LinksPlatform's Platform.Communication.Protocol.Lino Class Library

1.1 ./Platform.Communication.Protocol.Lino/Link.cs

```csharp
using System;
using System.Collections.Generic;
using System.Runtime.CompilerServices;
using System.Text;
using Platform.Collections;
using Platform.Collections.Lists;

#pragma warning disable CS1591 // Missing XML comment for publicly visible type or member

namespace Platform.Communication.Protocol.Lino
{
    public struct Link : IEquatable<Link>
    {
        public string Id
        {
            [MethodImpl(MethodImplOptions.AggressiveInlining)]
            get;
            [MethodImpl(MethodImplOptions.AggressiveInlining)]
            set;
        }

        public IList<Link> Values
        {
            [MethodImpl(MethodImplOptions.AggressiveInlining)]
            get;
            [MethodImpl(MethodImplOptions.AggressiveInlining)]
            set;
        }

        [MethodImpl(MethodImplOptions.AggressiveInlining)]
        public Link(string id, IList<Link> values) => (Id, Values) = (id, values);

        [MethodImpl(MethodImplOptions.AggressiveInlining)]
        public Link(IList<Link> values) : this(null, values) { }

        [MethodImpl(MethodImplOptions.AggressiveInlining)]
        public Link(string id) : this(id, null) { }

        [MethodImpl(MethodImplOptions.AggressiveInlining)]
        public override string ToString() => Values.IsNullOrEmpty() ? $"({Id})" :
            GetLinkValuesString();

        [MethodImpl(MethodImplOptions.AggressiveInlining)]
        private string GetLinkValuesString() => Id == null ? $"({GetValuesString()})" :
            $"({Id}: {GetValuesString()})";

        [MethodImpl(MethodImplOptions.AggressiveInlining)]
        public string GetValuesString()
        {
            var sb = new StringBuilder();
            for (int i = 0; i < Values.Count; i++)
            {
                if (i > 0)
                {
                    sb.Append(' ');
                }
                sb.Append(GetValueString(Values[i]));
            }
            return sb.ToString();
        }

        [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(ValueTuple<string, IList<Link>> value) => new
            Link(value.Item1, value.Item2);

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

```
            [MethodImpl(MethodImplOptions.AggressiveInlining)]
            public override int GetHashCode() => (Id, Values.GenerateHashCode()).GetHashCode();

            [MethodImpl(MethodImplOptions.AggressiveInlining)]
            public bool Equals(Link other) => Id == other.Id && Values.EqualTo(other.Values);

            [MethodImpl(MethodImplOptions.AggressiveInlining)]
            public static bool operator ==(Link left, Link right) => left.Equals(right);

            [MethodImpl(MethodImplOptions.AggressiveInlining)]
            public static bool operator !=(Link left, Link right) => !(left == right);
        }
    }
```

1.2 ./Platform.Communication.Protocol.Lino.Tests/ParserTests.cs

```
using Xunit;

namespace Platform.Communication.Protocol.Lino.Tests
{
    public static class ParserTests
    {
        [Fact]
        public static void BasicTest()
        {
            var parser = new Parser();
            var links = parser.Parse(@"(papa (3: loves mama))
(son 3)
(everyone 3)
(3 (is (loves mama)))
(point)
(point: point point)");
            Assert.Equal("papa", links[0].Values[0].Id);
            Assert.Equal("point", links[4].Id);
        }
    }
}
```

Index