```
LinksPlatform's Platform.Interfaces Class Library
     ./csharp/Platform.Interfaces/ICounter[TResult, TArgument].cs
   namespace Platform. Interfaces
       /// <summary>
3
       /// <para>Defines a counter that requires passing an argument to perform a count.</para>
4
       /// <para>Определяет счётчик, который требует передачи аргумента для выполнения
          подсчёта.</para>
        /// </summary>
       /// <typeparam name="TArgument"><para>The argument type.</para><para>Тип
           аргумента.</para></typeparam>
       /// <typeparam name="TResult"><para>The count result type.</para><para>Тип результата
           подсчёта.</para></typeparam>
       public interface ICounter<out TResult, in TArgument>
10
            /// <summary>
11
            /// <para>Performs a count that requires passing an argument.</para>
            /// <para>Выполняет подсчёт, для которого требуется передача аргумента.</para>
            /// </summary>
14
            /// <param name="argument"><para>The argument.</para><para>Аргумент.</para></param>
15
            /// <returns><para>The count result.</para><para>Результат подсчёта.</para></returns>
           TResult Count(TArgument argument);
17
18
   }
    ./csharp/Platform.Interfaces/ICounter[TResult].cs
   namespace Platform.Interfaces
       /// <summary>
3
       /// <para>Defines a counter.</para>
4
       /// <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>
            /// <summary>
10
            /// <para>Performs a count.</para>
11
            /// <para>Выполняет подсчёт.</para>
           /// </summary>
13
            /// <returns><para>The count result.</para><para>Peзультат подсчёта.</para></returns>
14
           TResult Count();
       }
16
   }
17
    ./csharp/Platform.Interfaces/ICriterionMatcher.cs
   namespace Platform.Interfaces
2
       /// <summary>
3
       /// <para>Defines a criterion matcher, that contains a specific method for determining
           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>
            /// <summary>
10
           /// <para>Determines whether the argument matches the criterion.</para>
11
           /// <para>Определяет, соответствует ли аргумент критерию.</para>
12
           /// </summary>
            /// <param name="argument"><para>The argument.</para><para>Аргумент.</para></param>
            /// <returns><para>A value that determines whether the argument matches the
15
            criterion.</para><pаra>Значение, определяющие соответствует ли аргумент
               критерию.</para></returns>
           bool IsMatched(TArgument argument);
16
       }
17
   }
18
    /csharp/Platform.Interfaces/IFactory.cs
   namespace Platform.Interfaces
2
       /// <summary>
3
       /// <para>Defines a factory that produces instances of a specific type.</para>
       /// <para>Определяет фабрику, которая производит экземпляры определенного типа.</para>
```

```
/// </summary>
       /// <typeparam name="TProduct"><para>Туре of produced instances.</para><para>Тип
       → производимых экземпляров.</para></typeparam>
public interface IFactory<out TProduct>
            /// <summary>
10
            /// <para>Creates an instance of <typeparamref name="TProduct"/> type.</para>
11
            /// <para>Создает экземпляр типа <typeparamref name="TProduct"/>.</para>
12
            /// </summary>
13
            /// <returns > <para > The instance of <typeparamref name = "TProduct" />
14
            → type.</para><para>Экземпляр типа <typeparamref name="TProduct"/>.</para></returns>
           TProduct Create();
15
       }
16
   }
    ./csharp/Platform.Interfaces/IProperties.cs
   namespace Platform. Interfaces
1
2
        /// <summary>
3
       /// <para>Defines a properties operator that is able to get or set values of properties of a
4
           object of a specific type.</para>
       /// <para>Определяет оператор свойств, который может получать или установливать значения
           свойств объекта определенного типа.</para>
       /// </summary>
       /// <typeparam name="TObject"><para>Object type.</para>Tип объекта.</para></typeparam>
       /// <typeparam name="TProperty"><para>Property reference type.</para><para>Тип ссылки на
           свойство.</para></typeparam>
       /// <typeparam name="TValue"><para>Property value type.</para><para>Тип значения
           свойства.</para></typeparam>
       public interface IProperties<in TObject, in TProperty, TValue>
10
11
12
            /// <para>Gets the value of the property in the specified object.</para>
13
            /// <para>Получает значение свойства в указанном объекте.</para>
14
            /// </summary>
            /// <param name="object"><para>The object reference.</para><para>Ссылка на

→ объект.</para></param>

            /// <param name="property"><para>The property reference.</para><para>Ссылка на
17
               свойство.</para></param>
            /// <returns><para>The value of the property.</para><para>Значение
18
               свойства.</para></returns>
            TValue GetValue(TObject @object, TProperty property);
19
20
            /// <summary>
2.1
            /// <para>Sets the value of a property in the specified object.</para>
            /// <para>Устанавливает значение свойства в указанном объекте.</para>
23
            /// </summary>
24
            /// <param name="object"><para>The object reference.</para><para>Ссылка на
25
               объект.</para></param>
            /// <param name="property"><para>The property reference.</para><para>Ссылка на
               свойство.</para></param>
            /// <param name="value"><para>The value.</para><para>Значение.</para></param>
            void SetValue(TObject @object, TProperty property, TValue value);
28
       }
29
     ./csharp/Platform.Interfaces/IProperty.cs
1.6
   namespace Platform. Interfaces
1
2
       /// <summary>
3
       /// <para>Defines a specific property operator that is able to get or set values of that
4
           property.</para>
       /// <рага>Определяет оператор определённого свойства, который может получать или
       \rightarrow установливать его значения.
       /// <typeparam name="TObject"><para>Object type.</para><para>Тип объекта.</para></typeparam>
       /// <typeparam name="TValue"><para>Property value type.</para><para>Тип значения
           свойства.</para></typeparam>
       public interface IProperty in TObject, TValue : ISetter TValue, TObject : IProvider TValue,
9
           TObject>
        \hookrightarrow
10
11
   }
12
```

```
./csharp/Platform.Interfaces/IProvider[TProvided, TArgument].cs
   namespace Platform. Interfaces
1
2
        /// <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
        → аргумент.</para>
       /// </summary>
       /// <typeparam name="TProvided"><para>Type of provided objects/values.</para>Тип
           предоставляемых объектов/значений.</para></typeparam>
       /// <typeparam name="TArgument"><para>Argument type.</para><para>Тип
           аргумента. </para></typeparam>
       public interface IProvider out TProvided, in TArgument>
10
            /// <summary>
11
            /// <para>Provides an object(s)/value(s).</para>
12
            /// <para>Предоставляет объект(ы)/значение(\bar{x}).</para>
13
            /// </summary>
14
            /// <param name="argument"><para>The argument required to acquire the
            object(s)/value(s).</para><para>Аргумент, необходимый для получения
               объекта(ов)/значения(ий).</para></param>
           /// <returns><para>The
16
               object(s)/value(s).</para><para>Объект(ы)/значение(я).</para></returns>
           TProvided Get(TArgument argument);
       }
18
   }
19
    ./csharp/Platform.Interfaces/IProvider[TProvided].cs
   namespace Platform.Interfaces
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>Tип

¬ предоставляемого объекта/значения.
       public interface IProvider<out TProvided>
            /// <summary>
10
            /// <para>Provides an object(s)/value(s).</para>
11
           /// \sqrt{\text{para}}Предоставляет объект(ы)/значение(я).\sqrt{\text{para}}
12
           /// </summary>
13
           /// <returns><para>The
14
            → object(s)/value(s).</para><para>Объект(ы)/значение(я).</para></returns>
           TProvided Get();
       }
16
   }
17
     ./csharp/Platform.Interfaces/ISetter[TValue, TArgument].cs
   namespace Platform. Interfaces
1
   {
2
       /// <summary>
       /// <para>Defines an setter that requires an argument to set the passed value as a new
4
           state.</para>
       /// <para>Onределяет установщик, которому для установки переданного значения в качестве
        → нового состояния требуется аргумент.</para>
       /// </summary>
       /// <typeparam name="TValue"><para>Type of setted value.</para>Tип устанавливаемого

→ значения.</para></typeparam>

       /// <typeparam name="TArgument"><para>The argument type.</para><para>Тип
           аргумента.</para></typeparam>
       public interface ISetter<in TValue, in TArgument>
10
            /// <summary>
11
           /// <para>Sets the value of a specific property in the specified object.</para>
12
           /// <para>Устанавливает значение определённого свойства в указанном объекте.</para>
13
            /// </summary>
            /// <param name="argument"><para>The argument.</para><para>Аргумент.</para></param>
15
            /// <param name="value"><para>The value.</para><para>Значение.</para></param>
16
            void Set(TArgument argument, TValue value);
       }
18
   }
19
```

```
./csharp/Platform.Interfaces/ISetter[TValue].cs
   namespace Platform. Interfaces
1
2
        /// <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>
6
        /// <typeparam name="TValue"><para>Туре of setted value.</para>Тип устанавливаемого
           значения.</para></typeparam>
        public interface ISetter<in TValue>
9
            /// <summary>
10
            /// <para>Sets the value of a specific property in the specified object.</para>
            /// <para>Устанавливает значение определённого свойства в указанном объекте.</para>
12
            /// <\braces\summary>
13
            /// <param name="value"><para>The value.</para><para>Значение.</para></para>>
14
            void Set(TValue value);
15
        }
16
17
     ./csharp/Platform.Interfaces.Tests/InterfacesTests.cs
1.11
   using Xunit;
   #pragma warning disable CS0168 // Variable is declared but never used
3
   namespace Platform. Interfaces. Tests
5
        public static class InterfacesTests
            [Fact]
9
            public static void BuildTest()
10
11
                ICounter<int, int> c1;
ICounter<int> c2;
12
13
                 ICriterionMatcher<int> cm1;
14
                 IFactory<int> f1;
                 IProperties<int, int, int> p1;
16
                IProperty<int, int> p2;
IProvider<int, int> p3;
17
                 IProvider<int> p4;
19
                ISetter<int, int> s1;
ISetter<int> s2;
20
            }
        }
23
```

24 }

Index

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