Unity®

Game design

Curricular framework

A resource for educators and administrators to bring interactive application and game design to the classroom

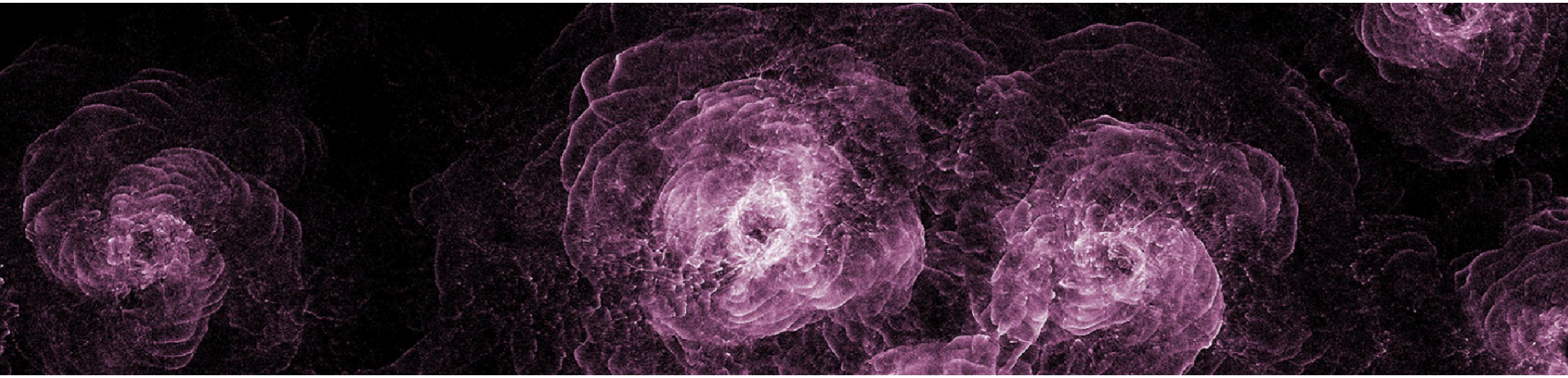


Table of Contents

Introduction

Why this framework?

This document is a comprehensive framework for teaching game design with Unity at the post-secondary level. With an emphasis on modular design and certification alignment, this document aims to make it easy for teachers to assemble courses of study that will outfit students with all of the technical skills they'll need to succeed as professionals in the field.

The framework has three guiding principles:

- **Professional targeting:** The framework covers both technical and soft skills, including receiving critique, code and asset review, and portfolio development, all of which is crucial for budding professionals going into the field of game design or 3D asset development.
- Certification alignment: The framework's modules are marked to indicate where they align with exam objectives for all of Unity's Associate and Professional certifications.
- Learn-based resources: The skills outlined in the framework modules are scaffolded with Unity Learn resources that can be used to support instructor and student learning.

The curricular framework provides links to free learning resources from Unity Learn, the Unity User's Manual, and suggested readings to meet the learning objectives and support all pedagogical approaches (synchronous, asynchronous, blended, in person or distance learning). These resources are updated as the real-time visualization landscape and the platform development tools change, so we recommend that you check back periodically to ensure you have the latest version.

How to use this document

Unity is used in various contexts across schools with learners of diverse backgrounds and prerequisite knowledge. For this reason, building a teaching guide that caters to the needs of all users is impractical. This document provides an in-depth overview of the skills involved in creating real-time 3D experiences to support the planning of your unique learning path.

Each module presents a table of skills , followed by a table of suggested learning objectives (see sample below) . This table displays the relevant learning objective , the available resources for the objective , and any certification exam objectives covered by the content .

Skill and Description	Learning Objectives	Resources	Related Certifications
Analyze the impact of art assets and lighting on performance (polycount, particles, visual effects, lighting, and shadows) Analyzing the impact on performance of factors such as poly count, particles, visual effects, lighting, and shadows involves assessing how these elements affect the frame rate and overall performance of a Unity project. **Assessment Suggestion:** Provide the learner with a Unity scene featuring various performance-intensive elements, and have them systematically identify, measure, and address performance issues related to factors like poly count, particles, visual effects, lighting, and shadows to optimize the scene's performance while maintaining an acceptable visual quality.	• Recognize the effects that Rigidbody and Collider components have on performance • Set up the Unity Profiler to identify elements that cause performance impact • Apply Unity's Stats window in order to investigate performance issues caused by assets	**Unity Learn** • Optimization	

Sample of module

Learning objectives

Each module includes suggested learning objectives. We have identified these objectives based on typical knowledge or skills that are related to the specific module. While it is not critical to cover every learning objective in a module, the objectives are designed to complement each other in helping you fulfill the module aims.

Resources

Suggested resources throughout this framework support the mastery of skills outlined in each module. These include free learning resources from Unity Learn, our official online learning platform, so that you can continue your learning and help students meet their objectives. We also highlight material from the Unity User's Manual, as well as other suggested readings. When using the Unity User Manual, ensure that it reflects the Unity version you are using by selecting the correct version from the drop down menu in the upper left hand corner of the page.

Related Certifications

The learning objectives in this framework have been aligned to Unity Certified Associate Certification exam objectives for educators aiming to prepare students to be certified. To ensure that students can be adequately prepared for the exams, it is recommended that educators take the exam themselves to gain a firm understanding of the exam's content and format.

Preparing students for certification

One of the goals of this document is to help teachers develop programs that will lead students towards achieving a Unity Certification. Certifications test the core skills needed for a variety of roles so that students can validate their expertise and showcase their readiness for a role using Unity. The framework aligns with several Unity Certifications including the Associate: Game Developer, Associate: Programmer, and Associate Artist certifications. These are appropriate credentials for students looking to make the transition into professional work. Holding an Associate Certification indicates that a student has a mature understanding of Unity and is ready to begin in a junior or associate position on a professional team.

Unity Certified Associate **Game Developer**	Unity Certified Associate **Programmer**	Unity Certified Associate **Artist**
This certification is designed for future game developers who want to showcase their mastery of core Unity skills and concepts to obtain their first professional Unity role. Successful exam takers have a background in computer science or have a solid grasp of the skills required to become a Unity game developer. **Covers** ● Scripting, building, debugging, and optimization ● Prototyping art assets, whiteboxing levels	This certification is designed for students interested in a first professional role as a Unity developer, software engineer, software developer, mobile application developer, or gameplay programmer. Successful exam takers have a background in computer science. **Covers** ● Advanced Unity tools ● Scripting in C# for Unity ● Scripting UI elements	This certification is designed for students interested in a first professional role as a 3D artist, 3D generalist, game artist, level designer, environment artist, or 3D visualization artist. Successful exam takers have a mix of artistic and technical skills. **Covers** ● 3D asset creation, terrain generation, Scene building ● Basic understanding of C# in Unity

Additional teaching and learning resources

As well as providing the tutorials and projects that support the learning objectives throughout this framework, Unity Learn offers guided learning pathways that may be integrated into, or used in addition to, the materials in your program. These longer, self-paced experiences are designed to help anyone interested in coding and breaking into the gaming and tech industries expand their professional opportunities by gaining the skills they need to obtain a job, regardless of prior experience.

[Unity Essentials pathway]
Designed for anyone new to Unity, this guided learning journey is a first step toward gaining the background, context, and skills needed to confidently create in the Unity Editor. Completing this Pathway will equip students with the foundation needed to further their learning and specialize in their area of interest.

[Junior Programmer pathway]
Designed for anyone interested in learning to code or obtaining an entry-level Unity role, this pathway assumes a basic knowledge of Unity and has no math prerequisites. By the end of the Junior Programmer pathway, students will be equipped to take the Unity Certified Associate: Programmer exam.

[Creative Core pathway]
Creative Core is your next step toward becoming a Unity creator. This free learning path will teach you all the core elements you need to bring your imagination to life with Unity. Once you've completed Unity Essentials as an introduction to the fundamentals of the Unity Editor, take this pathway to learn Visual Effects (VFX), Lighting, Animation, Audio, UI, and other creative skills, no programming required.

[VR Development pathway]
Welcome to VR Development! This learning pathway is designed for anyone interested in learning to create experiences for VR. This pathway assumes a basic knowledge of Unity and basic knowledge of programming.

[Mobile AR Development Pathway]
Ready to create AR experiences? In this learning pathway, you'll develop AR apps compatible with iOS and Android devices!

For those interested in how Unity can be a tool for Metaverse related technologies and applications, a Live Learning series, called Road to the Metaverse is available on Unity Learn.

Unity eBooks
You can find these and many more resources here.

- Unity Game Dev Field Guide - This guide will help you jump-start your familiarity with the latest in Unity's rich feature set and intuitive workflows
- Unity for Technical Artists - provides an overview of the toolsets and systems in Unity that Technical Artists can use
- The definitive guide to lighting in the High Definition Render Pipeline - learn how to harness the power of physically based lighting in the HDRP
- Top tips for improving your workflows and productivity with Unity 2020 LTS - a guide that collects over 70 time-saving tips to improve your day-to-day aggregate workflow with Unity
- UI design and implementation - a treasure trove of useful tips for advancing your UI development skills with the default Unity UI and the new UI Toolkit.
- Create modular game architecture in Unity with ScriptableObjects - This guide provides tips and tricks from professional developers for deploying ScriptableObjects in production.
- The definitive guide to creating advanced visual effects in Unity- This e-book provides a complete overview of how to use visual effects authoring tools in Unity to create any kind of effect.
- User interface design and implementation in Unity - Written by experienced Unity creators and UI professionals, the e-book provides step-by-step guidance on how to make UIs that look great across a wide range of devices.
- Best Practices From Successful Mobile Indies - Learn best practices for mobile success with tips from indie experts.
- Introduction to the Universal Render Pipeline for advanced Unity creators - This e-book was created by a highly experienced Unity developer in collaboration with senior graphics engineers at Unity.
- Build Industrial Digital Twins| Free Guide for Robotics & Automation | Unity - provides a practical, hands-on guide for robotics engineers, automation professionals, and OEMs ready to accelerate their digital transformation with Unity Industry.

Unity hardware requirements

You can find the latest Unity hardware requirements in the Unity documentation. Go to Unity – Manual and then select **Working in Unity** > **Installing Unity** > **System requirements for Unity [version]**.

Getting support from the community

The Unity creator community is a vibrant and engaged network of Unity enthusiasts who embody vast knowledge. Whether you're researching your own area of interest or guiding students to troubleshoot, we recommend starting with the following resources within the Unity ecosystem:

Unity Discussions
Beginners and experts alike post to this platform, so they can help each other out with Unity. The built-in voting system helps you find the best answers faster.

While we would love for you to find the answers to all of your questions here on the Unity Learn platform or within the wider Unity learning ecosystem, we know that our community is much broader. We encourage you to research and connect in the many spaces in which our creator community lives. Here are a few of the better-known resources in the Unity creator community:

YouTube
There are many channels and videos dedicated to learning Unity. Some popular channels include Game Dev Unlocked (created by established creator David Wehle), Brackeys, Code Monkey, and Dani, as well as our own official Unity channel.

Discord
Discuss Unity in real-time. Join the Discord server.

Stack Exchange and Stack Overflow
These open communities help creators in diverse fields get their questions answered with a reputation award process. Stack Overflow is dedicated to programming. On Stack Exchange, check out questions tagged "unity" in the gamedev exchange.

Reddit
A network of communities based on people's interests. Take a look at the Unity, Unity3D, and Unity2D communities just for starters.

X (previously Twitter)
Follow @unity3D and watch #unity and other hashtags to see what the Unity community is creating.

Creating in Unity without programming

Although programming is a helpful skill to have when developing projects with complex interactivity in Unity, it is not necessary to be a coder to create with Unity. For example:

Certain types of projects, such as 3D visualizations and animations, don't require code at all.

- Visual scripting allows developers to implement logic in their projects using intuitive drag-and-drop graphical connectors without any knowledge of code or IDEs.
- The Unity Asset Store provides pre-made scripts and tools for the development of common features, such as a first-person controller or an inventory system.

Using Google, combined with sites like Unity Discussions, Unity Discussions, and Stack Overflow, developers can copy, paste, and modify the coding solutions provided by other developers. (It is surprising how far you can get with a little Googling and a lot of perseverance!

Modules in this Curricular Framework

Introduction to the Unity real-time platform	This module is intended as an introduction to the Unity Editor and how to use it. Students who will be doing practical projects in game design need to be familiar and comfortable with the Unity Editor. If students will be using their own devices they would ideally be given time outside of class to complete the first few steps of onboarding in Unity.
Building scenes	In this module, you will explore common techniques for creating and importing objects into a scene in Unity as well as setting up and building your scene. While the Unity real-time engine provides tools to quickly create prototypes or basic scenes, most advanced assets will probably be created in other dedicated software applications. This module will also go over the process of setting up a workflow between the Unity real-time engine and these third party 3D modeling applications. Section B in this module deals with the development of 2D applications using the Unity Editor. We provide links to relevant tutorials on the Unity Learn

	platform, but would also suggest the 2D game art, animation, and lighting for artists ebook as a useful resource for all things 2D in Unity.
Work with 2D assets in Unity	In this module you'll delve into the essentials of creating 2D projects in Unity. This includes understanding the unique tools and workflows tailored for 2D environments, such as sprite manipulation, animation techniques, and efficient scene management. You'll learn how to effectively use Unity's 2D Renderer for creating visually appealing and interactive 2D experiences. The module will also guide you through the nuances of 2D physics and character control, ensuring you can create dynamic and responsive 2D games or applications. By the end of this module, you'll be equipped with the knowledge to craft engaging and polished 2D projects in Unity.
Working with materials and shaders	This module provides an in-depth exploration of materials and shaders within the context of computer graphics. Through this module, you will develop a comprehensive understanding of techniques for creating and manipulating textures, surfaces, and visual effects to enhance digital projects. Whether you have prior experience in 3D art or are new to computer graphics, this module offers essential knowledge and practical skills for achieving realistic and visually compelling results.
Cameras and Cinemachine	In this module, you'll explore the power and versatility of Unity's camera systems and the Cinemachine framework. This course will guide you through the process of setting up and manipulating cameras to capture your game world effectively. You'll learn how to use Cinemachine's intelligent camera tools to create dynamic, cinematic game sequences with ease. The focus will be on understanding camera properties, behaviors, and how to leverage Cinemachine's advanced features for smooth and responsive camera movement. By mastering these tools, you'll be able to enhance the visual storytelling and immersion of your Unity projects, creating engaging and visually compelling experiences.
Animating in Unity	In this module, you will learn the essentials of animating in Unity. We will cover key concepts such as keyframe animation, rigging, and using the Animator component, alongside practical scripting techniques. By the end of this course, you'll be equipped to create dynamic and responsive animations for your Unity projects.

Lighting in Unity	Welcome to the module on lighting in Unity, where we'll delve into the essential aspects of illuminating your virtual worlds. Throughout this module, we'll explore the diverse array of light types available in Unity, including directional, point, and spotlights, and how to effectively utilize them to shape your scenes. Additionally, we'll discuss the concept of light baking, a vital technique for optimizing real-time rendering performance. Understanding the pivotal role of lighting in crafting cinematic graphics, we'll guide you through the principles and practices that bring your virtual environments to life with dynamic and visually compelling illumination.
User interfaces in Unity	In this module, you will delve into the fundamentals of User Interface (UI) design in Unity, a crucial component for creating engaging and user-friendly applications. UI in Unity encompasses the visual elements that users interact with, such as buttons, menus, and text, and is pivotal for ensuring a seamless and intuitive user experience. You will explore how to effectively design, implement, and optimize these UI elements within the Unity environment, highlighting their significance in guiding user interactions and enhancing the overall functionality of your application. This module aims to equip you with the skills to craft visually appealing and responsive UIs, ensuring your Unity projects are not only functional but also aesthetically pleasing and easy to navigate. By mastering UI in Unity, you'll be taking a significant step towards creating more immersive and user-centric digital experiences.
Audio in Unity	In this module, you'll explore the intricacies of audio in Unity, learning how to effectively integrate and manipulate sound to enhance the immersion and impact of your projects. You'll delve into Unity's audio engine, discovering techniques for adding depth and realism through spatial sound, environmental audio, and dynamic sound effects. This focused approach will equip you with the skills to create rich, engaging audio landscapes, elevating the overall experience of your Unity creations.
VFX in Unity	In this module, you will learn about Visual Effects (VFX) in Unity, focusing on the dynamic and versatile tools of particle systems and the VFX Graph. These powerful features are essential for creating stunning visual effects that can bring your Unity projects to life. You will learn how to use particle systems for simulating complex phenomena like fire, smoke, and water, as well as how to leverage the VFX Graph for more advanced, customizable effects.

Introduction to C# in Unity	While it's certainly possible to create an application in Unity without scripting, it will be severely restricted in functionality. C# scripting in Unity unlocks new functionality and allows you to create amazing VR experiences. In this module, you'll learn about the goals of the Unity C# Scripting Fundamentals project, including scripting basics, controlling code flow, basic GameObject manipulation, and GameObject interactions.
Unity Gaming Services	Unity Gaming Services is an end-to-end platform that is designed to help you build, engage, and grow your game. These services allow you to take your game to the next level without having to worry about maintaining or scaling your back-end infrastructure and simplify many game development tasks and challenges.
Optimization and publishing	In this module, you will learn to balance aesthetics and performance in Unity by analyzing the impact of art assets and lighting. This includes understanding how poly count, particles, visual effects, and shadows affect performance. You'll create and deploy basic builds, implement Level of Detail (LOD) groups and objects to optimize scenes, and use mobile SDKs for testing and publishing applications. The module emphasizes optimizing application performance for smooth framerates, ensuring an immersive and responsive experience.
Professional skills	Professional skills are some of the most broadly applicable and easily transferable of the skills that are highlighted in the curricular framework. The learning objectives here focus on the soft skills students should have to secure a position in the industry and for ongoing growth and success as part of a team.
Unity AI	AI can help you to be more productive while staying fully in control of your vision. It offers the possibility of in-game features and capabilities that couldn't be built otherwise, potentially revolutionizing player experiences by embedding AI models in the runtime so content reacts and responds to players and users in new ways.

Introduction to the Unity real-time platform

Module introduction

Unity is the world's leading platform for creating and operating interactive, real-time 3D content, providing the tools to make amazing experiences and publish them to a wide range of devices.

The cross-platform nature of the Unity 3D platform means you can build your content once, and then deploy across over 20 platforms, including Windows, Mac, iOS, Android,PlayStation, Xbox, Nintendo Switch, and the leading AR and VR platforms.

This module is intended as an introduction to the Unity Editor and how to use it. Students who will be doing practical projects in game design need to be familiar and comfortable with the Unity Editor. If students will be using their own devices, they would ideally be given time outside of class to complete the first few steps of onboarding in Unity.

We suggest giving the following free resources to students for preparatory self-study before classes commence: Editor Essentials module from the Unity Essentials pathway.

If you are interested in a more comprehensive deep dive into 2D development in the Unity Editor, our most comprehensive 2D game development guide is now available, as well as the new Sprite Flight and 2D adventure Game tutorials.

Unity also provides a growing range of services, most with free tiers of use, to help developers build, manage, and grow their business from their applications, as well as extend and integrate into 3rd party applications. Below is a list of notable services that may be useful in the industries this curricular framework caters to, but the full range can be viewed on the Unity Gaming Services reference and the Unity Cloud onboarding guide.

Unity Gaming Services

- Accounts
- Mulltiplayer
- Content Management
- Analytics
- Community Tools
- Monetization tools
- Game Crash Reporting Tools

Unity Cloud

- Unity Asset Manager
- Unity Version Control
- Unity Build Automation

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Create and manage projects in the Unity Hub Unity uses the Unity Hub to install and manage the various Unity versions and additional components. A Unity ID is	• Install a version of the Unity Editor using the Unity Hub • Create a new Unity project using a template in the Unity Hub	**Unity Learn** • Install Unity • Install a new Unity Editor • Create a new project • Add new modules to a Unity Editor	

required to access a lot of the functionality of the Unity Editor and will have all the licenses and assets from the Asset Store linked to it. With purposeful organization, learners can avoid being overwhelmed and create Unity projects that are easy to navigate. **Assessment suggestion** Evaluate students' Unity understanding by having them install Unity Hub, create a Unity ID, access the Asset Store, and organize a Unity project for efficient navigation. Additionally, test their ability to switch between different Unity versions within Unity Hub.	• Open an existing Unity project from the Unity Hub • Explain the differences between and purposes of LTS and TECH Stream releases • Update a project to a newer version of the Unity Editor using the Unity Hub • Explain the role of Unity Hub in creating and managing projects • Explain the purposes and uses of the sections of the Unity Hub interface • Explain the uses of the 3D, 2D, and Microgame templates in the Unity Hub • Add a Unity project from another source to the Unity Hub • Explain why version control is essential in real-time development among teams	• Install a package via the Package Manager • Project Organization **Unity Manual** • Install the Unity Hub • The Project window	
Create and manage Scenes	• Explain the role of scenes in a Unity project	**Unity Learn** • Create a new Scene • Open a Scene	• Associate Game Developer

Scenes in Unity are fundamental containers that hold and organize game objects, assets, and the environment for a specific part or level of a game. They are crucial for game development because they allow developers to structure and manage different parts of their game, enabling seamless transitions between gameplay elements, efficient asset loading, and streamlined testing and iteration, ultimately contributing to a more organized and manageable game development process. **Assessment Suggestion** Have students demonstrate their comprehension by instructing them to generate a fresh scene within their project, labelling it as "New Scene." Request that they showcase their ability to identify their current working scene and explain the process of switching to the newly created scene.	• Create a new empty 3D Scene • Create a new empty 2D Scene • Open a scene in a Unity project	**Unity Manual** • Scenes • Scene view navigation	
Identify and use essential features of the Unity Editor	• Identify and describe the windows that appear in	**Unity Learn**	• Associate Game Developer

The Unity Editor interface consists of various areas, each designed for specific tasks. Learners should familiarize themselves with these features before focusing on VR development. Since the Unity Editor is a professional tool, there's a lot to learn. The free Unity Essentials Pathway provides learners with a comprehensive guide to these essential features. **Assessment Suggestion** Have students demonstrate their comprehension by instructing them to open a new Unity project and identify the default layout of the Unity Editor. Ask them to take a screenshot and label the following windows: Scene view, Game view, Hierarchy, Project window, Inspector, and Console. Then, have them rearrange the layout by undocking the Inspector window and docking it on the opposite side of the Editor. Finally, ask students to	the Unity Editor's default view • Start and stop Play mode (Game view) • Rearrange, dock, and undock windows in the Unity Editor • Explain the differences between the Project and Hierarchy windows • Explain the relationship between the Hierarchy window and the Scene view • Explain when to use the Scene view and the Game view • Explain the purpose and functionality of the Package Manager • Use the Package Manager to add functionality to the Unity Editor • Explain the relationship between the Assets folder in the Project window and the Asset folder in file explorer • Organize assets using folders in the Project window	• Explore the Editor Interface **Unity Manual** • The Project window • The Editor interface • Scene view navigation • GameObjects • Tags and Layers • Unity's Asset Store	

explain the purpose of each window and how rearranging them might help their workflow.			
Employ Unity Version Control in a project Unity Version Control is a version control solution built to help teams manage changes and collaborate effectively within the Unity Editor. Originally known as Plastic SCM, it was acquired by Unity and has since been integrated directly into the Unity development workflow, offering tools for branching, merging, and managing project history tailored to real-time content creation. **Assessment suggestion:** Have students explain how they collaborated with PlasticSCM and identify the successes and challenges of setting up collaboration.	• Identify changed files of publish and update operations • Describe additions during publish operations • Recognize when to perform a publish or update • Recognize when to revert unintended changes prior to publishing • Locate where a project resides in the Unity Development Dashboard • Restore previous commits by using the version history • Explain the primary purposes of version control when working in Unity	**Unity Learn** • Get started with Unity Version Control • Unity Version Control: Quick start guide • Collaborate with Unity Version Control **Unity Manual** • Unity Version Control (previously Plastic SCM)	• Associate Game Developer • Associate: Programmer
Evaluate Unity and real-time 3D in order to determine whether they are suited to your needs	• Define the term real-time • Explain what the Unity real-time engine does	**Unity Learn** • Real-time creation	

The term real-time is used frequently in creative industries but is rarely clearly explained. Understanding what this term means as well as the impact a real-time 3D platform like Unity has on the creative workflow will allow learners to determine potential use cases and identify the problems it solves for creators. **Assessment suggestion:** Have students articulate how they may be able to implement real-time functionality in their projects to assist or enhance the desired outcome.	• Describe how real-time creation software is used in different industries • Identify a variety of real-time creators by their usage of Unity or their job role • Explain what a real-time game engine is and how it is used		

Building scenes

Module introduction

In this module, we'll explore the essential techniques for creating and importing objects into a Unity scene, along with the steps required to build and organise a functional environment. The focus is on equipping students with the practical skills needed to construct dynamic, interactive scenes.

While Unity provides powerful tools for rapid prototyping and basic scene creation, more advanced assets are typically produced in specialised 3D modelling applications. This module also introduces workflows for integrating third-party tools with the Unity Editor to support a professional production pipeline.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Choose an appropriate render pipeline for a project, given certain requirements. Unity provides two render pipelines that are optimized for specific hardware and use cases, as well as one legacy system for backward compatibility. Knowing which pipeline to use will allow students to create optimized experiences from the initial planning to the final production stages. **Assessment suggestion:** Have students create a cinematic scene using the High Definition Render Pipeline (HDRP), and then recreate the same scene in the Universal Render Pipeline (URP) to make it mobile-friendly.	• Define key terms of real-time graphics including rendering, render pipeline, and scriptable render pipeline • Identify the differences between Unity's provided render pipelines, including advantages, disadvantages, and common use cases for each • Create a new project using a particular render pipeline	**Unity Learn** • Understanding Scriptable Render Pipelines • Introduction to URP • Introduction to HDRP **Unity Manual** • Render pipelines • Universal Render Pipeline overview • High Definition Render Pipeline overview **Other Sources** • Introduction to URP for advanced creators • 3 studios on URP migration Unity • Guide to lighting in the HDRP • Unity for Technical Artists: key toolsets and workflows	• Associate: Artist
Configure a post-processing profile to achieve a specific visual style	• Set up a camera to allow for post-processing	**Unity Learn** • Post-processing	• Associate: Artist • 3D Artist

Post-processing allows the user to enhance their visual aesthetic using cinematic effects that are applied by the real-time engine just before rendering to the scene. This process allows for a global or local volume control over the visuals of your scene, and it can go a long way to adding a professional veneer to your work. **Assessment suggestions:** Have students try to recreate a chosen cinematic scene using post-processing effects. Alternatively, give students a still from a film and have them identify the effects used. CG-heavy films, like those from Marvel, are good candidates for this.	• Add a new global post-processing volume to the scene and assign a new post-processing profile • Describe the purpose of a post-processing profile • Modify a post-processing profile by adding and removing module overrides • Describe common post-processing effects, such as bloom, depth of field, tonemapping, and color adjustments • Add a new local post-processing volume to the scene and edit its boundaries • Describe scenarios where a global volume or local volume would be more appropriate • Import post-processing effects into a Unity project	• Post-processing part of Unity for Animation: Road to Real-time Live Series • Post-processing **Unity Manual** • Post-processing overview	
Work with GameObjects	• Create GameObjects • Transform GameObjects • Manage GameObjects with prefabs	**Unity Learn** • Add furniture to the kid's room	• Associate Game Developer • Associate: Artist • 3D Artist

		• Make a tower of prefab blocks	
Employ basic physics for GameObjects The Unity physics system allows the artist to simulate real-world, or unrealistic physics in their scene. Implementing the correct physics allows an artist to create the required interaction and effects in their scene. **Assessment suggestion:** Have students add physics to assets to make them behave like their real-world equivalents, for example changing a sphere into a bouncing ball.	• Apply Rigidbody or Rigidbody 2D components in order to enable GameObjects to act under the control of physics • Create and configure a Physic Material to add physical properties to a GameObject • Modify the basic properties of a Rigidbody or Rigidbody 2D component in order to control how the GameObject is affected by physics, including its mass, its drag, and the Scene's gravity • Add a collider to a GameObject	**Unity Learn** • Add physical properties to 3D GameObjects • Make a bouncy ball - Unity Learn • Working with Physics in Unity **Unity Manual** • Physics • Rigidbodies • Collision	• Associate Game Developer • Associate: Artist • 3D Artist
Employ prefabs in order to manage the GameObjects in a scene or project Prefabs and Nested Prefabs let students build complex objects	• Explain how to use prefabs in a scene • Add a prefab to a project • Identify a prefab in the Project window • Edit a prefab in prefab mode	**Unity Learn** • Prefabs • Add furniture to the kid's room • Make a tower of prefab blocks	• Associate: Programmer • Associate Game Developer

with parts that have the flexibility and power of Prefabs. **Assessment suggestion:** Have students implement a Nested Prefab and understand how to move in and out of layered Prefab editors.	• Make a prefab variant • Apply or revert changes to a prefab variant • Identify when a nested prefab or prefab variant is in use • Describe the process and outcomes for changing a nested prefab or prefab variant	**Unity Manual** • Prefabs • Nested Prefabs	
Import model files and custom packages into Unity Unity provides tools to optimize the workflow between the Editor and most industry-standard DCC applications. Understanding how to make use of this workflow will allow the student to speed up the process of bridging 3D art and assets in the Unity Editor. **Assessment suggestion:** Have students export a scene or asset from the Unity Editor using the FBX exporter, update the materials or model in the third-party modeling software of choice, and then re-import the	• Import assets • Import models with materials and textures • Import and configure assets from a custom package	**Unity Learn** • Get 3D assets **Unity Manual** • Import a local asset • FBX Exporter	• User Digital Artist • Associate Game Developer

model, keeping the same orientation, position, and scale.			
Integrate external assets and tools into your prototype Unity provides tools to optimize the workflow between the Editor and most industry-standard DCC applications. Understanding how to make use of this workflow will allow the student to speed up the process of bridging 3D art and assets in the Unity Editor. **Assessment suggestion:** Have students export a scene or asset from the Unity Editor using the FBX exporter, update the materials or model in the third-party modelling software of choice, and then re-import the model, keeping the same orientation, position, and scale.	• Import a third-party character controller • Identify third-party assets and resources for a prototype • Create an asset inventory	**Unity Learn** • 3DS Max to Unity Pipeline • The Maya to Unity Pipeline • Get 3D assets **Unity Manual** • Importing assets **Unity Resources** • Introduction to game level design	• 3D Artist
Obtain assets from the Unity Asset Store GameObject assets are at the heart of every Unity scene. Understanding how to create high-quality assets will allow the	• Navigate to the Asset Store from the Unity Editor • Search and filter a search in the Asset Store	**Unity Manual** • Unity's Asset Store	• User Digital Artist • Associate: Artist • Associate Game Developer

student to create interesting scenes that meet the aesthetic requirements of the project. **Assessment suggestion:** Have students build a scene, based on a suggested theme, using assets from the Asset Store and other online repositories like - TurboSquid, - CGTrader • Poliigon, or - Poliigon	• Import an asset from the Asset Store automatically via a Unity account • Navigate to the Asset Store in a web browser • Import an asset downloaded from the Asset Store in a web browser		
Refine a prototype environment using ProBuilder	• Explain how ProBuilder can support prototype development • Create meshes using Probuilder • Configure geometry to make basic scenery for a prototype • Set a Collider for a mesh • Set a mesh as a trigger	**Unity Learn** • Building 3D Models with ProBuilder • Prototyping **Unity Manual** • About ProBuilder **Unity Resources** • Introduction to game level design	• Associate: Artist
Refine a prototype environment using Terrain	• Explain how Terrain can support prototype development • Create a new Terrain • Customize Terrain using the Paint Terrain tool	**Unity Learn** • Introduction to Terrain Editor • Working with the Terrain Toolbox • Prototyping - Unity Learn **Unity Manual**	• User Digital Artist • Associate: Artist

		• World building • Enhance your prototype with Terrain **Other Resources** • Introducing the Terrain Editor | Unite Now 2020 • How to build beautiful landscapes in Unity using Terrain Tools	
Set up a new NavMesh in a scene Unity's navigation system lets developers create AI-powered enemy agents that can intelligently move around the game world, using navigation meshes (NavMeshes) created automatically from the scene geometry. Students implementing navigation and pathfinding should have some understanding of scripting. **Assessment suggestion:** Have students bake a NavMesh allowing agents to move across the terrain, avoid objects, and chase a player avatar.	• Create a NavMesh • Create a NavMesh agent • Create a NavMesh obstacle • Create a NavMesh link • Use NavMesh Agents with other components • Build a HeightMesh for Accurate Character Placement	**Unity Learn** • NavMesh Baking - Unity Learn • NavMesh Agents - Unity Learn • Working with NavMesh Agents - Unity Learn **Unity Manual** • Create a NavMesh	

Work with 2D assets in Unity

Module introduction

In this module you'll delve into the essentials of creating 2D projects in Unity. This includes understanding the unique tools and workflows tailored for 2D environments, such as sprite manipulation, animation techniques, and efficient scene management. You'll learn how to effectively use Unity's 2D Renderer for creating visually appealing and interactive 2D experiences. The module will also guide you through the nuances of 2D physics and character control, ensuring you can create dynamic and responsive 2D games or applications. By the end of this module, you'll be equipped with the knowledge to craft engaging and polished 2D projects in Unity.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Use basic state machines and blend trees to create and manage multiple animations The Unity real-time engine provides numerous tools and plugins that allow the artist to link the Unity Editor with their modeling or texturing applications of choice. Understanding the available options and how to implement them will assist the artist in setting up an efficient workflow and technology stack. **Assessment suggestion:** Have students use Substance Designer or Substance Painter to create a workflow setup between the Unity Editor and their Substance application and create materials for at least one of their assets in a scene.	• Distinguish between transition-based and layer-based approaches to building state machines. (ProArt) • Recognize uses for sub-state machines within an animator controller. (ProArt) • Recognize uses for blend trees within an animator controller. (ProArt) • Identify approaches to working with different parameter types for animation state transitions. (ProArt) • Modify individual animations for use within a state machine. (ProArt)	**Unity Learn** • Explore State Machines	• Associate: Artist • Professional: Artist • Associate: Game Developer
Use the sprite editor to slice sprite sheets	• Identify parameters in the sprite inspector to configure sprite sheets	**Unity Learn** • Introduction to Sprite Editor and Sheets	• User: Artist • Associate: Artist • Professional: Artist

Students can use the Sprite editor to build sprites. The rigger allows sprites to be given bones and weights, which will make them easier to animate. **Assessment suggestion:** Have students create and rig a sprite, ensuring that bones and weights are placed and adjusted correctly.	according to project requirements. (ProArt) • Determine sprite editor slice parameters to isolate sprites in sheets. (ProArt) • Determine settings to manually prepare sprite sheets, including specifying positions, borders, and pivots. (ProArt) • Recognize uses of the edit outline feature in the sprite editor to customize the shape of a sprite. (ProArt)	**Unity Manual** • Sprite Editor	
Use tilemaps to create 2D environments Students can use the Sprite Editor to create tilemaps. **Assessment suggestion:** Have students create a tilemap and use it to decorate a background and a set of foreground tiles.	• Create a tilemap using the Sprite Editor • Use a tilemap on a background and a GameObject	**Unity Learn** • Introduction to Tilemaps **Unity Manual** • Tilemap	• Associate: Artist

Working with materials and shaders

Module introduction

This module provides an in-depth exploration of materials and shaders within the context of computer graphics. Through this module, you will develop a comprehensive understanding of techniques for creating and manipulating textures, surfaces, and visual effects to enhance digital projects. Whether you have prior experience in 3D art or are new to computer graphics, this module offers essential knowledge and practical skills for achieving realistic and visually compelling results.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Create a simple shader and material using Shader Graph The Shader Graph tool in the Unity real-time engine allows the	• Explain Shader Graph and its uses • Create a new shader in Shader Graph	**Unity Learn** • Make a Flag Wave with Shadergraph • Introduction to ShaderGraph	• 3D Artist • Associate: Artist

user to create custom shaders without code. Understanding this functionality will allow the artist to create special and custom effects for specific render pipelines that are optimized for the target publishing hardware without the need for shader coding knowledge. **Assessment suggestion:** Have students use Shader Graph to create a simple shader effect, like a shimmering material. The Creative Core pathway can be used as a guide for this.	• Navigate in the Shader Graph editor window • Connect commonly used Shader Graph nodes to create desired effects • Make a shader with configurable material properties • Make a material from a custom Shader Graph shader	• Get started with Shader Graph **Unity Manual** • Shader Graph **Other Resources** Create shaders and visual effects with URP	
Create and edit shaders using Shader Graph Students must understand the basics of what shaders are and how they are used to affect how the audience experiences objects in Unity. **Assessment suggestion:** Have students describe the creation of and uses for shaders, including object and environment applications.	• Create a shader using Unity Shader Graph	**Unity Learn** • Get started with Shader Graph • Introduction to ShaderGraph • Shader Graph: Multiply • Shader Graph: TIme Node **Unity Manual** • Getting started with Shader Graph • Shader Graph • Node Library	• Associate: Artist

Create materials for the URP/Lit Shader on a 3D GameObject Students will learn to use Unity's fully-featured suite of tools to create, apply, and alter textures and materials to modify the appearance of their models. **Assessment suggestion:** Have students dress models using materials and textures created in and imported into Unity, and adjusted using Unity's native tools.	• Create a new material • Organize materials as project assets • Adjust the Base Map of a material using a color • Adjust the Base Map of a material using an image • Apply the Specular and Metallic workflows to achieve desired effects • Apply alpha clipping in a material • Apply the transparent surface type to a material • Add a normal map to a material • Fix broken (magenta) materials	**Unity Learn** • Shaders and Materials **Unity Manual** • Unity - Manual: Materials and shaders	• User Digital Artist • Associate Game Developer
Decide among common shaders to use for a given project Students must understand the basics of what shaders are and how they are used to affect how the audience experiences objects in Unity. **Assessment suggestion:** Have students describe the creation of and uses for shaders, including	• Define a mesh, its characteristics, and its use in rendering a 3D GameObject • Determine the shader type for an object based on the design requirements • Explain the role of shaders in the rendering process	**Unity Learn** • Introduction to ShaderGraph • Shaders and Materials **Unity Manual** • Shader Graph • Unity - Manual: Materials and shaders	• User Digital Artist

object and environment applications.	• Explain the difference between physically-based and non-physically-based rendering, and reasons for using each • Explain the difference between a Lit and Unlit shader, and the reasons for using each • Explain vertex and fragment (pixel) shaders • Describe use cases for the Universal Render Pipeline shaders provided with Unity		
Simulate common substances with physically-based materials As computers have become more powerful and rendering technology has evolved, Physically Based Rendering (PBR) has become more widely available. PBR simulates the real-world principles of physics and light to generate realistic shadows, reflections, ambient light, and other effects of light on 3D surfaces.	• Identify the characteristics of a real-world surface to be configured in a new material • Adjust material properties to simulate a given solid substance • Given a collection of texture files, select appropriate maps to simulate a material	**Unity Learn** • Physically based shaders and rendering • Unity DCC live link with Substance Painter • Baking Texture Maps in Substance Painter - Unity Learn **Other Resources** • Adobe Substance 3D • Substance 3D plugin for Unity - Adobe Substance 3D • Substance 3D Tutorials	

Assessment suggestion: Explain the difference between physically-based and non-physically-based rendering, and reasons for using each		• Substance Forum	

Cameras and Cinemachine

Module introduction

In this module, you'll explore the power and versatility of Unity's camera systems and the Cinemachine framework. This course will guide you through the process of setting up and manipulating cameras to capture your game world effectively. You'll learn how to use Cinemachine's intelligent camera tools to create dynamic, cinematic game sequences with ease. The focus will be on understanding camera properties, behaviors, and how to leverage Cinemachine's advanced features for smooth and responsive camera movement. By mastering these tools, you'll be able to enhance the visual storytelling and immersion of your Unity projects, creating engaging and visually compelling experiences.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Configure a multiple-camera setup	• Create a split-screen effect using two cameras	**Unity Learn** • Working with cameras	• Associate: Artist

Configuring a multiple-camera setup in Unity involves setting up and coordinating multiple cameras to capture different perspectives or render specific portions of a scene, enabling complex scene rendering and gameplay mechanics. **Assessment Suggestion:** Have the learners create a Unity project with a multiple-camera setup where each camera renders a different part of a scene.		**Unity Manual** • Cameras • Cinemachine package • Split screen and multiple Unity Cameras • Cinemachine and Timeline **Other Sources** • Cinemachine • See what's new with Cinemachine 3	
Configure a single Unity Camera in a 2D or 3D scene Configuring a single Unity Camera in a 2D or 3D scene is to define the perspective, view, and rendering properties necessary to capture and display a specific portion of the scene to achieve desired visuals and gameplay interactions. **Assessment Suggestion:** Have the learner set up a Unity scene	• Capture the desired view of the Scene by controlling the position and rotation of the Main Camera • Fill in background of the Main Camera view • Control the field of view of the Main Camera by adjusting the frustum • Control the depth of view of the Main Camera by configuring the clipping planes	**Unity Learn** • Cameras • Set up your 2D game world • Camera and lighting **Unity Manual** • Cameras	• 3D Artist

with a single camera and various objects, and then task them with adjusting the camera's properties (e.g., field of view, position, depth, and target) to achieve a specific desired framing or visual effect within the scene.	• Set up an perspective camera view in a 3D Scene • Set up an orthographic camera view in a 3D Scene • Set up a camera for a specified/predetermined point of view		
Configure cameras for desired effects in a scene Configuring cameras for desired effects in a scene in Unity is to adjust camera settings and parameters to achieve specific visual outcomes and enhance the overall aesthetic and gameplay experience. **Assessment suggestion:** Have students configure a camera for a given effect.	• Add a camera component to a GameObject • Use Physical Camera properties to change the camera's view and effects • Explain occlusion culling and where it can be used • Bake occlusion culling data for a simple Scene • Configure an isometric camera in a 2D project	**Unity Learn** • Cameras • Camera and lighting • Set up your 2D game world **Unity Manual** • Cameras • Occlusion culling • Cinemachine and 2D • Cinemachine and top-down games	
Control camera views and movements procedurally using Cinemachine The Timeline tool gives users more complete control over	• Identify approaches to configuring Cinemachine cameras for use within applications. (ProArt) • Determine methods for configuring Cinemachine cameras to implement	**Unity Manual** • Cinemachine package • Cinemachine and Timeline	• 3D Artist

Cinemachine's virtual camera setup. **Assessment suggestion:** Have students use Timeline to toggle between separate virtual cameras as well as create a blend between them.	common camera set-ups such as first-person and third-person. (ProArt) • Identify methods to configuring Cinemachine cameras to target and automatically track specific components. (ProArt) • Identify methods for configuring Cinemachine camera range of movement using dolly paths. (ProArt) • Select Cinemachine camera type based on scene requirements. (ProArt)		

Animating in Unity

Module introduction

In this module, you will learn the essentials of animating in Unity. We will cover key concepts such as keyframe animation, rigging, and using the Animator component, alongside practical scripting techniques. By the end of this course, you'll be equipped to create dynamic and responsive animations for your Unity projects.

For a deep dive into the whole animation development cycle in Unity, we suggest you make use of Unity for Animation: Road to Realtime Live Series, a series of live sessions in Unity that takes a deep dive into every part of the animated storytelling process, from previz to final pixels, as well as the The definitive guide to animation in Unity ebook.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications

Animate using the 2D Animation package The 2D Animation package is a solution for effortless skeletal animation of 2D sprites, featuring in-editor rigging, bone weight painting, and Inverse Kinematics support. Assessment Suggestion: Have students install the 2D Animation package and animate a simple sprite using the available tools.	• Create bones using the 2D Animation Package's Skinning Editor • Generate a mesh for a 2D rig in the 2D Animation Package • Adjust weights on a 2D rig in the 2D Animation Package • Use a Sprite Skin component to generate a 2D rig in the 2D Animation Rigging package	**Unity Learn** • Rigging a Sprite with the 2D Animation Package • Animating a Sprite with the 2D Animation Package **Unity Manual** • 2D Sprite Shape • Animation Rigging • Introduction to 2D Animation • AI Navigation | AI Navigation | 2.0.8 **Unity Blog** • Getting Started with Unity's 2D Animation Package	
Configure animation clips imported from digital content creation (DCC) software or the Asset Store for use in a project The world around you is in constant motion. The same is true for digital worlds. A static environment tends to appear unfinished or cold and unfeeling;	• Apply imported animation clips to rigged models in Unity • Configure a humanoid rig to share animations between characters • Trim animation clips to access specific keyframed sequences within them	**Unity Learn** • Creative Core: Animation • Working with Animation Rigging • Working with Animations and Animation Curves • Retargeting and Reusing Animation **Unity Manual** • Animation	• Associate Game Developer

animation is all about creating the illusion of life. In this mission, you will learn how to create animations in the Unity Editor and how to configure animations imported from an external program. You'll apply these concepts to add animation to objects and characters in your scenes, and you'll even control when the animation gets played. **Assessment suggestion:** Have students create a looping animation using imported animated assets or create their own animated asset in the Unity Editor.	• Identify the purpose of a specified parameter of an animator controller • Describe the relationship between parameters and transitions • Define animator • Define different rig types and their uses • Describe how an avatar is used with a humanoid rig to share animation • Describe how an avatar is used with an animator controller to control animation • Define the different rig types and their uses		
Create a keyframed animation sequence using Unity's animation editor The world around you is in constant motion. The same is true for digital worlds. A static environment tends to appear unfinished or cold and unfeeling; animation is all about creating the illusion of life.	• Add keyframes to an Animation Clip • Move keyframes along the timeline in the Animation Editor • Assign an animator controller to a GameObject's Animator component	**Unity Learn** • Creative Core: Animation • Introduction to 3D animation systems • Unity for Animation: Road to Realtime Live Series • Get started with animation **Unity Manual** • Animation • Animation Rigging	• Associate: Artist • 3D Artist

In this module, you will learn how to create animations in the Unity Editor and how to configure animations imported from an external program. You'll apply these concepts to add animation to objects and characters in your scenes, and even control when the animation gets played. **Assessment suggestion:** Have students create a looping animation using imported animated assets, or create their own animated asset in the Unity Editor.	• Record a GameObject animation using Record Mode • Set up a new Animation Clip • Move an animation in a scene using a parent GameObject • Open the Animation Editor window • Define keyframes • Define tweening • Explain what a playhead does • Select the view of the Animation Editor's timeline to display seconds or frames • Set the sample rate of an animation clip • Explain how the dopesheet is used in the Animation Editor window	• Blend Trees	
Create basic 2D animations with Sprites Parameters allow students to control a number of elements about an animation, such as its speed or state.	• Recognize the process for automatically generating 2D animation from Sprite sheets (ProArt) • Identify methods within the animation controller used to customize	**Unity Manual** • Sprite editor • Introduction to 2D Animation • 2D Animation • AI Navigation	• 3D Artist

Assessment suggestion: Have students create three animation states for an animation and three parameters that can be altered at runtime with a script.	animation states for Sprites (ProArt) • Identify editor animation parameters necessary to fine-tune sprite animations (ProArt)		
Evaluate the various animation types in order to determine which one to use Unity provides internal animation tools, as well as the option to import animations from their party applications. Understanding the difference between imported and Unity-created animations will assist the user in determining the most efficient workflow. **Assessment suggestion:** Have students import an animation and create a native keyframe animation in Unity, then contrast and discuss the advantages and disadvantages of each.	• Explain the difference between animations imported into Unity and animations created within Unity • Name software products from which you can import models with animations into Unity • DIfferentiate movements created with physics from those created with animation	**Unity Learn** • Editor Essentials • Creative Core: Animation • Control animation with an Animator • AI Navigation • Introduction to 3D animation systems • Unity for Animation: Road to Realtime Live Series	• 3D Artist • Associate Game Developer
Set up a new NavMesh in a scene	• Explain backfilling with a NavMesh	**Unity Manual** • Navmesh | AI Navigation • Create a NavMesh • Create a NavMesh Agent	

Unity's navigation system lets developers create AI-powered enemy agents that can intelligently move around the game world, using navigation meshes (NavMeshes) created automatically from the scene geometry. Students implementing navigation and pathfinding should have some understanding of scripting. **Assessment suggestion:** Have students bake a NavMesh allowing agents to move across the terrain, avoid objects, and chase a player avatar.	• Explain how maximum slope is used with a NavMesh • Explain how obstacle avoidance is used with a NavMesh	• Create a Navmesh Obstacle • Create a NavMesh Link • Use NavMesh Agent with Other Components • Build a HeightMesh for Accurate Character Placement	
Use basic state machines and blend trees to create and manage multiple animations The Unity real-time engine provides numerous tools and plugins that allow the artist to link the Unity Editor with their modeling or texturing applications of choice. Understanding the available options and how to implement them will assist the artist in	• Distinguish between transition-based and layer-based approaches to building state machines. (ProArt) • Recognize uses for sub-state machines within an animator controller. (ProArt) • Recognize uses for blend trees within an animator controller. (ProArt)	**Unity Learn** • Control animation with an Animator • Explore the Animator Controller • Explore State Machines	• Associate: Artist • 3D Artist • Associate Game Developer

setting up an efficient workflow and technology stack. **Assessment suggestion:** Have students use Substance Designer or Substance Painter to create a workflow setup between the Unity Editor and their Substance application and create materials for at least one of their assets in a scene.	• Identify approaches to working with different parameter types for animation state transitions. (ProArt) • Modify individual animations for use within a state machine. (ProArt)		

Lighting in Unity

Module introduction

Welcome to the module on lighting in Unity, where we'll delve into the essential aspects of illuminating your virtual worlds. Throughout this module, we'll explore the diverse array of light types available in Unity, including directional, point, and spotlights, and how to effectively utilize them to shape your scenes. Additionally, we'll discuss the concept of light baking, a vital technique for optimizing real-time rendering performance. Understanding the pivotal role of lighting in crafting cinematic graphics, we'll guide you through the principles and practices that bring your virtual environments to life with dynamic and visually compelling illumination.

Suggested skills and learning objectives

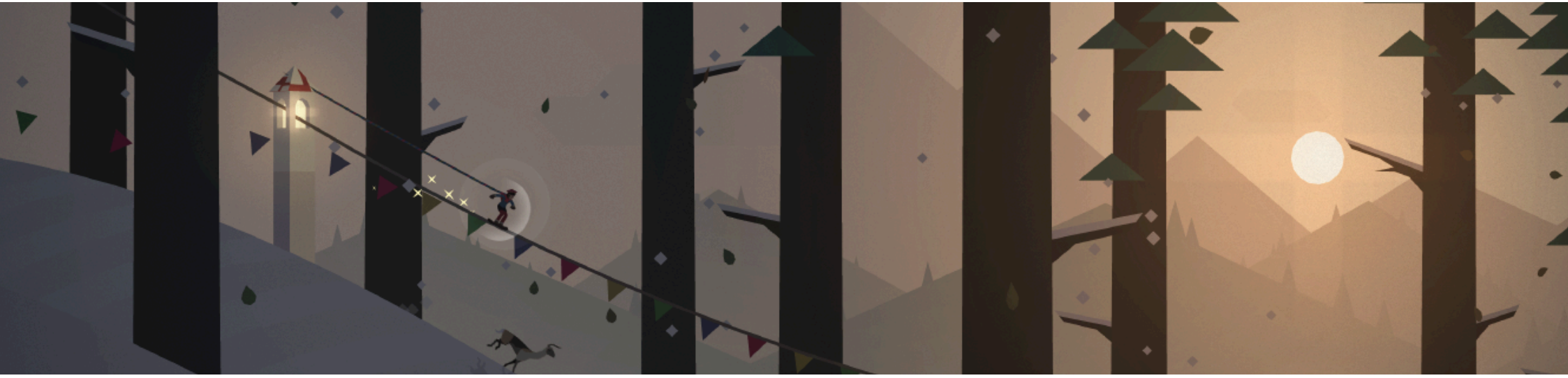
Skill and Description	Learning Objectives	Resources	Related Certifications
Configure light sources and shadows in order to functionally light a scene	• Describe the role of the Directional Light in a scene	**Unity Learn** • Lighting • Lighting in Unity	• 3D Artist • Associate Game Developer

The Unity Editor provides different light types that simulate various real-world light sources. Understanding when and where to use a specific light type will assist students in creating believable and well-designed scenes. **Assessment suggestions:** Have students use different light types and techniques to create two different aesthetics or moods in a scene.	• Configure the Directional Light in a scene to achieve common effects • Identify the differences between the different types of Light component • Configure Light components to achieve common lighting effects • Configure shadows in the Render Pipeline asset to achieve realistic effects • Add emissive materials to a scene • Check emissive materials in a diagnostic view	• Introduction to Lighting and Rendering **Unity Manual** • Lighting • Types of light **Other Sources** • 7 Ways to Optimize your Unity Project with URP	
Decide the appropriate lighting system in order to achieve common outcomes in a Universal Render Pipeline (URP) project Lighting is a complex topic, and the intricacies of lighting can make a huge difference both in how a scene is perceived and how it performs in play. **Assessment suggestion:** Have students light a scene to	• Define the term global illumination • Identify Unity's Global Illumination system for URP • Explain the main differences between real-time and baked lighting in Unity	**Unity Learn** • Bake a lightmap for your scene - Unity Learn • Configuring Light Probes **Unity Manual** • Lighting configuration workflow • Pipeline | Universal RP • Lighting in the Universal Render Pipeline • Direct and indirect lighting **Other Sources**	• Associate: Artist

resemble a famous still from a given movie, paying attention to the quality, color, and performance of the scene.		• Guide to lighting in the HDRP • The URP 3D Sample • Introduction to the Universal Render Pipeline for advanced creators	
Identify and configure lighting techniques and effects using Light settings Unity's advanced lighting options allow for much more complex lighting, creating deeper, more immersive experiences. **Assessment suggestion:** Have students use Unity's advanced lighting options to create scenes with expressive, interesting lighting.	• Interpret design requirements to determine whether to use real time or mixed lighting modes (ProArt) • Configure shadow settings, including width and bias, to achieve realistic effects • Determine the settings of the mixed lighting mode to satisfy application design requirements (ProArt) • Create lighting effects such as halos and flares • Select lighting effects to achieve stylistic results • Recognize uses of area lights to create lighting for specialized scenarios such as shaped lights and architectural visualizations. (ProArt)	**Unity Learn** • Lighting • Creating Believable Visuals • Configuring Light Probes • Types of light **Unity Manual** • Introduction to lighting • Lighting • Types of light	• Associate Game Developer

	• Determine Lightmapper type based on project needs (ProArt) • Identify the light type required for a specific effect based on project needs. (ProArt) • Configure the shape of a light to achieve a desired effect		
Light a scene in a manner that will simulate the real-world behavior of light Lighting is a complex topic, and the intricacies of lighting can make a huge difference both in how a scene is perceived and how it performs in play. **Assessment suggestion:** Have students light a scene to resemble a famous still from a given movie, paying attention to the quality, color, and performance of the scene.	• Describe the fundamentals of the behavior of light • Explain at a high level the difference between direct and indirect light • Identify light sources in an image • Explain at a high level the relationship between lighting and post-processing • Identify key considerations for lighting indoor and outdoor scenes realistically • Identify research topics and resources to develop your understanding of foundational lighting	**Unity Learn** • Lighting • Configuring Light Probes • Lighting • Types of light **Unity Manual** • Direct and indirect lighting • Lighting • Types of light • Post-Processing overview	• User Digital Artist • Associate Game Developer

	science and design principles		



User interfaces in Unity

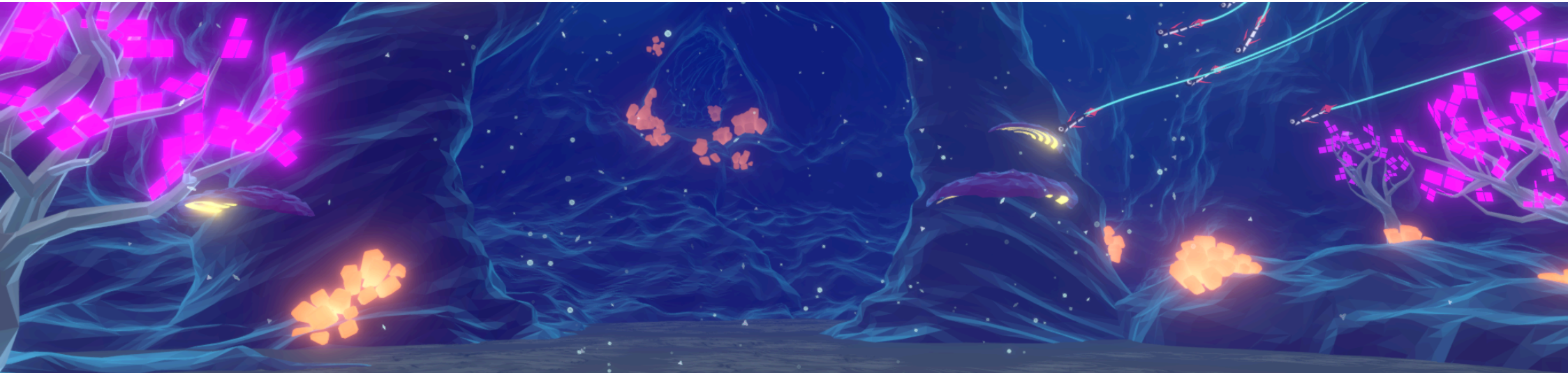
Module introduction

In this module, you will delve into the fundamentals of User Interface (UI) design in Unity using the UI Toolkit, Unity's modern system for building responsive and flexible user interfaces. UI Toolkit replaces the legacy UI system, offering a more structured and performance-optimised approach to UI development through stylesheets, hierarchies, and reusable components. You will explore how to effectively design, implement, and optimise UI elements, such as buttons, menus, and labels, within this new framework, gaining insight into how UI Toolkit enhances both workflow and user experience.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Create and configure visual UI components in a manner that will respond appropriately to	• Use a visual authoring tool to create and edit UI Toolkit assets such as UXML and USS files.	**Unity Learn** • UI Toolkit Fundamentals • UI Toolkit - First steps	

different screen sizes and resolutions UI Toolkit is a new system that will eventually be the primary system for developing UIs within Unity. The goal is for it to include all of the features from uGUI and IMGUI. However, this system is still in development. **Assessment suggestion:** Have students use UI Toolkit to design and implement a multi-level menu flow in the application.	• Structure your UI with either UXML • Style your UI with USS	• Getting started with UI toolkit **Unity Manual** • Introduction to UI Toolkit • Get started with UI Toolkit • UI Builder • Structure UI • Style UI • UI Toolkit examples	
Program scripts for interactive user interfaces Unity offers a suite of advanced UI management tools to create complex UI interactions. **Assessment suggestion:** Have students design and implement a complex menu flow in the application state.	• Map user interactions to elements • Link properties to the controls that modify their values.	**Unity Learn** • Getting started with UI toolkit • UI Toolkit in Unity 6: Crafting Custom Controls with Data Bindings **Unity Manual** • UI Toolkit examples • Control behavior with events • Data binding	



Audio in Unity

Module introduction

In this module, you'll explore the intricacies of audio in Unity, learning how to effectively integrate and manipulate sound to enhance the immersion and impact of your projects. You'll delve into Unity's audio engine, discovering techniques for adding depth and realism through spatial sound, environmental audio, and dynamic sound effects. This focused approach will equip you with the skills to create rich, engaging audio landscapes, elevating the overall experience of your Unity creations.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Configure audio in a scene to produce customized results	• Apply custom rolloffs to simulate different types of audio sources	**Unity Learn** • Audio Essentials • Audio • Beginning Audio in Unity	

	• Choose time-based or action-based methods, such as triggers or events in order to play audio clips • Add special audio effects to a scene	**Unity Manual** • Audio	
Create a plan to design audio for a real-time 3D application The Unity real-time engine allows the user to insert soundtracks and location-based sound effects into their scene. This functionality allows the artist to create immersive scenes and environments in their project. **Assessment suggestion:** Have students implement simple audio into a scene that will change based on the distance of the user from the source.	• Describe the science of audio in digital environments • Describe the primary types of audio found in real-time projects	**Unity Learn** • Essentials of real-time audio • Audio and Haptics • The Basics of the Audio Random Container **Unity Manual** • Audio • VR Audio Spatializers	• Associate Game Developer
Create realistic spatialized 3D audio effects by applying audio experience design principles Spatial audio provides a method through which to build and place audio assets so that – from the	• Explain the difference between diegetic and nondiegetic sound • Explain the role of audio in developing atmosphere	**Unity Learn** • Beginning Audio in Unity • Working with Audio Components • Audio • The Basics of the Audio Random Container	

VR user's perspective – a given sound originates from a particular position in a 3D scene. This is like surround-sound in a home theatre setup or at the cinema, and very important to presence and immersion in VR. **Assessment suggestion:** Have students set up location-based audio in a scene to enhance the immersion of the user.	• Explain the role of audio in supporting narrative and worldbuilding	**Unity Manual** • VR Audio Spatializers	
Develop 3D audio for a scene Spatial audio provides a method through which to build and place audio assets so that – from the VR user's perspective, a given sound originates from a particular position in a 3D scene. This is like surround sound in a home theatre setup or at the cinema, and very important to presence and immersion in VR. **Assessment suggestion:** Have students set up location-based audio in a scene to enhance the immersion of the user.	• Add background music to a 3D scene • Implement spatial audio in a 3D scene • Identify and procure audio assets	**Unity Learn** **Unity Manual** • Audio • VR Audio Spatializers	• Associate Game Developer

Refine existing audio in a Unity project	• Control the priority of different audio sources in a scene • Recommend audio source file formats that can be used in a given project • Recommend optimization techniques for audio, given a target platform	**Unity Manual** • Audio in Unity	
Solve accessibility challenges in an audio design	• Add subtitles to a Unity project	**Unity Learn** • Accessibility considerations for audio	

VFX in Unity

Module introduction

In this module, you will learn about Visual Effects (VFX) in Unity, focusing on the dynamic and versatile tools of particle systems and the VFX Graph. These powerful features are essential for creating stunning visual effects that can bring your Unity projects to life. You will learn how to use particle systems for simulating complex phenomena like fire, smoke, and water, as well as how to leverage the VFX Graph for more advanced, customizable effects.

When you are ready to take a deep dive into this topic, check out the free eBook, The Definitive Guide to Creating Advanced Visual Effects in Unity as well as Create shaders and visual effects with URP.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications

Build and customize a particle system Particles are a low-overhead way to create simple but impressive visual effects. **Assessment suggestion:** Have students use particle effects to both recreate environmental phenomena and highlight player achievement.	• Create a Particle System	**Unity Learn** • VFX • Introduction to Particle System • Getting started with Particle Systems **Unity Manual** • Particle effects	
Decide whether to use Unity's Particle Systems or VFX Graph in order to produce an effect in your scene Using Unity's Particle Systems or VFX Graph to produce an effect in your scene is to create visually compelling and dynamic special effects or simulations by harnessing the power of real-time particle-based visual effects. **Assessment Suggestion:** Challenge the learner to design and implement a custom visual effect, like a magical spell,	• Define the acronym VFX • Explain different applications of VFX in real-time 3D experiences, such as gameplay and environmental effects • Describe the impact that VFX can have on the level of polish in a project • Understand the differences between Unity's Particle System and VFX Graph in order to select the appropriate tool for a given use case	**Unity Learn** • VFX • Get started with VFX **Unity Manual** • Visual Effect Graph • Particle effects	• Associate: Artist • Associate Game Developer

weather simulation, or explosion, using either Unity's Particle Systems or VFX Graph, and assess their proficiency in configuring particle behaviors, textures, shaders, and interactions to achieve the desired effect within the scene.			
Interpret a simple VFX graph Unity's VFX Graph is crucial for generating complex, high-quality visual effects in a scene by manipulating particle systems, shaders, and simulations to achieve immersive and realistic results. **Assessment Suggestion:** Encourage the learner to create a Unity project where they leverage the VFX Graph to craft an intricate visual effect	• Recognize whether a particle effect has been created using the Particle System or the VFX graph • Add a new VFX graph to the scene • Explain the role of each of the four default context nodes in a VFX Graph: Spawn, Initialize Particle, Update Particle, and Output Particle • Navigate in the VFX graph editor window by using the keyboard and mouse • Make simple edits to an existing VFX Graph, such as changing the emission rate or particle lifetime	**Unity Learn** • Get started with VFX **Unity Manual** • Visual Effect Graph	
Produce environmental and burst effects by configuring Unity's Particle System object	• Set up a new Particle System in the scene	**Unity Manual** • Visual Effect Graph • Particle effects	• 3D Artist

Producing environmental and burst effects by configuring Unity's Particle System object is to create dynamic and immersive visual elements within a game or application, such as fire, smoke, rain, or explosions, by controlling the behavior and appearance of particle emitters. **Assessment Suggestion:** Have the learners create a Unity scene where they configure a Particle System to generate a specific environmental or burst effect (e.g., rain, fireworks, or a campfire), and then assess their ability to adjust parameters like particle emission rate, size, color, and movement to achieve the desired visual result in real-time.	• Play, pause, stop, and restart a particle system in Scene view by using the Particle Effect window • Explain how individual Particle Systems can be combined to create more complex effects • Locate the Main module of a Particle System • Recall the three modules enabled by default in a Particle System: Emission, Shape, and Renderer • Configure a Particle System's main properties, such as lifetime, size, and max particles by modifying the Main module • Control the location and initial direction of particles by modifying the Shape module • Control the rate and timing of particles by modifying the Emission module • Control the appearance of individual particles by		

	modifying the Renderer module • Create a single burst of particles, rather than a continuous emission over time by using the Bursts section of the Emission module • Add randomness to a Particle System by using the Random Between Two Constants feature • Change the color of a particle over its lifetime by using the gradient editor and the Color Over Lifetime module • Change the size of a particle over its lifetime by using the curve editor in the Size Over Lifetime module		

Introduction to C# in Unity

Module introduction

While it's certainly possible to create a VR experience in Unity without scripting, it will be severely restricted in functionality. C# scripting in Unity unlocks new functionality and allows you to create amazing VR experiences. In this module, you'll learn about the goals of the Unity C# Scripting Fundamentals project, including scripting basics, controlling code flow, basic GameObject manipulation, and GameObject interactions.

As a primer, we suggest students complete the first two missions in the **Junior Programmer pathway** on their own time before the course begins.

Ideally, a Unity project should feel like it's been developed by a single author, no matter how many developers actually work on it. A style guide can help unify your approach for creating a more cohesive codebase. In partnership with internal and external Unity experts, we released a new e-book, **Create a C# style guide** and **Level up your code with design patterns and SOLID**.

Write cleaner code that scales for inspiration, based on **Microsoft's comprehensive C# style**.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Analyze the principal pillars of object-oriented programming C# is an advanced scripting language with many features that enable complex functionality in Unity. Advanced skills and knowledge will give the student the freedom to create complex applications and achieve their required application goals. **Assessment suggestion:** Have students set up a version control system for their code.	• Define abstraction • Define inheritance • Define polymorphism • Define encapsulation • Explain how the pillars of OOP work together to create organized, efficient code	**Unity Learn** • Apply object-oriented principles	
Create basic application interactions with Visual Scripting The Visual Scripting module (formerly known as Bolt) is a node-based tool that allows you to create the same logic and interaction in your scene as standard C# scripting, without requiring knowledge of C#. This	• Create a player inventory using the List object type in a visual script • Detect a button press or other user action in a visual script • Play audio from a visual script • Make a visual script that changes a GameObject's properties	**Unity Learn** • Visual Scripting application **Unity Manual** • Basic concepts of Visual Scripting • Developing game flow using script graphs	

is a useful approach if you are not familiar with coding but still want to add additional functionality to your scenes. **Assessment suggestion:** Have students can work through and complete the visual scripting course on Unity Learn - Visual Scripting application: Clive the Cat's 'Visual Crypting'		• Developing logic transitions using state graphs • Developer's guide and references • Basic concepts in Visual Scripting **Unity Resources** • Visual scripting	
Apply events in visual scripts The Visual Scripting module (formerly known as Bolt) is a node-based tool that allows you to create the same logic and interaction in your scene as standard C# scripting, without requiring knowledge of C#. This is a useful approach if you are not familiar with coding but still want to add additional functionality to your scenes. **Assessment suggestion:** Have students can work through and complete the visual scripting course on Unity Learn - Visual	• Add a new custom event trigger to a visual script • Construct a visual script that responds to a custom event • Pass any number of arguments from one script to another by way of a custom event	**Unity Learn** • Visual Scripting application **Unity Manual** • Basic concepts of Visual Scripting • Developing game flow using script graphs • Developing logic transitions using state graphs • Developer's guide and references • Basic concepts in Visual Scripting **Unity Resources** • Unity Visual Scripting	

Scripting application: Clive the Cat's 'Visual Crypting'			
Apply variables in visual scripts The Visual Scripting module (formerly known as Bolt) is a node-based tool that allows you to create the same logic and interaction in your scene as standard C# scripting, without requiring knowledge of C#. This is a useful approach if you are not familiar with coding but still want to add additional functionality to your scenes. **Assessment suggestion:** Have students can work through and complete the visual scripting course on Unity Learn - Visual Scripting application: Clive the Cat's 'Visual Crypting'	• Create Graph, Object, and Scene variables and explain their uses • Add Get Variable nodes to a Graph using the Blackboard • Make variables available to be changed in the Inspector window • Troubleshoot adjusting variable values in Scene and Game views • Explain the Scene Variables object that appears in the Hierarchy of projects with Visual Scripts	**Unity Learn** • Visual Scripting application **Unity Manual** • Basic concepts of Visual Scripting • Developing game flow using script graphs • Developing logic transitions using state graphs • Developer's guide and references • Basic concepts in Visual Scripting **Unity Resources** • Visual scripting	
Construct a visual script with basic code flow and logic The Visual Scripting module (formerly known as Bolt) is a node-based tool that allows you to create the same logic and interaction in your scene as	• Apply Boolean logic and conditional branching in visual scripts • Use the switch statement in visual scripts • Make mathematical calculations in visual scripts	**Unity Learn** • Visual Scripting application **Unity Manual** • Basic concepts of Visual Scripting	

standard C# scripting, without requiring knowledge of C#. This is a useful approach if you are not familiar with coding but still want to add additional functionality to your scenes. **Assessment suggestion:** Have students can work through and complete the visual scripting course on Unity Learn - Visual Scripting application: Clive the Cat's 'Visual Crypting'	• Detect keyboard input in a visual script • Use and interpret common object types in visual scripts • Identify essential programming structures in order to comprehend a visual script	• Developing game flow using script graphs • Developing logic transitions using state graphs • Developer's guide and references • Basic concepts in Visual Scripting **Unity Resources** • Visual scripting	
Control the execution of code with common logic structures As a rule, code will flow in a linear way. Operators and loops allow the user to stop and change the flow of code based on conditions. **Assessment suggestion:** Have students adjust the color script from above, but alter it to make the color loop through different values assigned to an array.	• Use if and if-else statements in code • Control the execution of code by using logical operators such as AND and OR in conditional statements • Control how many times certain lines of code run by using for loops, foreach loops, and while loops • Control the order and timing of executed code by using coroutines	**Unity Learn** • IF Statements • Loops • Switch Statements • Arrays • Enumerations • Implement data persistence between scenes • Implement data persistence between sessions	• Associate: Programmer • Associate Game Developer

	• Control the execution of code by using switch statements • Modify the values of numeric variables by using mathematical operators		
Create a GameObject component with a script Unity applications revolve around the GameObject. Accessing the GameObject via script at runtime is an essential skill for game coding and will give the student the ability to manipulate the GameObjects based on conditions and user input.	• Explain the relationship between scripts and components • Make a new script component • Open the IDE from the Unity Editor • Explain the purpose of the default code generated within a newly created C# script • Apply tags or layers to GameObjects in order to identify specific objects from within a script • Add a script component to a GameObject • Change a variable's accessibility in the Inspector by editing its access modifier to public or private	**Unity Learn** • GetComponent • Translate and rotate • GetButton and GetKey • Collision decisions • Instantiate • Destroy **Unity Manual** • Instantiating Prefabs at runtime	• Associate: Artist • Associate: Programmer • Associate Game Developer

	• Print debug messages to the console by calling the Debug.Log method		
Diagnose and fix common compilation errors Very few people can write errorless code on the first try. Understanding how to debug your code will allow you to efficiently search for and fix errors in your scripts.	• Locate a bug in code that produces a compilation error • Recommend the fix for a compilation error • Recognize when a new namespace needs to be imported	**Unity Learn** • Introduction to the Console window	• Associate: Programmer
Employ a State Machine in a visual script The Visual Scripting module (formerly known as Bolt) is a node-based tool that allows you to create the same logic and interaction in your scene as standard C# scripting, without requiring knowledge of C#. This is a useful approach if you are not familiar with coding but still want to add additional functionality to your scenes. **Assessment suggestion:** Have students can work through and complete the visual scripting	• Distinguish a State Graph from a Script Graph • Build a new State Graph • Build Script Graphs for the states in a State Machine • Navigate among the various scripts in a State Machine • Devise and configure transitions in a State Graph • Interpret an existing complex visual script • Adjust an existing Script Graph for use in a State Machine	**Unity Learn** • Visual Scripting application **Unity Manual** • Basic concepts of Visual Scripting • Developing game flow using script graphs • Developing logic transitions using state graphs • Developer's guide and references • Basic concepts in Visual Scripting **Unity Resources**	

course on Unity Learn - Visual Scripting application: Clive the Cat's 'Visual Crypting'		• Visual scripting	
Interpret simple code within a code base C# scripts allow you to create and extend custom functionality and properties on a GameObject. A solid understanding of C# script anatomy will give you more freedom when creating new applications and enable you to create custom functionality. **Assessment suggestion:** Have students create a simple script and apply it to a GameObject. The script could be used to print the current material color on the object to the log.	• Identify the purpose of common methods found in MonoBehaviours such as Start() and Update() • Define the major features of a script such as namespaces, classes, variables, and methods • Identify essential programming structures in order to comprehend simple code • Choose the appropriate data types for a specific situation including but not limited to floats, bools, and strings • Distinguish object-oriented code from data-oriented code • Distinguish an ECS (Entity Component System) class from any other type of class, given a code block containing a class definition	**Unity Learn** • Get Started with Visual Studio and Unity • Essentials of Programming in Unity • Scripts as behavior components • Beginner scripting **Unity Manual** • Visual Studio C# integration • Creating and using scripts • Creating and Using Scripts	• Associate Game Developer

	• Explain the Vector2 data type • Recognize naming conventions conforming to Unity standards, given a set of code blocks		
Manage visual scripts in a project The Visual Scripting module (formerly known as Bolt) is a node-based tool that allows you to create the same logic and interaction in your scene as standard C# scripting, without requiring knowledge of C#. This is a useful approach if you are not familiar with coding but still want to add additional functionality to your scenes. **Assessment suggestion:** Have students can work through and complete the visual scripting course on Unity Learn - Visual Scripting application: Clive the Cat's 'Visual Crypting'	• Group nodes in a visual script • Add titles and comments to a visual script using groups • Create and edit a subgraph that you can call from other visual scripts • Specify the inputs and outputs to a subgraph in the Graph Inspector	**Unity Learn** • Visual Scripting application **Unity Manual** • Basic concepts of Visual Scripting • Developing game flow using script graphs • Developing logic transitions using state graphs • Developer's guide and references • Basic concepts in Visual Scripting **Unity Resources** • Visual scripting	
Program efficient, organized, and comprehensible scripts by correctly implementing the	• Organize classes so that each has a single purpose, in order to	**Unity Learn** • ECS survival guide	

principles of object-oriented programming The Visual Scripting module (formerly known as Bolt) is a node-based visual scripting module that allows the user to create the same logic and interaction in their scene as standard C# scripting without requiring knowledge of the C# language. This is a useful approach for users who are not familiar with coding but still want to add additional functionality to their scenes. Assessment suggestion: Have students work through and complete the visual scripting course on Unity Learn - Visual Scripting application: Clive the Cat's 'Visual Crypting'	enable easier readability and debugging • Add new functionality to non-editable classes by applying extension methods • Organize and prevent conflicts between scripts by using namespaces • Use events to relay a GameObject's status changes to other objects in the application	• Principles of object-oriented programming • Introduction to ScriptableObjects **Unity Manual** • Unity Manual: ScriptableObject	
Simplify code and make it reusable by correctly implementing the principles of inheritance and polymorphism C# is an advanced scripting language with many features that	• Explain how abstraction is used to expose only necessary script components • Explain how inheritance is used to share	**Unity Learn** • Principles of object-oriented programming	• Associate: Programmer

enable complex functionality in Unity. Advanced skills and knowledge will give the student the freedom to create complex applications and achieve their required application goals.	functionality between a parent and child class • Define the relationship between a parent and child class, including what a child class can and cannot do with respect to its parent class • Recognize opportunities where inheritance could be used to simplify code • Describe how polymorphism can be applied at compile time (method overloads) and run time (method overrides) • Explain how polymorphism is used to modify parent class functionality in a child class • Explain how encapsulation is used to write code that can only be used as intended by the programmer • Recommend a high-level system architecture for a given project		

Use appropriate data types for a specific situation Variables allow the user to store data in the code. Understanding how this works and how to implement it will give the user the ability to process data and access GameObjects in the script. **Assessment suggestion:** Have students create a simple script and apply it to a GameObject. The scripts could be used to print the current material color assigned to the object to the debug log, and change the material to a new color as specified in a public variable.	• Initialize variables of a given data type, including ints, floats, doubles, bools, strings, arrays, lists, and dictionaries • Select the correct data type for a variable in a given situation • Select appropriate variable modifiers including public, private, static, protected, and const • Choose the appropriate commonly used data structures for a specific situation including but not limited to lists, arrays, and dictionaries	**Unity Learn** • Variables and Functions **Unity Manual** • Variables and the Inspector	• Associate: Programmer • Associate Game Developer

Unity Gaming Services

Module introduction

Unity Gaming Services is an end-to-end platform that is designed to help you build, engage, and grow your game.
These services allow you to take your game to the next level without having to worry about maintaining or scaling your back-end infrastructure and simplify many game development tasks and challenges.
UGS support your entire development lifecycle and can be used to build your foundation, engage your players, and grow your game.

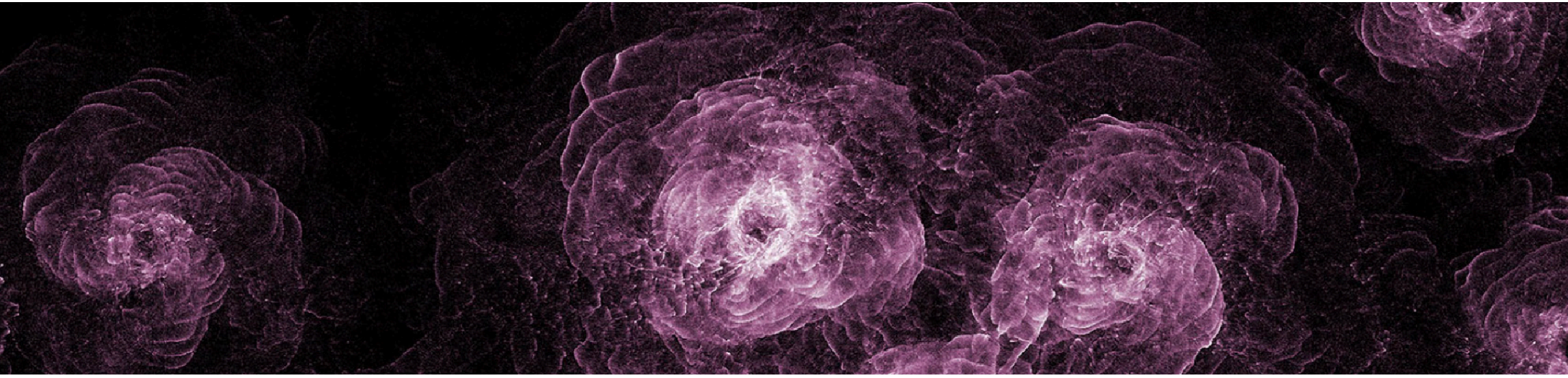
Examples include:
-Add multiplayer and social features to your game.
-Use server-side game logic to ensure a level playing field.
-Enable your players to access their game data across various gaming platforms.
-Run A/B tests and measure gameplay data from various services to inform design decisions.
-Deliver fresh content without updating your app.
-Run scheduled events and provide varied content to your game during those events.
-Engage players with fun, progressive reward and loyalty systems.

Read more about Unity Gaming Services

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Create a multiplayer game using Unity services (CFW) Unity Gaming Services (UGS) provides a host of services to assist you in creating multiplayer functionality in your game without having to worry about building and maintaining servers and related online products. **Assessment suggestion** Have students register their game with UGS and implement basic multiplayer functions.	• Set up multiplayer over internet functionality for a Unity game using Unity Gaming Services (CFW) • Set up local multiplayer functionality for a Unity game using Unity Gaming Services (CFW)	**Unity Learn** • Unity 6: Multiplayer Creation • **Unity Manual** • Multiplay • Matchmaker • Vivox Unity SDK • Friends **Unity Blog** • Master multiplayer **Other Resources** • Netcode for GameObjects • Boss Room • Lobby • Relay • Vivox voice chatting • How to set up Matchmaker • VALORANT: A Unity case study	

Set up backend services for a game using Unity services (CFW) Unity Gaming Services (UGS) provides a host of services to assist you in building and growing your user base, as well as increasing engagement for user retention. **Assessment suggestion: Have** students register their game with UGS and implement basic engagement tools.	• Set up backend services to manage and improve player retention (CFW)	**Unity Manual** • Unity Analytics • Unity Authentication • Cloud Code • Unity Cloud Content delivery • Unity Cloud Diagnostics • Cloud Save • Economy • Unity Game Overrides • Leaderboards • Push Notifications • User Generated Content	



Optimization and publishing

Module introduction

In this module, you will learn to balance aesthetics and performance in Unity by analyzing the impact of art assets and lighting. This includes understanding how poly count, particles, visual effects, and shadows affect performance. You'll create and deploy basic builds, implement Level of Detail (LOD) groups and objects to optimize scenes, and use mobile SDKs for testing and publishing applications. The module emphasizes optimizing application performance for smooth framerates, ensuring an immersive and responsive experience. You will also gain expertise in optimizing scene art assets and lighting for specific build targets, which is crucial for maintaining high performance without compromising visual quality.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Analyze the impact of art assets and lighting on performance	• Recognize the effects that Rigidbody and Collider	**Unity Learn** • Optimization	

(polycount, particles, visual effects, lighting, and shadows) Analyzing the impact on performance of factors such as poly count, particles, visual effects, lighting, and shadows involves assessing how these elements affect the frame rate and overall performance of a Unity project. **Assessment Suggestion:** Provide the learner with a Unity scene featuring various performance-intensive elements, and have them systematically identify, measure, and address performance issues related to factors like poly count, particles, visual effects, lighting, and shadows to optimize the scene's performance while maintaining an acceptable visual quality.	components have on performance • Set up the Unity Profiler to identify elements that cause performance impact • Apply Unity's Stats window in order to investigate performance issues caused by assets		
Create and deploy a basic build of a project Unity provides project settings and analysis tools that allow your application to target different	• Add the required modules for a basic build to the current Unity install • Adjust Build Settings to create a basic build	**Unity Learn** • Publish your project **Unity Manual** • Publishing builds	• Associate: Programmer

hardware and software platforms. Knowing which templates to use and which packages to install will allow a student to create optimized applications. **Assessment suggestion:** Have students identify the target build platform and adjust the build settings to produce the optimized build outcome.	• Deploy a build as a Unity Learn submission		
Create LOD groups and objects to optimize a scene Levels of Detail are used to render objects at a distance as an optimizing tool. **Assessment suggestion:** Have students create and rig Levels of Detail on complex objects to optimize performance.	• Recognize processes for creating LOD groups • Determine adjustments needed to transition phases for LOD objects to satisfy design requirements • Recognize uses of different fade modes and their application to an LOD group • Interpret design requirements to identify needs for LOD groups	**Unity Manual** • Level of Detail (LOD) for meshes	• Associate: Artist • 3D Artist • Associate Game Developer
Employ mobile SDKs to test and publish applications.	• Deploy a mobile application to an Android device	**Unity Learn** • Publish your project • Publish to Android • Publish to iOS	

Employing mobile SDKs to test and publish applications involves utilizing software development kits (SDKs) provided by mobile platform providers to build, test, and deploy mobile applications on specific platforms like iOS or Android. **Assessment Suggestion:** Require the learner to develop a mobile application (either for iOS or Android) using a chosen mobile SDK, and assess their ability to successfully build, test, and publish the app to an app store, ensuring it meets platform-specific requirements and guidelines.	• Deploy a mobile application to an iOS device • Build an application to WebGL or a personal computer	• Create and publish WebGL builds **Other Resources** • Developing extended reality apps for Horizon OS in Unity	
Optimize application performance to achieve smooth framerates in order to ensure an immersive and responsive experience Optimizing application performance to achieve smooth framerates in order to ensure an immersive and responsive experience involves fine-tuning	• Recall approximate frame rate performance targets for popular head-mounted displays • Select appropriate profiling tools to identify the sources of performance problems	**Unity Learn** • Optimization • Introduction/Optimization and Profiling • Optimizing your VR/AR Experiences • Lighting optimization with Precomputed Realtime GI	• Associate: Programmer

various aspects of a Unity project, such as rendering settings, asset management, and scripting, to maintain consistent and high frame rates during gameplay. **Assessment Suggestion:** Challenge the learner to take an existing Unity project with performance issues and task them with identifying and implementing optimizations across different areas (e.g., rendering, asset management, scripting) to achieve and maintain a stable and smooth frame rate while ensuring the game or application remains immersive and responsive			
Optimize lighting for performance Optimizing lighting for performance in Unity entails adjusting lighting settings, baking techniques, and shader choices to enhance rendering efficiency while preserving the desired visual quality.	• Lighting Optimization • Configure baked lighting to improve performance at runtime • Explain how to use light baking, reflection probes, and other light optimization techniques to increase performance	**Unity Learn** • Creative Core: Lighting • Lighting optimization with Precomputed Realtime GI	

Assessment Suggestion: Present the learner with a Unity project containing a scene with complex lighting setups, and ask them to demonstrate their ability to analyze and optimize the lighting elements to achieve improved performance without compromising the scene's overall visual appeal and realism.			
Optimize scene art assets for a particular build target The Unity Asset Store provides almost any conceivable assets you may want for your application, but in some cases creating your own assets may be the only option. Students who know how to use external Digital Content Creation (DCC) applications like Blender, Maya or 3ds Max can use their skills to create more advanced content for their own projects. **Assessment suggestion:** Have students create a product model	• Select appropriate profiling tools to identify the sources of performance problems • Decrease polycount of assets to optimize graphics	**Unity Learn** • Introduction/Optimization and Profiling • Working with the Stats Window • Working with the Frame Debugger **Unity Manual** • Working with the Frame Debugger • The Rendering Statistics window • Frame Debugger **Other Resources** • Performance Optimization Tips Webinar	• Associate: Artist

in a 3rd party modeling application and optimize it for use in an Augmented reality application using the tools provided by			

Professional skills

Module introduction

Professional skills are some of the most broadly applicable and easily transferable of the skills that are highlighted in the curricular framework. The learning objectives here focus on the soft skills students should have to secure a position in the industry and for ongoing growth and success as part of a team.

This module prepares students for a new career move by introducing them to the specific roles available to them in the industry, as well as the importance of showcasing their work and skills through the creation of compelling portfolios that present them in the best light possible. Students are also introduced to different iterative design approaches and the fundamentals of project management.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Create a portfolio for a job in real-time development	• Describe the goals, purposes, and uses of a portfolio	**Unity Learn** • Introduction to portfolios	

To successfully begin a career journey in their chosen industry, students should take an active role in choosing, achieving, and demonstrating competency in their learning goals and using that knowledge to prepare for work. **Assessment suggestion:** Have students write a short description of a specific role or set of roles in a game studio, explaining the skills required to complete the role successfully, the kinds of duties usually associated with the role, and the expectations that the role requires of applicants.	• Describe various types of portfolios • Describe tools for building a portfolio • Explain what goes into a professional portfolio • Plan a portfolio by using a flowchart • Organize content in a portfolio		
Lead projects in the real-time development cycle In the industry, successful teams use various technologies within a design process to identify and solve problems by creating new, practical, or imaginative solutions.	• Explain how downloaded AssetBundles and content catalogs are cached • Advise clients with contextual information to make the technology more understandable to them • Solve problems to address client needs with efficiency and creativity	**Unity Learn** • Roles and careers for real-time creators • Career research and preparation • Develop your learning plan • Job preparation	

Assessment suggestion: Describe and enact the steps of iterative design: identifying a problem, researching the context, enacting a solution, and iterating on the solution.			
Manage projects in the real-time development cycle In the industry, successful teams use various technologies within a design process to identify and solve problems by creating new, practical, or imaginative solutions. **Assessment suggestion:** Describe and enact the steps of iterative design: identifying a problem, researching the context, enacting a solution, and iterating on the solution.	• Explain the importance of time management in the project management process • Explain the roles of communication and professionalism in the project management process • Organize a QA testing plan for a project • Explain the reasons to conduct a retrospective after a project is completed	**Unity Learn** • Introduction to project management and teamwork • Introduction to user feedback and testing • The real-time production cycle	
Plan projects in the real-time development cycle In the industry, successful teams use a variety of technologies within a design process to identify and solve problems by	• Explain the importance of defining purpose, goal, and audience • Describe the structure and content of design documents • Explain the uses of a project charter	**Unity Learn** • Introduction to real-time 3D experience design • Introduction to user feedback and testing • The real-time production cycle	

creating new, useful or imaginative solutions. **Assessment suggestion:** Have students describe and enact the steps of iterative design, identifying a problem, researching the context, enacting a solution, and then iterating on the solution.	• Organize project tasks based on production roles • Investigate appropriate applications for project management • Explain how a design document or project brief is used in a project	• Introduction to project management and teamwork	

Unity AI

AI can help you to be more productive while staying fully in control of your vision. It offers the possibility of in-game features and capabilities that couldn't be built otherwise, potentially revolutionizing player experiences by embedding AI models in the runtime so content reacts and responds to players and users in new ways.

We're harnessing the power of AI to drive innovation, accelerate content creation, and increase your productivity across games, entertainment, and industrial use cases. We've been building a suite of AI tools that promise to accelerate creation time and complement your workflows by finding information and generating draft assets as quickly as typing in a text prompt or scribbling a sketch. From there, you could integrate work with familiar tooling to revise and edit the assets you need at a speed that's unimaginable with today's workflows.

Suggested skills and learning objectives

Skill and Description	Learning Objectives	Resources	Related Certifications
Set up your project with Unity AI Set up your Unity project and prepare to use Unity AI tools in your prototyping workflow.	• Set up your Unity project • Use Unity AI tools in your prototyping workflow.	**Unity Learn** • Set up your project with Unity AI	

Assessment suggestion: Have students set up a new Unity project and configure it for use with Unity AI tools, demonstrating their readiness to integrate AI-assisted features in a prototyping workflow.			
Use Assistant to set up your scene Use Unity's built-in AI Assistant to block out your scene through natural language prompts. **Assessment suggestion:** Have students use Unity's built-in AI Assistant to block out a basic scene by entering natural language prompts, demonstrating their ability to translate design intentions into functional scene layouts using AI tools.	• Block out a scene in Unity using natural language prompts with the built-in AI Assistant.	**Unity Learn** • Use Assistant to set up your scene	
Use Animation Generator to bring your character to life. Use Animation Generator to bring your character to life	• Animate a character using the Animation Generator to bring it to life within a Unity project.	**Unity Learn** • Use Animation Generator to bring your character to life	

Use Material and Texture Generators to make the environment Customize the look of your scene by generating a background texture and a ground material. **Assessment suggestion:** Have students use the Animation Generator tool to animate a character model, demonstrating how AI-generated animations can be applied to bring characters to life within a Unity project.	• Customise the visual style of your scene by generating a background texture and ground material.	**Unity Learn** • Use Material and Texture Generators to make the environment	
Use Sprite Generator to make an avatar icon Use the Sprite Generator to create a character avatar through natural language prompts and image references. **Assessment suggestion:** Have students use the Sprite Generator to create a character avatar by combining natural language prompts with image	• Create a character avatar using the Sprite Generator with natural language prompts and image references.	**Unity Learn** • Use Sprite Generator to make an avatar icon	

references, demonstrating how to generate and refine 2D assets using AI-assisted workflows.			
Use Sound Generator for background audio Use the Sound Generator to create custom sound effects, which can play at random intervals in the background to make your scene feel more immersive. **Assessment suggestion:** Have students use the Sound Generator to create custom ambient sound effects and implement them to play at random intervals in the background, demonstrating their ability to enhance scene immersion through procedural audio design.	• Generate custom sound effects with the Sound Generator and implement them to play at random intervals for enhanced scene immersion.	Use Sound Generator for background audio - Unity Learn	