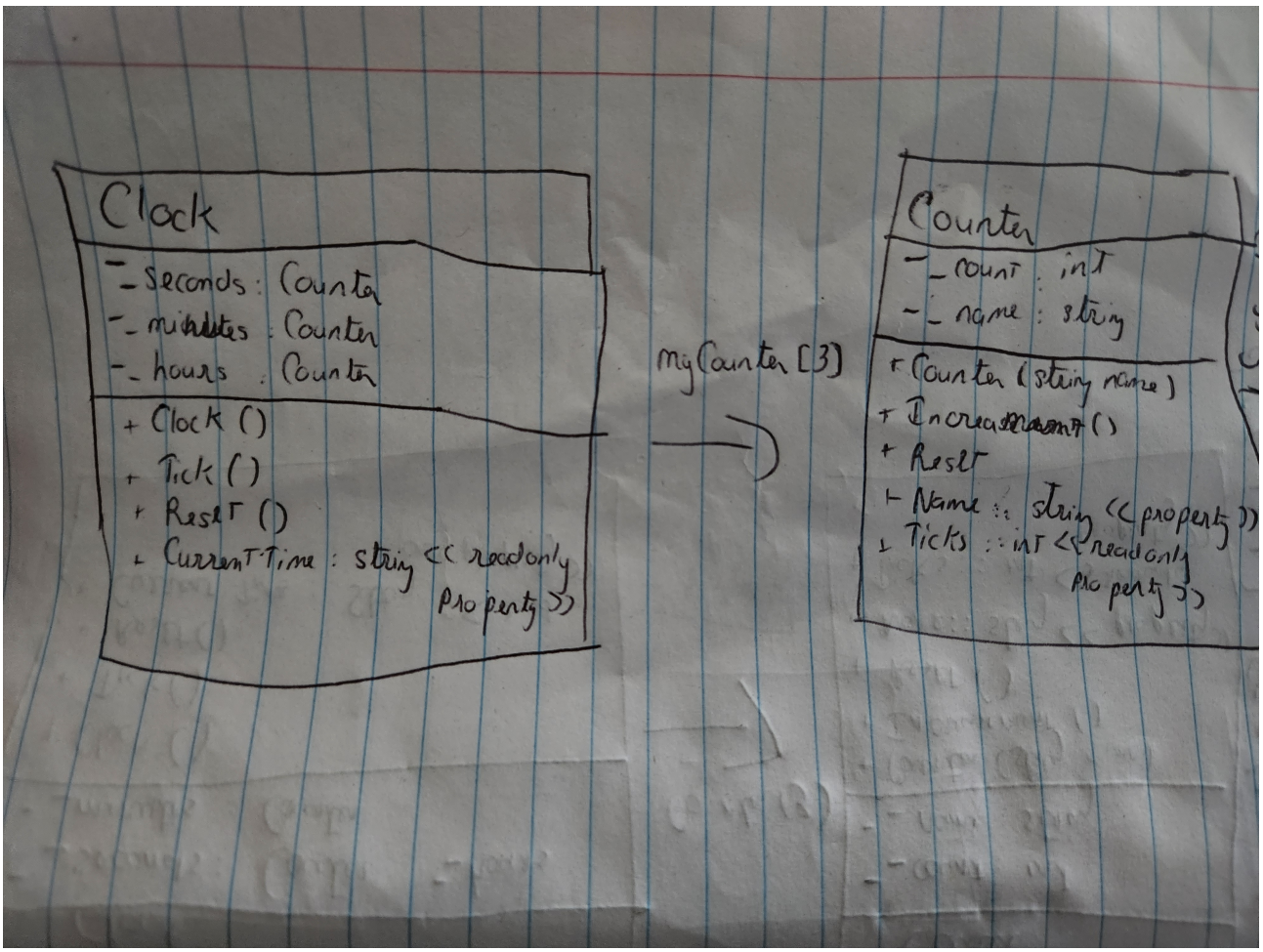
Swinburne University of Technology

COS20007 Object Oriented Programming

Clock Class

PDF generated at 13:45 on Tuesday 29th August, 2023


Clock
- seconds : Counter
- minutes : Counter
- hours : Counter
+ Clock ()
+ Tick ()
+ Reset ()
+ CurrentTime : string <<readonly Property>>
myCounter [3]
Counter
- count : int
- name : string
+ Counter (string name)
+ Increment ()
+ Reset
+ Name : string <<property>>
+ Ticks : int <<readonly Property>>

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Threading;


namespace ClockClass
{
    class Program
    {
        static void Main(string[] args)
        {
            Clock clock = new Clock();
            int i;

            for (i = 0; i < 86400; i++)
            {
                Thread.Sleep(1000);
                Console.Clear();
                clock.Tick();
                Console.WriteLine(clock.CurrentTime());
            }
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ClockClass
{

    public class Clock
    {
        private Counter _seconds;
        private Counter _minutes;
        private Counter _hours;

        public Clock()
        {
          _seconds = new Counter("secconds");
          _minutes = new Counter("minutes");
          _hours = new Counter("hours");
        }

        public void Tick()
        {
            _seconds.Increment();
            if (_seconds.Ticks == 60)
            {
                _minutes.Increment();
                _seconds.Reset();
                if (_minutes.Ticks == 60)
                {
                    _hours.Increment();
                    _minutes.Reset();
                    if (_hours.Ticks == 24)
                    {
                        Reset();
                    }
                }
            }
        }



        public void Reset()
        {
            _seconds.Reset();
            _minutes.Reset();
            _hours.Reset();
        }

        public string CurrentTime()
        {
            return $"{_hours.Ticks:D2}:{_minutes.Ticks:D2}:{_seconds.Ticks:D2}";
```

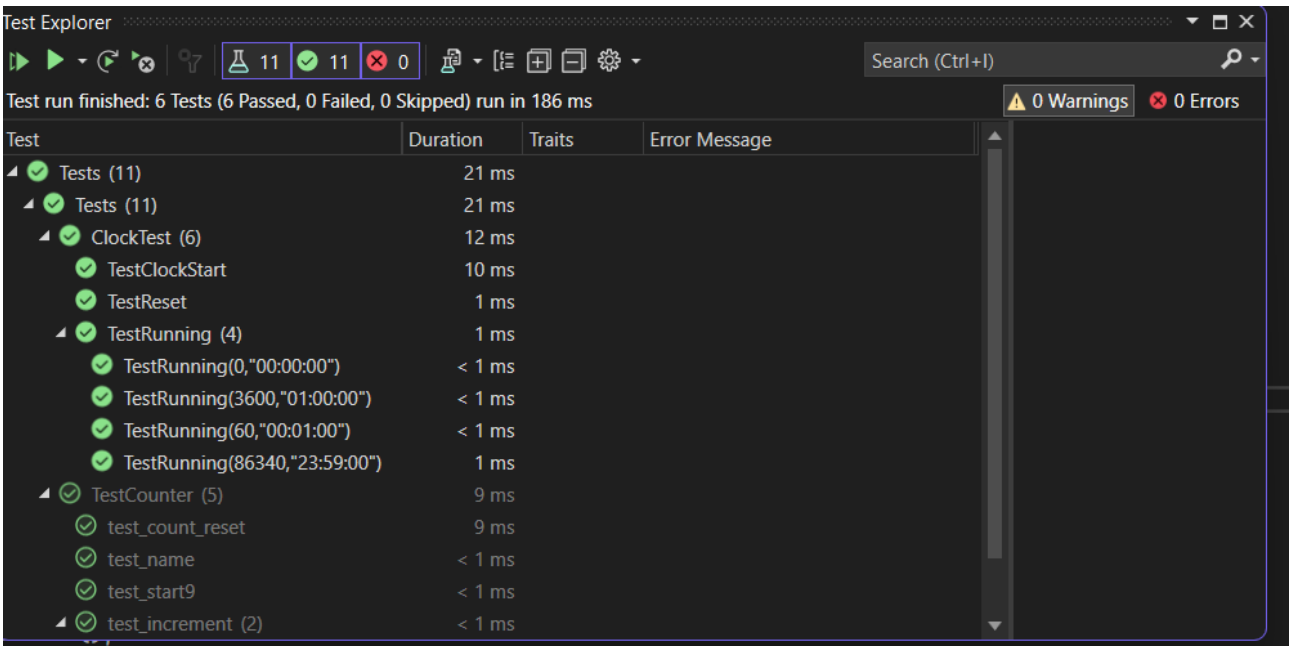
```
54            }
55        }
56    }
```

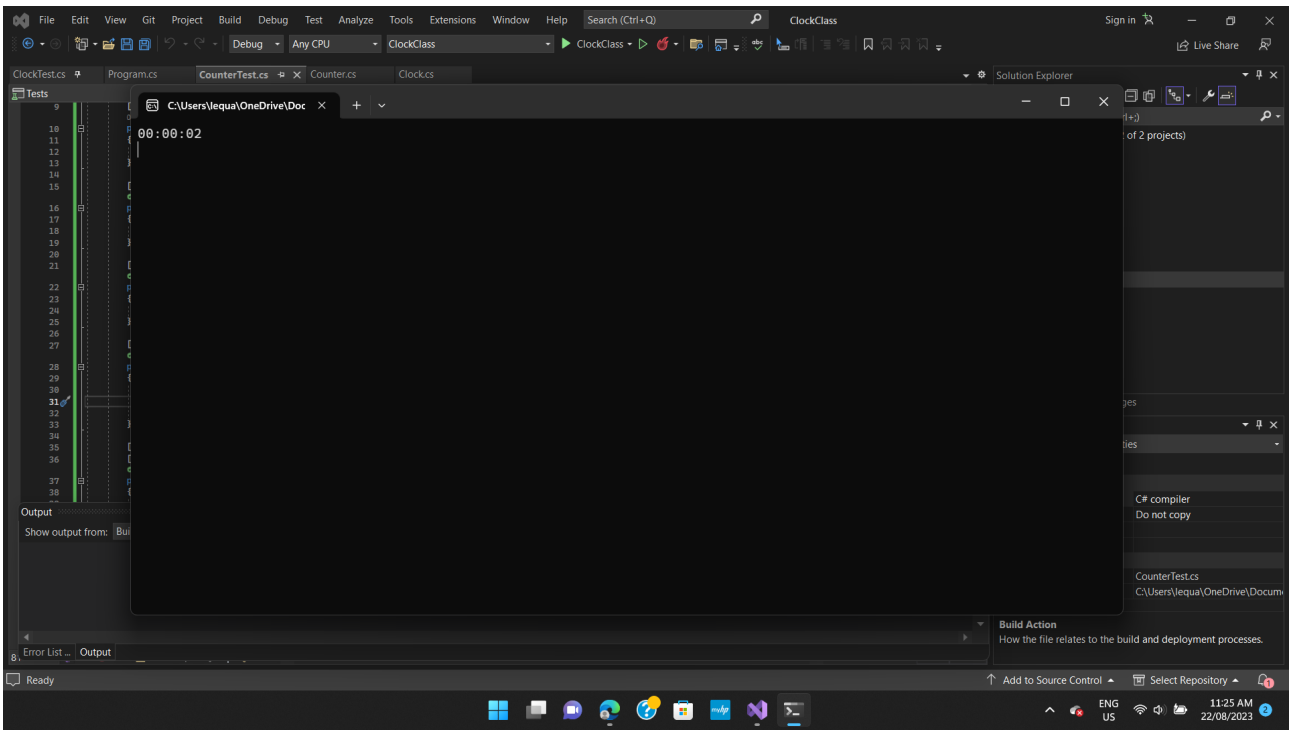
```
using NUnit.Framework;
using ClockClass;
namespace Tests
{
    public class ClockTest
    {
        private Clock _clock;

        [SetUp]
        public void Setup()
        {
            _clock = new Clock();
        }

        [Test]
        public void TestClockStart()
        {
            Assert.AreEqual("00:00:00", _clock.CurrentTime());
        }

        [Test]
        public void TestReset()
        {
            int i;
            for (i = 0; i < 86400; i++)
            {
                _clock.Tick();
            }
            _clock.Reset();
            Assert.AreEqual("00:00:00", _clock.CurrentTime());

        }

        [TestCase(0, "00:00:00")] //0 min
        [TestCase(60, "00:01:00")] //1 min
        [TestCase(3600, "01:00:00")] //1 hour
        [TestCase(86340, "23:59:00")] //1 day

        public void TestRunning(int tick, string currenttime)
        {
            int i;
            for (i = 0; i < tick; i++)
            {
                _clock.Tick();
            }
            Assert.AreEqual(currenttime, _clock.CurrentTime());

        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace ClockClass
{
    public class Counter
    {

        private int _count;
        private string _name;

        public Counter(string name)
        {
            _count = 0;
            _name = name;
        }

        public Counter(string name, int count)
        {
            _name = name;
            _count = count;
        }

        public void Reset()
        {
            _count = 0;
        }

        public void Increment()
        {
            _count++;
        }

        public string Name
        {
            get
            {
                return _name;
            }
            set
            {
                _name = value;
            }
        }

        public int Ticks
        {
            get
            {
                return _count;
            }
```

```
54            }
55        }
56    }
```

```
using NUnit.Framework;
using ClockClass;
namespace Tests
{
    public class TestCounter
    {
        private Counter _counterTest;

        [SetUp]
        public void Setup()
        {
            _counterTest = new Counter("Test");
        }

        [Test]
        public void test_start9()
        {
            Assert.AreEqual(0, _counterTest.Ticks);
        }

        [Test]
        public void test_name()
        {
            Assert.AreEqual("Test", _counterTest.Name);
        }

        [Test]
        public void test_count_reset()
        {
            _counterTest.Increment();
            _counterTest.Reset();
            Assert.AreEqual(0, _counterTest.Ticks);
        }

        [TestCase(60, 60)]
        [TestCase(100, 100)]
        public void test_increment(int Ticks, int result)
        {
            int i;
            for (i = 0; i < Ticks; i++)
            {
                _counterTest.Increment();
            }
            Assert.AreEqual(result, _counterTest.Ticks);
        }
    }
}
```


Test Explorer
11
11
0
Search (Ctrl+I)
Test run finished: 6 Tests (6 Passed, 0 Failed, 0 Skipped) run in 186 ms
0 Warnings
0 Errors
Test
Duration
Traits
Error Message
Tests (11) 21 ms
Tests (11) 21 ms
ClockTest (6) 12 ms
TestClockStart 10 ms
TestReset 1 ms
TestRunning (4) 1 ms
TestRunning(0,"00:00:00") < 1 ms
TestRunning(3600,"01:00:00") < 1 ms
TestRunning(60,"00:01:00") < 1 ms
TestRunning(86340,"23:59:00") 1 ms
TestCounter (5) 9 ms
test_count_reset 9 ms
test_name < 1 ms
test_start9 < 1 ms
test_increment (2) < 1 ms


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
ClockClass
Sign in
Live Share
Debug
Any CPU
ClockClass
ClockClass
ClockTest.cs
Program.cs
CounterTest.cs
Counter.cs
Clock.cs
Solution Explorer
Tests
C:\Users\lequa\OneDrive\Doc
00:00:02
of 2 projects)
Output
Show output from:
C# compiler
Do not copy
CounterTest.cs
Build Action
How the file relates to the build and deployment processes.
Error List ...
Output
Ready
Add to Source Control
Select Repository
ENG
US
11:25 AM
22/08/2023