OverPlate!

A Couch Co-Op Clone by Cameron Lee

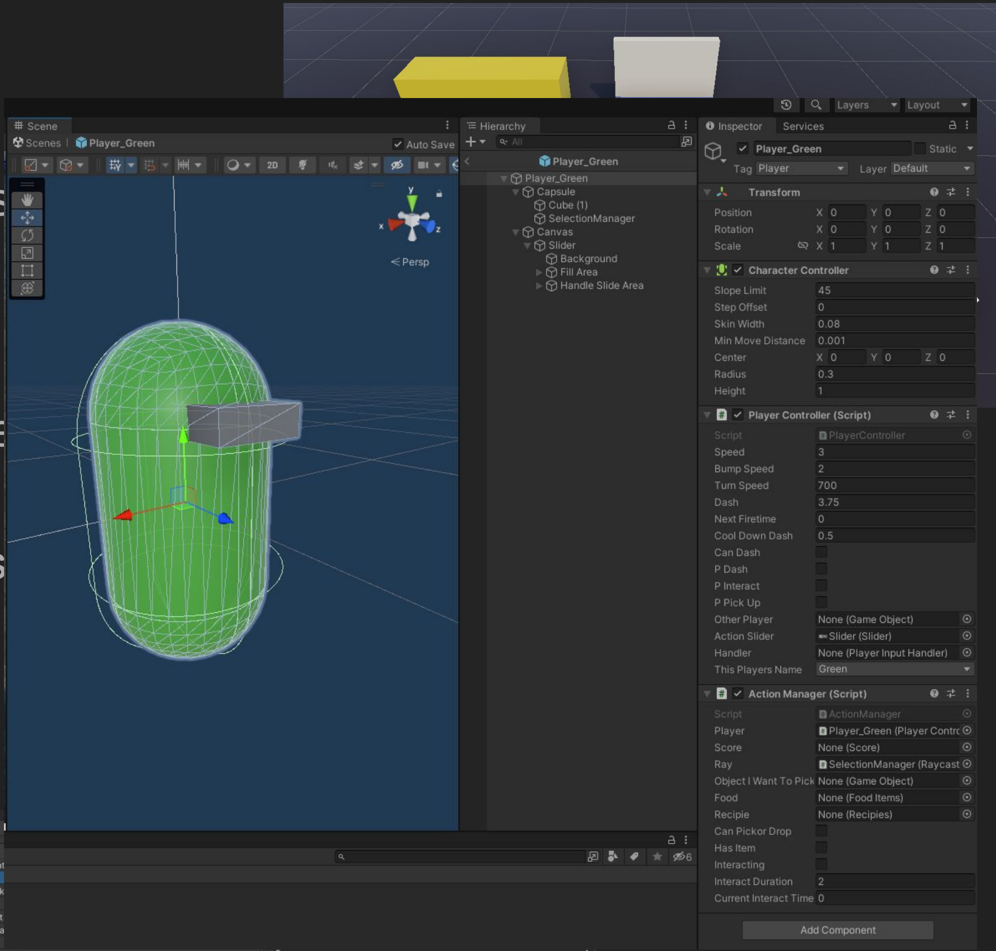
Overview

- Drop in drop out 4 player Co-op, players can join or leave at any time
- Recipe and combination system to allow players to fuse ingredients and make dishes.
- Action system that processes data to so player can perform actions
- “Magic” station, players can refine ingredients to make dishes
- Selection system, objects that player is looking at gets highlighted
- UI, Pause Menu to quit program and resume play
- Score system which increase when completed dishes are put in goal
    - Score is saved when player quits and is loaded back in when the player launches program

Unity scripts and Gui

Since Unity is a game engine, it handles
processes for the developer such as:

- Shapes to build objects
- GameObject, a reference given to e
  object
- Input from hardware devices (mous
  keyboard, gamepad, stick)
- UI elements
- Lighting
- Rays



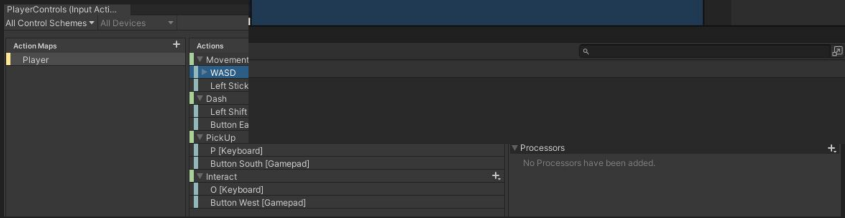
Player_Green Static
Tag Player Layer Default

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Character Controller
Slope Limit 45
Step Offset 0
Skin Width 0.08
Min Move Distance 0.001
Center X 0 Y 0 Z 0
Radius 0.3
Height 1

Player Controller (Script)
Script PlayerController
Speed 3
Bump Speed 2
Turn Speed 700
Dash 3.75
Next Firetime 0
Cool Down Dash 0.5
Can Dash
P Dash
P Interact
P Pick Up
Other Player None (Game Object)
Action Slider Slider (Slider)
Handler None (Player Input Handler)
This Players Name Green

Action Manager (Script)
Script ActionManager
Player Player_Green (Player Contr
Score None (Score)
Ray SelectionManager (Raycast
Object I Want To Pick None (Game Object)
Food None (Food Items)
Recipie None (Recipies)
Can Pick Drop
Has Item
Interacting
Interact Duration 2
Current Interact Time 0



PlayerControls (Input Acti...
All Control Schemes All Devices

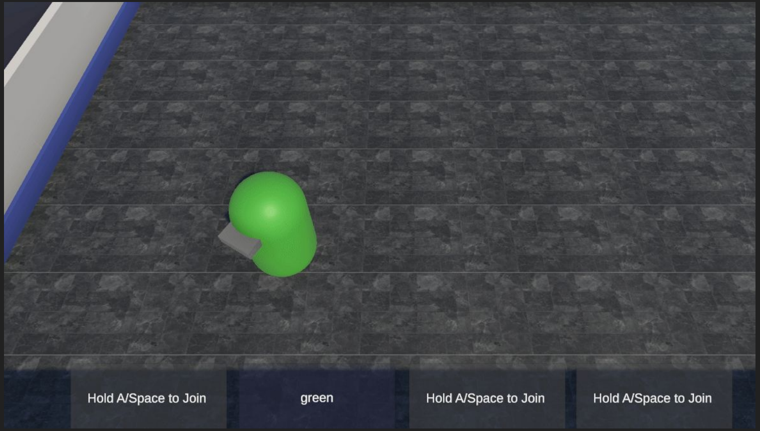
Action Maps
Player

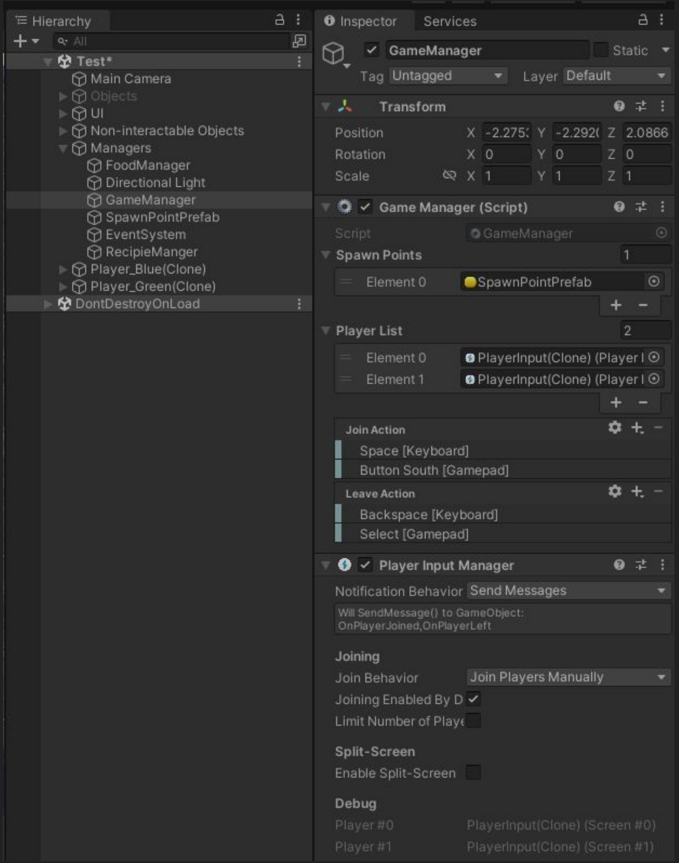
Actions
Movement
WASD
Left Stick
Dash
Left Shift
Button Ea
PickUp
P [Keyboard]
Button South [Gamepad]
Interact
O [Keyboard]
Button West [Gamepad]

Processors
No Processors have been added.

Drop in Drop Out Co-op - Party of four?

- When input manager receives input from new player, it initializes a new player. (more about this in next slide)
- New player is then stored in a player arraylist, which references the input handler, allowing new player to reference all common scripts in game.





Hierarchy

- Test*
  - Main Camera
  - Objects
  - UI
  - Non-interactable Objects
  - Managers
    - FoodManager
    - Directional Light
    - GameManager
    - SpawnPointPrefab
    - EventSystem
    - RecipieManger
  - Player_Blue(Clone)
  - Player_Green(Clone)
- DontDestroyOnLoad

Inspector | Services

GameManager ✓ — Static
Tag: Untagged — Layer: Default

Transform

	X	Y	Z
Position	-2.275:	-2.292(	2.0866
Rotation	0	0	0
Scale	1	1	1

Game Manager (Script) ✓

Script: GameManager

Spawn Points: 1
- Element 0: SpawnPointPrefab

Player List: 2
- Element 0: PlayerInput(Clone) (Player I
- Element 1: PlayerInput(Clone) (Player I

Join Action
- Space [Keyboard]
- Button South [Gamepad]

Leave Action
- Backspace [Keyboard]
- Select [Gamepad]

Player Input Manager ✓

Notification Behavior: Send Messages

Will SendMessage() to GameObject: OnPlayerJoined,OnPlayerLeft

Joining
- Join Behavior: Join Players Manually
- Joining Enabled By D: ✓
- Limit Number of Playe: ☐

Split-Screen
- Enable Split-Screen: ☐

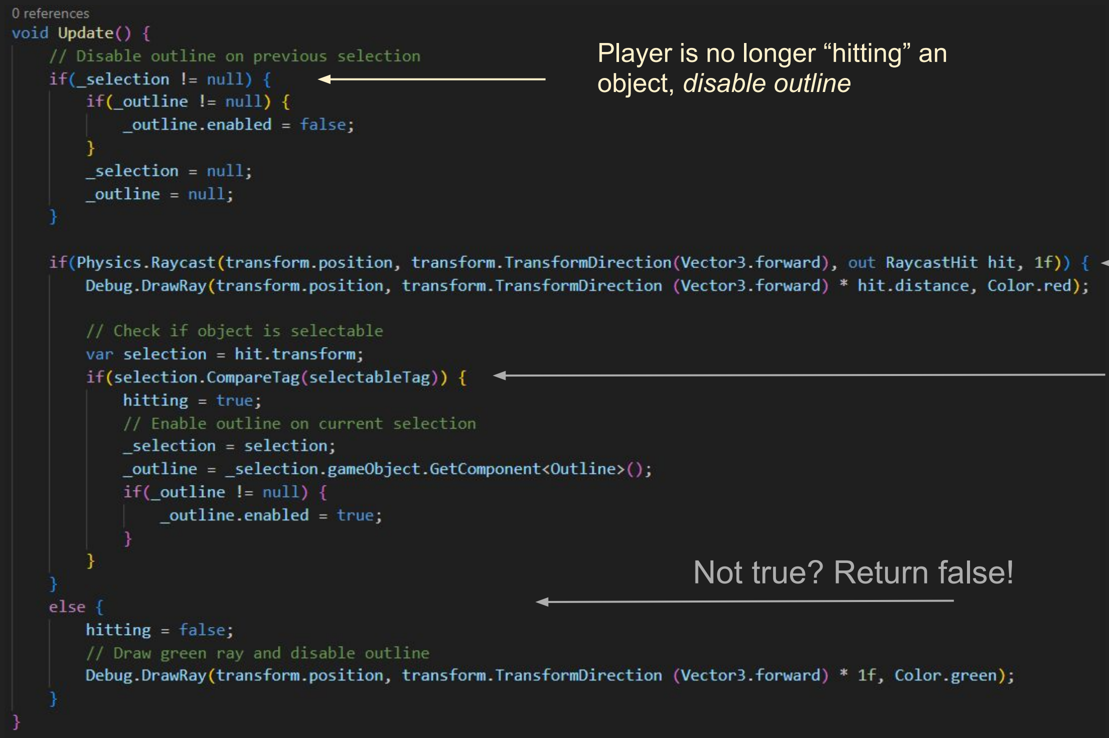
Debug
- Player #0: PlayerInput(Clone) (Screen #0)
- Player #1: PlayerInput(Clone) (Screen #1)

Selector - Raycasting

Shoots a ray out of player, returns true or false if it hits a selectable item

True: enable outline

False: Disable outline



```
0 references
void Update() {
    // Disable outline on previous selection
    if(_selection != null) {
        if(_outline != null) {
            _outline.enabled = false;
        }
        _selection = null;
        _outline = null;
    }

    if(Physics.Raycast(transform.position, transform.TransformDirection(Vector3.forward), out RaycastHit hit, 1f)) {
        Debug.DrawRay(transform.position, transform.TransformDirection (Vector3.forward) * hit.distance, Color.red);

        // Check if object is selectable
        var selection = hit.transform;
        if(selection.CompareTag(selectableTag)) {
            hitting = true;
            // Enable outline on current selection
            _selection = selection;
            _outline = _selection.gameObject.GetComponent<Outline>();
            if(_outline != null) {
                _outline.enabled = true;
            }
        }
    }
    else {
        hitting = false;
        // Draw green ray and disable outline
        Debug.DrawRay(transform.position, transform.TransformDirection (Vector3.forward) * 1f, Color.green);
    }
}
```

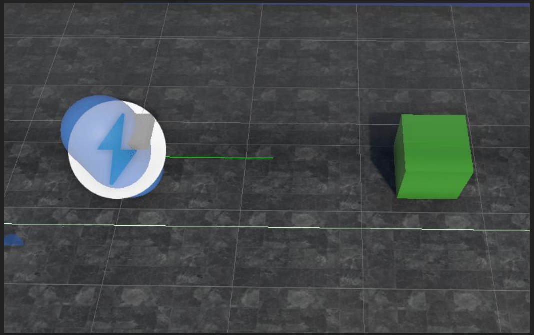
Player is no longer “hitting” an object, *disable outline*

If ray hits a gameobject, store it’s information in “hit”

Is “hit” a selectable item?

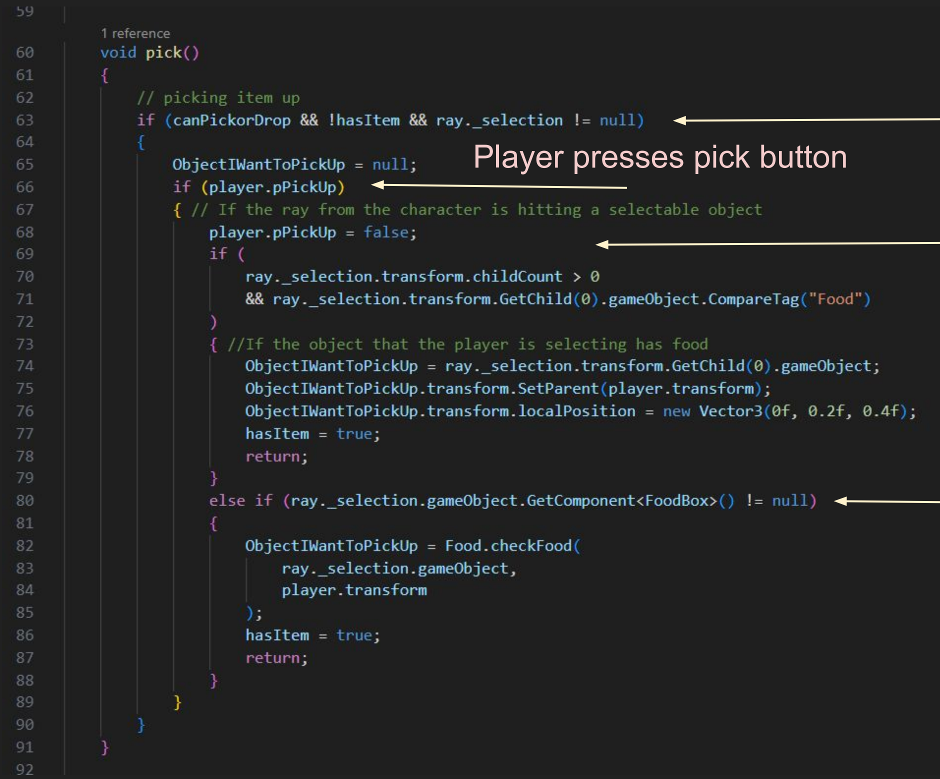
If so, player is **“hitting”** an object!
Return True!
Enable outline

Not true? Return false!



Action Manager - processing data

Processes **input data** and **object data** from selector to **perform actions!**

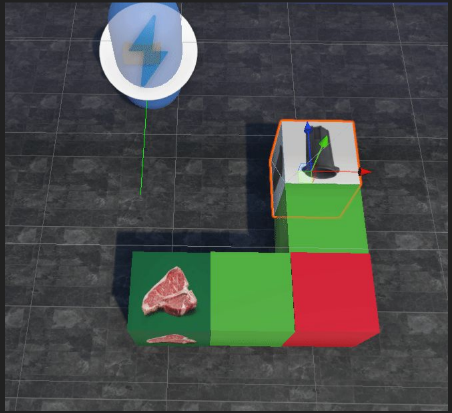


```
1 reference
void pick()
{
    // picking item up
    if (canPickorDrop && !hasItem && ray._selection != null)
    {
        ObjectIWantToPickUp = null;
        if (player.pPickUp)
        { // If the ray from the character is hitting a selectable object
            player.pPickUp = false;
            if (
                ray._selection.transform.childCount > 0
                && ray._selection.transform.GetChild(0).gameObject.CompareTag("Food")
            )
            { //If the object that the player is selecting has food
                ObjectIWantToPickUp = ray._selection.transform.GetChild(0).gameObject;
                ObjectIWantToPickUp.transform.SetParent(player.transform);
                ObjectIWantToPickUp.transform.localPosition = new Vector3(0f, 0.2f, 0.4f);
                hasItem = true;
                return;
            }
            else if (ray._selection.gameObject.GetComponent<FoodBox>() != null)
            {
                ObjectIWantToPickUp = Food.checkFood(
                    ray._selection.gameObject,
                    player.transform
                );
                hasItem = true;
                return;
            }
        }
    }
}
```

If the player is **"hitting"** a selectable object and has nothing in hands

Player presses pick button

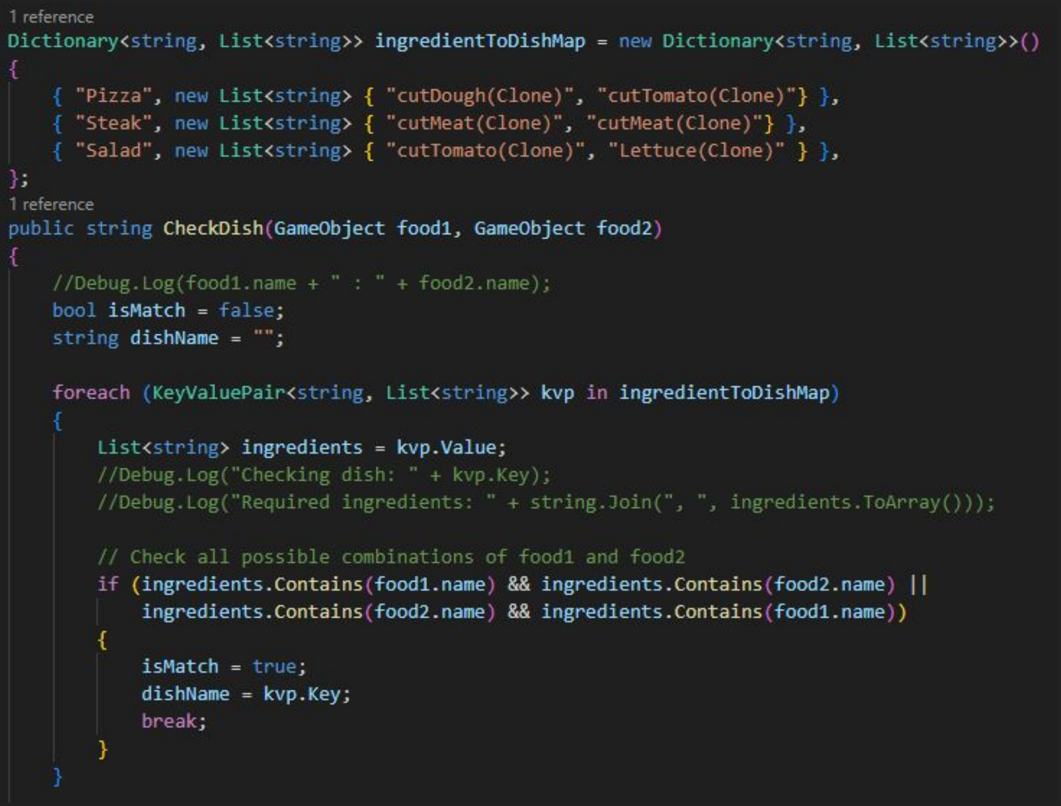
If that object is a food item, Un-parent it from the object, and parent it to the player.



However,

If the object is a foodbox, send the position of the player to checkFood which returns the player a food item

Recipe - Create dishes!

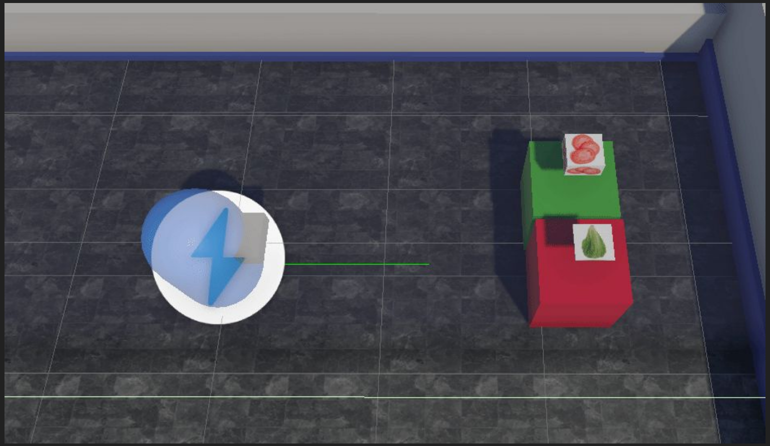


```
1 reference
Dictionary<string, List<string>> ingredientToDishMap = new Dictionary<string, List<string>>()
{
    { "Pizza", new List<string> { "cutDough(Clone)", "cutTomato(Clone)"} },
    { "Steak", new List<string> { "cutMeat(Clone)", "cutMeat(Clone)"} },
    { "Salad", new List<string> { "cutTomato(Clone)", "Lettuce(Clone)" } },
};
1 reference
public string CheckDish(GameObject food1, GameObject food2)
{
    //Debug.Log(food1.name + " : " + food2.name);
    bool isMatch = false;
    string dishName = "";

    foreach (KeyValuePair<string, List<string>> kvp in ingredientToDishMap)
    {
        List<string> ingredients = kvp.Value;
        //Debug.Log("Checking dish: " + kvp.Key);
        //Debug.Log("Required ingredients: " + string.Join(", ", ingredients.ToArray()));

        // Check all possible combinations of food1 and food2
        if (ingredients.Contains(food1.name) && ingredients.Contains(food2.name) ||
            ingredients.Contains(food2.name) && ingredients.Contains(food1.name))
        {
            isMatch = true;
            dishName = kvp.Key;
            break;
        }
    }
```

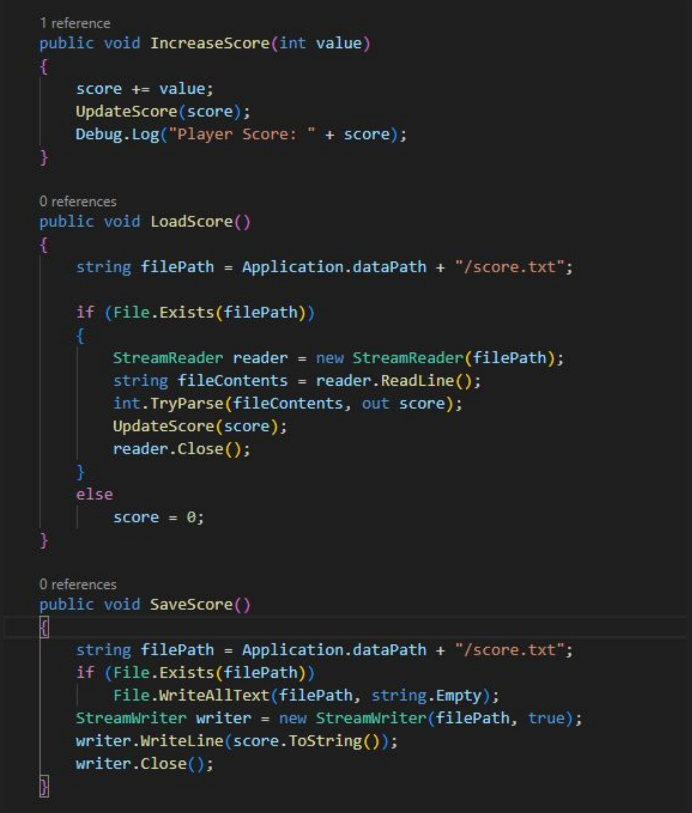
Uses a dictionary structure to hold recipe data

When a player attempts to combine two items, checkDish will compare each them to each value in the dictionary. If successful, it will return the key as a string.

Passes the data into a food class which **traverses through an array** to find which food to spawn



Long term data - Settling the score..

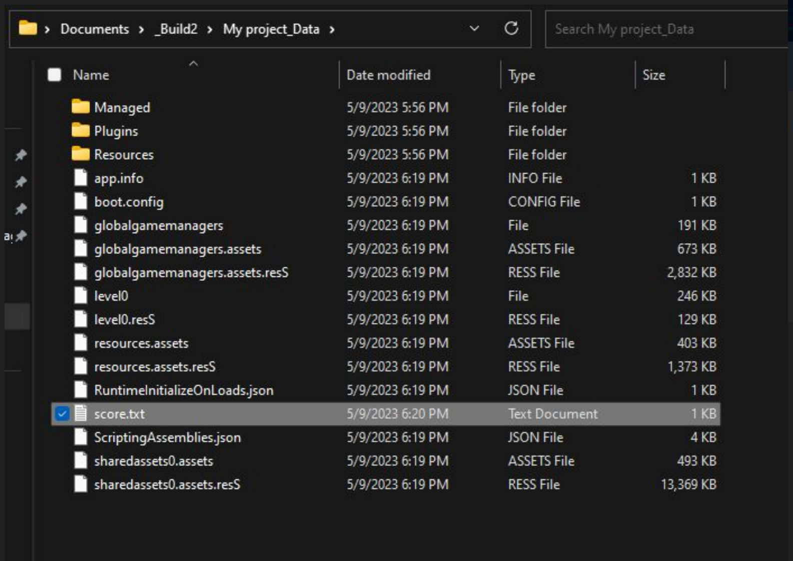


```
1 reference
public void IncreaseScore(int value)
{
    score += value;
    UpdateScore(score);
    Debug.Log("Player Score: " + score);
}

0 references
public void LoadScore()
{
    string filePath = Application.dataPath + "/score.txt";

    if (File.Exists(filePath))
    {
        StreamReader reader = new StreamReader(filePath);
        string fileContents = reader.ReadLine();
        int.TryParse(fileContents, out score);
        UpdateScore(score);
        reader.Close();
    }
    else
        score = 0;
}

0 references
public void SaveScore()
{
    string filePath = Application.dataPath + "/score.txt";
    if (File.Exists(filePath))
        File.WriteAllText(filePath, string.Empty);
    StreamWriter writer = new StreamWriter(filePath, true);
    writer.WriteLine(score.ToString());
    writer.Close();
}
```

Score is saved to a text file to store **long term data**.

- If player quits game, score is saved
- If player joins game, score is loaded from txt

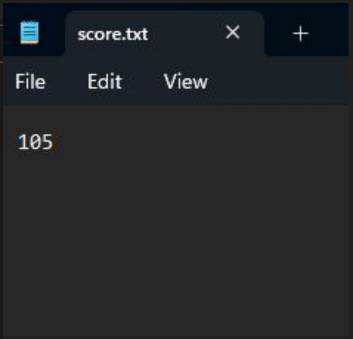
Turning in completed dishes will increase score



Documents › _Build2 › My project_Data ›

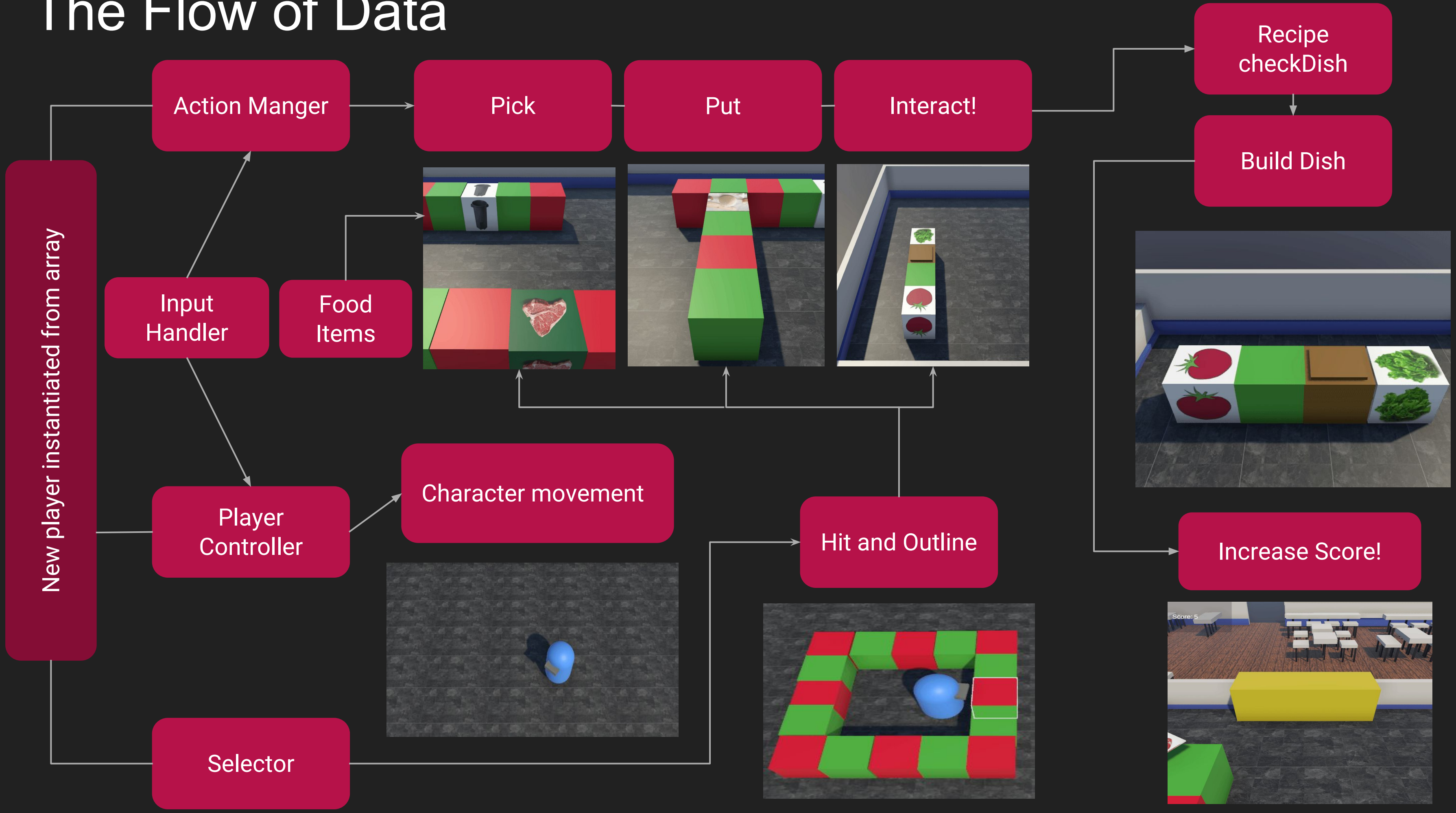
Search My project_Data

Name	Date modified	Type	Size
Managed	5/9/2023 5:56 PM	File folder	
Plugins	5/9/2023 5:56 PM	File folder	
Resources	5/9/2023 5:56 PM	File folder	
app.info	5/9/2023 6:19 PM	INFO File	1 KB
boot.config	5/9/2023 6:19 PM	CONFIG File	1 KB
globalgamemanagers	5/9/2023 6:19 PM	File	191 KB
globalgamemanagers.assets	5/9/2023 6:19 PM	ASSETS File	673 KB
globalgamemanagers.assets.resS	5/9/2023 6:19 PM	RESS File	2,832 KB
level0	5/9/2023 6:19 PM	File	246 KB
level0.resS	5/9/2023 6:19 PM	RESS File	129 KB
resources.assets	5/9/2023 6:19 PM	ASSETS File	403 KB
resources.assets.resS	5/9/2023 6:19 PM	RESS File	1,373 KB
RuntimeInitializeOnLoads.json	5/9/2023 6:19 PM	JSON File	1 KB
score.txt	5/9/2023 6:20 PM	Text Document	1 KB
ScriptingAssemblies.json	5/9/2023 6:19 PM	JSON File	4 KB
sharedassets0.assets	5/9/2023 6:19 PM	ASSETS File	493 KB
sharedassets0.assets.resS	5/9/2023 6:19 PM	RESS File	13,369 KB

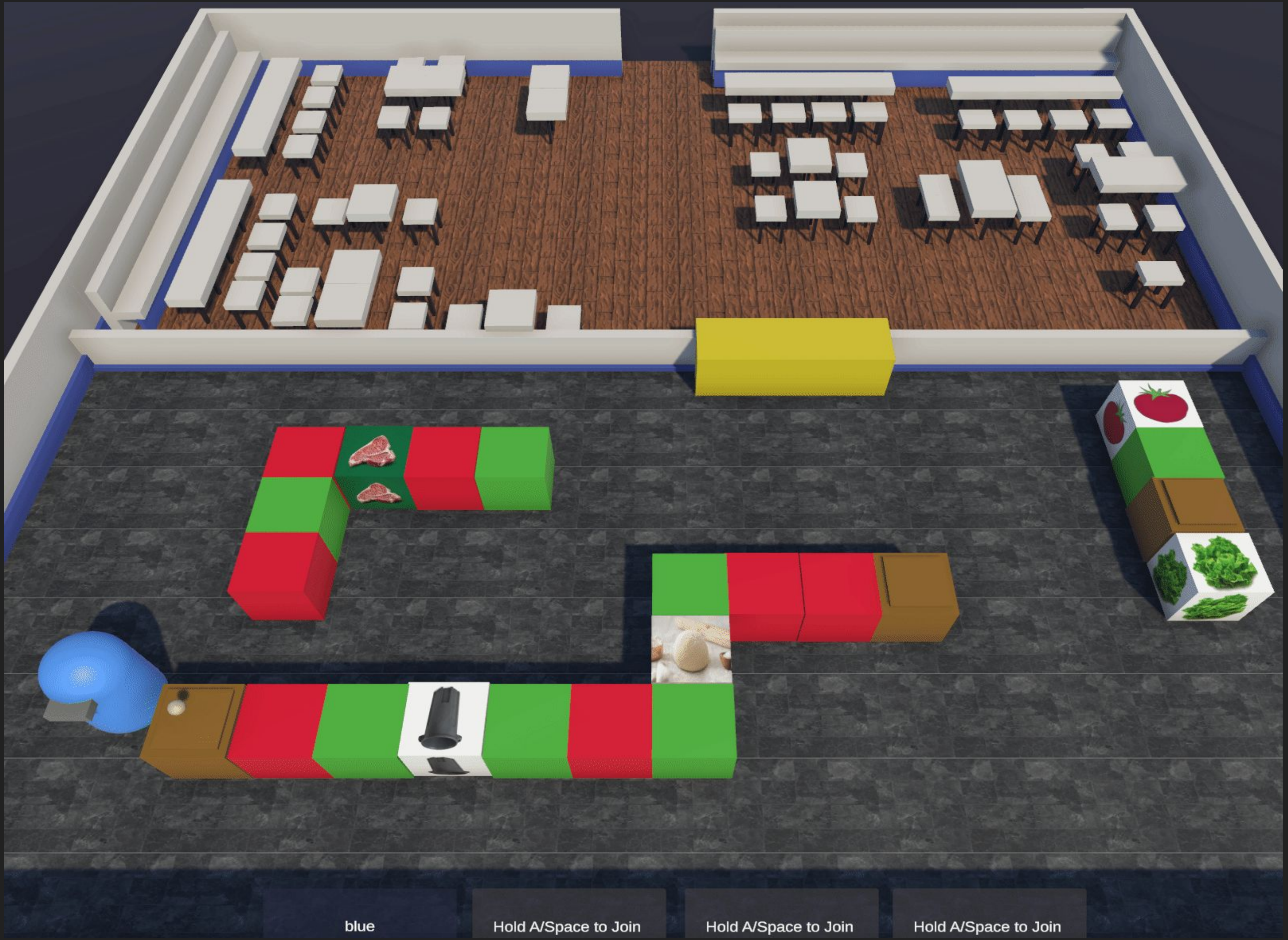


```
105
```

The Flow of Data



Final Demo

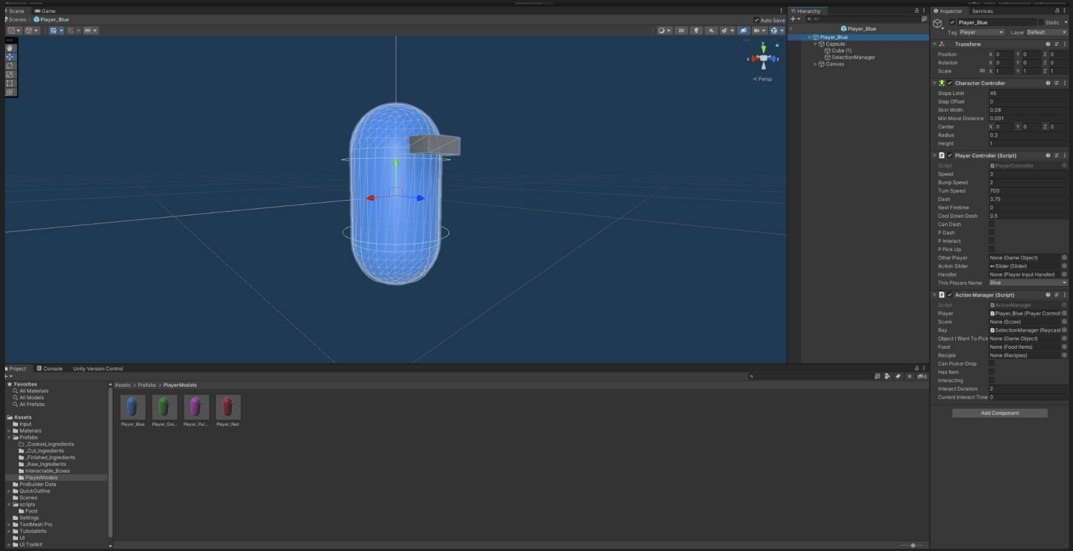

blue
Hold A/Space to Join
Hold A/Space to Join
Hold A/Space to Join

Future Implementations

- Order system - gives players orders to fulfil with a time limit
- Use actual textures instead of stock images
- Character Animations, VFX, and music
- Levels
- Title Screen and Pre-game area to select level
- More dishes!
- Allow player to make dishes
- Make the long term data store more information, like player names, or dish locations, level progress

Thank you

https://github.com/Cam-reee/OverPlate



```
Inspector   Services
Player_Blue                          Static
Tag Player            Layer Default
Transform
  Position              X 0    Y 0    Z 0
  Rotation              X 0    Y 0    Z 0
  Scale                 X 1    Y 1    Z 1
Character Controller
  Slope Limit           45
  Step Offset           0
  Skin Width            0.08
  Min Move Distance     0.001
  Center                X 0    Y 0    Z 0
  Radius                0.3
  Height                1
Player Controller (Script)
  Script                PlayerController
  Speed                 3
  Bump Speed            2
  Turn Speed            700
  Dash                  3.75
  Next Firetime         0
  Cool Down Dash        0.5
  Can Dash
  P Dash
  P Interact
  P Pick Up
  Other Player          None (Game Object)
  Action Slider         Slider (Slider)
  Handler               None (Player Input Handler)
  This Players Name     Blue
Action Manager (Script)
  Script                ActionManager
  Player                Player_Blue (Player Controll
  Score                 None (Score)
  Ray                   SelectionManager (Raycast
  Object I Want To Pick None (Game Object)
  Food                  None (Food Items)
  Recipie               None (Recipies)
  Can Pickor Drop
  Has Item
  Interacting
  Interact Duration     2
  Current Interact Time 0
```

I don't see a difference