

LinksPlatform's Platform.Random Class Library

1.1 ./csharp/Platform.Random/RandomExtensions.cs

```
1 using System.Runtime.CompilerServices;
2 using Platform.Ranges;
3
4 namespace Platform.Random
5 {
6     /// <summary>
7     /// <para>Contains extension methods for <see cref="System.Random"/> class.</para>
8     /// <para>Содержит методы расширения для класса <see cref="System.Random"/>.</para>
9     /// </summary>
10    public static class RandomExtensions
11    {
12        /// <summary>
13        /// <para>Returns a random 64-bit unsigned integer that is greater than or equal to <see
14        → cref="ulong.MinValue"/>, and less than or equal to <see
15        → cref="ulong.MaxValue"/>.</para>
16        /// <para>Возвращает случайное 64-разрядное целое число без знака, которое больше или
17        → равно <see cref="ulong.MinValue"/> и меньше или равно <see
18        → cref="ulong.MaxValue"/>.</para>
19        /// </summary>
20        /// <param name="random"><para>A pseudo-random number generator.</para><para>Генератор
21        → псевдослучайных чисел.</para></param>
22        /// <returns>
23        /// <para>A 64-bit unsigned integer that is greater than or equal to <see
24        → cref="ulong.MinValue"/>, and less than or equal to <see
25        → cref="ulong.MaxValue"/>.</para>
26        /// <para>64-разрядное целое число без знака, которое больше или равно <see
27        → cref="ulong.MinValue"/> и меньше или равно <see cref="ulong.MaxValue"/>.</para>
28        /// </returns>
29        [MethodImpl(MethodImplOptions.AggressiveInlining)]
30        public static ulong NextUInt64(this System.Random random) =>
31        → random.NextUInt64(Range.UInt64);
32
33        /// <summary>
34        /// <para>Returns a random 64-bit unsigned integer that is greater than or equal to
35        → minimum of specified range, and less than or equal to maximum of specified
36        → range.</para>
37        /// <para>Возвращает случайное 64-разрядное целое число без знака, которое больше или
38        → равно минимуму указанного диапазона и меньше или равно максимуму указанного
39        → диапазона.</para>
40        /// </summary>
41        /// <param name="random"><para>A pseudo-random number generator.</para><para>Генератор
42        → псевдослучайных чисел.</para></param>
43        /// <param name="range"><para>The range of possible values.</para><para>Диапазон
44        → возможных значений.</para></param>
45        /// <returns>
46        /// <para>A 64-bit unsigned integer that is greater than or equal to the minimum of
47        → specified range, and less than or equal to the maximum of the specified range.</para>
48        /// <para>64-разрядное целое число без знака, которое больше или равно минимуму
49        → указанного диапазона и меньше или равно максимуму указанного диапазона.</para>
50        /// </returns>
51        [MethodImpl(MethodImplOptions.AggressiveInlining)]
52        public static ulong NextUInt64(this System.Random random, Range<ulong> range) =>
53        → (ulong)(random.NextDouble() * range.Difference()) + range.Minimum;
54
55        /// <summary>
56        /// <para>Return a random <see cref="bool"/> value.</para>
57        /// <para>Возвращает случайное значение <see cref="bool"/>.</para>
58        /// </summary>
59        /// <param name="random"><para>A pseudo-random number generator.</para><para>Генератор
60        → псевдослучайных чисел.</para></param>
61        /// <returns><para>A random <see cref="bool"/> value.</para><para>Случайное значение
62        → <see cref="bool"/>.</para></returns>
63        [MethodImpl(MethodImplOptions.AggressiveInlining)]
64        public static bool NextBoolean(this System.Random random) => random.Next(2) == 1;
65    }
66 }
```

1.2 ./csharp/Platform.Random/RandomHelpers.cs

```
1 namespace Platform.Random
2 {
3     /// <summary>
4     /// <para>Contains field-helper for <see cref="System.Random"/> class.</para>
5     /// <para>Содержит вспомогательное поле для класса <see cref="System.Random"/>.</para>
6     /// </summary>
7     public static class RandomHelpers
```

```

8 {
9     /// <summary>
10    /// <para>Returns the pseudorandom number generator that is using the time of the first
    ↪ access to this field as seed.</para>
11    /// <para>Возвращает генератор псевдослучайных чисел использующий в качестве seed время
    ↪ первого обращения к этому полю.</para>
12    /// </summary>
13    public static readonly System.Random Default = new
    ↪ System.Random(System.DateTime.UtcNow.Ticks.GetHashCode());
14 }
15 }

```

1.3 ./csharp/Platform.Random.Tests/RandomExtensionsTests.cs

```

1 using Xunit;
2
3 namespace Platform.Random.Tests
4 {
5     /// <summary>
6     /// <para>
7     /// Represents the random extensions tests.
8     /// </para>
9     /// <para></para>
10    /// </summary>
11    public class RandomExtensionsTests
12    {
13        /// <summary>
14        /// <para>
15        /// Tests that next u int 64 test.
16        /// </para>
17        /// <para></para>
18        /// </summary>
19        [Fact]
20        public void NextUInt64Test()
21        {
22            var lastValue = 0UL;
23            var theSameCount = 0;
24            for (var i = 0; i < 10; i++)
25            {
26                var newValue = RandomHelpers.Default.NextUInt64();
27                if (newValue == lastValue)
28                {
29                    theSameCount++;
30                }
31                else
32                {
33                    lastValue = newValue;
34                    theSameCount = 0;
35                }
36                Assert.InRange(RandomHelpers.Default.NextUInt64((0UL, 5UL)), 0UL, 5UL);
37            }
38            Assert.True(theSameCount < 8);
39        }
40
41        /// <summary>
42        /// <para>
43        /// Tests that next boolean test.
44        /// </para>
45        /// <para></para>
46        /// </summary>
47        [Fact]
48        public void NextBooleanTest()
49        {
50            var trueCount = 0;
51            var falseCount = 0;
52            for (var i = 0; i < 10; i++)
53            {
54                var newValue = RandomHelpers.Default.NextBoolean();
55                if (newValue)
56                {
57                    trueCount++;
58                }
59                else
60                {
61                    falseCount++;
62                }
63            }
64            Assert.True(trueCount > 0);
65            Assert.True(falseCount > 0);
66        }
67    }
68 }

```

```
67     }
68 }
```

1.4 ./csharp/Platform.Random.Tests/RandomHelpersTests.cs

```
1 using Xunit;
2
3 namespace Platform.Random.Tests
4 {
5     /// <summary>
6     /// <para>
7     /// Represents the random helpers tests.
8     /// </para>
9     /// <para></para>
10    /// </summary>
11    public class RandomHelpersTests
12    {
13        /// <summary>
14        /// <para>
15        /// Tests that default field test.
16        /// </para>
17        /// <para></para>
18        /// </summary>
19        [Fact]
20        public void DefaultFieldTest()
21        {
22            Assert.NotNull(RandomHelpers.Default);
23        }
24    }
25 }
```

Index

- ./csharp/Platform.Random.Tests/RandomExtensionsTests.cs, 2
- ./csharp/Platform.Random.Tests/RandomHelpersTests.cs, 3
- ./csharp/Platform.Random/RandomExtensions.cs, 1
- ./csharp/Platform.Random/RandomHelpers.cs, 1