

Unity Jr. Programmer Pathway



Standards Alignment

International Society for Technology in Education (ISTE)

From the [ISTE Standards webpage](#): The ISTE Standards are a framework for students, educators, administrators, coaches, and computer science educators to rethink education and create innovative learning environments.



Yes- ✓, No- x, Partial- ●

| Domain | # | Standard | |
|------------------------------------|--------|--|---|
| 1 Empowered learner | 1 | Students articulate and set personal learning goals, develop strategies leveraging technology to achieve them, and reflect on the learning process itself to improve learning outcomes. | ● |
| | 1 c | Students use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways. | ✓ |
| | 1 d | Students understand the fundamental concepts of technology operations, demonstrate the ability to choose, use, and troubleshoot current technologies, and are able to transfer their knowledge to explore emerging technologies. | ✓ |
| 3 Knowledge constructor | 3 b | Students evaluate the accuracy, perspective, credibility, and relevance of information, media, data, or other resources. | ✓ |
| | 3 c | Students curate information from digital resources using a variety of tools and methods to create collections of artifacts that demonstrate meaningful connections or conclusions. | ● |
| | 3 d | Students build knowledge by actively exploring real-world issues and problems, developing ideas and theories, and pursuing answers and solutions. | ✓ |
| 4 Innovative designer | 4 a | Students know and use a deliberate design process for generating ideas, testing theories, creating innovative artifacts, or solving authentic problems. | ✓ |
| | 4 b | Students select and use digital tools to plan and manage a design process that considers design constraints and calculated risks. | ✓ |

| | | | |
|--------------------------------|----|--|---|
| | 4c | Students develop, test, and refine prototypes as part of a cyclical design process. | ✓ |
| | 4d | Students exhibit a tolerance for ambiguity, perseverance, and the capacity to work with open-ended problems. | ✓ |
| 5 Computational thinker | 5a | Students formulate problem definitions suited for technology-assisted methods such as data analysis, abstract models, and algorithmic thinking in exploring and finding solutions. | ● |
| | 5c | Students break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving. | ● |
| 6 Creative communicator | 6b | Students create original works or responsibly repurpose or remix digital resources into new creations. | ✓ |
| 7 Global collaborator | 7b | Students use collaborative technologies to work with others, including peers, experts, or community members, to examine issues and problems from multiple viewpoints. | ● |

Unity Certified User: Programmer

Future creators, start here on your path to a career within the real-time 3D ecosystem. Test your foundational Unity and C# programming skills, and tell the world that you're ready to create games and apps in Unity.



Yes- ✓, No- x, Partial- ●

| Domain | | | |
|---|--|---|---|
| Debugging, problem-solving, and interpreting the API | Given an example of a debug log message, create the code that created the log message. | Unit 1 - Player control | ✓ |
| | Given a code clip and its associated error message(s), determine which object(s) is(are) null. | Profile code to identify issues | ✓ |
| | Given a specific programming task requiring the use of a particular class in the API, determine the appropriate method and/or properties, arguments, or other syntax to use. | Unit 4 - Gameplay Mechanics Unit 3 - Sound and effects | ✓ |

| | | | |
|------------------------|--|---|---|
| Creating code | Indicate when and how to initialize and use variables including but not limited to the appropriate use of all variable modifiers and data collections such as Arrays, Lists, and Dictionaries. | Unit 1 - Player control Unit 3 - Sound and effects Unit 4 - Gameplay Mechanics | 0 |
| | Given a list of keywords and syntax elements, construct a viable Function declaration. | Unit 1 - Player Control - Unity Learn Unit 2 - Introduction - Unity Learn Unit 4 - Gameplay Mechanics Unit 3 - Sound and effects | ✓ |
| | Given a code clip and a description of its desired result, identify the appropriate function to control or trigger a state including but not limited to the Animator Controller. | Unit 1 - Player Control - Unity Learn Unit 2 - Introduction - Unity Learn Unit 4 - Gameplay Mechanics Unit 3 - Sound and effects | ✓ |
| | Given a scenario where a specific type of input is required and the building blocks needed are provided, construct the necessary input listener including but not limited to the keyboard and touch input. | Unit 1 - Player Control Unit 4 - Gameplay Mechanics Unit 3 - Sound and effects | ✓ |
| | Demonstrate when and/or how to use the various logic and flow control operators used in C# and Unity. | Unit 2 - Introduction Unit 4 - Gameplay Mechanics Unit 3 - Sound and effects | ✓ |
| | Given a scenario, identify appropriate actions to take when a UI element reports a change. | Unit 4 - Gameplay Mechanics Unit 3 - Sound and effects | ✓ |
| Evaluating code | Given a scenario about the need to manage an event function, determine the appropriate action to take including but not limited to the keyboard and touch input | Profile code to identify issues | ✓ |
| | Given a code clip that produces an error because of a variable whose data type is declared incorrectly, identify the error | Profile code to identify issues | ✓ |
| | Given a code clip that produces an error because a function or variable is declared or used incorrectly (public/private mismatch), identify | Profile code to identify issues | ✓ |

| | | | |
|---------------------------------|---|--|---|
| | the error including but not limited to the use of Animation events | | |
| | Given a code clip containing a class definition, distinguish whether the class is an ECS class or some other type of class. | Profile code to identify issues | ● |
| | Given a set of code clips, recognize the clip that uses naming conventions that observe Unity naming standards | Profile code to identify issues | ✓ |
| | Given a code clip (or a set of code clips), recognize the comments that accurately describe what the code is doing. | Profile code to identify issues | ✓ |
| Navigating the Interface | Describe the purpose, features, and functions of the various Unity IDE windows. | | ● |
| | Demonstrate how to change the default scripting IDE. | | x |
| | Given a scenario that includes the following, then create a functional state machine. <ol style="list-style-type: none"> 1. a limited portion of a gaming scenario 2. a set of animation clips 3. a list of property settings | Unit 2 - Introduction Unit 3 - Sound and effects Unit 4 - Gameplay mechanics | ✓ |
| | Create and program a function state machine within the Unity Animator Controller including but not limited to the use of Animator functions syntax | | x |

Unity Certified Associate: Programmer

Demonstrate core skills and competencies across programming, UI, debugging and asset management to help you obtain your first professional programming role with Unity. [Unity Certified Associate: Programmer](#)



Yes- ✓, No- x, Partial- ◐

| Domain | | | |
|-------------------|---|---|---|
| Unity Programming | Evaluate code for integration into an existing system created/architected by a lead | <ul style="list-style-type: none"> Profile code to identify issues | ◐ |
| | Make decisions required to prototype new concepts | <ul style="list-style-type: none"> Abstraction in object-oriented programming Inheritance and polymorphism in object-oriented programming Encapsulation in object-oriented programming | ◐ |
| | Determine code that would accomplish a specified interaction or programming logic | <ul style="list-style-type: none"> Unit 4 - Gameplay Mechanics | ✓ |
| | Decide how to implement scene management and transitions | <ul style="list-style-type: none"> Create a scene flow Implement data persistence between scenes Implement data persistence between sessions | ✓ |
| | Apply basic data persistence within a runtime session | <ul style="list-style-type: none"> Create a scene flow Implement data persistence between scenes Implement data persistence between sessions | ✓ |
| | Given a situation, determine proper usage and application of the Unity API | <ul style="list-style-type: none"> Abstraction in object-oriented programming Inheritance and polymorphism in object-oriented programming Encapsulation in object-oriented programming | ✓ |

| | | | |
|------------------|--|---|---|
| | Decide the appropriate properties, scripts, and components of GameObjects for required tasks | <ul style="list-style-type: none"> • Unit 4 - Gameplay Mechanics | ✓ |
| | Choose the appropriate data structures for a specific situation | <ul style="list-style-type: none"> • Abstraction in object-oriented programming • Inheritance and polymorphism in object-oriented programming • Encapsulation in object-oriented programming | ● |
| | Choose the appropriate data types for a specific situation | <ul style="list-style-type: none"> • Unit 2 - Introduction - Unity Learn • Unit 3 - Sound and effects • Unit 4 - Gameplay Mechanics | ● |
| | Identify the steps required to deploy a basic build | <ul style="list-style-type: none"> • Unit 2 - Introduction - Unity Learn • Unit 3 - Sound and effects • Unit 4 - Gameplay Mechanics | ● |
| | Apply concepts required to write code with basic inheritance and interfaces | <ul style="list-style-type: none"> • Abstraction in object-oriented programming • Inheritance and polymorphism in object-oriented programming • Encapsulation in object-oriented programming | ● |
| UI | Apply concepts required to lay out a user interface | <ul style="list-style-type: none"> • Unit 5 - User Interface | ✓ |
| | Identify the process required to bind data on the UI to application data | <ul style="list-style-type: none"> • Unit 5 - User Interfac | ✓ |
| | Decide how to capture and respond to UI input using the Event System | <ul style="list-style-type: none"> • Unit 2 - Introduction - Unity Learn • Unit 3 - Sound and effects • Unit 4 - Gameplay Mechanics | ✓ |
| | Decide how to capture and respond to UI input using the Event System | <ul style="list-style-type: none"> • Unit 4 - Gameplay Mechanics | ✓ |
| Debugging | Troubleshoot code that fails to perform as expected | <ul style="list-style-type: none"> • Profile code to identify issues | ✓ |

| | | | |
|-------------------------|---|---|---|
| | Troubleshoot common compilation bugs | <ul style="list-style-type: none"> • Profile code to identify issues | ✓ |
| | Troubleshoot runtime exceptions | <ul style="list-style-type: none"> • Profile code to identify issues | 0 |
| | Determine techniques required to refactor and improve code | <ul style="list-style-type: none"> • Profile code to identify issues | 0 |
| | Determine techniques required to profile and debug trivial performance issues | <ul style="list-style-type: none"> • Profile code to identify issues | 0 |
| Asset Management | Identify the process required to create a prefab from art and code | <ul style="list-style-type: none"> • Unit 4 - Gameplay Mechanics | ✓ |
| | Identify properties of nested prefabs and prefab variants | <ul style="list-style-type: none"> • Unit 4 - Gameplay Mechanics | ✓ |
| | Identify the primary purposes of version control when working with Unity | <ul style="list-style-type: none"> • Set up version control | ✓ |