CocosSharp

RYAN DAVIS

QUEENSLAND C# MOBILE DEVELOPERS MEETUP

10 DECEMBER 2014

whoami

- Ryan Davis
- Hobby developer
- Work in ~~Information Management~~ LINQPad all day baby

- NOT a CocosSharp Ninja
  - But I’ve been working with it a bit

to cover

- Overview of CocosSharp
- Walk through the framework
- Demo and code of a (very) basic game

CocosSharp Overview

what is it

*"CocosSharp is an easy to use **library for simple games** using **C#** and **F#**. It is a **.NET port** of the popular **Cocos2D engine**, derived from **the Cocos2D-X** engine via **Cocos2D-XNA.**"* *(github readme)*

- *Library for simple games:*
  - drawing, sprites, animation, transforms, sound, scenes, run loop/scheduling,
  - with extensions – physics library support
- *.NET port*
  - written on top of MonoGame (runs wherever monogame runs!)
  - write your code in C# or F# (yay); shared codebase via PCLs
  - access to the .NET ecosystem (nuget, etc.)

playstation mobile ps4 xbox360 win raspberry pi winphone android ios osx linux

what is it

*"CocosSharp is an easy to use **library for simple games** using **C#** and **F#**. It is a **.NET port** of the popular **Cocos2D engine**, derived from **the Cocos2D-X** engine via **Cocos2D-XNA.**"* *(github readme)*

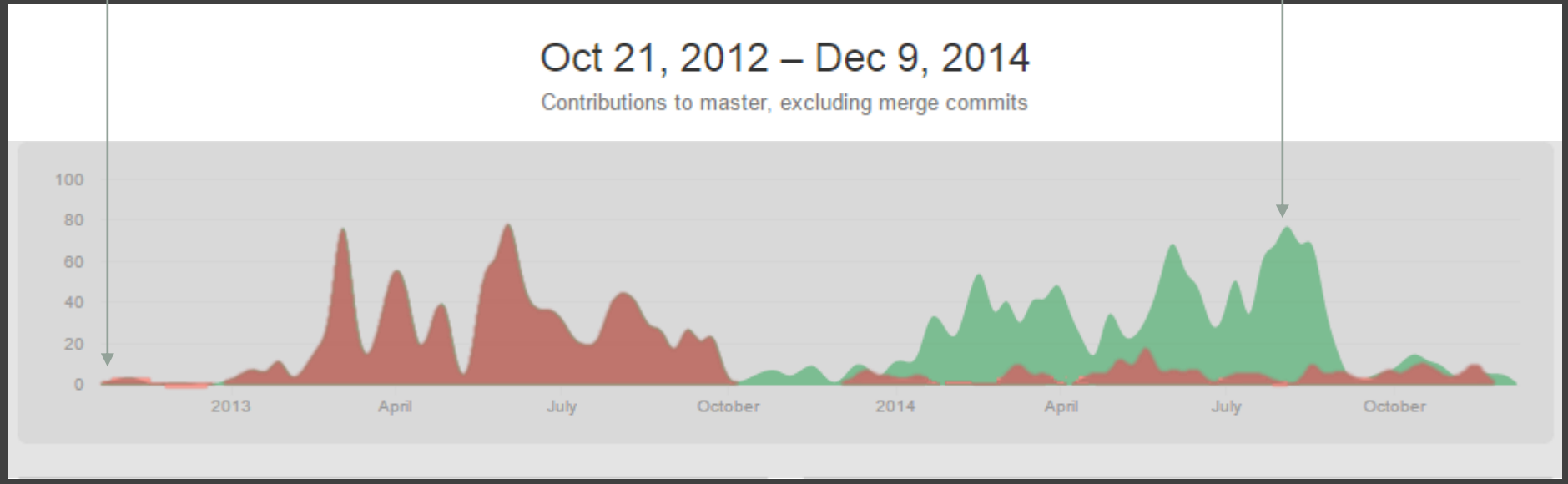
- *Port of the Cocos2d engine (derived from [extended lineage]):*
  - Based on a mature and widely used game framework
  - Many examples of the framework use across the internet and in books
  - CocosSharp is a far descendent of cocos2d, so not all documentation applies

the fork

cocos2d-xna → **cocosharp**

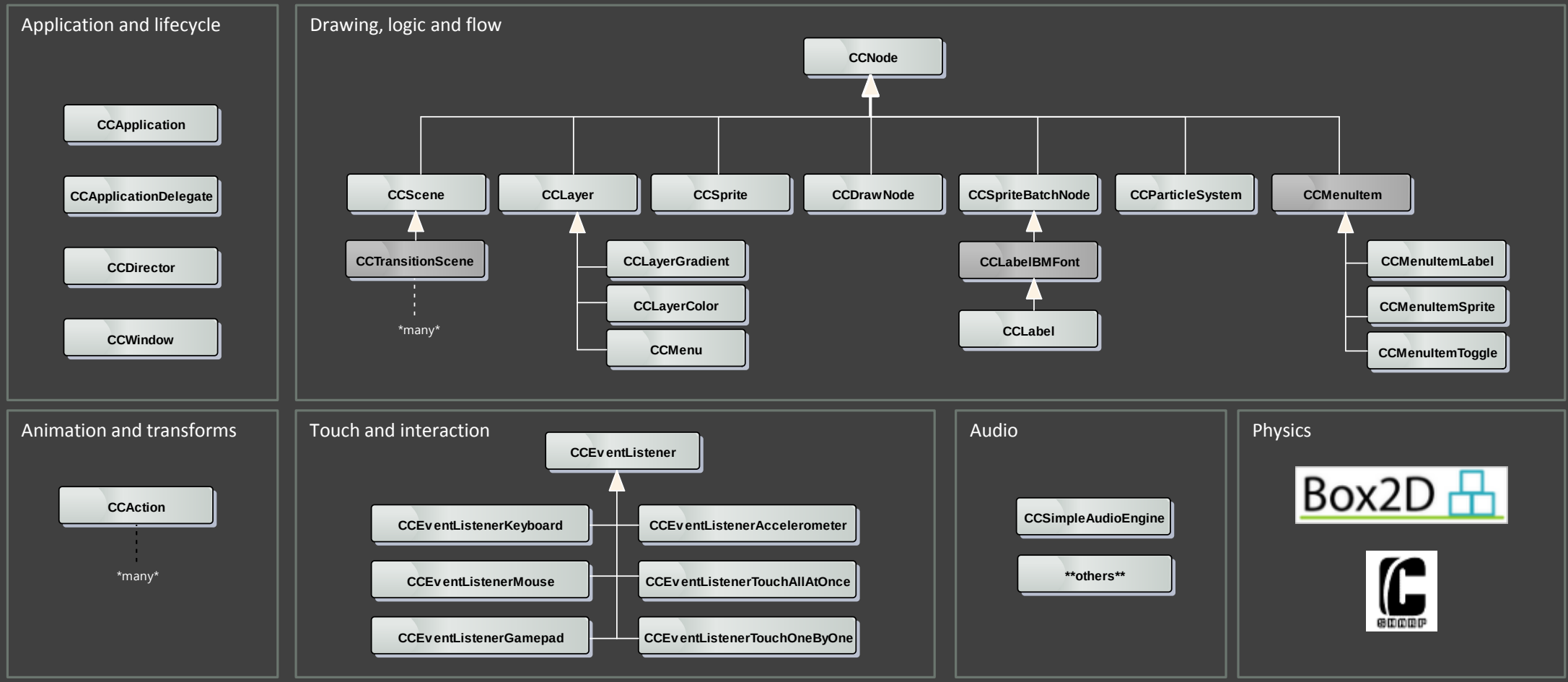
cocos2d-xna
First commit 10/2012

cocosharp
Announced 08/2014



The Framework

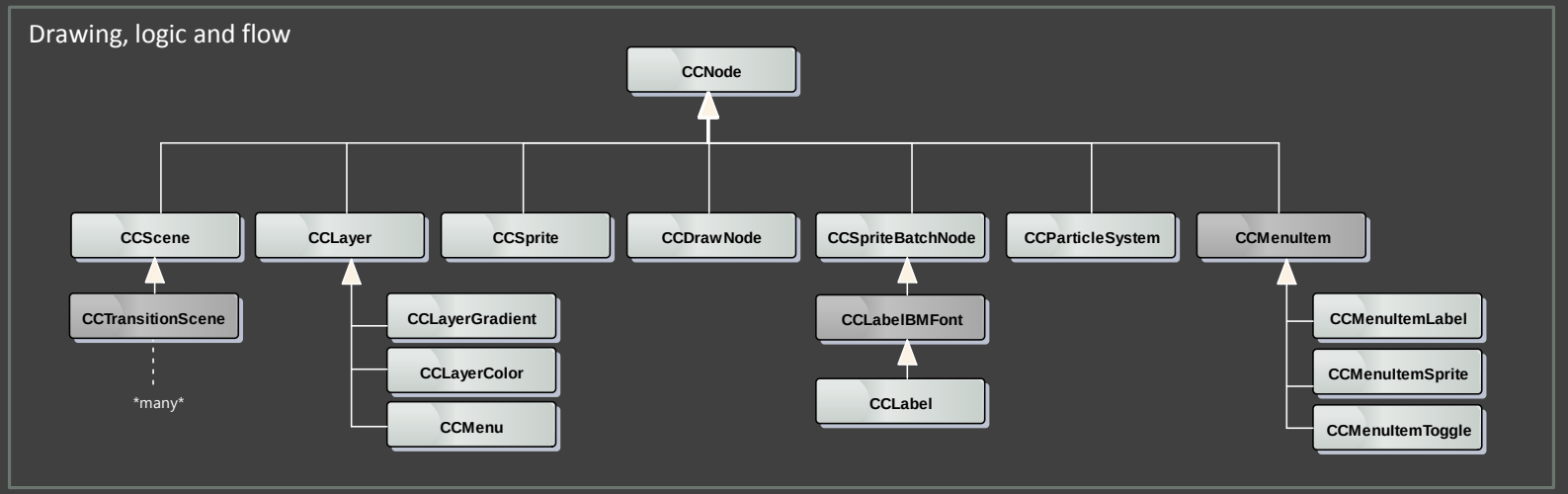
framework class structure (abridged)



application and lifecycle

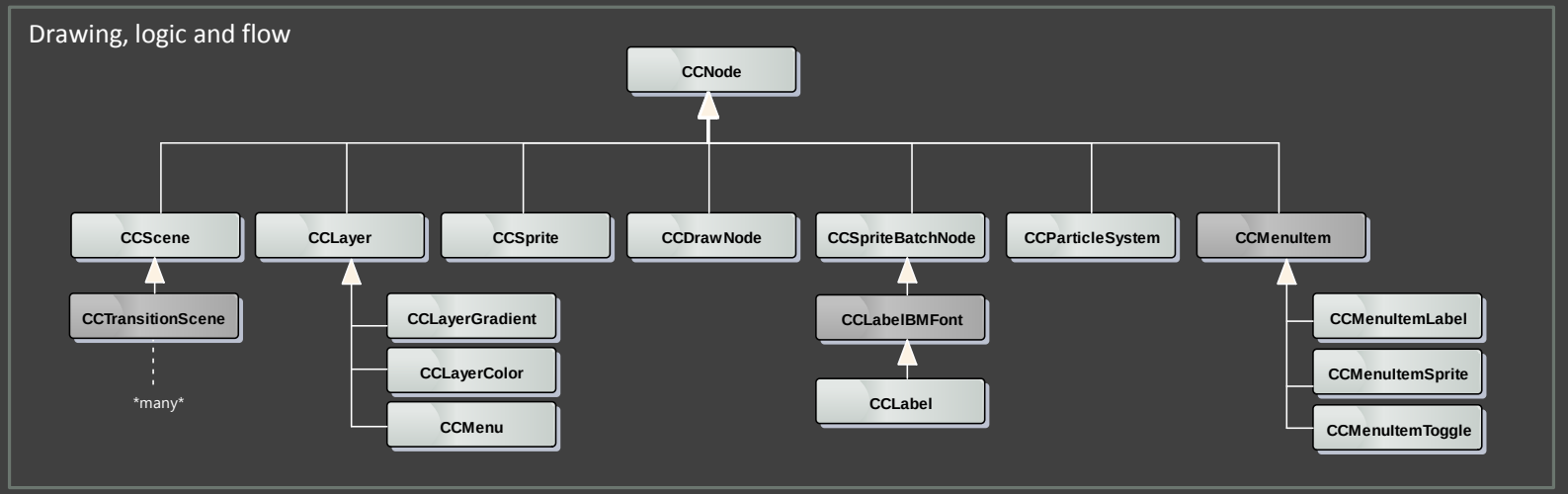
- `CCApplication`
  - Used for initial set up and launch of game. When cross platform, instantiate in native project
  - Set target fps using `AnimationInterval`, content path using `ContentSearchPath`
  - Set desired screen resolution in constructor
- `CCApplicationDelegate`
  - Handle lifecycle events with `DidFinishLaunching()`, `DidEnterBackground()`, `WillEnterForeground()`
  - Must subclass and load initial game scene in `DidFinishLaunching()`
  - Define subclass in PCL, instantiate and pass to `CCApplication` in native code
- `CCDirector`
  - Transition between scenes using `PushScene()` or `ReplaceScene()`
  - Accessed via any `CCNode` using `Director` property
- `CCWindow`
  - No real interaction necessary, but must be passed to constructor of `CCScene`
  - You can set `DisplayStats = true` to see draw/performance stats on some platforms

drawing, logic and flow



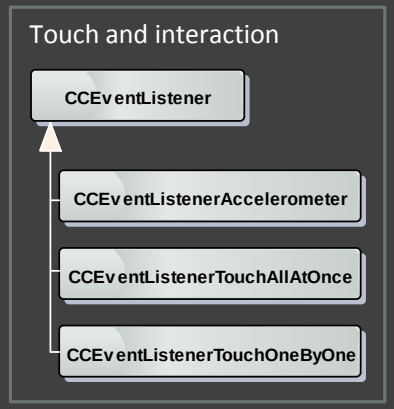
- `CCNode`
  - Base class of most cocos classes, has size, position, child nodes and the ability to run ‘actions’
  - Typically subclassed for custom game objects (player, enemy) or used directly for sprites
  - Child node co-ordinate systems act relative the parent. Check your `AnchorPoint`!
  - `Schedule(Action<float> action)` allows run-loop style processing for logic, etc.
- `CCSprite, CCSpriteBatchNode`
  - Used to load images and animate between images
  - `CCSpriteBatchNode` groups sprite images for performance purposes

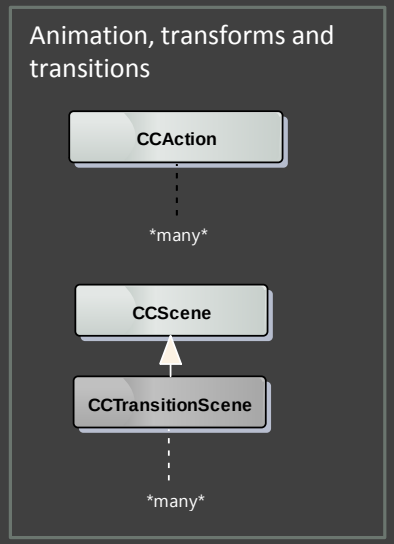
drawing, logic and flow



- `CCScene`
  - Typically used to hold a single 'screen' of your game (intro, menu, main game, game over)
  - Comprised of one or many `CCLayers` with your game content
- `CCLayer/Color/Gradient`
  - Useful for grouping objects in your game (e.g. background, foreground)
- `CCDrawNode`
  - Used to draw primitive shapes such as lines, rects, circles – `DrawCircle()`, `DrawRect()`, `DrawPolygon()`, etc.
  - Current has a bug where `ContentSize` does not update when draw methods are called. Watch out for that
- `CCLabel`
  - Used to display text. Is custom font loading buggy?

touch, actions and transitions

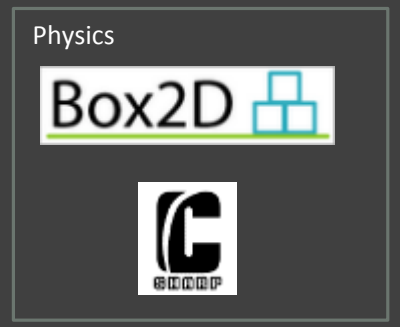




- `CCEventListenerTouchAllAtOnce/CCEventListenerTouchOneByOne`
  - Add a touch event listener to any node to listen for touches
  - `AllAtOnce` gives you all touches at once, `OneByOne` gives you individual touches; you 'swallow' a touch to follow it through the touch lifecycle (began, moved, ended)
  - There are also event listeners for the accelerometer, keyboard, mouse and gamepad
- `CCAction`
  - Superclass of all the 'actions', tweening/declarative definitions of movement or transforms e.g. `CCMoveBy/To`, `CCScaleBy/To`, `CCRotateBy/To`, `CCFadeBy/To`, `CCBezierBy/To`
  - Can be wrapped in 'timing' actions that affect the way the action is interpolated e.g. `CCEaseIn/Out/InOut`
  - Also flow-control actions like `CCSequence`, `CCRepeat/Forever`, `CCDelay`, `CCCallFunc/N`
- `CCTransitionScene`
  - When using `Director.PushScene()` or `Director.ReplaceScene()` you can wrap the new scene in a transition scene to animate the transition between scenes
  - E.g. `CCTransitionFade`, `CCTransitionPageTurn`, `CCTransitionShrinkGrow`

audio, physics

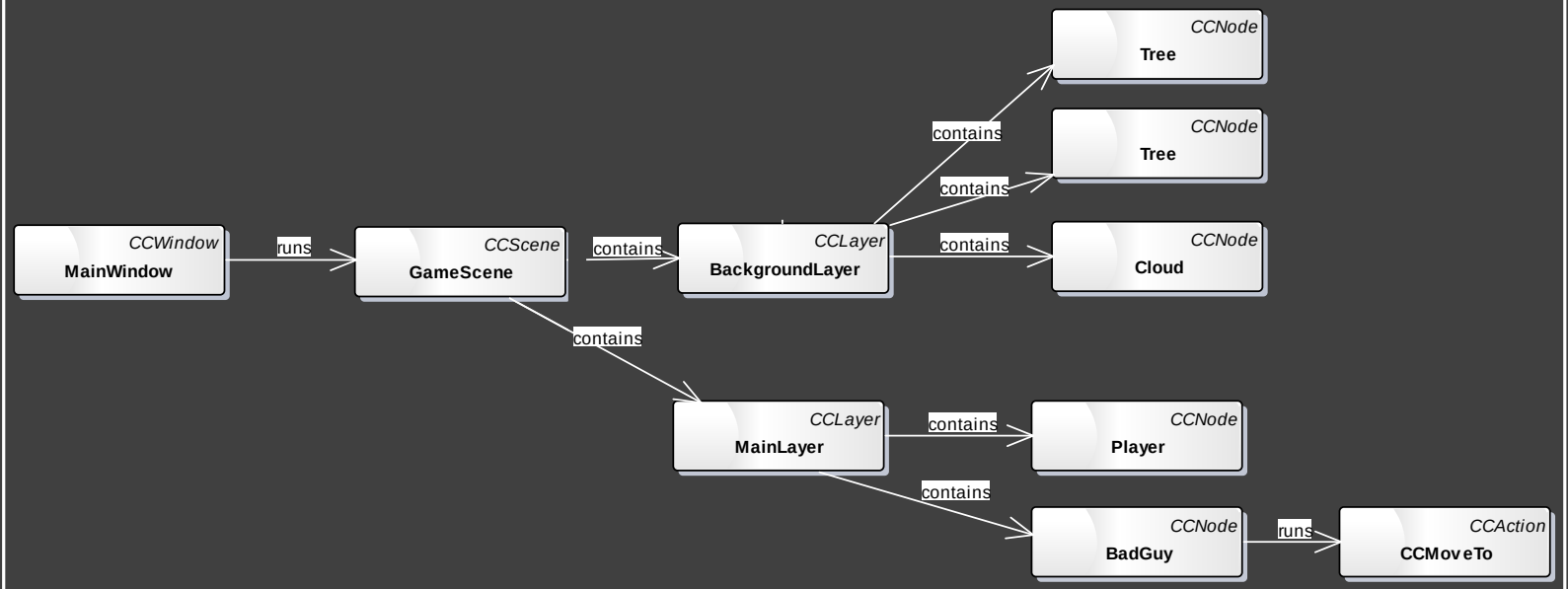
- `CCSimpleAudioEngine`
  - For basic sound needs
  - `CCSimpleAudioEngine.SharedEngine.PlayEffect("soundpath");`
  - Supports .m4a, .aac, .mp3, .wav, .aifc, .caf



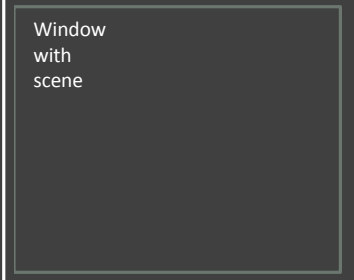
- `box2d, ChipmunkSharp`
  - Ported to work with CocosSharp and there are practical samples out there

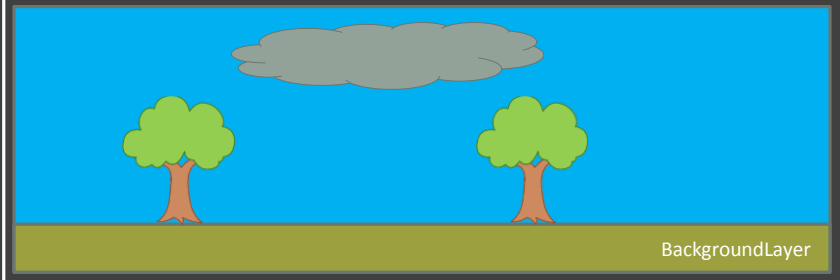
example of scene composition

Instance Tree



Result

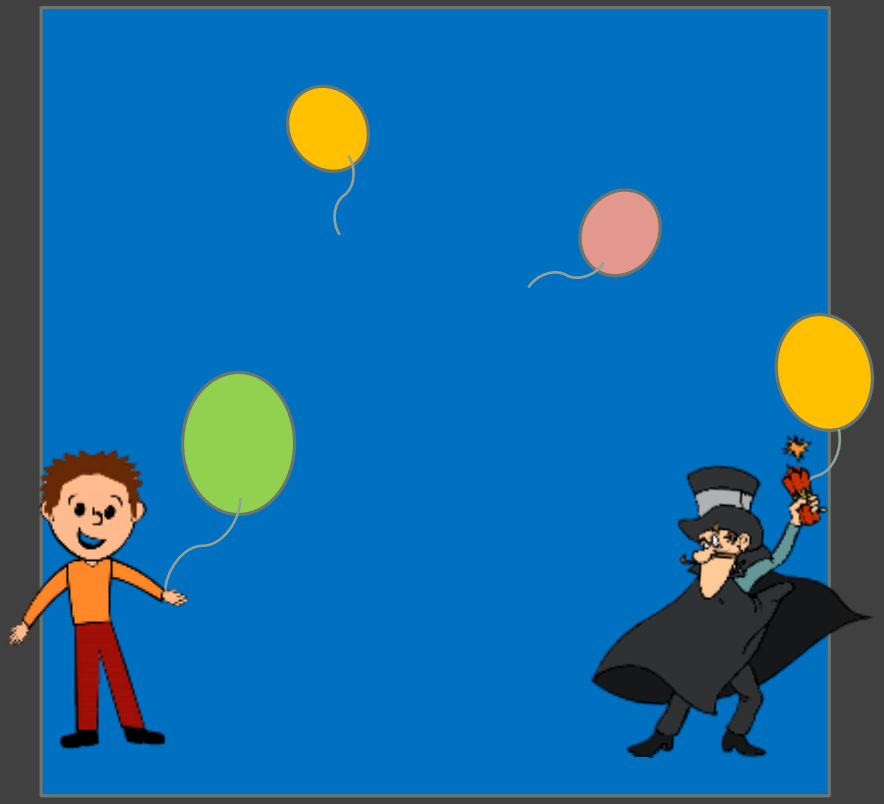




A Basic Game

a basic game

- Balloon Pop
  - App Launch
  - Scenes / Transitions
  - Sprites
  - Actions
  - Labels
  - Touch Interaction

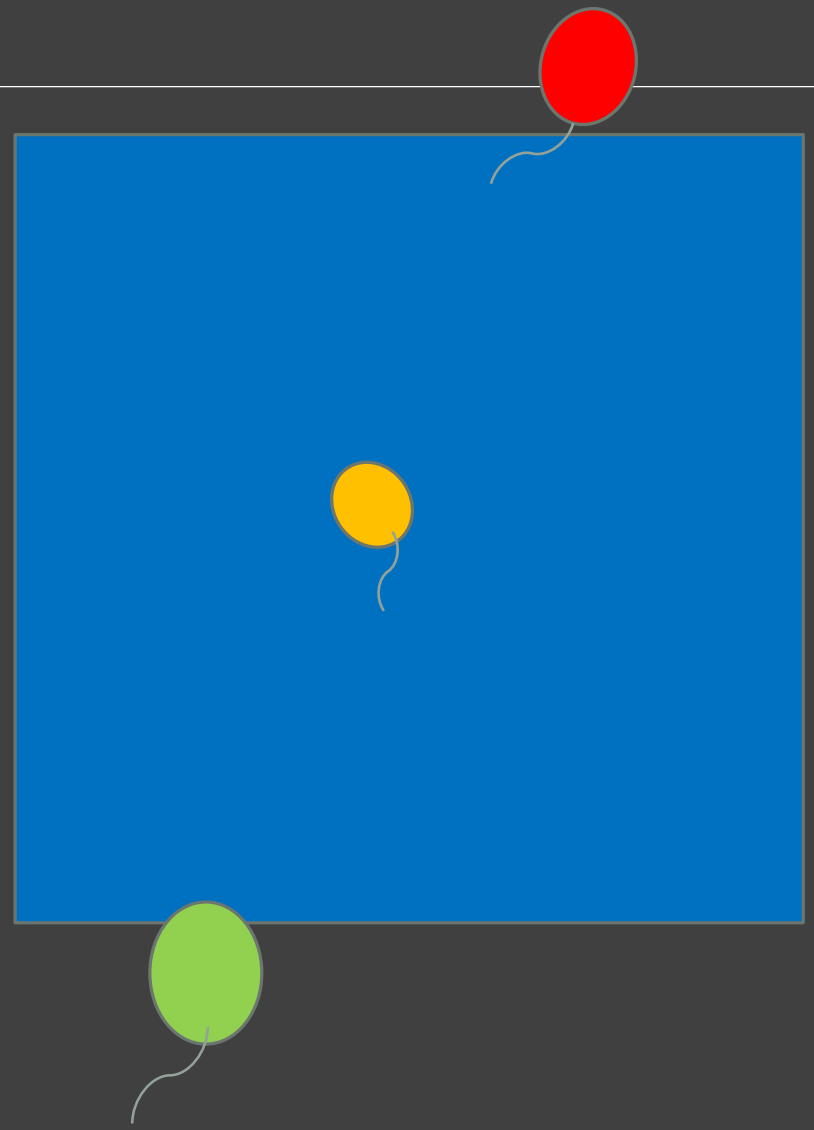


a basic game

- Using CCActions to avoid the need for `Schedule();`

while player has lives:

- randomly choose count of balloons to launch up to a maximum
- generate balloons below bottom of screen
- for each balloon
  - execute a sequence of
    - [animate to top of screen],
    - [subtract one life]
  - if tapped, terminate sequence, increase score and lives, remove balloon
- increase speed of the balloons and (every 5 rounds) increase the maximum number of balloons



resources

- cocosharp repo: https://github.com/mono/CocosSharp
- cocosharp forum: http://forums.xamarin.com/categories/cocossharp
- ray wenderlich blog: http://www.raywenderlich.com/tag/cocos2d (iOS, principles apply)
- Learn cocos2d 2 http://www.apress.com/9781430244165 (ditto)
- cocos2d-x : http://www.cocos2d-x.org/ (C++, principles apply)

Not covered today / things to look into:

- Integrating with physics engines
- Tooling, compatibility with cocos-derived assets and monogame/xna assets – SpriteBuilder, Particle Designer, CocosStudio
- Resource/content loading
- Anything 3D

thanks / questions