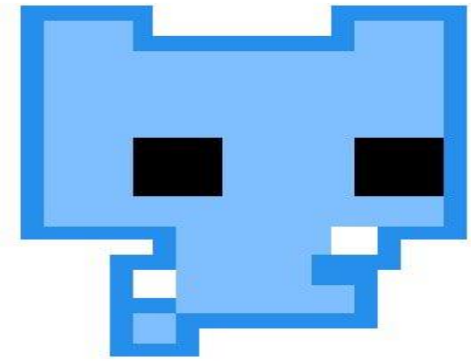
جامعة عبد المالك السعدي
Université Abdelmalek Essaadi

FACULTE DES SCIENCES ET TECHNIQUES
TANGER
كلية العلوم والتقنيات طنجة

Rapport : jeu Cocos2D « C++ »

PICO PARK



Encadré par :

Pr . ELAACHAk LOTFI

Réalisé par :

ECHRIYAH Oumaima grp1

BENABDELLAH Badr grp1

Année universitaire 2022-2023

Sommaire :

1- LES OUTILS :

-adobe photoshop

-pixilart

-Microsoft Visual Studio

- C++

- Cocos2D

2- Les étapes de réalisation du jeu :

2-1 création de la scène d'ouverture

fichiers concernés : HelloWorldScene.h et HelloWorldScene.cpp

- déclaration de la classe dérivée “HelloWorld” :

```
#include "cocos2d.h"
class HelloWorld : public cocos2d::Scene
{
public:
    static cocos2d::Scene* createScene();

    virtual bool init();
    void start(Ref* pSender);
    // implement the "static create()" method manually
    CREATE_FUNC(HelloWorld);
};
```

- création de la scène « HelloWorld » :

```
Scene* HelloWorld::createScene()
{
    return HelloWorld::create();
}
```

- initialisation de notre scène avec l’ajout d un background et un bouton ‘start ‘ comme item d’un menu :

```
bool HelloWorld::init()
{
1. super init first

    if (!Scene::init()) {
        return false;
    }
    Size visiblesize = Director::getInstance()->getVisibleSize();
    Point origin = Director::getInstance()->getVisibleOrigin();

    //partie background

    auto mysprite = Sprite::create("pic1.png");
    mysprite->setPosition(Point((visiblesize.width / 2) + origin.x, (visiblesize.height / 2 + origin.y)));
    mysprite->setScale(0.8 , 0.8);
    this->addChild(mysprite);
```

```
    //partie bouton start

    auto menu_item = MenuItemImage::create("start.png", "start.png", CC_CALLBACK_1(HelloWorld::start,
this));
    menu_item->setPosition((Point((visiblesize.width / 2) + origin.x, (visiblesize.height / 2 + origin.y - 10))));
    menu_item->setScale(0.2, 0.2);
    auto* menu = Menu::create(menu_item, NULL);
    menu->setPosition(Vec2(0, 0));
    this->addChild(menu);

    return true;
}
```

-définir la fonction void start qui permet de passer à une autre scène :

```
//fonction cliquer sur bouton start pour passer a la scene Levels

void HelloWorld::start(cocos2d::Ref* pSender) {
    auto scene = Levels::createScene();
    Director::getInstance()->pushScene(TransitionFade::create(0.2, scene)); // je peux faire ghir scene .
}
```

Résultat :

- apparition la scène ’ levels’ en cliquant sur ‘start’ :

2-2 procédure de création de la scène ‘ Levels’ :

fichiers concernés : Levels.h et Levels.cpp

- declaration de la classe dérivée “Levels” :

```
#include "cocos2d.h"

class Levels : public cocos2d::Scene
{
public:
    static cocos2d::Scene* createScene();
    virtual bool init();
    void level1(Ref* pSender);
    void level2(Ref* pSender);
    void level3(Ref* pSender);

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

```
};
```

- création de la scène « Levels » :

```
Scene* Levels::createScene()
{
    return Levels::create();
}
```

- initialisation de notre scène avec l'ajout d un background et un menu contenant des items :

```
bool Levels::init()
{
    if (!Scene::init()) {
        return false;
    }
    Size visiblesize = Director::getInstance()->getVisibleSize();
    Point origin = Director::getInstance()->getVisibleOrigin();

    //partie background

    auto mysprite = Sprite::create("whitebac.png");
    mysprite->setPosition(Point((visiblesize.width / 2) + origin.x, (visiblesize.height / 2 + origin.y + 46)));
    mysprite->setScale(8, 8);
    this->addChild(mysprite);

    //partie boutons levels

    //button lvl1

    auto menu_item = MenuItemImage::create("level01.png", "level01selected.png",
CC_CALLBACK_1(Levels::level1, this));
    menu_item->setPosition(Point(visiblesize.width / 2, (visiblesize.height / 4) * 3));


    //button lvl2

    auto menu_item_2 = MenuItemImage::create("level02.png", "level02selected.png",
CC_CALLBACK_1(Levels::level2, this));
   menu_item_2->setPosition(Point(visiblesize.width / 2, (visiblesize.height / 4) * 2));


    //button lvl3
```

```
    auto menu_item_3 = MenuItemImage::create("level03.png", "level03selected.png",
CC_CALLBACK_1(Levels::level3, this));
    menu_item_3->setPosition(Point(visiblesize.width / 2, (visiblesize.height / 4) * 1));

// création de menu
    auto* menu = Menu::create(menu_item, menu_item_2, menu_item_3, NULL);
    menu->setPosition(Point(0, 0));

    this->addChild(menu);

    return true} ;
```

-définir la fonction 'void Level1 ' qui permet de passer à la scene 'Level1' :

```
//fonction cliquer sur bouton lvl1
void Levels::level1(cocos2d::Ref* pSender) {
    auto scene1 = Level1::createScene();
    Director::getInstance()->replaceScene(TransitionFade::create(0.2, scene1));
}
```

2-3 procédure de création d'un Level exemple : « level1 » :

fichiers concernés : Level1.h et Level1.cpp

- declaration de la classe dérivée "Level1" :

```
#include "cocos2d.h"

class Level1 : public cocos2d::Scene

{
public:
    static cocos2d::Scene* createScene();

    virtual bool init();
    // implement the "static create()" method manually
    CREATE_FUNC(Level1);


private:
    cocos2d::PhysicsWorld* sceneWorld;

    bool onContactBegin(cocos2d::PhysicsContact& contact);
    void SetPhysicsWorld(cocos2d::PhysicsWorld* world) { sceneWorld = world; };

};
```

- création de la scène « Level1 » :

```
Scene* Level1::createScene()
{
  auto scene = Scene::createWithPhysics();
    scene->getPhysicsWorld()->setDebugDrawMask(PhysicsWorld::DEBUGDRAW_NONE);
    // 'layer' is an autorelease object
    auto layer = Level1::create();
    layer->SetPhysicsWorld(scene->getPhysicsWorld());

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

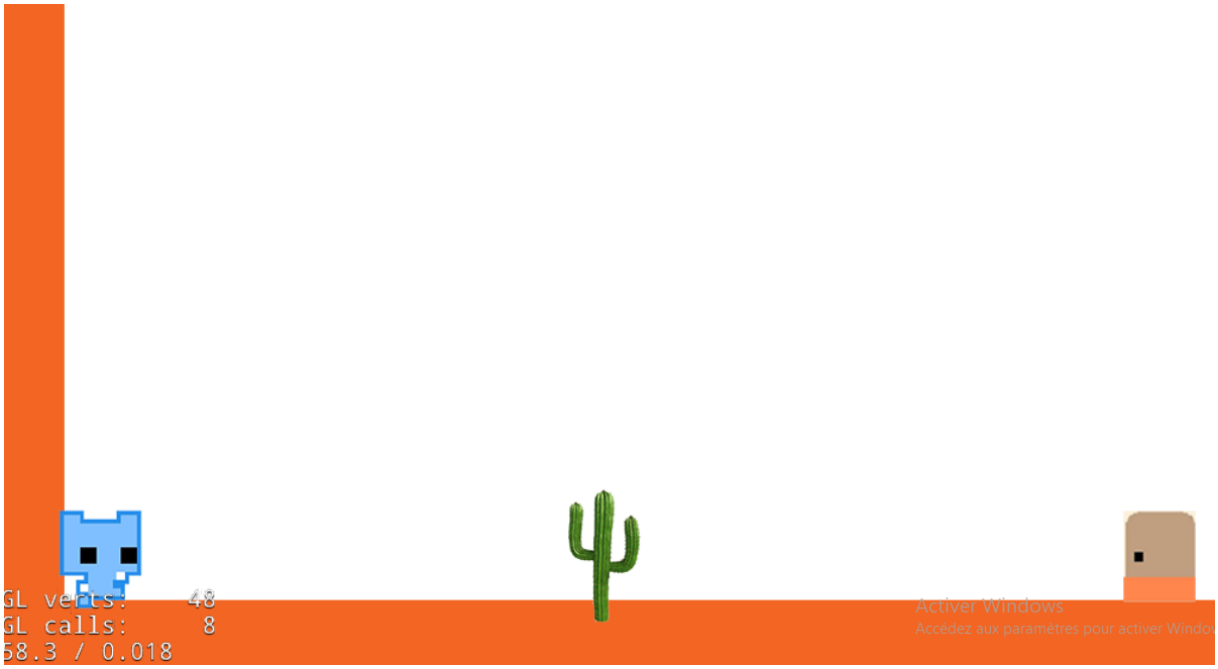
- initialisation de notre scène avec l'ajout d un background et d' un caractère qui se déplace selon le clavier et en utilisant le principe de collision pour les obstacles et pour le passage au level suivant , les détails sont fournis dans le code via des commentaires:

```
bool Level1::init()
{
    if (!Scene::init()) {
        return false;
    }
    Size visiblesize = Director::getInstance()->getVisibleSize();
    Point origin = Director::getInstance()->getVisibleOrigin();

    //partie background

    auto mysprite = Sprite::create("level1bac.png");
   mysprite->setPosition(Point((visiblesize.width / 2) + origin.x, (visiblesize.height / 2 + origin.y)));
    mysprite->setScale(0.8, 0.8);
    this->addChild(mysprite);

//-----------------------------------------------------------------
    //partie player

    auto character = Sprite::create("character.png");
    auto action = Place::create(Point(100, 100));
    character->runAction(action);
    // physic aspect
    auto characterBody = PhysicsBody::createBox(character->getContentSize(), PhysicsMaterial(0, 2, 0));
    character->setPhysicsBody(characterBody);
    characterBody->setDynamic(false);
    this->addChild(character);
    // for the collision part
    character->setTag(1);
    characterBody->setCollisionBitmask(1);
    characterBody->setCategoryBitmask(1);
    characterBody->setContactTestBitmask(1);
```

```
//-----------------------------------------------------------------------
// the door object
    auto door = Sprite::create("door.png");
    auto action1 = Place::create(Point(1150, 110));
    door->runAction(action1);
    // physic aspect
    auto doorBody = PhysicsBody::createBox(door->getContentSize(), PhysicsMaterial(0, 2, 0));
    door->setPhysicsBody(doorBody);
    doorBody->setDynamic(false);
    this->addChild(door);
    // for thr collision part 3
    door->setTag(3);
    doorBody->setCollisionBitmask(3);
    doorBody->setCategoryBitmask(3);
    doorBody->setContactTestBitmask(3);




   //-----------------------------------------------------------------------
    // obstacle1
    auto obstacle1 = Sprite::create("obstacle11.png");
    auto action2 = Place::create(Point(600, 110));
    obstacle1->runAction(action2);
    // physic aspect
    auto obstacle1Body = PhysicsBody::createBox(obstacle1->getContentSize(), PhysicsMaterial(0, 2, 0));
    obstacle1->setPhysicsBody(obstacle1Body);
    obstacle1Body->setDynamic(false);
    this->addChild(obstacle1);
    // for thr collision part 3
    obstacle1->setTag(4);
    obstacle1Body->setCollisionBitmask(4);
    obstacle1Body->setCategoryBitmask(4);
    obstacle1Body->setContactTestBitmask(4);

//-----------------------------------------------------------------
    //keyborad movement
    auto eventListener = EventListenerKeyboard::create();

    eventListener->onKeyReleased = [character](EventKeyboard::KeyCode keyCode, Event* event) {

        if (keyCode == EventKeyboard::KeyCode::KEY_UP_ARROW) {
            // sequence d'actions
            auto jump = JumpBy::create(0.7f, Vec2(80, 50), 60.0f, 1);

            auto moveBy = MoveBy::create(0.8, Vec2(0, -50));
            auto seq = Sequence::create(jump,moveBy, nullptr);

            // run it
            character->runAction(seq);

        }
        if (keyCode == EventKeyboard::KeyCode::KEY_RIGHT_ARROW) {
```

```
        // Move sprite to position to right in 2 seconds.
        auto moveBy = MoveBy::create(2, Vec2(30,0));
            character->runAction(moveBy);

    }
    if (keyCode == EventKeyboard::KeyCode::KEY_LEFT_ARROW) {
        // Move sprite to position left in 2 seconds.
        auto moveBy = MoveBy::create(2, Vec2(-20, 0));
        character->runAction(moveBy);


    }
};

Director::getInstance()->getEventDispatcher()->addEventListenerWithSceneGraphPriority(eventListener,
this);


// add an event listener for contcat events

auto contactListener = EventListenerPhysicsContact::create();
contactListener->onContactBegin = [character](PhysicsContact& contact) {


    PhysicsBody* x = contact.getShapeA()->getBody();
    PhysicsBody* y = contact.getShapeB()->getBody();


    if (1 == x->getCollisionBitmask() && 3 == y->getCollisionBitmask() || 3 == x->getCollisionBitmask()
&& 1 == y->getCollisionBitmask()) {

        auto scene = Level2::createScene();
        Director::getInstance()->pushScene(TransitionFade::create(0.5, scene));

    }

    if (1 == x->getCollisionBitmask() &&4== y->getCollisionBitmask() ||4 == x->getCollisionBitmask() &&
1 == y->getCollisionBitmask()) {

        auto scene = Level2::createScene();
        Director::getInstance()->pushScene(TransitionFade::create(0.5, scene));

    }



    return true;
};
this->getEventDispatcher()->addEventListenerWithSceneGraphPriority(contactListener, this);
return true;
}
```

-Résultat de Level 1 :

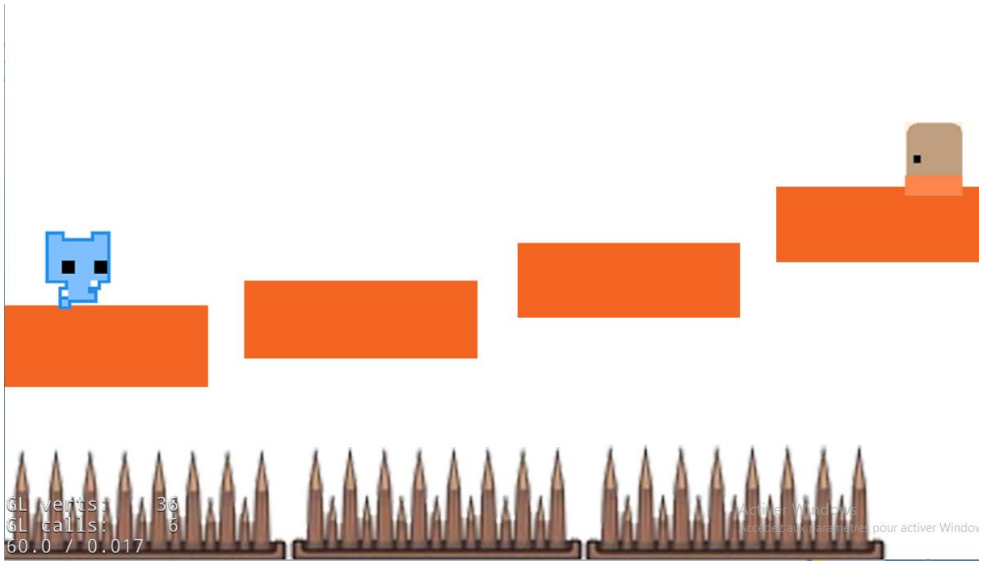


- même procédure et étapes s’appliquent pour les autres levels

En modifiant principalement les images et quelques paramètres .

Resultat level 2 :

Resultat level 3 :



Scène finale :

Well Done !

iL verts: 6
iL calls: 1
50.0 / 0.017

3- Bibliographie.

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

https://youtube.com/playlist?list=PLRtjMdoYXLf4od_bOKN3WjAPr7snPXzoe

https://youtube.com/playlist?list=PLRtjMdoYXLf7GSD9crXIjMQiRuIZ7mUVp