

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

1.1 ./csharp/Platform.RegularExpressions.Transformer.HasuraSQLSimplifier/HasuraSQLSimplifierTransformer.cs

```

1 using System.Collections.Generic;
2 using System.Linq;
3 using System.Text.RegularExpressions;
4
5 #pragma warning disable CS1591 // Missing XML comment for publicly visible type or member
6
7 namespace Platform.RegularExpressions.Transformer.HasuraSQLSimplifier
8 {
9     /// <summary>
10    /// <para>
11    /// Represents the hasura sql simplifier transformer.
12    /// </para>
13    /// <para></para>
14    /// </summary>
15    /// <seealso cref="TextTransformer"/>
16    public class HasuraSQLSimplifierTransformer : TextTransformer
17    {
18        /// <summary>
19        /// <para>
20        /// The to list.
21        /// </para>
22        /// <para></para>
23        /// </summary>
24        public static readonly IList<ISubstitutionRule> DefaultRules = new List<SubstitutionRule>
25        {
26            // HTML clean up
27            (new Regex(@"<span class=""[^"""]*"">>(^[^<]*)</span>", "$1", 0),
28             ('describe'),
29             'describe'
30            (new Regex(@"\([\\s\\n]*('['']+')[\\s\\n]*\)"), "$1", int.MaxValue),
31             AND ('true' AND 'true')
32            //
33            (new Regex(@"\[\\s\\n]*AND\[\\s\\n]*\([\\s\\n]*'true'[\\s\\n]*AND\[\\s\\n]*'true'[\\s\\n]*\)"), "",
34              → 0),
35            // AND ('true')
36            //
37            (new Regex(@"\[\\s\\n]*AND\[\\s\\n]*'true'"), "", 0),
38            // ::
39            (new Regex(@"\[\\s]*::\[\\s]*"), "::", 0),
40            // ('describe':text)
41            // 'describe':text
42            (new Regex(@"\([\\s\\n]*('['']+'::text)[\\s\\n]*\)"), "$1", 0),
43            // ("_0__be_0_nodes"."target_id")
44            // "_0__be_0_nodes"."target_id"
45            (new Regex(@"\([\\s\\n]*("[^"""]+") [\\s\\n]*\. [\\s\\n]*("[^"""]+") [\\s\\n]*\)"), "$1.$2",
46              → 0),
47            // ("public"."nodes"."_id")
48            // "public"."nodes"."_id"
49            (new Regex(@"\([\\s\\n]*(""[^"""]+"") [\\s\\n]*\. [\\s\\n]*(""[^"""]+"") [\\s\\n]*\. [\\s\\n]*(""[^"""]+"") [\\s\\n]*\)"), "$1.$2.$3",
50              → 0),
51            // LIMIT\n\t\t\tt1
52            // LIMIT 1
53            (new Regex(@"(LIMIT) [\\s\\n]*(\\d+)"), "$1 $2", 0),
54            // ("_0__be_0_nodes"."type" = 'describe':text)
55            // "_0__be_0_nodes"."type" = 'describe':text
56            (new Regex(@"(\W)\([\\s\\n]*((?!SELECT) [^\s\n(){}])*\([^\s\n(){}]\)*\) [\\s\\n]*\)"), "$1$2",
57              → int.MaxValue),
58            // (EXISTS (...))
59            // EXISTS (...)
60            (new Regex(@"(\W)\([\\s\\n]*((?!SELECT) [^\s\n(){}])*\([^\s\n(){}]\)*\) [\\s\\n]*\)"), "$1$2",
61              → int.MaxValue),
62            } .Cast<ISubstitutionRule>().ToList();
63
64    /// <summary>
65    /// <para>
66    /// Initializes a new<see cref="HasuraSQLSimplifierTransformer"/> instance.
67    /// </para>
68    /// <para></para>
69    /// </summary>

```

```

69     public HasuraSQLSimplifierTransformer()
70         : base(DefaultRules)
71     {
72     }
73 }
74 }

```

1.2 ./csharp/Platform.RegularExpressions.Transformer.HasuraSQLSimplifier.Tests/HasuraSQLSimplifierTransformerTests

```

1  using Xunit;
2
3  namespace Platform.RegularExpressions.Transformer.HasuraSQLSimplifier.Tests
4  {
5      /// <summary>
6      /// <para>
7      /// Represents the hasura sql simplifier transformer tests.
8      /// </para>
9      /// <para></para>
10     /// </summary>
11     public class HasuraSQLSimplifierTransformerTests
12     {
13         /// <summary>
14         /// <para>
15         /// Tests that empty line test.
16         /// </para>
17         /// <para></para>
18         /// </summary>
19         [Fact]
20         public void EmptyLineTest()
21         {
22             // This test can help to test basic problems with regular expressions like incorrect
23             // ↪ syntax
24             var transformer = new HasuraSQLSimplifierTransformer();
25             var actualResult = transformer.Transform("");
26             Assert.Equal("", actualResult);
27         }
28
29         /// <summary>
30         /// <para>
31         /// Tests that basic request test.
32         /// </para>
33         /// <para></para>
34         /// </summary>
35         [Fact]
36         public void BasicRequestTest()
37         {
38             var original = @"SELECT
39 coalesce(json_agg("root"), '[]') AS "root"
40 FROM
41 (
42     SELECT
43         row_to_json(
44             (
45                 SELECT
46                     ""_2_e""
47                 FROM
48                     (
49                         SELECT
50                             ""_1_root.base"".id AS id""
51                         ) AS ""_2_e""
52             ) AS ""root""
53     FROM
54         (
55             SELECT
56                 *
57             FROM
58                 ""public"".nodes""
59             WHERE
60                 (
61                     (""public"".nodes.type") = (('auth_token') :: text)
62                 )
63             AND (
64                 EXISTS (
65                     SELECT
66                         1
67                     FROM
68                         ""public"".nodes AS ""_0__be_0_nodes""
69                     WHERE
70                         (
71

```

```

72         (
73             (
74                 ("_0__be_0_nodes"."_source_id") = ("public"."nodes"."_id")
75             )
76             AND ('true')
77         )
78         AND (
79             (
80                 (
81                     ("_0__be_0_nodes"."type") = (('describe') :: text))
82                     AND ('true')
83                 )
84                 AND (
85                     (
86                         (
87                             ("_0__be_0_nodes"."target_id") = (('X-Hasura-User-Id') :: text)
88                         )
89                         AND ('true')
90                     )
91                     AND ('true')
92                 )
93             )
94             AND (
95                 ('true')
96                 AND ('true')
97             )
98         )
99     )
100 )
101 )
102 ) AS ""_1_root.base""
103 LIMIT
104 1
105 ) AS ""_3_root"";
106
107     var expected = @"SELECT
108 coalesce(json_agg(""root""), '[]') AS ""root""
109 FROM
110 (
111     SELECT
112         row_to_json(
113             (
114                 SELECT
115                     ""_2_e""
116                 FROM
117                     (
118                         SELECT
119                             ""_1_root.base"."id"" AS ""id""
120                         ) AS ""_2_e""
121                     )
122             ) AS ""root""
123         FROM
124         (
125             SELECT
126                 *
127             FROM
128                 ""public"."nodes""
129             WHERE
130                 ""public"."nodes"."type"" = 'auth_token'::text
131                 AND EXISTS (
132                     SELECT
133                         1
134                     FROM
135                         ""public"."nodes"" AS ""_0__be_0_nodes""
136                     WHERE
137                         ""_0__be_0_nodes"."_source_id"" = ""public"."nodes"."_id""
138                         AND ""_0__be_0_nodes"."type"" = 'describe'::text
139                         AND ""_0__be_0_nodes"."target_id"" = 'X-Hasura-User-Id'::text
140                     )
141                 ) AS ""_1_root.base""
142             LIMIT 1
143         ) AS ""_3_root"";
144     var transformer = new HasuraSQLSimplifierTransformer();
145     var actual = transformer.Transform(original);
146     Assert.Equal(expected, actual);
147 }
148 }
149 }
150 }

```

Index

./csharp/Platform.RegularExpressions.Transformer.HasuraSQLSimplifier.Tests/HasuraSQLSimplifierTransformerTests.cs, 2
./csharp/Platform.RegularExpressions.Transformer.HasuraSQLSimplifier/HasuraSQLSimplifierTransformer.cs, 1