Plunge interactive

Cocos2d-x!

Videojocs multi-plataforma amb C++

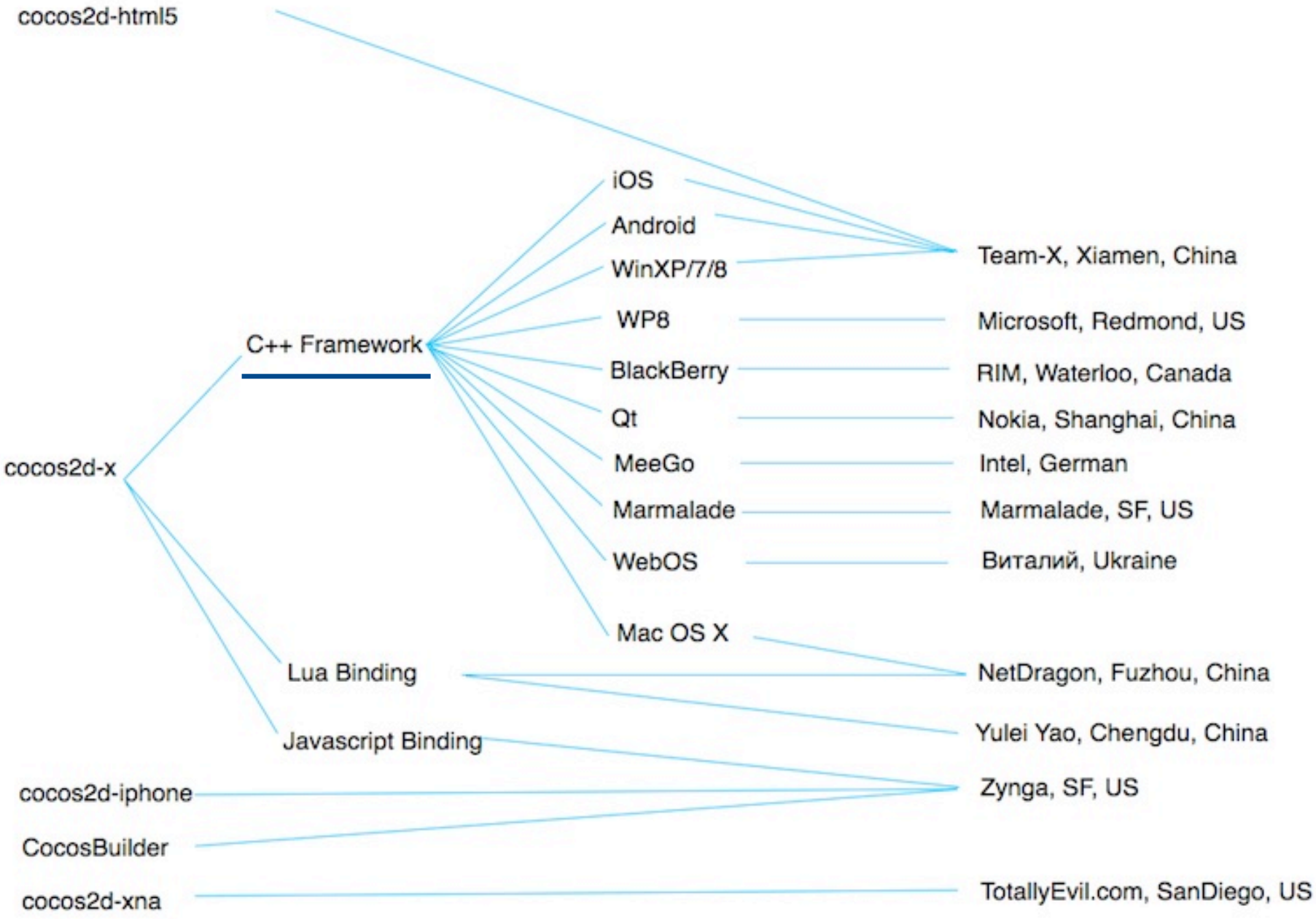
Plunge interactive

Hola Cocos!

Plunge interactive

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

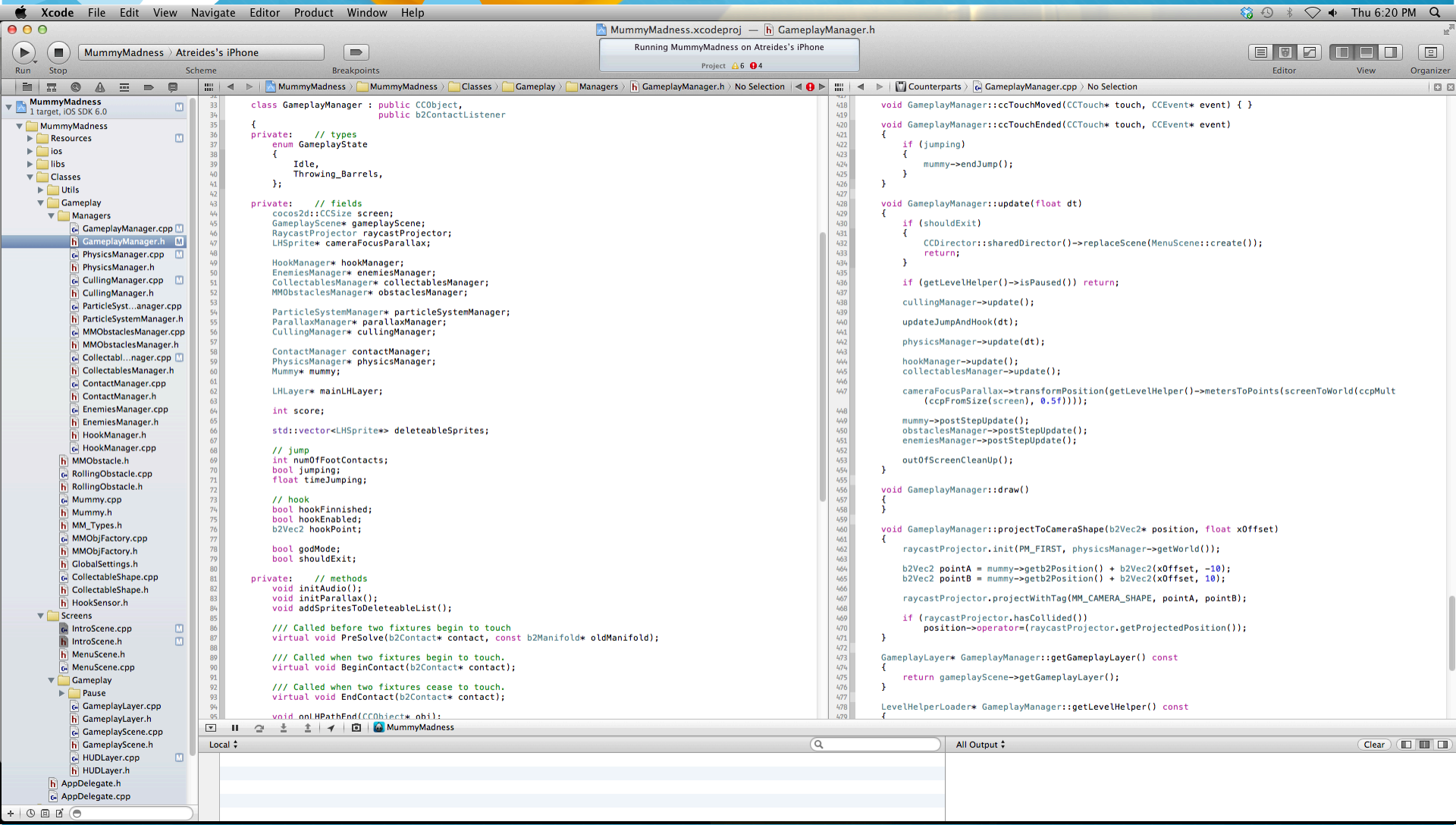

cocos2d-html5
cocos2d-x
C++ Framework
Lua Binding
Javascript Binding
iOS
Android
WinXP/7/8
WP8
BlackBerry
Qt
MeeGo
Marmalade
WebOS
Mac OS X
Team-X, Xiamen, China
Microsoft, Redmond, US
RIM, Waterloo, Canada
Nokia, Shanghai, China
Intel, German
Marmalade, SF, US
Виталий, Ukraine
NetDragon, Fuzhou, China
Yulei Yao, Chengdu, China
cocos2d-iphone
CocosBuilder
Zynga, SF, US
cocos2d-xna
TotallyEvil.com, SanDiego, US

Tools

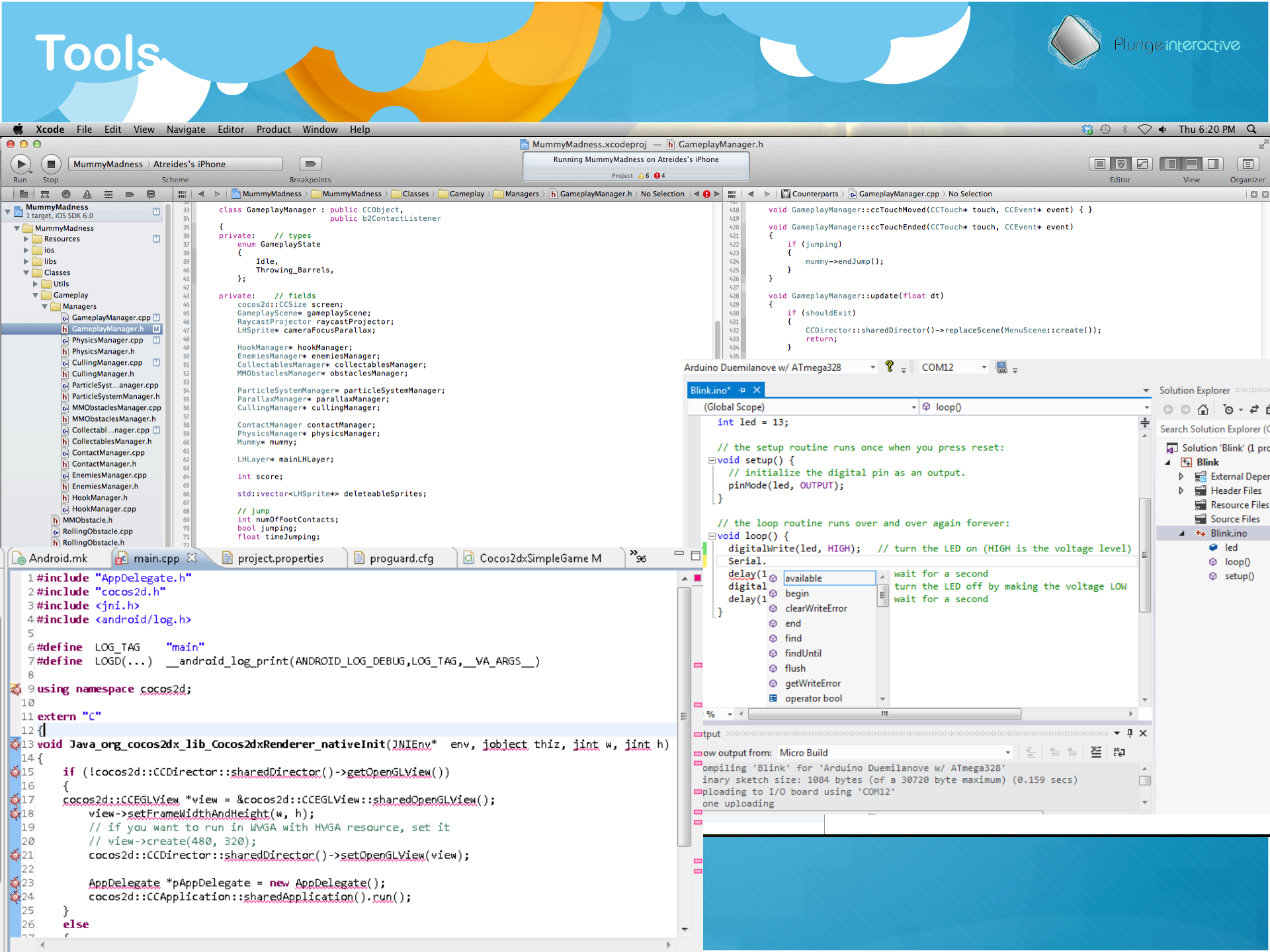
Plunge interactive

Tools

Plunge interactive

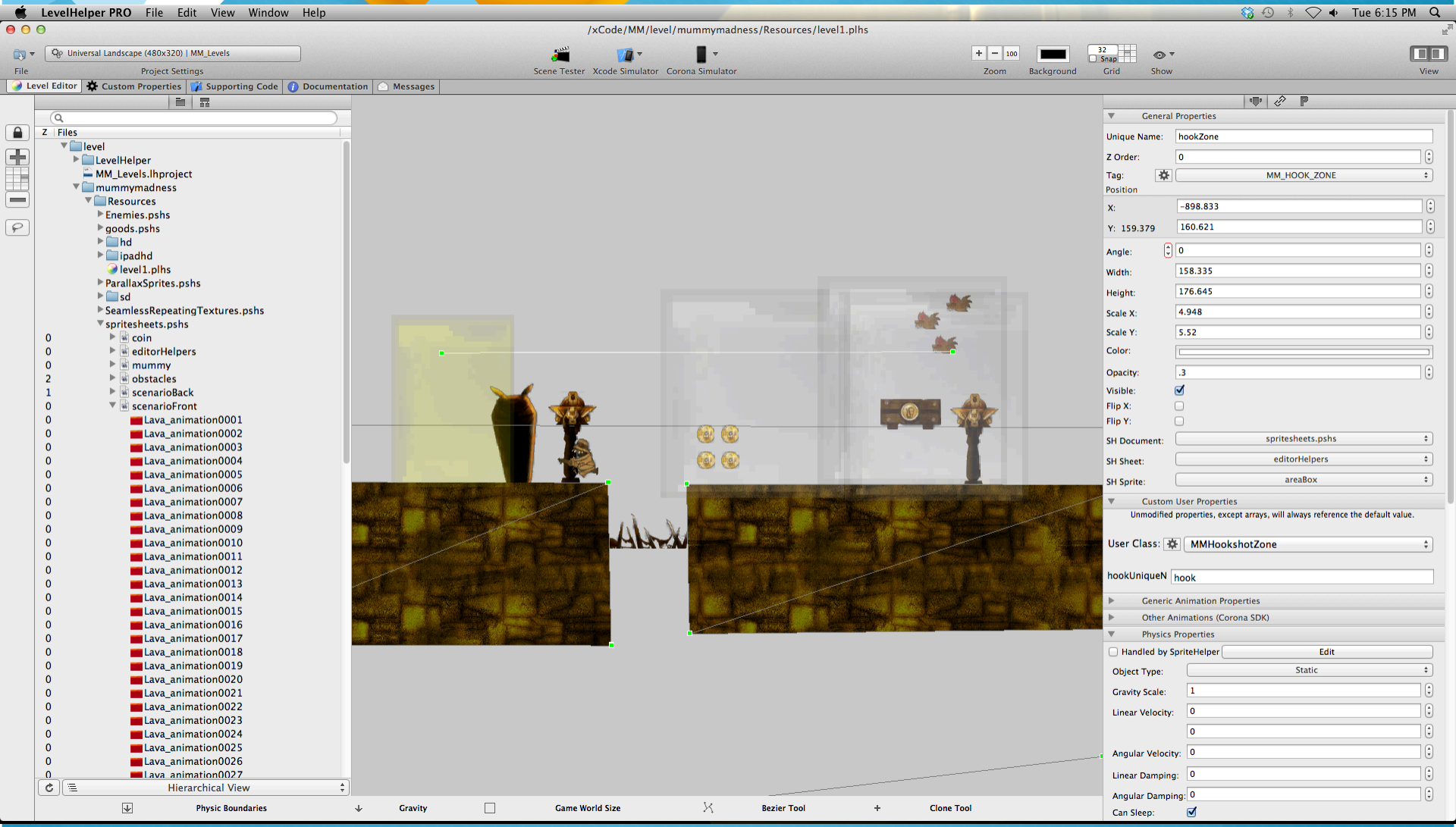


Tools


Xcode File Edit View Navigate Editor Product Window Help
Thu 6:20 PM
MummyMadness.xcodeproj — GameplayManager.h
Running MummyMadness on Atreides's iPhone
Run Stop Scheme Breakpoints Editor View Organizer
MummyMadness
1 target, iOS SDK 6.0
Resources
ios
libs
Classes
Utils
Gameplay
Managers
GameplayManager.cpp
GameplayManager.h
PhysicsManager.cpp
PhysicsManager.h
CullingManager.cpp
CullingManager.h
MMObstacle.h
RollingObstacle.cpp
RollingObstacle.h
class GameplayManager : public CCObject,
public b2ContactListener
{
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 numOfFootContacts;
bool jumping;
float timeJumping;
void GameplayManager::ccTouchMoved(CCTouch* touch, CCEvent* event) { }
void GameplayManager::ccTouchEnded(CCTouch* touch, CCEvent* event)
{
if (jumping)
{
mummy->endJump();
}
}
void GameplayManager::update(float dt)
{
if (shouldExit)
{
CCDirector::sharedDirector()->replaceScene(MenuScene::create());
return;
}
Android.mk
main.cpp
project.properties
proguard.cfg
Cocos2dxSimpleGame M
#include "AppDelegate.h"
#include "cocos2d.h"
#include <jni.h>
#include <android/log.h>
#define LOG_TAG "main"
#define LOGD(...) __android_log_print(ANDROID_LOG_DEBUG,LOG_TAG,__VA_ARGS__)
using namespace cocos2d;
extern "C"
{
void Java_org_cocos2dx_lib_Cocos2dxRenderer_nativeInit(JNIEnv* env, jobject thiz, jint w, jint h)
{
if (!cocos2d::CCDirector::sharedDirector()->getOpenGLView())
{
cocos2d::CCEGLView *view = &cocos2d::CCEGLView::sharedOpenGLView();
view->setFrameWidthAndHeight(w, h);
// if you want to run in WVGA with HVGA resource, set it
// view->create(480, 320);
cocos2d::CCDirector::sharedDirector()->setOpenGLView(view);
AppDelegate *pAppDelegate = new AppDelegate();
cocos2d::CCApplication::sharedApplication().run();
}
else
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.
delay(1
digital
delay(1
}
available
begin
clearWriteError
end
find
findUntil
flush
getWriteError
operator bool
wait for a second
turn the LED off by making the voltage LOW
wait for a second
Solution Explorer
Search Solution Explorer
Solution 'Blink' (1 pro
Blink
External Deper
Header Files
Resource Files
Source Files
Blink.ino
led
loop()
setup()
tput
ow output from: Micro Build
ompiling 'Blink' for 'Arduino Duemilanove w/ ATmega328'
inary sketch size: 1084 bytes (of a 30720 byte maximum) (0.159 secs)
ploading to I/O board using 'COM12'
one uploading

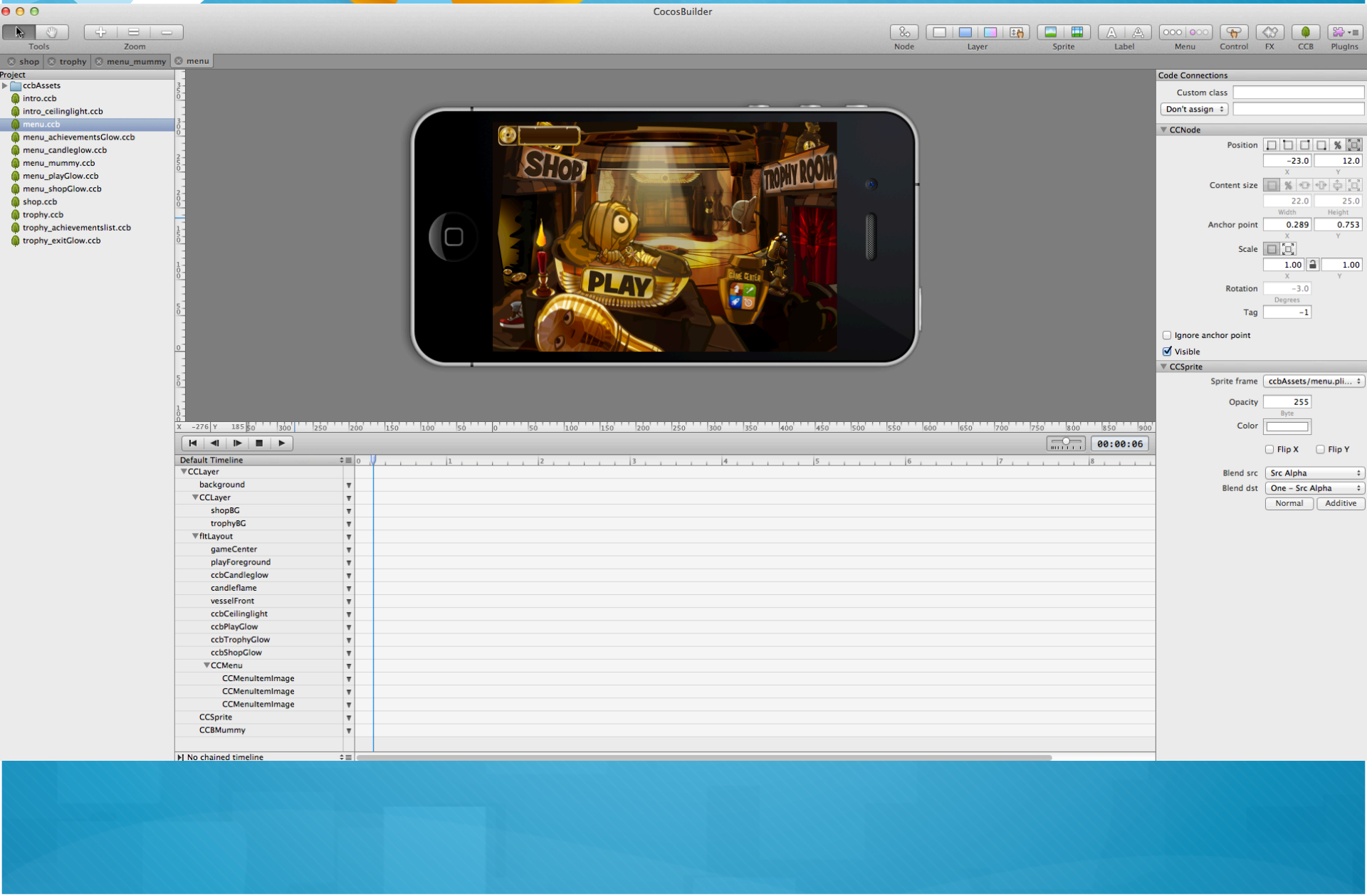
Tools

Plunge interactive



Tools

Plunge interactive



Tools

Plunge interactive

bmGlyph

Open | Save | Save as | Publish | work on selected glyph(s) only | Preview

Anti Alias

Texture

Width x Height

512px x 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

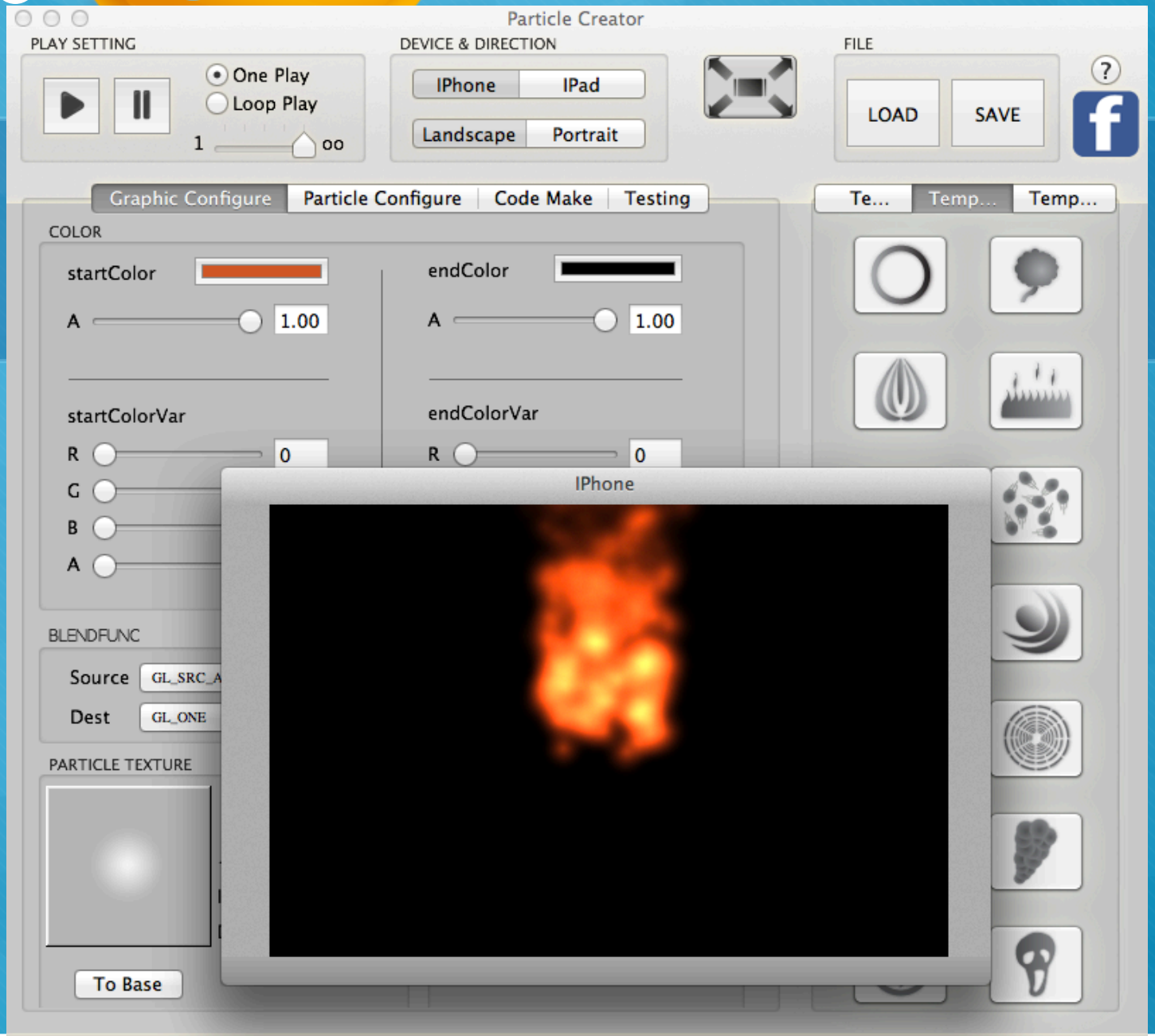
Padding : 2px

Sort Method : Special

Reversed :

Refresh

Tools


Particle Creator
PLAY SETTING
One Play
Loop Play
1
oo
DEVICE & DIRECTION
IPhone
IPad
Landscape
Portrait
FILE
LOAD
SAVE
Graphic Configure
Particle Configure
Code Make
Testing
Te...
Temp...
Temp...
COLOR
startColor
A
1.00
endColor
A
1.00
startColorVar
R
0
G
B
A
endColorVar
R
0
IPhone
BLENDFUNC
Source
GL_SRC_A
Dest
GL_ONE
PARTICLE TEXTURE
To Base

Tools

Plunge interactive

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: pritesheet/page16-ipadhd.pvr.ccz
Premultiply alpha: ☑
Flip PVR: ☐
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

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

Size: 512x1024 (2048kB)

Tools

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

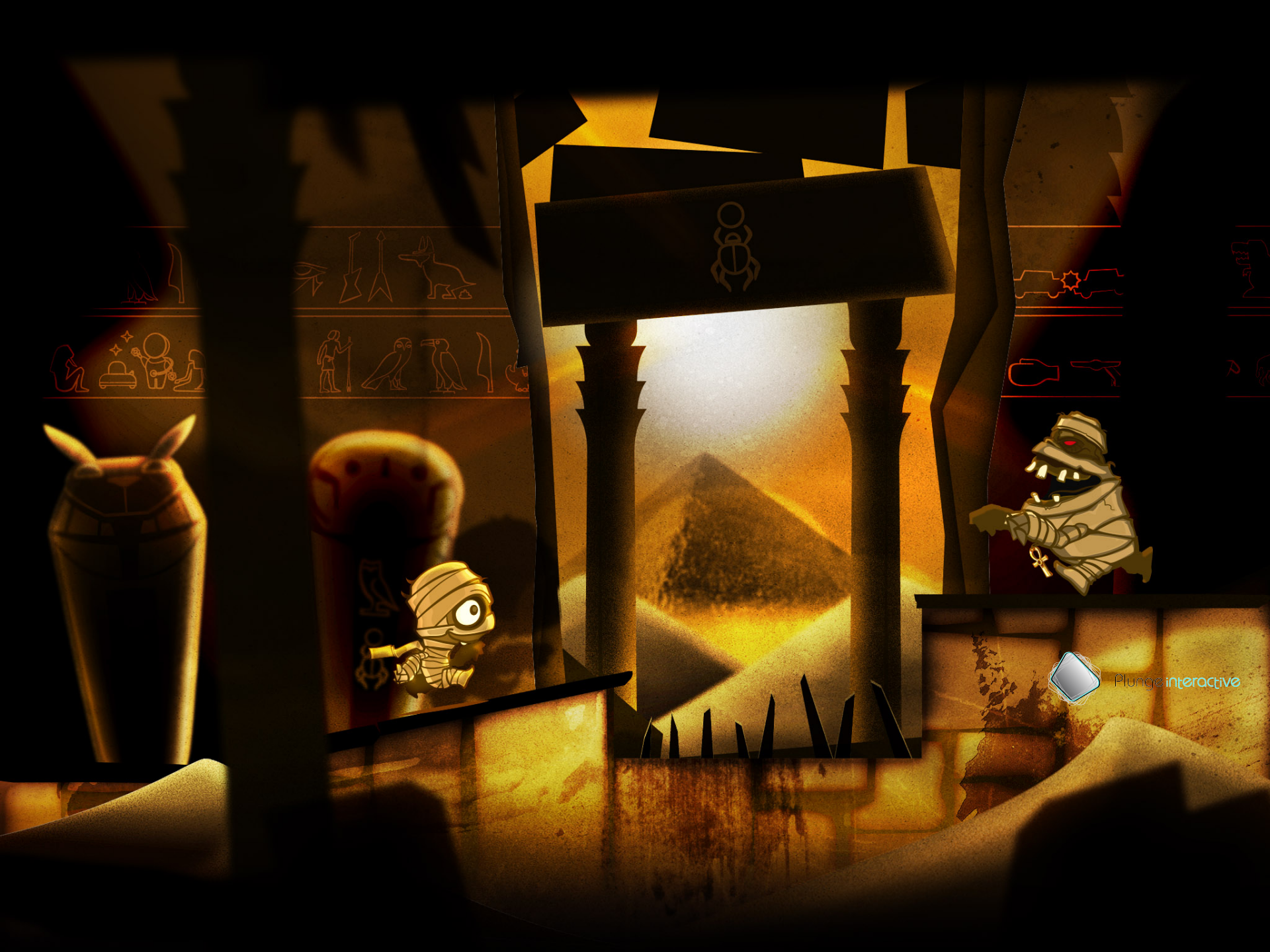
*** El codi ha d'existir per a les coses importants**

Exemples

Plunge interactive

Exemples: jocs fets a Plunge Interactive

SAY HELLO
TO CHARLIE
AND HIS FRIENDS!


Plunge interactive


Plunge interactive
INTERACTÚA
CON EL CUENTO
3

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

Disclaimer

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

Introducció a Cocos2d-x

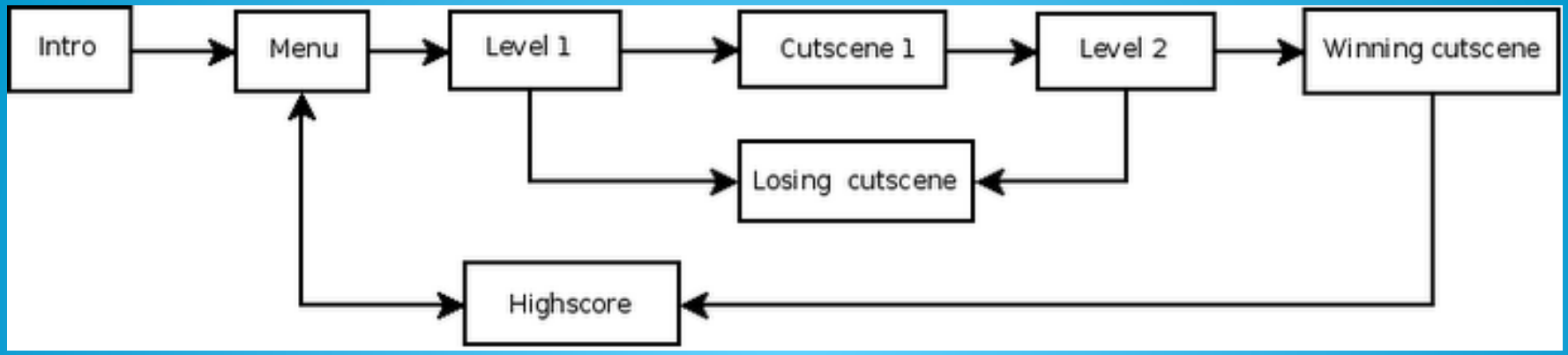
Plunge interactive

Disclaimer

Plunge interactive

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

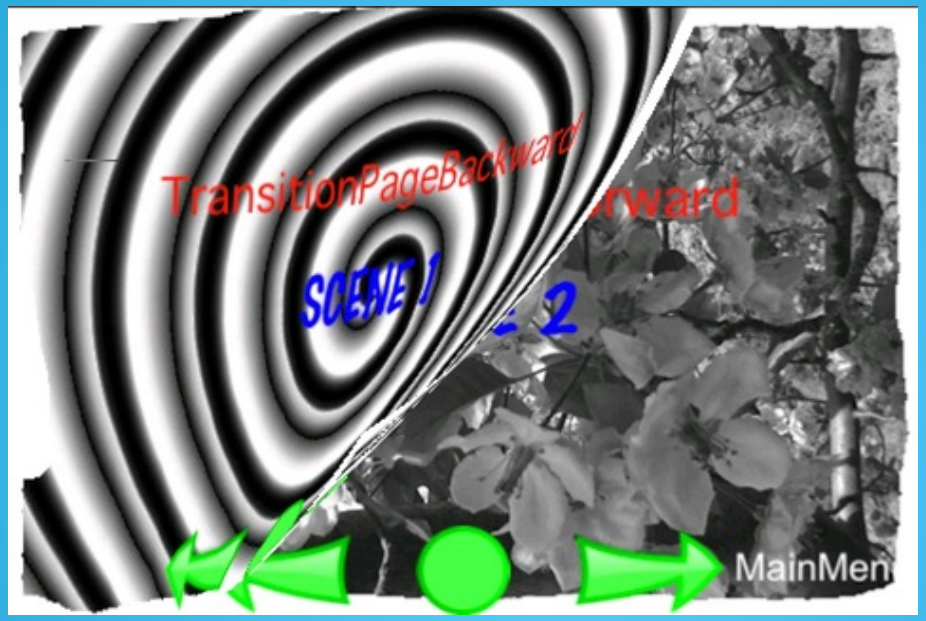
Conceptes bàsics: CCDirector


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

Conceptes bàsics: CCTransition

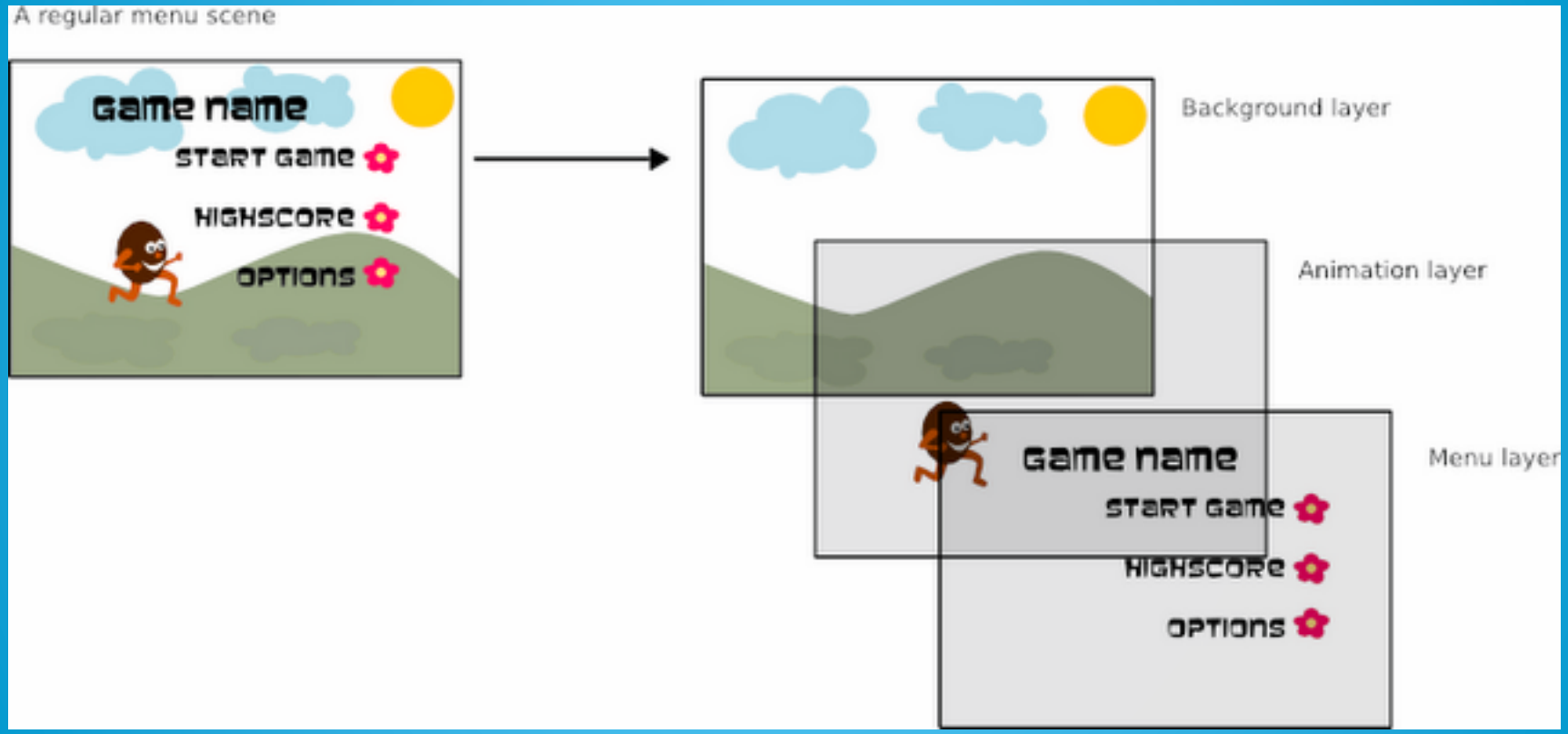
Plunge interactive

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



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

Conceptes bàsics: CCLayer



Subclasses interessants:

- **CCLayerColor**
  - **CCLayerColor**

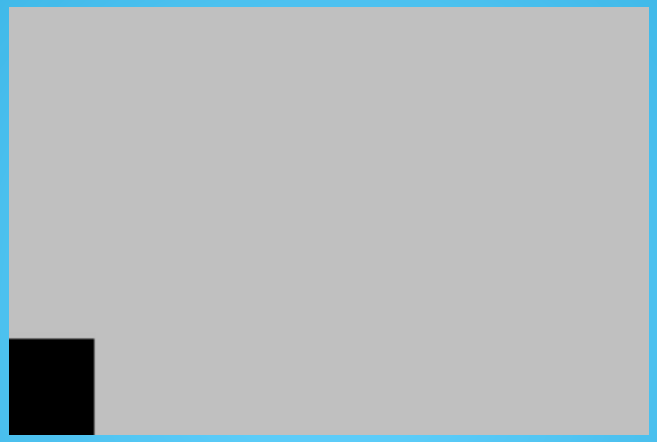
Conceptes bàsics: CCSprite

Plunge interactive

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

Plunge interactive

Molts sprites poden decrementar el framerate de manera dràstica

Conceptes bàsics: CCSprite

Plunge interactive

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

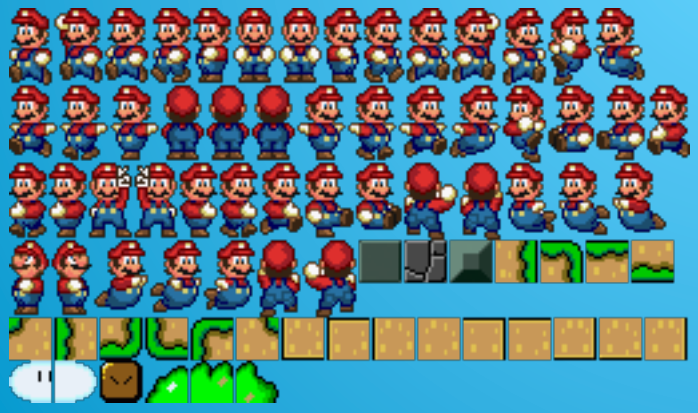
Conceptes bàsics: Sprite Batch

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)



```
xt Document ion="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.apple.
<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


```
        // 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


```
// 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

Plunge interactive

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

Plunge interactive

Acció de escala

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

Conceptes bàsics: CCAction

Plunge interactive

Interpolació de la velocitat

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

myObject->runAction(easeIn);
```

Conceptes bàsics: CCAction

Plunge interactive

Concatenant accions en sequències

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

myObject->runAction(actF);
```

Conceptes bàsics: CCAction

Plunge interactive

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

Plunge interactive

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

Plunge interactive

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

Plunge interactive

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

Plunge interactive

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);
```

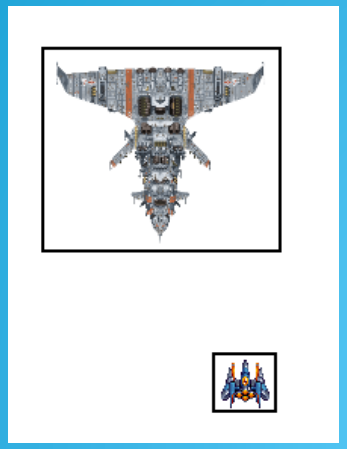
A tenir en compte:

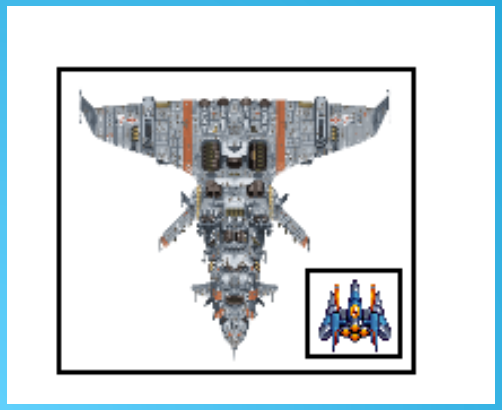
- Posar-ne molts afecta al rendiment, i a la mantenibilitat del codi
- Pauses / resumes automàtics (CCDirector::sharedDirector()->pause();

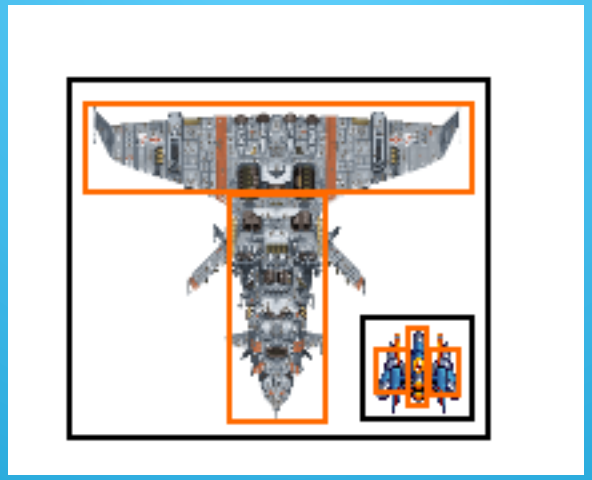
Conceptes bàsics: detecció de col·lisions

Plunge interactive

Basades en rectangles







Conceptes bàsics: detecció de col·lisions

Plunge interactive

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, aarrrggghh! (so, do something here)
}
```

Alguns trucs

Plunge interactive

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 ☺