Set Visual Studio as External Script Editor

1. Preferences
2. External Tools
3. Visual Studio 2022 [17.9.34]
Scripting in Unity

- Scripting allows developers to extend functionality
- Unity uses C#
- C# syntax is almost identical to what you learned in C++
- Typically scripts apply to a game object
Scripts as Behavior Components

- Scripts can be added to objects as components
- Scripts are used to create behavior
  - Change size or color
  - Apply intelligence to an object
Variables

{type} variableName = {value};

```csharp
using UnityEngine;
using System.Collections;

public class BasicScripting : MonoBehaviour {
    int number = 2; // Created a integer number
    string hello = "Hello World!"; // Created a String
    double bigNumber = 12314.324234; // Created a Double

    // Use this for initialization
    void Start () {
        Debug.Log (hello);
    }

    // Update is called once per frame
    void Update () {
    }
}
```
Functions

```csharp
1 using UnityEngine;
2 using System.Collections;
3
4 public class BasicScripting : MonoBehaviour {
5
6     int number = 2; // Created a integer number
7
8     // Multiplies returns the passed in number multiplied by 2
9     int multiplyByTwo(int number){
10        return number * 2;
11    }
12
13     // Use this for initialization
14     void Start () {
15        Debug.Log (multiplyByTwo(number));
16    }
17
18     // Update is called once per frame
19     void Update () {
20    }
21    }
```
using UnityEngine;
using System.Collections;

public class BasicScripting : MonoBehaviour {
    int number = 2; // Created a integer number

    // Multiplies returns the passed in number multiplied by 2
    int multiplyByTwo(int number){
        return number * 2;
    }

    // Use this for initialization
    void Start () {
        int answer = multiplyByTwo(number);
        if (answer > 0) {
            Debug.Log("Our answer is greater than zero!");
        } else {
            Debug.Log("Our answer is less than zero :( ");
        }
    }

    // Update is called once per frame
    void Update () {
    }
}
Loops

- Supports for, while, and do-while loops

```csharp
using UnityEngine;
using System.Collections;

public class BasicScripting : MonoBehaviour
{
    int number = 5; // Created a integer number

    // Use this for initialization
    void Start ()
    {
        for (int i = 0; i < number; i++)
        {
            Debug.Log("Times through the loop" + i);
        }
    }

    // Update is called once per frame
    void Update ()
    {
    }
}
```
Passing in Values Through Inspector

- Set a variable to public
- Or
- Set a variable as SerializeField
Built in Unity Functions

- Awake()
- Start()
- Update()
- Other functions

(https://docs.unity3d.com/Manual/ExecutionOrder.html)
Awake Function

- A default function in Unity objects
- Should be treated like a class constructor
- Called once, only once, upon initialization
Start Function

- Called after Awake() upon the first frame if and only if the object is enabled

- Called once and only once

- Called before Update()
Getting a Component

```csharp
using UnityEngine;
using System.Collections;

public class BasicScripting : MonoBehaviour {
    public Color color; // Variable is now setable in Inspector

    // Use this for initialization
    void Start () {
        Debug.Log (GetComponent<Transform> ().position);
    }

    // Update is called once per frame
    void Update () {
    }
}
```
Update Function

- Update() is called every frame when the object is enabled.
- This is the most used function in Unity.
- Time.deltaTime gives you the amount of time since Update() was called last. Use this for animating!
The Transform Component of an object holds its Position, Rotation, and Scale

Use GetComponent<>() to change these values

```csharp
using UnityEngine;
using System.Collections;

public class BasicScripting : MonoBehaviour {
    public Color color; // Variable is now setable in Inspector

    void Start () {
        Debug.Log (GetComponent<Transform> ().position);
    }

    void Update () {
    }
}
```
In Class Activity

- Create a first-person view “Player” for your farm
- Implement a script so that WASD keys control your player to move forward & backward and rotate left & right
- Bonus: Enable mouse control for smooth player rotation
- Be creative!