UNREAL ENGINE

THE KEY TO UNLOCKING THE FINAL AREA:

WORKING WITH CONDITIONAL STATEMENTS IN UNREAL ENGINE

Lesson/Author/Class Information

Lesson Title: Working with Conditional Statements in Unreal Engine

Content/Grade: Computer Science/Hour of Code™: Grades 8–12

Lesson Timeframe: One hour

Teacher Guide

Student Guide

Author Contact

Steve Isaacs teaches Game Design and Development as a quest or choice-based learning environment that provides students with opportunities to take different approaches to meet learning outcomes based on their own interests, in terms of content and project options.

Brian Dickman studied Computer Science and operates a full-time game development studio that produces entertaining and educational content inside popular video games.

Ian Southwell works at the Rocky Mountain College of Art and Design where he teaches in the Game Art and Animation department, as well as running the RMCAD Fabrication Laboratory. He also works as the Creative Director for Cleverlike Studios.

Email: stevei2071@gmail.com | brian@cleverlike.com | ian@cleverlike.com

Twitter: @mr_isaacs | @cleverlike | @IanSouthwell2

LinkedIn: https://www.linkedin.com/in/steve-isaacs/ | https://www.linkedin.com/in/cleverlike/
https://www.linkedin.com/in/southwellian/

Description of class/learning environment

This lesson is designed for Hour of Code™ during Computer Science Education Week. It can be used in any curricular area interested in participating in Hour of Code™. It can also be used outside of Hour of Code™ in a game development or computer science course. It can be used as a stand-alone lesson, or in conjunction with the other activities to complete a larger project.

Lesson Overview

You've jumped through several levels and risked everything—jumping high in the sky across treacherous, moving islands. You finally made it to the castle where you believe you will find the final treasure! You would love to just walk in, wouldn't you? As you might expect, the door is locked and you don't have the key! Find the key and unlock the door so you can reach the final goal.

In this lesson, you'll learn how to work with **conditional statements**. The condition is finding a key to open the door. Once inside, you can complete the game.
Good Luck!

DESIRED RESULTS

Essential Questions/Big Ideas

- Can students learn computer science concepts as part of a meaningful activity rather than simply learning syntax as an isolated skill?
- Will learning computer science concepts, like conditional statements, through an activity in Unreal Engine create a general understanding of the concept in a coding environment?
- Can students learn computer science concepts through game mechanics?
- Will students show more motivation to learn computer science when the concepts are introduced in a game environment?

Learning Outcomes/Objectives

The student will be able to:

- Demonstrate an understanding of the conditional statements.
- Demonstrate an understanding of Boolean values.
- Apply the understanding of conditional statements and Boolean values in the context of a game.
- Create and modify a game level in a true game engine (Unreal Engine) that incorporates conditional statements.
- Work with Boolean values to determine if a condition is met in a coding environment.

LESSON PLAN

Learning Activities

How to Use the Unreal Engine in Hour of Code ™ Lessons

This series of lessons has been designed to introduce students to computer science concepts in the context of using the industry-standard game development tool, Unreal Engine.

Each lesson is set up as a stand-alone lesson to teach a single coding concept in the span of about an hour as part of the Hour of Code™ initiative. The lessons are presented to encourage students to work through them in linear order, as they acquire skills built on previous skills they learned.

The lessons also work together so that a student can complete all five lessons and create a game experience with five different levels, demonstrating the different concepts. The activities lend well to students working in small groups to leverage the idea of *pair programming*.

Each lesson is accompanied by a student guide and a teacher guide with notes for the educator to deliver the lesson and support students in the process.

Using Unreal Engine

It is expected that students have some experience with the Unreal Engine interface prior to starting. To facilitate teaching with Unreal Engine, educators can familiarize themselves with the tool and how to use it in the classroom, using a short course we have developed for this purpose. We encourage you to take the course and earn the badge!

Getting Ready for This Activity

If students completed the previous activity (Working with Public Variables in Unreal Engine) they already completed the Lesson 3 activity and can continue. If not, they can enable Lessons 1, 2, and 3 in the Unreal Engine project by making Lessons 1, 2, and 3 visible, right-clicking on them, and choosing **Change Streaming Method > Always Loaded**. Directions on how to do this are available in the Lesson 4 Teacher Guide.

Hook

Have students launch and play the game.

Students should be able to make it to the castle door, but will notice that they need a key to gain entry.

Discuss the gameplay experience:

- You've made it this far but the door is locked. Do you have some ideas of what you can add to gain access?
- Did you encounter any issues or bugs in the current build?
- Do you have any ideas of how to create a system that allows us to use a key to open the door?

Students should notice that the door is locked and they are prompted that they need a key to gain entry.

Explain to students that in this activity, they will be learning game development and programming concepts about **conditional statements** and **Boolean values**.

Review: Introduction to Blueprints

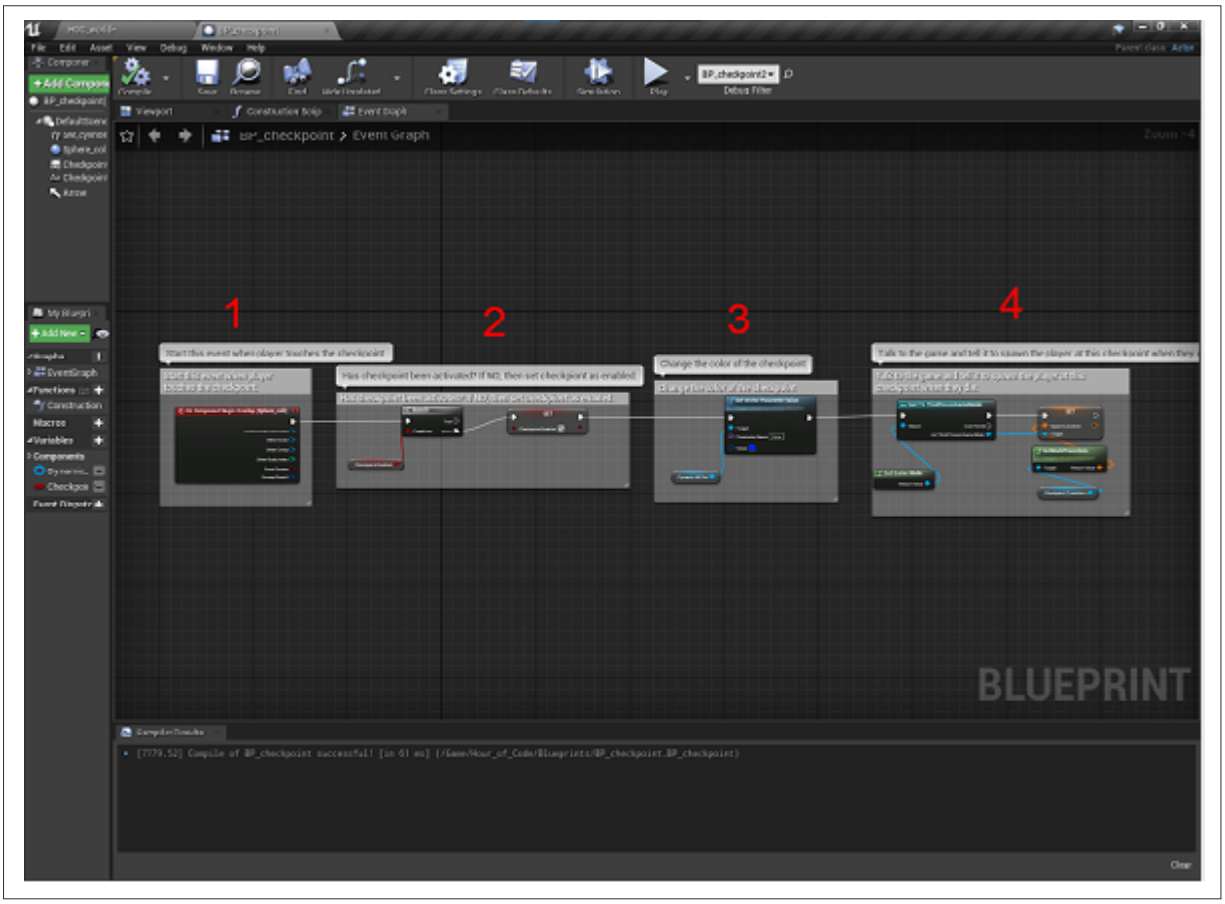
(This will serve as a review if students have completed the previous lesson or as an introduction if you started with this activity)

Blueprints:

The Blueprints Visual Scripting system in Unreal Engine is a complete gameplay scripting system based on the concept of using a node-based interface to create gameplay elements from within Unreal Editor. As with many common scripting languages, it is used to define object-oriented (OO) classes or objects in the engine. As you use UE4, you'll often find that objects defined using Blueprint are colloquially referred to as just "Blueprints."

This system is extremely flexible and powerful as it provides the ability for designers to use virtually the full range of concepts and tools generally only available to programmers. In addition, Blueprint-specific markup available in Unreal Engine's C++ implementation enables programmers to create baseline systems that can be extended by designers.

– from https://docs.unrealengine.com/en-US/Engine/Blueprints/GettingStarted/index.html



A Blueprint showing the drag-and-drop visual coding in Unreal Engine

Conditional statement. A set of rules performed if a certain condition is met. It is sometimes referred to as an If-Then statement, because IF a condition is met, THEN an action is performed.

From: Computer Hope: Conditional statement

For example, IF you have the key THEN you can unlock the door. This example is pretty clear. In order to get into the castle, you will need the key.

Here's a simple video that explains If-Then-Else statements: https://youtu.be/D5fSbCKobko

Conditional statements can be used in any coding language, and also environments like Blueprints in Unreal Engine where you can set up a scenario that tests a condition, and, only if that condition is met, something is activated.

For our purposes, think of the fact that the player either has the key or does not have the key. This represents a **Boolean value**, the value is either true or false. We can check if the player has the key and if the value for has_key is true, then the player can unlock the door.

Pseudocode is a way of writing coding concepts in a simple format that is easy for people to communicate and understand. The actual code in different programming languages will have different rules (or syntax), but pseudocode allows us to think about the code based on what we are trying to accomplish.

In terms of pseudocode, this could look like:

Declare / set variables

```
Var has_key = false
```

Change the boolean value when the player collides with the key

```
If player collides with key, has_key = true
Destroy key
```

Check if player has key in order to open door

```
Check Player for has_key
Does has_key = true
If YES, unlock door
```

In the following activity, we will use **Blueprint Visual Scripting** in Unreal Engine to implement a lock and key system in a game.

Activity

In this lesson, you will get under the hood and create a script from scratch using **Blueprints**, Unreal Engine's visual scripting environment. We will add a key to the game and add code to determine whether or not the player has picked up (collided with) the key. Based on the code, if the player has the key, they will be able to open the locked door and gain entry into the final area of the game.

For step-by-step directions, provide students with the Student Guide refer to the Teacher Guide to help guide students through the process.

External Resources

Your First Hour with Unreal Engine:
https://www.unrealengine.com/en-US/onlinelearning-courses/your-first-hour-with-unreal-engine

Introduction to Blueprints Visual Scripting:
https://docs.unrealengine.com/en-US/Engine/Blueprints/GettingStarted/index.html

CS Discoveries: Conditionals Part I (video):
https://youtu.be/D5fSbCKobko (from code.org)

ASSESSMENT

Assessments

Rubric

Hour of Code™The Key to Unlocking the Final Area: Working with Conditional Statements in Unreal Engine

	Developing	Competent	Proficient	Distinguished
Project Content and Learning Objectives	Project does not convey the required information or understanding as it pertains to the learning objectives.	Project shows a basic understanding of Blueprints and conditional statements.	Project reflects understanding of the basic structure of Blueprints and how to create a Blueprint to toggle Boolean values.	Project reflects exemplary understanding and application of Blueprints and how visual scripting works in Unreal Engine. Mastery of the learning objectives are met or exceeded. Student effectively incorporates the use of blueprints and conditional statements to get the desired result.
Project Development and Functionality	Project does not work, or has major flaws that prevent its intended use.	Project demonstrates basic functionality, and has only minor flaws.	Project functions in the way the student intended and is intuitive for the end user.	Project is functional and refined, with extra features that exceed the requirements.

Project Aesthetic and Design	Project requires more attention to the look and feel of the experience and the general design.	Project shows some attention to aesthetics and thoughtful design, but is incomplete or lacking in some aspects of layout and design.	Project is well organized and pleasing to the eye. The design makes sense in the context of the activity and creates a well-designed experience for the player.	Great attention to design. The environment is inviting, and provides the user with an engaging world to explore in order to experience the puzzle activities.
Reflection	Student demonstrates difficulty describing Blueprint visual scripting and conditional statements and how they are represented in this activity.	Student describes the basics of Blueprints and conditional statements, and has a general understanding of how they relate to game development and programming.	Student provides a thoughtful reflection or explanation of Blueprints and conditional statements and incorporates the concepts into practical use in Unreal Engine.	Student can eloquently explain the concepts of Blueprints and visual scripting. Conditional statements and Boolean values are effectively represented using Unreal Engine.

Standards Mapping

CSTA Standards for Students: https://csteachers.org/Page/standards

1A-AP-09
Model the way programs store and manipulate data by using numbers or other symbols to represent information.

1B-AP-09
Create programs that use variables to store and modify data

1B-AP-10
Create programs that include sequences, events, loops, and conditionals.

1B-AP-12
Modify, remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.

1A-AP-14
Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops

1B-AP-15
Test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended.

2-AP-10
Use flowcharts and/or pseudocode to address complex problems as algorithms.
2-AP-11
Create clearly named variables that represent different data types and perform operations on their values.

2-AP-13
Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.

2-AP-17

Systematically test and refine programs using a range of test cases.

3A-AP-13
Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3A-AP-16
Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.

3A-AP-17
Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

3B-AP-22
Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).

Interdisciplinary and 21st-Century Connections

This lesson covers areas related to coding and Computer Science.
21st Century Connections:

- Critical thinking
- Creativity
- Collaboration
- Communication
- Technology literacy
- Flexibility
- Leadership
- Initiative
- Social skills

Modifications and Accommodations

Provide modifications and accommodations as appropriate based on student needs, IEP, 504, and so on.

Students can work in teams to integrate a paired programming approach

The completed project can be provided for students to deconstruct or modify.