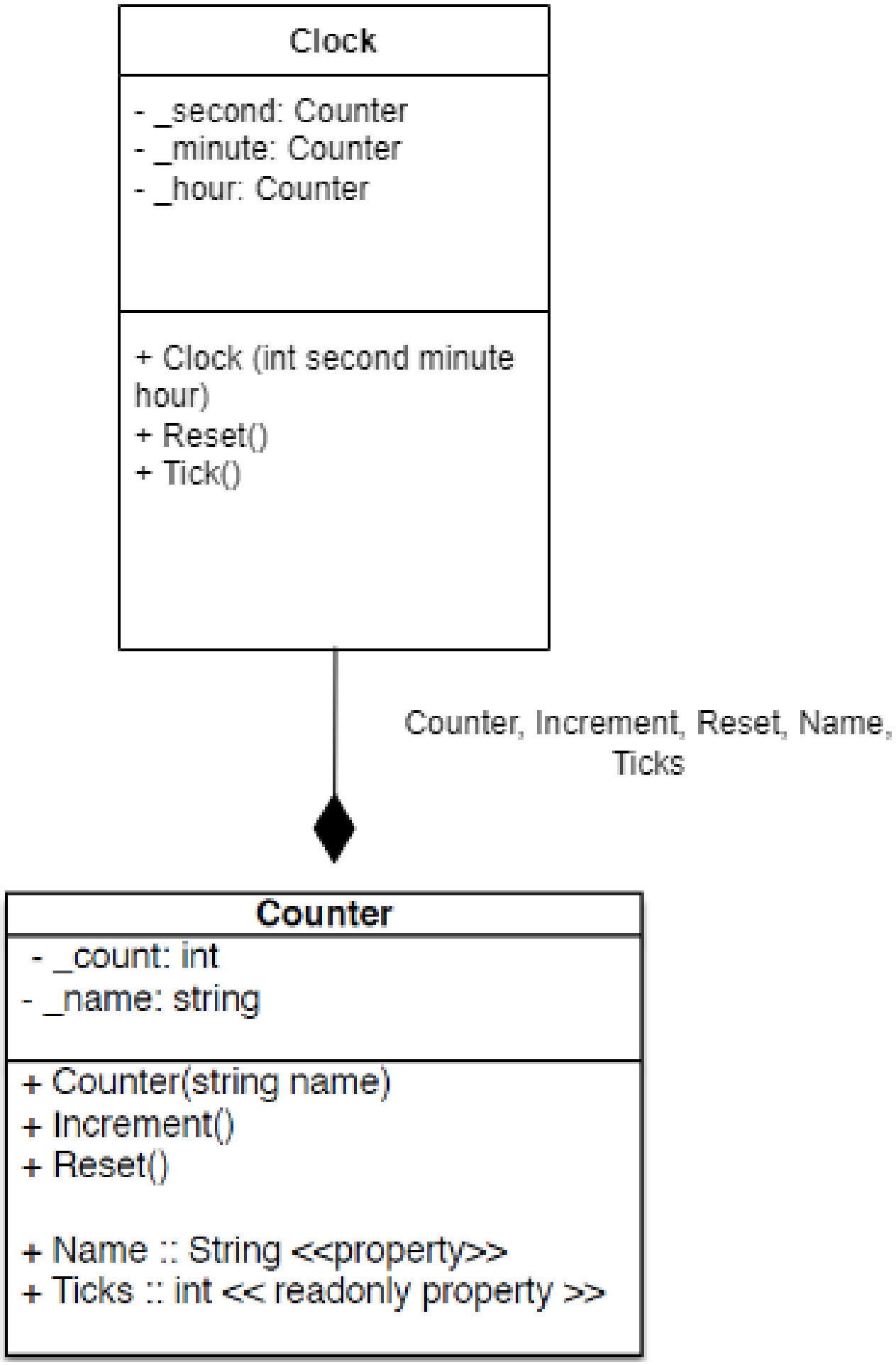
Swinburne University of Technology

COS20007 Object Oriented Programming

Clock Class

PDF generated at 20:05 on Saturday 23rd September, 2023


Clock
- _second: Counter
- _minute: Counter
- _hour: Counter
+ Clock (int second minute hour)
+ Reset()
+ Tick()
Counter, Increment, Reset, Name, Ticks
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;

namespace Program
{
    public class Program
    {
        public static void Main(string[] args)
        {
            Clock clock = new Clock();
            Console.WriteLine(clock.Show());
            clock.Tick();
            Console.WriteLine(clock.Show());
        }
    }
}
```

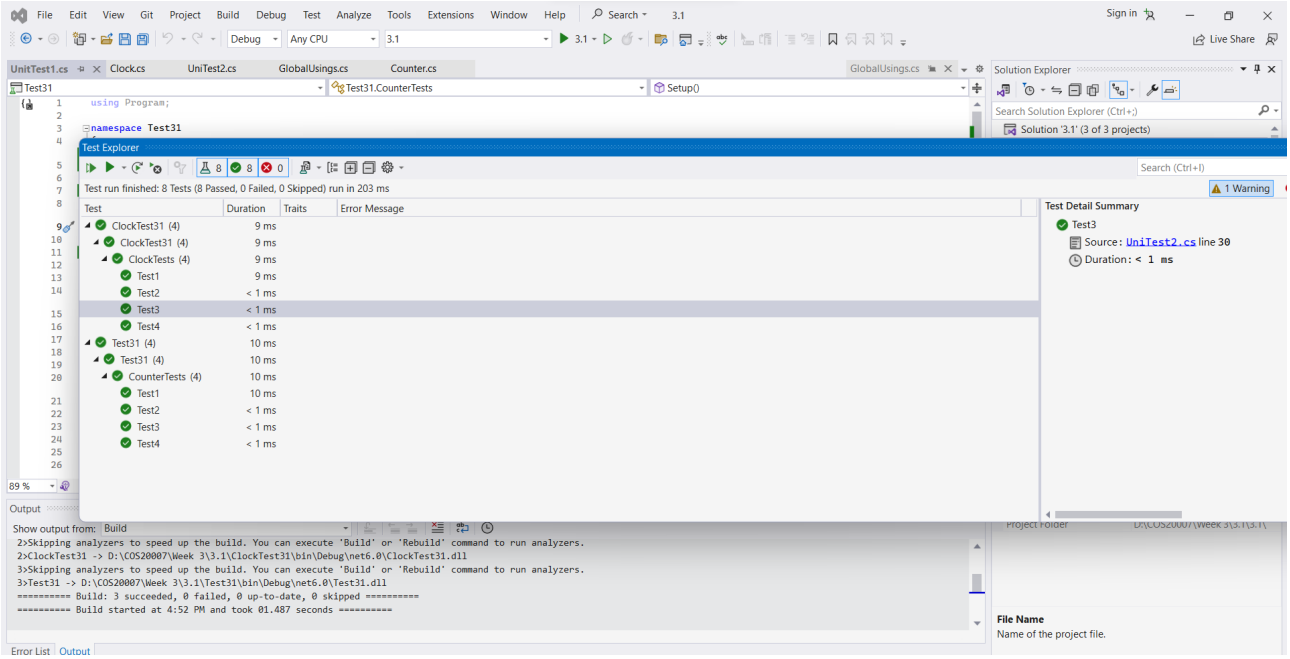
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Program
{
    public class Clock
    {
        private Counter seconds;
        private Counter minutes;
        private Counter hours;

        public Clock()
        {
            seconds = new Counter("Seconds");
            minutes = new Counter("Minutes");
            hours = new Counter("Hours");

        }


        public void Tick()
        {

            if (seconds.Ticks == 59)
            {
                if (minutes.Ticks == 59)
                {
                    if (hours.Ticks == 23)
                    {
                        ClockReset();
                    }
                    else
                    {
                        hours.Increment();
                        seconds.Reset();
                        minutes.Reset();
                    }
                }
                else
                {
                    minutes.Increment();
                    seconds.Reset();
                }
            }
            else
            {
                seconds.Increment();
            }


```

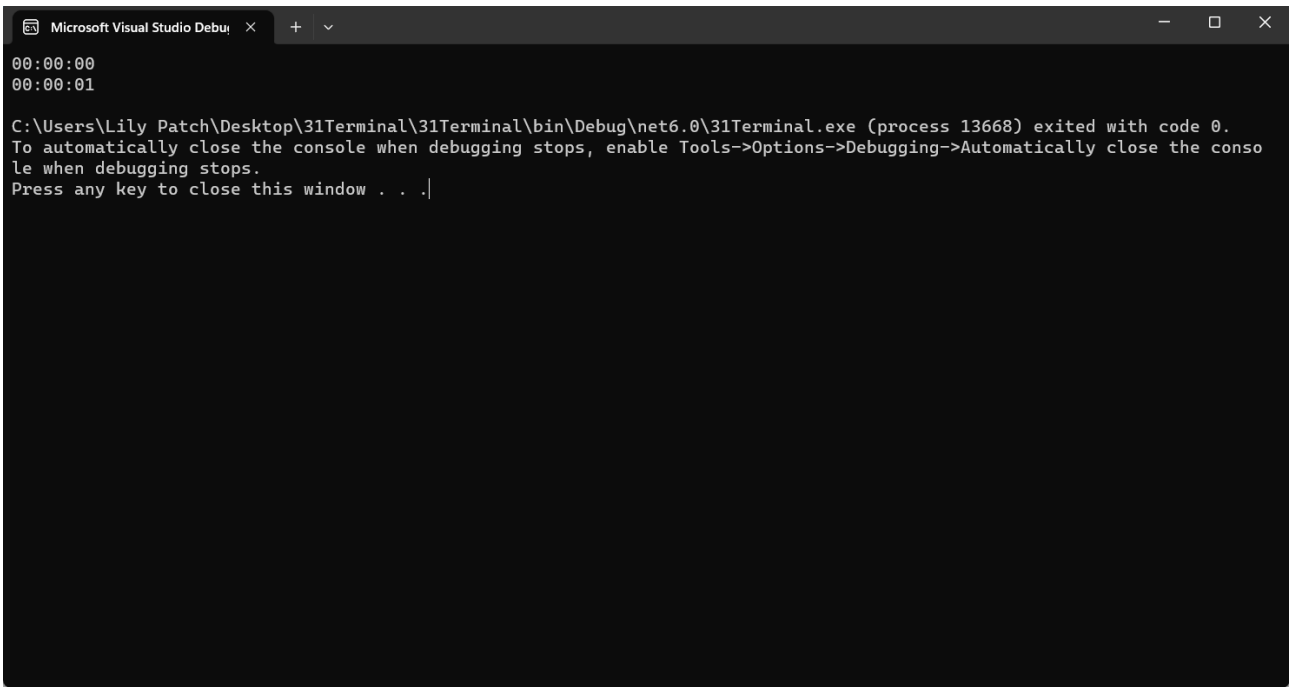
```
        }

        public string Show()
        {
            string _result = $"{hours.Ticks.ToString("D2")}:{minutes.Ticks.ToString("D2")}:{seconds.Ticks.ToString("D2")}
            return _result;
        }
        public void ClockReset()
        {
            seconds.Reset();
            minutes.Reset();
            hours.Reset();
        }
    }
}
```

```
using Program;

namespace ClockTest31
{
    public class ClockTests
    {
        Program.Clock _test;
        [SetUp]
        public void Setup()
        {
            _test = new Program.Clock();
        }

        [Test]
        public void Test1()
        {

            Assert.That(_test.Show(), Is.EqualTo("00:00:00"));
        }

        [Test]
        public void Test2()
        {
            _test.Tick();
            _test.Tick();
            Assert.That(_test.Show(), Is.EqualTo("00:00:02"));
        }

        [Test]
        public void Test3()
        {
            for (int i = 0; i <60;i++) { _test.Tick(); }
            Assert.That(_test.Show(), Is.EqualTo("00:01:00"));
        }

        [Test]
        public void Test4()
        {
            _test.Tick();
            _test.Tick();
            _test.ClockReset();
            Assert.That(_test.Show(), Is.EqualTo("00:00:00"));
        }
    }
}
```

```
using System.Security.Cryptography;
using System.Xml.Linq;


namespace Program
{
    public class Counter
    {
        private int _count;
        private string _name;

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

        public void Increment()
        {
            _count++;
        }
        public void Reset()
        {
            _count = 0;
        }

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

        public int Ticks
        {
            get => _count;
        }

    }









    }
```

```
using Program;

namespace Test31
{
    public class CounterTests
    {
        Program.Counter _test;
        [SetUp]
        public void Setup()
        {
            _test = new Program.Counter("");
        }

        [Test]
        public void Test1()
        {
            int _result = _test.Ticks;
            Assert.That(_result,Is.EqualTo(0));
        }
        [Test]
        public void Test2()
        {
            _test.Increment();
            Assert.That(1, Is.EqualTo(_test.Ticks));
        }
        [Test]
        public void Test3()
        {
            _test.Increment();
            _test.Increment();
            _test.Increment();
            Assert.That(3, Is.EqualTo(_test.Ticks));
        }

        [Test]
        public void Test4()
        {
            _test.Increment();
            _test.Reset();
            Assert.That(0, Is.EqualTo(_test.Ticks));
        }
    }
}
```


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
3.1
Sign in
Live Share
Debug
Any CPU
UnitTest1.cs
Clock.cs
UniTest2.cs
GlobalUsings.cs
Counter.cs
Test31
Test31.CounterTests
Setup()
using Program;
namespace Test31
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution '3.1' (3 of 3 projects)
Test Explorer
Search (Ctrl+I)
Test run finished: 8 Tests (8 Passed, 0 Failed, 0 Skipped) run in 203 ms
1 Warning
Test
Duration
Traits
Error Message
ClockTest31 (4) 9 ms
ClockTest31 (4) 9 ms
ClockTests (4) 9 ms
Test1 9 ms
Test2 < 1 ms
Test3 < 1 ms
Test4 < 1 ms
Test31 (4) 10 ms
Test31 (4) 10 ms
CounterTests (4) 10 ms
Test1 10 ms
Test2 < 1 ms
Test3 < 1 ms
Test4 < 1 ms
Test Detail Summary
Test3
Source: UniTest2.cs line 30
Duration: < 1 ms
Output
Show output from: Build
2>Skipping analyzers to speed up the build. You can execute 'Build' or 'Rebuild' command to run analyzers.
2>ClockTest31 -> D:\COS20007\Week 3\3.1\ClockTest31\bin\Debug\net6.0\ClockTest31.dll
3>Skipping analyzers to speed up the build. You can execute 'Build' or 'Rebuild' command to run analyzers.
3>Test31 -> D:\COS20007\Week 3\3.1\Test31\bin\Debug\net6.0\Test31.dll
========== Build: 3 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
========== Build started at 4:52 PM and took 01.487 seconds ==========
File Name
Name of the project file.
Error List
Output


Microsoft Visual Studio Debu
00:00:00
00:00:01
C:\Users\Lily Patch\Desktop\31Terminal\31Terminal\bin\Debug\net6.0\31Terminal.exe (process 13668) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .