<2D One Button game prototype>

12.05.2021

—

Luc Partridge
801224946

Contents

Task #1

Question #1: Identify and describe common game programming languages, their syntax, and command structures/ *C#*

C#

C#'s syntax makes use of name spaces to define certain objects within the game for example, GameObject vs gameObject, GameObject refers to a class in which it holds many components to make up the entity within unity.
Unity specifically has specific data types called Access modifiers which allows for reference of the script from other different scripts (e.g a script that controls movement may reference the Player script.)
A programmer may use these two bits of code for different purposes. such as declaring a public GameObject so that he may then refer to that GameObject in a different script and then call upon gameObject to call a component within the specific GameObject the script is influencing in order to change it later on in the game.

To run lines of code, C#'s syntax makes use of " { }" usually within functions. Any line of code within unity with few exceptions are found within "{}".
In games, in order to detect inputs, if statements are usually used so that the game will understand when an input has been made by the player, however the syntax for an if statement needs specific syntax in order for a computer using the language to be understood correctly.
For example, C#'s syntax uses semi colons to end statements so that a line of code can execute properly. However, at the start of an if statement, because you want to determine what is going to be called and when to call upon it, you need to put your semi colon after the statements within the curly braces and not straight after the if line so that no errors may pop up because of that. Because if you were to do that, the language wouldn't understand what happens when the if statement tries to determine something.

C++

C++'s syntax is similar to C#'s in that in order to end statements, it also requires a semi colon. C++ in game development like many other programming languages, may use comments in order to describe something about a particular line of code so that one might better understand it (the ideal scenario). Commenting in C++ is done by two forward slashes together with no spaces (//) . Much like C#, which can also use the same syntax for comments, also uses a forward slash and an asterisk(star) to perform multi line comments which would then be ended with an asterisk then followed by a forward slash. (E.g /* This is a multiline comment */) You would not end this with a semi colon because that is the incorrect syntax for comments.

Unlike C# however is the use of headers inside its code. While both C# and C++ may use the key word “using” (the keyword syntax to import namespaces) C++ can utilize headers to include a set of predetermined function libraries. The keyword to include headers is aptly named “#include” the hash key must be used as syntax for making use of a header so that it can reference it the library

Lua;

Lua scripting in game development is rather different compared to C# and C++, while its keywords might be similar, but there is a lack of a semi colon because the language is smart enough to know when the statement ends. The cases where Lua uses semi colons are purely for readability (To help ease in possible C users) or separating something from each other on the same line of code. For instance; declaring A+B and C+D on the same line would be “ A+B ; C+D ”

For commenting Lua uses –-[[within –]] to help with understanding their code better should the programmer choose to include comments within their script.

/

Question #2 - Explain Boolean algebra and how you can apply it to your game

/ Boolean algebra is a simplified algorithm that is comprised of two possible values and determines a single value from these two. These values are Binary, so it’s only possible between 0 and 1. And that makes up both values needed to make a boolean algorithm. Typically, they are seen as “ON” or “OFF” switches where “ON” may equal 1 and “OFF” may equal 0 depending on the language and software used.

In my game, by using the boolean data type that uses boolean algebra, I can use this as a check to see whether or not the player “a cannon ball” has entered the cannon or is currently out of the cannon. This will help the game understand that if the player is inside the cannon, the game can now perform another task that allows the player to fire out of said cannon. I.e, if the player has entered the cannon, set boolean to “true” or “1” in Unity’s case. /