

LinksPlatform's Platform.Setters Class Library

1.1 ./csharp/Platform.Setters/SetterBase.cs

```
1 using System.Runtime.CompilerServices;
2 using Platform.Interfaces;
3
4 #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
5
6 namespace Platform.Setters
7 {
8     /// <summary>
9     /// <para>Represents a range between mininum and maximum values.</para>
10    /// <para>Представляет диапазон между минимальным и максимальным значениями.</para>
11    /// </summary>
12    /// <remarks>
13    /// Must be class, not struct (in order to persist access to Result property value).
14    /// </remarks>
15    public abstract class SetterBase<TResult> : ISetter<TResult>
16    {
17        /// <summary>
18        /// <para>Represents a range between mininum and maximum values.</para>
19        /// <para>Представляет диапазон между минимальным и максимальным значениями.</para>
20        /// </summary>
21        protected TResult _result;
22
23        /// <summary>
24        /// <para>Represents a range between mininum and maximum values.</para>
25        /// <para>Представляет диапазон между минимальным и максимальным значениями.</para>
26        /// </summary>
27        public TResult Result => _result;
28
29        /// <summary>
30        /// <para>Represents a range between mininum and maximum values.</para>
31        /// <para>Представляет диапазон между минимальным и максимальным значениями.</para>
32        /// </summary>
33        [MethodImpl(MethodImplOptions.AggressiveInlining)]
34        protected SetterBase() { }
35
36        /// <summary>
37        /// <para>Represents a range between mininum and maximum values.</para>
38        /// <para>Представляет диапазон между минимальным и максимальным значениями.</para>
39        /// </summary>
40        /// <param name="minimum"><para>The minimum value of the range.</para><para>Минимальное
41        ↪ значение диапазона.</para></param>
42        [MethodImpl(MethodImplOptions.AggressiveInlining)]
43        protected SetterBase(TResult defaultValue) => _result = defaultValue;
44
45        /// <summary>
46        /// <para>Represents a range between mininum and maximum values.</para>
47        /// <para>Представляет диапазон между минимальным и максимальным значениями.</para>
48        /// </summary>
49        /// <param name="minimum"><para>The minimum value of the range.</para><para>Минимальное
50        ↪ значение диапазона.</para></param>
51        [MethodImpl(MethodImplOptions.AggressiveInlining)]
52        public void Set(TResult value) => _result = value;
53    }
54 }
```

1.2 ./csharp/Platform.Setters/Setter[TResult, TDecision].cs

```
1 using System.Collections.Generic;
2 using System.Runtime.CompilerServices;
3
4 #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
5
6 namespace Platform.Setters
7 {
8     public class Setter<TResult, TDecision> : SetterBase<TResult>
9     {
10         private readonly TDecision _trueValue;
11         private readonly TDecision _falseValue;
12
13         [MethodImpl(MethodImplOptions.AggressiveInlining)]
14         public Setter(TDecision trueValue, TDecision falseValue, TResult defaultValue)
15             : base(defaultValue)
16         {
17             _trueValue = trueValue;
18             _falseValue = falseValue;
19         }
20
21         [MethodImpl(MethodImplOptions.AggressiveInlining)]
```

```

22     public Setter(TDecision trueValue, TDecision falseValue) : this(trueValue, falseValue,
    ↪     default) { }
23
24     [MethodImpl(MethodImplOptions.AggressiveInlining)]
25     public Setter(TResult defaultValue) : base(defaultValue) { }
26
27     [MethodImpl(MethodImplOptions.AggressiveInlining)]
28     public Setter() { }
29
30     [MethodImpl(MethodImplOptions.AggressiveInlining)]
31     public TDecision SetAndReturnTrue(TResult value)
32     {
33         _result = value;
34         return _trueValue;
35     }
36
37     [MethodImpl(MethodImplOptions.AggressiveInlining)]
38     public TDecision SetAndReturnFalse(TResult value)
39     {
40         _result = value;
41         return _falseValue;
42     }
43
44     [MethodImpl(MethodImplOptions.AggressiveInlining)]
45     public TDecision SetFirstAndReturnTrue(ICollection<TResult> list)
46     {
47         _result = list[0];
48         return _trueValue;
49     }
50
51     [MethodImpl(MethodImplOptions.AggressiveInlining)]
52     public TDecision SetFirstAndReturnFalse(ICollection<TResult> list)
53     {
54         _result = list[0];
55         return _falseValue;
56     }
57 }
58 }

```

1.3 ./csharp/Platform.Setters/Setter[TResult].cs

```

1 using System.Runtime.CompilerServices;
2
3 #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
4
5 namespace Platform.Setters
6 {
7     public class Setter<TResult> : Setter<TResult, bool>
8     {
9         [MethodImpl(MethodImplOptions.AggressiveInlining)]
10        public Setter(TResult defaultValue) : base(true, false, defaultValue) { }
11
12        [MethodImpl(MethodImplOptions.AggressiveInlining)]
13        public Setter() : base(true, false) { }
14    }
15 }

```

1.4 ./csharp/Platform.Setters.Tests/SetterTests.cs

```

1 using Xunit;
2
3 namespace Platform.Setters.Tests
4 {
5     public class SetterTests
6     {
7         [Fact]
8         public void ParameterlessConstructedSetterTest()
9         {
10             Setter<int> setter = new Setter<int>();
11             Assert.Equal(default, setter.Result);
12         }
13
14         [Fact]
15         public void ConstructedWithDefaultValueSetterTest()
16         {
17             Setter<int> setter = new Setter<int>(9);
18             Assert.Equal(9, setter.Result);
19         }
20
21         [Fact]
22         public void MethodsWithBooleanReturnTypeTest()

```

```

23     {
24         Setter<int> setter = new Setter<int>();
25         Assert.True(setter.SetAndReturnTrue(1));
26         Assert.Equal(1, setter.Result);
27         Assert.False(setter.SetAndReturnFalse(2));
28         Assert.Equal(2, setter.Result);
29         Assert.True(setter.SetFirstAndReturnTrue(new int[] { 3 }));
30         Assert.Equal(3, setter.Result);
31         Assert.False(setter.SetFirstAndReturnFalse(new int[] { 4 }));
32         Assert.Equal(4, setter.Result);
33     }
34
35     [Fact]
36     public void MethodsWithIntegerReturnTypeTest()
37     {
38         Setter<int, int> setter = new Setter<int, int>(1, 0);
39         Assert.Equal(1, setter.SetAndReturnTrue(1));
40         Assert.Equal(1, setter.Result);
41         Assert.Equal(0, setter.SetAndReturnFalse(2));
42         Assert.Equal(2, setter.Result);
43         Assert.Equal(1, setter.SetFirstAndReturnTrue(new int[] { 3 }));
44         Assert.Equal(3, setter.Result);
45         Assert.Equal(0, setter.SetFirstAndReturnFalse(new int[] { 4 }));
46         Assert.Equal(4, setter.Result);
47     }
48 }
49

```

Index

`./csharp/Platform.Setters.Tests/SetterTests.cs`, 2
`./csharp/Platform.Setters/SetterBase.cs`, 1
`./csharp/Platform.Setters/Setter[TResult, TDecision].cs`, 1
`./csharp/Platform.Setters/Setter[TResult].cs`, 2