```
LinksPlatform's Platform.Communication.Protocol.Lino Class Library
     ./csharp/Platform.Communication.Protocol.Lino/ILinksGroupListExtensions.cs
   using System.Collections.Generic;
   using System.Runtime.CompilerServices;
2
   #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
4
   namespace Platform.Communication.Protocol.Lino
6
        public static class ILinksGroupListExtensions
9
            [MethodImpl(MethodImplOptions.AggressiveInlining)]
10
            public static List<Link> ToLinksList(this IList<LinksGroup> groups)
11
12
                var list = new List<Link>();
13
                for (var i = 0; i < groups.Count; i++)</pre>
1.5
                     groups[i].AppendToLinksList(list);
16
17
                return list;
18
            }
19
        }
21
1.2
     ./csharp/Platform.Communication.Protocol.Lino/IListExtensions.cs
   using Platform.Collections;
   using System;
2
   using System.Collections.Generic;
using System.Linq;
4
   using System.Runtime.CompilerServices;
   #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
   namespace Platform.Communication.Protocol.Lino
9
   {
10
        public static class IListExtensions
12
            [MethodImpl(MethodImplOptions.AggressiveInlining)]
13
            public static string Format(this IList<Link> links) => string.Join(Environment.NewLine,
14
               links.Select(l => 1.ToString()));
1.5
            [MethodImpl(MethodImplOptions.AggressiveInlining)]
16
            public static string Format(this IList<Link> links, bool lessParentheses)
17
18
                if (lessParentheses == false)
19
20
                    return links.Format();
21
                }
22
                else
23
24
                     return string.Join(Environment.NewLine, links.Select(1 =>
25
                     → 1.ToString().TrimSingle('(').TrimSingle(')')));
                }
            }
27
        }
28
   }
29
1.3
     ./csharp/Platform.Communication.Protocol.Lino/Link.cs
   using System;
   using System.Collections.Generic;
   using System.Runtime.CompilerServices;
3
   using System. Text;
   using Platform.Collections;
5
   using Platform.Collections.Lists;
   #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
9
   namespace Platform.Communication.Protocol.Lino
10
   {
11
        public struct Link : IEquatable<Link>
12
13
            public readonly string Id;
14
15
            public readonly IList<Link> Values;
16
17
            [MethodImpl(MethodImplOptions.AggressiveInlining)]
18
            public Link(string id, IList<Link> values) => (Id, Values) = (id, values);
19
            [{\tt MethodImpl}({\tt MethodImpl}{\tt Options.AggressiveInlining}) \, \rfloor \,
21
            public Link(IList<Link> values) : this(null, values) { }
```

```
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public Link(params Link[] values) : this(null, values) { }
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public Link(string id) : this(id, null) { }
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public override string ToString() => Values.IsNullOrEmpty() ? $\frac{\$"({Id})"}{!}":

    GetLinkValuesString();
[MethodImpl(MethodImplOptions.AggressiveInlining)]
private string GetLinkValuesString() => Id == null ? $\propto (\{\text{GetValuesString()}\})\right" :
   $|"({Id}: {GetValuesString()})";
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public string GetValuesString()
    var sb = new StringBuilder();
    for (int i = 0; i < Values.Count; i++)</pre>
        if (i > 0)
            sb.Append(' ');
        sb.Append(GetValueString(Values[i]));
    return sb.ToString();
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public Link Simplify()
    if (Values.IsNullOrEmpty())
        return this;
    }
    else if (Values.Count == 1)
    {
        return Values[0];
    }
    else
    {
        var newValues = new Link[Values.Count];
        for (int i = 0; i < Values.Count; i++)</pre>
            newValues[i] = Values[i].Simplify();
        return new Link(Id, newValues);
    }
}
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public Link Combine(Link other) => new Link(this, other);
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public static string GetValueString(Link value) => value.ToLinkOrIdString();
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public string ToLinkOrIdString() => Values.IsNullOrEmpty() ? Id : ToString();
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public static implicit operator Link(string value) => new Link(value);
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public static implicit operator Link((string, IList<Link>) value) => new

→ Link(value.Item1, value.Item2);
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public static implicit operator Link((Link, Link) value) => new Link(value.Item1,
→ value.Item2);
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public static implicit operator Link((Link, Link, Link) value) => new Link(value.Item1,

    value.Item2, value.Item3);
[MethodImpl(MethodImplOptions.AggressiveInlining)]
public override bool Equals(object obj) => obj is Link link ? Equals(link) : false;
```

23

25

27

28 29

30

31

33

34

35

36

38

39

40 41

42

44

46

48 49

51

52

54 55

56

57

58

60

61

62

63

64

66

67 68

69

70

71 72 73

75

77 78

79

80

82

83 84

85

86

87

88

90

91

93

```
96
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
            public override int GetHashCode() => (Id, Values.GenerateHashCode()).GetHashCode();
9.8
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
100
            public bool Equals(Link other) => Id == other.Id && Values.EqualTo(other.Values);
101
102
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
103
            public static bool operator ==(Link left, Link right) => left.Equals(right);
104
105
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
106
            public static bool operator !=(Link left, Link right) => !(left == right);
107
        }
108
109
1.4
     ./csharp/Platform.Communication.Protocol.Lino/LinksGroup.cs
    using System;
    using System.Collections.Generic;
    using System.Runtime.CompilerServices;
using Platform.Collections;
 3
    using Platform.Collections.Lists;
 5
    #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
 7
    namespace Platform.Communication.Protocol.Lino
10
        public struct LinksGroup : IEquatable<LinksGroup>
11
12
            public Link Link
13
14
                 [MethodImpl(MethodImplOptions.AggressiveInlining)]
16
                 [MethodImpl(MethodImplOptions.AggressiveInlining)]
                 set:
18
             }
20
            public IList<LinksGroup> Groups
21
22
                 [MethodImpl(MethodImplOptions.AggressiveInlining)]
23
24
                 get;
                 [MethodImpl(MethodImplOptions.AggressiveInlining)]
25
                 set;
26
             }
27
28
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
29
            public LinksGroup(Link link, IList<LinksGroup> groups)
30
                 Link = link;
32
                 Groups = groups;
33
             }
34
35
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
36
            public LinksGroup(Link link) : this(link, null) { }
37
38
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
39
            public static implicit operator List<Link>(LinksGroup value) => value.ToLinksList();
40
41
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
42
            public List<Link> ToLinksList()
44
                 var list = new List<Link>();
45
                 AppendToLinksList(list);
                 return list;
47
             }
48
49
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
50
             public void AppendToLinksList(List<Link> list) => AppendToLinksList(list, Link, this);
52
             [MethodImpl(MethodImplOptions.AggressiveInlining)]
             public static void AppendToLinksList(List<Link> list, Link dependency, LinksGroup group)
54
55
                 list.Add(dependency);
56
                 var groups = group.Groups;
57
                 if (!groups.IsNullOrEmpty())
58
                     for (int i = 0; i < groups.Count; i++)</pre>
60
61
                          var innerGroup = groups[i];
62
                          AppendToLinksList(list, dependency.Combine(innerGroup.Link), innerGroup);
63
                     }
64
```

```
6.5
67
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
           public override bool Equals(object obj) => obj is LinksGroup linksGroup ?
69
            70
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
           public override int GetHashCode() => (Link, Groups.GenerateHashCode()).GetHashCode();
72
73
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
74
           public bool Equals(LinksGroup other) => Link == other.Link &&
75

    Groups.EqualTo(other.Groups);
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
77
           public static bool operator ==(LinksGroup left, LinksGroup right) => left.Equals(right);
78
79
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
80
           public static bool operator !=(LinksGroup left, LinksGroup right) => !(left == right);
81
       }
82
83
1.5
    ./csharp/Platform.Communication.Protocol.Lino/ .cs
   #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
   using System.Runtime.CompilerServices;
   namespace Platform.Communication.Protocol.Lino
5
   {
       public struct _
7
8
           public readonly Link Link;
9
10
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
11
           public _(Link id) => Link = id;
12
13
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
14
           public static implicit operator _(Link value) => new _(value);
15
16
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
17
           public static implicit operator _(string id) => new _(id);
18
19
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
20
           public static implicit operator _((string, Link) value) => new Link(value.Item1, new
            22
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
           public static implicit operator _((string, Link, Link) value) => new Link(value.Item1,
24
              new Link[] { value.Item2, value.Item3 });
25
           [MethodImpl(MethodImplOptions.AggressiveInlining)]
           public static implicit operator _((string, Link, Link, Link) value) => new
27

→ Link(value.Item1, new Link[] { value.Item2, value.Item3, value.Item4 });

           [MethodImpl(MethodImplOptions.AggressiveInlining)]
29
           public static implicit operator Link(_ value) => value.Link;
30
       }
31
32
    ./csharp/Platform.Communication.Protocol.Lino.Tests/ParserTests.cs
  using Xunit;
1
   namespace Platform.Communication.Protocol.Lino.Tests
3
   {
5
       public static class ParserTests
6
           [Fact]
           public static void SingleLinkTest()
               var source = 0"(address: source target)";
10
               var parser = new Parser();
               var links = parser.Parse(source);
12
               var target = links.Format();
13
               Assert.Equal(source, target);
           }
15
16
           [Fact]
           public static void TwoLinksTest()
18
19
```

```
var source = 0"(first: x y)
20
    (second: a b)";
^{21}
                 var parser = new Parser();
22
                 var links = parser.Parse(source);
23
                 var target = links.Format();
                 Assert. Equal(source, target);
25
26
27
             [Fact]
28
             public static void ParseAndStringifyTest()
29
30
                 var source = 0"(papa (lovesMama: loves mama))
31
    (son lovesMama)
32
33
    (daughter lovesMama)
    (all (love mama))";
                 var parser = new Parser();
35
                 var links = parser.Parse(source);
36
                 var target = links.Format();
                 Assert.Equal(source, target);
38
39
40
             [Fact]
41
             public static void ParseAndStringifyWithLessParenthesesTest()
42
                 var source = @"lovesMama: loves mama
44
    papa lovesMama
45
    son lovesMama
46
    daughter lovesMama
47
    all (love mama)";
48
                 var parser = new Parser();
49
                 var links = parser.Parse(source);
50
                 var target = links.Format(lessParentheses: true);
51
                 Assert.Equal(source, target);
             }
53
54
             [Fact]
             public static void SignificantWhitespaceTest()
56
57
                 var source = @"
58
59
    users
        user1
60
             id
                 43
62
63
             name
                 first
                      John
65
                 last
66
                      Williams
67
             location
68
69
                 New York
70
                 23
7.1
        user2
72
             id
73
                 56
             name
75
76
                 first
                      Igor
77
                 middle
78
                      Petrovich
79
                 last
80
                      Ivanov
81
             location
82
                 Moscow
83
84
                 20";
85
                 var target = @"(users)
86
    (users user1)
87
    ((users user1) id)
88
    (((users user1) id) 43)
    ((users user1) name)
90
    (((users user1) name) first)
91
    ((((users user1) name) first) John)
92
    (((users user1) name) last)
    ((((users user1) name) last) Williams)
94
    ((users user1) location)
    (((users user1) location) (New York))
    ((users user1) age)
97
    (((users user1) age) 23)
98
    (users user2)
99
100
    ((users user2) id)
```

```
((users user2) name)
    (((users user2) name) first)
103
    ((((users user2) name) first) Igor)
104
    (((users user2) name) middle)
    ((((users user2) name) middle) Petrovich)
106
    (((users user2) name) last)
107
    ((((users user2) name) last) Ivanov)
108
    ((users user2) location)
    (((users user2) location) Moscow)
110
    ((users user2) age)
111
    (((users user2) age) 20)";
112
                 var parser = new Parser();
113
                  var links = parser.Parse(source);
114
                 var formattedLinks = links.Format();
Assert.Equal(target, formattedLinks);
115
116
             }
117
        }
118
    }
119
1.7
      ./csharp/Platform.Communication.Protocol.Lino.Tests/TupleTests.cs
    using System.Collections.Generic;
    using Xunit;
    namespace Platform.Communication.Protocol.Lino.Tests
 4
         public class TupleTests
 6
             [Fact]
             public void TupleToLinkTest()
 9
10
                  var source = 0"(papa (lovesMama: loves mama))
11
    (son lovesMama)
    (daughter lovesMama)
13
    (all (love mama))";
14
                  var parser = new Parser();
15
                  var links = parser.Parse(source);
16
                  var targetFromString = links.Format();
17
                  IList<Link> constructedLinks = new List<Link>()
19
20
                      ("papa", (_)("lovesMama", "loves", "mama")), ("son", "lovesMama"),
22
                      ("daughter", "lovesMama"),
23
                      ("all", ("love", "mama")),
25
                  var targetFromTuples = constructedLinks.Format();
26
                  Assert.Equal(targetFromString, targetFromTuples);
27
             }
        }
29
    }
30
```

(((users user2) id) 56)

101

Index

- ./csharp/Platform.Communication.Protocol.Lino.Tests/ParserTests.cs, 4
- ./csharp/Platform.Communication.Protocol.Lino.Tests/ParserTests.cs, 4
 ./csharp/Platform.Communication.Protocol.Lino.Tests/TupleTests.cs, 6
 ./csharp/Platform.Communication.Protocol.Lino/ILinksGroupListExtensions.cs, 1
 ./csharp/Platform.Communication.Protocol.Lino/Link.cs, 1
 ./csharp/Platform.Communication.Protocol.Lino/Link.cs, 1

- ./csharp/Platform.Communication.Protocol.Lino/LinksGroup.cs, 3 ./csharp/Platform.Communication.Protocol.Lino/_.cs, 4