400 Update() calls vs one Update() call iterating 400 times

What would be more performant in Udon?

- 400 GameObjects executing some code every frame with an Update event :

```
public class EveryFrame : UdonSharpBehaviour
{
    void Update()
    {
        transform.position = Vector3.zero;
    }
}
```

- One GameObject with one Update event, but that Update event iterates though 400 GameObjects

```
public class EveryFrameHandler : UdonSharpBehaviour
{
    public EveryFrameCustomUpdate[] ArrayElements; //this array contains 400 elements

    void Update()
    {
        foreach(var el in ArrayElements)
        {
            el.CustomUpdate();
        }
    }
}

public class EveryFrameCustomUpdate : UdonSharpBehaviour
{
```

```
    public void CustomUpdate()
    {
        transform.position = Vector3.zero;
    }
}
```

I benchmarked it using the Udon Profiler by Merlin :
https://gist.github.com/MerlinVR/2da80b29361588ddb556fd8d3f3f47b5

> First example : 0.90ms per frame in average
> Second example : 2.06ms per frame in average

There are mostly two reasons explaining this difference :

- For-loops have a noticeable overhead, which I benchmarked here : https://vrclibrary.com/wiki/books/udon-benchmarking-and-performance-tests/page/for-loop
- Executing methods from a separate script also have a noticeable overhead : https://vrclibrary.com/wiki/books/udon-benchmarking-and-performance-tests/page/calling-methods-from-a-separate-script

Out of curiosity, I also replaced the for-loop with 400 lines of code I generated :

```
// ...
// ArrayElements[0~114].CustomUpdate();
ArrayElements[115].CustomUpdate();
ArrayElements[116].CustomUpdate();
ArrayElements[117].CustomUpdate();
ArrayElements[118].CustomUpdate();
ArrayElements[119].CustomUpdate();
ArrayElements[120].CustomUpdate();
// ArrayElements[120~399].CustomUpdate();
// ...
```

And I got 1.65ms per frame in average.

Revision #3
Created 30 March 2023 18:47:55 by MyroP
Updated 30 March 2023 20:09:19 by MyroP