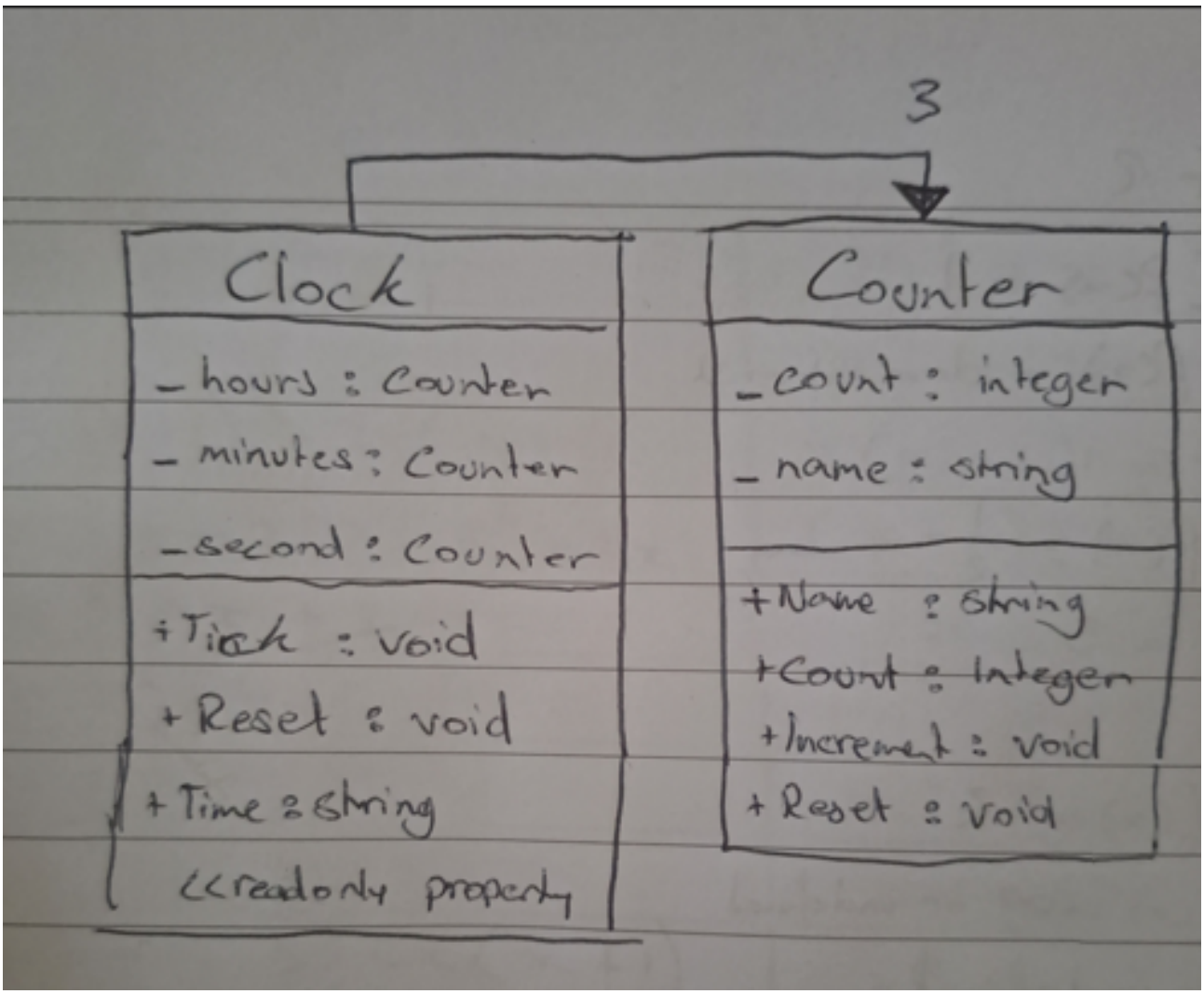
Swinburne University of Technology

COS20007 Object Oriented Programming

3.1P - Clock Class

PDF generated at 18:12 on Wednesday 12th April, 2023


3
Clock
- hours : Counter
- minutes : Counter
- second : Counter
+ Tick : void
+ Reset : void
+ Time : String
<<readonly property
Counter
- count : integer
- name : string
+ Name : String
+ Count : Integer
+ Increment : void
+ Reset : void

```
using System;
using CounterTask;
using Clock_Class;

namespace Clocks
{

    class MainClass
    {

        public static void Main(string[] args)
        {
            Clock main_clock = new Clock();

            for (int i = 0; i < 356; i++)
            {
                main_clock.Tick();
            }

            Console.WriteLine(main_clock.Time);
        }
    }
}
```

```
using System;
using CounterTask;

namespace Clock_Class
{

    public class Clock
    {

        private Counter _hours = new Counter("Hours");
        private Counter _minutes = new Counter("Minutes");
        private Counter _seconds = new Counter("Seconds");

        public void Tick()
        {

            _seconds.Increment();

            if (_seconds.Count == 60)
            {

                _seconds.Reset();

                _minutes.Increment();
            }

            if (_minutes.Count == 60)
            {

                _minutes.Reset();

                _hours.Increment();
            }

            if (_hours.Count == 24)
            {

                _hours.Reset();

            }
        }

        public void Reset()
        {

            _seconds.Reset();

            _minutes.Reset();

            _hours.Reset();
        }

        public string Time
```

```
54          {
55
56              get
57              {
58
59                  return String.Format("{0:00}:{1:00}:{2:00}", _hours.Count,
    ↪  _minutes.Count, _seconds.Count);
60              }
61          }
62
63      }
64
65  }
```

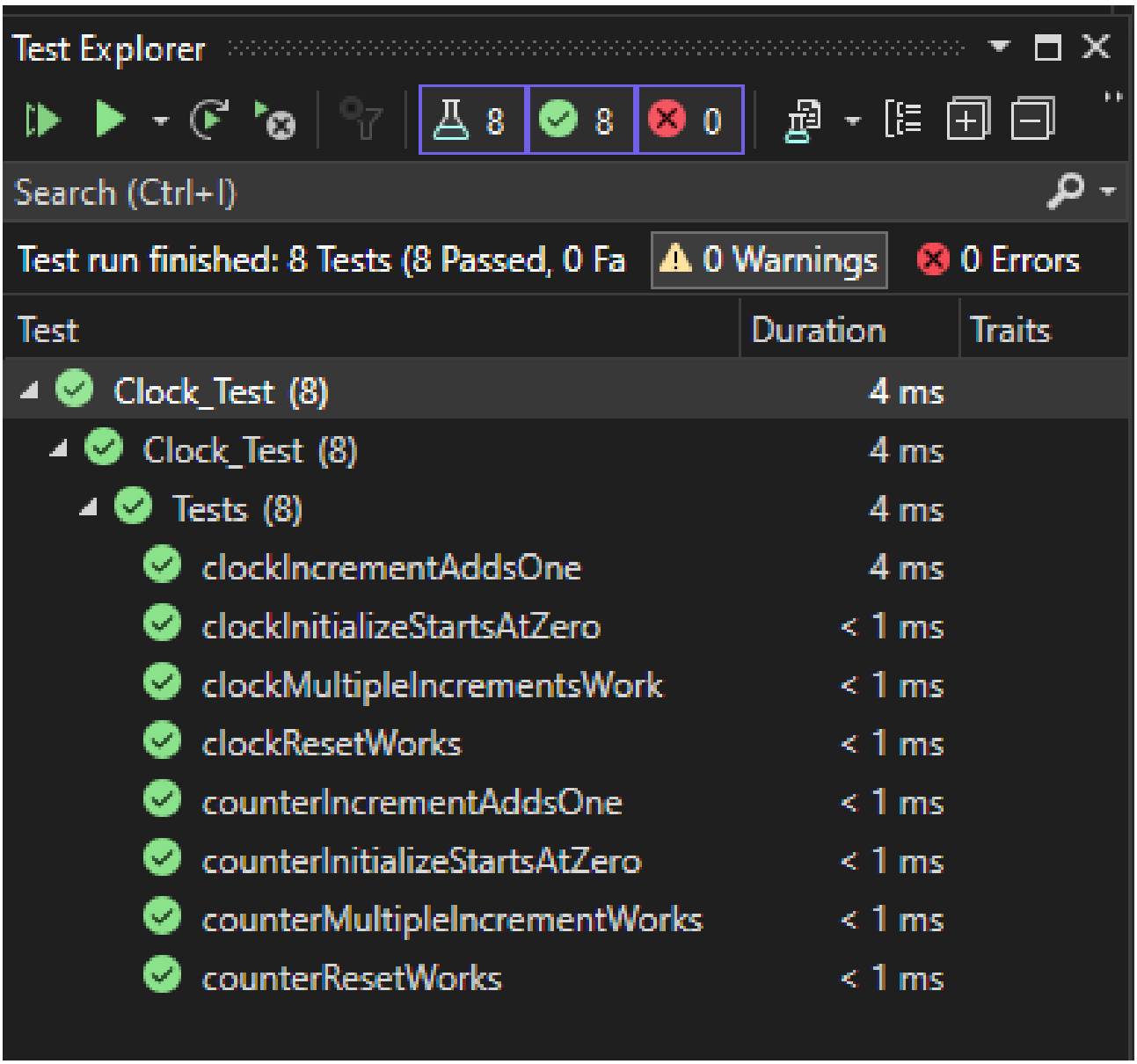
```
using Clock_Class;
using CounterTask;

namespace Clock_Test
{
    public class Tests
    {
        [SetUp]
        public void Setup()
        {
        }

        [Test]
        public void clockInitializeStartsAtZero()
        {

            Clock test_clock = new Clock();

            Assert.That(test_clock.Time == "00:00:00");
        }

        [Test]

        public void clockIncrementAddsOne()
        {
            Clock test_clock = new Clock();

            test_clock.Tick();

            Assert.That(test_clock.Time == "00:00:01");
        }

        [Test]

        public void clockMultipleIncrementsWork()
        {

            Clock test_clock = new Clock();

            for (int i = 0; i < 115; i++)
            {
                test_clock.Tick();
            }

            Assert.That(test_clock.Time == "00:01:55");
        }

        [Test]

        public void clockResetWorks()
        {
            Clock test_clock = new Clock();

```

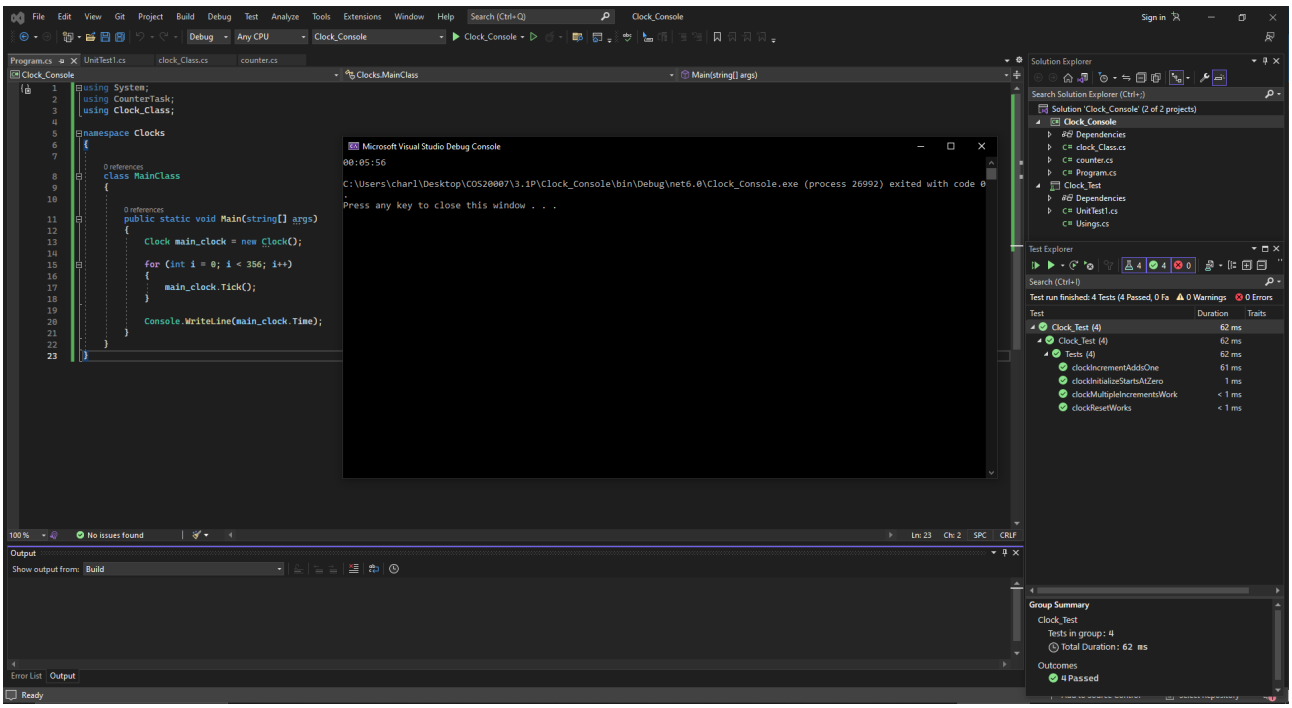
```
54              for (int i = 0; i < 115; i++)
55              {
56                  test_clock.Tick();
57              }
58
59              test_clock.Reset();
60
61              Assert.That(test_clock.Time == "00:00:00");
62          }
63
64      }
65  }
```

```
using System;

namespace CounterTask {

    public class Counter {

        private int _count;
        private string _name;

        public string Name {

            get {
                return _name;
            }

            set {
                _name = value;
            }

        }

        public int Count {

            get {
                return _count;
            }

            set {
                _count = value;
            }

        }

        public Counter(string name) {

            _name = name;

            _count = 0;

        }

        public void Increment() {

            _count ++;

        }

        public void Reset() {

            _count = 0;

        }

```

```
54
55      }
56  }
```

```
using Clock_Class;
using CounterTask;

namespace counter_Test
{
    internal class counter_Tests
    {
        [Test]

        public void counterInitializeStartsAtZero()
        {
            Counter test_counter = new Counter("test");

            Assert.That(test_counter.Count == 0);
        }

        [Test]
        public void counterIncrementAddsOne()
        {
            Counter test_counter = new Counter("test");

            test_counter.Increment();

            Assert.That(test_counter.Count == 1);
        }
        [Test]
        public void counterMultipleIncrementWorks()
        {
            Counter test_counter = new Counter("test");

            for (int i = 0; i < 15; i++) { test_counter.Increment(); }

            Assert.That(test_counter.Count == 15);

        }
        [Test]
        public void counterResetWorks()
        {
            Counter test_counter = new Counter("test");

            for (int i = 0; i < 15; i++) { test_counter.Increment(); }

            test_counter.Reset();

            Assert.That(test_counter.Count == 0);
        }
    }
}
```


Test Explorer
8
8
0
Search (Ctrl+I)
Test run finished: 8 Tests (8 Passed, 0 Fa
0 Warnings
0 Errors
Test
Duration
Traits
Clock_Test (8)
4 ms
Clock_Test (8)
4 ms
Tests (8)
4 ms
clockIncrementAddsOne
4 ms
clockInitializeStartsAtZero
< 1 ms
clockMultipleIncrementsWork
< 1 ms
clockResetWorks
< 1 ms
counterIncrementAddsOne
< 1 ms
counterInitializeStartsAtZero
< 1 ms
counterMultipleIncrementWorks
< 1 ms
counterResetWorks
< 1 ms


using System;
using CounterTask;
using Clock_Class;
namespace Clocks
{
class MainClass
{
public static void Main(string[] args)
{
Clock main_clock = new Clock();
for (int i = 0; i < 356; i++)
{
main_clock.Tick();
}
Console.WriteLine(main_clock.Time);
}
}
}
Microsoft Visual Studio Debug Console
00:05:56
C:\Users\charl\Desktop\COS20007\3.1P\Clock_Console\bin\Debug\net6.0\Clock_Console.exe (process 26992) exited with code 0
.
Press any key to close this window . . .
Solution Explorer
Solution 'Clock_Console' (2 of 2 projects)
Clock_Console
Dependencies
clock_Class.cs
counter.cs
Program.cs
Clock_Test
Dependencies
UnitTest1.cs
Usings.cs
Test Explorer
Clock_Test (4)
Clock_Test (4)
Tests (4)
clockIncrementAddsOne
clockInitializeStartsAtZero
clockMultipleIncrementsWork
clockResetWorks
Group Summary
Clock_Test
Tests in group: 4
Total Duration: 62 ms
Outcomes
4 Passed