```
LinksPlatform's Platform RegularExpressions Transformer CSharpToCpp Class Library
./Platform.RegularExpressions.Transformer.CSharpToCpp/CSharpToCppTransformer.cs
   using System;
   using System.Collections.Generic;
2
   using System.Linq;
   using System. Text. Regular Expressions;
4
   #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
   namespace Platform.RegularExpressions.Transformer.CSharpToCpp
8
        public class CSharpToCppTransformer : Transformer
10
11
            public static readonly IList<ISubstitutionRule> FirstStage = new List<SubstitutionRule>
12
13
14
                //
15
                (new Regex(0"(\r?\n)?[\t]+//+.+"), "", null, 0),
16
                // #pragma warning disable CS1591 // Missing XML comment for publicly visible type
                    or member
18
                (new Regex(@"^\s*?\#pragma[\sa-zA-Z0-9]+$"), "", null, 0),
19
                // [MethodImpl(MethodImplOptions.AggressiveInlining)]
                (new Regex(@"$\s+\[MethodImpl\(MethodImplOptions\.AggressiveInlining\)\]"), "",
22
                   null, 0),
                // [Fact]
                //
                (new Regex(@"$\s+\[Fact\]"), "", null, 0),
25
                // \{ n n 
26
                // {
27
                (new Regex(0"{\s+[\r\n]+"), "{" + Environment.NewLine, null, 0),
28
                // Platform.Collections.Methods.Lists
29
                // Platform::Collections::Methods::Lists
                (new Regex(@"(namespace[^\r\n]+?)\.([^\r\n]+?)"), "$1::$2", null, 20),
                // public abstract class
32
                // class
33
                (new Regex(@"(public abstract|static) class"), "class", null, 0),
                // class GenericCollectionMethodsBase {
3.5
                // class GenericCollectionMethodsBase { public:
36
                (new Regex(0"class ([a-zA-ZO-9]+)(\s+){"), "class $1$2{" + Environment.NewLine + "
                     public:", null, 0),
                // class GenericCollectionMethodsBase<TElement> {
                // template <typename TElement> class GenericCollectionMethodsBase { public:
39
                (\text{new Regex}(@"class}([a-zA-Z0-9]+)<([a-zA-Z0-9]+)>([^{{]}+){"}}, "template < typename $2>)
40
                   class $1$3{" + Environment.NewLine + "
                                                                public:", null, 0),
                // static void
41
                    TestMultipleCreationsAndDeletions<TElement>(SizedBinaryTreeMethodsBase<TElement>
                    tree, TElement* root)
                // template<typename T> static void
                   TestMultipleCreationsAndDeletions<TElement>(SizedBinaryTreeMethodsBase<TElement>
                 \hookrightarrow tree, TElement* root)
                (\text{new Regex}(@"\text{static}([a-zA-Z0-9]+)([a-zA-Z0-9]+)<([a-zA-Z0-9]+)>\(([^\)]+)\)"),
43
                    "template <typename $3> static $1 $2($4)", null, 0),
                // (this
                // (
                (new Regex(0"\(this "), "(", null, 0),
                // Func<TElement> treeCount
47
                // TElement(*treeCount)()
48
                (\text{new Regex}(@"Func<([a-zA-Z0-9]+)> ([a-zA-Z0-9]+)"), "$1(*$2)()", null, 0),
49
                // Action<TElement> free
50
                // void (*free)(TElement)
51
                (\text{new Regex}(@^*Action}<([a-zA-Z0-9]+)>([a-zA-Z0-9]+)"), "void (*$2)($1)", null, 0),
53
                // private const int MaxPath = 92;
                // static const int MaxPath = 92;
54
                (\text{new Regex}(@"\text{private const}([a-zA-Z0-9]+)([_a-zA-Z0-9]+) = ([a-zA-Z0-9]+);"),
55
                 \rightarrow "static const $1 $2 = $3;", null, 0),
                // protected virtual
                // virtual
57
                (new Regex(0"protected virtual"), "virtual", null, 0),
5.8
                // protected abstract TElement GetFirst();
                // virtual TElement GetFirst() = 0;
60
                (new Regex(@"protected abstract ([^;]+);"), "virtual $1 = 0;", null, 0),
61
                // public virtual
62
                // virtual
                (new Regex(@"public virtual"), "virtual", null, 0),
64
                // protected readonly
65
```

```
66
                  (new Regex(@"protected readonly "), "", null, 0),
                 // protected readonly TreeElement[] _elements;
68
                 // TreeElement _elements[N];
69
                 (new Regex(@"(protected|private) readonly ([a-zA-Z<>0-9]+)([\[\]]+)
                     ([_a-zA-Z0-9]+);"), "$2 $4[N];", null, 0),
                 // protected readonly TElement Zero;
                 // TElement Zero;
72
                 (new Regex(0"(protected|private) readonly ([a-zA-Z<>0-9]+) ([_a-zA-Z0-9]+);"), "$2
7.3
                  \rightarrow $3;", null, 0),
                 // private
                 //
75
                 (new Regex(@"(\W)(private|protected|public|internal) "), "$1", null, 0),
76
                 // SizeBalancedTree(int capacity) => a = b;
77
                 // SizeBalancedTree(int capacity) { a = b; }
                 (new Regex(0"(^s+)(override)?(void)?([a-zA-Z0-9]+)\(([^s(]+)\)\s+=>\s+([^s]+);"),
79
                      "$1$2$3$4($5) { $6; }", null, 0),
                 // () => Integer<TElement>.Zero,
80
                 // () { return Integer<TElement>.Zero;
81
                 (new Regex(@"\(\)\s+=>\s+([^\r\n,;]+?),"), "() { return $1; },", null, 0),
82
                 // => Integer<TElement>.Zero;
83
                 // { return Integer<TElement>.Zero; }
84
                 (new Regex(0"\)\\bar{s}+=>\s+([^\r\n;]+?);"), ") { return $1; }", null, 0),
                 // () { return avlTree.Count; }
                 // []()-> auto { return avlTree.Count; }
(new Regex(0", \(\)) { return ([^;]+); }"), ", []()-> auto { return $1; }", null, 0),
87
88
                 // Count => GetSizeOrZero(Root)
89
                 // GetCount() { return GetSizeOrZero(Root); }
90
                 (new Regex(@"([A-Z][a-z]+)\s+=>\s+([^;]+);"), "Get$1() { return $2; }", null, 0),
91
                 // var
                 // auto
93
                 (new Regex(@"(\W)var(\W)"), "$1auto$2", null, 0),
94
                 // unchecked
95
                 //
96
                 (new Regex(@"[\r\n]{2}\s*?unchecked\s*?$"), "", null, 0),
97
98
                 (new Regex(@"\$"""), "\"",
                                              null, 0),
100
                 // Console.WriteLine("...")
// printf("...\n")
101
102
                 (new Regex(@"Console\.WriteLine\(""([^""]+)""\)"), "printf(\"$1\\n\")", null, 0),
                 // throw new InvalidOperationException
104
                 // throw std::exception
105
                 (new Regex(@"throw new (InvalidOperationException|Exception)"), "throw

    std::exception", null, 0),
                 // override void PrintNode(TElement node, StringBuilder sb, int level)
107
                 // void PrintNode(TElement node, StringBuilder sb, int level) override
108
                  (new Regex(@"override ([a-zA-Z0-9 \*\+]+)(\([^\)]+?\))"), "$1$2 override", null, 0),
109
                 // string
                 // char*
111
                 (new Regex(0"(\W)string(\W)"), "$1char*$2", null, 0),
112
                 // sbyte
                 // std::int8_t
114
                 (new Regex(0"(\W)sbyte(\W)"), "$1std::int8_t$2", null, 0),
115
                 // uint
116
                  // std::uint32_t
117
                 (new Regex(@"(\W)uint(\W)"), "$1std::uint32_t$2", null, 0),
118
                 // char*[] args
119
                 // char* args[]
                 (\text{new Regex}(\tilde{Q}"([_a-zA-ZO-9:\*]?)\[\] ([_a-zA-ZO-9]+)"), "$1 $2[]", null, 0),
121
                 // using Platform.Numbers;
122
123
                 (new Regex(0"([\r\n]{2}|^)\s*?using [\.a-zA-Z0-9]+;\s*?$"), "", null, 0),
                 // struct TreeElement { }
125
                 // struct TreeElement { };
126
                 (new Regex(@"(struct|class) ([a-zA-Z0-9]+)(\s+){([\sa-zA-Z0-9;:_]+?)}([^;])"), "$1
                     $2$3{$4};$5", null, 0),
                 // class Program { }
128
                 // class Program { };  
   (new Regex(@"(struct|class) ([a-zA-Z0-9]+[^\r\n]*)([\r\n]+(?<indentLevel>[\t
129
130
                     ]*)?)\{([\S\s]+?[\r\n]+\k<indentLevel>)\}([^;]|$)"), "$1 $2$3{$4};$5", null, 0),
                 // class SizedBinaryTreeMethodsBase : GenericCollectionMethodsBase
                 // class SizedBinaryTreeMethodsBase : public GenericCollectionMethodsBase
132
                 (\text{new Regex}(0"\text{class ([a-zA-Z0-9]+)} : ([\bar{a}-zA-Z0-9]+)"), "class $1 : public $2", null, [a-zA-Z0-9]+)")
133
                  \rightarrow 0).
             }.Cast<ISubstitutionRule>().ToList();
```

135

```
public static readonly IList<ISubstitutionRule> LastStage = new List<SubstitutionRule>
136
                  // (expression)
138
                 // expression
139
                  (\text{new Regex}(@"(\(| )(([a-zA-Z0-9_{*:}]+))(, | |;|))"), "$1$2$3", null, 0),
                 // (method(expression))
141
                 // method(expression)
142
                  (new Regex(@"(?<firstSeparator>(\()
143
                      ))\((?<method>[a-zA-Z0-9_\->\*:]+)\((?<expression>((?<parenthesis>\()|(?<-parent
                      hesis > ) | [a-zA-ZO-9_\-> *:] *) + ) (?(parenthesis)(?!)) ) ) (? < last Separator > (,
                      |;|\)))"), "${firstSeparator}${method}(${expression})${lastSeparator}", null, 0),
                  // ref sizeBalancedTree2.Root
                  // &sizeBalancedTree2.Root
145
                 (\text{new Regex}(@"\text{ref}([a-zA-Z0-9]+)\.([a-zA-Z0-9]*]+)"), "&$1->$2", null, 0),
146
                 // ref GetElement(node).Right
147
                 // &GetElement(node).Right
                 (new Regex(0"ref ([a-zA-Z0-9]+)\(([a-zA-Z0-9\*]+)\)\.([a-zA-Z0-9]+)"),
149
                      "&$1($2)->$3", null, 0),
                 // GetElement(node).Right
150
                  // GetElement(node)->Right
151
                  (\text{new Regex}(@"([a-zA-Z0-9]+))(([a-zA-Z0-9]*)+))).([a-zA-Z0-9]+)"), "$1($2)->$3",
152
                     null, 0),
                 // = ref GetLeftReference(root);
153
                 // = GetLeftReference(root);
154
                  (\text{new Regex}(@" = \text{ref }([a-zA-Z0-9]+)\setminus(([a-zA-Z0-9]*]+)\setminus);"), " = $1($2);", null, 0),
                 // ref this->GetElement(node)
156
                 // this->GetElement(node)
157
                  (\text{new Regex}(@\text{"ref this} - ([a-zA-Z0-9]+) \setminus (([a-zA-Z0-9]*]+) \setminus))"), "this->$1($2)", null,
158

→ 0),
// ref GetElement(node)
                 // GetElement(node)
160
                  (\text{new Regex}(@"\text{ref}([a-zA-Z0-9]+))(([a-zA-Z0-9]*]+))"), "$1($2)", null, 0),
161
                 // = ref left:
                 // = left;
163
                 (new Regex(0" = ref ([a-zA-Z0-9]+);"), " = $1;", null, 0),
164
                 // (ref
                          left)
165
                  // (left)
                  (new Regex(0"\(ref ([a-zA-Z0-9]+)(\)|\(|,)"), "($1$2", null, 0),
167
                     ref TElement
168
                     {\tt TElement*}
                 (new Regex(0"(|\()ref ([a-zA-Z0-9]+)"), "$1$2* ", null, 0),
170
                 // return ref _elements[node];
171
                  // return &_elements[node];
172
                  (new Regex(0"return ref ([_a-zA-Z0-9]+)\[([_a-zA-Z0-9\*]+)\];"), "return &$1[$2];",
                   \rightarrow null, 0),
                 // default
174
                 // 0
175
                  (new Regex(0"(\W)default(\W)"), "${1}0$2", null, 0),
                 // //#define ENABLE_TREE_AUTO_DEBUG_AND_VALIDATION
177
178
                  (\text{new Regex}(@")//[ t]*\#\text{define}[ t]+[_a-zA-Z0-9]+[ t]*"), "", null, 0),
179
                 // #if USEARRAYPOOL\r\n#endif
180
181
                 (new Regex(0"#if [a-zA-Z0-9]+\s+\#endif"), "", null, 0),
182
                 // \n ... namespace
183
                 // namespace
184
                  (\text{new Regex}(@"(\S[\r\n]{1,2})?[\r\n]+namespace"), "$1namespace", null, 0),
185
                    \n ... class
186
                  // class
                  (new Regex(0"(\S[\r\n]{1,2})?[\r\n]+class"), "$1class", null, 0),
188
             }.Cast<ISubstitutionRule>().ToList();
189
190
             public CSharpToCppTransformer(IList<ISubstitutionRule> extraRules) :
191
                base(FirstStage.Concat(extraRules).Concat(LastStage).ToList()) { }
192
             public CSharpToCppTransformer() : base(FirstStage.Concat(LastStage).ToList()) { }
193
         }
194
    }
./Platform.RegularExpressions.Transformer.CSharpToCpp.Tests/CSharpToCppTransformerTests.cs
    using Xunit;
 2
 3
    namespace Platform.RegularExpressions.Transformer.CSharpToCpp.Tests
 4
         public class CSharpToCppTransformerTests
 5
             [Fact]
```

```
public void HelloWorldTest()
8
                const string helloWorldCode = @"using System;
10
   class Program
11
12
       public static void Main(string[] args)
13
14
            Console.WriteLine(""Hello, world!"");
15
16
   }";
17
                const string expectedResult = @"class Program
18
19
   {
       public:
20
       static void Main(char* args[])
^{21}
            printf(""Hello, world!\n"");
^{23}
^{24}
25
                var transformer = new CSharpToCppTransformer();
^{26}
                var actualResult = transformer.Transform(helloWorldCode, new Context(null));
27
                Assert.Equal(expectedResult, actualResult);
28
            }
        }
30
31
```

Index

 $./Platform. Regular Expressions. Transformer. CSharp ToCpp. Tests/CSharp ToCpp Transformer Tests. cs, \ 3./Platform. Regular Expressions. Transformer. CSharp ToCpp/CSharp ToCpp Transformer. cs, \ 1...$