Université Abdelmalek Essaadi

Faculté des sciences et techniques de Tanger

Département Génie informatique

LST GI S5

C++

RAPPORT DE PROJET JEU COCOS 2D :

PROGRAMMATION ORIENTER OBJET C++

Encadre Par :

- **EL ACHAK LOTFI**

Réaliser Par :

- **ASSOUFI OUASSIM**

Groupe :

- **1**

1. LES FONCTIONS ET LES NOTES LES PLUS IMPORTANTS DANS MON PROJET

- **Afin de développer une jeu vidéo de 2d qui contient 3 niveaux avec cocos2d :**

✓ **créer sprites**
✓ **créer les physiques de Sprite**
✓ **Créer les mouvements des objets**
✓ **Fonction update() qui va suivre le jeu lors d'action**
✓ **Détection des collisions**
✓ **Remplacer la scène par gagner ou perdre après collision**
✓ **Ajouter les buttons avec ces fonction**
✓ **Keyboard listener**
✓ **Floor scrolling**

CREER LES SPRITES

Nous créons un Sprite en utilisant la classe prédéfinie "Sprite ::Create", puis on fixe sa position et on l'ajoute à la scène courante en en utilisant "this->addChild" .

```
 player = Sprite::create("gloem.png");
player->setPosition((Vec2(visibleSize.width *0.1, visibleSize.height * 0.5)));
player->setScale(0.3);
```

```
this->addChild(player, 1);
```

La deuxième argument de fonction addChild est pour définis l'axe Z.

CREER LES PHYSIQUES DE SPRITE

Pour créer un physique, on commence par récupérer la taille du sprite , on utilisez une autre classe prête pour cocos2d "PhysicsBody". (Nous devons ajouter Physics world dans nos scène afin d'utiliser cette bibliothèque).

```
 player = Sprite::create("gloem.png");
player->setPosition((Vec2(visibleSize.width *0.1, visibleSize.height * 0.5)));
player->setScale(0.3);
auto physicsBody1 = PhysicsBody::createBox(player->getContentSize() /1.5, PhysicsMaterial(1.0f, 1.0f, 1.0f));
physicsBody1->setGravityEnable(true);
physicsBody1->setDynamic(true);

player->setPhysicsBody(physicsBody1);
```

Ajouter le physique en Sprite player , les arguments de fonction PhysicsMaterial concerne la partie de gravitation et la densité de Sprite , puis on set Dynamics concerne activation de gravite.

CREER LES MOUVEMENT DES OBJETS

On applique sur une sprite ce code:

```
auto position4 = obstacle1->getPosition();
        position4.x -= 250 * dt;
        if (position4.x < 0 - (obstacle1->getBoundingBox().size.width / 2))
            position4.x = this->getBoundingBox().getMaxX() + obstacle1->getBoundingBox().size.width / 2;
        obstacle1->setPosition(position4);
```

Pour faire bouger nos Sprite horizontalement sur nos background, nous obtenons d'abord la position actuelle, nous la faisons déplacer en utilisant la deuxième ligne du code ci-dessus, On rend la nouvelle position à chaque fois , On met ce code en fonction update() que on va le discutera en dessous il fonctionne donc en permanence.

J ai utiliser la même méthode en Sprite de gagner qui s'appelle KNZ juste je l'ai mis un peu loin et elle n'a va pas plus que une seule fois.

```
knz = Sprite::create("golf.png");
knz->setPosition((Vec2(visibleSize.width * 5, visibleSize.height * 0.5)));
knz->setScale(0.300);
auto physicsBody11 = PhysicsBody::createBox(knz->getContentSize() / 1.5, PhysicsMaterial(1.0f, 1.0f, 1.0f));
physicsBody11->setGravityEnable(true);
physicsBody11->setDynamic(true);
knz->setPhysicsBody(physicsBody11);

this->addChild(knz, 1);
```

Ajouter le code qui va le mettre a la fin de niveau :

```
auto position3 = knz->getPosition();//move the win signe
position3.x -= 250 * dt;
if (position3.x < 0 - (knz->getBoundingBox().size.width / 2))
    position3.x = this->getBoundingBox().getMaxX() + knz->getBoundingBox().size.width / 2;
knz->setPosition(position3);
```

FONCTION UPDATE() QUI VA SUIVRE LE JOUEUR LORS A'ACTION

Eh bien, je pense qu'il est important de mentionner que la fonction update() est l'endroit où nous ajoutons toutes les actions qui se produisent dans le jeu. La fonction de mise à jour est une chose de base dans tous les Framework et moteurs de jeu , tous les mouvement que j'ai utiliser en haut comme les mouvement des objet je les ai mis dans cette fonction, par

exemple si je veux que quelque chose fasse quelque chose pendant le jeu je le mettrai dans cette fonction.

```
if (floor->getPosition().x < -1950.0f) {
    auto scene = GameOver::createScene();
    Director::getInstance()->replaceScene(scene); // go to losing scene after pass it
}
```

Comme l'exemple en haut il traite le cas de si le joueur passera le signe de gagner il va au Game over scène

```
void GameScreen::update(float dt) {

        player->setRotation(0.0f);
```

Ou bien pour ne faire par des rotations dans les 2 directions en train de jouer

DETECTION DES COLLISIONS

Ce code remplacera la scène actuelle par Perdre la scène lorsqu'il y a une collision. Utilisation simple de la classe Cocos2d et des fonctions intégrées. Comme nous le voyons, les deux Sprites (joueur et signe) sont les corps nous attendons leurs collisions. Nous mettons ce code dans update().

```
Rect joueur = player->getBoundingBox();
Rect signe = knz->getBoundingBox();

if (joueur.intersectsRect(signe))
{
    auto scene = Win1::createScene();
    Director::getInstance()->replaceScene(scene);
}
```

REMPLACER LA SCENE PAR GAGNER OU PERDRE APRES COLLISION

Une utilise la collision de deux Sprite pour déplace de scène à l'autre Comme on a voir en haut si il y a une collision on déplace au niveau scène qui a une class Win1, c'est juste décrire le code replace(scène) dans update() ou scène représente la scène que tu veux aller à.

AJOUTER LES BUTTONS AVEC CES FONCTION

Premièrement il faut déclarer les bibliothèques que on va utiliser namespace ui ;

#include "ui/CocosGUI.h".

```
ui::Button* btnresume =  ui::Button::create("resume.png");
btnresume->setPosition(Vec2(visibleSize.width * 0.5, visibleSize.height * 0.8));
btnresume->addTouchEventListener(CC_CALLBACK_1(PauseMenu::Resume, this));
btnresume->setScale(0.6);
this->addChild(btnresume, 1);
```

Juste de ajouter ce code la dans la fonction init() est largement suffisant pour faire apparaitre une Button puis ajouter a la troisième ligne dans la fonction CC_CALLBACK_1 la fonction qui va faire fonctionner notre bouton exactement comme nous le voulons.

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

KEYBOARD LISTENER

```cpp
auto eventListener = EventListenerKeyboard::create();

eventListener->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(0.1, Vec2(-10, 0)));
        break;
    case EventKeyboard::KeyCode::KEY_RIGHT_ARROW:
    case EventKeyboard::KeyCode::KEY_D:
        event->getCurrentTarget()->runAction(MoveBy::create(0.1, Vec2(10, 0)));
        break;
    case EventKeyboard::KeyCode::KEY_UP_ARROW:
    case EventKeyboard::KeyCode::KEY_W:
        event->getCurrentTarget()->runAction(JumpBy::create(0.5, Vec2(30, 50), 50, 1));
        break;

    }
};

this->_eventDispatcher->addEventListenerWithSceneGraphPriority(eventListener, player);
```

On créer un EventListener dans init(), dans ce cas un EventListenerKeyboard, implémente le gestionnaire d'événements onKeyPressed. Le premier paramètre transmis est l'énumération EventKeyboard::KeyCode, qui est une valeur représentant la touche qui a été enfoncée. La deuxième valeur était la cible de l'événement, dans ce cas notre Sprite. Nous utilisons le pointeur d'événement pour obtenir le nœud cible et mettre à jour sa position dans une direction en fonction de la touche enfoncée. Enfin, nous connectons le _eventDispatcher de notre scène pour recevoir des événements.

FLOOR SCROLLING

Pour faire le « floor » défilant on doit ajouter la même code qu' on a ajouté dans la mouvement des objet sur le Sprite de « floor » :

```cpp
    auto position = floor->getPosition();
    position.x -= 250 * dt;
    if (position.x < 0 - (floor->getBoundingBox().size.width / 2))
        position.x = this->getBoundingBox().getMaxX() + floor->getBoundingBox().size.width / 2;
    floor->setPosition(position);
```

Mais ce fois je vais expliquer la méthodologie de code :

Premièrement j'ai fait une longue image à l'aide de canva.com qui contient une largeur de 7000pixels et j'ai le met au-dessous de l'écran pour faire apparaitre comme une « floor » et elle va défiler jusqu'à a la Sprite de gagne ou il va faire une collision avec

- **Lorsque le premier Sprite sort a 100% de l'écran , repositionnez-le sur le point départ**

AJOUTER LA MUSIQUE

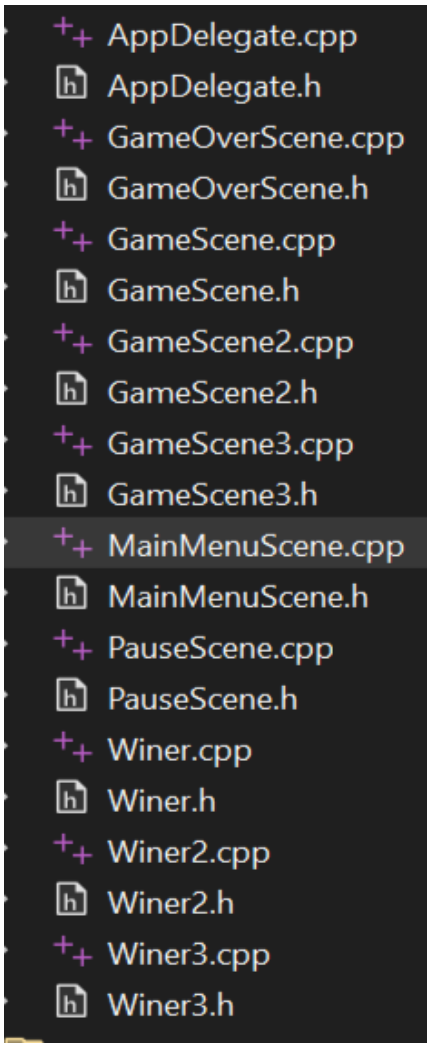
Avant de penser da jouter la musique il faut déclarer une bibliothèque en haut #include ''AudioEngine.h'' après on ajoute le code de preload et play2d avant le retour de fonction init().

```
cocos2d::AudioEngine::preload("jinglebell.mp3");
cocos2d::AudioEngine::play2d("jinglebell.mp3");
```

2. LES NIVEAUX QUI J'AI DEVELOPPER

Dans ce projet j'ai développer 3 niveaux le nom de niveau 1 est GameScene , pour le niveau 2 GameScene2 et pour la troisième niveau GameScene3.

Mes fichier C++ et header qui j’ai fait :



Dans les fichier des trois niveaux j’ai ajouté tous les fonctions et les astuces qu’ on a discuté en haut comme :

1. **créer sprites**
2. **créer les physiques de Sprite**
3. **Créer les mouvements des objets**
4. **Fonction update() qui va suivre le jeu lors d’action**

5. Détection des collisions
6. Remplacer la scène par gagner ou perdre après collision
7. Ajouter les buttons avec ces fonction
8. Keyboard listener
9. Floor scrolling

- **Ces fonctions sera répété dans presque 90% des 3 niveaux.**

- **Les problèmes que j'ai combattre c'est pour faire atterrir parfaitement le joueur sur le « floor » , j'ai résoudre ce problème par manipulation de vecteur de Gravity dans Physicsmaterials.**
- **Autre problème de la fonction update() fonction avant le fonction init ce qu'il fait les spirites devient nulle j'ai résoudre ca de déclarer les spirites dans init() sans auto.**
- **les deux cases rouges des Sprites qui représentent la physique étaient trop larges la collision se produisait avant que les deux Sprites ne se touchent**

NIVEAU 1

Dans ce niveau j'ai met une joueur sprite parcouru sur un floor et soter sur un obstacle jusqua le drapeau de golf ce sprite sappelle KNZ le floor est de largeur de 7000pixels.

Init() :

```
bool GameScreen::init()
{

    if (!Layer::init())
    {
        return false;

    }

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

    auto pauseItem = MenuItemImage::create("pausebtn.png",
        "pausebtn.png",
        CC_CALLBACK_1(GameScreen::GoToPauseScene, this));

    pauseItem->setPosition(Point(pauseItem->getContentSize().width - (pauseItem->getContentSize().width / 2) + origin.x,
        visibleSize.height - pauseItem->getContentSize().height  + (pauseItem->getContentSize().width / 2) + origin.y));

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


    auto backgroundSprite = Sprite::create("BG_02.png"); /*background for the main menu*/
    backgroundSprite->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
    this->addChild(backgroundSprite, -1);
    float rx = visibleSize.width / backgroundSprite->getContentSize().width;
    float ry = visibleSize.height / backgroundSprite->getContentSize().height;

    backgroundSprite->setScaleX(rx);
    backgroundSprite->setScaleY(ry);


     player = Sprite::create("gloem.png");
    player->setPosition((Vec2(visibleSize.width *0.1, visibleSize.height * 0.5)));
    player->setScale(0.3);
    auto physicsBody1 = PhysicsBody::createBox(player->getContentSize() /1.5, PhysicsMaterial(1.0f, 1.0f, 1.0f));
    physicsBody1->setGravityEnable(true);
    physicsBody1->setDynamic(true);

    player->setPhysicsBody(physicsBody1);

    this->addChild(player, 1);



    floor = Sprite::create("floor.png");
```

Update() :

```
void GameScreen::update(float dt) {
    if (player != NULL) {
        player->setRotation(0.0f);
    }
        Size visibleSize = Director::getInstance()->
            getVisibleSize();
        Point origin = Director::getInstance()->getVisibleOrigin();



        Rect joueur = player->getBoundingBox();
        Rect signe = knz->getBoundingBox();

        if (joueur.intersectsRect(signe))
        {
            auto scene = Win1::createScene();
            Director::getInstance()->replaceScene(scene);
        }

        auto position = floor->getPosition();
        position.x -= 250 * dt;
        if (position.x < 0 - (floor->getBoundingBox().size.width / 2))
            position.x = this->getBoundingBox().getMaxX() + floor->getBoundingBox().size.width / 2;
        floor->setPosition(position);


        auto position3 = knz->getPosition();//move the win signe
        position3.x -= 250 * dt;
        if (position3.x < 0 - (knz->getBoundingBox().size.width / 2))
            position3.x = this->getBoundingBox().getMaxX() + knz->getBoundingBox().size.width / 2;
        knz->setPosition(position3);


auto position4 = obstacle1->getPosition();
        position4.x -= 250 * dt;
        if (position4.x < 0 - (obstacle1->getBoundingBox().size.width / 2))
            position4.x = this->getBoundingBox().getMaxX() + obstacle1->getBoundingBox().size.width / 2;
        obstacle1->setPosition(position4);


        if (floor->getPosition().x < -1950.0f) {
            auto scene = GameOver::createScene();
            Director::getInstance()->replaceScene(scene);
        }
```

Niveau 2 :

Comme niveau 1 juste une Sprite additionnel de fantômes si le joueur toucher le fantômes le fonction de détection des collisions faire en action et la scène sera déplacer au GameOverscene.

Init() :

```
bool GameScreen2::init()
{

    if (!Layer::init())
    {
        return false;

    }


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

    auto pauseItem = MenuItemImage::create("pausebtn.png",
        "pausebtn.png",
        CC_CALLBACK_1(GameScreen2::GoToPauseScene, this));

    pauseItem->setPosition(Point(pauseItem->getContentSize().width - (pauseItem->getContentSize().width / 2) + origin.x,
        visibleSize.height - pauseItem->getContentSize().height + (pauseItem->getContentSize().width / 2) + origin.y));

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


    auto backgroundSprite = Sprite::create("BG_02.png"); /*background for the main menu*/
    backgroundSprite->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
    this->addChild(backgroundSprite, -1);
    float rx = visibleSize.width / backgroundSprite->getContentSize().width;
    float ry = visibleSize.height / backgroundSprite->getContentSize().height;

    backgroundSprite->setScaleX(rx);
    backgroundSprite->setScaleY(ry);


     player = Sprite::create("gloem.png");
    player->setPosition((Vec2(visibleSize.width * 0.1, visibleSize.height * 0.5)));
    player->setScale(0.3); //scale dyal player
    player->setName("player");
    auto physicsBody1 = PhysicsBody::createBox(player->getContentSize() / 1.5, PhysicsMaterial(1.0f, 1.0f, 1.0f));
    physicsBody1->setGravityEnable(true);
    physicsBody1->setDynamic(true);
    physicsBody1->setContactTestBitmask(1);
    physicsBody1->setCollisionBitmask(1);
    physicsBody1->setCategoryBitmask(1);
    player->setRotation(0.0f);
    player->setPhysicsBody(physicsBody1);

    this->addChild(player, 1);


    //  creating physique for player

    floor = Sprite::create("floor.png");

    floor->setPosition(Vec2(visibleSize.width / 2, visibleSize.height * 0.1));
    this->addChild(floor, 1);
    floor->setScaleX(3);

    //creating physique for THE FLOOR

    auto physicsBody_floor = PhysicsBody::createBox(floor->getContentSize(), PhysicsMaterial(1.0f, 0.0f, 1.0f));

    physicsBody_floor->setDynamic(false);
```

Update() :

```cpp
void GameScreen2::update(float dt) {


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

    if (player != NULL) {
        player->setRotation(0.0f);
    }

    Rect joueur = player->getBoundingBox();
    Rect signe = ghost->getBoundingBox();

    if (joueur.intersectsRect(signe))
    {
        auto scene = GameOver::createScene();
        Director::getInstance()->replaceScene(scene);
    }

    Rect joueur1 = player->getBoundingBox();
    Rect signe1 = knz->getBoundingBox();

    if (joueur1.intersectsRect(signe1))
    {
        auto scene = Win2::createScene();
        Director::getInstance()->replaceScene(scene);
    }

    auto position = floor->getPosition();
    position.x -= 250 * dt;
    if (position.x < 0 - (floor->getBoundingBox().size.width / 2))
        position.x = this->getBoundingBox().getMaxX() + floor->getBoundingBox().size.width / 2;
    floor->setPosition(position);



    auto position3 = knz->getPosition();
    position3.x -= 250 * dt;
    if (position3.x < 0 - (knz->getBoundingBox().size.width / 2))
        position3.x = this->getBoundingBox().getMaxX() + knz->getBoundingBox().size.width / 2;
    knz->setPosition(position3);


    if (obstacle1 != NULL) {
        auto position4 = obstacle1->getPosition();
        position4.x -= 250 * dt;
        if (position4.x < 0 - (obstacle1->getBoundingBox().size.width / 2))
            position4.x = this->getBoundingBox().getMaxX() + obstacle1->getBoundingBox().size.width / 2;
        obstacle1->setPosition(position4);
    }

    if (floor->getPosition().x < -2000.0f) {
        auto scene = GameOver::createScene();
        Director::getInstance()->replaceScene(scene); // mbghatx tkhdm !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!1
    }

    auto position6 = ghost->getPosition();
    position6.x -= 250 * dt;
    if (position6.x < 0 - (ghost->getBoundingBox().size.width / 2))
        position6.x = this->getBoundingBox().getMaxX() + ghost->getBoundingBox().size.width / 2;
    ghost->setPosition(position6);
```

Niveau 3 :

La troisième niveau concerne a faire de transformer le joueur a autre joueur qui vole et il doit éviter des obstacle qui sont des fantômes, et une fois une collision faire entre eux la scène remplacer par la scène de GameOverScene .

Init() :

```cpp
bool GameScreen3::init()
{

    if (!Layer::init())
    {
        return false;

    }


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

    auto pauseItem = MenuItemImage::create("pausebtn.png",
        "pausebtn.png",
        CC_CALLBACK_1(GameScreen3::GoToPauseScene, this));

    pauseItem->setPosition(Point(pauseItem->getContentSize().width - (pauseItem->getContentSize().width / 2) + origin.x,
        visibleSize.height - pauseItem->getContentSize().height + (pauseItem->getContentSize().width / 2) + origin.y));

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


    auto backgroundSprite = Sprite::create("BG_02.png"); /*background for the main menu*/
    backgroundSprite->setPosition(Point((visibleSize.width / 2) + origin.x, (visibleSize.height / 2) + origin.y));
    this->addChild(backgroundSprite, -1);
    float rx = visibleSize.width / backgroundSprite->getContentSize().width;
    float ry = visibleSize.height / backgroundSprite->getContentSize().height;

    backgroundSprite->setScaleX(rx);
    backgroundSprite->setScaleY(ry);


     player = Sprite::create("pilot.png");
    player->setPosition((Vec2(visibleSize.width * 0.1, visibleSize.height * 0.5)));
    player->setScale(0.5); //scale dyal player
    player->setName("player");
    auto physicsBody1 = PhysicsBody::createBox(player->getContentSize() / 1.5, PhysicsMaterial(1.0f, 1.0f, 1.0f));
    physicsBody1->setGravityEnable(true);
    physicsBody1->setDynamic(true);
    player->setPhysicsBody(physicsBody1);

    this->addChild(player, 1);
```

Update() :

```cpp
void GameScreen3::update(float dt) {


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

    if (player != NULL) {
        player->setRotation(0.0f);
    }


    Rect joueur = player->getBoundingBox();
    Rect fontome = ghost->getBoundingBox();

    if (joueur.intersectsRect(fontome))
    {
        auto scene = GameOver::createScene();
        Director::getInstance()->replaceScene(scene);
    }


    Rect joueur1 = player->getBoundingBox();
    Rect floor1 = floor->getBoundingBox();

    if (joueur1.intersectsRect(floor1))
    {
        auto scene = GameOver::createScene();
        Director::getInstance()->replaceScene(scene);
    }

    Rect joueur2 = player->getBoundingBox();
    Rect fontome1 = ghost1->getBoundingBox();

    if (joueur2.intersectsRect(fontome1))
    {
        auto scene = GameOver::createScene();
        Director::getInstance()->replaceScene(scene);
    }

    Rect joueur3 = player->getBoundingBox();
    Rect signe = knz->getBoundingBox();

    if (joueur3.intersectsRect(signe))
```

NOTE :

Le code que j ai utiliser pour remplacer la scène par GameOverScene au cas de perdre ou échouée .

```
if (floor->getPosition().x < -2300.0f) {
    auto scene = GameOver::createScene();
    Director::getInstance()->replaceScene(scene);
}
```

Les outils utiliser :

- ✓ Photoshop.
- ✓ Visual Studio Community
- ✓ https://www.freepik.com
- ✓ https://www.canva.com
- ✓ https://craftpix.net

LA FIN