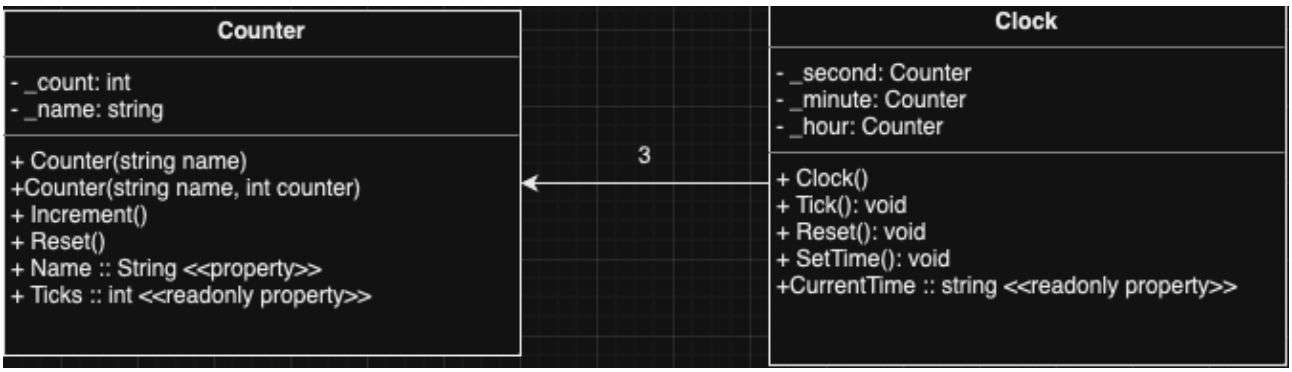
Swinburne University of Technology

COS20007 Object Oriented Programming

Clock Class

PDF generated at 15:56 on Friday 11th August, 2023


Counter
- _count: int
- _name: string
+ Counter(string name)
+Counter(string name, int counter)
+ Increment()
+ Reset()
+ Name :: String <<property>>
+ Ticks :: int <<readonly property>>
3
Clock
- _second: Counter
- _minute: Counter
- _hour: Counter
+ Clock()
+ Tick(): void
+ Reset(): void
+ SetTime(): void
+CurrentTime :: string <<readonly property>>

```
using System;
using System.Threading;
namespace ClockClassMain
{
    public class Program
    {
        public static void Main(string[] args)
        {
            Clock c = new Clock();
            while(true)
            {
                Console.WriteLine(c.CurrentTime + "\n");
                c.Tick();
                Thread.Sleep(1000);

            }
        }
    }
}
```

```
using System;
using CounterTask;
namespace ClockClassMain
{
    public class Clock
    {
        private Counter _second;
        private Counter _minute;
        private Counter _hour;

        public Clock()
        {
            _second = new Counter("second");
            _minute = new Counter("minute");
            _hour = new Counter("hour");
        }

        public void Tick()
        {
            _second.Increment();
            if(_second.Ticks > 59)
            {
                _minute.Increment();
                _second.Reset();
                if(_minute.Ticks > 59)
                {
                    _hour.Increment();
                    _minute.Reset();
                    if(_hour.Ticks > 23)
                    {
                        Reset();
                    }
                }
            }
        }

        public void Reset()
        {
            _second.Reset();
            _minute.Reset();
            _hour.Reset();
        }

        public void SetTime(string time)
        {
            // "dd:mm:ss"
            string[] parsed_time = time.Split(":");
            _hour = new Counter("hour", int.Parse(parsed_time[0]));
            _minute = new Counter("minute", int.Parse(parsed_time[1]));
            _hour = new Counter("second", int.Parse(parsed_time[2]));
        }

        public string CurrentTime
```

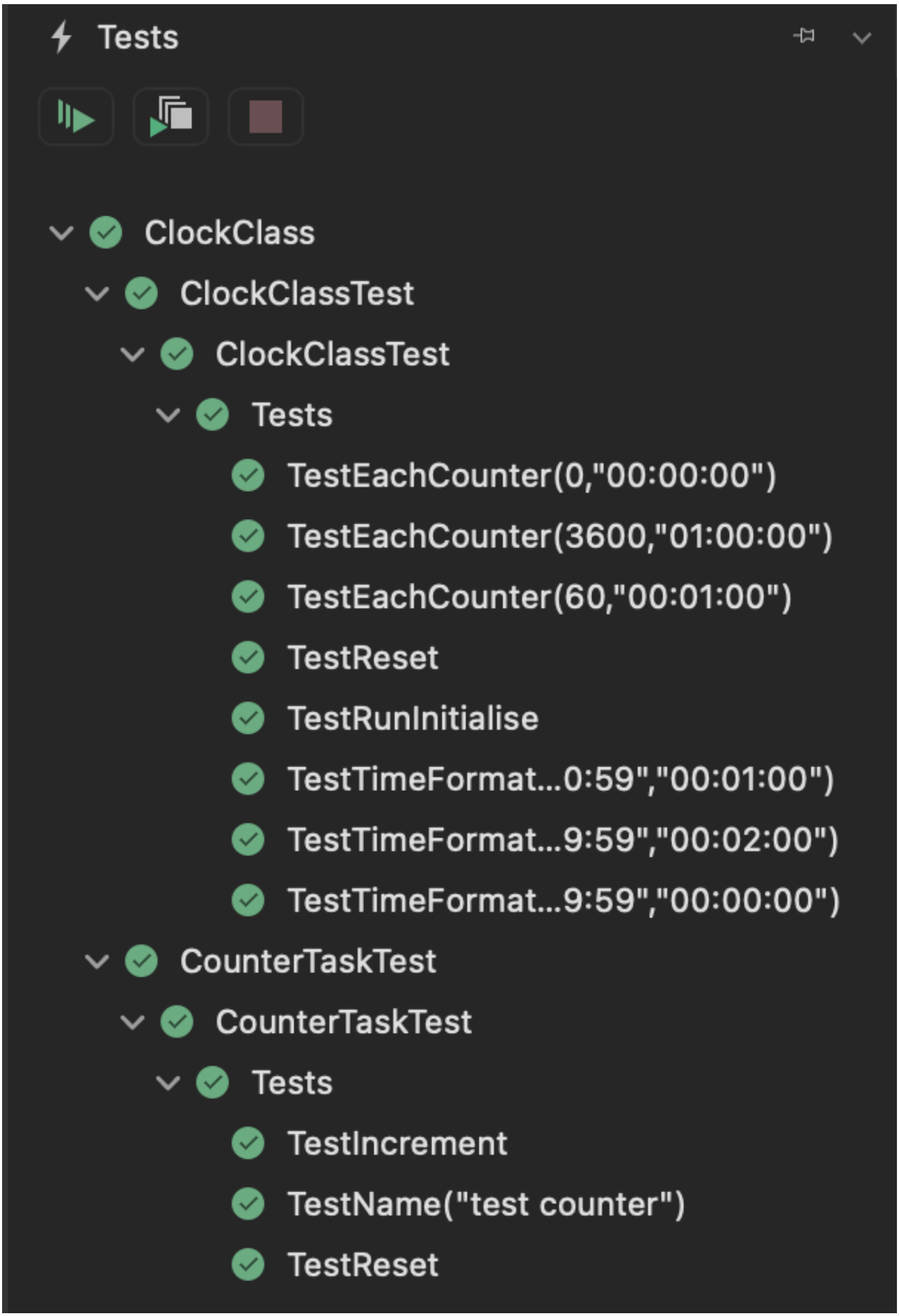
```
54          {
55              get
56              {
57                  return $"{_hour.Ticks:D2}:{_minute.Ticks:D2}:{_second.Ticks:D2}";
58              }
59          }
60      }
61  }
62
```

```
/*
 * Usings.cs
   global using ClockClassMain;
   global using NUnit.Framework;
 */

namespace ClockClassTest;

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

    [Test]
    public void TestRunInitialise()
    {
        Assert.That(_clock.CurrentTime, Is.EqualTo("00:00:00"));
    }

    [Test]
    public void TestReset()
    {
        const int MAX = 15000;
        for(int i = 0; i <= MAX; i++)
        {
            _clock.Tick();
        }
        _clock.Reset();
        Assert.That(_clock.CurrentTime, Is.EqualTo("00:00:00"));
    }

    [TestCase(0, "00:00:00")]
    [TestCase(60, "00:01:00")]
    [TestCase(3600, "01:00:00")]
    public void TestEachCounter(int tick, string curr_time)
    {
        for(int i = 0; i < tick; i++)
        {
            _clock.Tick();
        }
        Assert.That(_clock.CurrentTime, Is.EqualTo(curr_time));
    }

    [TestCase("00:00:59", "00:01:00")]
    [TestCase("01:59:59", "00:02:00")]
    [TestCase("23:59:59", "00:00:00")]
    public void TestTimeFormatWhenChange(string par_time, string expected_time)
    {
        _clock.SetTime(par_time);
```

```
54          _clock.Tick();
55          Assert.That(expected_time, Is.EqualTo(expected_time));
56      }
57
58  }
```

```
using System;
namespace CounterTask
{
    public class Counter
    {
        private int _count;
        private string _name;

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

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

        public void Increment()
        {
            _count += 1;
        }

        public void Reset()
        {
            _count = 0;
        }

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

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

```

```
/*
 * Usings.cs
 global using NUnit.Framework;
 global using CounterTask;
 */

namespace CounterTaskTest;

public class Tests
{
    private Counter _testCounter;
    [SetUp]
    public void Setup()
    {
        _testCounter = new Counter("test counter");
    }

    [Test]
    public void TestIncrement()
    {
        _testCounter.Increment();
        Assert.That(_testCounter.Ticks, Is.EqualTo(1));
    }

    [Test]
    public void TestReset()
    {
        const int MAX = 15000;
        for(int i = 0; i < MAX; i++)
        {
            _testCounter.Increment();
        }
        _testCounter.Reset();
        Assert.That(_testCounter.Ticks, Is.EqualTo(0));
    }

    [TestCase("test counter")]
    public void TestName(object obj)
    {
        Assert.That(_testCounter.Name, Is.EqualTo(obj));
    }
}
```


Tests
ClockClass
ClockClassTest
ClockClassTest
Tests
TestEachCounter(0,"00:00:00")
TestEachCounter(3600,"01:00:00")
TestEachCounter(60,"00:01:00")
TestReset
TestRunInitialise
TestTimeFormat...0:59","00:01:00")
TestTimeFormat...9:59","00:02:00")
TestTimeFormat...9:59","00:00:00")
CounterTaskTest
CounterTaskTest
Tests
TestIncrement
TestName("test counter")
TestReset


00:00:55
00:00:56
00:00:57
00:00:58
00:00:59
00:01:00
00:01:01
00:01:02
00:01:03
00:01:04
00:01:05