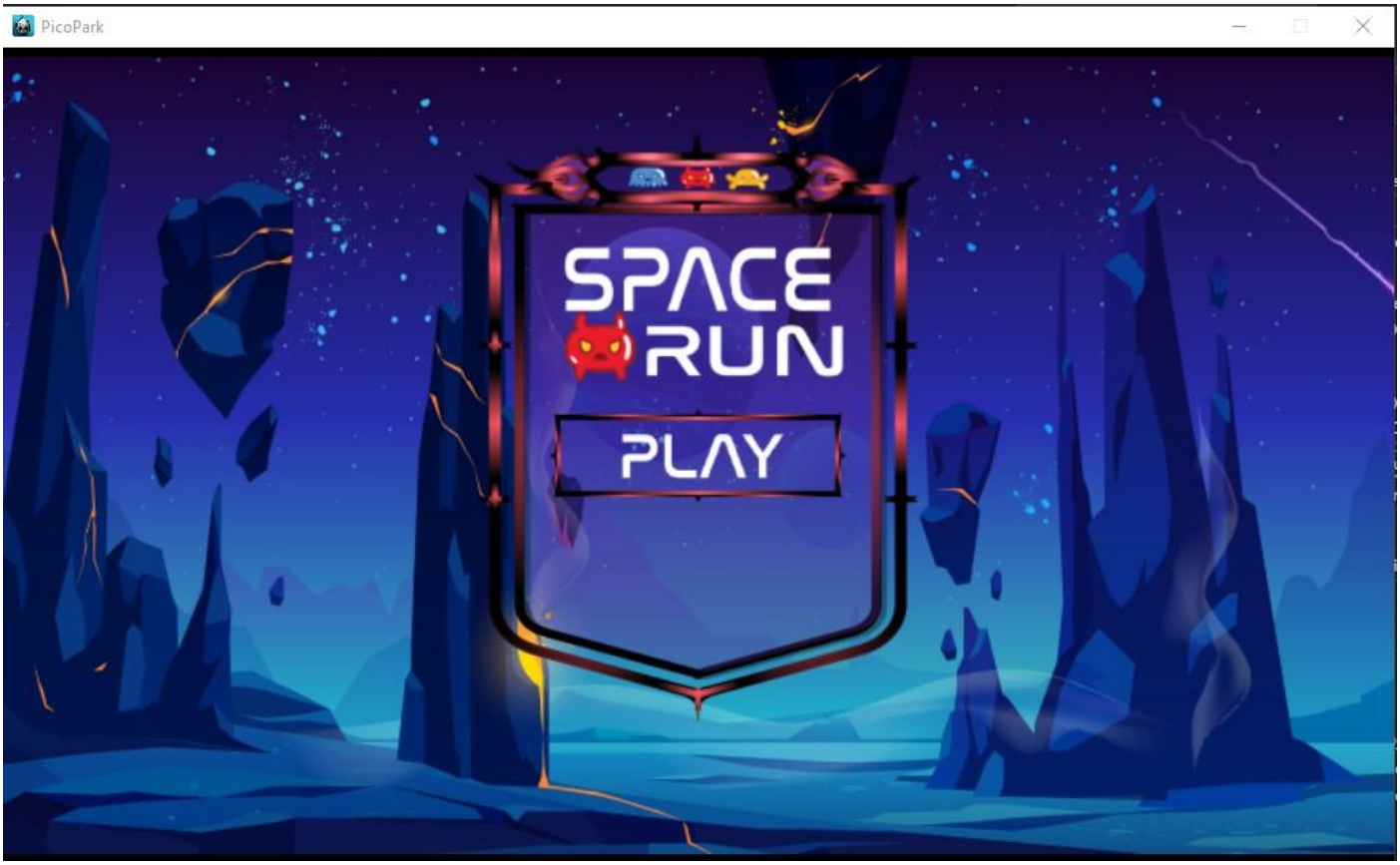
Rapport du projet

Pico Park



Realisation par : CHAHID ANASS

ECHEBRAAL AHMAD REDOUANE

Un petit mot :

Le document que vous avez sous vos yeux représente le rapport du projet final pour le module du *POO* en *C++* qui été sous forme d'un jeux vidéo nome ' *Pico Park* ' créé en appliquant les connaissances requissent tout le long du module et en se basant sur le moteur de jeu ' *cocos2s-x* '.

Avant de commencer, nous tenons à remercier note professeurs madame Abdelouahab Ikram et El Aachak Lotfi qui nous ont encadre tous le long du projet, et on espère que ce simple jeu vous plait.

Table de contenu :

I-préparation :

Avant d'entamer la partie saisie du code, il fallait faire quelques modifications au niveau du code source donné par défaut lors de la création d'un nouveau projet sur 'Cocos2d-x'.

La 1ier chose à modifier était la résolution de la fenêtre du jeu à travers le fichier AppDelegate.cpp a la ligne suivante :

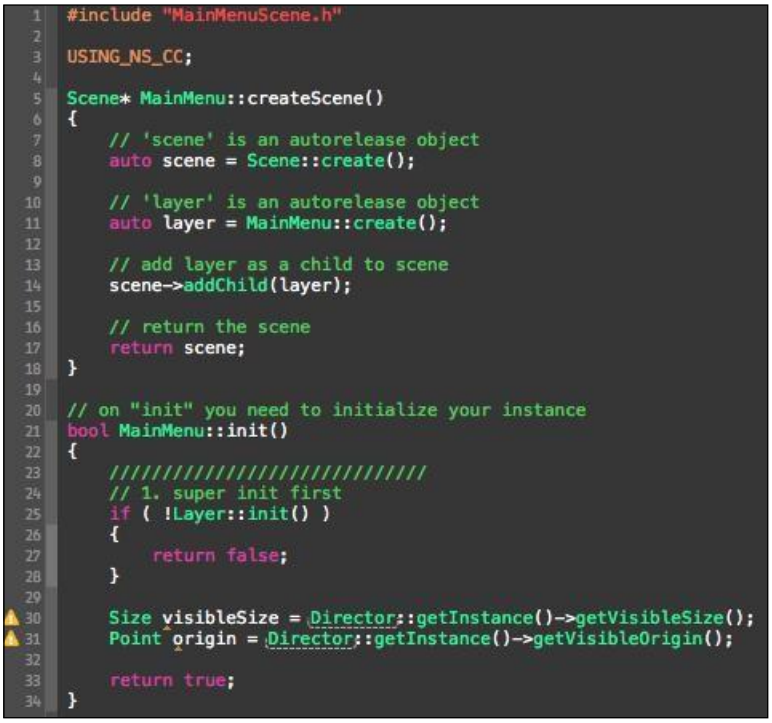
```
13    static cocos2d::Size designResolutionSize = cocos2d::Size(480, 320);
14    static cocos2d::Size smallResolutionSize = cocos2d::Size(800, 600);
15    static cocos2d::Size mediumResolutionSize = cocos2d::Size(1024, 768);
16    static cocos2d::Size largeResolutionSize = cocos2d::Size(2048, 1536);
```

Puis, désactiver l'affichage des statistiques du jeu (nombre des FPS …) sur l'écran à travers la ligne suivante :

```
61        // turn on display FPS
62        director->setDisplayStats(false);
```

Finalement effacer le contenu du fichier HelloWorldScene.cpp sauf la déclaration du fonction **HelloWorld::init()** puis renommer la nom du classe déclarée sur **HelloWorldScene.h** vers **MainMenuScene** puis tout le nom du fichier cpp et header.

```
#ifndef __MAINMENU_SCENE_H__
#define __MAINMENU_SCENE_H__

#include "cocos2d.h"

class MainMenu : public cocos2d::Layer
{
public:
    // there's no 'id' in cpp, so we recommend returning the class instance pointer
    static cocos2d::Scene* createScene();

    // Here's a difference. Method 'init' in cocos2d-x returns bool, instead of returning 'id' in cocos2d-iphone
    virtual bool init();

    // implement the "static create()" method manually
    CREATE_FUNC(MainMenu);
};
```



```cpp
#include "MainMenuScene.h"

USING_NS_CC;

Scene* MainMenu::createScene()
{
    // 'scene' is an autorelease object
    auto scene = Scene::create();

    // 'layer' is an autorelease object
    auto layer = MainMenu::create();

    // add layer as a child to scene
    scene->addChild(layer);

    // return the scene
    return scene;
}

// 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();

    return true;
}
```

2-Creation des scènes :

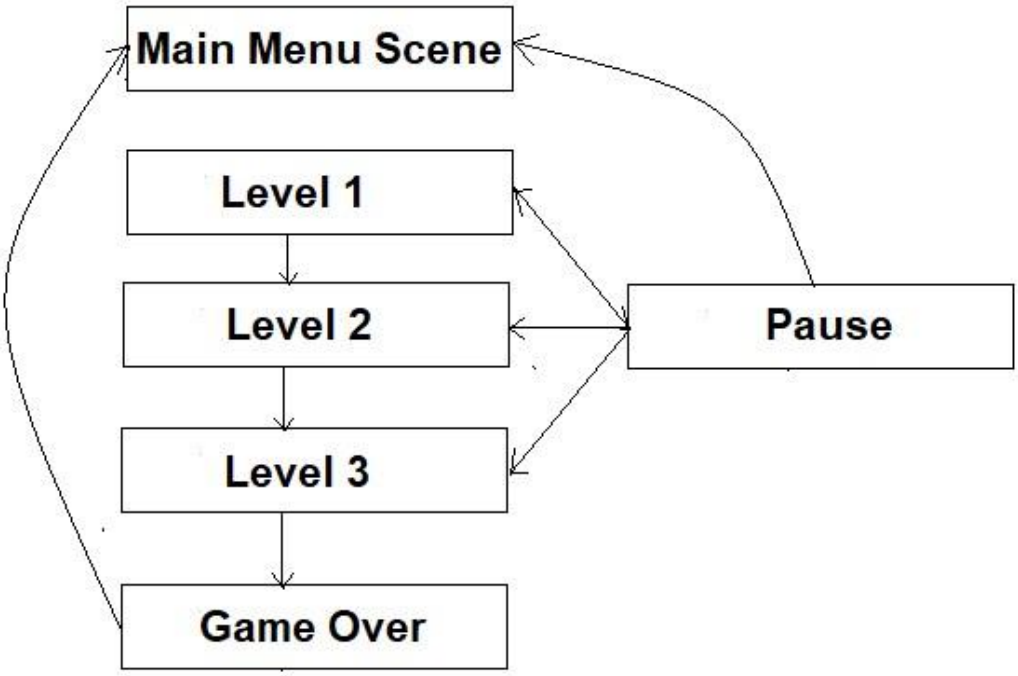
Le jeu sera composé d'une scène **MainMenuScene** lancée au démarrage du jeu. Cette Scène va permettre de lancer le 1ier niveau etc…

Chaque niveau aura son fichier diffèrent des autres niveaux et sera considéré comme une scène indépendante.

Le jeu aura aussi une scène Pause qui va permettre au joueur de :

- Continuer de jouer
- Réessayer le niveau
- Revenir au menu principal

Le schéma ci-dessous représente le pathing du jeu :



- *Main menu scene:*

On ajoute ce code au **MainMenuScene.cpp** document :

```
void MainMenu::GoToLvl1Scene(Ref* pSender)
{
    auto scene = Lvl1Scene::createScene();
    Director::getInstance()->replaceScene(scene);
}
```

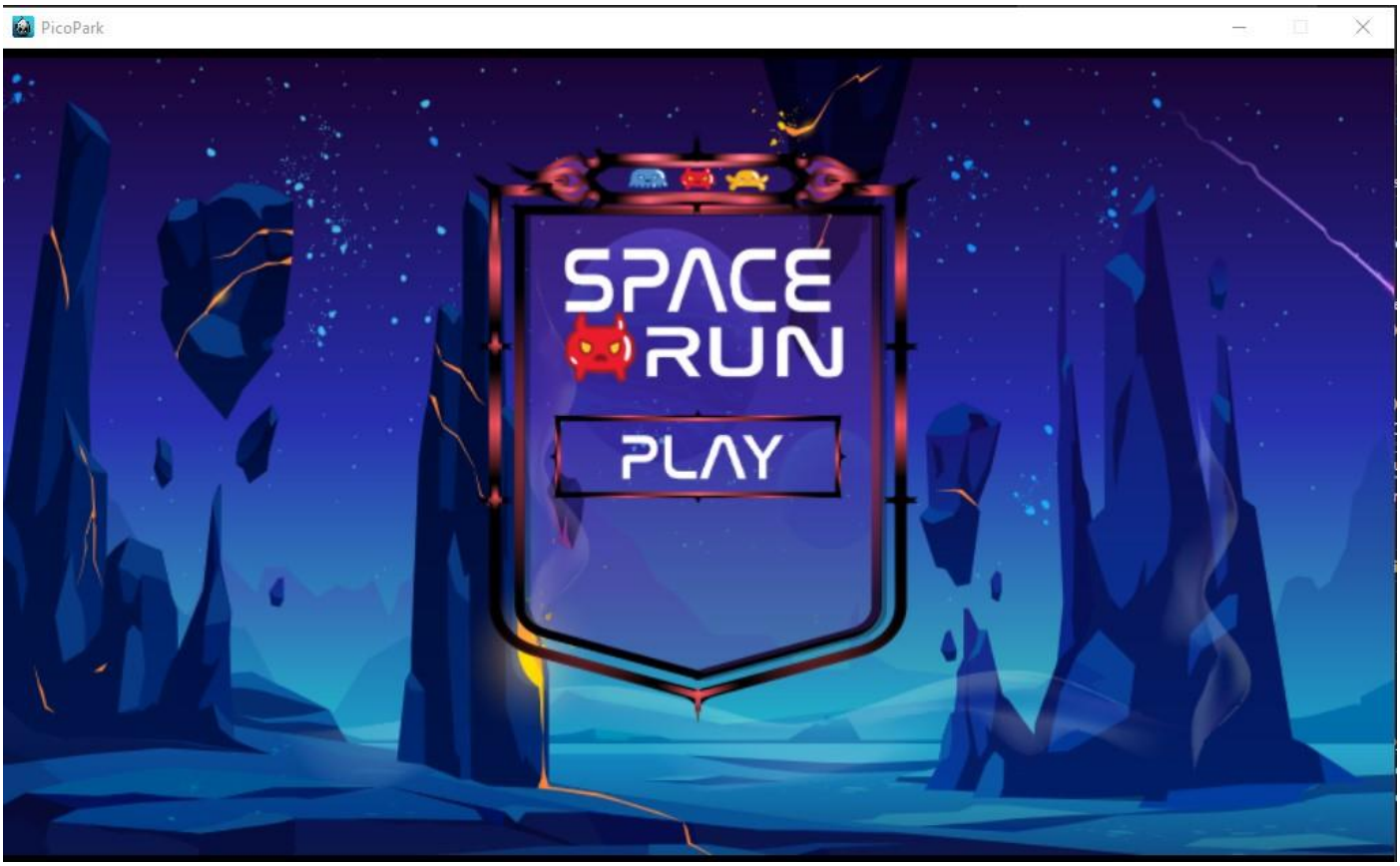
Dans le code précèdent, le 1ere ligne est utiliser pour inclusion du Game scene du 1ere niveau . La fonction GoTolvl1Scene remplace le scène actuel usant le directeur Cocos2d-x . Le document doit rassembler au code suivant :

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

USING_NS_CC;

Scene* MainMenu::createScene()
{
    // 'scene' is an autorelease object
    auto scene = Scene::create();

    // 'layer' is an autorelease object
    auto layer = MainMenu::create();

    // add layer as a child to scene
    scene->addChild(layer);

    // return the scene
    return scene;
}

// 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();

    return true;
}

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

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

Lors de l'appui sur le bouton, la fonction précédente sera invoquée et permettra l'accès au niveau à l'aide de la commande :

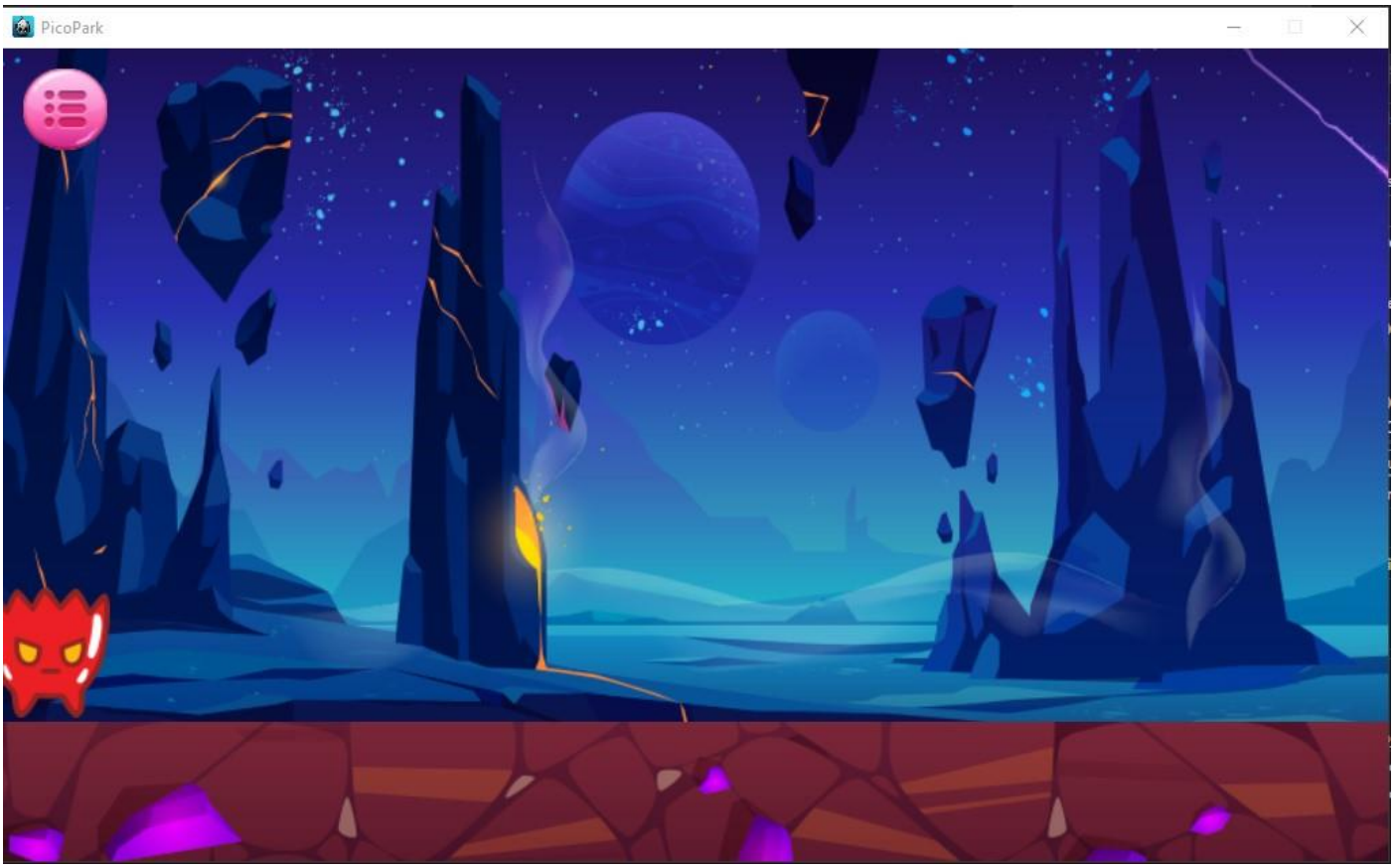
```
 CC_CALLBACK_1(MainMenu::GoToLvl1Scene, this));
//fonction pour switcher au scene prochaine
```

Cette image représente le main menu la 1 ère chose que vois le joueur quand-il lance le jeu:



- *Game scene 1*

Cette scène représente le premier niveau du jeu. La structure du niveau est la suivante :



Dans cette scène, et au niveau du fichier **lvl1Scene.cpp**, un Sprite a été en utilisant les lignes suivantes :

```
    auto sprite = Sprite::create("player.png");
    sprite->setPosition(16,110);
    auto physicsBody = PhysicsBody::createCircle(sprite->getContentSize().width / 2);
    physicsBody->setDynamic(true);
    sprite->setPhysicsBody(physicsBody);
    this->addChild(sprite);
```

Cette Sprite va prendre **la position (16,110)** qui représente sa position initiale puis ajoutée à la scène.

On a aussi un bouton qui va permettre de mettre le jeu en pause. Cela a été implémenté avec :

```
auto pauseItem =
        MenuItemImage::create("pauseb.png","pauseb.png",
            CC_CALLBACK_1(Lvl1Scene::GoToPauseScene, this));
```

La fonction **CC_CALLBACK** permet d'appeler la fonction **GoToPauseScene** qui change la scène vers la scène **PauseScene**.

Ce bouton prend la position haute à gauche avec :

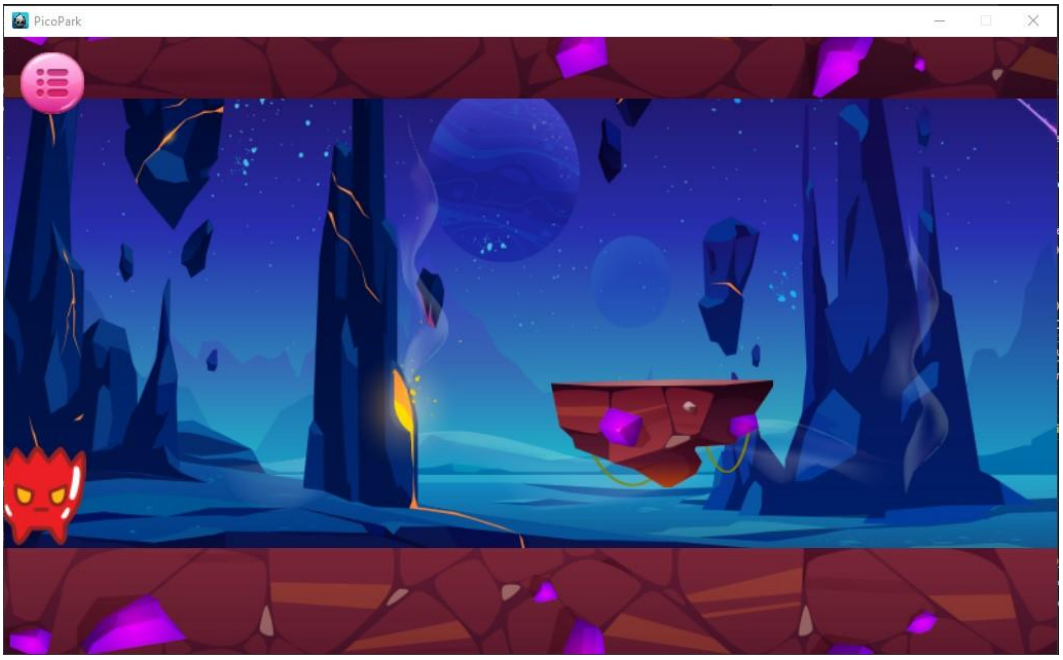
```
pauseItem->setPosition(Point(pauseItem->getContentSize().width -
       (pauseItem->getContentSize().width / 4) + origin.x,
       visibleSize.height - pauseItem->getContentSize().height +
       (pauseItem->getContentSize().width / 4) + origin.y));
```

De même pour le joueur, l'arrière-plan a été ajoutée prendra la ''couche'' (layer)-1

```
   auto backgroundSprite = Sprite::create("backg2.png");
    backgroundSprite->setPosition(Point((visibleSize.width / 2) +
        origin.x, (visibleSize.height / 2) + origin.y + 40));
    this->addChild(backgroundSprite, -1);

    auto backgroundSprite1 = Sprite::create("backg1.png");
    backgroundSprite1->setPosition(Point((visibleSize.width / 2) + origin.x,
(visibleSize.height / 2) + origin.y - 130));
    this->addChild(backgroundSprite1, -1);
```

Game scene 2

Cette scène représente le 2eme niveau du jeu. La structure du niveau est la suivante :



Dans cette scène, et au niveau du fichier **GameScene2.cpp**, un Sprite a été en utilisant la ligne suivante :

```
auto sprite = Sprite::create("player.png");
    sprite->setPosition(16, 110);
    auto physicsBody = PhysicsBody::createCircle(sprite-
>getContentSize().width / 2);
    physicsBody->setDynamic(true);
    sprite->setPhysicsBody(physicsBody);
    this->addChild(sprite);
```

Cette Sprite va prendre **la position (16,110)** qui représente sa position initiale puis ajoutée à la scène.

On a aussi un bouton qui va permettre de mettre le jeu en pause. Cela a été implémenté avec :

```
  auto pauseItem =
        MenuItemImage::create("pauseb.png", "pauseb.png",
            CC_CALLBACK_1(Lvl2Scene::GoToPauseScene, this));
```

La fonction **CC_CALLBACK** permet d'appeler la fonction **GoToPauseScene** qui change la scène vers la scène **PauseScene**.

Ce bouton prend la position haute à gauche avec :

```
pauseItem->setPosition(Point(pauseItem->getContentSize().width -
        (pauseItem->getContentSize().width / 4) + origin.x,
        visibleSize.height - pauseItem->getContentSize().height +
        (pauseItem->getContentSize().width / 4) + origin.y));
```

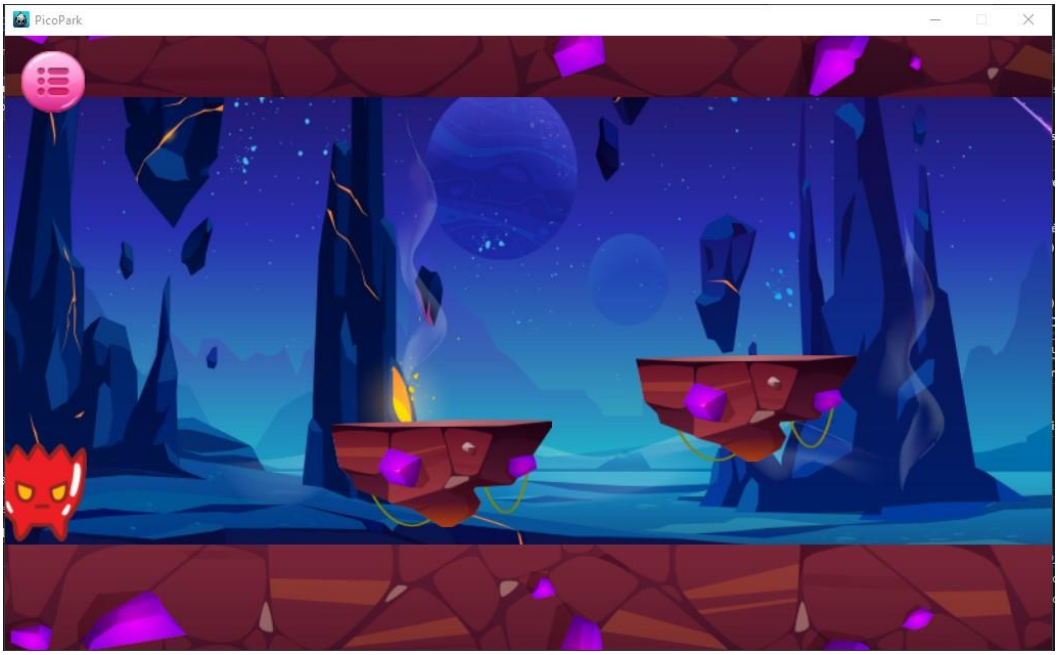
De même pour le joueur, l'arrière-plan a été ajoutée et prendra la ''couche'' (layer) -1 de l'écran avec la ligne suivante :

```
auto backgroundSprite = Sprite::create("backg2.png");

    backgroundSprite->setPosition(Point((visibleSize.width / 2) +
        origin.x, (visibleSize.height / 2) + origin.y + 40));
    this->addChild(backgroundSprite, -1);
```

```
    auto backgroundSprite1 = Sprite::create("backg1.png");
    backgroundSprite1->setPosition(Point((visibleSize.width / 2) + origin.x ,
(visibleSize.height / 2) + origin.y - 130));
    this->addChild(backgroundSprite1, -1);
    auto backgroundSprite11 = Sprite::create("backg1.png");
    backgroundSprite11->setPosition(Point((visibleSize.width / 2) + origin.x,
(visibleSize.height / 2) + origin.y +150));
    this->addChild(backgroundSprite11, -1);

    auto rock = Sprite::create("backgg3.png");
    rock->setPosition(300, 120);
    this->addChild(rock, -1);
```

- *Game scene 3*

Cette scène représente le 3eme niveau du jeu. La structure du niveau est la suivante :



Dans cette scène, et au niveau du fichier **GameScene3.cpp**, une Sprite a été en utilisant les lignes suivantes :

```
   auto sprite = Sprite::create("player.png");
    sprite->setPosition(16, 110);
    auto physicsBody = PhysicsBody::createCircle(sprite-
>getContentSize().width / 2);
    physicsBody->setDynamic(true);
    sprite->setPhysicsBody(physicsBody);
    this->addChild(sprite);
```

Cette Sprite va prendre **la position (16,110)** qui représente sa position initiale puis ajoutée à la scène.

On a aussi un bouton qui va permettre de mettre le jeu en pause. Cela a été implémenté avec :

```
 auto pauseItem =
        MenuItemImage::create("pauseb.png", "pauseb.png",
            CC_CALLBACK_1(Lvl3Scene::GoToPauseScene, this));
```

La fonction **CC_CALLBACK** permet d'appeler la fonction **GoToPauseScene** qui change la scène vers la scène **PauseScene**.

Ce bouton prend la position haute à gauche avec :

```
pauseItem->setPosition(Point(pauseItem->getContentSize().width -
        (pauseItem->getContentSize().width / 4) + origin.x,
        visibleSize.height - pauseItem->getContentSize().height +
        (pauseItem->getContentSize().width / 4) + origin.y));
```

De même pour le joueur, l'arrière-plan a été ajoutée et prendra la ''couche'' (layer) -1 de l'écran avec la ligne suivante :

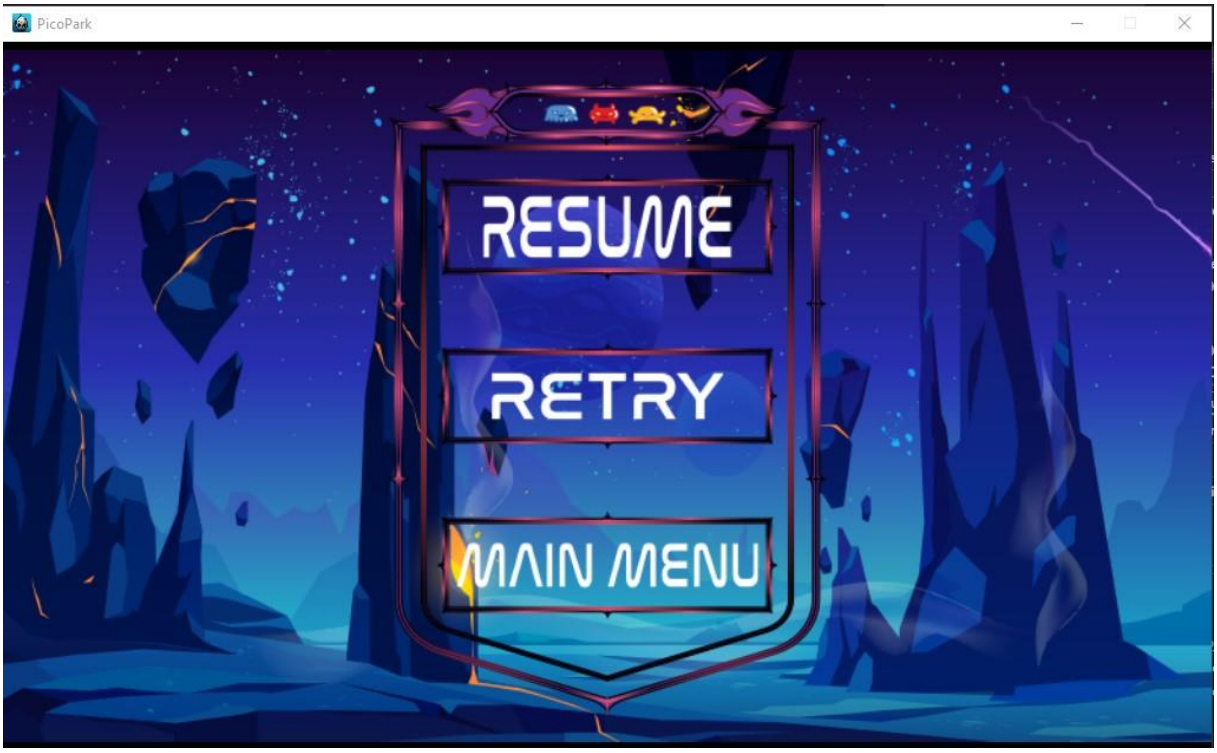
```
auto backgroundSprite = Sprite::create("backg2.png");

    backgroundSprite->setPosition(Point((visibleSize.width / 2) +
        origin.x, (visibleSize.height / 2) + origin.y + 40));
    this->addChild(backgroundSprite, -1);

    auto backgroundSprite1 = Sprite::create("backg1.png");
    backgroundSprite1->setPosition(Point((visibleSize.width / 2) + origin.x,
(visibleSize.height / 2) + origin.y - 130));
    this->addChild(backgroundSprite1, -1);
    auto backgroundSprite11 = Sprite::create("backg1.png");
    backgroundSprite11->setPosition(Point((visibleSize.width / 2) + origin.x,
(visibleSize.height / 2) + origin.y + 150));
    this->addChild(backgroundSprite11, -1);

    auto rock = Sprite::create("backgg3.png");
    rock->setPosition(340, 130);
    this->addChild(rock, -1);
    auto rock1 = Sprite::create("backgg3.png");
    rock1->setPosition(200, 100);
    this->addChild(rock1, -1);
```

- *Pause scene*

Cette Scène est sous forme d'un menu avec une arrière-plan. Elle peut être accessible depuis la scène du jeu avec le tamplate du fonction suivante :

```
void LvlXScene ::GoToPauseScene(cocos2d::Ref* pSender)
{
    auto scene = PauseScene::createScene();
    Director::getInstance()->pushScene(scene);
}
```

L'arriéré plan est sous forme d'une Sprite (image) :



Cette scène est composée d'un menu qui permet de retourner (continuer le niveau), réessayer depuis le ier niveau ou bien revenir au menu d'accueil
Retourner au niveau :

```
auto resumeItem =
        MenuItemImage::create("resume.png",
            "bk.png",
            CC_CALLBACK_1(PauseScene::Resume, this));
```

La fonction précédente permet de revenir vers la scène précédente en supprimant celle actuelle Réessayer :

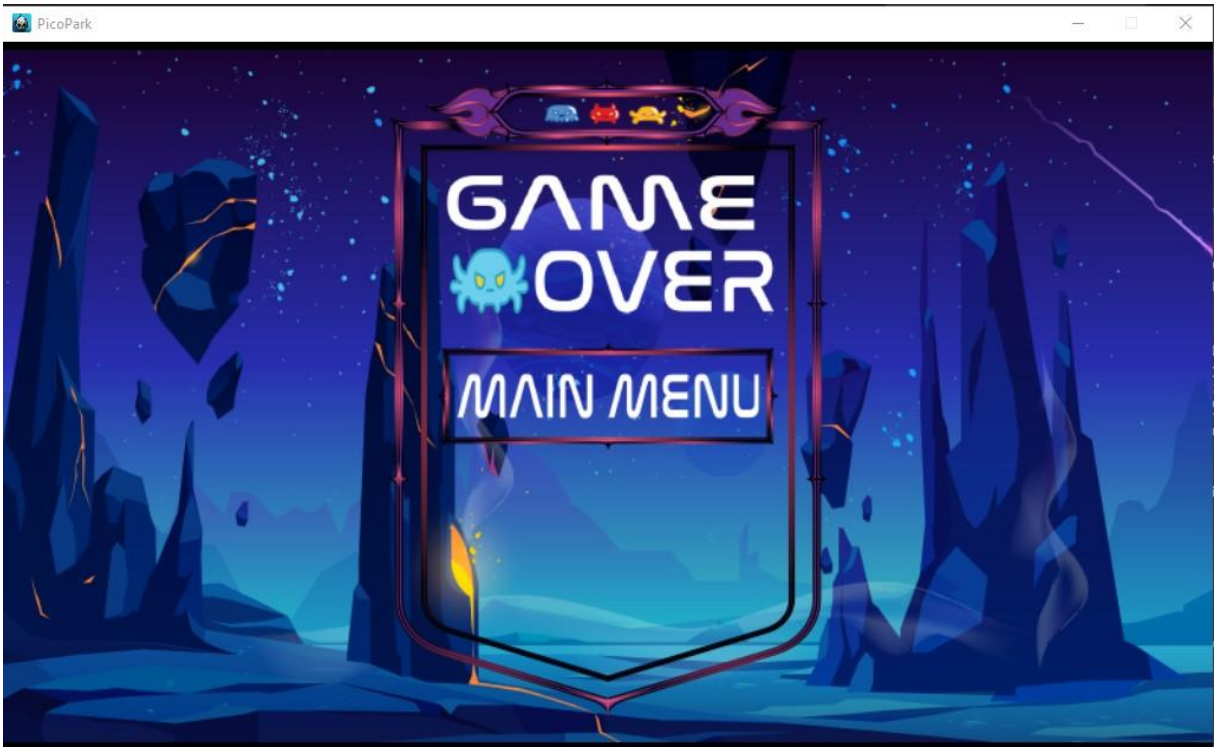
```
auto retryItem =

        MenuItemImage::create("retry.png",
            "bk.png",
            CC_CALLBACK_1(PauseScene::Retry, this));
```

La fonction précédente permet de revenir relancer le niveau 1 en remplaçant la scène actuelle avec celle du niveau 1 Revenir au menu d'accueil :

```
auto mainMenuItem =
        MenuItemImage::create("main menu.png",
            "bk.png",
            CC_CALLBACK_1(PauseScene::GoToMainMenu, this));
    auto menu = Menu::create(resumeItem, retryItem, mainMenuItem,
        NULL);
    menu->alignItemsVerticallyWithPadding(visibleSize.height / 18);
    this->addChild(menu);
```

- *Game over scene*

Cette scène est accessible seulement après avoir terminé le jeu.



Elle se compose d'un arrière-plan qui félicite et remercie le joueur d'avoir joué à ce jeu. L'arrière-plan est créé et positionnée avec :

```
auto backgroundSprite = Sprite::create("backg2.png");
    backgroundSprite->setPosition(Point((visibleSize.width / 2) +
        origin.x, (visibleSize.height / 2) + origin.y ));
    this->addChild(backgroundSprite, -1);

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

3-Le déplacement du joueur :

Pour permettre de déplacer le joueur avec le clavier, un écouteur (listener) a été créé afin de trouver le bouton clique du clavier puis effectuer un déplacement selon la position du joueur (afin d'éviter l'utilisation des collisions). C'est-à-dire on limite le déplacement du joueur a seulement les déplacements possibles. L'écouteur :

```
 auto eventListener = EventListenerKeyboard::create();
    eventListener->onKeyPressed = [](EventKeyboard::KeyCode keyCode, Event*
event) {
        int offsetX = 0, offsetY = 0;
        float x, y;
```

Les variables **offsetX** et **offsetY** déterminent le déplacement du joueur selon le 2 axes X et Y, ainsi que les variables x et y vont servir à stocker la position du joueur. Exemple du déplacement à droite du niveau 1 :

```
 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;
                x = event->getCurrentTarget()->getPositionX();

                if (x>=420) {
                    // transition between level 1 and level 2
                    auto scene = Lvl2Scene::createScene();
```

```
                    Director::getInstance()-
>replaceScene(TransitionFade::create(2, scene));
            }

        break;
        case EventKeyboard::KeyCode::KEY_UP_ARROW:
        case EventKeyboard::KeyCode::KEY_W:
        case EventKeyboard::KeyCode::KEY_SPACE:
                offsetY = +30;
        break;
        case EventKeyboard::KeyCode::KEY_DOWN_ARROW:
        case EventKeyboard::KeyCode::KEY_S:
                offsetY = -30;
        break;
```

Dans le code ci-dessus, on va recevoir la position actuelle du joueur puis vérifier s'il est possible de se déplacer à droite. Si oui, la variable de l'axe des abscisses va recevoir la valeur 172 (valeur change celon le niveau) puis dessiner un segment à partir de la position initiale du joueur avant mouvement jusqu'à sa position après. L'arrière-plan blanche initialisée précédent va couvrir la partie dans laquelle se déplace le joueur et le segment dessiné sera dans la couche entre l'arrière-plan de niveau et le joueur puisque cette partie de l'arrière-plan est vide.

```
  auto moveTo = MoveTo::create(0.3, Vec2(event->getCurrentTarget()-
>getPositionX() + offsetX, event->getCurrentTarget()->getPositionY() +
offsetY));

        event->getCurrentTarget()->runAction(moveTo);
```

Le code ci-dessus va exécuter le mouvement avec la fonction **MoveTo** et affecter ce déplacement à la Sprite (le joueur)

4-Le son :

Lors du démarrage du jeu, un fichier audio est lancé avec la commande suivante :

```
void AppDelegate::applicationDidEnterBackground() {
    Director::getInstance()->stopAnimation();

#if USE_AUDIO_ENGINE
    AudioEngine::pauseAll();
#endif
}

// this function will be called when the app is active again
void AppDelegate::applicationWillEnterForeground() {
    Director::getInstance()->startAnimation();

#if USE_AUDIO_ENGINE
    AudioEngine::resumeAll();
#endif
}
```

Cette commande appartient à la bibliothèque des audios inclue dans cocos2d est déclarée au début du fichier **MainMenuScene.cpp**

```
#include "audio/include/AudioEngine.h"
```