1. Record Nr. UNINA9910617306603321

Autore	Ilett Daniel
Titolo	Building quality shaders for Unity® : using shader graphs and HLSL shaders / / Daniel Ilett
Pubbl/distr/stampa	Berkeley, California : , : Apress L. P., , [2022] ©2022
ISBN	1-4842-8652-9
Descrizione fisica	1 online resource (745 pages)
Disciplina	929.605
Soggetti	Computer graphics Computer games - Programming Three-dimensional display systems
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Includes index.
Nota di contenuto	Intro -- Table of Contents -- About the Author -- About the Technical Reviewer -- Acknowledgments -- Introduction -- Chapter 1: Introduction to Shaders in Unity -- Rendering -- Game Data -- The Vertex Shader -- The Fragment Shader -- Unity's Render Pipelines -- Shader Graph -- Summary -- Chapter 2: Math for Shader Development -- Vectors -- Position and Direction Vectors -- Vector Addition and Subtraction -- Scalar Multiplication -- Vector Magnitude -- Vector Normalization -- Basis Vectors and Linear Combinations -- Dot Product -- Cross Product -- Matrices -- Matrix Addition and Subtraction -- Scalar Multiplication -- Square, Diagonal, and Identity Matrices -- Matrix Transpose -- Matrix Determinant -- Matrix Multiplication -- Matrix Inverse -- Matrix Transformations -- Scaling Matrices -- Rotation Matrices -- Translation Matrices -- Homogeneous Coordinates -- Space Transformations -- Object-to-World Space Transformation -- World-to-View Space Transformation -- View-to-Clip Space Transformation -- Perspective Divide -- Summary -- Chapter 3: Your Very First Shader -- Project Setup -- Creating a Project -- Setting Up the Scene -- Writing a Shader in Unity -- Writing ShaderLab Code -- Adding a SubShader -- SubShader Tags -- Adding a Pass -- Pragma Directives and Includes -- Controlling Data Flow with Structs -- The appdata Struct -- The v2f Struct -- Variables

in HLSL -- The Vertex Shader -- The Fragment Shader -- The SRP Batcher and Constant Buffers -- Common Shader Syntax -- Scalar Types -- Vector Types -- Matrix Types -- Included Variables -- Transformation Matrices -- Time-Based Variables -- Summary -- Chapter 4: Shader Graph -- Shader Graph Overview -- Creating a Basic Shader in Shader Graph -- Scene Setup -- The Shader Graph Interface -- The Toolbar -- Togglable Windows -- The Graph Environment -- Adding Properties -- Property Naming.
Common Property Options -- Color Property Options -- Building the Graph -- Shader Graph Features -- Adding Blocks to the Master Stack -- Redirect Node -- Preview Node -- Color Mode -- Shader Path -- Node Groups -- Sticky Notes -- Node Snapping -- Sub Graphs -- Color Invert Sub Graph -- Custom Function Node -- Color Invert Custom Function -- Custom Function String Mode -- Custom Function File Mode -- Summary -- Chapter 5: Textures and UV Coordinates -- Basic Texturing -- Texture Support in HLSL -- Adding Texture Properties -- Texture Coordinates -- Texture Sampling -- Texture Support in Shader Graph -- Applying Tiling and Offset Vectors -- Sampling Textures -- Mipmaps and Level of Detail -- Level of Detail -- Texture LOD in HLSL -- Texture LOD in Shader Graph -- Sampling Options -- Wrap Mode -- Filter Mode -- Sampler States -- Sampler States in HLSL -- Sampler States in Shader Graph -- Summary -- Chapter 6: Advanced Texturing -- Base Shader in HLSL -- Modifying Texture Coordinates -- UV Rotation -- UV Rotation in HLSL -- UV Rotation in Shader Graph -- Flipbook Mapping -- Flipbooks in HLSL -- Flipbooks in Shader Graph -- Polar Coordinate Mapping -- Polar Coordinates in HLSL -- Polar Coordinates in Shader Graph -- Triplanar Mapping -- Triplanar Mapping in HLSL -- Triplanar Node in Shader Graph -- UV Shear -- Shear in HLSL -- Shear in Shader Graph -- Texture3D -- Importing 3D Textures from 2D Textures -- Blended Particles Using Texture3D -- Blended Particles in HLSL -- Blended Particles in Shader Graph -- Cubemaps -- Importing Cubemaps from 2D Textures -- Using Cubemaps in HLSL -- Sampling a Cubemap in HLSL -- Sampling a Reflected Cubemap in HLSL -- Using Cubemaps in Shader Graph -- Sampling a Cubemap in Shader Graph -- Sampling a Reflected Cubemap in Shader Graph -- Summary -- Chapter 7: The Depth Buffer -- How Rendering Opaque Objects Works.
How the Depth Buffer Works -- Render Queues -- Depth Testing and Writing -- The ZTest Keyword -- The ZWrite Keyword -- Depth Settings with Shader Graph -- Early Depth Testing -- Z-Fighting -- Shader Effects Using Depth -- Silhouette -- Silhouette Effect in HLSL -- Preparing the Depth Texture -- Camera Depth Texture -- Reading Depth Values -- Converting Depth Values -- Silhouette Effect in Shader Graph -- Writing to the Depth Texture Using a Depth-Only Pass -- Depth-Only Pass in the Built-In Pipeline -- Depth-Only Pass in URP -- Scene Intersections -- Scene Intersections in HLSL -- Scene Intersections in Shader Graph -- The Stencil Buffer -- Stencil Testing in HLSL -- Stencil Mask Shader -- Reading the Stencil Buffer -- URP Renderer Features -- Render Objects -- X-Ray Effect Using Depth -- X-Ray Effect with HLSL -- X-Ray Effect with HLSL in the Built-in Pipeline -- Textured Pass -- X-Ray Pass -- X-Ray Effect with HLSL in URP -- X-Ray Effect with URP Render Objects (No Code) -- Rendering Objects Behind Walls -- Rendering Unobstructed Objects -- Impossible Room Effect Using Stencils -- Impossible Room Effect with HLSL -- Impossible Room Effect with Render Objects -- Summary -- Chapter 8: Transparency and Alpha -- How Rendering Transparent Objects Works -- How Alpha Blending Works -- Alpha Blending in HLSL -- Premultiplied Blending -- Additive Blending -- Multiplicative Blending

-- Blend Mode Properties -- Alpha Blending in Shader Graph -- Other Blend Modes -- Alpha Clipping -- Alpha Cutout in HLSL -- Alpha Cutout in Shader Graph -- Dithered Transparency with Alpha Clip -- Dithered Transparency in HLSL -- Dithered Transparency in Shader Graph -- Dissolve Effect with Alpha Clip -- Dissolve Effect in HLSL -- Dissolve Effect in Shader Graph -- Summary -- Chapter 9: More Shader Fundamentals -- Interacting with C# Code.
Cycling Colors with the HelloWorld Shader -- World-Space Dissolve -- World-Space Dissolve in HLSL -- World-Space Dissolve in Shader Graph -- World-Space Dissolve Scripting -- Controlling Shaders with Animations -- Shader Keywords -- Shader Keywords in HLSL -- Declaring Keyword Properties -- Using Keywords -- Shader Keywords in Shader Graph -- Modifying Keywords Using C# Scripting -- UsePass in ShaderLab -- GrabPass in ShaderLab -- Sepia Tone Effect Using GrabPass in the Built-in Pipeline -- Sepia Tone Effect Using Camera Opaque Texture in URP -- Sepia Tone Effect Using the Scene Color Node in Shader Graph -- Summary -- Chapter 10: Lighting and Shadows -- Lighting Models -- Ambient Light -- Diffuse Light -- Specular Light -- Fresnel Light -- Blinn-Phong Reflection Model -- Flat Shading -- Flat Shading in HLSL -- Accessing Lights in the Built-In Pipeline -- Accessing Lights in URP -- Flat Shading in Shader Graph -- Gouraud Shading -- Gouraud Shading in HLSL -- Gouraud Vertex Shader in the Built-In Pipeline -- Gouraud Vertex Shader in URP -- Gouraud Shading in Shader Graph -- GetMainLight Sub Graph -- GetAmbientLight Sub Graph -- Gouraud Shading with Custom Interpolators -- Phong Shading -- Phong Shading in HLSL -- Phong Shading in the Built-In Pipeline -- Phong Shading in URP -- Fresnel Light Modification -- Phong Shading in Shader Graph -- Physically Based Rendering -- Smoothness -- Metallic Mode -- Specular Mode -- Normal Mapping -- Ambient Occlusion -- Emission -- PBR in the Built-In Pipeline -- PBR in URP -- PBR in Shader Graph -- Shadow Casting -- Shadow Casting in the Built-In Pipeline -- Shadow Casting in URP -- Shadow Casting in Shader Graph -- Summary -- Chapter 11: Image Effects and Post-Processing -- Render Textures -- Post-Processing Effects -- Grayscale Image Effect -- Post-Processing in the Built-In Pipeline.
Grayscale C# Scripting in the Built-In Pipeline -- Grayscale Shader in the Built-In Pipeline -- Post-Processing in URP -- Grayscale C# Scripting in URP -- The GrayscaleSettings C# Script -- The GrayscaleRenderPass C# Script -- The GrayscaleFeature C# Script -- The Grayscale Shader -- Using Volumes with Grayscale in URP -- Post-Processing in HDRP -- Grayscale C# Scripting in HDRP -- Grayscale Shader in HDRP -- Using Volumes with Grayscale in HDRP -- Gaussian Blur Image Effect -- Gaussian Blur in the Built-In Pipeline -- Gaussian Blur C# Scripting in the Built-In Pipeline -- Gaussian Blur Shader in the Built-In Pipeline -- Gaussian Blur in URP -- Gaussian Blur C# Scripting in URP -- The GaussianBlurSettings C# Script -- The GaussianBlurRenderPass C# Script -- The GaussianBlurFeature C# Script -- Gaussian Blur Shader in URP -- Gaussian Blur in HDRP -- Gaussian Blur C# Scripting in HDRP -- Gaussian Blur Shader in HDRP -- Summary -- Chapter 12: Advanced Shaders -- Tessellation Shaders -- Water Wave Effect with Tessellation -- Wave Tessellation in Shader Code -- Wave Vertex Shader -- Wave Tessellation Control (Hull) Shader -- Wave Tessellation Evaluation (Domain) Shader -- Wave Tessellation tessVert Function and Fragment Shader -- Wave Tessellation in Shader Graph -- Level of Detail Using Tessellation -- Level of Detail Tessellation in Shader Code -- Level of Detail Tessellation in Shader Graph -- Geometry Shaders -- Visualizing Normals with Geometry

	Shaders -- Compute Shaders -- Grass Mesh Instancing -- The ProceduralGrass C# Script -- ProceduralGrass Properties -- ProceduralGrass Start Method -- ProceduralGrass RunComputeShader Method -- ProceduralGrass Script Update Method -- ProceduralGrass Script OnDestroy Method -- The ProceduralGrass Compute Shader -- The Grass Blade Shader -- Summary -- Chapter 13: Profiling and Optimization -- Profiling Shaders. FPS or Milliseconds?.
Sommario/riassunto	Understand what shaders are and what they're used for: Shaders are often seen as mystical and difficult to develop, even by skilled programmers, artists, and developers from other game design disciplines. This book dispels that idea by building up your shader knowledge in stages, starting with fundamental shader mathematics and how shader development mindset differs from other types of art and programming, and slowly delves into topics such as vertex and fragment shaders, lighting, depth-based effects, texture mapping, and Shader Graph. This book presents each of these topics with a comprehensive breakdown, the required theory, and some practical applications for the techniques learned during each chapter. The HLSL (High Level Shading Language) code and Shader Graphs will be provided for each relevant section, as well as plenty of screenshots. By the end of this book, you will have a good understanding of the shader development pipeline and you will be fully equipped to start making your own aesthetic and performant shader effects for your own games! You Will Learn To " Use shaders across Unity's rendering pipelines " Write shaders and modify their behavior with C# scripting " Use Shader Graph for codeless development " Understand the important math behind shaders, particularly space transformations " Profile the performance of shaders to identify optimization targets Who Is This Book For This book is intended for beginners to shader development, or readers who may want to make the jump from shader code to Shader Graph. It will also include a section on shader examples for those who already know the fundamentals of shaders and are looking for specific use cases.