HOUR OF CODE

UNREAL ENGINE

BUILD YOUR FIRST 3D GAME:
LEARN COLLISION DETECTION IN UNREAL ENGINE

Lesson/Author/Class Information

Lesson Title: Collision Detection 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 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

Have you ever wondered how developers create platform games that involve running and jumping, while hopefully not falling to your demise? Now it's your turn to become the developer and build your own 3D game using Unreal Engine, one of the industry standard tools for game development.

In this lesson, you will learn how to build a simple parkour course to get the player through a hallway, and across a treacherous void. You will learn about collision detection and how important it is in computer programming, especially game development. You will also have a chance to get familiar with navigating the Unreal Engine user interface, more specifically the Viewport. You will explore the interface in order to modify objects to include collision detection and place objects throughout the level to develop the game world.

DESIRED RESULTS

What are the learning outcomes for students?

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 collision detection, 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 collision detection as a concept.
- Apply the understanding of collision detection in the context of a game.
- Create and modify a game level in a true game engine (Unreal Engine) that incorporates the use of collision detection.

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*.

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!

Hook

Have students launch and play the game.

Discuss the gameplay experience:

- Were you able to move around in the game?
- Were you able to jump?
- Did you encounter any issues or bugs in the current build?
- What do you need to do to fix this and make the game playable?

Students should notice that they can move around and jump, but when they attempt to jump on the first platform they fall to the ground and get stuck there. In terms of fixing the game, they may mention troubleshooting ideas, including fixing the platforms, and making the player respawn if they hit the ground (or another means to get back up).

Explain to students that, in this activity, they will be learning game development and programming concepts about collision detection.

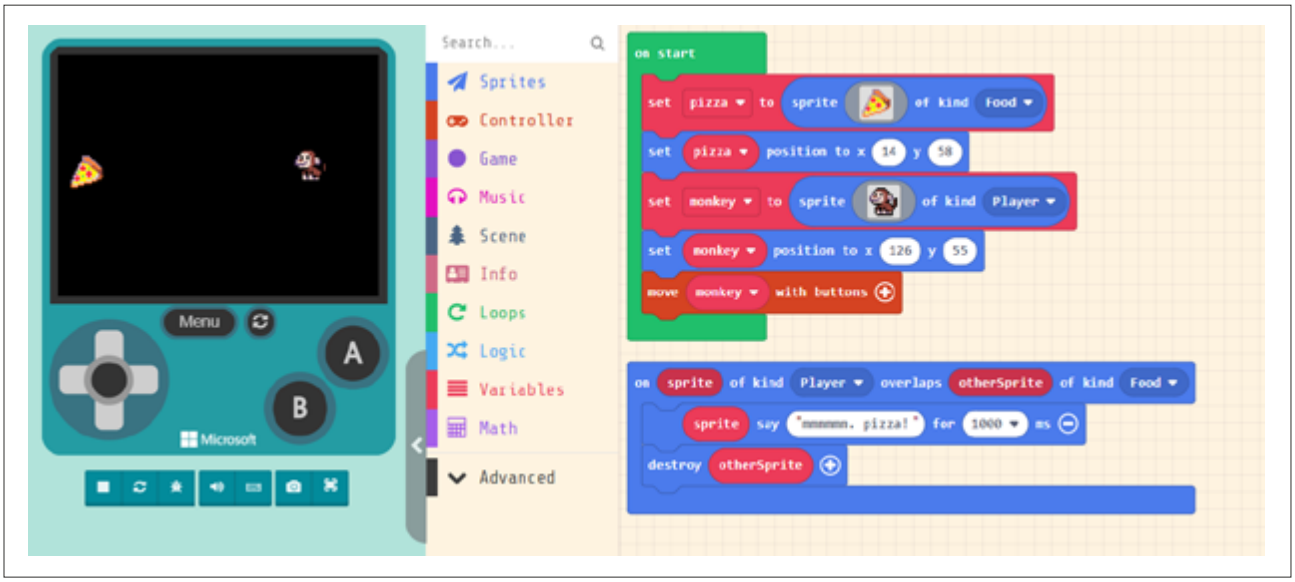
Introduction to Collision Detection

Collision Detection. In game development and programming, collision detection refers to the act of objects entering the space occupied by another object, or overlapping one another.

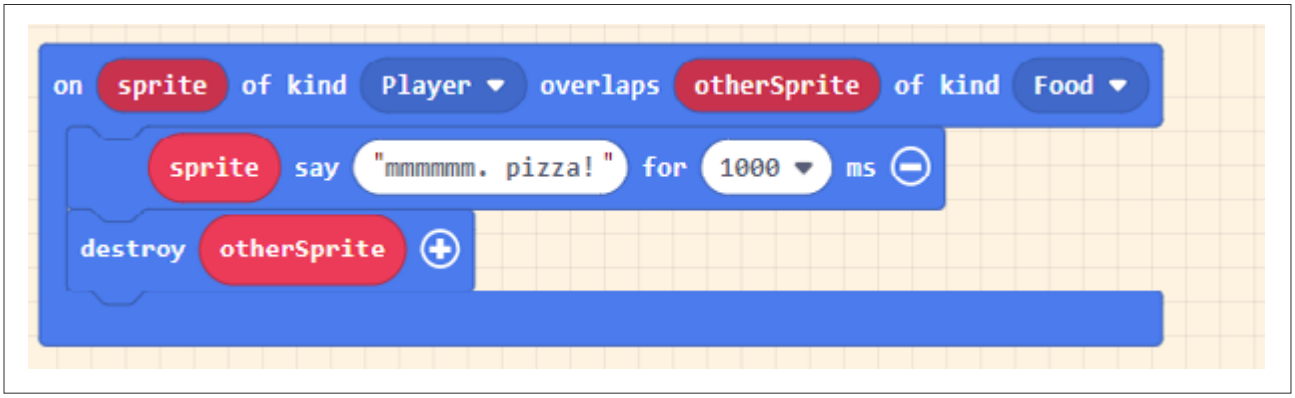
Examples:

- A player loses a life when it runs into (collides with) an enemy.
- A player gains a point when a projectile hits (collides with) an enemy. Also, the enemy is destroyed when the projectile hits it.
- A player lands on a platform after jumping and colliding with it.

Here's a simple example using Make Code (http://arcade.makecode.com).



In the image above, the monkey has pizza on its mind. When she gets the pizza, she will say, "mmmmmm. pizza!" and the pizza will be destroyed.



The block-based code above indicates that when the player sprite overlaps with the food sprite, the player sprite will say "mmmmmm. pizza!" for 1 second and the food (otherSprite) will be destroyed. The overlap command block is used to represent a collision event.

In JavaScript, the code would look something like this:

```
sprites.onOverlap(SpriteKind.Player, SpriteKind.Food, function (sprite, otherSprite) {
    sprite.say("mmmmmm. pizza!", 1000)
    otherSprite.destroy()
})
```

You can access the example game and code here:
https://makecode.com/_48e6y2CoLiip

Activity

Students will modify an existing project in Unreal Engine to incorporate the concept of collision detection. They will add a number of platforms to the game to create a parkour level for others to play.

For our purposes, *parkour* refers to the player running and navigating jumps of varied difficulty over a deadly void. The jumps that they set up will give the player a variety of challenges to overcome.

Students will also add and configure a danger zone using the **pain-causing volume** in Unreal Engine to account for situations when the player misses the platform. They will also modify the **Kill Z** setting to account for players who fall off the edge so they do not fall forever.

Designing for a positive player experience

Students can make the course as easy or as difficult as they like. They should be mindful that difficult does not always equal fun. You might think your level is too easy to complete because you are the designer and have played through your level many times. Peers will playtest and provide feedback. Developers should be sure to take all the feedback from their playtesters seriously because this will make the game more enjoyable for a wider group of players. The more playtesters they work with, the more successful their game will become.

For step-by-step directions, provide students with the Student Guide and 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

MakeCode Arcade Collision Detection sample game:
https://makecode.com/_48e6y2CoLiip

ASSESSMENT

Assessments

Rubric

Hour of Code™ Build Your First 3D Game: Learn Collision Detection 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 collision detection and learning objectives.	Project reflects understanding of collision detection and coding, and how that can be accomplished through modifying object properties and values in Unreal Engine. Demonstrates understanding of the importance of peer testing and feedback.	Project reflects exemplary understanding and application of collision detection through gameplay. Mastery of the learning objectives is met or exceeded. Student actively incorporates peer feedback to iterate on original design.
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 provides general guidance 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 collision detection and the connection between code and this activity.	Student describes the basics of collision detection and has a general understanding of how collision detection relates to game development and programming.	Student provides a thoughtful ref lection or explanation of collision detection and how to incorporate the concept into practical use in Unreal Engine.	Student provides a thoughtful reflection or explanation of collision detection and how to incorporate the concept into practical use in 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-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.

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-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
- Communication
- Information literacy
- Technology literacy
- Flexibility
- Initiative

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.