```
LinksPlatform's Platform.Interfaces Class Library
    ./csharp/Platform.Interfaces/ICounter[TResult, TArgument].cs
   namespace Platform. Interfaces
        /// <summary>
3
       /// <para>Defines a counter that requires an argument to perform a count.</para>
4
       /// <para>Определяет счётчик, которому требуется аргумент для выполнения подсчёта.</para>
       /// </summary>
       /// <typeparam name="TArgument">
       /// <para>The argument type.</para>
       /// <para>Тип аргумента.</para>
       /// </typeparam>
10
       /// <typeparam name="TResult">
11
       /// <para>The count result type.</para>
       /// <para>Тип результата подсчёта.</para>
       /// </typeparam>
14
       public interface ICounter<out TResult, in TArgument>
16
            /// <summary>
17
            /// <para>Performs a count.</para>
            /// <para>Выполняет подсчёт.</para>
            /// </summary>
20
            /// <param name="argument">
21
            /// <para>The argument.</para>
           /// <para>Аргумент.</para>
23
           /// </param>
24
            /// <returns>
            /// <para>The count result.</para>
            /// <para>Результат подсчёта.</para>
27
            /// </returns>
28
           TResult Count(TArgument argument);
       }
30
31
    ./csharp/Platform.Interfaces/ICounter[TResult].cs
   namespace Platform.Interfaces
       /// <summary>
       /// <para>Defines a counter.</para>
       /// <para>Определяет счётчик.</para>
       /// </summary>
       /// <typeparam name="TResult">
       /// <para>The count result type.</para>
       /// <para>Тип результата подсчёта.</para>
       /// </typeparam>
       public interface ICounter<out TResult>
11
12
            /// <summary>
13
            /// <para>Performs a count.</para>
14
            /// <para>Выполняет подсчёт.</para>
15
            /// </summary>
            /// <returns>
17
            /// <para>The count result.</para>
18
            /// <para>Результат подсчёта.</para>
            /// </returns>
20
           TResult Count();
21
       }
22
     ./csharp/Platform.Interfaces/ICriterionMatcher.cs
1.3
   namespace Platform. Interfaces
2
        /// <summary>
3
       /// <para>Defines a criterion matcher, that contains a specific method for determining
4
           whether the argument matches the criterion or not.</para>
       /// <para>Определяет объект который проверяет соответствие критерию и содержит конкретный
           метод для определения, соответствует ли аргумент критерию или нет.</para>
       /// </summary>
       /// <typeparam name="TArgument">
       /// <para>Argument type.</para>
       /// <para>Тип аргумента.</para>
       /// </typeparam>
       public interface ICriterionMatcher<in TArgument>
11
            /// <summary>
13
            /// <para>Determines whether the argument matches the criterion.</para>
14
            /// <para>Определяет, соответствует ли аргумент критерию.</para>
```

```
/// </summary>
16
            /// <param name="argument">
17
            /// <para>The argument.</para>
18
            /// <para>Аргумент.</para>
19
            /// </param>
            /// <returns>
21
            /// <para>A value that determines whether the argument matches the criterion.</para>
22
            /// <para>Значение, определяющие соответствует ли аргумент критерию.</para>
23
            /// </returns>
           bool IsMatched(TArgument argument);
25
26
   }
    ./csharp/Platform.Interfaces/IFactory.cs
   namespace Platform.Interfaces
        /// <summary>
3
       /// <para>Defines a factory that produces instances of a specific type.</para>
4
       /// <para>Определяет фабрику, которая производит экземпляры определенного типа.</para>
       /// </summary>
       /// <typeparam name="TProduct">
       /// <para>Type of produced instances.</para>
       /// <para>Tип производимых экземпляров.</para>
       /// </typeparam>
10
       public interface IFactory<out TProduct>
11
12
            /// <summary>
13
            /// <para>Creates an instance of <typeparamref name="TProduct"/> type.</para>
14
            /// <para>Создает экземпляр типа <typeparamref name="TProduct"/>.</para>
            /// <\bar{\summary>
            /// <returns>
17
            /// <para>The instance of <typeparamref name="TProduct"/> type.</para>
18
            /// <para>Экземпляр типа <typeparamref name="TProduct"/>.</para>
19
            /// </returns>
20
            TProduct Create();
       }
   }
    ./csharp/Platform.Interfaces/IProperties.cs
   namespace Platform.Interfaces
2
       /// <summary>
3
       /// <para>Defines a properties operator that is able to get or set values of properties of a
           object of a specific type.</para>
       /// <para>Определяет оператор свойств, который может получать или устанавливать значения
           свойств объекта определенного типа.</para>
       /// </summary>
       /// <typeparam name="TObject">
       /// <para>Object type.</para>
       /// <para>Тип объекта.</para>
9
       /// </typeparam>
10
       /// <typeparam name="TProperty">
11
       /// <para>Property reference type.</para>
12
       /// <para>Тип ссылки на свойство.</para>
13
       /// </typeparam>
14
       /// <typeparam name="TValue">
15
       /// <para>Property value type.</para>
       /// <para>Тип значения свойства.</para>
17
       /// </typeparam>
18
       public interface IProperties<in TObject, in TProperty, TValue>
19
20
            /// <summary>
            /// <para>Gets the value of the property in the specified object.</para>
22
            /// <para>Получает значение свойства в указанном объекте.</para>
23
            /// </summary>
24
            /// <param name="object">
            /// <para>The object reference.</para>
26
            /// <para>Ссылка на объект.</para>
27
            /// </param>
            /// <param name="property">
            /// <para>The property reference.</para>
30
31
            /// <para>Ссылка на свойство.</para>
            /// </param>
32
           /// <returns>
33
           /// <para>The value of the property.</para>
34
            /// <para>Значение свойства.</para>
            /// </returns>
```

```
TValue GetValue(TObject @object, TProperty property);
38
            /// <summary>
39
            /// <para>Sets the value of a property in the specified object.</para>
            /// <para>Устанавливает значение свойства в указанном объекте.</para>
41
            /// </summary>
42
            /// <param name="object">
43
            /// <para>The object reference.</para>
44
            /// <para>Ссылка на объект.</para>
45
            /// </param>
46
            /// <param name="property">
47
            /// <para>The property reference.</para>
            /// <para>Ссылка на свойство.</para>
49
            /// </param>
50
            /// <param name="value">
            /// <para>The value.</para>
52
            /// <para>Значение.</para>
53
            /// </param>
            void SetValue(TObject @object, TProperty property, TValue value);
55
       }
56
   }
57
     ./csharp/Platform.Interfaces/IProperty.cs
   namespace Platform.Interfaces
2
        /// <summary>
3
       /// <para>Defines a specific property operator that is able to get or set values of that
4
           property.</para>
       /// <para>Oпределяет оператор определённого свойства, который может получать или
5
           устанавливать его значения.</para>
       /// </summary>
       /// <typeparam name="TObject">
       /// <para>Object type.</para>
       /// <para>Тип объекта.</para>
9
        /// </typeparam>
10
       /// <typeparam name="TValue">
11
       /// <para>Property value type.</para>
12
       /// <para>Тип значения свойства.</para>
13
       /// </typeparam>
       public interface IProperty<in TObject, TValue> : ISetter<TValue, TObject>, IProvider<TValue,
15
           TObject>
16
17
   }
18
     ./csharp/Platform.Interfaces/IProvider[TProvided, TArgument].cs
1.7
   namespace Platform.Interfaces
        /// <summary>
3
       /// <para>Defines the provider of objects/values for which an argument must be
4
           specified.</para>
       /// <para>Определяет поставщика объектов/значений, для получения которых необходимо указать
5
           аргумент.</para>
       /// </summary>
       /// <typeparam name="TProvided">
       /// <para>Type of provided objects/values.</para>
       /// <para>Тип предоставляемых объектов/значений.</para>
       /// </typeparam>
10
       /// <typeparam name="TArgument">
11
       /// <para>Argument type.</para>
12
        /// <para>Тип аргумента.</para>
13
       /// </typeparam>
14
       public interface IProvider<out TProvided, in TArgument>
            /// <summary>
17
            /// <para>Provides an object(s)/value(s).</para>
18
            /// <para>Предоставляет объект(ы)/значение(я).</para>
19
            /// </summary>
20
            /// <param name="argument">
21
            /// <para>The argument required to acquire the object(s)/value(s).</para>
            /// <para>Аргумент, необходимый для получения объекта(ов)/значения(ий).</para>
            /// </param>
24
            /// <returns>
25
26
            /// <para>The object(s)/value(s).</para>
            /// <para>Объект(ы)/значение(я).</para>
27
            /// </returns>
28
            TProvided Get(TArgument argument);
```

```
30
   }
31
    ./csharp/Platform.Interfaces/IProvider[TProvided].cs
   namespace Platform.Interfaces
1
2
        /// <summary>
3
       /// <para>Defines the provider of objects/values.</para>
4
       /// <para>Определяет поставщика объектов/значений.</para>
       /// </summary>
       /// <typeparam name="TProvided">
       /// <para>Type of provided object/value.</para>
       /// <para>Tun предоставляемого объекта/значения.</para>
       /// </typeparam>
10
       public interface IProvider < out TProvided >
11
12
            /// <summary>
13
            /// <para>Provides an object(s)/value(s).</para>
14
            /// <para>Предоставляет объект(ы)/значение(я).</para>
            /// </summary>
16
            /// <returns>
17
            /// <para>The object(s)/value(s).</para>
18
            /// <para>Объект(ы)/значение(я).</para>
19
            /// </returns>
20
            TProvided Get();
21
       }
22
   }
23
     ./csharp/Platform.Interfaces/ISetter[TValue, TArgument].cs
   namespace Platform.Interfaces
2
       /// <summary>
3
       /// <para>Defines an setter that requires an argument to set the passed value as a new
           state.</para>
       /// <para>Определяет установщик, которому для установки переданного значения в качестве
          нового состояния требуется аргумент.</para>
       /// </summary>
       /// <typeparam name="TValue">
        /// <para>Type of set value.</para>
       /// <para>Тип устанавливаемого значения.</para>
9
       /// </typeparam>
10
       /// <typeparam name="TArgument">
11
        /// <para>The argument type.</para>
       /// <para>Тип аргумента.</para>
13
       /// </typeparam>
14
       public interface ISetter<in TValue, in TArgument>
15
16
            /// <summary>
17
            /// <para>Sets the value of a specific property in the specified object.</para>
            /// <para>Устанавливает значение определённого свойства в указанном объекте.</para>
19
            /// </summary>
20
            /// <param name="argument">
            /// <para>The argument.</para>
22
            /// <para>Аргумент.</para>
23
            /// </param>
            /// <param name="value">
            /// <para>The value.</para>
26
            /// <para>Значение.</para>
27
            /// </param>
            void Set(TArgument argument, TValue value);
29
       }
30
      ./csharp/Platform.Interfaces/ISetter[TValue].cs
1.10
   namespace Platform. Interfaces
       /// <summary>
3
       /// <para>Defines an setter that sets the passed value as a new state.</para>
4
       /// <para>Определяет установщик, который устанавливает переданное значение в качестве нового
           состояния.</para>
       /// </summary>
       /// <typeparam name="TValue">
       /// <para>Type of set value.</para>
       /// <para>Тип устанавливаемого значения.</para>
        /// </typeparam>
       public interface ISetter<in TValue>
11
```

```
/// <summary>
13
               /// <para>Sets the value of a specific property in the specified object.</para>
14
              /// <para>Устанавливает значение определённого свойства в указанном объекте.</para>
15
              /// </summary>
16
              /// <param name="value">
               /// <para>The value.</para>
              /// <para>Значение.</para>
/// </param>
19
20
              void Set(TValue value);
         }
22
23
      ./csharp/Platform.Interfaces.Tests/InterfacesTests.cs
1.11
    using Xunit;
1
    #pragma warning disable CS0168 // Variable is declared but never used
3
    #pragma warning disable CS0219 // Variable is assigned but its value is never used
4
    namespace Platform.Interfaces.Tests
         public static class InterfacesTests
{
8
9
               [Fact]
10
              public static void BuildTest()
11
12
                    ICounter<int, int> c1 = null;
ICounter<int> c2 = null;
13
                    ICriterionMatcher<int> cm1 = null;
15
                    IFactory<int> f1 = null;
                    IPactory int> 11 - null;
IProperties int, int, int> p1 = null;
IProperty int, int> p2 = null;
IProvider int, int> p3 = null;
IProvider int> p4 = null;
ISetter int, int> s1 = null;
ISetter int> s2 = null;
17
18
20
21
              }
         }
24
    }
```

Index

```
./csharp/Platform.Interfaces.Tests/InterfacesTests.cs, 5
./csharp/Platform.Interfaces/ICounter[TResult, TArgument].cs, 1
./csharp/Platform.Interfaces/ICounter[TResult].cs, 1
./csharp/Platform.Interfaces/ICriterionMatcher.cs, 1
./csharp/Platform.Interfaces/IFactory.cs, 2
./csharp/Platform.Interfaces/IProperties.cs, 2
./csharp/Platform.Interfaces/IProperty.cs, 3
./csharp/Platform.Interfaces/IProvider[TProvided, TArgument].cs, 3
./csharp/Platform.Interfaces/IProvider[TProvided].cs, 4
./csharp/Platform.Interfaces/ISetter[TValue, TArgument].cs, 4
./csharp/Platform.Interfaces/ISetter[TValue].cs, 4
```