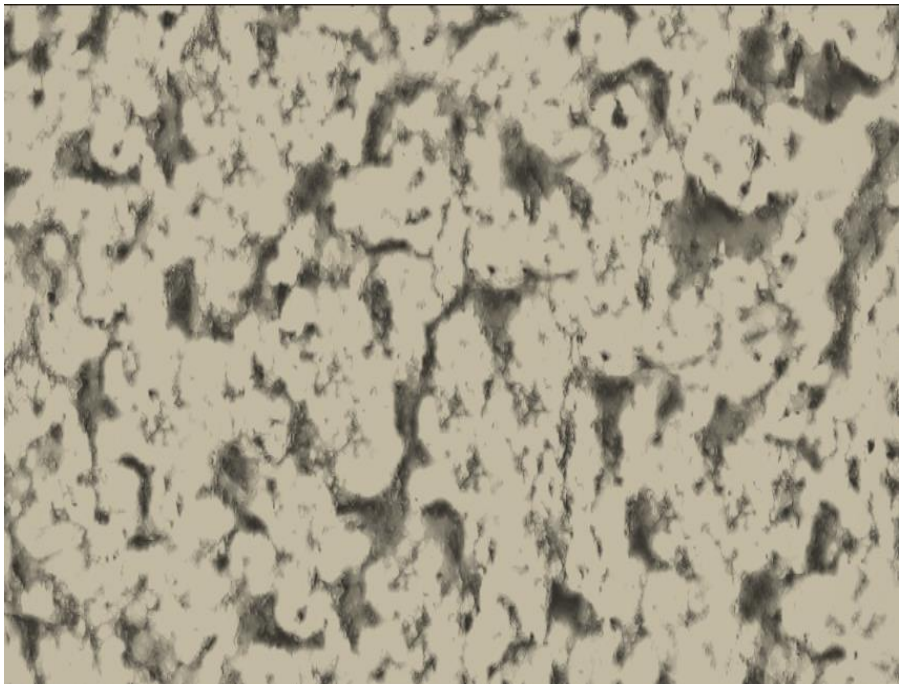
# Self-Reflective Report Specialism Briefs

## Instanced Scrolling Materials

### Programming

```
public class ScrollingTexture : MonoBehaviour
{
    public float speedX = 0.1f;
    public float speedY = 0.1f;
    private float curX;
    private float curY;
    Unity Message | 0 references
    private void Start()
    {
        curX = GetComponent<Renderer>().material.mainTextureOffset.x;
        curY = GetComponent<Renderer>().material.mainTextureOffset.y;
    }
    // Update is called once per frame
    Unity Message | 0 references
    void FixedUpdate()
    {
        curX += Time.deltaTime * speedX;
        curY += Time.deltaTime * speedY;
        GetComponent<Renderer>().material.SetTextureOffset("_MainTex", new Vector2(curX, curY));
    }
}
```

### Demo



## Brief Breakdown

The first Brief I have attended was the Scrolling Materials Brief. For this Brief I had to create a texture which scrolls based on time rather than FPS.

## What have I achieved & learnt from this Brief?

I have learnt to keep stuff as simple and efficient as possible when programming. Trying to find shortcuts and little loopholes in Programming can be very fun and essential. I have also learnt and picked up a good method of having a scrolling background which could be good for 2D games. I have also learnt not to rush things and work if you need to check something a couple of times you might as well do it rather than stare at solutions online.

## How would I have approached this Brief Differently?

If I were to do this Brief again, I would use the parallax scrolling method. This method uses a copy of a material a few times and then keeps moving in a direction and changes from one material to another. When the materials get close to being out of bounds and you can see game space, the scrolling would start again from the first material. This method is used on the main camera but the method I have currently used, is used on the actual plane that scrolls the material.

## How can this help me in future projects?

Scrolling material can be good for 2D games as well as to represent the texture of a sea in 3D. For 3D it can work well to represent waves but also it can represent fluids flowing on a flat plane. For 2D projects I can use this for Side-Scrollers or Endless runners. Once again this script is to be used on the background of 2D games.

# Framerate Counter

## Programming

```
Unity Script | 0 references
public class FramerateCounter : MonoBehaviour
{
    private int frameCounter = 0;
    private float timeCounter = 0.0f;
    private float refreshTime = 0.1f;
    private float minFrameRate = 0f;
    private float maxFrameRate = 0f;
    PerformanceCounter cpuCounter;
    PerformanceCounter ramCounter;

    int totalHits;
    [SerializeField]
    private Text framerateText;
    [SerializeField]
    private Text minframerateText;
    [SerializeField]
    private Text maxframerateText;
    [SerializeField]
    private Text cpuText;
    [SerializeField]
    private Text ramText;
    0 references
    public object getCPUCounter()
    {
        PerformanceCounter cpuCounter = new PerformanceCounter();
        cpuCounter.CategoryName = "Processor";
        cpuCounter.CounterName = "% Processor Time";
        cpuCounter.InstanceName = "_Total";
        dynamic firstValue = cpuCounter.NextValue();
        System.Threading.Thread.Sleep(1000);
        dynamic secondValue = cpuCounter.NextValue();
        return secondValue;
    }
```

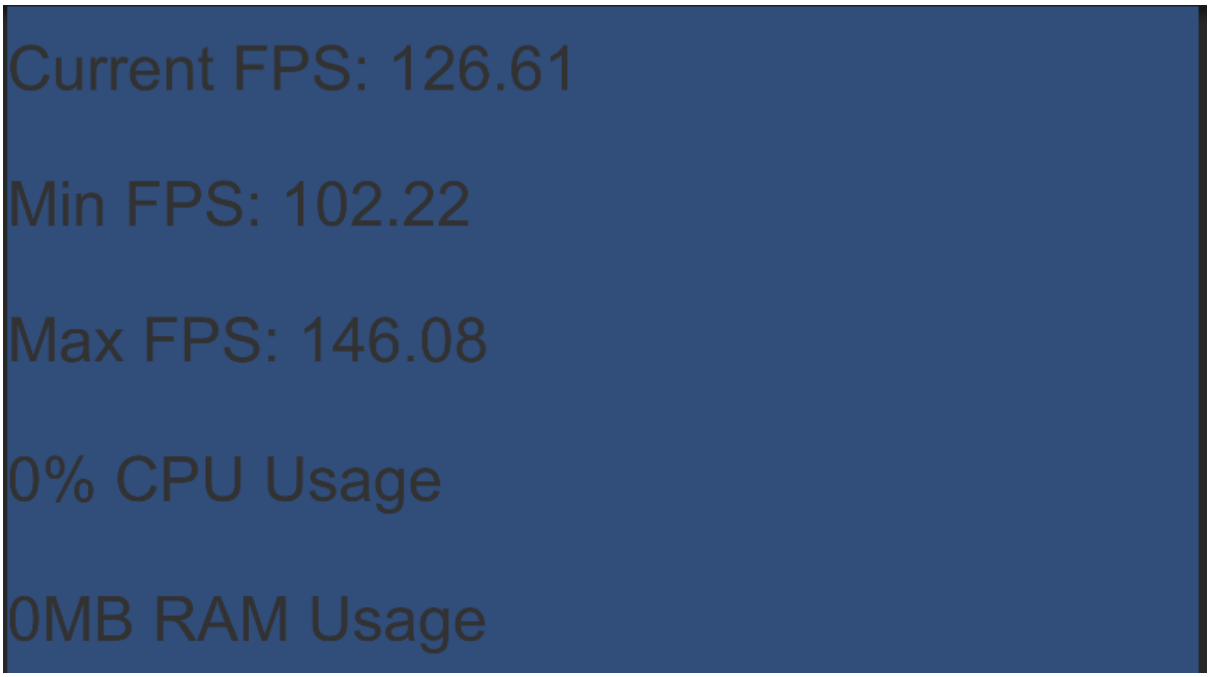
```
Unity Message | 0 references
private void Start()
{

    StartCoroutine(ResetMinFramerate());

}
1 reference
private IEnumerator ResetMinFramerate()
{
    yield return new WaitForSeconds(1f);
    minFrameRate = 1000f;
}
Unity Message | 0 references
void Update()
{
    cpuCounter = new PerformanceCounter("Processor", "% Processor Time", "_Total");
    ramCounter = new PerformanceCounter("Memory", "Available MBytes");
    if(timeCounter < refreshTime)
    {
        timeCounter += Time.deltaTime;
        frameCounter++;
    }
    else
    {
        float lastFramerate = frameCounter / timeCounter;
        if (minFrameRate > lastFramerate) minFrameRate = lastFramerate;
        if (maxFrameRate < lastFramerate) maxFrameRate = lastFramerate;
        frameCounter = 0;
        timeCounter = 0.0f;
        framerateText.text = "Current FPS: " + lastFramerate.ToString("n2");
        minframerateText.text ="Min FPS: " +  minFrameRate.ToString("n2");
        maxframerateText.text = "Max FPS: " + maxFrameRate.ToString("n2");
        cpuText.text = getCurrentCpuUsage() + " CPU Usage";
        ramText.text = getAvailableRAM() + " RAM Usage";


    }
}
1 reference
public string getCurrentCpuUsage()
{
    return cpuCounter.NextValue() + "%";
}
1 reference
public string getAvailableRAM()
{
    return ramCounter.NextValue() + "MB";
}
```

## Demo




## Brief Breakdown

The second Brief I have attended was the Scrolling Materials Brief. For this brief I had to make a counter for Frames Per Second which also shows the usage of the system and where the system fails to work properly

## What have I achieved & learnt from this Brief?

By creating this brief, I have learnt how games works out how well of a match the system you are using is for the game that you are playing. I have managed to achieve a working Framerate counter and processor & Memory tracker to see how much of the system the game uses constantly. This is good for players to determine whether they should upgrade their PCs anytime soon to get better performance on Games or the systems they are using are alright.

## How would I have approached this Brief Differently?

If I were to do this Brief again, I would use a Chart to export the history of the RAM Usage, CPU Usage, and Framerates and predict whether the game will keep up working the way it does or it would go up or down based on whether other programs are currently running while playing the game or the player sticks to only playing the game.

## How can this help me in future projects?

I could use this for stress tests on games to see how many Game Objects the game can take before it crashes and to check what kind of system it would take for the game to be run without any issues or crashes.

# Name Generator

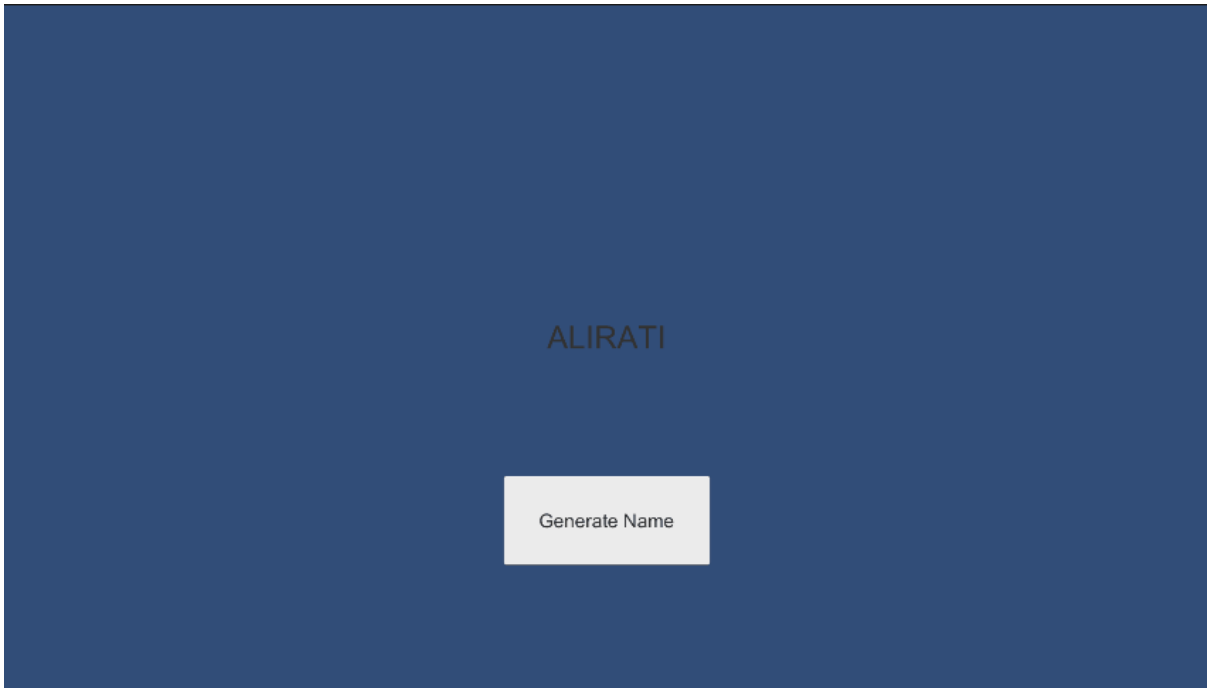
## Programming

```
using System.Text;
using UnityEngine;
using UnityEngine.UI;
Unity Script | 0 references
public class NameGenerator : MonoBehaviour
{
    public List<char> used = new List<char>();

    public Text generator;
    //public TextAsset textFile;
    int ranNum;
    0 references
    public void Generator()
    {

        Random random = new Random();
        string consonants = "BCDFGHJKLMNQRSTWXYZ";
        string vowels = "AEIOU";
        StringBuilder result = new StringBuilder();
        int NameLength = Random.Range(4,8);
        for(int i = 0; i <NameLength; i++)
        {
            char theChar;
            if(i%2 == 0)
            {
                theChar = vowels[Random.Range(0, vowels.Length)];
            }
            else
            {
                theChar = consonants[Random.Range(0, consonants.Length)];
            }
            result.Append(theChar);
        }
        generator.text = result.ToString();
    }
      }
```

## Demo




## Brief Breakdown

The last Brief I have attended was the Random Name Generator Brief. For this brief I had to make script which generates names which are obscene-free and there are no duplicates when running it.

## What have I achieved & learnt from this Brief?

By creating this brief, I have learnt how to use the Random Function a lot better and how to combine it with other types of functions and data types. I have also discovered how to use two different types of lists and get them randomised at the same time. With this Brief I have managed to create a script which can work for different types of projects as I will mention below.

## How would I have approached this Brief Differently?

I would have made it button based too.

## How can this help me in future projects?

This Script can be useful for RPG games where you need a name for your character as well for random points generated or random attack values for RPG Games. Also, 3-match games could use this script for combos as they would get a random points value for each combo they do between a specific range.