Junior Programmer Pathway

Mission 3: Manage scene flow and data

Pathway Description

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. Junior Programmer prepares you to get Unity Certified so that you can demonstrate your job-readiness to employers.

Key details

- ➔ This Mission will take you approximately 14 hours to complete. Take it at your own pace — you'll receive XP each step of the way.
- ➔ Connect with the Unity community as you learn and check the Learn Live calendar for follow-along practical sessions with established Unity creators.
- ➔ When you've finished the Mission, you'll get the Mission badge for your profile and portfolio.

Skills covered in this course

Basic Code Comprehension

- ➔ Interpret simple code
- ➔ Improve simple code using the features of an IDE

Basic Application Scripting

- ➔ Use common logic structures to control the execution of code.
- ➔ Write code that utilizes the various Unity APIs
- ➔ Implement appropriate data types
- ➔ Write code that integrates into an existing system
- ➔ Implement a code style that is efficient and easy to read
- ➔ Prototype new concepts

Basic Debugging

- ➔ Diagnose and fix code that compiles, but fails to perform as expected
- ➔ Diagnose and fix common compilation errors
- ➔ Diagnose and fix compilation errors related to Unity's Scripting API
- ➔ Diagnose and fix the cause of an exception

Beginner Application scripting

- ➔ Create the scene flow in an application state
- ➔ Implement data persistence across scenes and user sessions
- ➔ Level 1 Version control
- ➔ Maintain a project by correctly implementing version control
- ➔ Implement best practices of version control using Unity Collaborate

Basic Code optimization

- Maximize code efficiency by correctly executing coding best practices
- Debug performance issues

Beginner Programming theory

- Analyze the principal pillars of object-oriented programming
- Simplify code and make it reusable by correctly implementing the principles of inheritance and polymorphism
- Make code more secure and usable by correctly implementing the principles of abstraction and encapsulation, including the use of interfaces
- Write efficient, organized, and comprehensible code by correctly implementing the principles of object-oriented programming

How to use the Pathway		
The Unity Essentials Pathway is broken up into 3 “Missions,” with each Mission containing multiple tutorials and assessments. The following Missions make up the complete Pathway:		
	Junior Programmer: Create with Code 1	13 hours and 45 Minutes
	Junior Programmer: Create with Code 2	24 hours and 15 Minutes
	Junior Programmer: Manage scene flow and data	2 hours
	Junior Programmer: Apply object-oriented principles	1 Hour 45 Minutes
Students are encouraged to complete all the Missions in the correct sequence to ensure the best learning experience.		

Table of contents

Mission 3: Manage scene flow and data

Part of the Junior Programmer Pathway

Mission overview

In this Junior Programmer mission, you'll learn about data persistence, or how to manage the loading and unloading of data between both scenes and sessions. Along the way, you will be introduced to the core pillars of object-oriented programming, which will help you structure your code throughout the mission. There will only be one practical example throughout the mission, which will build on the new functionality you create with each tutorial. At the end of this mission, you'll take everything you learned and apply it to a new project by building a persistent score system for a small game.

Key details

- You're now over halfway through the Junior Programmer learning pathway!
- This mission will take you approximately 5 hours to complete. Take it at your own pace — you'll receive XP each step of the way.
- Remember, you're not alone: connect with the Unity community throughout the mission and check the Learn Live calendar for follow-along practical sessions with established Unity creators.
- When you've finished the mission, you'll get the mission badge for your profile and portfolio.

Skills

Basic version control

- Maintain a project by correctly implementing version control
- Implement best practices of version control using Unity Collaborate

Beginner application scripting

- Create the scene flow in an application state
- Implement data persistence across scenes and user sessions

Introduction to real-time 3D experience design

Lesson link	Introduction to real-time 3D experience design
Length	**8 hours**
Summary In this tutorial, you'll explore the basics of real-time 3D experience design. You'll: ● Recap the different phases of production ● Review the core design process for creating a digital experience ● Consider common approaches to real-time 3D experience creation ● Identify your own custom design process for a specific scenario	
Skills Basic design process ● Implement an iterative design process ● Practice putting the user first	
Steps 1. Overview 2. Experience design and the real-time production cycle 3. Review the design process: ideation and research 4. Review the design process: clarified brief and initial prototypes 5. Review the design process: iterative design and development 6. Common real-time experience design and development approaches 7. Exercise: Plan a custom design process 8. Next steps	

Set up version control

Lesson link	Set up version control
Length	**5 minutes**

Summary

In this tutorial, you'll learn about the basics of Version Control, and the reasons to implement it in your own projects, even if you're developing applications by yourself. You'll also learn about the different version control options available to use with Unity, and put your new skills into practice by downloading the project that you'll use throughout the upcoming tutorials.

Outcome

By the end of this tutorial, you will be able to:

- Identify the different version control solutions that can be used to maintain a project.
- Describe how version control is used to maintain a project

Skills

Basic version control

- Maintain a project by correctly implementing version control

Materials

Junior-Programmer-Starter-Files (.zip)

Steps

1. Overview
2. What is version control?
3. Who uses version control?
4. Select your version control tool
5. Get started with GitHub Desktop
6. Create a new repo on your hard drive
7. Download, open, and start tracking your project
8. Do an initial commit and publish your repo
9. How does version control work?
10. Summary

Principles of object-oriented programming

Lesson link	Principles of object-oriented programming
Length	**10 minutes**
Summary In this tutorial you’ll learn about the basics of the object-oriented programming paradigm and its four associated principles. **Outcome** By the end of this tutorial, you will be able to: ● Define encapsulation ● Define inheritance ● Define polymorphism ● Define abstraction ● Explain how the pillars of OOP work together to create organized, efficient code	
Steps 1. Overview 2. What is object-oriented programming? 3. The Four Pillars 4. Object-oriented programming in the missions 5. Next steps	
Steps 1. Overview 2. Before you begin 3. Welcome to the project 4. Explore the simulation 5. Review the project brief 6. Your goal in this mission 7. Next steps	

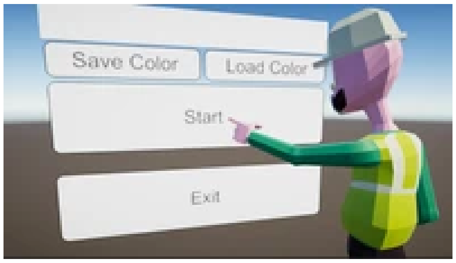
Create a scene flow

Lesson link	Create a scene flow
Length	**20 minutes**

Summary
In this tutorial, you'll set up the scene flow between the Menu and Main scenes, and the exit flow for the application in the application.

Outcome
By the end of this tutorial, you will be able to:

- Call the appropriate start-up methods in the initialization sequence of the app
- Load the next scene when triggered by an event



Steps

1. Overview
2. Review the required scene flow
3. Review the UI menu
4. Write a method to load the Main scene
5. Set the Menu scene index
6. Configure the Start button
7. Write a method to quit the application
8. Revise your method with conditional compiling
9. Add a new namespace
10. Challenge: Set up the transition back to the Menu scene
11. Next steps

Implement data persistence between scenes

Lesson link	Implement data persistence between scenes
Length	**20 minutes**

In this tutorial, you'll learn how to use data persistence to preserve information across different scenes by taking a color that the user selects in the Menu scene and applying it to the transporter units in the Main scene.

Outcome
By the end of this tutorial, you will be able to:

- Ensure data is preserved throughout an application session by using the Unity DontDestroyOnLoad method
- Recognize when to use static classes, singletons, and static variables to implement data persistence
- Modify a GameObject with a script that contains the variables to be saved between Scenes

Steps

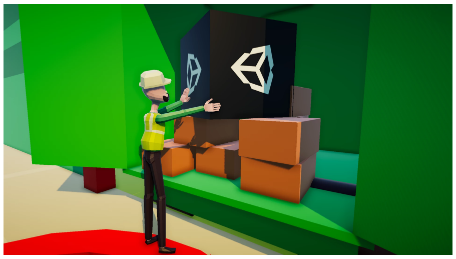
1. Overview
2. What is data persistence?
3. Review your brief

4. Create a new script
5. Review the code
6. Test in Unity Editor
7. Modify the Awake method
8. Store and pass the selected color
9. Next steps

Implement data persistence between sessions

Lesson link	Implement data persistence between sessions
Length	**20 minutes**

Summary
In this tutorial, you'll write code to save and load the color that the user selects so that it persists between sessions of the application.



Outcome
By the end of this tutorial, you will be able to:

- Call the appropriate start-up methods in the initialization sequence of the app
- Store and organize data by applying data structures such as lists and dictionaries
- Save user data in a the shutdown sequence of the app

Steps

1. Overview
2. Evaluate your brief
3. How can data persist between sessions?
4. What is JSON?
5. Why is JSON a good fit for your brief?
6. Add a SaveData class
7. Add a SaveColor method
8. Add a LoadColor method
9. Load and save the color in the application
10. Add testing functionality
11. Next steps

Mission 3 checkpoint

Quiz: Manage scene flow and data
Submission task: Data persistence in a new repro

A successful submission will include

- A link to your project’s GitHub repo, showing multiple commits with commit messages
- Some kind of data persistence between scenes
- Some kind of data persistence between sessions