

LinksPlatform's Platform.RegularExpressions.Transformer.CSharpToCpp Class Library

./Platform.RegularExpressions.Transformer.CSharpToCpp/CSharpToCppTransformer.cs

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text.RegularExpressions;
5
6 #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
7
8 namespace Platform.RegularExpressions.Transformer.CSharpToCpp
9 {
10     public class CSharpToCppTransformer : Transformer
11     {
12         public static readonly IList<ISubstitutionRule> FirstStage = new List<SubstitutionRule>
13         {
14             // // ...
15             //
16             (new Regex(@"(\r?\n)?[ \t]+//+.+"), "", null, 0),
17             // #pragma warning disable CS1591 // Missing XML comment for publicly visible type
18             // or member
19             //
20             (new Regex(@"^-s*?\#pragma[sa-zA-Z0-9]+$"), "", null, 0),
21             // [MethodImpl(MethodImplOptions.AggressiveInlining)]
22             //
23             (new Regex(@"$\s+[MethodImpl\(MethodImplOptions\.AggressiveInlining\)\\]"), "",
24             // null, 0),
25             // [Fact]
26             //
27             (new Regex(@"$\s+[Fact\\]"), "", null, 0),
28             // {
29             // {
30             (new Regex(@"$\s+[\\r\\n]+"), "{" + Environment.NewLine, null, 0),
31             // Platform.Collections.Methods.Lists
32             // Platform::Collections::Methods::Lists
33             (new Regex(@"(namespace[~\\r\\n]+?)\\.([~\\r\\n]+?)"), "$1::$2", null, 20),
34             // public abstract class
35             // class
36             (new Regex(@"(public abstract|static) class"), "class", null, 0),
37             // class GenericCollectionMethodsBase {
38             // class GenericCollectionMethodsBase { public:
39             (new Regex(@"class ([a-zA-Z0-9]+)(\\s+){", "class $1$2{" + Environment.NewLine + "
40             // public:", null, 0),
41             // class GenericCollectionMethodsBase<TElement> {
42             // template <typename TElement> class GenericCollectionMethodsBase { public:
43             (new Regex(@"class ([a-zA-Z0-9]+)<([a-zA-Z0-9]+)>([~\\r\\n]+){", "template <typename $2>
44             // class $1$3{" + Environment.NewLine + " public:", null, 0),
45             // static void
46             // TestMultipleCreationsAndDeletions<TElement>(SizedBinaryTreeMethodsBase<TElement>
47             // tree, TElement* root)
48             // template<typename T> static void
49             // TestMultipleCreationsAndDeletions<TElement>(SizedBinaryTreeMethodsBase<TElement>
50             // tree, TElement* root)
51             (new Regex(@"static ([a-zA-Z0-9]+) ([a-zA-Z0-9]+)<([a-zA-Z0-9]+)>\\((([~\\r\\n]+)\\))"),
52             // "template <typename $3> static $1 $2($4)", null, 0),
53             // (this
54             // (
55             (new Regex(@"\\(this "), "(", null, 0),
56             // Func<TElement> treeCount
57             // TElement(*treeCount)()
58             (new Regex(@"Func<([a-zA-Z0-9]+)> ([a-zA-Z0-9]+)", "$1(*$2)()", null, 0),
59             // Action<TElement> free
60             // void (*free)(TElement)
61             (new Regex(@"Action<([a-zA-Z0-9]+)> ([a-zA-Z0-9]+)", "void (*$2)($1)", null, 0),
62             // private const int MaxPath = 92;
63             // static const int MaxPath = 92;
64             (new Regex(@"private const ([a-zA-Z0-9]+) ([_a-zA-Z0-9]+) = ([a-zA-Z0-9]+);"),
65             // "static const $1 $2 = $3;", null, 0),
66             // protected virtual
67             // virtual
68             (new Regex(@"protected virtual"), "virtual", null, 0),
69             // protected abstract TElement GetFirst();
70             // virtual TElement GetFirst() = 0;
71             (new Regex(@"protected abstract ([~;]+);"), "virtual $1 = 0;", null, 0),
72             // public virtual
73             // virtual
74             (new Regex(@"public virtual"), "virtual", null, 0),
75             // protected readonly
```

```

66 //
67 (new Regex(@"protected readonly "), "", null, 0),
68 // protected readonly TreeElement[] _elements;
69 // TreeElement _elements[N];
70 (new Regex(@"(protected|private) readonly ([a-zA-Z<>0-9]+)([\\[\\]]+)
    ↳ ([_a-zA-Z0-9]+);"), "$2 $4[N];", null, 0),
71 // protected readonly TElement Zero;
72 // TElement Zero;
73 (new Regex(@"(protected|private) readonly ([a-zA-Z<>0-9]+) ([_a-zA-Z0-9]+);"), "$2
    ↳ $3;", null, 0),
74 // private
75 //
76 (new Regex(@"(\W)(private|protected|public|internal) "), "$1", null, 0),
77 // SizeBalancedTree(int capacity) => a = b;
78 // SizeBalancedTree(int capacity) { a = b; }
79 (new Regex(@"(^\\s+)(override )?(void )?([a-zA-Z0-9]+)\\(((\\[\\])\\)\\s+=>\\s+([~;]+);"),
    ↳ "$1$2$3$4($5) { $6; }", null, 0),
80 // () => Integer<TElement>.Zero,
81 // () { return Integer<TElement>.Zero; },
82 (new Regex(@"\\(\\)\\s+=>\\s+([~\\r\\n;]+?);"), "() { return $1; }", null, 0),
83 // => Integer<TElement>.Zero;
84 // { return Integer<TElement>.Zero; }
85 (new Regex(@"\\(\\)\\s+=>\\s+([~\\r\\n;]+?);"), "() { return $1; }", null, 0),
86 // () { return avlTree.Count; }
87 // []()-> auto { return avlTree.Count; }
88 (new Regex(@"\\(\\) { return ([~;]+); }"), "[]()-> auto { return $1; }", null, 0),
89 // Count => GetSizeOrZero(Root);
90 // GetCount() { return GetSizeOrZero(Root); }
91 (new Regex(@"([A-Z][a-z]+)\\s+=>\\s+([~;]+);"), "Get$1() { return $2; }", null, 0),
92 // var
93 // auto
94 (new Regex(@"(\\W)var(\\W)"), "$1auto$2", null, 0),
95 // unchecked
96 //
97 (new Regex(@"[\\r\\n]{2}\\s*?unchecked\\s*?$"), "", null, 0),
98 // $"
99 // "
100 (new Regex(@"\\$"""), "\\\"", null, 0),
101 // Console.WriteLine("...")
102 // printf("...\\n")
103 (new Regex(@"Console\\.WriteLine\\(\"([~\"']+)\")\\", "printf(\"$1\\n\")", null, 0),
104 // throw new InvalidOperationException
105 // throw std::exception
106 (new Regex(@"throw new (InvalidOperationException|Exception)", "throw
    ↳ std::exception", null, 0),
107 // override void PrintNode(TElement node, StringBuilder sb, int level)
108 // void PrintNode(TElement node, StringBuilder sb, int level) override
109 (new Regex(@"override ([a-zA-Z0-9 \\*+]+)\\((\\[\\])\\)+)", "$1$2 override", null, 0),
110 // string
111 // char*
112 (new Regex(@"(\\W)string(\\W)"), "$1char*$2", null, 0),
113 // sbyte
114 // std::int8_t
115 (new Regex(@"(\\W)sbyte(\\W)"), "$1std::int8_t$2", null, 0),
116 // uint
117 // std::uint32_t
118 (new Regex(@"(\\W)uint(\\W)"), "$1std::uint32_t$2", null, 0),
119 // char*[] args
120 // char* args[]
121 (new Regex(@"([_a-zA-Z0-9:~*+]?)[\\] ([a-zA-Z0-9]+)", "$1 $2[]", null, 0),
122 // using Platform.Numbers;
123 //
124 (new Regex(@"([\\r\\n]{2}|^~)\\s*?using [\\.a-zA-Z0-9+;\\s*?$]", "", null, 0),
125 // struct TreeElement { }
126 // struct TreeElement { };
127 (new Regex(@"(struct|class) ([a-zA-Z0-9]+)(\\s+){([\\sa-zA-Z0-9;:_]+?)}([~;])", "$1
    ↳ $2$3{$4};$5", null, 0),
128 // class Program { }
129 // class Program { };
130 (new Regex(@"(struct|class) ([a-zA-Z0-9]+)[~\\r\\n]*([\\r\\n]+(?<indentLevel>[\\t
    ↳ ]*))?){([\\S\\s]+?[\\r\\n]+\\k<indentLevel>)}([~;]|$)", "$1 $2$3{$4};$5", null, 0),
131 // class SizedBinaryTreeMethodsBase : GenericCollectionMethodsBase
132 // class SizedBinaryTreeMethodsBase : public GenericCollectionMethodsBase
133 (new Regex(@"class ([a-zA-Z0-9]+) : ([a-zA-Z0-9]+)", "class $1 : public $2", null,
    ↳ 0),
134 }.Cast<ISubstitutionRule>().ToList();
135

```

```

136 public static readonly IList<ISubstitutionRule> LastStage = new List<SubstitutionRule>
137 {
138     // (expression)
139     // expression
140     (new Regex(@"\(|\)|\((([a-zA-Z0-9_]*:)+)\)|(|;|\\)"), "$1$2$3", null, 0),
141     // (method(expression))
142     // method(expression)
143     (new Regex(@"(?<firstSeparator>\(|\)|\(((?<method>[a-zA-Z0-9_\->]*:)+)\(((?<expression>((?<parenthesis>\(|\)|(?<-parent_
    ↪ hesis>)\)|[a-zA-Z0-9_\->]*:)+)(?(parenthesis)(?!))\)\)\((?<lastSeparator>(|;|\\))\)", "${firstSeparator}${method}${expression}${lastSeparator}", null, 0),
144     // ref sizeBalancedTree2.Root
145     // &sizeBalancedTree2.Root
146     (new Regex(@"ref ([a-zA-Z0-9_]+)\.([a-zA-Z0-9_]*+)", "&$1->$2", null, 0),
147     // ref GetElement(node).Right
148     // &GetElement(node).Right
149     (new Regex(@"ref ([a-zA-Z0-9_]+)\((([a-zA-Z0-9_]*+)\)\.([a-zA-Z0-9_]+)",
    ↪ "&$1($2)->$3", null, 0),
150     // GetElement(node).Right
151     // GetElement(node)->Right
152     (new Regex(@"([a-zA-Z0-9_]+)\((([a-zA-Z0-9_]*+)\)\.([a-zA-Z0-9_]+)", "$1($2)->$3",
    ↪ null, 0),
153     // = ref GetLeftReference(root);
154     // = GetLeftReference(root);
155     (new Regex(@" = ref ([a-zA-Z0-9_]+)\((([a-zA-Z0-9_]*+)\);", " = $1($2);", null, 0),
156     // ref this->GetElement(node)
157     // this->GetElement(node)
158     (new Regex(@"ref this->([a-zA-Z0-9_]+)\((([a-zA-Z0-9_]*+)\)", "this->$1($2)", null,
    ↪ 0),
159     // ref GetElement(node)
160     // GetElement(node)
161     (new Regex(@"ref ([a-zA-Z0-9_]+)\((([a-zA-Z0-9_]*+)\)", "$1($2)", null, 0),
162     // = ref left;
163     // = left;
164     (new Regex(@" = ref ([a-zA-Z0-9_]+);", " = $1;", null, 0),
165     // (ref left)
166     // (left)
167     (new Regex(@"\((ref ([a-zA-Z0-9_]+)\)|\(|,)", "($1$2", null, 0),
168     // ref TElement
169     // TElement*
170     (new Regex(@"( |\()ref ([a-zA-Z0-9_]+ )", "$1$2* ", null, 0),
171     // return ref _elements[node];
172     // return &elements[node];
173     (new Regex(@"return ref ([_a-zA-Z0-9_]+)\([([_a-zA-Z0-9_]*+)\];", "return &$1[$2];",
    ↪ null, 0),
174     // default
175     // 0
176     (new Regex(@"(\W)default(\W)", "${1}0$2", null, 0),
177     // // #define ENABLE_TREE_AUTO_DEBUG_AND_VALIDATION
178     //
179     (new Regex(@"\\\/[ \t]*#define[ \t]+[_a-zA-Z0-9_]+[ \t]*")", "", null, 0),
180     // #if USEARRAYPOOL\r\n#endif
181     //
182     (new Regex(@"#if [a-zA-Z0-9_]+\s+#endif")", "", null, 0),
183     // \n ... namespace
184     // namespace
185     (new Regex(@"(\\S[\\r\\n]{1,2})?[\\r\\n]+namespace)", "$1namespace", null, 0),
186     // \n ... class
187     // class
188     (new Regex(@"(\\S[\\r\\n]{1,2})?[\\r\\n]+class)", "$1class", null, 0),
189 }.Cast<ISubstitutionRule>().ToList();
190
191 public CSharpToCppTransformer(IList<ISubstitutionRule> extraRules) :
    ↪ base(FirstStage.Concat(extraRules).Concat(LastStage).ToList()) { }
192
193 public CSharpToCppTransformer() : base(FirstStage.Concat(LastStage).ToList()) { }
194 }
195 }

```

./Platform.RegularExpressions.Transformer.CSharpToCpp/obj/Release/netstandard2.1/Platform.RegularExpressions.

```

1  //-----
2  // <auto-generated>
3  //     Generated by the MSBuild WriteCodeFragment class.
4  // </auto-generated>
5  //-----
6
7  using System;

```

```

8 using System.Reflection;
9
10 [assembly: System.Reflection.AssemblyConfigurationAttribute("Release")]
11 [assembly: System.Reflection.AssemblyCopyrightAttribute("Konstantin Diachenko")]
12 [assembly: System.Reflection.AssemblyDescriptionAttribute("LinksPlatform\'s
↳ Platform.RegularExpressions.Transformer.CSharpToCpp Class Library" +
13 "")]
14 [assembly: System.Reflection.AssemblyFileVersionAttribute("0.0.2.0")]
15 [assembly: System.Reflection.AssemblyInformationalVersionAttribute("0.0.2")]
16 [assembly: System.Reflection.AssemblyTitleAttribute("Platform.RegularExpressions.Transformer.CSh
↳ arpToCpp")]
17 [assembly: System.Reflection.AssemblyVersionAttribute("0.0.2.0")]

```

./Platform.RegularExpressions.Transformer.CSharpToCpp.Tests/CSharpToCppTransformerTests.cs

```

1 using Xunit;
2
3 namespace Platform.RegularExpressions.Transformer.CSharpToCpp.Tests
4 {
5     public class CSharpToCppTransformerTests
6     {
7         [Fact]
8         public void HelloWorldTest()
9         {
10             const string helloWorldCode = @"using System;
11 class Program
12 {
13     public static void Main(string[] args)
14     {
15         Console.WriteLine("Hello, world!");
16     }
17 }";
18             const string expectedResult = @"class Program
19 {
20     public:
21     static void Main(char* args[])
22     {
23         printf("Hello, world!\n");
24     }
25 };";
26             var transformer = new CSharpToCppTransformer();
27             var actualResult = transformer.Transform(helloWorldCode, new Context(null));
28             Assert.Equal(expectedResult, actualResult);
29         }
30     }
31 }

```

./Platform.RegularExpressions.Transformer.CSharpToCpp.Tests/obj/Release/netcoreapp3.0/Platform.RegularExpres

```

1 //-----
2 // <auto-generated>
3 //     Generated by the MSBuild WriteCodeFragment class.
4 // </auto-generated>
5 //-----
6
7 using System;
8 using System.Reflection;
9
10 [assembly: System.Reflection.AssemblyCompanyAttribute("Platform.RegularExpressions.Transformer.C
↳ SharpToCpp.Tests")]
11 [assembly: System.Reflection.AssemblyConfigurationAttribute("Release")]
12 [assembly: System.Reflection.AssemblyFileVersionAttribute("1.0.0.0")]
13 [assembly: System.Reflection.AssemblyInformationalVersionAttribute("1.0.0")]
14 [assembly: System.Reflection.AssemblyProductAttribute("Platform.RegularExpressions.Transformer.C
↳ SharpToCpp.Tests")]
15 [assembly: System.Reflection.AssemblyTitleAttribute("Platform.RegularExpressions.Transformer.CSh
↳ arpToCpp.Tests")]
16 [assembly: System.Reflection.AssemblyVersionAttribute("1.0.0.0")]

```

Index

- ./Platform.RegularExpressions.Transformer.CSharpToCpp.Tests/CSharpToCppTransformerTests.cs, 4
- ./Platform.RegularExpressions.Transformer.CSharpToCpp.Tests/obj/Release/netcoreapp3.0/Platform.RegularExpressions.Transform4
- ./Platform.RegularExpressions.Transformer.CSharpToCpp/CSharpToCppTransformer.cs, 1
- ./Platform.RegularExpressions.Transformer.CSharpToCpp/obj/Release/netstandard2.1/Platform.RegularExpressions.Transform3