Unity 2022 by Example : A Project-Based Guide to Building 2D and 3D Games, Enhanced for AR, VR, and MR Experiences /

Cameron, Scott H.,
author

Monografía

Start building commercial and playable games such as 2D collection and adventure games, 3D FPS game in Unity with C#, and add AR/VR/MR experiences to them with this illustrated guide Key Features Create game apps, including a 2D adventure game, a 3D first-person shooter, and more Get up to speed with Unity Gaming Services available for creating commercially viable games Follow steps for publishing, marketing, and maintaining your games effectively Purchase of the print or Kindle book includes a free PDF eBook Book Description Unity 2022 by Example is a complete introduction to building games in Unity following a project-based approach. You'll be introduced to the Unity game engine and the tools available for building and customizing a game exactly the way you want it, while maintaining a good code foundation to build upon. Once you get to grips with the fundamentals of Unity game development, you'll start creating a 2D collection game and an adventure game, followed by a 3D first person shooter game. Next, you'll explore advanced topics, such as using machine learning to create AI-based enemy behavior, virtual reality for extending the first-person game, and augmented reality for developing a farming simulation game in a real-world setting. The book will help you gain hands-on knowledge of these topics as you build projects using the latest game tool kits. You'll also learn how to commercialize your game by publishing it to a distribution platform and maintain and support it throughout its lifespan. As you progress, you'll gain real-world knowledge and experience by taking your games from conceptual design to completion. By the end of this Unity book, you'll have strong foundational knowledge of how to structure a Unity project that is both maintainable and extensible for commercially released games. What you will learn Build game environments and design levels, and implement game mechanics using Unity's features Explore 3D game creation, focusing on gameplay mechanics and player animation Develop customizable game systems using object-oriented architecture Build an MR experience using the XR Interaction Toolkit while learning how to merge virtual and real-world elements Get up to speed with advanced AI interactions using sensors and Unity's machine learning toolkit, ML-Agents Implement dynamic content in games using Unity LiveOps services like Remote Config Who this book is for If you find yourself struggling with completing game projects in Unity and want to follow best practices while maintaining a good coding structure, then this book is for you. This book is also for aspiring game developers and hobbyists with some experience in developing games, who want to design basic playable and commercial games in Unity with a core loop, player verbs, simple mechanics, and win/lose conditions. Experience with the Unity Editor interface and implementing functionality by creating C# scripts is required to get the most out of this book

Título: Unity 2022 by Example A Project-Based Guide to Building 2D and 3D Games, Enhanced for AR, VR, and MR Experiences Scott H. Cameron and Edward Falzon

Edición: First edition

Editorial: Birmingham, UK Packt Publishing Ltd [2024] 2024

Descripción física: 1 online resource (596 pages)

Contenido: Cover -- Title Page -- Copyright and Credits -- Foreword -- Contributors -- Table of Contents -- Preface -- Part 1: Introduction to Unity -- Chapter 1: Foundational Knowledge of Unity 2022 -- Technical requirements -- Unity Hub - Choosing the 2D URP template -- Installing Unity Hub -- Installing the Unity Editor - What version? -- What is a render pipeline? -- Creating a project -- Getting to know the Unity Editor and installing packages -- The Unity Package Manager -- Introducing the GameObject - All about Transform and components -- Adding GameObjects to the scene -- The Transform component -- Components -- 2D sprites with Sprite Creator - Understanding the Sprite Renderer and draw ordering -- Creating a new scene -- Creating a sprite using Sprite Creator -- Navigating the scene View -- Creating our player character -- Using Manipulation Tools -- Sprite Layers and Ordering -- Game Design Document (GDD) - Introducing the 2D collection game -- Summary -- Image sources -- Part 2: 2D Game Design -- Chapter 2: Creating a 2D Collection Game -- Technical requirements -- Creating a 2D, top-down game environment with Tilemap -- Level design - Guiding the player -- Creating Tile Palettes -- Building the collection game environment with Tilemap -- Making the level playable - Tilemap Collider 2D -- Introduction to creating scripts in C# - IDE, SOLID principles, and design patterns -- The IDE - Visual Studio Community 2022 -- The C# Language - Object-Oriented Programming (OOP) -- SOLID principles -- Design patterns -- Coding a simple player controller with the new Input System -- New Input System -- Player controller script -- Summary -- Chapter 3: Completing the Collection Game -- Technical requirements -- Using CM to follow the Player and playtesting -- Creating a Player Prefab -- Creating a 2D follow camera -- Playtesting the level Game mechanics and how to create with code (components) -- What is a game mechanic? -- Adding to our GDD -- Collecting pickups -- Hitting hazards -- Introduction to uGUI, the timer, counting, and winning -- Canvas -- TextMesh Pro -- Updating the pickup count -- The Timer script -- Winning the game -- Summary -- Part 3: 2D Game Design Continued -- Chapter 4: Creating a 2D Adventure Game -- Technical requirements -- Extending the GDD - Introducing the 2D adventure game -- Importing assets to use with Sprite Shape - A different kind of 2D environment builder -- Importing and preparing the artwork -- Level and environment design - Guiding the player -- Signposting -- Creating platforms -- Moving platforms and triggers - Creating a dynamic interactable environment -- Moving a Sprite Shape platform with Splines -- Triggering actions in the level -- Adding polish to our environment to immerse the player and optimizing -- Polishing the environment -- Optimizing draw calls -- Summary -- Chapter 5: Continuing the Adventure Game -- Technical requirements -- Setting up the player character with PSD Importer -- Rigging the actor -- Generating the sprite mesh geometry -- Adjusting the bone influence -- Editing sprite bone weights -- Setting up inverse kinematics (IK) -- Creating actor animations -- Using an Input Action Map -- Moving the player with a player controller script -- Processing Player Input -- Creating the PlayerController script -- Physics materials -- Animating the character with Mecanim -- Transitioning animation states -- Changing the animation state with code -- Flipping the player character -- Summary -- Chapter 6: Introduction to Object Pooling in Unity 2022 -- Technical requirements -- The object pooling pattern -- The Unity object pooling API -- Creating a new object pool -- Additional parameters affecting the object pool A pooled player shooting model -- Creating the pooled player shooting model -- Implementing the pooled shooting model -- Adding pooled shooting to the player character -- Summary -- Chapter 7: Polishing the Player's Actions and Enemy Behavior -- Technical requirements -- Polishing with Shader Graph and Trail Renderer -- Enabling post-processing -- Applying glow to the bullet with Shader Graph -- Creating a new Shader Graph 2D material -- Adding a 2D light to the player -- Polishing is easy with Trail Renderer -- Enemy Prefabs and variants - Configuring with SOs -- Creating an enemy Prefab with configurations -- Creating an enemy variant for alternate enemy types -- Creating a Prefab Variant -- Implementing basic enemy behavior using an FSM -- State Model -- A simple FSM pattern -- Changing state behaviors -- Summary -- Chapter 8: Extending the Adventure Game -- Technical requirements -- Health and inflicting damage -- Health system -- Interfaces required! -- Taking damage - IDamage interface --

ProjectileDamage component -- Healing - IHeal interface -- Controlling what damages/heals what -- Updating the player and enemy to use health -- Assigning the object with health - IHaveHealth interface -- Process changes to health -- Enemy wave spawner -- Creating the enemy spawner Prefab -- Integrating spawning with patrol behavior -- Summary -- Chapter 9: Completing the Adventure Game -- Technical requirements -- Creating an event system in C# to tie things together loosely -- The new event system -- Systems GameObject -- Creating a quest system for a collecting keys mission -- The quest system -- Script Execution Order -- The quest -- Collecting keys -- Solving the key puzzle and winning the game -- Sliding tile puzzle -- Winning -- Timeline -- Summary -- Part 4: 3D Game Design -- Chapter 10: Creating a 3D First Person Shooter (FPS) Technical requirements -- Designing for 3D while continuing the GDD -- Greyboxing a 3D environment with ProBuilder and Prefabs -- Habitat interior level -- Installing ProBuilder -- Modular parts, Prefabs, and Variants -- Greyboxing the level design -- Creating an FPS player character with the Unity Starter Asset -- Installing the Unity Starter Assets -- Starter Assets Playground scene -- Getting around -- Adding the first-person controller to our level -- Playtesting the level -- Refactoring environment interactions to 3D API methods -- Revisiting the TriggeredEvent component -- Implementing a TriggeredEvent in our level design -- Animating the door opening -- Code reuse in practice - Adding premade components to the player -- Constant damage script -- Inspector Debug -- Recharging aka healing -- Summary -- Chapter 11: Continuing the FPS Game -- Technical requirements -- Decorating the 3D environment -- Updating and replacing Prefabs -- Applying new materials -- Immersing the player using Polybrush and Decals -- Painting objects with Polybrush -- Painting/scattering objects -- Surface story with Decals -- Lighting design - Probes, Decals, light baking, and performance -- Setting the URP Forward+ Rendering Path -- Proxy lighting with Decals (yes, Decals) -- Bake that lighting? -- Light Probes -- Baked lighting dynamic shadows -- Blob shadows -- Summary -- Chapter 12: Enhancing the FPS Game with Audio -- Technical requirements -- Adding audio using the Audio Mixer -- Sound design 101 for games -- Adding audio to the game -- Building an immersive soundscape with music, SFX, and ambiance -- Playing music -- Playing SFX -- Playing SFX 3D -- Playing ambient sound -- Enhancing the audio experience with footsteps and reverb zones -- Reusing audio player code -- Adding a method overload to AudioPlayerSFX -- Implementing AudioPlayerFootsteps Implementing sprinting -- Adding reverb zones -- Deeper SOLID refactoring -- Summary -- Part 5: Enhancing and Finishing Games -- Chapter 13: Implementing AI with Sensors, Behavior Trees, and ML-Agents -- Technical requirements -- Refactoring the 2D enemy systems to 3D with NavMesh -- Importing scripts from the 2D project -- Refactoring the PatrolWaypoints class for NavMesh -- Configuring the enemy NavMesh Agent (Prefab) -- Adding waypoints to the level and testing -- Dynamic enemies with sensors and behavior trees -- Creating sensory behaviors -- Wrangling behaviors with a behavior tree -- Introducing ML with ML-Agents -- Navigating training efficiency with NavMesh -- Examining an ML-Agents setup -- Summary -- Chapter 14: Entering Mixed Reality with the XR Interaction Toolkit -- Technical requirements -- Introduction to MR and development frameworks -- XR Interaction Toolkit (XRI) -- AR Foundation -- OpenXR: Meta package -- Designing a boss room -- Setting up the physical space -- Creating the Unity project -- Laying out the boss room scene -- Working with AR planes (AR Foundation) -- Spawning using planes with AR Plane Manager -- Instantiating on a table plane -- Instantiating using the floor plane -- Instantiating with wall planes -- Toggling MR visuals with XR Input -- Placing interactable objects in the world -- Making objects XR interactables -- Placing the modules in the room -- Making the module slots interactable -- Configuring the laser gun -- Implementing the boss room mechanics -- Solving the crystal modules puzzle -- Spawning enemies -- Completing the game loop -- Summary -- Chapter 15: Finishing Games with Commercial Viability -- Technical requirements -- Introducing GaaS - UGS -- Introducing Unity DevOps -- Introducing Unity LiveOps -- Safeguarding your investment! Source code management with Unity Version Control Catering VCS for programmers

ISBN: 1-80323-795-3

Materia Título preferido: Unity (Electronic resource)

Materia: Video games- Programming

Autores: Falzon, Edward, author

Enlace a formato físico adicional: 1-80323-459-8

Baratz Innovación Documental

- Gran Vía, 59 28013 Madrid
- (+34) 91 456 03 60
- informa@baratz.es