Mostly Codeless Game Development

New School Game Engines

Robert Ciesla

Apress®

Mostly Codeless Game Development

New School Game Engines

Robert Ciesla

Apress®

Mostly Codeless Game Development: New School Game Engines

Robert Ciesla
Helsinki, Finland

ISBN-13 (pbk): 978-1-4842-2969-9 ISBN-13 (electronic): 978-1-4842-2970-5
DOI 10.1007/978-1-4842-2970-5

Library of Congress Control Number: 2017948735

Cover image designed by Freepik

Managing Director: Welmoed Spahr
Editorial Director: Todd Green
Acquisitions Editor: Pramila Balan
Development Editor: Matthew Moodie
Technical Reviewer: Nakul Verma
Coordinating Editor: Prachi Mehta
Copy Editor: Kim Wimpsett
Compositor: SPi Global
Indexer: SPi Global
Artist: SPi Global

This book is dedicated to Dennis Ritchie (1941–2011),
the creator of the C programming language

and

my friend Tuomas Mäkelä.

Contents at a Glance

Contents

About the Author

Robert Ciesla is a freelance writer from Helsinki, Finland. He earned a bachelor of arts degree in journalism and has a knack for writing urban fiction and directing short films. Robert has worked on many video games on several platforms since being a kid in the mid-1990s. His latest venture is Soiree Games, a burgeoning games company specializing in products with a socially aware slant. Robert's personal web site is at `robertciesla.com`.

About the Technical Reviewer

Nakul Verma is a professional game developer and currently works as a senior unity developer at Aquimo Sports Pvt Ltd. He has worked in a variety of game genres using multiple technologies. Specifically, he has worked on casual puzzle games, an endless runner, an endless casual game, card games (rummy on Cocos2d-JS and an African game), and a physics simulation sample and is currently working independently on his own game that will be hitting stores soon. He is proficient in game technologies such as Unity, Cocos2d-x/JS, Construct, and Allegro. Gaming has always been one of his favorite hobbies along with sports, music, and break dancing. His favorite game genres are first-person shooters, platformers, and puzzlers. When he is not making or playing games, he is working out, break dancing, or messing around with some gadget.

He earned a bachelor of technology degree from PEC University of Technology in the field of electronics and electrical communication.

Acknowledgments

I'd like to thank the entire Apress editorial team for their support and constructive criticism. Their input greatly helped shape my vision for this book for the better. As Robocop once so eloquently put it, thank you for your cooperation.

Introduction

Why would anyone get into the video game industry? I'll give you two pretty good reasons.

- As of 2017, the global games market was estimated to be worth more than $100 billion.
- It won't stop at $100 billion.

From the rectangles on the TV that used to excite people in the 1970s to the painstakingly drawn pixel art of the 1980s and the 1990s to the 3D revolution of the 2000s, video games have been increasingly influencing the aesthetic enterprises around them. Now, in 2017, we have truly reached the Golden Age of the Video Game. The line between cutting edge and retro has never been this blurry. It's hip again to be pixelated. Unlike those early millennial times, a gigantic software team is no longer a must-have prerequisite for success. Thanks to some new tools and digital delivery systems, the one-person operation is back.

Not only do we have the technical know-how to produce nearly movie-quality game experiences, but some very powerful pieces of game making–software are becoming both numerous and widely available. In addition to having top-notch hardware support, this type of software is finally becoming accessible to all. There are tools for every budget and every skill level. And the end results can be more impressive than the audiovisuals in Hollywood. Finally, after 40 years or so, making game creation software is a truly viable option for investors and programmers alike. This is where we, the small-time entrepreneurs, cash in. This is it.

But as great as all of these tools are, this book is not about productivity software per se; it's more about every future game visionary out there. This book is about you, my friend, and together we will take *your* ideas from *your* consciousness to all those little screens in the world. Video games are no longer a mere industry. They are a culture, and you're part of it. Now more than ever.

—Robert Ciesla
CEO, Soiree Games

CHAPTER 1

Getting Ready!

Before you felt the urge to create your own games, you probably were a consumer of games for quite some time. You've played a lot of them over the years, and you have intuitive ideas of what appeals to gamers like you.

Prior to getting serious about game development, you should consciously seek factors that make a game great. The biggest overall successes in entertainment software history all share common traits. To a degree, you are wise to emulate them. Let's take look at these titles, shall we?

- *Tetris for mobile devices by Alexey Pajitnov & EA*: More than 100 million copies sold. It was originally released in 1984 for the Electronika 60, a then-hip computer from the Soviet Union.
- *Wii Sports by Nintendo*: More than 82 million copies sold.
- *Minecraft by Mojang*: More than 70 million copies sold.
- *Grand Theft Auto V (GTA V) by Rockstar Games*: More than 52 million copies sold.
- *Super Mario Bros by Nintendo*: More than 40 million copies sold.

Now let's take a look at the indie contestants.

- *Minecraft by Mojang*: More than 70 million copies sold
- *Super Meat Boy by Team Meat*: More than 40 million copies sold
- *Fez by Polytron Corporation*: More than 40 million copies sold
- *World of Goo by 2D Boy*: More than 40 million copies sold
- *Bastion by Supergiant Games*: More than 3 million copies sold

What do all of these titles have in common? They're multiplatform. In some cases, they're very, very multiplatform. One of them, *Wii Sports*, came bundled with a cool new system—see if you can get on that bandwagon! These ten games are intuitive to grasp and control, offering a smooth gaming experience. There are also memorable main characters with highly merchandisable gimmicks—Italian plumbers, anyone? And there's a lot of

R. Ciesla, *Mostly Codeless Game Development*, DOI 10.1007/978-1-4842-2970-5_1

violence, in the case of *GTA V* (and many hit games that didn't quite make the list), which is a gangbanger simulation of the highest caliber. For some reason, the people of this planet really, really enjoy their extreme violence.

So, the following features are what sells:

- *Flawless game mechanics*: The player doesn't have to struggle with controls or grasping the idea of the franchise. The basics are simple, and they work at all times.
- *Lasting challenge*: The game is virtually unbeatable (like *Tetris* and *Wii Sports*) and/or provides tons of replayability.
- *Deployment for multiple platforms*: Keep those Windows-only games to a minimum.
- *Memorable, merchandisable characters*.
- *Conflict and violence*. In general, this planet loves it. In games, it certainly helps.

Now, as a small developer, it's likely you will need to wear many hats. Let's take a look at these roles in the video game industry next. Working with modern video game–making software, it's unlikely you will need to ever dwell very deep into more complex areas of programming, but it's still a good idea to get acquainted with some common industry job titles.

Who Does What in the Video Game Industry

Many of the following development team roles are increasingly becoming specialized as video games rival blockbuster movies in their armies of creative people working on them.

Producer

Responsible for keeping the whole project together, a video game producer benefits from both hands-on experience in as many related fields as possible and a sense of overall vision. The number of a producer's creative responsibilities varies within development teams. In some cases, a producer solely works as management, solving conflicts and keeping a team going.

This may be an unnecessary post in smaller projects, however. A tiny operation obviously doesn't benefit much from a hired producer.

Some typical producer duties include the following:

- Building and maintaining a functional team
- Contracting out work and delegating responsibilities
- Media relations

Designer

Video game designers come up with the conceptual part of a product. Good game design is timeless. Think of chess: it was designed in the sixth century and is still going strong. Creating balanced game dynamics and a low enough learning curve are the designer's job. Also, as sprawling 3D games are all the rage, level designers are very much in demand. Modern level designers usually work with dedicated software, sometimes provided by the programmers in the team.

Programmer

Programmers handle perhaps the most diverse bunch of duties within game production. There are numerous specialized programming fields needed in creating a competitive product. In the early days of the 1980s, being a programmer meant you were the sole person behind a title. Not so much in modern times, although there are exceptions (*Minecraft*, for one, is a one-person operation). Being a programmer can mean these things and much more.

- *Game core creator*: This is what people usually mean by programmer. The game core creator is responsible for game mechanics, main visuals, and player controls.
- *Artificial intelligence developer*: This person is responsible for making smart enemies within a game.
- *Problem solver*: If you're really good, you may be hired as a mercenary programmer to solve a development team's issues within a project.
- *Physics expert*: This person is responsible for creating realistic maps/levels for games with a set of artificial laws of physics governing the game world.
- *Networking specialist*: Many games are run online these days as multiplayer war zones. This creates a whole host of challenges to a project.

Usually, being a good programmer requires a strong sense of logic and/or mathematics. The importance of math is somewhat exaggerated in most programming literature, but it always helps. Some fields of programming work, such as physics, simply do require strong math skills.

Many programmers have a "pet language," which is one they are most comfortable with. Make sure yours is one of the more useful ones, such as C++ or Java.

Visual Artist

Video games used to mostly feature simple on-screen shapes for visuals. Since the advent of 3D graphics in the mid-to-late 1980s, visual artists span an increasingly large group of subfields. These include the following:

- *2D artist*: This includes duties such as presentation and, in the case of 2D games, in-game visuals.
- *3D artist/3D modeler*: These artists create 3D objects with software such as Blender or Maya.
- *3D animator*: This may in many cases be the job of the modeler also. An animator works with the 3D objects created by the modeler and crafts fetching animated sequences, such as a 3D human walking, running, or fighting.
- *Texture artist*: In essence, 3D objects need a coat of digital paint on them to make them look less bland and more realistic, which is the duty of the texture artist.
- *Environmental artist*: Most 3D games need compelling vistas to make them draw the players in.
- *Conceptual artist*: Especially bigger projects benefit from unified art direction. Concept drawings on whatever media help with this goal.

Sound Designer/Musician

If you are an experienced musician, you can in theory make it as a video game composer. Earlier on, as in the 1980s, computer musicians were required to create their compositions pretty much using programming skills. As of late, as long as the output is in digital form, your audio work can be quickly incorporated into a video game project.

Sound designers may or may not also be musicians. What they need is the ability to create audio usable in a video game context, meaning mostly sound effects and atmospheres.

Tester

Testing is a very important phase in a video game's life. If you are a one-person developer, you should put considerable effort into testing your products thoroughly before release.

There are roughly two stages of game testing: alpha (in-house) and beta. Beta testing refers to the public at large volunteering to spot issues in your game. Beta testing can be either by invitation only or in public.