

Clock.cs

...\PC\Desktop\COS20007\Week_3\3.1P\CounterTask\Clock.cs

1

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace CounterTask
8 {
9     public class Clock
10    {
11        private Counter _hours;
12        private Counter _minutes;
13        private Counter _seconds;
14        public Clock()
15        {
16            _hours = new Counter("");
17            _minutes = new Counter("");
18            _seconds = new Counter("");
19        }
20
21        public int Hours
22        {
23            get => _hours.Tick;
24        }
25
26        public int Minutes
27        {
28            get => _minutes.Tick;
29        }
30
31        public int Seconds
32        {
33            get => _seconds.Tick;
34        }
35
36        public void Tick()
37        {
38            if (_seconds.Tick < 59)
39            {
40                _seconds.Increment();
41            }
42            else if (_minutes.Tick < 59)
43            {
44                _seconds.Reset();
45                _minutes.Increment();
46            }
47            else if (_hours.Tick < 23)
48            {
49                _seconds.Reset();
```

```
50         _minutes.Reset();
51         _hours.Increment();
52     }
53     else
54     {
55         _hours.Reset();
56         _minutes.Reset();
57         _seconds.Reset();
58     }
59 }
60 public void Reset()
61 {
62     _hours.Reset();
63     _minutes.Reset();
64     _seconds.Reset();
65 }
66 public string PrintTime()
67 {
68     string currentTime = _hours.Tick.ToString("D2") + ":" +
        _minutes.Tick.ToString("D2") + ":" + _seconds.Tick.ToString
        ("D2");
69     Console.WriteLine(currentTime);
70
71     return currentTime;
72 }
73 public void StartClock(int seconds)
74 {
75     for (int i = 0; i < seconds; i++)
76     {
77         Thread.Sleep(1000);
78         Tick();
79         PrintTime();
80     }
81 }
82 }
83 }
84
```

Counter.cs

...C\Desktop\COS20007\Week_3\3.1P\CounterTask\Counter.cs

1

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Security.Cryptography;
5 using System.Text;
6 using System.Threading.Tasks;
7
8 namespace CounterTask
9 {
10     public class Counter
11     {
12         private string _name;
13         public string Name
14         {
15             get => _name;
16             set => _name = value;
17         }
18
19         private int _count;
20         public int Tick
21         {
22             get => _count;
23         }
24
25         public Counter(string name)
26         {
27             _name = name;
28             _count = 0;
29         }
30         public int Increment()
31         {
32             return _count++;
33         }
34         public int Reset()
35         {
36             return _count = 0;
37         }
38     }
39 }
40
41
```

Program.cs

...C\Desktop\COS20007\Week_3\3.1P\CounterTask\Program.cs

1

```
1 namespace CounterTask
2 {
3     internal class Program
4     {
5         static void Main(string[] args)
6         {
7             Clock myClock = new Clock();
8             myClock.StartClock(56);
9         }
10    }
11 }
12
```

Program Output

The screenshot displays the Visual Studio IDE with a C# project named 'CounterTask'. The code in `Program.cs` defines a `namespace CounterTask` containing an `internal class Program` with a `Main` method. The `Main` method creates a `Clock` object, starts a timer, and prints timestamps from 00:00:01 to 00:00:09. The output window shows the following timestamps:

```
00:00:01
00:00:02
00:00:03
00:00:04
00:00:05
00:00:06
00:00:07
00:00:08
00:00:09
```

The bottom status bar indicates 'Build succeeded'.

```
namespace CounterTask
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Clock myClock = new Clock();
            myClock.StartTimer();
        }
    }
}
```

```
Build started at 11:23 PM...
1>----- Build started: Project: counterTask, Configuration: Debug Any CPU -----
1>Skipping analyzers to speed up the build. You can execute 'Build' or 'Rebuild' command to run analyzers.
1>CounterTask -> C:\Users\PC\Desktop\COS2007\Week 2\3.1P\3.1 - Answer\CounterTask\bin\Debug\net8.0\CounterTask.dll
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
===== Build completed at 11:23 PM and took 03.207 seconds =====
```

UnitTest - Clock

...C\Desktop\COS20007\Week_3\3.1P\TestClock\UnitTest1.cs

1

```
1 using CounterTask;
2 using System.Diagnostics.CodeAnalysis;
3
4 namespace TestClock
5 {
6     public class Tests
7     {
8         private Clock myClock;
9         [SetUp]
10        public void Setup()
11        {
12            myClock = new Clock();
13        }
14
15        [Test]
16        public void TestClockTick()
17        {
18            // Test clock tick less than 1 minute
19            for (int i = 0; i < 59; i++)
20            {
21                myClock.Tick();
22            }
23            Assert.That(myClock.Hours, Is.EqualTo(0));
24            Assert.That(myClock.Minutes, Is.EqualTo(0));
25            Assert.That(myClock.Seconds, Is.EqualTo(59));
26
27            // Test clock tick when reaching over 1 minute
28            for (int i = 0; i < 60; i++)
29            {
30                myClock.Tick();
31            }
32            Assert.That(myClock.Hours, Is.EqualTo(0));
33            Assert.That(myClock.Minutes, Is.EqualTo(1));
34            Assert.That(myClock.Seconds, Is.EqualTo(59));
35
36            // Test clock tick when reaching over 1 hour
37            for (int i = 0; i < 3600; i++)
38            {
39                myClock.Tick();
40            }
41            Assert.That(myClock.Hours, Is.EqualTo(1));
42            Assert.That(myClock.Minutes, Is.EqualTo(0));
43            Assert.That(myClock.Seconds, Is.EqualTo(0));
44
45            // Test clock tick when reaching over 24 hours
46            for (int i = 0; i < 3600*24; i++)
47            {
48                myClock.Tick();
49            }
```

```
50         Assert.That(myClock.Hours, Is.EqualTo(1));
51         Assert.That(myClock.Minutes, Is.EqualTo(1));
52         Assert.That(myClock.Seconds, Is.EqualTo(59));
53
54         Assert.Pass();
55     }
56
57     [Test]
58     public void TestReset()
59     {
60         for (int i = 0; i < 3676; i++)
61         {
62             myClock.Tick();
63         }
64
65         myClock.Reset();
66         Assert.That(myClock.Hours, Is.EqualTo(0));
67         Assert.That(myClock.Minutes, Is.EqualTo(0));
68         Assert.That(myClock.Seconds, Is.EqualTo(0));
69
70         Assert.Pass();
71     }
72
73     [Test]
74     public void TestTimeFormat()
75     {
76         for (int i = 0; i < 3676; i++)
77         {
78             myClock.Tick();
79         }
80         string a = myClock.PrintTime();
81         Assert.That(a, Is.EqualTo("01:01:16"));
82
83         Assert.Pass();
84     }
85 }
86 }
```

UnitTest Clock Output

The screenshot displays the Visual Studio IDE with a C# project named 'CounterTask'. The main window shows the source code for 'TestClockTests.cs', which includes a 'Tests' class with a 'Setup' method and two 'Test' methods. The first test, 'TestClock', verifies the initial state of the clock (0 hours, 0 minutes, 59 seconds). The second test, 'TestCounter', verifies the clock's state after 60 ticks (1 hour, 0 minutes, 0 seconds). The 'Test Explorer' window shows that all three tests passed successfully. The 'Output' window shows the build results, indicating that the build was successful and completed at 11:25 PM on 5/27/2024.

```
using CounterTask;
using System.Diagnostics.CodeAnalysis;

namespace TestClock
{
    public class Tests
    {
        [SetUp]
        public void Setup()
        {
            myClock = new Clock();
        }

        [Test]
        public void TestClockTick()
        {
            // Test clock tick less than 1 minute
            for (int i = 0; i < 59; i++)
            {
                myClock.Tick();
            }

            Assert.That(myClock.Hours, Is.EqualTo(0));
            Assert.That(myClock.Minutes, Is.EqualTo(0));
            Assert.That(myClock.Seconds, Is.EqualTo(59));

            // Test clock tick when reaching over 1 minute
            for (int i = 0; i < 60; i++)
            {
                myClock.Tick();
            }

            Assert.That(myClock.Hours, Is.EqualTo(1));
            Assert.That(myClock.Minutes, Is.EqualTo(0));
        }
    }
}
```

Test Explorer

Test	Duration	Traits	Error Message
TestClock (3)	21 ms		
TestCounter (3)	15 ms		

Group Summary

TestCounter

Tests in group: 3

Total Duration: 15

Outcomes

3 Passed

Output

Build started at 11:25 PM...

Build: 0 succeeded, 0 failed, 2 up-to-date, 0 skipped

Build completed at 11:25 PM and took 00.065 seconds

UnitTest Counter

...Desktop\COS20007\Week_3\3.1P\TestCounter\UnitTest1.cs

1

```
1 using CounterTask;
2
3 namespace TestCounter
4 {
5     public class Tests
6     {
7         private Counter myCounter;
8
9         [SetUp]
10        public void Setup()
11        {
12            myCounter = new Counter("My Counter");
13        }
14
15        [Test]
16        public void TestIncrement()
17        {
18            myCounter.Increment();
19
20            Assert.That(myCounter.Tick, Is.EqualTo(1));
21
22            Assert.Pass();
23        }
24
25        [Test]
26        public void TestIncrementMultiple()
27        {
28            myCounter.Increment();
29            myCounter.Increment();
30            myCounter.Increment();
31
32            Assert.That(myCounter.Tick , Is.EqualTo(3));
33
34            Assert.Pass();
35        }
36
37        [Test]
38        public void TestReset()
39        {
40            myCounter.Increment();
41
42            myCounter.Reset();
43
44            Assert.That(myCounter.Tick, Is.EqualTo(0));
45
46            Assert.Pass();
47        }
48    }
49 }
```

UnitTest Counter Output

The screenshot displays the Visual Studio IDE with the following components:

- Code Editor:** Shows the `TestCounterTests.cs` file with the following code:

```
using CounterTask;

namespace TestCounter
{
    public class Tests
    {
        private Counter myCounter;

        [SetUp]
        public void Setup()
        {
            myCounter = new Counter("My Counter");
        }

        [Test]
        public void TestIncrement()
        {
            myCounter.Increment();
            Assert.That(myCounter.Tick, Is.EqualTo(1));
            Assert.Pass();
        }

        [Test]
        public void TestIncrementMultiple()
        {
            myCounter.Increment();
            myCounter.Increment();
            myCounter.Increment();
            Assert.That(myCounter.Tick, Is.EqualTo(3));
        }
    }
}
```
- Test Explorer:** Shows the test results for the `TestCounter` test group. The test run finished with 3 tests passed, 0 failed, and 0 skipped, running in 114 ms. The test results table is as follows:

Test	Duration	Traits	Error Message
TestClock (3)	21 ms		
TestCounter (3)	15 ms		
- Output Window:** Shows the build log with the following text:

```
Build started at 11:25 PM...
***** Build: 0 succeeded, 0 failed, 2 up-to-date, 0 skipped *****
***** Build completed at 11:25 PM and took 00.074 seconds *****
```
- Solution Explorer:** Shows the project structure for `CounterTask`, including `CounterTask`, `Clocks`, `Counter.cs`, `Program.cs`, `TestClock`, `TestCounter`, and `UnitTest1.cs`.

UML Diagram

