A list of Unity and C# things to consider learning about, coming into CTIN-532
by Richard Lemarchand, with help from Margaret Moser and Peter Brinson
Updated 30 May 2024

In the following pages, you can find a list of things that I think it will benefit everyone to know about, when you arrive in my class CTIN-532, if you plan to use Unity in my class. (You can use engines other than Unity in our class, but only by prior arrangement with me, so please come talk to me early if you don't want to use Unity!)

Before taking a class like ours, that is focused on production methodology, it's a good idea to have a solid grounding in how to use a game engine like Unity. That way, you can focus most of your effort on learning production skills, instead of learning how to use Unity.

So the list at the end of this email represents what we think is a decent breadth and depth of knowledge and concepts that you would encounter in most game engines, and which you should try to learn about before class starts. People who took all of our "introduction to programming" classes should already have learned at least a little bit about everything on this list.

If you didn't take those classes, and the list below seems scary—***don't panic!*** Just email me, and we'll figure out what's right for you! Knowing even some of what's on this list will give you the ability to do well in my class. We can talk together to work out what would be best for you to learn.

I'm hoping that you'll be able to use this list to build on the existing knowledge you have, using whatever learning resources fit your learning style. These could include Unity's own Tutorial Videos, Manual and Scripting Reference, the Unity tutorials on Lynda.com/LinkedIn Learning (log in using your USC ID), and any tutorial videos you can Google up from YouTube.

For a more structured learning approach to Unity and programming, I'd like to recommend my friend Jeremy Gibson Bond's book, *Introduction to Game Design, Prototyping, and Development: From Concept to Playable Game with Unity and C# (3rd edition)*, for which, in the interests of full disclosure, I wrote the foreword.

https://a.co/d/2iChVNl

I took Jeremy's class while he taught here at USC, and that was how I learned Unity and C#. I've also worked through about half the exercises in the book, which provide very clear step-by-step instructions that will onboard you to a large number of important concepts in a short amount of time. It's a good book, and easy to learn from!

I hope this is helpful! Please let me know if you have any questions or comments, going forward! Good luck in your learnings!

Very best wishes,

Richard

Good C# and Unity Things to Know Something About Before CTIN-532 Starts:

Variables and data types
- Floats, ints, string and chars, etc.
- including scope and access levels (i.e. when and how to make something public)

Conditionals
- The whole family of if, if...else and while, etc.

Loops
- Of various kinds
- Including foreach statements

Input
- Input using the keyboard
- Any other input mechanisms that you're interested in using, e.g. mouse, touchscreen

The Physics System
- Colliders
- Colliders as Triggers
- Rigidbodies
- Physics Forces
- Physics Materials
- Physics Joints
- Detecting Collisions with OnCollisionEnter
- Raycasting

Collections
- Lists (most important)
- Arrays (good to know about, but not as important as Lists)
- Adding, removing and accessing their elements

Functions,
- Defining and calling a function
- Passing parameters
- Returning values

MonoBehaviour Messaging

- The difference Awake and Start
- The difference between Update and FixedUpdate
- Collisions
- Callbacks
    - Some familiarity with a variety of these
    - OnEnable, OnMouseDown, OnTriggerEnter, etc.
- Vector3 static variables and functions
    - Including Vector3.zero, Vector3.up etc.
    - Including Vector3.normalized, Vector3.magnitude, etc.
    - Including Vector3.Distance, Vector3.MoveTowards, Vector3.RotateTowards etc.

Various

- Instantiate
- Destroy
- Using the GUI interface (Text and Buttons are the two that everyone will need)
- State Machines
    - The big-picture concept
    - Building state machines with Switch statements
- The meaning of Time.deltaTime
- How to find and reference other GameObjects
    - ...using approaches other than GameObject.Find()" such as public variables and collision callbacks
    - ...for the sake of referencing and changing their components, especially their script components, to access their variables and call their functions

A couple of suggestions for things that you should get comfortable doing, that integrate a few of the concepts and skills listed above:

- Writing a timer, either by hand or using coroutines
- Writing a spawner, that instantiates objects, adds them to a list and removes them when they are destroyed