

Plunge **interactive**

Cocos2d-x!

Videojocs multi-plataforma amb C++



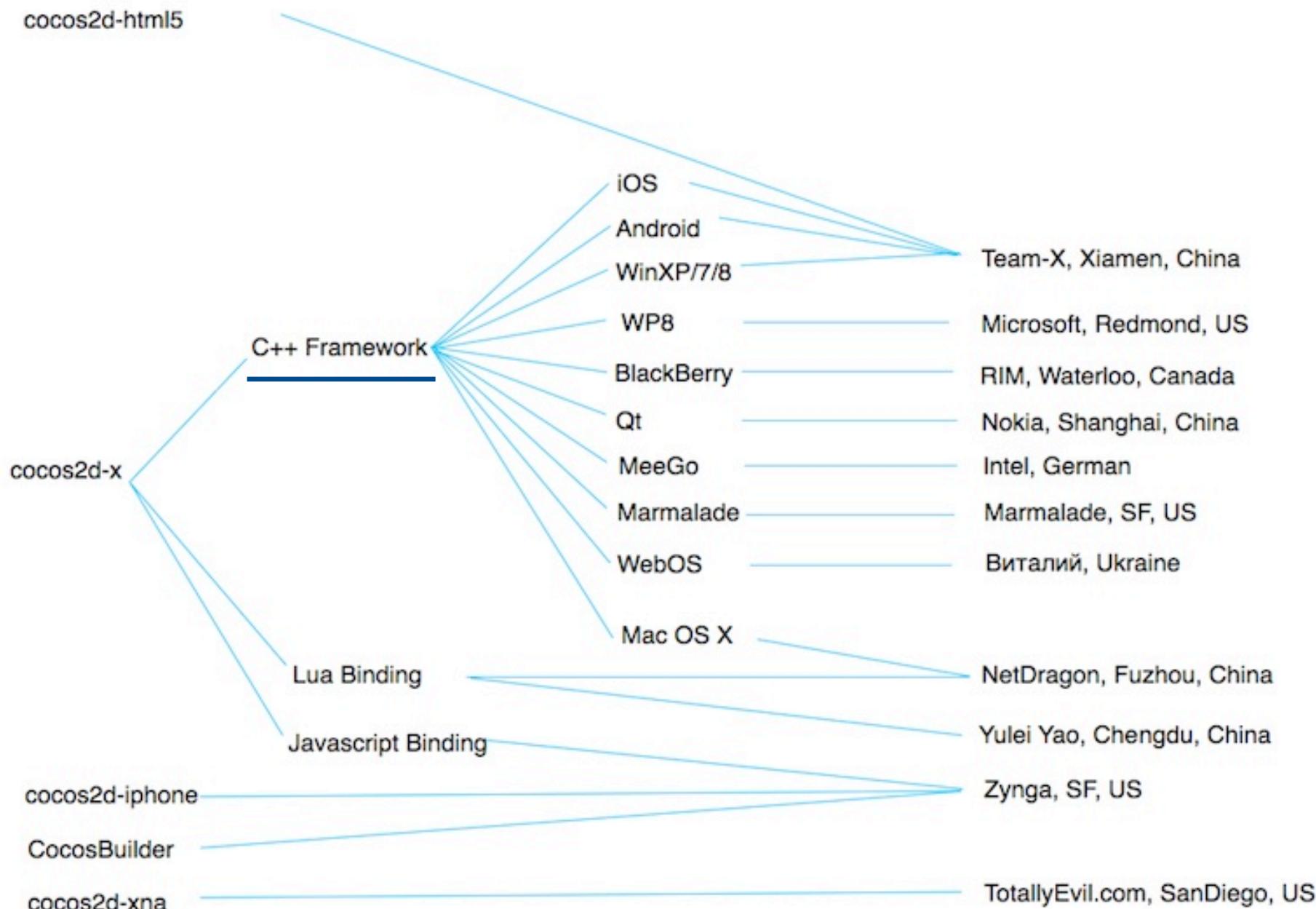
Plunge^{interactive}

Hola Cocos!



Posem-nos en situació

- Orígens en Cocos2d
- Open source (MIT)
- Utilitzat per Zynga, Glu, Gree, Knoami...
- Diferents llenguatges de programació
- Suportat per múltiples plataformes
- Enorme quantiat de tools disponibles



Tools



Plunge interactive

Tools



Xcode File Edit View Navigate Editor Product Window Help

MummyMadness.xcodeproj — h GameplayManager.h
Running MummyMadness on Atreides's iPhone

Project 6 0 4

Editor View Organizer

MummyMadness 1 target, iOS SDK 6.0

MummyMadness Resources ios libs Classes Utils Gameplay Managers

GameplayManager.cpp

GameplayManager.h

PhysicsManager.cpp

PhysicsManager.h

CullingManager.cpp

CullingManager.h

HookManager.h

ParticleSystemManager.cpp

ParticleSystemManager.h

MMObstaclesManager.cpp

MMObstaclesManager.h

MMObstaclesManager.h

CollectablesManager.cpp

CollectablesManager.h

ContactManager.h

PhysicsManager.h

Mummy* mummy;

LHLayer* mainLHLayer;

int score;

std::vector<LHSprite*> deleteableSprites;

// jump

int numOffFootContacts;

bool jumping;

float timeJumping;

// hook

bool hookFinished;

bool hookEnabled;

b2Vec2 hookPoint;

bool godMode;

bool shouldExit;

private: // types

enum GameplayState

{

Idle,

Throwing_Barrels,

};

private: // fields

cocos2d::CCSize screen;

GameplayScene* gameplayScene;

RaycastProjector raycastProjector;

LHSprite cameraFocusParallax;

HookManager* hookManager;

EnemiesManager* enemiesManager;

CollectablesManager* collectablesManager;

MMObstaclesManager* obstaclesManager;

ParticleSystemManager* particleSystemManager;

ParallaxManager* parallaxManager;

CullingManager* cullingManager;

ContactManager contactManager;

PhysicsManager* physicsManager;

Mummy* mummy;

LHLayer* mainLHLayer;

int score;

std::vector<LHSprite*> deleteableSprites;

// jump

int numOffFootContacts;

bool jumping;

float timeJumping;

// hook

bool hookFinished;

bool hookEnabled;

b2Vec2 hookPoint;

bool godMode;

bool shouldExit;

private: // methods

void initAudio();

void initParallax();

void addSpritesToDeleteableList();

/// Called before two fixtures begin to touch

virtual void PreSolve(b2Contact* contact, const b2Manifold* oldManifold);

/// Called when two fixtures begin to touch.

virtual void BeginContact(b2Contact* contact);

/// Called when two fixtures cease to touch.

virtual void EndContact(b2Contact* contact);

void onHPATHEnd(CCObject* obj);

void GameplayManager::ccTouchMoved(CTouch* touch, CCEvent* event) {}

void GameplayManager::ccTouchEnded(CTouch* touch, CCEvent* event)

{

if (jumping)

{

mummy->endJump();

}

}

void GameplayManager::update(float dt)

{

if (shouldExit)

{

CCDirector::sharedDirector()->replaceScene(MenuScene::create());

return;

}

if (getLevelHelper()->isPaused()) return;

cullingManager->update();

updateJumpAndHook(dt);

physicsManager->update(dt);

hookManager->update();

collectablesManager->update();

cameraFocusParallax->transformPosition(getLevelHelper()->metersToPoints(ccpMult(ccpFromSize(screen), 0.5f)));

mummy->postStepUpdate();

obstaclesManager->postStepUpdate();

enemiesManager->postStepUpdate();

outOfScreenCleanup();

}

void GameplayManager::draw()

{

}

void GameplayManager::projectToCameraShape(b2Vec2 position, float xOffset)

{

raycastProjector.init(PM_FIRST, physicsManager->getWorld());

b2Vec2 pointA = mummy->getb2Position() + b2Vec2(xOffset, -10);

b2Vec2 pointB = mummy->getb2Position() + b2Vec2(xOffset, 10);

raycastProjector.projectWithTag(MM_CAMERA_SHAPE, pointA, pointB);

if (raycastProjector.hasCollided())

position->operator=(raycastProjector.getProjectedPosition());

}

GameplayLayer* GameplayManager::getGameplayLayer() const

{

return gameplayScene->getGameplayLayer();

}

LevelHelperLoader* GameplayManager::getLevelHelper() const

{

All Output

Local

MummyMadness

Tools



Xcode File Edit View Navigate Editor Product Window Help

MummyMadness.xcodeproj — GamePlayManager.h

Running MummyMadness on Atreides's iPhone

Project 6 0 4

Editor View Organizer

MummyMadness 1 target, iOS SDK 6.0

MummyMadness Resources ios libs Classes Utils Gameplay Managers GamePlayManager.cpp GamePlayManager.h PhysicsManager.cpp PhysicsManager.h CullingManager.cpp CullingManager.h HookManager.h HookManager.h ParticleSystemManager.cpp ParticleSystemManager.h MMOObstaclesManager.cpp MMOObstaclesManager.h MMOObstaclesManager.h CollectablesManager.h CollectablesManager.h ContactManager.h ContactManager.h EnemiesManager.cpp EnemiesManager.h HookManager.h HookManager.h MMOObstacle.h RollingObstacle.cpp RollingObstacle.h

```
33 class GamePlayManager : public CCObject,
34     public b2ContactListener
35 {
36     private: // types
37         enum GamePlayState
38     {
39         Idle,
40         Throwing_Barrels,
41     };
42
43     private: // fields
44     cocos2d::CCSize screen;
45     GamePlayScene* gamePlayScene;
46     RaycastProjector raycastProjector;
47     LHSprite cameraFocusParallax;
48
49     HookManager* hookManager;
50     EnemiesManager* enemiesManager;
51     CollectablesManager* collectablesManager;
52     MMOObstaclesManager* obstacleManager;
53
54     ParticleSystemManager* particleSystemManager;
55     ParallaxManager* parallaxManager;
56     CullingManager* cullingManager;
57
58     ContactManager contactManager;
59     PhysicsManager* physicsManager;
60     Mummy* mummy;
61
62     LHLayer* mainLHLayer;
63
64     int score;
65
66     std::vector<LHSprite*> deleteableSprites;
67
68     // jump
69     int numOfFootContacts;
70     bool jumping;
71     float timeJumping;
72 }
```

418 void GamePlayManager::ccTouchMoved(CCEvent* touch, CCEvent* event) {}
419 void GamePlayManager::ccTouchEnded(CCEvent* touch, CCEvent* event)
420 {
421 if (jumping)
422 {
423 mummy->endJump();
424 }
425 }
426
427 void GamePlayManager::update(float dt)
428 {
429 if (shouldExit)
430 {
431 CCDirector::sharedDirector()->replaceScene(MenuScene::create());
432 return;
433 }
434 }

Arduino Duemilanove w/ ATmega328 COM12

Blink.ino* (Global Scope) loop()

```
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
    // initialize the digital pin as an output.
    pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
    digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
    Serial.println("LED on");
    delay(1000); // wait for a second
    digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
    Serial.println("LED off");
    delay(1000); // wait for a second
}
```

Output

Now output from: Micro Build
Compiling 'Blink' for 'Arduino Duemilanove w/ ATmega328'
Binary sketch size: 1084 bytes (of a 30720 byte maximum) (0.159 secs)
Uploading to I/O board using 'COM12'
One uploading

Solution Explorer

Search Solution Explorer (C:\Users\Public\Documents\Microsoft Visual Studio\Projects\Blink)

Solution 'Blink' (1 project)

- External Dependencies
- Header Files
- Resource Files
- Source Files
- Blink
 - led
 - loop()
 - setup()

Tools



LevelHelper PRO File Edit View Window Help Tue 6:15 PM Q

Universal Landscape (480x320) | MM_Levels

Scene Tester Xcode Simulator Corona Simulator

Zoom 32 Snap Background Grid Show

File Project Settings Documentation Messages

Level Editor Custom Properties Supporting Code Documentation Messages

Z Files

level

MM_Levels.lproj

mummymadness

Resources

Enemies.pshs

goods.pshs

hd

ipadhd

level1.pshs

ParallaxSprites.pshs

sd

SeamlessRepeatingTextures.pshs

spritesheets.pshs

coin

editorHelpers

mummy

obstacles

scenarioBack

scenarioFront

Lava_animation0001

Lava_animation0002

Lava_animation0003

Lava_animation0004

Lava_animation0005

Lava_animation0006

Lava_animation0007

Lava_animation0008

Lava_animation0009

Lava_animation0010

Lava_animation0011

Lava_animation0012

Lava_animation0013

Lava_animation0014

Lava_animation0015

Lava_animation0016

Lava_animation0017

Lava_animation0018

Lava_animation0019

Lava_animation0020

Lava_animation0021

Lava_animation0022

Lava_animation0023

Lava_animation0024

Lava_animation0025

Lava_animation0026

Lava_animation0027

Hierarchical View

Physical Boundaries

Gravity

Game World Size

Bezier Tool

Clone Tool

General Properties

Unique Name: hookZone

Z Order: 0

Tag: MM_HOOK_ZONE

Position

X: -898.833

Y: 160.621

Angle: 0

Width: 158.335

Height: 176.645

Scale X: 4.948

Scale Y: 5.52

Color:

Opacity: .3

Visible:

Flip X:

Flip Y:

SH Document: spritesheets.pshs

SH Sheet: editorHelpers

SH Sprite: areaBox

Custom User Properties

User Class: MMHookshotZone

hookUniqueN hook

Generic Animation Properties

Other Animations (Corona SDK)

Physics Properties

Handled by SpriteHelper Edit

Object Type: Static

Gravity Scale: 1

Linear Velocity: 0

Angular Velocity: 0

Linear Damping: 0

Angular Damping: 0

Can Sleep:

The screenshot shows a game level titled 'mummymadness' in the Level Editor. The scene features a stone floor with lava patches and a stone wall. A mummy enemy is standing on the floor, and a treasure chest is on the wall. Two floating coins are in the air. A large hookshot zone, represented by a yellow rectangle, is positioned above the mummy. The right side of the screen displays the 'General Properties' panel for the 'hookZone' object, showing its unique name, position, and physics properties like gravity scale and linear velocity.

Tools

CocosBuilder

Tools Zoom shop trophy menu_mummy menu

Project

- ccbAssets
- intro.ccb
- intro_ceilinglight.ccb
- menu.ccb
- menu_achievementsGlow.ccb
- menu_candleglow.ccb
- menu_mummy.ccb
- menu_playGlow.ccb
- menu_shopGlow.ccb
- shop.ccb
- trophy.ccb
- trophy_achievementslist.ccb
- trophy_exitGlow.ccb

Code Connections

Custom class: Don't assign

CCNode

Position: -23.0, 12.0
Content size: 22.0, 25.0
Anchor point: 0.289, 0.753
Scale: 1.00, 1.00
Rotation: -3.0 Degrees
Tag: -1
 Ignore anchor point
 Visible
CCSprite

Sprite frame: ccbAssets/menu.pli...
Opacity: 255
Color:
 Flip X Flip Y
Blend src: Src Alpha
Blend dst: One - Src Alpha
Normal Additive

Default Timeline

0 1 2 3 4 5 6 7 8

CCLayer

- background
- CCLayer
- shopBG
- trophyBG
- fitLayout

 - gameCenter
 - playForeground
 - ccbCandleglow
 - candleflame
 - vesselFront
 - ccbCeilinglight
 - ccbPlayGlow
 - ccbTrophyGlow
 - ccbShopGlow

- CCMenu

 - CCMenuItemImage
 - CCMenuItemImage
 - CCMenuItemImage

- CCSprite
- CCBMummy

No chained timeline



Tools



Plunge interactive

bmGlyph

Open Save Save as Publish

work on selected glyph(s) only

Anti Alias

Texture

Width x Height
512px 512px

Background :

CheckerBoard :

Auto Size :

Publish Settings

Color : RGBA8888

Directory :

Font Name :

Format : Cocos2...
Auto SD Level : HD (@2...
SD Quality : Default
Redraw when downscaling

Packing

Bounding Box : + 0 px
Padding : 2px
Sort Method : Special
Reversed :
Refresh

The tool interface includes various settings for file operations (Open, Save, Publish), anti-aliasing, texture dimensions (512x512), and publishing options (Color, Directory, Font Name, Format, Auto SD Level, SD Quality). It also features packing parameters (Bounding Box, Padding, Sort Method, Reversed) and a Refresh button.

Tools



Plunge interactive

Particle Creator

PLAY SETTING

One Play (radio button selected) Loop Play

DEVICE & DIRECTION

iPhone iPad

Landscape Portrait

FILE

LOAD SAVE

?

f

Graphic Configure Particle Configure Code Make Testing

COLOR

startColor

A 1.00

endColor

A 1.00

startColorVar

R 0 G B A

endColorVar

R 0

BLENDFUNC

Source GL_SRC_ALPHA

Dest GL_ONE

PARTICLE TEXTURE

To Base

iPhone

The Particle Creator interface allows users to configure particle effects for iPhone and iPad. It includes sections for Play Setting (One Play or Loop Play), Device & Direction (Landscape or Portrait), and File (Load or Save). The main configuration area features tabs for Graphic Configure, Particle Configure, Code Make, and Testing. Under Particle Configure, users can set colors (startColor and endColor with alpha values from 0.00 to 1.00), color variables (startColorVar and endColorVar with sliders for R, G, B, and A), and blend functions (Source GL_SRC_ALPHA and Dest GL_ONE). A preview window shows a bright orange and yellow flame effect against a black background. A texture preview panel at the bottom left shows a grayscale gradient, and a "To Base" button is located at the bottom.

Tools

page16.tps

New Open Save Save defaults Add Sprites Add Folder Delete Publish PVR Viewer

TextureSettings

Output

- Data Format: cocos2d
- Data file: /Spritesheet/page16-ipadhd.plist
- Texture format: zlib compr. PVR (.pvr.ccz, Ver.2)
- Texture file: spritesheet/page16-ipadhd.pvr.ccz
- Premultiply alpha:
- Flip P/R:
- Image format: RGBA8888
- Dithering: NearestNeighbour
- AutoSD: 

Geometry

- Max size: W: 4096 H: 2048
- Fixed size: W: H:
- Size constraints: POT (Power of 2)
- Force squared:
- Force word aligned:
- Pack: Best
- Scale: 0.5
- Scale mode: Smooth

Layout

- Algorithm: MaxRects
- Heuristics: Best
- Border padding: 2
- Shape Padding: 2
- Inner Padding: 0
- Extrude: 0
- Common divisor: x 1 y 1
- Reduce border artifacts:
- Allow rotation:

1.00   



Sprites

- braç-llop.png
- cella-llop1.png
- cella-llop2.png
- fi.png
- fin.png
- fum1.png
- fum2.png
- fum3.png
- fum4.png
- ilop-cara.png
- ilop-cos.png
- ilop-cua.png
- ilop-ulls-oberts.png
- ilop-ulls-tancats.png
- pajarita.png
- the-end.png
- toll-aigua.png
- transparencia-aigua-cua.png
- transparencia-aigua-cul.png

Size: 512x1024 (2048kB)

- Tools = més dades i menys codi
- Menys codi = menys bugs
- Menys codi = més productivitat

* El codi ha d'existir per a les coses importants

Exemples



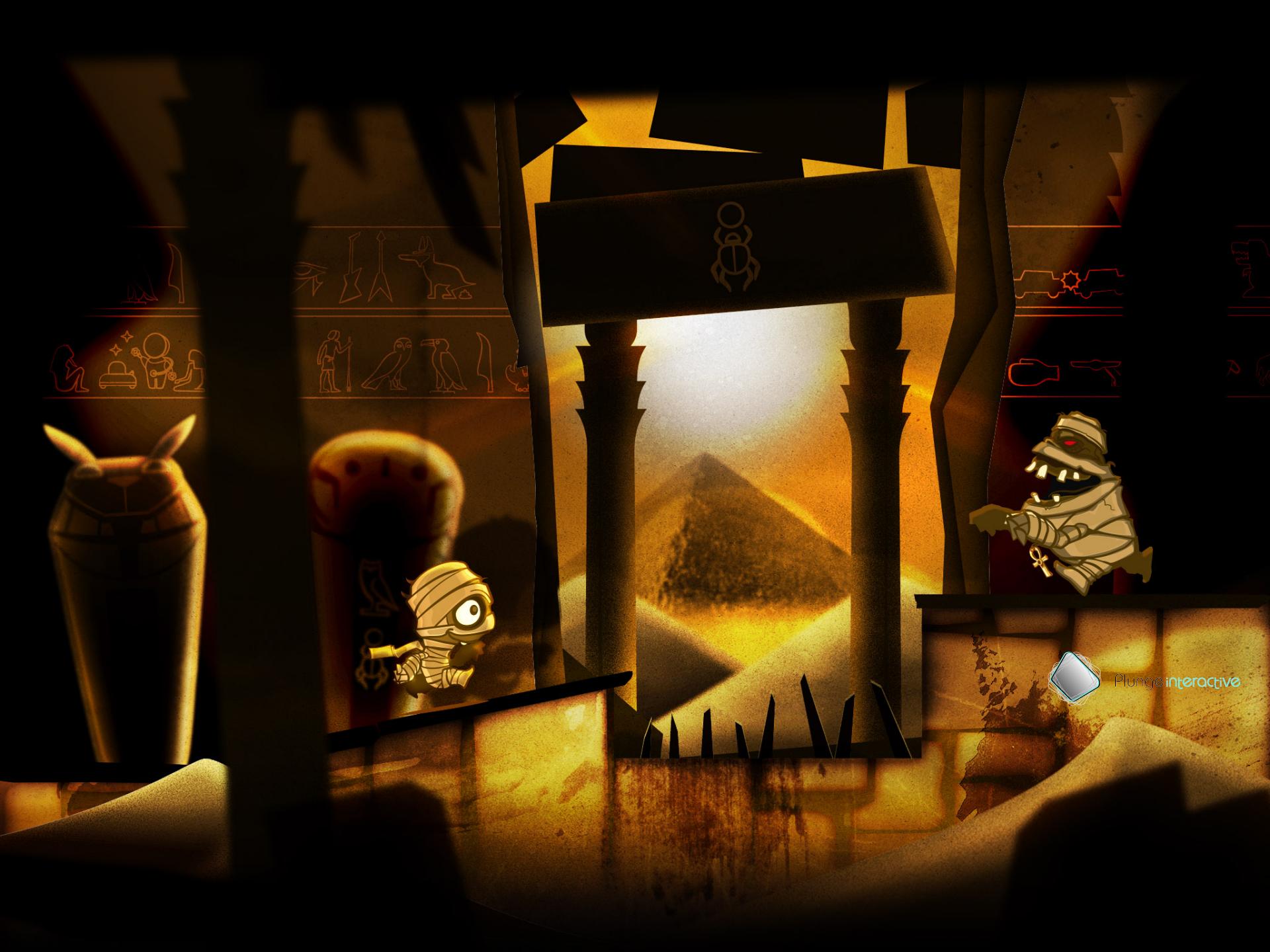
Plunge **interactive**

Exemples: jocs fets a Plunge Interactive



Plunge interactive





Plunge interactive

INTERACTÚA CON EL CUENTO



Altres exemples



Plunge interactive

Featured



WW II



我叫MT Online



渡劫成仙



FengYunTianXia



神仙道



DaZhangMen



WoLongYin



XingChenBian



TouchMix 2



Fishing Joy 2



Warring States



BattleLand:Warrior

Last Updated



Spot the Number



Boomlings



Roger jolly pirate



Memory



Duro Race



Tap Bubble



Maya Matrix



Hashi Puzzles:



김치스토리



YiGuo



Xamalga



打鼹鼠Super



김치스토리



Christmas Eve Rush



Hobbit The Beggar



Hobbit Jetpack



EgyptDash



Touch Music



WuZhiXiYou



The TIme Master



Missile Missio



Fearless



Doggie Destroyer



Credit - ElScootro



Super Darwin



Captain Torrent



Card Puzzle



卡片拼图

Consells previs



Plunge **interactive**

- Entorns més fàcils de configurar:
 - iOS
 - Windows (win32)
- Entorns més complicats
 - Android (especialment sobre windows)
 - Windows 8
 - Windows Phone 8

Introducció a Cocos2d-x



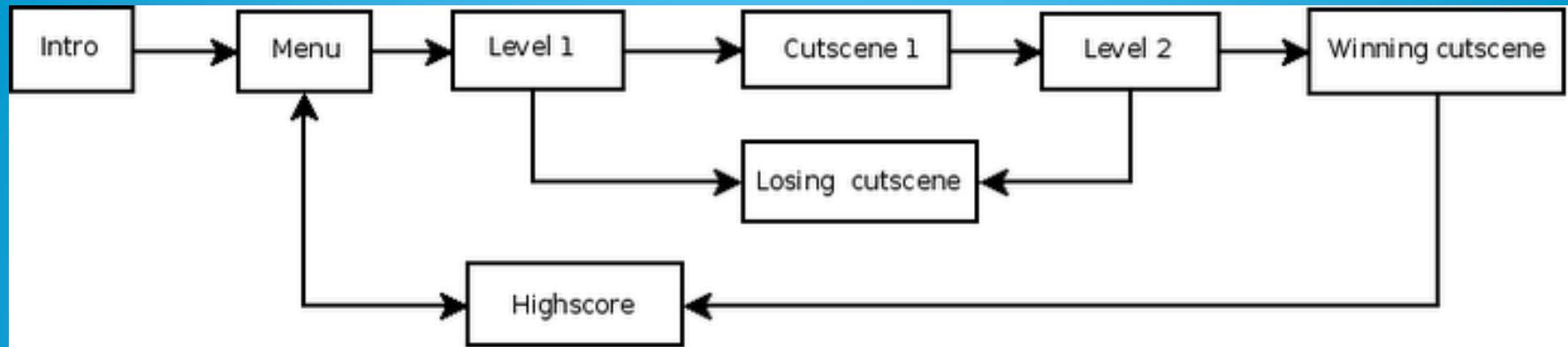
Plungeinteractive

Disclaimer



Algunes nomenclatures i classes
existeixen únicament per facilitar la
portabilitat de codi existent fet amb
Cocos2d-iPhone (Objective-C) a C++

Conceptes bàsics: CCScene



Conceptes bàsics: CCScene

HelloWorldScene.h

```
#ifndef __HELLOWORLDSCENE_H__
#define __HELLOWORLDSCENE_H__

#include "cocos2d.h"

class HelloWorldScene: public cocos2d::CCScene
{
private: // methods
    HelloWorldScene () {}
    ~HelloWorldScene () {}

    bool init();

public: // methods
    CREATE_FUNC(HelloWorldScene);
};

#endif
```

HelloWorldScene.cpp

```
#include "HelloWorldScene.h"

using namespace cocos2d;

bool HelloWorldScene::init()
{
    if (!CCScene::init())
        return false;

    // do something

    return true;
}
```

Important:

- Hereda de CCNode

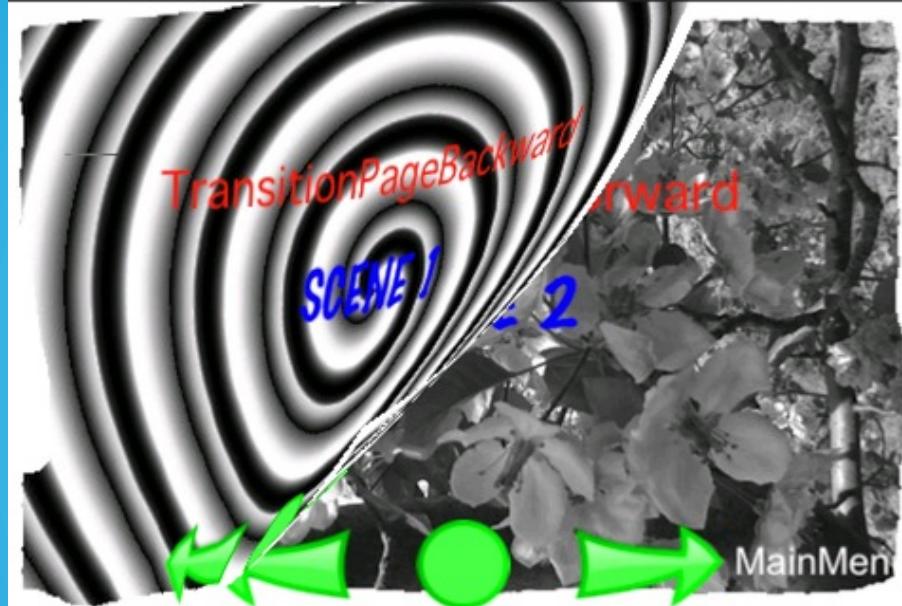
Conceptes bàsics: CCDirector



```
CCDirector::sharedDirector()->replaceScene(HelloWorldScene);
```

Conceptes bàsics: CCTransition

```
CCDirector::sharedDirector()->replaceScene(CCTransitionFade::create(0.5,newScene));
```

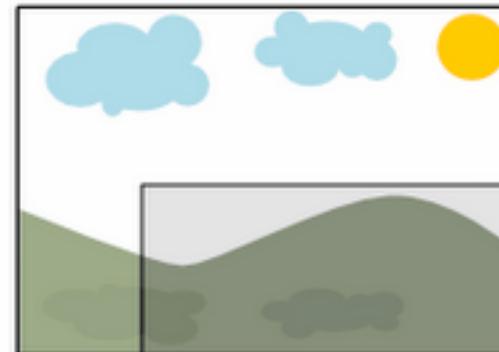
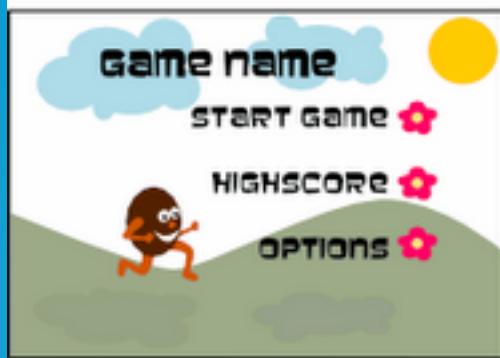


[CCTransitionEaseScene](#)
[CCTransitionSceneOriented](#)
[CCTransitionRotoZoom](#)
[CCTransitionJumpZoom](#)
[CCTransitionMoveInR](#)
[CCTransitionMoveInL](#)
[CCTransitionSplitRows](#)
I moltes mes...



Conceptes bàsics: CCLayer

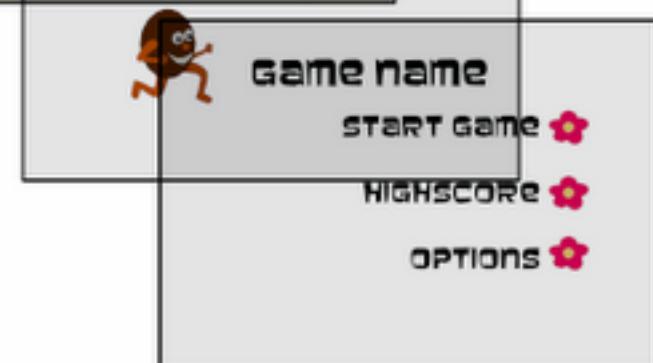
A regular menu scene



Background layer



Animation layer



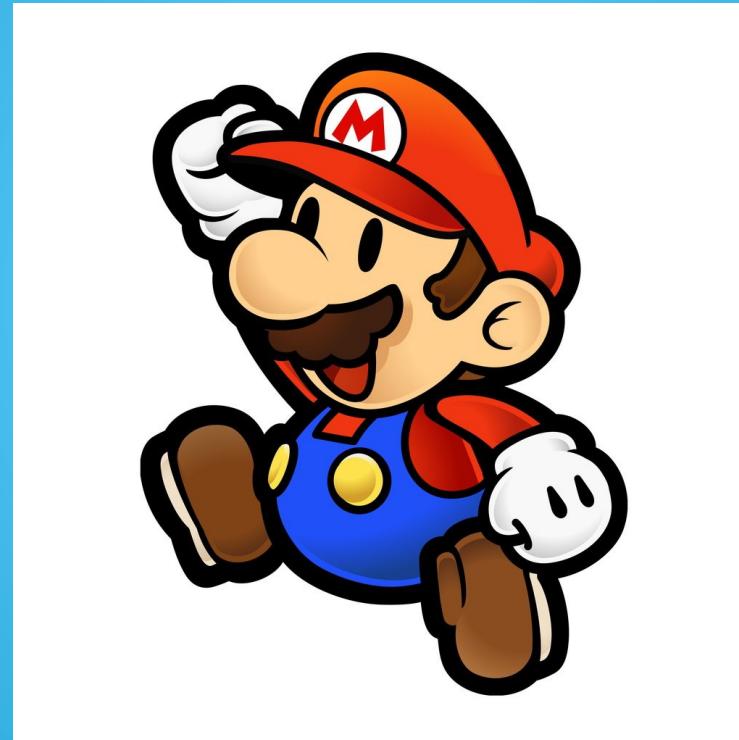
Menu layer

Subclasses interessants:

- CCLayerColor
- CCLayerColor

Conceptes bàsics: CCSprite

```
CCSprite::create("mario.png");
```



Conceptes bàsics: CCSprite

```
mySprite->setAnchorPoint(cpp(0, 0));  
mySprite->setPosition(cpp(10, 10));
```



El anchor Point importa per:

- Posicionament
- Rotació

Conceptes bàsics: CCSprite

Molts sprites poden decrementar el framerate de manera **dràstica**



Conceptes bàsics: CCSprite

Problemes dels sprites:

- Escalats a potència de 2
- Consum de memòria
- El render necessita temps



64x128

La solució són els Sprite sheets

- Escalats a potència de 2
- Consum de memòria
- El render necessita temps

Conceptes bàsics: Sprite Batch

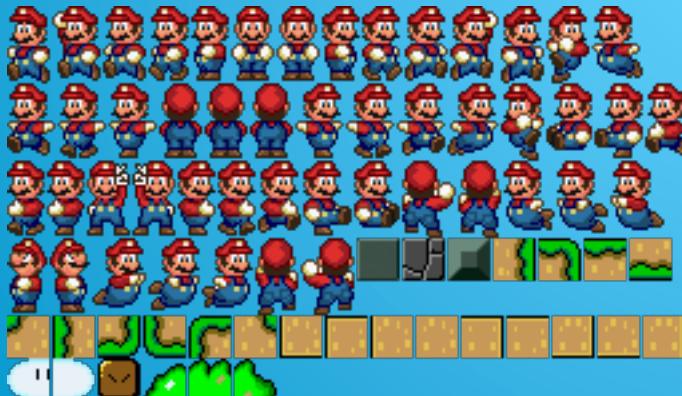


La solució són els sprite sheets combinats amb el sprite batch

- Escalats a potència de 2 (també)
- Millor aprofitament de la memòria
- Render més fàcil (amb sprite batch)

Classes clau:

- **CCSpriteBatchNode** (textura)
- **CCSpriteFrameCache** (informació de la textura)



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>frames</key>
    <dict>
        <key>BACKGROUND-cel-color.png</key>
        <dict>
            <key>frame</key>
            <string>{{995,916},{25,25}}</string>
            <key>offset</key>
            <string>{0,0}</string>
            <key>rotated</key>
            <false/>
            <key>sourceColorRect</key>
            <string>{{0,0},{25,25}}</string>
            <key>sourceSize</key>
            <string>{25,25}</string>
        </dict>
        <key>BACKGROUND-gespa-color.png</key>
        <dict>
            <key>frame</key>
            <string>{{995,889},{25,25}}</string>
            <key>offset</key>
            <string>{0,0}</string>
            <key>rotated</key>
            <false/>
            <key>sourceColorRect</key>
            <string>{{0,0},{25,25}}</string>
            <key>sourceSize</key>
            <string>{25,25}</string>
        </dict>
        <key>arbre-darrera-1.png</key>
        <dict>
            <key>frame</key>
            <string>{{545,992},{160,272}}</string>
            <key>offset</key>
            <string>{0,1}</string>
            <key>rotated</key>
            <true/>
            <key>sourceColorRect</key>
            <string>{{120,13},{160,272}}</string>
            <key>sourceSize</key>
            <string>{400,300}</string>
        </dict>
        <key>arbre-darrera-3.png</key>
        <dict>
            <key>frame</key>
            <string>{{545,828},{162,278}}</string>
            <key>offset</key>
            <string>{0,1}</string>
            <key>rotated</key>
            <true/>
            <key>sourceColorRect</key>
            <string>{{119,10},{162,278}}</string>
            <key>sourceSize</key>
            <string>{400,300}</string>
        </dict>
        <key>arbres-darrera-2.png</key>
        <dict>
            <key>frame</key>
            <string>{{348,1036},{160,270}}</string>
```

Conceptes bàsics: Sprite Batch

```
CCSpriteBatchNode *spritebatch = CCSpriteBatchNode::create("assets/arbreLectura.pvr.ccz");
treeMenu->addChild(spritebatch);
CCSpriteFrameCache::sharedSpriteFrameCache()->addSpriteFramesWithFile("assets/arbreLectura.plist");

CCSprite* background = CCSprite::createWithSpriteFrameName("bg1.png");
background->setPosition(ccp(10, 10));
CCSprite* background2 = CCSprite::createWithSpriteFrameName("bg2.png");
background2->setPosition(ccp(10, 10));
treeMenu->addChild(background);
treeMenu->addChild(background2);
```

Conceptes bàsics: CCAnimation

```
CCSpriteFrameCache* cache = CCSpriteFrameCache::sharedSpriteFrameCache();
cache->addSpriteFramesWithFile("Screens/Cover/cover.plist");

// Initialize spritebatch for the foreground
CCSpriteBatchNode *foregroundBatch = CCSpriteBatchNode::create("Screens/Cover/cover.pvr.ccz");

// add the sprite as a child to this layer
this->addChild(foregroundBatch);

CCSize s = CCDirector::sharedDirector()->getWinSize();
cache->addSpriteFramesWithFile("Animations/Spritesheet.plist");
CCSpriteBatchNode *animationBatch = CCSpriteBatchNode::create("Animations/Spritesheet.pvr.ccz");
this->addChild(animationBatch);

CCSprite *sprite = CCSprite::createWithSpriteFrameName("BirdFlop001.png");
sprite->setPosition(ccp(s.width * 0.6f, s.height * 0.5f));

CCArray* animFrames = CCArray::create();
char str[100] = {0};
for(int i = 1; i < 8; i++)
{
    // Obtain frames by alias name
    sprintf(str, "BirdFlop%03d.png", i);
    CCSpriteFrame *frame = cache->spriteFrameByName(str);
    animFrames->addObject(frame);
}

CCAnimation *animation = CCAnimation::create(animFrames, 1.0f/30.0f);

sprite->runAction(CCRepeatForever::create(CCAnimate::create(animation)));
animationBatch->addChild(sprite);
```

Conceptes bàsics: CCMenu

Menu#1

Menu#2

Menu#3

Menu#1

Menu#2

Menu#3

```
// Setup menu backgrounds
CCSprite *menuBG1 = CCSprite::createWithSpriteFrameName("1white-button-large.png");
CCSprite *menuBG2 = CCSprite::createWithSpriteFrameName("2white-button-large.png");
CCSprite *menuBG3 = CCSprite::createWithSpriteFrameName("3white-button-large.png");

CCMenuItemSprite *menu1 = CCMenuItemSprite::create(menuBG1, NULL, NULL, this,
menu_selector(SubShop::clickAvatar));
menu1->setAnchorPoint(ccp(0,0));
menu1->setPosition(ccp(150, 470));

CCMenuItemSprite *menu2 = CCMenuItemSprite::create(menuBG2, NULL, NULL, this,
menu_selector(SubShop::clickMoves));
menu2->setAnchorPoint(ccp(0,0));
menu2->setPosition(ccp(150, 360));

CCMenuItemSprite *menu3 = CCMenuItemSprite::create(menuBG3, NULL, NULL, this,
menu_selector(SubShop::clickBackground));
menu3->setAnchorPoint(ccp(0,0));
menu3->setPosition(ccp(150, 250));

CCMenu* menu = CCMenu::create(menu1, menu2, menu3, NULL);
menu->setAnchorPoint(ccp(0,0));
menu->setPosition(ccp(0,0));

this->addChild(menu);
```

Conceptes bàsics: CCMenu

Menu#1

Menu#2

Menu#3

Menu#1

Menu#2

Menu#3

```
// Setup menu backgrounds
CCSprite *menuBG1 = CCSprite::createWithSpriteFrameName("1white-button-large.png");
CCSprite *menuBG2 = CCSprite::createWithSpriteFrameName("2white-button-large.png");
CCSprite *menuBG3 = CCSprite::createWithSpriteFrameName("3white-button-large.png");

CCMenuItemSprite *menu1 = CCMenuItemImage::create(menuBG1, NULL, NULL, this, menu_selector(SubShop::clickAvatar));
menu1->setAnchorPoint(ccp(0,0));
menu1->setPosition(ccp(150, 470));

CCMenuItemSprite *menu2 = CCMenuItemImage::create(menuBG2, NULL, NULL, this, menu_selector(SubShop::clickMoves));
menu2->setAnchorPoint(ccp(0,0));
menu2->setPosition(ccp(150, 360));

CCMenuItemSprite *menu3 = CCMenuItemImage::create(menuBG3, NULL, NULL, this, menu_selector(SubShop::clickBackground));
menu3->setAnchorPoint(ccp(0,0));
menu3->setPosition(ccp(150, 250));

CCMenu* menu = CCMenu::create(menu1, menu2, menu3, NULL);
menu->setAnchorPoint(ccp(0,0));
menu->setPosition(ccp(0,0));

this->addChild(menu);
```

Conceptes bàsics: CCAction

Es poden aplicar a qualsevol CCNode (Layers, Sprites, Menús, Escenes...)

Exemple:

```
CCRepeatForever *repeat = CCRepeatForever::create(CCRotateBy::create(10.0f, 360));
myCCNode->runAction(repeat);
```

Conceptes bàsics: CCAction

Acció de escala

```
CCActionInterval* actionTo = CCScaleTo::create(0.8f, 2.5f);  
myObject->runAction(actionTo);
```

Conceptes bàsics: CCAction

Interpolació de la velocitat

```
CCActionInterval* actionTo = CCScaleTo::create(0.8f, 2.5f);
CCAction *easeln = CCEaseIn::actionWithAction(actionTo, 5);

myObject->runAction(easeln);
```

Conceptes bàsics: CCAction

Concatenant accions en sequències

```
CCActionInterval *actionTo = CCScaleTo::create(0.8f, 2.5f);
CCAction *easeln = CCEaseln::actionWithAction(actionTo, 5);
CCMoveBy *move = CCMoveBy::create(0.8f, ccp(100, 200));
CCFiniteTimeAction* actF = CCSequence::create(move, easeln, NULL);

myObject->runAction(actF);
```

Conceptes bàsics: CCAction

Executant accions en paral·lel

```
CCActionInterval *actionTo = CCScaleTo::create(0.8f, 2.5f);
CCMoveBy *move = CCMoveBy::create(0.8f, ccp(100, 200));
CCAction *finalAction = CCSpawn::create(actionTo, move);

menuPlayItem->runAction(finalAction);
```

Conceptes bàsics: CCAction

position

[CCMoveBy](#)
[CCMoveTo](#)
[CCJumpBy](#)
[CCJumpTo](#)
[CCBezierBy](#)
[CCBezierTo](#)
[CCPlace](#)

scale

[CCScaleBy](#)
[CCScaleTo](#)

rotation

[CCRotateBy](#)
[CCRotateTo](#)

visible

[CCShow](#)
[CCHide](#)
[CCBlink](#)
[CCToggleVisibility](#)

opacity

[CCFadeIn](#)
[CCFadeOut](#)
[CCFadeTo](#)

color

[CCTintBy](#)
[CCTintTo](#)

Conceptes bàsics: Estructures de dades



Classes importants

`CCArray`

`CCDictionary` (es pot inicialitzar amb un XML! i els XML es poden binaritzar!)

`CCPoint`

`CCSize`

`CCRect`

`CCInteger`

`CCString`

`CCStringCompare`

Conceptes bàsics: Sistemes de partícules



Gravity Mode

```
CCParticleSystemQuad* m_emitter = newCCParticleSystemQuad();
m_emitter = CCParticleFire::create();

// Gravity Mode
this->m_nEmitterMode = kCCParticleModeGravity;

// Gravity Mode: gravity
this->modeA.gravity = ccp(0,-90);
```



Conceptes bàsics: Sistemes de partícules



Gravity Mode

```
// Radius Mode  
this->m_nEmitterMode = kCCParticleModeRadius;  
  
// Radius Mode: startRadius  
this->modeB.startRadius = 0;  
this->modeB.startRadiusVar = 0;//ccp(0,0);
```



Conceptes bàsics: schedulers

Mètodes que es criden repetitivament en un interval de temps

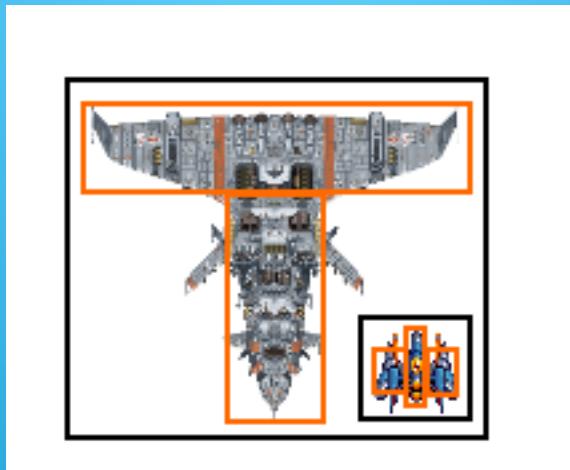
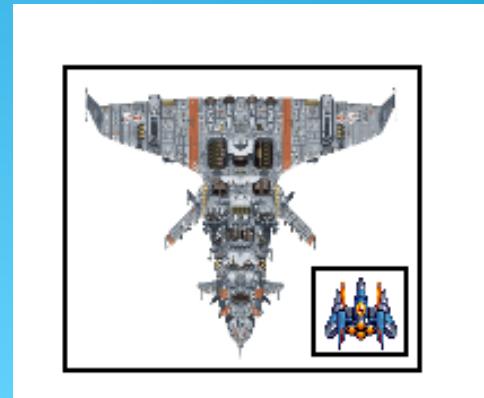
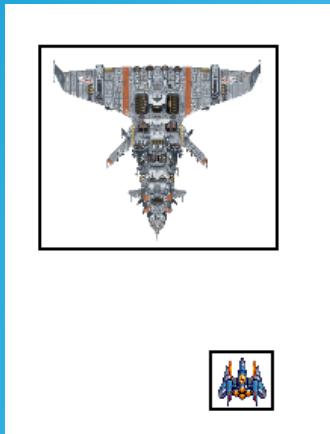
```
this->schedule(MenuBottom::methodName, 1.0f);
```

A tenir en compte:

- Posar-ne molts afecta al rendiment, i a la mantenibilitat del codi
- Pauses / resumes automàtics (CCDirector::sharedDirector()->pause();

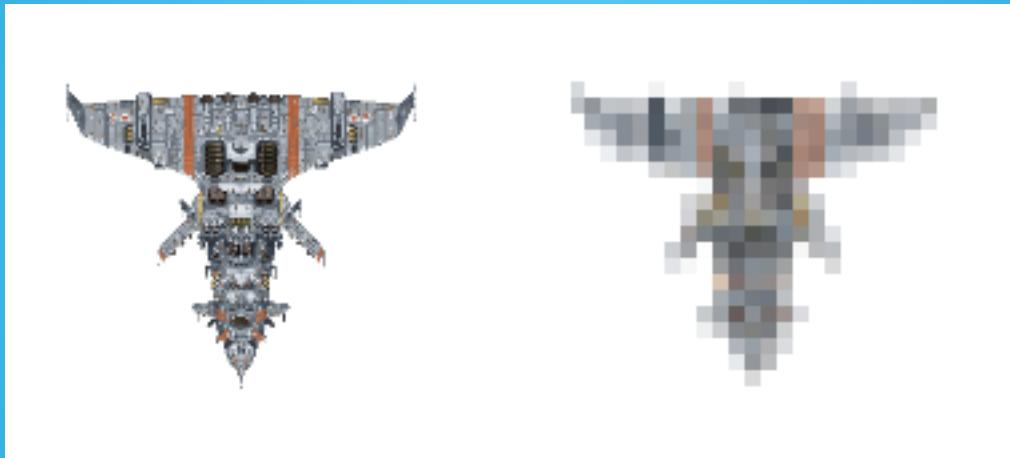
Conceptes bàsics: detecció de col·lisions

Basades en rectangles



Conceptes bàsics: detecció de col·lisions

Pixel perfect



Conceptes bàsics: detecció de col·lisions



Exemple bàsic sense fantasies

```
CCRect targetRect = CCRectMake(  
    target->getPosition().x - (target->getContentSize().width/2),  
    target->getPosition().y - (target->getContentSize().height/2),  
    target->getContentSize().width,  
    target->getContentSize().height);  
  
if (projectileRect.intersectsRect(targetRect))  
{  
    // bullet impact, aarrrgggh! (so, do something here)  
}
```

Algunes trucs



Plungeinteractive

Trucs, consells, bones pràctiques...

- Utilitza sprite sheets + sprite batch
- Utilitza sprite fonts (no labels)

CCLabelBMFont enlloc de CCLabelTTF

- Sempre que sigui possible, utilitza textures de 16 o 4 bits i no 32

CCTexture2D::setDefaultAlphaPixelFormat(kTexture2DPixelFormat_RGBA4444);

- Elimina les textures de la caché! O es queden en memòria...

```
CCSpriteFrameCache::sharedSpriteFrameCache()->removeUnusedSpriteFrames();
CCTextureCache::sharedTextureCache()->removeUnusedTextures();
```

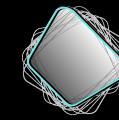
```
CCTextureCache::sharedTextureCache()->removeAllTextures();
```

- No programis els schedulers perquè vagin més ràpid que els FPS del joc
- Inicialitza els CCNodes al init() preferiblement
- No creis una jerarquia massa gran de CCLayers

Tests



Plunge interactive



Plunge **interactive**

Ara us toca a vosaltres ☺