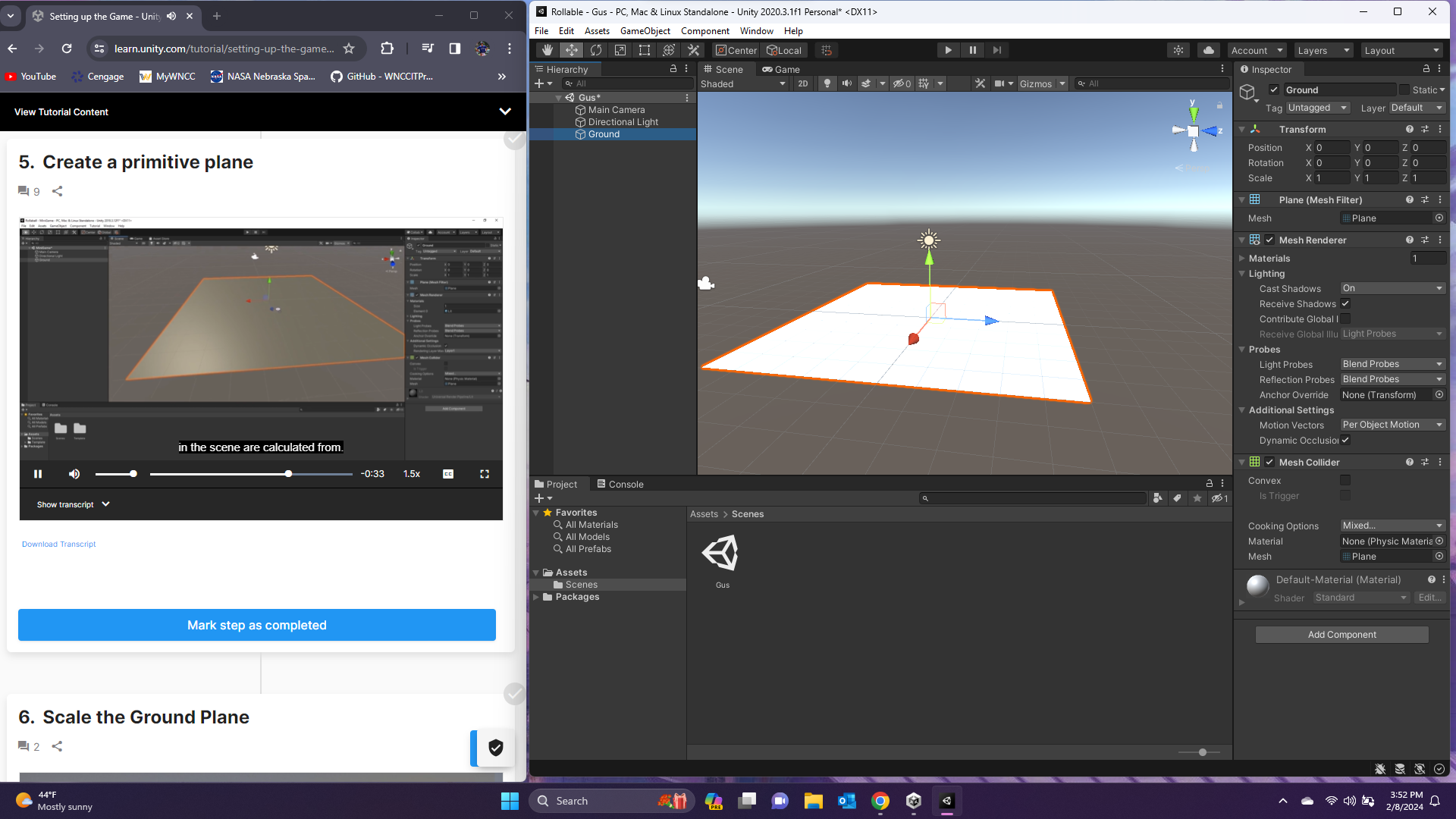
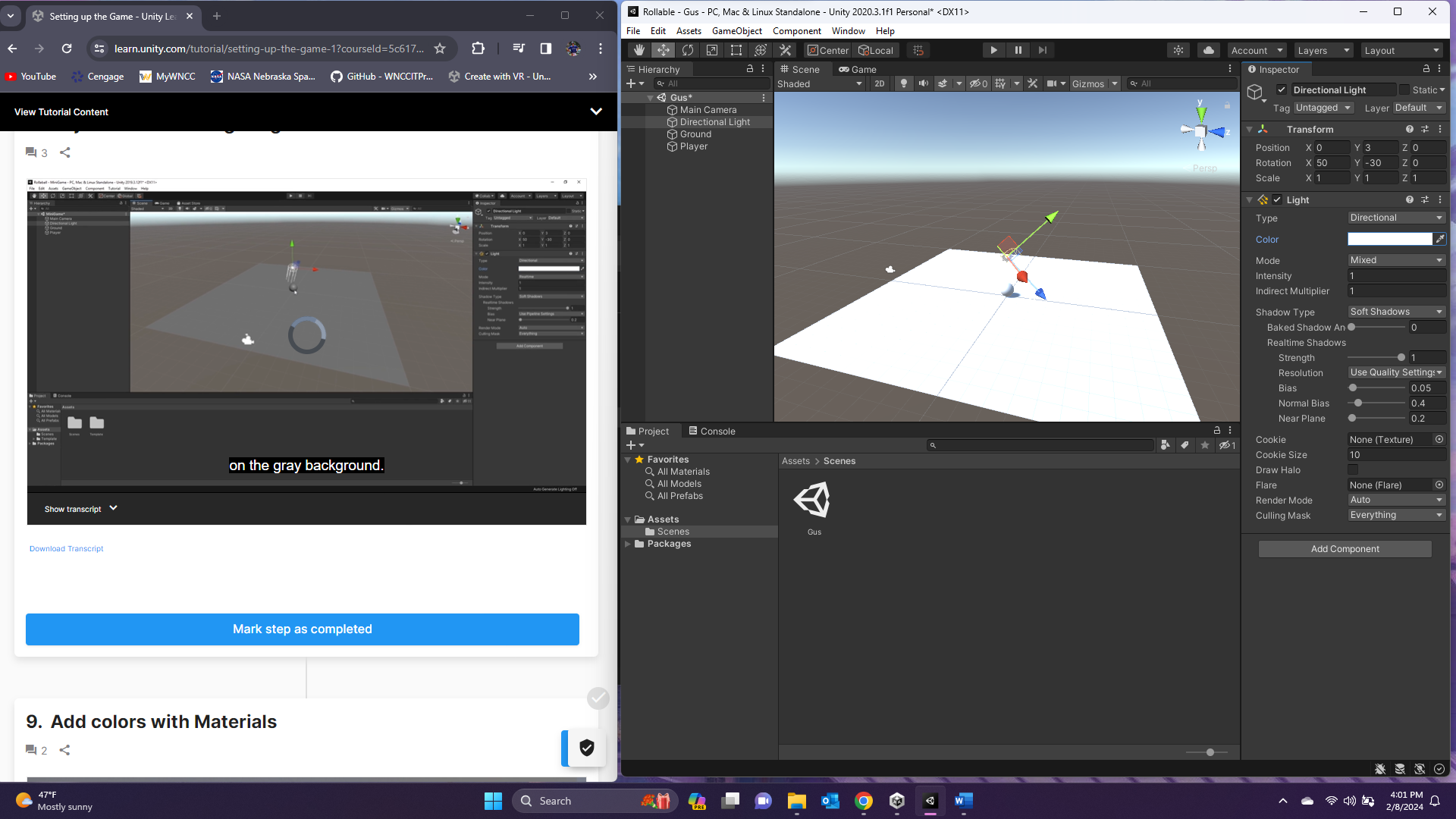
**Think Aloud 1**

This week, I decided to change the direction of my think aloud. I’m instead going to be focusing on learning scripting in C# for Unity because that’s a more practical application to real-world game development than basically learning a deep dive on how game engines work directly. I figure that this way, I can still learn some deeper game development concepts while also being able to apply them. I want to start with the Beginner Scripting course on Unity’s website. There, I will learn the basics and get a better understanding of how things interact within games.

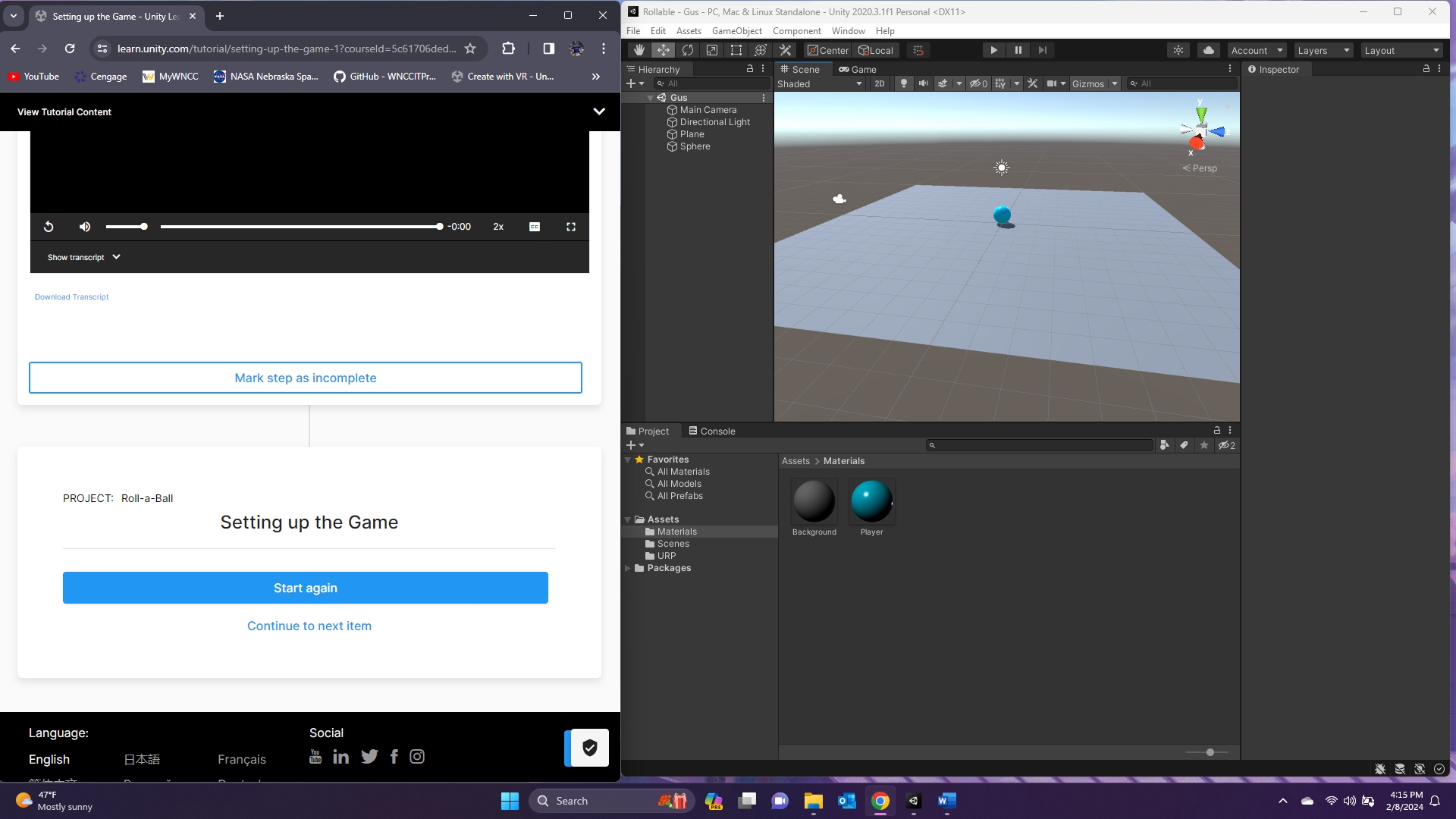
I’m first setting up a new game and placing items. After this, I’ll write a simple script for a ball to roll on a plane that I’ve placed as the ground.



I then added a sphere for the player object.



Then, I ended up having to search and find how to import the Universal Render Pipeline into an existing Unity project, so I did that. Then, I applied URP to the project and upgraded the materials to get the right compatibility of objects. Finally, I made some materials and applied them to the plane and sphere.



I’ve enjoyed playing around with Unity some more, but I’m excited to start diving into the coding side of game development. I haven’t seen much of that side for practical game development. I’ve only done very retro games, so I’m excited to work on more modern 3D type games.

**Think Aloud 2**

This week, I’m continuing working on my rolling ball game and learning scripting. I started by opening Unity and adding a rigid body to my player object. I then installed the input system package and added player input for moving the ball. I had to make sure to apply that player input to the component so game would take player input. I added an empty script to the player object, and I’m not adding code to that script.

A screenshot of a computer

Description automatically generated

I next added input data to the player by adding a rigid body variable and on move function to collect input from the user.

A screenshot of a computer program

Description automatically generated

After that, I added a FixedUpdate function where I added force to the player object. I had to convert the Vector2 to Vector3 in this function, and I added a speed variable that I multiplied the movement by to change player speed. This made my ball roll correctly.

A black background with white text

Description automatically generated

A screenshot of a computer program

Description automatically generated

A computer screen shot of a blue ball

Description automatically generated

A screenshot of a computer

Description automatically generated

Overall, I’m really enjoying where I’m going with this, and it’s really cool to see my changes affecting my game so directly. I feel like I’m making good progress into learning how to apply physics to my game.

**Think Aloud 3**

I’m continuing on scripting and working with my rolling ball game with the Unity tutorial. I first moved the camera and then made it a child of the player object. This, however, made the camera rotate with the player and made it disorienting to play. Instead, I will make a script for the camera to follow the player’s location, but not orientation.

I added variables for the player and for the offset value of the camera. I then calculated positioning of the camera and made the Update function a LateUpdate function to make sure it was called after other updates.

A screenshot of a computer program

Description automatically generated

Then, I tested my script in my game, and the camera followed the position of the ball without rotating. This makes the game much more intuitive.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer

Description automatically generated

I’m happy with how smoothly this is going so far, and I’m excited to see where I continue to go with this project. I hope to be adding animation and other cool features soon.

**Think Aloud 4**

This week, I’m moving on the Setting up the Play Area, Creating Collectibles, and Detecting Collisions with Collectibles tutorials. I started by building creating the walls of the play area and adding material to them.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The ball now stays within the walls when I play the game, and I’m ready to move on to Creating Collectibles. I first added a cube and rotated and added a material to it. This is my collectable object.

A screenshot of a computer

Description automatically generated

I then added a script that continuously rotates the cube while the game is being played.

A screenshot of a computer

Description automatically generated

Next, I made the collectible object a prefab and made duplicates of the object all over the map.

A screenshot of a computer

Description automatically generated

Next, I worked on the Detecting Collisions with Collectibles tutorial. I added to the PlayerController.cs file.

A screen shot of a computer program

Description automatically generated

I then went back to the editor and added a new tag called Pickup. I applied that tag to the Pickup prefab.

A screenshot of a computer program

Description automatically generated

I then added a conditional statement to the script and changed the collider for the prefab to is trigger.

A screenshot of a computer

Description automatically generated

Finally, I added a Rigidbody component to the prefab. I also made the prefab kinematic so that it won’t respond to physics forces. It instead responds to its transform.

A screenshot of a computer

Description automatically generated

A screenshot of a video game

Description automatically generated

My game now has functionality and only needs a scoring system and some minor things to complete it. This week, I added the Rollable.exe file for everyone to test it out. Right now, there isn’t a way to exit the game other than ctrl+alt+del and ending the task in task manager, but I’ll get to that soon.