

U# vs C#

U# is noticeably slower than regular C#, to benchmark it I decided to execute two scripts in U# and C#, one calculates the n-th Fibonacci number, the second one generates a maze.

To make those tests a bit fairer, I disable the C# compiler optimization, since the U# compiler does not optimize your scripts during compilation.

Recursive functions

For this test I executed a script that calculates the n-th Fibonacci number, the Fibonacci method looks like this

```
[RecursiveMethod]
public int FibonacciRecursive(int n)
{
    if (n <= 1)
        return n;
    else
        return FibonacciRecursive(n - 1) + FibonacciRecursive(n - 2);
}
```

Since `FibonacciRecursive` is a recursive function, the `[RecursiveMethod]` attribute need to be added.

This is really a horrible way to calculates the n-th Fibonacci number because certain method calls get called multiple times, and the number of method calls rise very quickly :

- `Fibonacci(1)` calls the Fibonacci method 1 time
- `Fibonacci(2)` calls the Fibonacci method 3 times
- `Fibonacci(10)` calls the Fibonacci method 177 times
- `Fibonacci(22)` calls the Fibonacci method 57313 time

So I felt like this test is a great way to determine how well U# performs this task, and I executed `Fibonacci(22)` in C# and U#

C# time : 0.189 ms
U# time : 684.577 ms
U# was 3629 times slower!

Maze generator

In my "Circuit Master" world, I wrote a custom maze generator algorithm, which I unfortunately cannot share because the algorithm is about 1600 lines long.

I decided to compare the execution time of that algorithm because that would be a more realistic test, it uses pretty much everything (Unity functions, custom list implementations, for-loops, bit manipulations etc.).

The algorithm does not call any recursive functions, but implements a custom stack made out of a ring buffer (https://en.wikipedia.org/wiki/Circular_buffer.) which is a very fast container to build and read a stack

C# time : 1.609 ms
U# time : 972.712 ms
U# was 604 times slower!

Based on the previous results, we can see that U# is much slower than regular C#, so keeping your code optimized is even more important in U#!

Revision #7

Created 2023-02-15 18:29:35 UTC by MyroP

Updated 2023-02-16 21:13:53 UTC by MyroP