



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using System.Threading;
7 using ClockTask;
8
9
10 namespace CounterTask
11 {
12     class Program
13     {
14         static void Main()
15         {
16             Clock clock = new Clock();
17
18
19             for (int i = 0; i < 3853; i++)
20             {
21                 clock.Tick();
22
23             }
24
25             Console.WriteLine(clock.Time());
26         }
27     }
28 }
```

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

```
50         _minute.Reset();
51         _hour.Reset();
52     }
53
54     public string Time()
55     {
56         return $"{_hour.Ticks:D2}:{_minute.Ticks:D2}:
57             {_second.Ticks:D2}";
58     }
59 }
60
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Diagnostics.Contracts;
4 using System.Linq;
5 using System.Security.Cryptography;
6 using System.Text;
7 using System.Threading.Tasks;
8
9 namespace CounterTask
10 {
11     public class Counter
12     {
13         private int _count;
14         private string _name;
15
16         public Counter(string name)
17         {
18             _name = name;
19             _count = 0;
20         }
21         public void Increment()
22         {
23             _count++;
24         }
25         public void Reset()
26         {
27             _count = 0;
28         }
29         public string Name
30         {
31             get
32             {
33                 return _name;
34             }
35             set
36             {
37                 _name = value;
38             }
39         }
40         public int Ticks
41         {
42             get { return _count; }
43         }
44     }
45 }
46
```

```
1 using ClockTask;
2 using CounterTask;
3 using System.Diagnostics.Metrics;
4
5 namespace TestClock
6 {
7     public class Tests
8     {
9         public Clock clock;
10
11         [SetUp]
12         public void Setup()
13         {
14             clock = new Clock();
15         }
16
17         [Test]
18         public void TestTimeReal()
19         {
20             Assert.That(clock.Time(), Is.EqualTo("00:00:00"));
21         }
22
23         [Test]
24         public void TestTickSecond()
25         {
26             clock.Tick();
27             Assert.That(clock.Time(), Is.EqualTo("00:00:01"));
28         }
29
30         [Test]
31         public void TestTickMinute()
32         {
33             for (int i = 0; i < 60; i++)
34             {
35                 clock.Tick();
36             }
37             Assert.That(clock.Time(), Is.EqualTo("00:01:00"));
38         }
39
40         [Test]
41         public void TestTickHour()
42         {
43             for (int i = 0; i < 3600; i++)
44             {
45                 clock.Tick();
46             }
47             Assert.That(clock.Time(), Is.EqualTo("01:00:00"));
48         }
49     }
```

50 }

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


Test Explorer



Test run finished: 8 Tests (8 Passed, 0 Failed, 0 Skipped) run in 112 ms

| Test | Duration | Traits | Error Message |
|-----------------------|----------|--------|---------------|
| ▲ ✓ TestClock (4) | 6 ms | | |
| ▲ ✓ TestClock (4) | 6 ms | | |
| ▲ ✓ Tests (4) | 6 ms | | |
| ✓ TestTickHour | 6 ms | | |
| ✓ TestTickMinute | < 1 ms | | |
| ✓ TestTickSecond | < 1 ms | | |
| ✓ TestTimeReal | < 1 ms | | |
| ▲ ✓ TestCounter (4) | 6 ms | | |
| ▲ ✓ TestCounter (4) | 6 ms | | |
| ▲ ✓ Tests (4) | 6 ms | | |
| ✓ TestIncrement | 6 ms | | |
| ✓ TestIncrementAddOne | < 1 ms | | |
| ✓ TestReset | < 1 ms | | |
| ✓ TestTickInitial | < 1 ms | | |

Microsoft Visual Studio Debug Console

01:04:13

C:\assignments\OOP\Tasks\Tasks\Pass\3.1\ClockTask\ClockTask\bin\Debug\net8.0\ClockTask.exe (process 31528) exited with code 0.

To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.

Press any key to close this window . . .|