

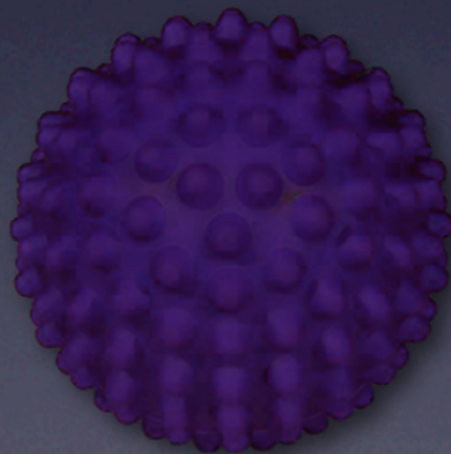
Box2D and OpenGL

Bouncing balls are fun!

Doug Davies, Owner Funky Visions

Box2D and OpenGL

Bouncing balls are fun!



Doug Davies, Owner Funky Visions

Who Am I?

- Writing software since 12 yrs old. My Dad built a PDP-11 that I learned BASIC on and we wire-wrapped my first Apple II (ya... I was deprived)
- Worked for WordPerfect, Comuserve/AOL, Big Lots, and most recently OCLC
- Wrote Apple II shareware games but until the App Store, never had much exposure
- Programmed NeXTStep WordPerfect from 1991-1993 (highlight: meeting Steve Jobs)
- Began writing iPhone Apps in January 2009 after purchasing my first Mac (why did I wait so long?). My first App (Jiggle Balls) hit the App Store in March 2009

Why This Presentation?

- I had written several iPhone games but didn't understand all the boilerplate code
- Wanted to clean up and modularize my code
- Public speaking Newbie! Wanted some experience
- Wanted to share this knowledge since I know how hard it is to get started on a project and get the frameworks setup

What is Box2D?

- Open source 2d Physics Engine
- C++ but also available for Flash, Java, C#, Python, JavaScript, etc. iPhone port by Simon Oliver who wrote Rolando
- Used by MANY games in the App Store either directly or via cocos2d framework
- For many game types, you get your objects on the screen and it does the rest. For example a Labyrinth type game

What is OpenGL?

- Open Graphics Library
- Developed by Silicon Graphics in 1992
- Uniform interface for dealing with different graphics hardware
- State Machine, Low-Level Procedural API (verbose!)
- OpenGL ES subset of OpenGL designed for embedded devices. Run by Khronos Group Inc

Hello World...

Ball Style!

- Wireframe bouncing ball (no physics or textures just yet)
- Using Box2D project layout. Modified main and removed xib dependency so that we can run multiple Demo AppDelegatees
- Added Helper (OpenGL, Box2D, Texture2D) and Demo directories
- In main.m change DEMO define to select which demo to run

Demo #1

(main.m, DemoAppDelegate.mm,
DemoView.mm, OpenGL.mm)



So What Did We Just Do?

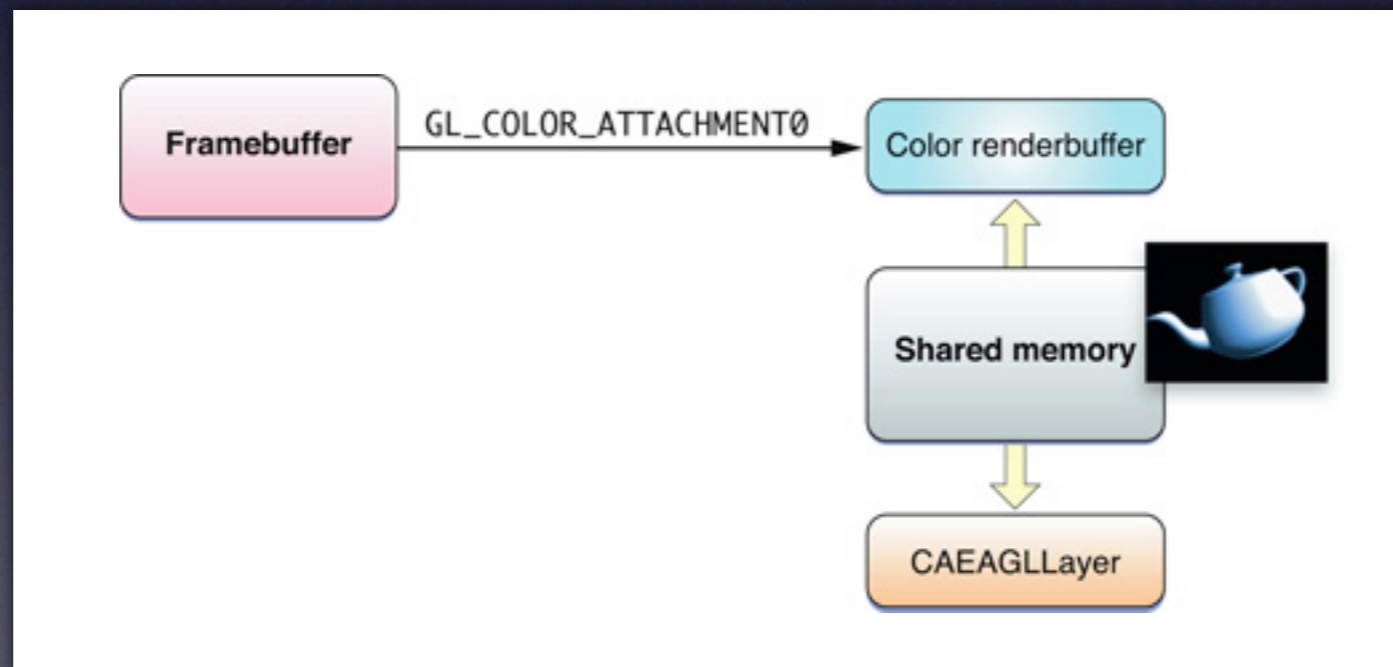
- Created our own UIWindow and UIView
- Created OpenGL context and frame buffer bound to our view. Lots of boilerplate
- Setup a timer to call drawView every 1/60th second
- Rendered into the buffer and asked it to display

Concepts

- Framebuffer - receives drawing commands to produce image and writes that to its color attachment (renderbuffer)
- Renderbuffer - shares data store with UIView and is presented to the screen
- Viewport - the area of the screen we will project our view onto

From Apple's OpenGL ES Programming Guide for iPhone OS

- In order for your application to present OpenGL ES content to the screen, your application needs a UIView object as the target. Further, that view must be backed by a special Core Animation layer, a CAEAGLLayer object. A CAEAGLLayer object is aware of OpenGL ES and references a renderbuffer, as shown. When your application wants to display these results, the contents of this renderbuffer are animated and composited with other Core Animation layers and sent to the screen (although I wouldn't recommend overlaying other UIViews on top of this one for performance reasons)



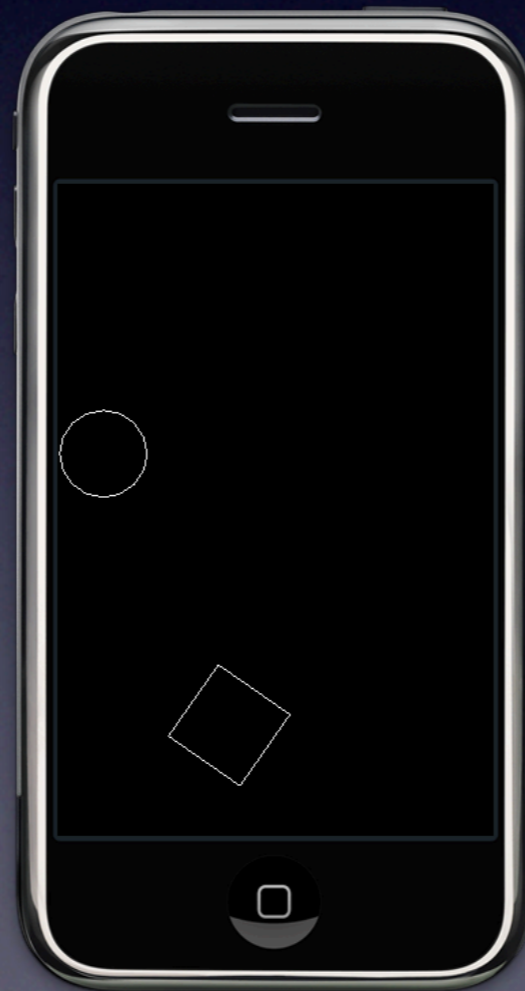
- Orthographic view - project world coordinates onto device coordinates. Why didn't we just use pixels? As you'll see Box2D works in MKS (meters-kilogram-second units) to achieve real world physics
- glBlendFunc - defines how source pixels will be blended with destination pixels already in the framebuffer. In our case source is `GL_SRC_ALPHA` and destination is `GL_ONE_MINUS_SRC_ALPHA` which allows pixels to show through the transparency (alpha channel) border on our shape

Let's Add Some Physics

- Create Box2D world and add bounding boxes
- Create Box2D objects (circle & box)
- Add some properties (bounciness, etc.)
- Drop them and see how they behave

Demo #2

(Demo2View.mm, Box2DHelper.mm)



Concepts

- Body - A chunk of matter that is so strong that the distance between any two bits of matter on the chunk is completely constant. They are hard like a diamond
- Shape - A 2D geometrical object, such as a circle or polygon
- Fixture - A fixture binds a shape to a body and adds material properties such as density, friction, and restitution (bounciness)

- Define `b2BodyDef` - position, angle, velocity
- Create `b2Body` based on that definition
- Create `b2Shape` (convex polygon or circle) - dimensions, vertices
- Create `b2FixtureDef` with `b2Shape` - density, restitution, friction
- Create `b2Fixture` and attach it to `b2Body` using its factory method

- Setup world gravity (slight x vector to make things interesting). Can be tied to accelerometer
- World step - moves the simulation along one iteration. Usually called from drawView but could called independently with another timer (protect with mutex)
- OpenGL functions glTranslatef and glRotatef applied to current matrix and affects subsequent drawing (use glPushMatrix and glPopMatrix to restore)

Bye Bye Wireframes.... Hello Textures

- Use Apple Texture2D Class from sample code (CrashLanding) to draw bitmaps instead of wireframes
- Draw background into framebuffer and then add ball and box bitmaps

Demo #3

(background.jpg, box.png, ball.png,
Demo3View.mm, Texture2D.m)



Concepts

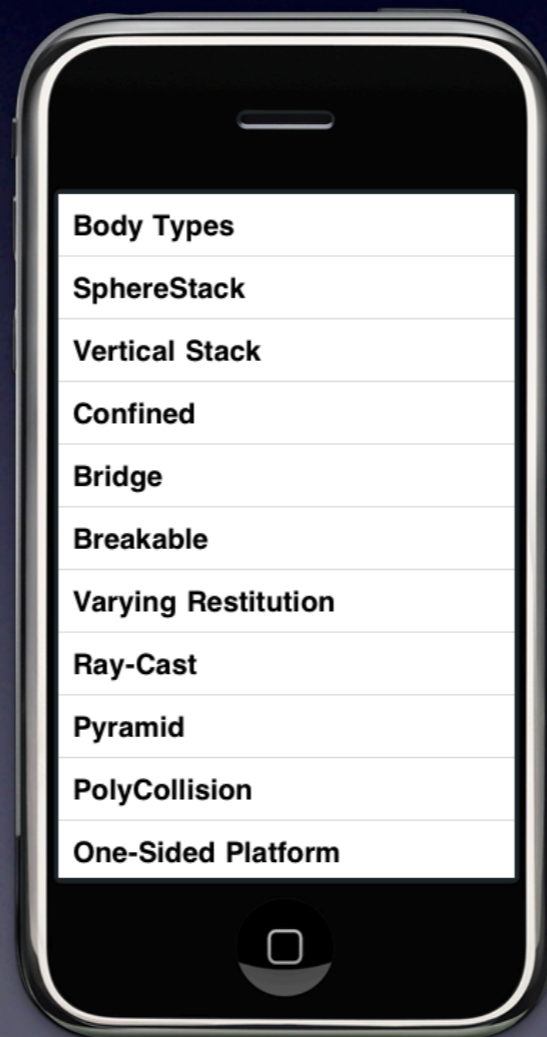
- Need to enable `GL_TEXTURE_2D` and `GL_TEXTURE_COORD_ARRAY` in OpenGL
- Use `Texture2D` alloc init to load jpegs or pngs with alpha channel. Makes sure dimensions are a power of 2.
- Since we are using ortho projection we have to fit bitmap inside rectangle (using `drawInRect`) OR set ortho projection to pixels and translate coordinates

Play Around

- Change world size
- Change gravity
- Change restitution (bounce)
- Change alpha on drawInRect

Box2D Testbed

Demo #0



Links

- www.funkyvisions.com/presentations/box2dopengl
- www.box2d.org
- www.khronos.org
- cocos2d-iphone.org

More Links

- 7Isquared.com - Michael Daley iPhone Game Programming Tutorials
- Working with OpenGL ES Contexts and Framebuffers - http://developer.apple.com/iphone/library/documentation/3DDrawing/Conceptual/OpenGLES_ProgrammingGuide/WorkingwithEAGLContexts/WorkingwithEAGLContexts.html

Questions???



Funky Visions

Look for Jiggle Balls Studio coming soon to an iPad near you!