Projet : Jeux vidéo 2D avec Cocos2D

NASSER BILAL

EL HOUITI SOUHAILA

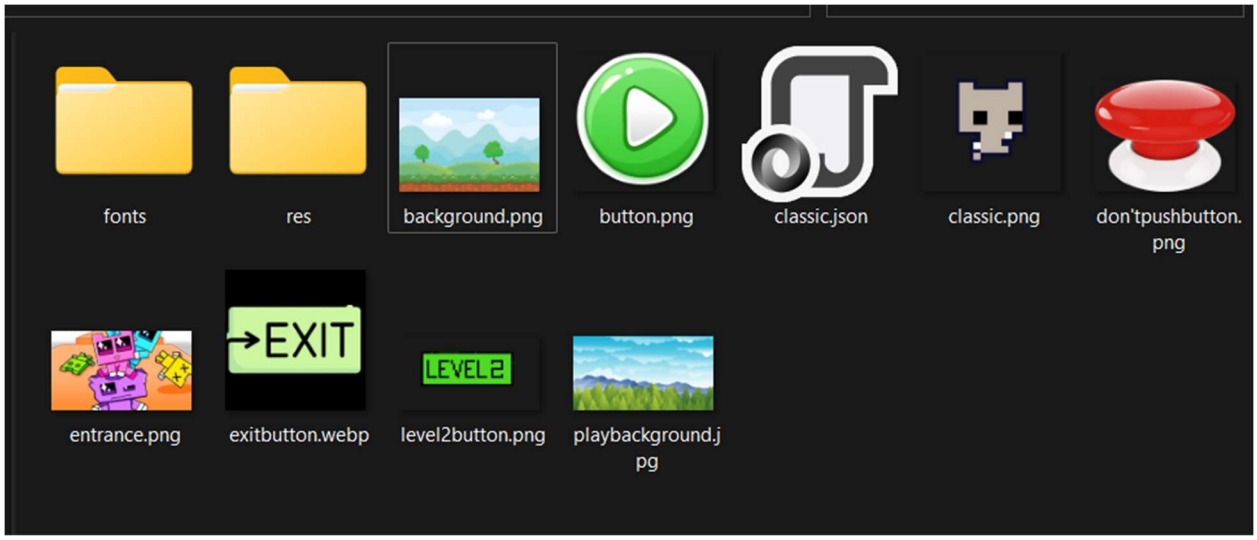
- ENCADRE PAR

Ikram Ben ABDELOUAHAB | Lotfi ELAACHAK

Les classes

AppDelegate.cpp	2023-01-05 22:40	CPP File	4 KB
AppDelegate.h	2022-12-21 00:58	C/C++ Header	1 KB
Definitions.h	2023-01-05 16:54	C/C++ Header	1 KB
GameScene.cpp	2023-01-05 19:11	CPP File	3 KB
GameScene.h	2023-01-05 18:14	C/C++ Header	1 KB
Level1Scene.cpp	2023-01-05 20:09	CPP File	3 KB
Level1Scene.h	2023-01-05 18:14	C/C++ Header	1 KB
MainMenuScene.cpp	2023-01-05 20:09	CPP File	2 KB
MainMenuScene.h	2023-01-04 21:34	C/C++ Header	1 KB
SplashScene.cpp	2023-01-05 16:56	CPP File	2 KB
SplashScene.h	2022-12-29 13:47	C/C++ Header	1 KB

Les resources



AppDelegate.h

```cpp
#ifndef  _APP_DELEGATE_H_
#define  _APP_DELEGATE_H_

#include "cocos2d.h"

/**
@brief    The cocos2d Application.

Private inheritance here hides part of interface from Director.
*/
class  AppDelegate : private cocos2d::Application
{
public:
    AppDelegate();
    virtual ~AppDelegate();

    virtual void initGLContextAttrs();

    /**
    @brief    Implement Director and Scene init code here.
    @return true    Initialize success, app continue.
    @return false   Initialize failed, app terminate.
    */
    virtual bool applicationDidFinishLaunching();

    /**
    @brief  Called when the application moves to the background
    @param  the pointer of the application
    */
    virtual void applicationDidEnterBackground();

    /**
    @brief  Called when the application reenters the foreground
    @param  the pointer of the application
    */
    virtual void applicationWillEnterForeground();
};

#endif // _APP_DELEGATE_H_
```

AppDelegate.cpp

```cpp
#include "AppDelegate.h"
#include "SplashScene.h"

// #define USE_AUDIO_ENGINE 1

#if USE_AUDIO_ENGINE
#include "audio/include/AudioEngine.h"
using namespace cocos2d::experimental;
#endif

USING_NS_CC;

static cocos2d::Size designResolutionSize = cocos2d::Size(480, 320);
static cocos2d::Size smallResolutionSize = cocos2d::Size(480, 320);
static cocos2d::Size mediumResolutionSize = cocos2d::Size(1024, 768);
static cocos2d::Size largeResolutionSize = cocos2d::Size(2048, 1536);

AppDelegate::AppDelegate()
{
}

AppDelegate::~AppDelegate()
{
#if USE_AUDIO_ENGINE
    AudioEngine::end();
#endif
}

// if you want a different context, modify the value of glContextAttrs
// it will affect all platforms
void AppDelegate::initGLContextAttrs()
{
    // set OpenGL context attributes: red,green,blue,alpha,depth,stencil,
    GLContextAttrs glContextAttrs = {8, 8, 8, 8, 24, 8, 0};

    GLView::setGLContextAttrs(glContextAttrs);
}
```

```cpp
// if you want to use the package manager to install more packages,
// don't modify or remove this function
static int register_all_packages()
{
    return 0; //flag for packages manager
}
bool AppDelegate::applicationDidFinishLaunching() {
    // initialize director
    auto director = Director::getInstance();
    auto glview = director->getOpenGLView();
    if(!glview) {
#if (CC_TARGET_PLATFORM == CC_PLATFORM_WIN32) || (CC_TARGET_PLATFORM == CC_PLATFORM_MAC) || (CC_TARGET_PLATFORM == CC_PLATFORM_LINUX)
        glview = GLViewImpl::createWithRect("picopark", cocos2d::Rect(0, 0, designResolutionSize.width, designResolutionSize.height));
#else
        glview = GLViewImpl::create("picopark");
#endif
        director->setOpenGLView(glview);
    }

    // turn on display FPS
    director->setDisplayStats(true);

    // set FPS. the default value is 1.0/60 if you don't call this
    director->setAnimationInterval(1.0f / 60);

    // Set the design resolution
    glview->setFrameSize(1280, 720);
    glview->setDesignResolutionSize(designResolutionSize.width, designResolutionSize.height, ResolutionPolicy::NO_BORDER);
    auto frameSize = glview->getFrameSize();
    // if the frame's height is larger than the height of medium size.
    if (frameSize.height > mediumResolutionSize.height)
    {
    director->setContentScaleFactor(MIN(largeResolutionSize.height/designResolutionSize.height,largeResolutionSize.width/designResolutionSize.width));
    }
    // if the frame's height is larger than the height of small size.
    else if (frameSize.height > smallResolutionSize.height)
    {
    director->setContentScaleFactor(MIN(mediumResolutionSize.height/designResolutionSize.height, mediumResolutionSize.width/designResolutionSize.width));
    }
```

```cpp
    {
    director->setContentScaleFactor(MIN(mediumResolutionSize.height/designResolutionSize.height, mediumResolutionSize.width/designResolutionSize.width));
    }
    // if the frame's height is smaller than the height of medium size.
    else
    {
    director->setContentScaleFactor(MIN(smallResolutionSize.height/designResolutionSize.height, smallResolutionSize.width/designResolutionSize.width));
    }
    register_all_packages();

    // create a scene. it's an autorelease object
    auto scene = SplashScene::createScene();

    // run
    director->runWithScene(scene);

    return true;
}

// This function will be called when the app is inactive. Note, when receiving a phone call it is invoked.
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
}
```

Level1Scene.h

```
#ifndef __LEVEL1_SCENE_H__
#define __LEVEL1_SCENE_H__

#include "cocos2d.h"
class Level1Scene : public cocos2d::Layer
{
public:
    static cocos2d::Scene* createScene();

    virtual bool init();
    CREATE_FUNC(Level1Scene);


private:
    void SetPysicsWorld(cocos2d::PhysicsWorld *world) { sceneworld = world; };
    void firstground(float dt);
    cocos2d::PhysicsWorld *sceneworld;
    cocos2d::Sprite* pico;

    void GoToGameScene(cocos2d::Ref* sender);

};

#endif // __LEVEL1_SCENE_H__
```

Level1Scene.cpp

```cpp
#include "Level1Scene.h"
#include "GameScene.h"
#include "Definitions.h"

USING_NS_CC;

Scene* Level1Scene::createScene()
{



    auto scene = Scene::createWithPhysics();
    PhysicsWorld* world = scene->getPhysicsWorld();

    auto layer = Level1Scene::create();
    layer->SetPysicsWorld( scene->getPhysicsWorld( ) );

    scene->addChild(layer);

    return scene;
}


// on "init" you need to initialize your instance
bool Level1Scene::init()
{
    ////////////////////////////
    // 1. super init first
    if (!Layer::init())
    {

        return false;
    }

    auto visibleSize = Director::getInstance()->getVisibleSize();
    Vec2 origin = Director::getInstance()->getVisibleOrigin();
    auto background = Sprite::create("background.png");
    background->setScaleX(1.3);
    background->setPosition(250,165);
    this->addChild(background);
```

```
auto pico = Sprite::create("classic.png");
pico->setPosition(20,87);
pico->setScale(0.5);



this->addChild(pico, 1);


auto eventListner = EventListenerKeyboard::create();
eventListner->onKeyPressed = [](EventKeyboard::KeyCode KeyCode, Event* event) {
    Vec2 loc = event->getCurrentTarget()->getPosition();
    switch (KeyCode)
    {
    case EventKeyboard::KeyCode::KEY_LEFT_ARROW:
    case EventKeyboard::KeyCode::KEY_A:
        event->getCurrentTarget()->runAction(MoveBy::create(0.09f, Vec2(-20, 0)));

        break;
    case EventKeyboard::KeyCode::KEY_RIGHT_ARROW:
    case EventKeyboard::KeyCode::KEY_D:
        event->getCurrentTarget()->runAction(MoveBy::create(0.09f, Vec2(20, 0)));
        break;

    case EventKeyboard::KeyCode::KEY_SPACE:
    case EventKeyboard::KeyCode::KEY_W:

        event->getCurrentTarget()->runAction(JumpBy::create(0.5, Vec2(30, 0), 30, 1));
        break;
    case EventKeyboard::KeyCode::KEY_DOWN_ARROW:
    case EventKeyboard::KeyCode::KEY_S:
        event->getCurrentTarget()->runAction(JumpBy::create(0.5, Vec2(-30, 0), 30, 1));
        break;
    }

};
```

```
this->_eventDispatcher->addEventListenerWithSceneGraphPriority(eventListner, pico);
this->scheduleUpdate();


Rect frame = Rect(0, 0, 01000, 1000);
auto followplayer = Follow::create(pico, frame);
this->runAction(followplayer);

auto playbtn = MenuItemImage::create("level2button.png","level2button.png", CC_CALLBACK_1(Level1Scene::GoToGameScene, this));
playbtn->setScale(0.5);
playbtn->setPosition(Point(450,250));

auto menu = Menu::create(playbtn,NULL);
menu->setPosition(Point::ZERO);
this->addChild(menu);
return true;


id Level1Scene::GoToGameScene(cocos2d::Ref* sender)

auto scene = GameScene::createScene();
Director::getInstance()->replaceScene(TransitionFade::create(TRANSITION_TIME, scene));
```

GameScene.h

```
#ifndef __GAME_SCENE_H__
#define __GAME_SCENE_H__

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

    virtual bool init();
    CREATE_FUNC(GameScene);

private:
    void SetPysicsWorld(cocos2d::PhysicsWorld* world) { sceneworld = world; };
    void firstground(float dt);
    cocos2d::PhysicsWorld* sceneworld;
    cocos2d::Sprite* pico;

    void GoToGameScene(cocos2d::Ref* sender);

};

#endif // __GAME_SCENE_H__
```

GameScene.cpp

```cpp
#include "GameScene.h"
#include "Definitions.h"

USING_NS_CC;

Scene* GameScene::createScene()
{
    auto scene = Scene::createWithPhysics();
    PhysicsWorld* world = scene->getPhysicsWorld();
    auto layer = GameScene::create();
    layer->SetPysicsWorld(scene->getPhysicsWorld());
    scene->addChild(layer);
    return scene;
}
// on "init" you need to initialize your instance
bool GameScene::init()
{
    ////////////////////////////
    // 1. super init first
    if (!Layer::init())
    {

        return false;
    }
    auto visibleSize = Director::getInstance()->getVisibleSize();
    Vec2 origin = Director::getInstance()->getVisibleOrigin();
    auto background = Sprite::create("background.png");
    background->setScaleX(1.3);
    background->setPosition(250, 165);
    this->addChild(background);

    auto dontpush = Sprite::create("don'tpushbutton.png");
    dontpush->setScale(0.15);
    dontpush->setPosition(200,75);
    this->addChild(dontpush , 2);
```

```
auto exit = Sprite::create("exitbutton.webp");
exit->setScale(0.5);
exit->setPosition(450,250);
this->addChild(exit,2);

auto pico = Sprite::create("classic.png");
pico->setPosition(20, 87);
pico->setScale(0.5);

this->addChild(pico, 2);
auto eventListner = EventListenerKeyboard::create();
eventListner->onKeyPressed = [](EventKeyboard::KeyCode KeyCode, Event* event) {
    Vec2 loc = event->getCurrentTarget()->getPosition();
    switch (KeyCode)
    {
    case EventKeyboard::KeyCode::KEY_LEFT_ARROW:
    case EventKeyboard::KeyCode::KEY_A:
        event->getCurrentTarget()->runAction(MoveBy::create(0.09f, Vec2(-20, 0)));

        break;
    case EventKeyboard::KeyCode::KEY_RIGHT_ARROW:
    case EventKeyboard::KeyCode::KEY_D:
        event->getCurrentTarget()->runAction(MoveBy::create(0.09f, Vec2(20, 0)));
        break;

    case EventKeyboard::KeyCode::KEY_SPACE:
    case EventKeyboard::KeyCode::KEY_W:

        event->getCurrentTarget()->runAction(JumpBy::create(0.5, Vec2(30, 0), 30, 1));
        break;
    case EventKeyboard::KeyCode::KEY_DOWN_ARROW:
    case EventKeyboard::KeyCode::KEY_S:
        event->getCurrentTarget()->runAction(JumpBy::create(0.5, Vec2(-30, 0), 30, 1));
        break;
    }

};
```

```
this->_eventDispatcher->addEventListenerWithSceneGraphPriority(eventListner, pico);
this->scheduleUpdate();


Rect frame = Rect(0, 0, 01000, 1000);
auto followplayer = Follow::create(pico, frame);
this->runAction(followplayer);




return true;
```

MainMenuScene.h

```
#ifndef __MAIN_MENU_SCENE_H__
#define __MAIN_MENU_SCENE_H__

#include "cocos2d.h"

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

    virtual bool init();
    CREATE_FUNC(MainMenuScene);

private:
    void GoToLevel1Scene(cocos2d::Ref* sender);

};

#endif // __MAIN_MENU_SCENE_H__
```

MainMenuScene.cpp

```cpp
#include "MainMenuScene.h"
#include "Level1Scene.h"
#include "Definitions.h"

USING_NS_CC;

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

    auto layer = MainMenuScene::create();

    scene->addChild(layer);

    return scene;
}


// on "init" you need to initialize  your instance
bool MainMenuScene::init()
{
    ////////////////////////////
    // 1. super init first
    if (!Layer::init())
    {

        return false;
    }

    auto visibleSize = Director::getInstance()->getVisibleSize();
    Vec2 origin = Director::getInstance()->getVisibleOrigin();
    //background
    auto background = Sprite::create("playbackground.jpg");
    background->setPosition(Point(visibleSize.width / 2 + origin.x, visibleSize.height / 2 + origin.y));
    this->addChild(background);
    //playItem

    auto playbtn = MenuItemImage::create ("button.png","button.png", CC_CALLBACK_1(MainMenuScene::GoToLevel1Scene, this));
    playbtn->setScale(0.5);
    playbtn->setPosition( Point(visibleSize.width / 2 + origin.x, visibleSize.height / 2 + origin.y) );
```

```cpp
    auto menu = Menu::create(playbtn, NULL);
    menu->setPosition(Point::ZERO);
    this->addChild(menu);

    return true;

void MainMenuScene::GoToLevel1Scene(cocos2d::Ref* sender)

    auto scene = Level1Scene::createScene();
    Director::getInstance()->replaceScene(TransitionFade::create(TRANSITION_TIME, scene));
```

SplashScene.h

```
#ifndef __SPLASH_SCENE_H__
#define __SPLASH_SCENE_H__

#include "cocos2d.h"

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

    virtual bool init();
    CREATE_FUNC(SplashScene);

private:
    void GoToMainMenuScene(float dt);

};

#endif // __HELLOWORLD_SCENE_H__
```

SplashScene.cpp

```cpp
#include "SplashScene.h"
#include "Definitions.h"
#include "MainMenuScene.h"

USING_NS_CC;

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

    auto layer = SplashScene::create();

    scene->addChild(layer);

    return scene;
}


// on "init" you need to initialize your instance
bool SplashScene::init()
{
    ////////////////////////////
    // 1. super init first
    if (!Layer::init())
    {

        return false;
    }

    auto visibleSize = Director::getInstance()->getVisibleSize();
    Vec2 origin = Director::getInstance()->getVisibleOrigin();

    this->scheduleOnce(CC_SCHEDULE_SELECTOR(SplashScene::GoToMainMenuScene), DISPLAY_TIME_SPLASH_SCENE);

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

    return true;
}
```

```cpp
void SplashScene::GoToMainMenuScene(float dt) {
    auto scene = MainMenuScene::createScene();
    Director::getInstance()->replaceScene(TransitionFade::create(TRANSITION_TIME, scene));
}
```