

# **Multimedia-Programmierung**

## **Übung 4**

Ludwig-Maximilians-Universität München  
Sommersemester 2018

# Today



# Overview – Alternative Engines

**libGDX**

<https://libgdx.badlogicgames.com/>



<http://www.pygame.org/hifi.html>



<https://unity3d.com/>



<http://www.yoyogames.com/>



**SpriteKit** for Apple platforms:

<https://developer.apple.com/spritekit/>



<https://coronalabs.com/>

**And many more...**

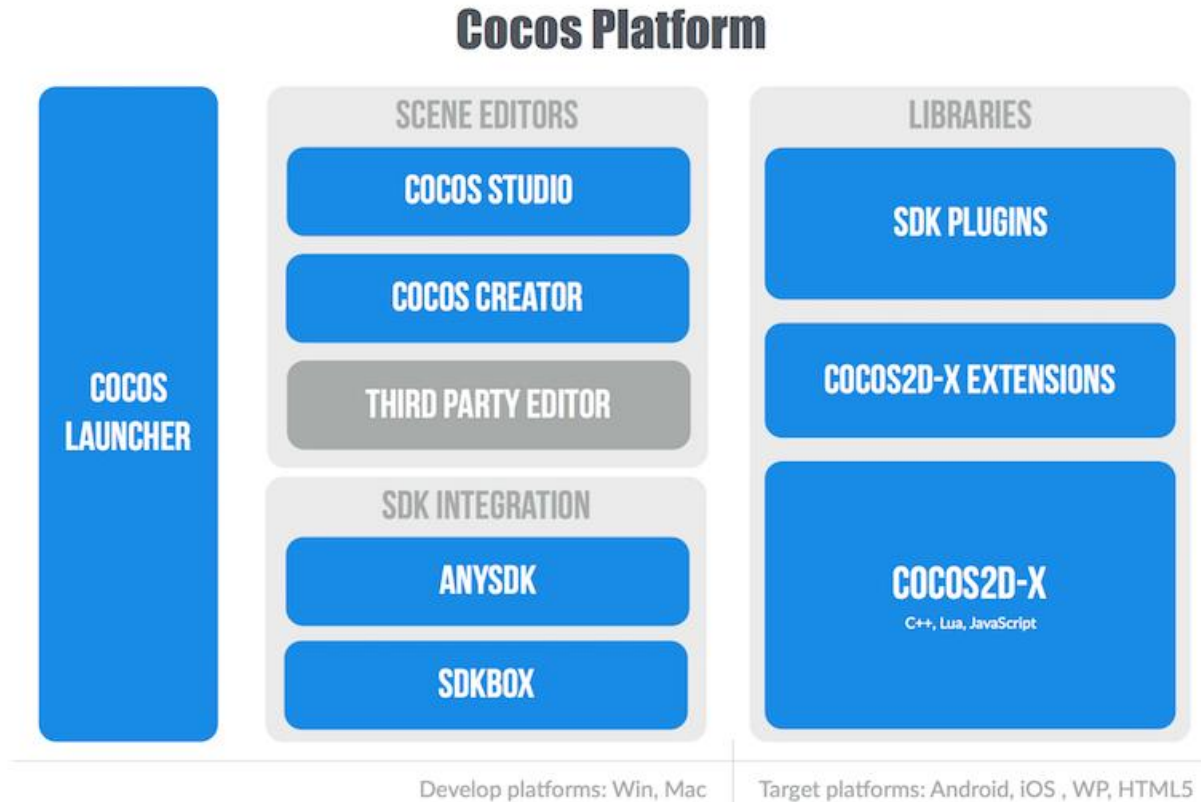
# What is Cocos and where does it come from?

- **Cocos2d-x** is an open source, cross-platform game engine
- **Cocos** is a platform to create games, including **Cocos2d-x**, editors, SDK integration



Los Cocos, Argentina

# The Platform



<http://www.cocos2d-x.org/docs/cocos/cocos/index.html>

# What is a game engine?

<http://www.cocos2d-x.org/docs/programmers-guide/1/index.html>

- A piece of software providing common functionality that all games need
- Important components:
  - Renderer
  - 2d/3d graphics
  - Collision detection
  - Physics engine
  - Animations
  - Sound
- Cross-platform: develop once and deploy to multiple platforms.

# Cocos street credibility!?



Many more: <http://www.cocos2d-x.org/games>

# Prerequisites

- A working C++ compiler
- Windows: Visual Studio Express
- Mac OS: Xcode
- Python **2.7.x** (version matters!)
- Optional: Android SDK
- And of course: cocos2d-x


<http://www.cocos2d-x.org/docs/static-pages/installation.html>

<http://www.cocos2d-x.org/docs/installation/A/index.html>



# Installation and Use of Cocos

<http://www.cocos2d-x.org/products>



**Cocos**

Cocos is a free and professional game-development toolkit, that enables developers to quickly create game content and remove the tedious work by simplifying it with straightforward GUI editors.

Cocos includes: the Cocos2d-x game engine, a game development environment and project management tool. With this suite, developers can focus on their roles and enjoy a better streamlined workflow. This saves game studios time and money by allowing them to collaborate with ease, and focus on what they do best to achieve better quality and faster turnaround time.

[DOWNLOAD](#)

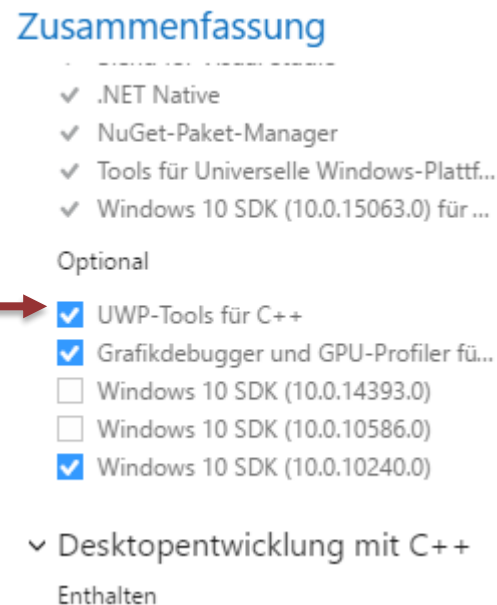
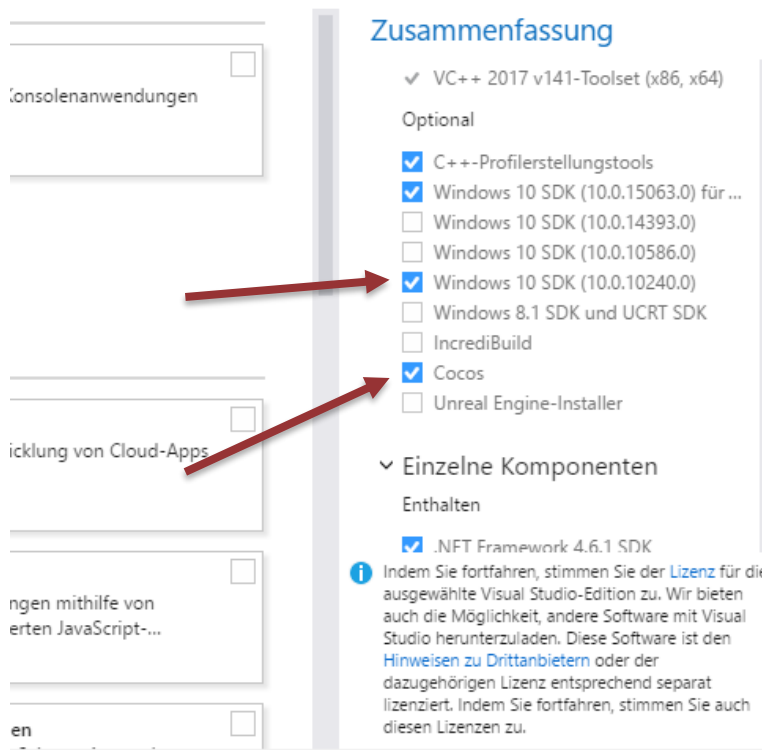
# Installation via Command Line

- Download cocos2d.zip from <http://www.cocos2d-x.org/download>
- Unzip and cd into that folder
- Use “python setup.py” to set environment variables
- Skip all SDK setups with Enter
- Restart concole

# Setup Visual Studio 2017

Check the following tools in the Visual Installer

>



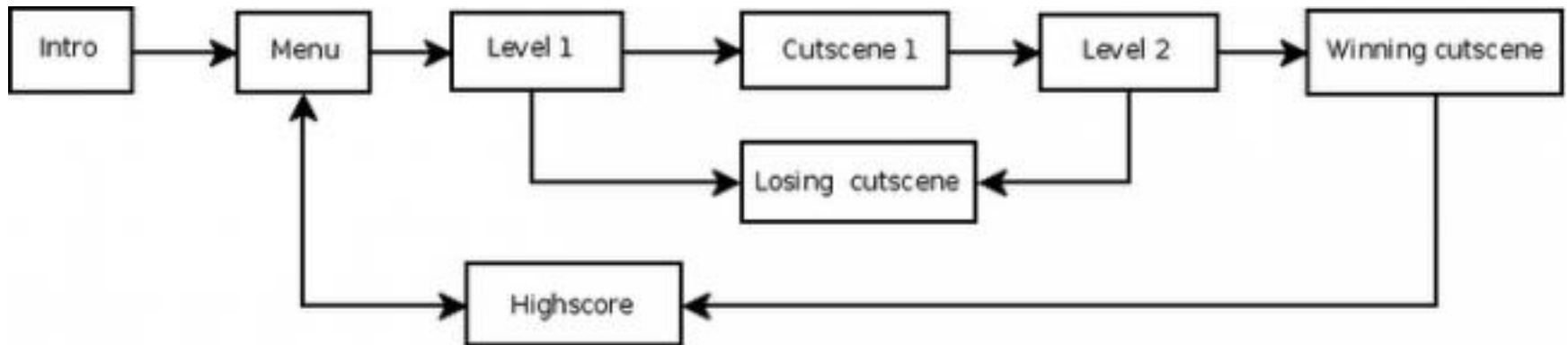
# Cocos2d-x basics

Core concepts:

Director, Scene, Node, Sprite, Action

# Cocos2d-x basics: Director

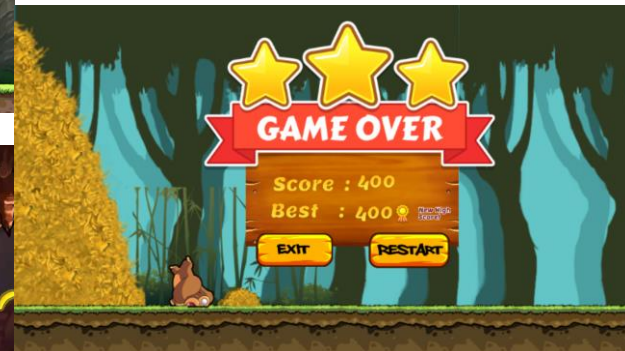
- Singleton object, always accessible
- Used to control game flow



# Cocos2d-x basics: Scene

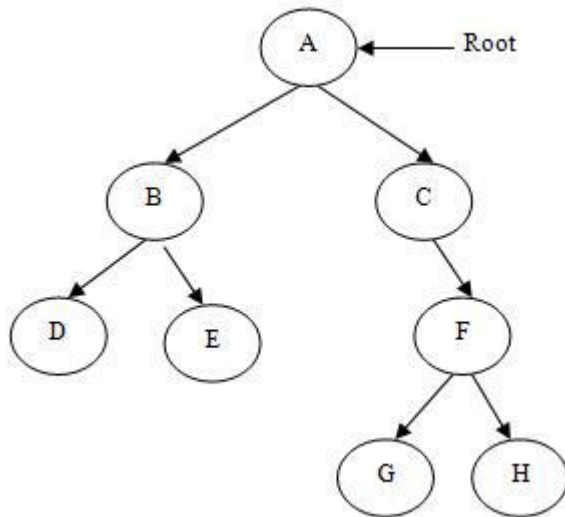
- Typical scenes: main menu, levels, game over
- Renders graphical scene objects

Game example: Banana Kong



# Cocos2d-x basics: Scenegraph

- Tree structure
- Arranges graphical scene objects in parent-child relationships
- Contains *node* objects

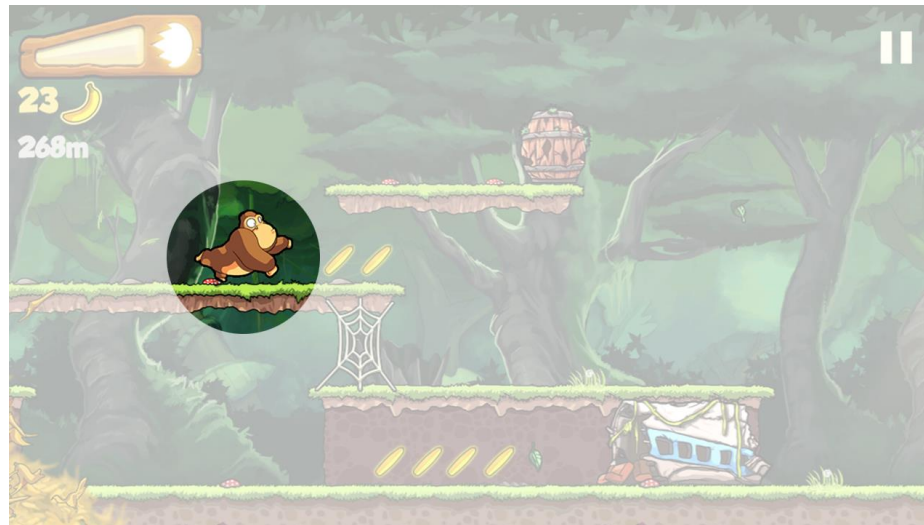


Nodes in this scene?



# Cocos2d-x basics: Sprites

- 2D images that can be moved and transformed
- Example: main character, enemies etc...
- Configurable properties: position, rotation, scale, opacity, color





# Cocos2d-x basics: Actions

- Action objects make a *Node* perform a change to its properties over time
- Example: move a sprite from one position to another over a span of time
- Example Actions: *MoveBy*, *Rotate*, *Scale*
- Applicable to all Node objects
- Also available: *sequences* and *spawns* of actions

# Building a new Project

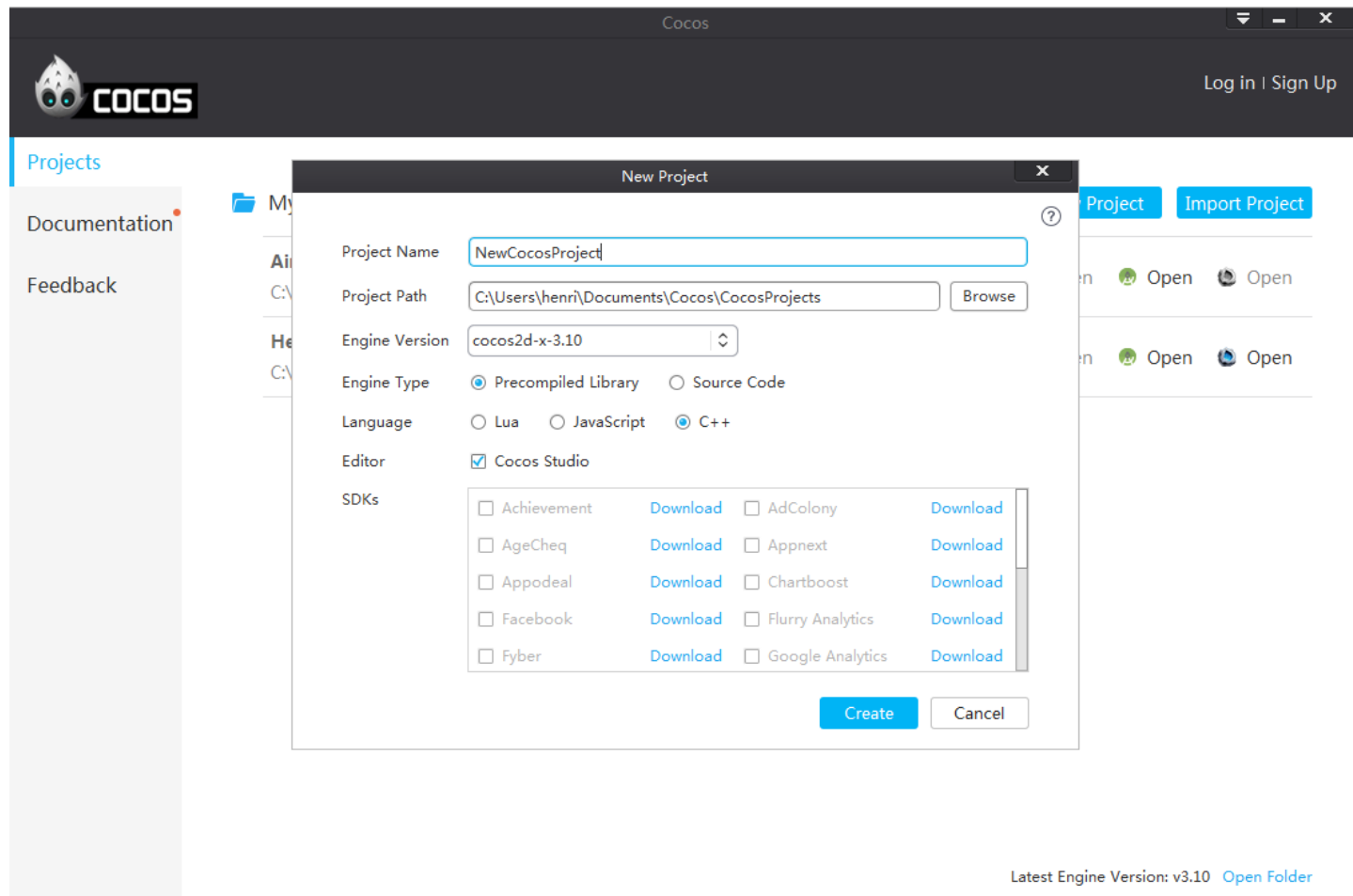
## Using Cocos GUI:

<http://www.cocos2d-x.org/docs/cocos/cocos/index.html#using-cocos>

## Using the command line tool (only for Python 2.7):

[http://www.cocos2d-x.org/docs/editors\\_and\\_tools/cocosCLTool/index.html](http://www.cocos2d-x.org/docs/editors_and_tools/cocosCLTool/index.html)

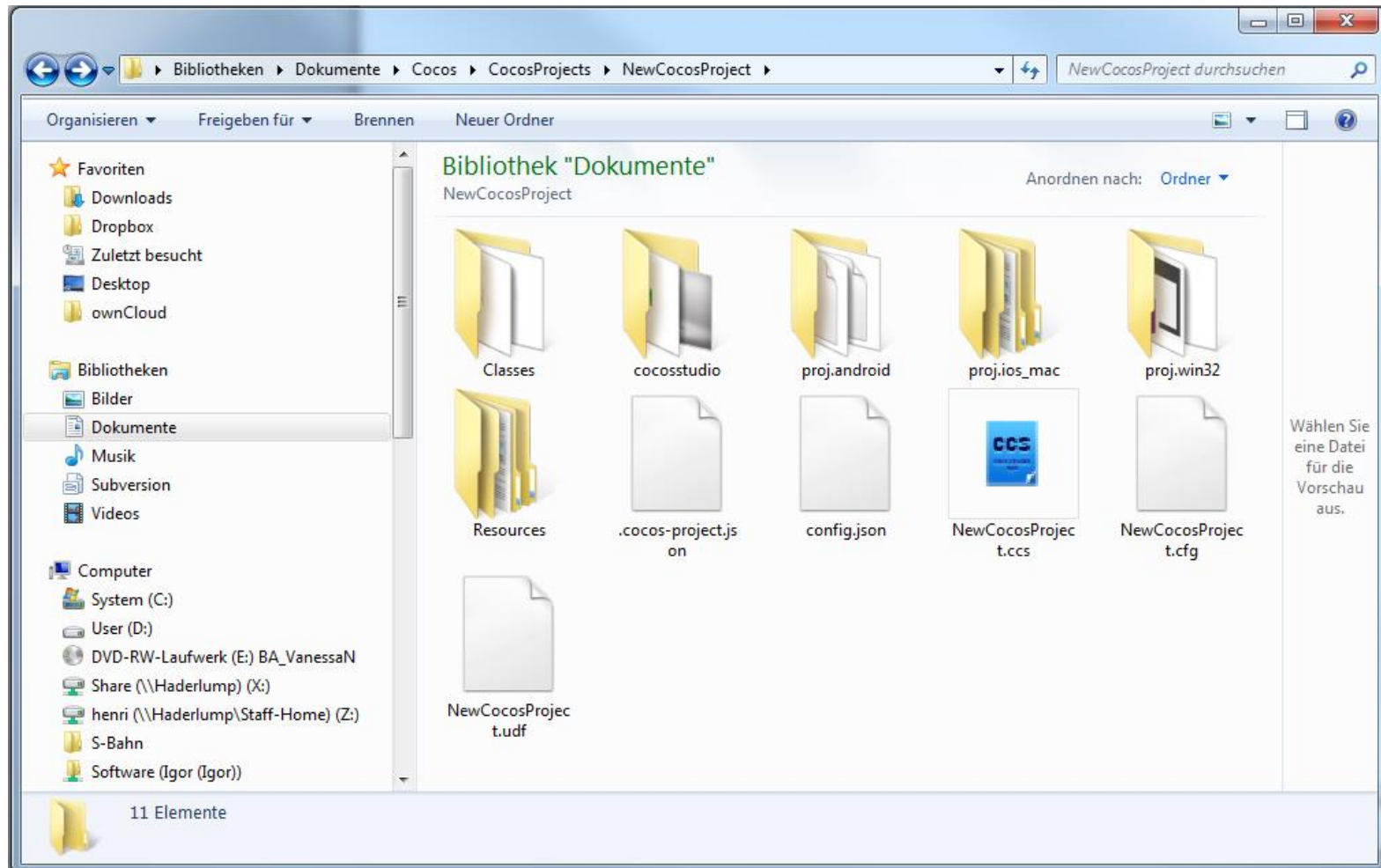
# Building a new Project using the GUI



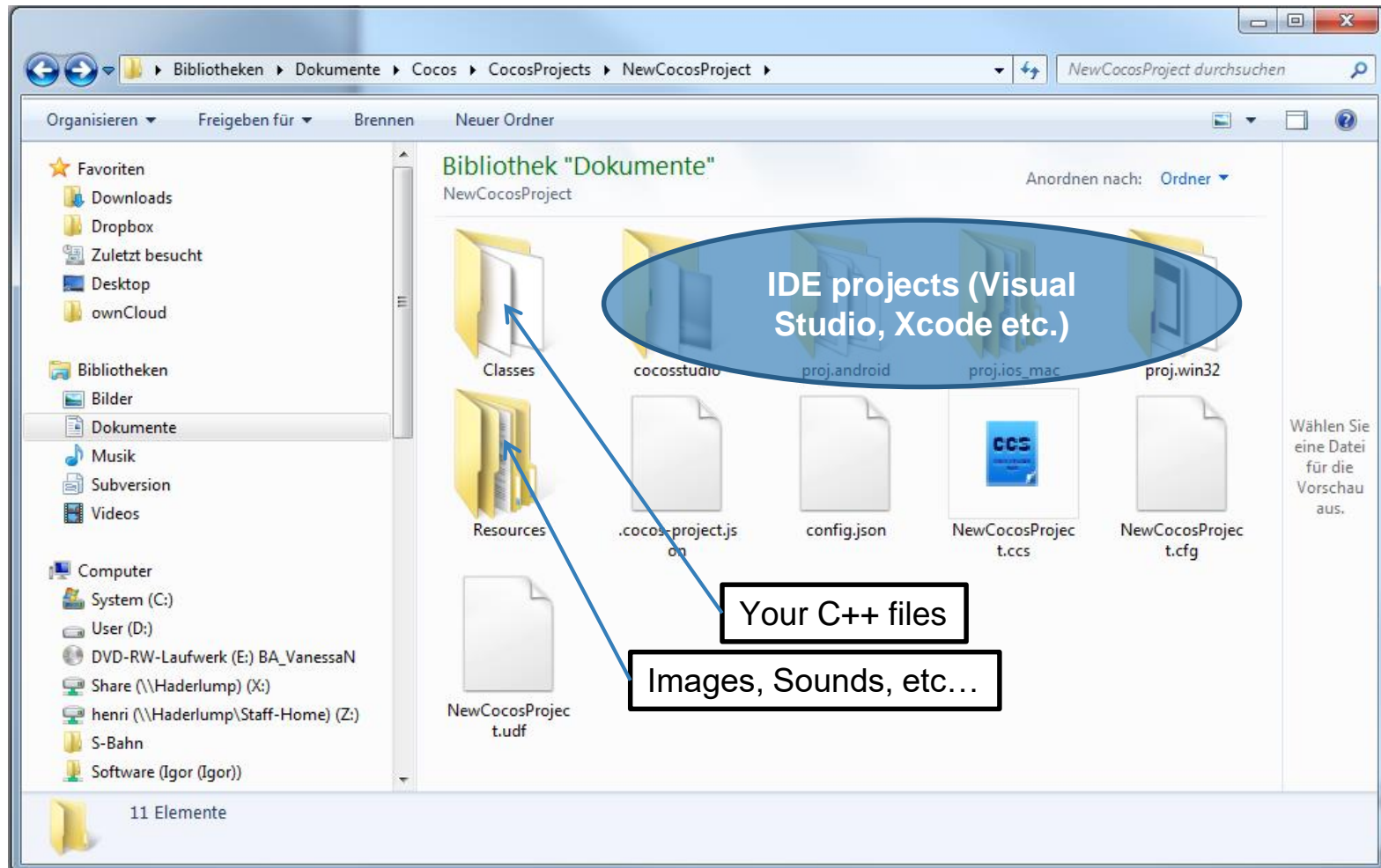
# Building a new Project using the Terminal

- Check `python --version`
- Version 2.7.x use command:  
**`cocos new -p ProjectName -d PathToDirectory`**
- Version 3.6.x need to install Python 2.7 and use console tool from cocos folder
- **`cd cocos2d-x-3.15\tools\cocos2d-console\bin`**
- Windows: **`py -2.7 cocos.py new ...`**
- Mac: **`python2 cocos.py new ...`**

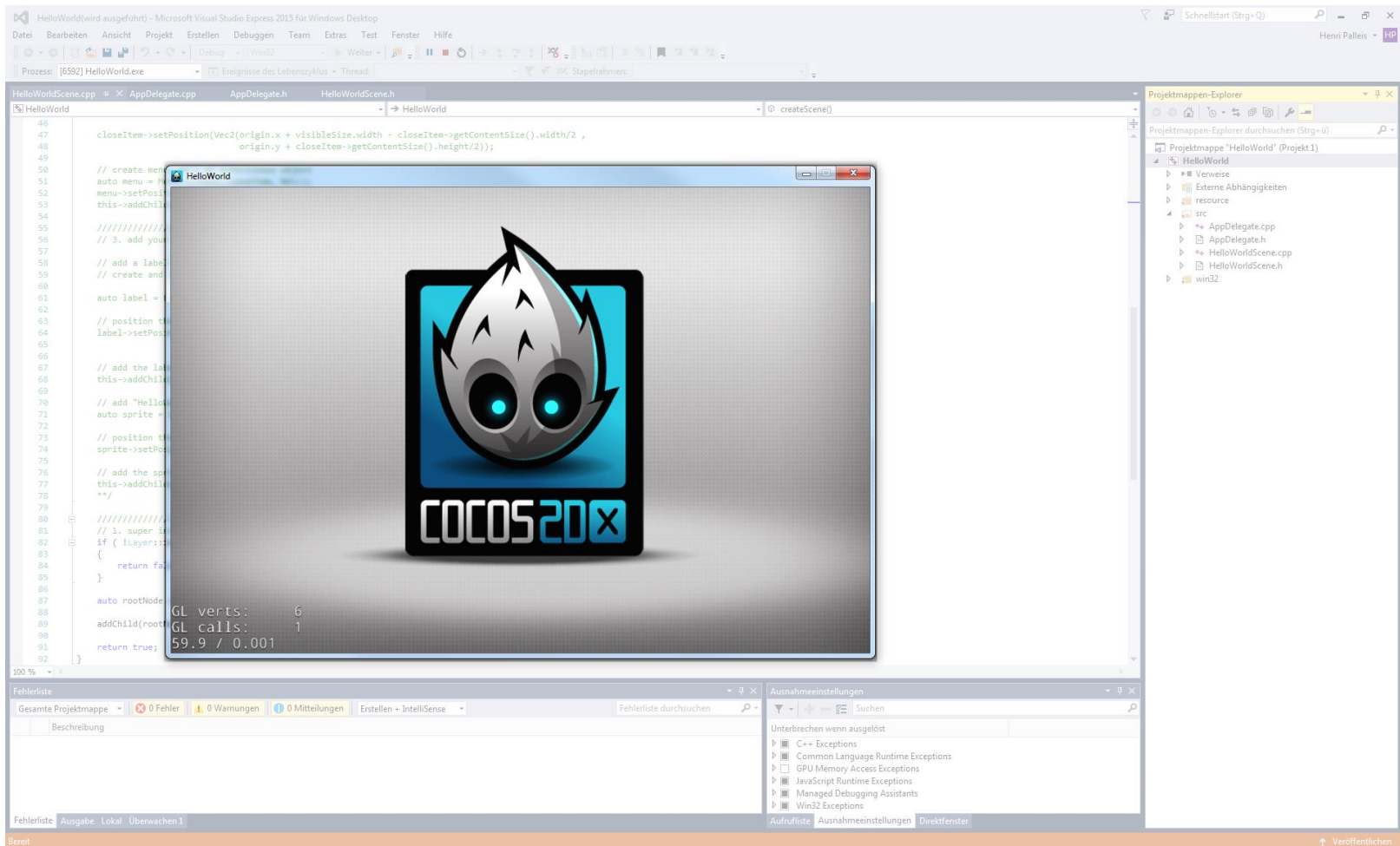
# Result



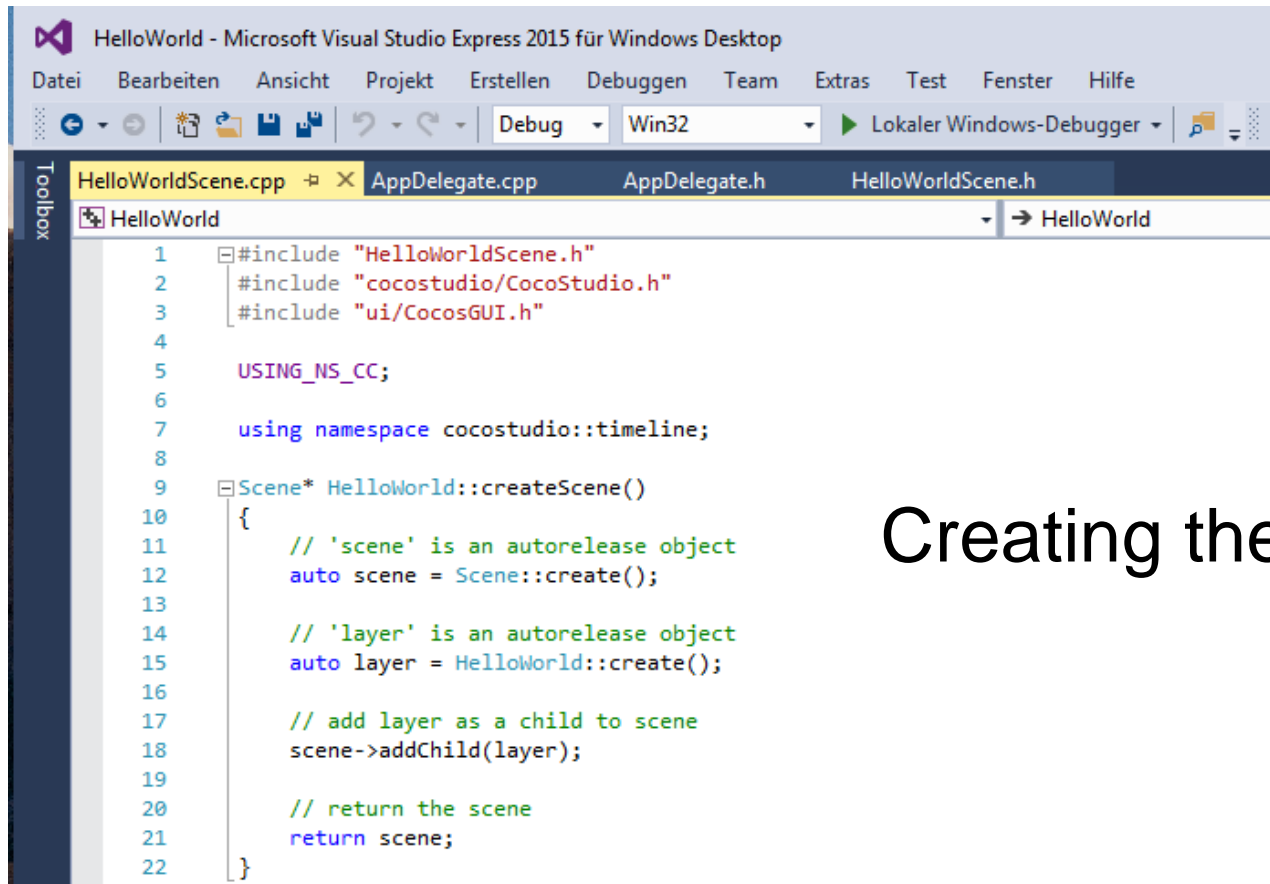
# Result



# Hello World – Auto-generated App



# Hello World – HelloWorldScene.cpp



```
1  #include "HelloWorldScene.h"
2  #include "cocostudio/CocoStudio.h"
3  #include "ui/CocosGUI.h"
4
5  USING_NS_CC;
6
7  using namespace cocostudio::timeline;
8
9  Scene* HelloWorld::createScene()
10 {
11     // 'scene' is an autorelease object
12     auto scene = Scene::create();
13
14     // 'layer' is an autorelease object
15     auto layer = HelloWorld::create();
16
17     // add layer as a child to scene
18     scene->addChild(layer);
19
20     // return the scene
21     return scene;
22 }
```

Creating the scene

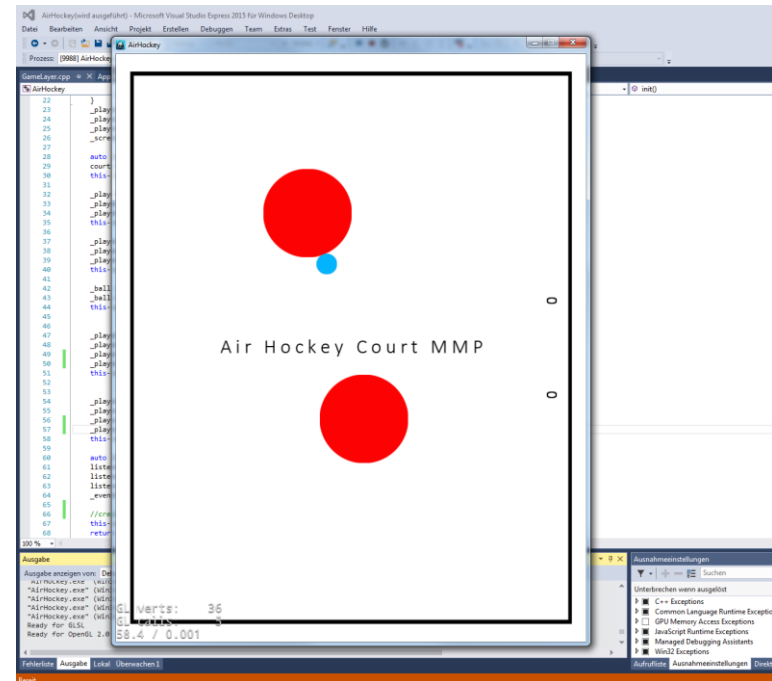


# Hello World – AppDelegate.cpp

```
24
25 bool AppDelegate::applicationDidFinishLaunching() {
26     // initialize director
27     auto director = Director::getInstance();
28     auto glview = director->getOpenGLView();
29     if(!glview) {
30         glview = GLViewImpl::createWithRect("HelloWorld", Rect(0, 0, 960, 640));
31         director->setOpenGLView(glview);
32     }
33
34     director->getOpenGLView()->setDesignResolutionSize(960, 640, ResolutionPolicy::SHOW_ALL);
35
36     // turn on display FPS
37     director->setDisplayStats(true);
38
39     // set FPS. the default value is 1.0/60 if you don't call this
40     director->setAnimationInterval(1.0 / 60);
41
42     FileUtils::getInstance()->addSearchPath("res");
43
44     // create a scene. it's an autorelease object
45     auto scene = HelloWorld::createScene();
46
47     // run
48     director->runWithScene(scene);
49
50     return true;
51 }
```

Using the director  
to run the game  
with the scene

# Today's Project: A small game!



# Airhockey-Game

## 3 Classes:

- AppDelegate  
to run the Game
- GameLayer ← Focus on GameLayer  
includes the game function
- GameSprite  
for the movable objects (players + ball)

# Airhockey-Game: GameLayer

Create init () which includes

- 2 player and 1 ball

```
_player1 = GameSprite::gameSpriteWithFile("res/mallet.png");  
_player1->setPosition(Vec2(_screenSize.width * 0.5, _player1->radius() * 2));  
_players.pushBack(_player1);  
this->addChild(_player1);
```

- EventListener

```
auto listener = EventListenerTouchAllAtOnce::create();  
listener->onTouchesBegan = CC_CALLBACK_2(GameLayer::onTouchesBegan, this);  
listener->onTouchesMoved = CC_CALLBACK_2(GameLayer::onTouchesMoved, this);  
listener->onTouchesEnded = CC_CALLBACK_2(GameLayer::onTouchesEnded, this);  
_eventDispatcher->addEventListenerWithSceneGraphPriority(listener, this);
```

# Airhockey-Game: GameLayer (2)

## Create update()

```
void GameLayer::update(float dt) {
    auto ballNextPosition = _ball->getNextPosition();
    auto ballVector = _ball->getVector();
    ballVector *= 0.98f;
    ballNextPosition.x += ballVector.x;
    ballNextPosition.y += ballVector.y;
    float squared_radii = pow(_player1->radius() + _ball->radius(), 2);
    for (auto player : _players) {
        auto playerNextPosition = player->getNextPosition();
        auto playerVector = player->getVector();
        float diffx = ballNextPosition.x - player->getPositionX();
        float diffy = ballNextPosition.y - player->getPositionY();
        float distance1 = pow(diffx, 2) + pow(diffy, 2);
        float distance2 = pow(_ball->getPositionX() - playerNextPosition.x, 2)
            + pow(_ball->getPositionY() - playerNextPosition.y, 2);
        if (distance1 <= squared_radii || distance2 <= squared_radii) {
            float mag_ball = pow(ballVector.x, 2) + pow(ballVector.y, 2);
            float mag_player = pow(playerVector.x, 2) + pow(playerVector.y, 2);
            float force = sqrt(mag_ball + mag_player);
            float angle = atan2(diffy, diffx);
            ballVector.x = force * cos(angle);
```

# Airhockey-Game: GameLayer (3)

## Define Eventlistener

- onTouchBegan
- onTouchMoved
- onTouchEnded

```
void GameLayer::onTouchesBegan(const std::vector<Touch*> &touches, Event* event)
{
    for (auto touch : touches) {
        if (touch != nullptr) {
            auto tap = touch->getLocation();
            for (auto player : _players) {
                if (player->boundingBox().containsPoint(tap)) {
                    player->setTouch(touch);
                }
            }
        }
    }
}
```

# Links

[http://www.cocos2d-x.org/wiki/External\\_Tutorials](http://www.cocos2d-x.org/wiki/External_Tutorials)

<http://wizardfu.com/book/cocos2d-x/tutorial/>

<http://www.gamefromscratch.com/page/Cocos2d-x-CPP-Game-Programming-Tutorial-Series.aspx>