Extracted from:

# **Build iOS Games with Sprite Kit**

#### Unleash Your Imagination in Two Dimensions

This PDF file contains pages extracted from *Build iOS Games with Sprite Kit*, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit http://www.pragprog.com.

Note: This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printed versions; the content is otherwise identical.

Copyright © 2014 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

The Pragmatic Bookshelf

Dallas, Texas • Raleigh, North Carolina

# Build iOS Games with Sprite Kit

Unleash Your Imagination in Two Dimensions



# Jonathan Penn and Josh Smith

Edited by Rebecca Gulick

# Build iOS Games with Sprite Kit

Unleash Your Imagination in Two Dimensions

Jonathan Penn Josh Smith

The Pragmatic Bookshelf

Dallas, Texas • Raleigh, North Carolina



Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and The Pragmatic Programmers, LLC was aware of a trademark claim, the designations have been printed in initial capital letters or in all capitals. The Pragmatic Starter Kit, The Pragmatic Programmer, Pragmatic Programming, Pragmatic Bookshelf, PragProg and the linking *g* device are trademarks of The Pragmatic Programmers, LLC.

Every precaution was taken in the preparation of this book. However, the publisher assumes no responsibility for errors or omissions, or for damages that may result from the use of information (including program listings) contained herein.

Our Pragmatic courses, workshops, and other products can help you and your team create better software and have more fun. For more information, as well as the latest Pragmatic titles, please visit us at <a href="http://pragprog.com">http://pragprog.com</a>.

The team that produced this book includes:

Rebecca Gulick (editor) Potomac Indexing, LLC (indexer) Cathleen Small (copyeditor) David J Kelly (typesetter) Janet Furlow (producer) Ellie Callahan (support)

For international rights, please contact rights@pragprog.com.

Copyright © 2014 The Pragmatic Programmers, LLC. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

Printed in the United States of America. ISBN-13: 978-1-94122-210-2 Encoded using the finest acid-free high-entropy binary digits. Book version: P1.0—July 2014

# Preface

Imagine going back in time to visit the people who wrote for the original Atari 2600 game console and showing them games on an iPhone. Jaws would drop. Minds would be blown. They'd probably check for smoke and mirrors.

We've come a long way from the video game industry's humble beginnings. Writing games was a challenge back then. It still is today, of course, but the challenges then involved shoving individual pixels around, saving CPU cycles for rudimentary sounds, and interpreting raw player input from analog joysticks. Today, our challenges are often bounded more by our imaginations than by technical constraints.

And that's why we think you've joined us here in this book. You have an unprecedented amount of power in a computer resting in the palm of your hand. You want to write a game, and you'd like to do it for iOS. We have good news for you.

Welcome to Sprite Kit! Apple's exciting 2D-game development engine sports an excellent API to help bring your 2D game idea from paper to pixels. If you're already an iOS developer, then there's nothing else you need to do. It comes with excellent Xcode support and gives you a template ready to get started. It doesn't get any easier than this.

Sprite Kit provides the scaffolding for you to organize your game code, animate objects on the screen, play sound effects, handle touch events, simulate physical movements and collisions, and more. Any game that functions in two dimensions, such as platformers, puzzles, or overhead action games, will work great with Sprite Kit's tools.

This book will help you learn enough to take your own 2D game idea and implement it with Sprite Kit's building blocks.

### How Do We Get There?

The best way to learn Sprite Kit is to build a game...or two! In this book, we'll walk through all the steps to build two actual games (that are quite fun, in the authors' not-so-humble opinions). We have chosen these games because they provide an opportunity to learn the way of the Sprite Kit APIs step by step.

Let's get to know these games.

#### Space Run

This will be an infinite runner game, like *Canabalt* but in space. The goal is just to stay alive as long as possible and rack up points. It's a single-finger game, which makes it a great fit for the casual game market. Check out the sketches in the following figure:



Figure 1—Paper prototype of Space Run

As the player, you are on a mission to race through light-years of space to rescue a distant science team that is in trouble. But this is no vacation cruise! You have to dodge things that will destroy your fragile ship (asteroids and enemy ships), and you can go on the offensive with your photon torpedoes when running isn't enough.

Here are the features we want to achieve:

- *Obstacles* We want simple asteroids that just float aimlessly along a straight line, and we want enemy ships that spin and turn along a path to make it harder to avoid them.
- *Weapons* The ship should shoot a photon torpedo at regular intervals. Any obstacle can be destroyed when hit.
- *Power-ups* We want to give players something they can collect that makes their weapon shoot faster for a certain amount of time.
- *Variable difficulty* We want to let players pick Easy mode or Hard mode, which determines the frequency of obstacles that appear on the screen.
- *Scoring* We want to keep track of and show the player's score. Forward progress is difficult in the game, so the points awarded for each obstacle destroyed increase as a multiple of the elapsed time. Also, Hard mode doubles the point values.
- *Special effects* What space game would be any fun without explosions? We need 'em—lots of 'em. We also need a thrilling deep-space star field zooming past to give the illusion of hyper-speed. The game should be a visual extravaganza of light and color.
- *Single-finger control* We want this to be a casual game that's easy to pick up and play and doesn't require a lot of commitment to learn. The ship will follow your finger as you move, and the cannon will fire continuously as long as your finger touches the screen.

*Space Run* is perfect to start with because you can jump right in and practice moving a ship image around on the screen by handling touch events. You'll riff on the idea and add new features as you learn about them in Sprite Kit's toolbox.

#### **Physics Ball**

Classic pinball at its finest! We're going to build a simple pinball game with all the fun and physics of the real thing. It will be an excellent casual game full of sound effects and will automatically scroll taller than the screen. Check out the sketch in the following figure:



Figure 2—Paper prototype of Physics Ball

Here are the goals we want to achieve:

- *Physics* This needs to feel like a real pinball table, with gravity, friction, ricochets, and spin.
- *Sound* As the ball bounces around, we need to play sound effects. Lots of them. To protect the player from auditory boredom, we'll randomly pick from different sounds for each hit.
- *Bonus scoring* If the ball flies past a special spinner, then that activates bonus score mode, and all scores are increased by a large factor. This bonus mode should be in effect as long as the spinner is in motion.
- *Camera panning* The screen real estate on even a four-inch iPhone is kind of small.
- *Special effects* We want to use little puffs and sparks whenever the ball hits targets or bumpers. All for the visual delight of the player!
- *Two-finger control* A pinball wizard can play by sense of smell. For mortals the game requires two fingers. Tap on the left side of the screen to flip the left paddle. Tap on the right side for the right paddle.

The mechanics of pinball are well known, so this type of game will be a wonderful introduction to the Sprite Kit physics engine. We'll need to figure out how to handle collisions, define the shapes and boundaries, and control the physical properties of the ball in real time. We'll even make the playing field taller than the screen and add some "impossible" physics into the mix to make it more interesting. This will be much easier to implement once you have the basics of Sprite Kit's APIs under your belt. You can jump ahead and dive right into these chapters if you want, but don't worry if you feel overwhelmed. This game builds on the knowledge from the earlier chapters. Take your time and enjoy the journey.

### The Road Ahead

Reading this book is kind of like playing a game, too. You're the player. Your goal is to learn about Sprite Kit and have fun along the way. Each of these chapters is like a level, and each one has a challenge to implement pieces of the game as we've sketched it out. Here's an overview of the progress you'll make:

- Chapter 1, *Introduction to Sprite Kit*, on page ?, is our intro level—an easy one meant to introduce you to the Sprite Kit template that comes with Xcode and the simplest way to interact with a spaceship node on the screen.
- Chapter 2, Actions: Go, Sprite, Gol, on page ?, is the next level, where we play with more complexity. In this chapter you'll get to know Sprite Kit's actions, how to apply them to nodes, how to chain them together, and how to use them to help simplify the control of the spaceship and other characters on the screen.
- Chapter 3, *Explosions and Particle Effects*, on page ?, starts giving our *Space Run* game some sparkle and panache. We've got the ship, asteroids, and photon torpedoes flying around on the screen, but we want explosions to happen when they collide. We also want a thrust effect out of the back of the ship. Through all this, you'll learn quite a bit about the built-in particle editor.
- Chapter 4, *Menus and Cutscenes*, on page ?, is where we'll start stitching the *Space Run* game together. You'll learn more about Sprite Kit scenes, how they interact with UIKit, how to transition, and how to make an opening scene for your game.
- Chapter 5, *Keeping Score with a Heads-Up Display*, on page ?, adds some more visual feedback of the player's current progress through a heads-up display. We'll talk about laying out nodes where you want them on the scene and updating the game state throughout play. By the time you reach this chapter, you'll have a fully functioning *Space Run* game!

- Chapter 6, *Pinball Physics 101*, on page ?, is where we'll start building our pinball game. We'll start playing around with physics bodies in a scene to understand how best to model the pinball mechanics.
- Chapter 7, *More Physics: Paddles and Collisions*, on page ?, builds on the knowledge about the Sprite Kit physics engine and talks about collision categories, complex bodies and edges, and more to complete the essence of the pinball game.
- Chapter 8, *Polishing the Pinball Game*, on page ?, takes us deeper into Sprite Kit to polish up the pinball game. We'll build a bonus spinner target, frame-based animations to cue when the user should pull the plunger, and overlay table graphics, and we'll clean up some of the rough edges!
- Chapter 9, *Where to Go Next*, on page ?, brings the book to a close, reflecting on the games we created, the things you learned about Sprite Kit, and resources to go further in game development.

### How to Get the Most out of This Book

Code is broken down by chapter and split up into different steps where it makes sense to take note of the code at that point. For the most part, you should be able to follow along and create all the pieces yourself on the fly. But if you want to double-check your work with the final product for that step or if you want to pick up in the middle, just find the appropriate code directory and start from there.

You can download the code from the book website.<sup>1</sup> Each code snippet mentioned in the book shows the path to the file where it came from. That will show you the chapter and step where you can catch up. If you are using an ebook format, then you can click or tap on the path of the file above the snippet to jump straight to the file hosted on the Pragmatic Programmers website. That makes it easy to cut and paste if you want to.

The book builds in *cognitive complexity*, meaning that the tasks you perform at the start will be very simple—just enough to get you started. It might feel rote at first, but that's because we don't want you to get lost in the complex possibilities that Sprite Kit provides later on. Each chapter assumes you've achieved the goals of the prior one.

If you think about, it's the same kind of progression that great games lead a player through. You don't know how to defeat the final boss when you first

<sup>1.</sup> http://pragprog.com/titles/pssprite/source\_code

sit down to learn the rules. You need to feel the basic mechanics of the game, the way the other characters interact, and the boundaries of what you can do. As each step builds on the previous one, you'll discover how much you've learned when you look back at the beginning.

This is why we think it's best to work your way through the book in one straight go. But should you want to skip around (and we certainly understand the curiosity and excitement behind that if you do), then you can use the code checkpoints at different chapters and steps to catch up to where the book is at.

## **Expectations and Technical Requirements**

This book assumes that you are at least somewhat familiar with the basics behind iOS development and Xcode. We recommend keeping these references handy as prerequisite reading:

- "Start Developing iOS Applications Today,"<sup>2</sup> an excellent starting place for Apple's official documentation
- *iOS SDK Development [AD12]*, by Chris Adamson and Bill Dudney
- Storyboards [Ste14], by Daniel Steinberg

You should at least be familiar with Apple's introductory material, know about how view controllers and memory management work, and know how to build and run an application in the Xcode GUI. We'll be working with at least Xcode 5.1 and iOS 7.0.

# Acknowledgments

We're so thankful for everyone who supported us while we experimented with the material in this book. To the CocoaConf team for the opportunity to run our one-day game workshop, again and again. To our workshop attendees, who gave us such great feedback. To Daniel Steinberg for all those deep lunchtime discussions when our paths crossed. To the Pragmatic Programmers for the opportunity to put our thoughts into this format. To our editor, Rebecca Gulick, for her patience and guidance. And to our families for putting up with the delirious antics of creatives under deadlines.

You've all impacted us. We hope we can do the same in return.

We also want to thank the technical reviewers for their work to test the narrative and code in this book: Janie Clayton-Hasz, James Dempsey, Mike

<sup>2.</sup> https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/FirstTutorial.html

Enriquez, Ash Furrow, Brian Hogan, Jeff Holland, John Moses, Kevin Munc, Zak Nixon, Romain Pouclet, Cesare Rocchi, Kim Shrier, Daniel Steinberg, T.J. Usiyan, and Miles Wright.

And now, let the games begin!

So, are you ready, player one? Shall we build a game?