

Summary

Generated on: 06.03.2014 - 10:13:02
Parser: OpenCoverParser
Assemblies: 2
Classes: 44
Files: 44
Coverage: 59.3%
Covered lines: 1570
Uncovered lines: 1076
Coverable lines: 2646
Total lines: 7601

Assemblies

Knot3	66.5%
Knot3.Framework.Audio.Knot3Sound	100%
Knot3.Game.Audio.Knot3AudioManager	90.9%
Knot3.Game.Data.Challenge	0%
Knot3.Game.Data.ChallengeFileIO	0%
Knot3.Game.Data.ChallengeMetaData	0%
Knot3.Game.Data.CircleEntry'1	92.3%
Knot3.Game.Data.CircleExtensions	100%
Knot3.Game.Data.Direction	100%
Knot3.Game.Data.Edge	100%
Knot3.Game.Data.Knot	84.8%
Knot3.Game.Data.KnotFileIO	27.7%
Knot3.Game.Data.KnotMetaData	64%
Knot3.Game.Data.KnotStringIO	73.1%
Knot3.Game.Data.Node	73.7%
Knot3.Game.Data.NodeMap	88.5%
Knot3.Game.Data.RectangleMap	0%
Knot3.Game.Utilities.FileIndex	44.4%
Knot3.Game.Utilities.SavegameLoader'2	38.7%
Knot3.Framework	51.9%
Knot3.Framework.Audio.AudioManager	15.5%
Knot3.Framework.Audio.LoopPlaylist	0%
Knot3.Framework.Audio.OggVorbisFile	0%
Knot3.Framework.Audio.Sound	28.5%
Knot3.Framework.Audio.SoundEffectFile	0%
Knot3.Framework.Core.Camera	65.3%
Knot3.Framework.Core.DisplayLayer	98.1%
Knot3.Framework.Core.World	15.9%
Knot3.Framework.Math.Angles3	100%
Knot3.Framework.Math.BoundingCylinder	90.4%
Knot3.Framework.Math.Bounds	94.5%
Knot3.Framework.Math.RayExtensions	68%
Knot3.Framework.Math.ScreenPoint	51.8%
Knot3.Framework.Platform.SystemInfo	100%
Knot3.Framework.Storage.BooleanOption	100%
Knot3.Framework.Storage.Config	66.6%
Knot3.Framework.Storage.ConfigFile	100%
Knot3.Framework.Storage.DistinctOption	100%
Knot3.Framework.Storage.FileUtility	50%
Knot3.Framework.Storage.FloatOption	92%
Knot3.Framework.Storage.IniFile	60.6%
Knot3.Framework.Storage.KeyOption	100%
Knot3.Framework.Storage.Language	0%
Knot3.Framework.Storage.LanguageOption	0%

Knot3.Framework.Storage.Localizer	0%
Knot3.Framework.Storage.Option	100%

Knot3.Framework.Audio.Knot3Sound

Summary

Class:	Knot3.Framework.Audio.Knot3Sound
Assembly:	Knot3
File(s):	:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Audio\Knot3Sound.cs
Coverage:	100%
Covered lines:	5
Uncovered lines:	0
Coverable lines:	5
Total lines:	42

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.cctor()	1	100	100

File(s)

:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Audio\Knot3Sound.cs

#	Line	Coverage
	1	/*
	2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
	3	* Gerd Augsburg, Christina Erler, Daniel Warzel
	4	*
	5	* This source code file is part of Knot3. Copying, redistribution and
	6	* use of the source code in this file in source and binary forms,
	7	* with or without modification, are permitted provided that the conditions
	8	* of the MIT license are met:
	9	*
	10	* Permission is hereby granted, free of charge, to any person obtaining a c
	11	* of this software and associated documentation files (the "Software"), to
	12	* in the Software without restriction, including without limitation the rig
	13	* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
	14	* copies of the Software, and to permit persons to whom the Software is
	15	* furnished to do so, subject to the following conditions:
	16	*
	17	* The above copyright notice and this permission notice shall be included i
	18	* copies or substantial portions of the Software.
	19	*
	20	* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
	21	* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
	22	* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
	23	* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
	24	* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
	25	* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
	26	* SOFTWARE.
	27	*
	28	* See the LICENSE file for full license details of the Knot3 project.
	29	*/
	30	
	31	using System.Diagnostics.CodeAnalysis;
	32	
	33	namespace Knot3.Framework.Audio
	34	{
	35	public static class Knot3Sound {
1	36	public static readonly Sound CreativeMusic = new Sound ("CreativeMusic
1	37	public static readonly Sound ChallengeMusic = new Sound ("ChallengeMus

```
1 38      public static readonly Sound MenuMusic = new Sound ("MenuMusic");
1 39      public static readonly Sound PipeMoveSound = new Sound ("PipeMoveSound
1 40      public static readonly Sound PipeInvalidMoveSound = new Sound ("PipeIn
  41      }
  42  }
```

Knot3.Game.Audio.Knot3AudioManager

Summary

Class:	Knot3.Game.Audio.Knot3AudioManager
Assembly:	Knot3
File(s):	\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Audio\Knot3AudioManager.cs
Coverage:	90.9%
Covered lines:	10
Uncovered lines:	1
Coverable lines:	11
Total lines:	57

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
Initialize(...)	1	87.5	100

File(s)

\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Audio\Knot3AudioManager.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                                     Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Diagnostics.CodeAnalysis;
33
34 using Knot3.Framework.Audio;
35 using Knot3.Framework.Core;

```

```
36
37 namespace Knot3.Game.Audio
38 {
39     public class Knot3AudioManager : AudioManager
40     {
41         public Knot3AudioManager (GameCore game)
42             : base (game)
43         {
44         }
45
46         public override void Initialize (string directory)
47         {
48             AudioDirectories [Knot3Sound.CreativeMusic] = "Music/Creative";
49             AudioDirectories [Knot3Sound.ChallengeMusic] = "Music/Challenge";
50             AudioDirectories [Knot3Sound.MenuMusic] = "Music/Menu";
51             AudioDirectories [Knot3Sound.PipeMoveSound] = "Sound/Pipe/Move";
52             AudioDirectories [Knot3Sound.PipeInvalidMoveSound] = "Sound/Pipe/I
53
54             base.Initialize (directory);
55         }
56     }
57 }
```

Knot3.Game.Data.Challenge

Summary

Class: Knot3.Game.Data.Challenge
Assembly: Knot3
File(s): c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Challenge.cs
Coverage: 0%
Covered lines: 0
Uncovered lines: 20
Coverable lines: 20
Total lines: 103

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	0	0
AddToHighscore(...)	1	0	0
Save()	1	0	0

File(s)

c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Challenge.cs

```

#   Line Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31  using System.Collections.Generic;
32  using System.Diagnostics.CodeAnalysis;
33
34  namespace Knot3.Game.Data

```

```

35  {
36      /// <summary>
37      /// Ein Objekt dieser Klasse repräsentiert eine Challenge.
38      /// </summary>
39      public sealed class Challenge
40      {
41          /// <summary>
42          /// Der Ausgangsknoten, den der Spieler in den Referenzknoten transfor
43          /// </summary>
0 44          public Knot Start { get; private set; }
45
46          /// <summary>
47          /// Der Referenzknoten, in den der Spieler den Ausgangsknoten transfor
48          /// </summary>
0 49          public Knot Target { get; private set; }
50
51          /// <summary>
52          /// Eine sortierte Bestenliste.
53          /// </summary>
0 54          private SortedList<int, string> highscore { get; set; }
55
56          /// <summary>
57          /// Ein ffentlicher Enumerator, der die Bestenliste unabhnig von de
58          /// </summary>
0 59          public IEnumerable<KeyValuePair<string, int>> Highscore { get { return
60
61          /// <summary>
62          /// Die Metadaten der Challenge.
63          /// </summary>
0 64          public ChallengeMetaData MetaData { get; private set; }
65
66          /// <summary>
67          /// Der Name der Challenge.
68          /// </summary>
69          public string Name
70          {
0 71              get { return MetaData.Name; }
0 72              set { MetaData.Name = value; }
73          }
74
75          /// <summary>
76          /// Erstellt ein Challenge-Objekt aus einem gegebenen Challenge-Metada
77          /// Erstellt ein Challenge-Objekt aus einer gegebenen Challenge-Datei.
78          /// </summary>
0 79          public Challenge (ChallengeMetaData meta, Knot start, Knot target)
0 80          {
0 81              MetaData = meta;
0 82              Start = start;
0 83              Target = target;
0 84          }
85
86          /// <summary>
87          /// Fgt eine neue Bestzeit eines bestimmten Spielers in die Bestenlis
88          /// </summary>
89          public void AddToHighscore (string name, int time)
0 90          {
0 91              MetaData.AddToHighscore (name, time);
0 92              Save ();
0 93          }
94
95          /// <summary>

```



```
96      /// Speichert die Challenge.
97      /// </summary>
98      public void Save ()
0 99      {
0 100          Metadata.Format.Save (this);
0 101      }
102  }
103 }
```

Knot3.Game.Data.ChallengeFileIO

Summary

Class:	Knot3.Game.Data.ChallengeFileIO
Assembly:	Knot3
File(s):	ers\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\ChallengeFileIO.cs
Coverage:	0%
Covered lines:	0
Uncovered lines:	95
Coverable lines:	95
Total lines:	232

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor()	1	0	0
Save(...)	2	0	0
Load(...)	9	0	0
LoadMetaData(...)	11	0	0
MoveNext()	8	0	0
MoveNext()	5	0	0
MoveNext()	7	0	0

File(s)

ers\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\ChallengeFileIO.cs

```

#   Line Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *   Permission is hereby granted, free of charge, to any person obtaining a c
11  *   of this software and associated documentation files (the "Software"), to
12  *   in the Software without restriction, including without limitation the rig
13  *   to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *   copies of the Software, and to permit persons to whom the Software is
15  *   furnished to do so, subject to the following conditions:
16  *
17  *   The above copyright notice and this permission notice shall be included i
18  *   copies or substantial portions of the Software.
19  *
20  *   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30

```

```

31 using System;
32 using System.Collections;
33 using System.Collections.Generic;
34 using System.Diagnostics.CodeAnalysis;
35 using System.IO;
36
37 using Ionic.Zip;
38
39 using Knot3.Framework.Platform;
40
41 namespace Knot3.Game.Data
42 {
43     /// <summary>
44     /// Implementiert das Speicherformat fr Challenges.
45     /// </summary>
46     public sealed class ChallengeFileIO : IChallengeIO
47     {
48         /// <summary>
49         /// Die fr eine Knoten-Datei gltigen Dateieindungen.
50         /// </summary>
0 51         public IEnumerable<string> FileExtensions
52         {
53             get {
0 54                 yield return ".challenge";
0 55                 yield return ".chl";
0 56                 yield return ".chn";
0 57                 yield return ".chg";
0 58                 yield return ".chlng";
0 59             }
60         }
61
62         /// <summary>
63         /// Erstellt ein ChallengeFileIO-Objekt.
64         /// </summary>
0 65         public ChallengeFileIO ()
0 66         {
0 67         }
68
69         /// <summary>
70         /// Speichert eine Challenge in dem Dateinamen, der in dem Challenge-0
71         /// </summary>
0 72         public void Save (Challenge challenge)
0 73         {
0 74             using (ZipFile zip = new ZipFile ()) {
75                 // Namen
0 76                 zip.AddEntry ("name.txt", challenge.Name);
77                 // Startknoten
0 78                 KnotStringIO parser = new KnotStringIO (challenge.Start);
0 79                 zip.AddEntry ("start.knot", parser.Content);
80                 // Zielknoten
0 81                 parser = new KnotStringIO (challenge.Target);
0 82                 zip.AddEntry ("target.knot", parser.Content);
83                 // Highscore
0 84                 zip.AddEntry ("highscore.txt", string.Join ("\n", printHighsco
85                 // ZIP-Datei speichern
0 86                 zip.Save (challenge.Metadata.Filename);
0 87             }
0 88         }
89
90         /// <summary>
91         /// Ldt eine Challenge aus einer angegebenen Datei.

```

```

92      /// </summary>
93      public Challenge Load (string filename)
94      {
95          ChallengeMetaData meta = LoadMetaData (filename: filename);
96          Knot start = null;
97          Knot target = null;
98
99          using (ZipFile zip = ZipFile.Read (filename)) {
100              foreach (ZipEntry entry in zip) {
101                  string content = entry.ReadContent ();
102
103                  // fr die Datei mit dem Startknoten
104                  if (entry.FileName.ToLower ().Contains ("start")) {
105                      KnotStringIO parser = new KnotStringIO (content: conte
106                      start = new Knot (
107                          new KnotMetaData (parser.Name, () => parser.CountE
108                          parser.Edges
109                      );
110                  }
111
112                  // fr die Datei mit dem Zielknoten
113                  else if (entry.FileName.ToLower ().Contains ("target")) {
114                      KnotStringIO parser = new KnotStringIO (content: conte
115                      target = new Knot (
116                          new KnotMetaData (parser.Name, () => parser.CountE
117                          parser.Edges
118                      );
119                  }
120              }
121          }
122
123          if (meta != null && start != null && target != null) {
124              return new Challenge (meta, start, target);
125          }
126          else {
127              throw new IOException (
128                  "Error! Invalid challenge file: " + filename
129                  + " (meta=" + meta + ",start=" + start + ",target=" + targ
130              );
131          }
132      }
133
134      /// <summary>
135      /// Ldt die Metadaten einer Challenge aus einer angegebenen Datei.
136      /// </summary>
137      public ChallengeMetaData LoadMetaData (string filename)
138      {
139          string name = null;
140          KnotMetaData start = null;
141          KnotMetaData target = null;
142          IEnumerable<KeyValuePair<string, int>> highscore = null;
143          using (ZipFile zip = ZipFile.Read (filename)) {
144              foreach (ZipEntry entry in zip) {
145                  string content = entry.ReadContent ();
146
147                  // fr die Datei mit dem Startknoten
148                  if (entry.FileName.ToLower ().Contains ("start")) {
149                      KnotStringIO parser = new KnotStringIO (content: conte
150                      start = new KnotMetaData (parser.Name, () => parser.Co
151                  }
152

```

```

153         // fr die Datei mit dem Zielknoten
0 154     else if (entry.FileName.ToLower ().Contains ("target")) {
0 155         KnotStringIO parser = new KnotStringIO (content: conte
0 156         target = new KnotMetaData (parser.Name, () => parser.C
0 157     }
158
159     // fr die Datei mit dem Namen
0 160     else if (entry.FileName.ToLower ().Contains ("name")) {
0 161         name = content.Trim ();
0 162     }
163
164     // fr die Datei mit den Highscores
0 165     else if (entry.FileName.ToLower ().Contains ("highscore"))
0 166         highscore = parseHighscore (content.Split (new char[]
0 167     }
0 168     }
0 169 }
0 170 if (name != null && start != null && target != null) {
0 171     Log.Debug ("Load challenge file: ", filename, " (name=", name,
0 172     return new ChallengeMetaData (
173         name: name,
174         start: start,
175         target: target,
176         filename: filename,
177         format: this,
178         highscore: highscore
179     );
180 }
0 181 else {
0 182     throw new IOException (
183         "Error! Invalid challenge file: " + filename
184         + " (name=" + name + ",start=" + start + ",target=" + targ
185     );
186 }
0 187 }
188
189 IEnumerable<string> printHighscore (IEnumerable<KeyValuePair<string, i
0 190 {
0 191     foreach (KeyValuePair<string, int> entry in highscore) {
0 192         Log.Debug (
193             "Save Highscore: "
194             + entry.Value.ToString ()
195             + ":"
196             + entry.Key.ToString ()
197         );
198
0 199         yield return entry.Value + ":" + entry.Key;
0 200     }
0 201 }
202
203 IEnumerable<KeyValuePair<string, int>> parseHighscore (IEnumerable<str
0 204 {
0 205     foreach (string line in highscore) {
0 206         Log.Debug ("Load Highscore: ",line);
0 207         if (line.Contains (":")) {
0 208             string[] entry = line.Split (new char[] {':'}, 2, StringSp
0 209             string name = entry [1].Trim ();
210             int time;
0 211             if (Int32.TryParse (entry [0], out time)) {
0 212                 Log.Debug ("=> ", name, ":", time);
0 213                 yield return new KeyValuePair<string, int> (name, time

```

```
0 214      }
0 215      }
0 216      }
0 217      }
218  }
219
220  [ExcludeFromCodeCoverageAttribute]
221  static class ZipHelper
222  {
223      public static string ReadContent (this ZipEntry entry)
224      {
225          MemoryStream memory = new MemoryStream ();
226          entry.Extract (memory);
227          memory.Position = 0;
228          var sr = new StreamReader (memory);
229          return sr.ReadToEnd ();
230      }
231  }
232 }
```

Knot3.Game.Data.ChallengeMetaData

Summary

Class:	Knot3.Game.Data.ChallengeMetaData
Assembly:	Knot3
File(s):	s\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\ChallengeMetaData.cs
Coverage:	0%
Covered lines:	0
Uncovered lines:	82
Coverable lines:	82
Total lines:	211

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	6	0	0
AddToHighscore(...)	2	0	0
formatTime(...)	1	0	0
Equals(...)	2	0	0
Equals(...)	2	0	0
op_Equality(...)	4	0	0
op_Inequality(...)	1	0	0

File(s)

s\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\ChallengeMetaData.cs

```
#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *   Permission is hereby granted, free of charge, to any person obtaining a c
11  *   of this software and associated documentation files (the "Software"), to
12  *   in the Software without restriction, including without limitation the rig
13  *   to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *   copies of the Software, and to permit persons to whom the Software is
15  *   furnished to do so, subject to the following conditions:
16  *
17  *   The above copyright notice and this permission notice shall be included i
18  *   copies or substantial portions of the Software.
19  *
20  *   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
```

```

31 using System;
32 using System.Collections.Generic;
33 using System.Diagnostics.CodeAnalysis;
34 using System.Linq;
35
36 using Knot3.Framework.Platform;
37 using Knot3.Framework.Storage;
38
39 namespace Knot3.Game.Data
40 {
41     /// <summary>
42     /// Enthlt Metadaten zu einer Challenge.
43     /// </summary>
44     public class ChallengeMetaData
45     {
46         /// <summary>
47         /// Der Name der Challenge.
48         /// </summary>
49         public string Name
50         {
51             get {
52                 return name;
53             }
54             set {
55                 name = value;
56                 if (Format == null) {
57                     Format = new ChallengeFileIO ();
58                 }
59                 string extension;
60                 if (Format.FileExtensions.Any ()) {
61                     extension = Format.FileExtensions.ElementAt (0);
62                 }
63                 else {
64                     throw new ArgumentException ("Every implementation of ICha
65                 }
66                 Filename = SystemInfo.SavegameDirectory + SystemInfo.PathSepar
67             }
68         }
69
70         private string name;
71
72         /// <summary>
73         /// Der Ausgangsknoten, den der Spieler in den Referenzknoten transfor
74         /// </summary>
75         public KnotMetaData Start { get; private set; }
76
77         /// <summary>
78         /// Der Referenzknoten, in den der Spieler den Ausgangsknoten transfor
79         /// </summary>
80         public KnotMetaData Target { get; private set; }
81
82         /// <summary>
83         /// Das Format, aus dem die Metadaten der Challenge gelesen wurden ode
84         /// </summary>
85         public IChallengeIO Format { get; private set; }
86
87         /// <summary>
88         /// Der Dateiname, aus dem die Metadaten der Challenge gelesen wurden
89         /// </summary>
90         public string Filename { get; private set; }
91

```



```

92      /// <summary>
93      /// Ein ffentlicher Enumerator, der die Bestenliste unabhnig von de
94      /// </summary>
0 95      public IEnumerable<KeyValuePair<string, int>> Highscore { get { return
96
97      private List<KeyValuePair<string, int>> highscore;
98
99      public float AvgTime
100     {
0 101         get {
0 102             if (    highscore != null
0 103                 && highscore.Any ()) {
0 104                 float amount =0;
0 105                 foreach (KeyValuePair<string, int> entry in highscore) {
0 106                     amount += (float)entry.Value;
0 107                 }
0 108                 return amount/((float)highscore.Count);
109             }
0 110             return 0f;
0 111         }
112
0 113         private set {}
114     }
115
116     public string FormatedAvgTime
117     {
0 118         get {
0 119             float time = AvgTime;
0 120             Log.Debug (time);
0 121             if (time != 0f) {
0 122                 return formatTime (time);
123             }
0 124             return "Not yet set.";
0 125         }
0 126         private set {
0 127             }
128     }
129
130     /// <summary>
131     /// Erstellt ein Challenge-Metadaten-Objekt mit einem gegebenen Namen
132     /// </summary>
0 133     public ChallengeMetaData (string name, KnotMetaData start, KnotMetaDat
134                             string filename, IChallengeIO format,
135                             IEnumerable<KeyValuePair<string, int>> highs
0 136     {
0 137         Name = name;
0 138         Start = start;
0 139         Target = target;
0 140         Format = format ?? Format;
0 141         Filename = filename ?? Filename;
142
0 143         this.highscore = new List<KeyValuePair<string, int>> ();
0 144         if (highscore != null) {
0 145             foreach (KeyValuePair<string, int> entry in highscore) {
0 146                 this.highscore.Add (entry);
0 147             }
0 148         }
0 149     }
150
151     /// <summary>
152     /// Fgt eine neue Bestzeit eines bestimmten Spielers in die Bestenlis

```

```

153     /// </summary>
154     public void AddToHighscore (string name, int time)
155     {
156         KeyValuePair<string, int> entry = new KeyValuePair<string, int> (n
157         if (!highscore.Contains (entry)) {
158             highscore.Add (entry);
159         }
160     }
161
162     public static string formatTime (float secs)
163     {
164         Log.Debug (secs);
165         TimeSpan t = TimeSpan.FromSeconds ( secs );
166
167         string answer = string.Format ("{0:D2}h:{1:D2}m:{2:D2}s",
168                                         t.Hours,
169                                         t.Minutes,
170                                         t.Seconds);
171
172         return answer;
173     }
174
175     public bool Equals (ChallengeMetaData other)
176     {
177         return other != null && name == other.name;
178     }
179
180     public override bool Equals (object other)
181     {
182         return other != null && Equals (other as ChallengeMetaData);
183     }
184
185     [ExcludeFromCodeCoverageAttribute]
186     public override int GetHashCode ()
187     {
188         return (name ?? String.Empty).GetHashCode ();
189     }
190
191     public static bool operator == (ChallengeMetaData a, ChallengeMetaData
192     {
193         // If both are null, or both are same instance, return true.
194         if (System.Object.ReferenceEquals (a, b)) {
195             return true;
196         }
197
198         // If one is null, but not both, return false.
199         if (((object)a == null) || ((object)b == null)) {
200             return false;
201         }
202
203         // Return true if the fields match:
204         return a.Equals (b);
205     }
206
207     public static bool operator != (ChallengeMetaData a, ChallengeMetaData
208     {
209         return !(a == b);
210     }
211 }

```

Knot3.Game.Data.CircleEntry'1

Summary

Class:	Knot3.Game.Data.CircleEntry'1
Assembly:	Knot3
File(s):	:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\CircleEntry.cs
Coverage:	92.3%
Covered lines:	205
Uncovered lines:	17
Coverable lines:	222
Total lines:	384

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
.ctor()	1	100	100
.ctor(...)	4	100	85.71
InsertBefore(...)	1	100	100
InsertAfter(...)	1	100	100
Remove()	1	100	100
Contains(...)	1	100	100
Contains(...)	1	100	100
Contains(...)	1	100	100
Contains(...)	1	100	100
Find(...)	1	100	100
IndexOf(...)	1	100	100
IndexOf(...)	3	92.31	80
System.Collections.I	1	0	0
op_Addition(...)	3	100	100
op_Subtraction(...)	1	100	100
op_Increment(...)	1	100	100
op_Decrement(...)	1	100	100
op_Implicit(...)	1	100	100
Contains(...)	1	100	100
Remove(...)	2	100	100
RemoveAt(...)	1	100	100
Insert(...)	1	0	0
Add(...)	2	100	100
Clear()	1	100	100
CopyTo(...)	3	100	80
MoveNext()	6	100	87.5
MoveNext()	6	100	87.5
MoveNext()	6	81.82	75
MoveNext()	5	0	0
MoveNext()	5	100	83.33
MoveNext()	5	100	83.33

File(s)

:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\CircleEntry.cs

#	Line	Coverage
	1	/*
	2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
	3	* Gerd Augsburg, Christina Erler, Daniel Warzel
	4	*

```

5  * This source code file is part of Knot3. Copying, redistribution and
6  * use of the source code in this file in source and binary forms,
7  * with or without modification, are permitted provided that the conditions
8  * of the MIT license are met:
9  *
10 * Permission is hereby granted, free of charge, to any person obtaining a c
11 * of this software and associated documentation files (the "Software"), to
12 * in the Software without restriction, including without limitation the rig
13 * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14 * copies of the Software, and to permit persons to whom the Software is
15 * furnished to do so, subject to the following conditions:
16 *
17 * The above copyright notice and this permission notice shall be included i
18 * copies or substantial portions of the Software.
19 *
20 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21 * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23 * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24 * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25 * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26 * SOFTWARE.
27 *
28 * See the LICENSE file for full license details of the Knot3 project.
29 */
30
31 using System;
32 using System.Collections;
33 using System.Collections.Generic;
34 using System.Diagnostics.CodeAnalysis;
35 using System.Linq;
36
37 namespace Knot3.Game.Data
38 {
39     /// <summary>
40     /// Eine doppelt verkettete Liste.
41     /// </summary>
42     public class CircleEntry<T> : IEnumerable<T>, ICollection<T>, IList<T>
43     {
44         public T Value { get; set; }
45
46         public CircleEntry<T> Next { get; set; }
47
48         public CircleEntry<T> Previous { get; set; }
49
1188 50         public CircleEntry (T value)
1188 51         {
1188 52             Value = value;
1188 53             Previous = this;
1188 54             Next = this;
1188 55         }
56
7 57         private CircleEntry ()
7 58         {
7 59             Previous = this;
7 60             Next = this;
7 61         }
62
46 63         public CircleEntry (IEnumerable<T> list)
46 64         {
46 65             bool first = true;

```

```

46      66      CircleEntry<T> inserted = this;
3693    67      foreach (T obj in list) {
1231    68          if (first) {
46      69              Value = obj;
46      70              Previous = this;
46      71              Next = this;
46      72          }
1139    73          else {
1139    74              inserted = inserted.InsertAfter (obj);
1139    75          }
1185    76          first = false;
1185    77      }
46      78  }
      79
      80  public static CircleEntry<T> Empty
      81  {
7       82      get {
7       83          return new CircleEntry<T> ();
7       84      }
      85  }
      86
      87  public CircleEntry<T> InsertBefore (T obj)
47      88  {
47      89      CircleEntry<T> insert = new CircleEntry<T> (obj);
47      90      insert.Previous = this.Previous;
47      91      insert.Next = this;
47      92      this.Previous.Next = insert;
47      93      this.Previous = insert;
47      94      return insert;
47      95  }
      96
      97  public CircleEntry<T> InsertAfter (T obj)
1141    98  {
      99      //Log.Debug (this, ".InsertAfter (", obj, ")");
1141   100      CircleEntry<T> insert = new CircleEntry<T> (obj);
1141   101      insert.Next = this.Next;
1141   102      insert.Previous = this;
1141   103      this.Next.Previous = insert;
1141   104      this.Next = insert;
1141   105      return insert;
1141   106  }
      107
      108  public void Remove ()
115     109  {
115     110      Previous.Next = Next;
115     111      Next.Previous = Previous;
115     112      Previous = null;
115     113      Next = null;
115     114  }
      115
      116  private bool IsEmpty
      117  {
29     118      get {
29     119          return (Next == this || Next == null) && (Previous == this ||
29     120              )
      121  }
      122
      123  public int Count
      124  {
27     125      get {
27     126          if (IsEmpty) {

```

```

0      127      return 0;
      128      }
27     129      else {
27     130          CircleEntry<T> current = this;
27     131          int count = 0;
244    132          do {
244    133              ++count;
244    134              current = current.Next;
244    135          }
244    136          while (current != this);
27     137          return count;
      138      }
27     139      }
      140  }
      141
      142  public bool Contains (T obj, out IEnumerable<CircleEntry<T>> item)
199    143  {
199    144      item = Find (obj);
199    145      return item.Count () > 0;
199    146  }
      147
      148  public bool Contains (Func<T, bool> func, out IEnumerable<CircleEntry<
199    149  {
199    150      item = Find (func);
199    151      return item.Count () > 0;
199    152  }
      153
      154  public bool Contains (T obj, out CircleEntry<T> item)
301    155  {
301    156      item = Find (obj).ElementAtOrDefault (0);
301    157      return item != null;
301    158  }
      159
      160  public bool Contains (Func<T, bool> func, out CircleEntry<T> item)
100    161  {
100    162      item = Find (func).ElementAtOrDefault (0);
100    163      return item != null;
100    164  }
      165
      166  public IEnumerable<CircleEntry<T>> Find (T obj)
707    167  {
27279  168      return Find ((t) => t.Equals (obj));
707    169  }
      170
      171  public IEnumerable<CircleEntry<T>> Find (Func<T, bool> func)
1909   172  {
1909   173      CircleEntry<T> current = this;
52516  174      do {
53929  175          if (func (current.Value)) {
1413   176              yield return current;
600    177          }
51703  178          current = current.Next;
51703  179      }
51703  180      while (current != this);
1096   181      yield break;
      182  }
      183
      184  public int IndexOf (T obj)
100    185  {
5150   186      return IndexOf ((t) => t.Equals (obj));
100    187  }

```

	188	
	189	public int IndexOf (Func<T, bool> func)
200	190	{
200	191	int i = 0;
200	192	CircleEntry<T> current = this;
10100	193	do {
10300	194	if (func (current.Value)) {
200	195	return i;
	196	}
9900	197	current = current.Next;
9900	198	++ i;
9900	199	}
9900	200	while (current != this);
0	201	return -1;
200	202	}
	203	
	204	public IEnumerable<T> RangeTo (CircleEntry<T> other)
4	205	{
4	206	CircleEntry<T> current = this;
16	207	do {
16	208	yield return current.Value;
16	209	current = current.Next;
16	210	}
16	211	while (current != other.Next && current != this);
4	212	}
	213	
	214	public IEnumerable<T> WayTo (T other)
1	215	{
1	216	CircleEntry<T> current = this;
3	217	while (!current.Value.Equals (other)) {
1	218	yield return current.Value;
1	219	current = current.Next;
1	220	if (current == this) {
0	221	break;
	222	}
1	223	}
1	224	}
	225	
	226	public IEnumerable<Tuple<T,T>> Pairs
	227	{
0	228	get {
0	229	CircleEntry<T> current = this;
0	230	do {
0	231	yield return Tuple.Create (current.Value, current.Next.Val
0	232	current = current.Next;
0	233	}
0	234	while (current != this);
0	235	}
	236	}
	237	
	238	public IEnumerable<Tuple<T,T,T>> Triples
	239	{
12	240	get {
12	241	CircleEntry<T> current = this;
52	242	do {
52	243	yield return Tuple.Create (current.Previous.Value, current
50	244	current = current.Next;
50	245	}
50	246	while (current != this);
10	247	}
	248	}

	249	
	250	public IEnumerator<T> GetEnumerator ()
98	251	{
98	252	CircleEntry<T> current = this;
936	253	do {
	254	//Log.Debug (this, " => ", current.Content);
936	255	yield return current.Value;
928	256	current = current.Next;
928	257	}
928	258	while (current != this);
90	259	}
	260	
	261	// explicit interface implementation for nongeneric interface
	262	IEnumerator IEnumerable.GetEnumerator ()
0	263	{
0	264	return GetEnumerator (); // just return the generic version
0	265	}
	266	
	267	[ExcludeFromCodeCoverageAttribute]
	268	public override string ToString ()
	269	{
	270	if (IsEmpty) {
	271	return "CircleEntry (" + Value.ToString () + ")";
	272	}
	273	else {
	274	return "CircleEntry.Empty";
	275	}
	276	}
	277	
	278	public static CircleEntry<T> operator + (CircleEntry<T> circle, int i)
579	279	{
579	280	CircleEntry<T> next = circle;
40549	281	while (i > 0) {
19985	282	next = next.Next;
19985	283	i--;
19985	284	}
989	285	while (i < 0) {
205	286	next = next.Previous;
205	287	i++;
205	288	}
579	289	return next;
579	290	}
	291	
	292	public T this [int index]
	293	{
238	294	get {
238	295	return (this + index).Value;
238	296	}
100	297	set {
100	298	(this + index).Value = value;
100	299	}
	300	}
	301	
	302	public static CircleEntry<T> operator - (CircleEntry<T> circle, int i)
30	303	{
30	304	return circle + (-i);
30	305	}
	306	
	307	public static CircleEntry<T> operator ++ (CircleEntry<T> circle)
189	308	{
189	309	return circle.Next;


```

189 310      }
      311
      312      public static CircleEntry<T> operator -- (CircleEntry<T> circle)
11 313      {
11 314          return circle.Previous;
11 315      }
      316
      317      public static implicit operator T (CircleEntry<T> circle)
700 318      {
700 319          return circle.Value;
700 320      }
      321
      322      public bool IsReadOnly { get { return false; } }
      323
      324      public bool Contains (T obj)
102 325      {
102 326          CircleEntry<T> item = Find (obj).ElementAtOrDefault (0);
102 327          return item != null;
102 328      }
      329
      330      public bool Remove (T value)
198 331      {
      332          CircleEntry<T> item;
297 333          if (Contains (value, out item)) {
99 334              item.Remove ();
99 335              return true;
      336          }
      337          else {
99 338              return false;
      339          }
198 340      }
      341
      342      public void RemoveAt (int i)
1 343      {
1 344          (this + i).Remove ();
1 345      }
      346
      347      public void Insert (int i, T value)
0 348      {
0 349          (this + i).InsertBefore (value);
0 350      }
      351
      352      public void Add (T value)
50 353      {
56 354          if (Value == null) {
6 355              Value = value;
6 356          }
44 357          else {
44 358              InsertBefore (value);
44 359          }
50 360      }
      361
      362      public void Clear ()
1 363      {
1 364          Remove ();
1 365          Next = Previous = this;
1 366      }
      367
      368      public void CopyTo (T[] array, int start)
1 369      {
303 370          foreach (T value in this) {

```

```
100 371         array.SetValue (value, start);
100 372         ++start;
100 373     }
    1 374     }
    375     }
    376
    377     public static class CircleExtensions
    378     {
    379         public static CircleEntry<T> ToCircle<T> (this IEnumerable<T> enumerab
    380         {
    381             return new CircleEntry<T> (enumerable);
    382         }
    383     }
    384 }
```

Knot3.Game.Data.CircleExtensions

Summary

Class: Knot3.Game.Data.CircleExtensions
Assembly: Knot3
File(s): :\\Users\\Pascal\\Documents\\GitHub\\knot3-code\\Game\\Knot3.Game\\Data\\CircleEntry.cs
Coverage: 100%
Covered lines: 3
Uncovered lines: 0
Coverable lines: 3
Total lines: 384

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
ToCircle(...)	1	100	100

File(s)

:\\Users\\Pascal\\Documents\\GitHub\\knot3-code\\Game\\Knot3.Game\\Data\\CircleEntry.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                                     Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31  using System;
32  using System.Collections;
33  using System.Collections.Generic;
34  using System.Diagnostics.CodeAnalysis;
35  using System.Linq;
36
37  namespace Knot3.Game.Data

```

```
38 {
39     /// <summary>
40     /// Eine doppelt verkettete Liste.
41     /// </summary>
42     public class CircleEntry<T> : IEnumerable<T>, ICollection<T>, IList<T>
43     {
44         public T Value { get; set; }
45
46         public CircleEntry<T> Next { get; set; }
47
48         public CircleEntry<T> Previous { get; set; }
49
50         public CircleEntry (T value)
51         {
52             Value = value;
53             Previous = this;
54             Next = this;
55         }
56
57         private CircleEntry ()
58         {
59             Previous = this;
60             Next = this;
61         }
62
63         public CircleEntry (IEnumerable<T> list)
64         {
65             bool first = true;
66             CircleEntry<T> inserted = this;
67             foreach (T obj in list) {
68                 if (first) {
69                     Value = obj;
70                     Previous = this;
71                     Next = this;
72                 }
73                 else {
74                     inserted = inserted.InsertAfter (obj);
75                 }
76                 first = false;
77             }
78         }
79
80         public static CircleEntry<T> Empty
81         {
82             get {
83                 return new CircleEntry<T> ();
84             }
85         }
86
87         public CircleEntry<T> InsertBefore (T obj)
88         {
89             CircleEntry<T> insert = new CircleEntry<T> (obj);
90             insert.Previous = this.Previous;
91             insert.Next = this;
92             this.Previous.Next = insert;
93             this.Previous = insert;
94             return insert;
95         }
96
97         public CircleEntry<T> InsertAfter (T obj)
98         {
```

```
99         //Log.Debug (this, ".InsertAfter (", obj, ")");
100         CircleEntry<T> insert = new CircleEntry<T> (obj);
101         insert.Next = this.Next;
102         insert.Previous = this;
103         this.Next.Previous = insert;
104         this.Next = insert;
105         return insert;
106     }
107
108     public void Remove ()
109     {
110         Previous.Next = Next;
111         Next.Previous = Previous;
112         Previous = null;
113         Next = null;
114     }
115
116     private bool IsEmpty
117     {
118         get {
119             return (Next == this || Next == null) && (Previous == this ||
120                 Previous == null);
121         }
122     }
123
124     public int Count
125     {
126         get {
127             if (IsEmpty) {
128                 return 0;
129             }
130             else {
131                 CircleEntry<T> current = this;
132                 int count = 0;
133                 do {
134                     ++count;
135                     current = current.Next;
136                 } while (current != this);
137                 return count;
138             }
139         }
140     }
141
142     public bool Contains (T obj, out IEnumerable<CircleEntry<T>> item)
143     {
144         item = Find (obj);
145         return item.Count () > 0;
146     }
147
148     public bool Contains (Func<T, bool> func, out IEnumerable<CircleEntry<
149         T>> item)
150     {
151         item = Find (func);
152         return item.Count () > 0;
153     }
154
155     public bool Contains (T obj, out CircleEntry<T> item)
156     {
157         item = Find (obj).ElementAtOrDefault (0);
158         return item != null;
159     }
```

```
160     public bool Contains (Func<T, bool> func, out CircleEntry<T> item)
161     {
162         item = Find (func).ElementAtOrDefault (0);
163         return item != null;
164     }
165
166     public IEnumerable<CircleEntry<T>> Find (T obj)
167     {
168         return Find ((t) => t.Equals (obj));
169     }
170
171     public IEnumerable<CircleEntry<T>> Find (Func<T, bool> func)
172     {
173         CircleEntry<T> current = this;
174         do {
175             if (func (current.Value)) {
176                 yield return current;
177             }
178             current = current.Next;
179         }
180         while (current != this);
181         yield break;
182     }
183
184     public int IndexOf (T obj)
185     {
186         return IndexOf ((t) => t.Equals (obj));
187     }
188
189     public int IndexOf (Func<T, bool> func)
190     {
191         int i = 0;
192         CircleEntry<T> current = this;
193         do {
194             if (func (current.Value)) {
195                 return i;
196             }
197             current = current.Next;
198             ++ i;
199         }
200         while (current != this);
201         return -1;
202     }
203
204     public IEnumerable<T> RangeTo (CircleEntry<T> other)
205     {
206         CircleEntry<T> current = this;
207         do {
208             yield return current.Value;
209             current = current.Next;
210         }
211         while (current != other.Next && current != this);
212     }
213
214     public IEnumerable<T> WayTo (T other)
215     {
216         CircleEntry<T> current = this;
217         while (!current.Value.Equals (other)) {
218             yield return current.Value;
219             current = current.Next;
220             if (current == this) {
```

```
221             break;
222         }
223     }
224 }
225
226 public IEnumerable<Tuple<T,T>> Pairs
227 {
228     get {
229         CircleEntry<T> current = this;
230         do {
231             yield return Tuple.Create (current.Value, current.Next.Value);
232             current = current.Next;
233         }
234         while (current != this);
235     }
236 }
237
238 public IEnumerable<Tuple<T,T,T>> Triples
239 {
240     get {
241         CircleEntry<T> current = this;
242         do {
243             yield return Tuple.Create (current.Previous.Value, current.Value, current.Next.Value);
244             current = current.Next;
245         }
246         while (current != this);
247     }
248 }
249
250 public IEnumerator<T> GetEnumerator ()
251 {
252     CircleEntry<T> current = this;
253     do {
254         //Log.Debug (this, " => ", current.Content);
255         yield return current.Value;
256         current = current.Next;
257     }
258     while (current != this);
259 }
260
261 // explicit interface implementation for nongeneric interface
262 IEnumerator IEnumerable.GetEnumerator ()
263 {
264     return GetEnumerator (); // just return the generic version
265 }
266
267 [ExcludeFromCodeCoverageAttribute]
268 public override string ToString ()
269 {
270     if (IsEmpty) {
271         return "CircleEntry (" + Value.ToString () + ")";
272     }
273     else {
274         return "CircleEntry.Empty";
275     }
276 }
277
278 public static CircleEntry<T> operator + (CircleEntry<T> circle, int i)
279 {
280     CircleEntry<T> next = circle;
281     while (i > 0) {
```

```
282         next = next.Next;
283         i--;
284     }
285     while (i < 0) {
286         next = next.Previous;
287         i++;
288     }
289     return next;
290 }
291
292 public T this [int index]
293 {
294     get {
295         return (this + index).Value;
296     }
297     set {
298         (this + index).Value = value;
299     }
300 }
301
302 public static CircleEntry<T> operator - (CircleEntry<T> circle, int i)
303 {
304     return circle + (-i);
305 }
306
307 public static CircleEntry<T> operator ++ (CircleEntry<T> circle)
308 {
309     return circle.Next;
310 }
311
312 public static CircleEntry<T> operator -- (CircleEntry<T> circle)
313 {
314     return circle.Previous;
315 }
316
317 public static implicit operator T (CircleEntry<T> circle)
318 {
319     return circle.Value;
320 }
321
322 public bool IsReadOnly { get { return false; } }
323
324 public bool Contains (T obj)
325 {
326     CircleEntry<T> item = Find (obj).ElementAtOrDefault (0);
327     return item != null;
328 }
329
330 public bool Remove (T value)
331 {
332     CircleEntry<T> item;
333     if (Contains (value, out item)) {
334         item.Remove ();
335         return true;
336     }
337     else {
338         return false;
339     }
340 }
341
342 public void RemoveAt (int i)
```



```
343     {
344         (this + i).Remove ();
345     }
346
347     public void Insert (int i, T value)
348     {
349         (this + i).InsertBefore (value);
350     }
351
352     public void Add (T value)
353     {
354         if (Value == null) {
355             Value = value;
356         }
357         else {
358             InsertBefore (value);
359         }
360     }
361
362     public void Clear ()
363     {
364         Remove ();
365         Next = Previous = this;
366     }
367
368     public void CopyTo (T[] array, int start)
369     {
370         foreach (T value in this) {
371             array.SetValue (value, start);
372             ++start;
373         }
374     }
375 }
376
377 public static class CircleExtensions
378 {
379     public static CircleEntry<T> ToCircle<T> (this IEnumerable<T> enumerab
1 380     {
1 381         return new CircleEntry<T> (enumerable);
1 382     }
383 }
384 }
```

Knot3.Game.Data.Direction

Summary

Class:	Knot3.Game.Data.Direction
Assembly:	Knot3
File(s):	c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Direction.cs
Coverage:	100%
Covered lines:	72
Uncovered lines:	0
Coverable lines:	72
Total lines:	217

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
FromAxis(...)	5	100	85.71
FromString(...)	3	100	100
op_Addition(...)	1	100	100
op_Subtraction(...)	1	100	100
op_Division(...)	1	100	100
op_Multiply(...)	1	100	100
op_Equality(...)	4	100	85.71
op_Inequality(...)	1	100	100
Equals(...)	2	100	66.67
Equals(...)	5	100	100
op_Implicit(...)	1	100	100
op_Implicit(...)	1	100	100
.cctor()	1	100	100

File(s)

c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Direction.cs

```

#   Line Coverage
    1  /*
    2  * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3  *                               Gerd Augsburg, Christina Erler, Daniel Warzel
    4  *
    5  * This source code file is part of Knot3. Copying, redistribution and
    6  * use of the source code in this file in source and binary forms,
    7  * with or without modification, are permitted provided that the conditions
    8  * of the MIT license are met:
    9  *
   10  * Permission is hereby granted, free of charge, to any person obtaining a c
   11  * of this software and associated documentation files (the "Software"), to
   12  * in the Software without restriction, including without limitation the rig
   13  * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14  * copies of the Software, and to permit persons to whom the Software is
   15  * furnished to do so, subject to the following conditions:
   16  *
   17  * The above copyright notice and this permission notice shall be included i
   18  * copies or substantial portions of the Software.
   19  *
   20  * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21  * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22  * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
   23  * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER

```

```

24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Collections.Generic;
33 using System.Diagnostics.CodeAnalysis;
34
35 using Microsoft.Xna.Framework;
36
37 namespace Knot3.Game.Data
38 {
39     /// <summary>
40     /// Eine Wertesammlung der mglichen Richtungen in einem dreidimensionalen
41     /// Wird benutzt, damit keine ungtigen Kantenrichtungen angegeben werden
42     /// Dies ist eine Klasse und kein Enum, kann aber
43     /// uneingeschrnkt wie eines benutzt werden (Typesafe Enum Pattern).
44     /// </summary>
45     public sealed class Direction : IEquatable<Direction>
46     {
47         /// <summary>
48         /// Links.
49         /// </summary>
50         public static readonly Direction Left = new Direction (Vector3.Left, "
51         /// <summary>
52         /// Rechts.
53         /// </summary>
54         public static readonly Direction Right = new Direction (Vector3.Right,
55         /// <summary>
56         /// Hoch.
57         /// </summary>
58         public static readonly Direction Up = new Direction (Vector3.Up, "Up")
59         /// <summary>
60         /// Runter.
61         /// </summary>
62         public static readonly Direction Down = new Direction (Vector3.Down, "
63         /// <summary>
64         /// Vorwrts.
65         /// </summary>
66         public static readonly Direction Forward = new Direction (Vector3.Forw
67         /// <summary>
68         /// Rckwrts.
69         /// </summary>
70         public static readonly Direction Backward = new Direction (Vector3.Bac
71         /// <summary>
72         /// Keine Richtung.
73         /// </summary>
74         public static readonly Direction Zero = new Direction (Vector3.Zero, "
75
76         public static readonly Direction[] Values = {
77             Left, Right, Up, Down, Forward, Backward
78         };
79         private static readonly Dictionary<Direction, Direction> ReverseMap
80             = new Dictionary<Direction, Direction> ()
81         {
82             { Left, Right }, { Right, Left },
83             { Up, Down }, { Down, Up },
84             { Forward, Backward }, { Backward, Forward },

```

```

85         { Zero, Zero }
86     };
87
1   88     private static readonly Dictionary<Direction, Axis> AxisMap
89         = new Dictionary<Direction, Axis> ()
90     {
91         { Left, Axis.X }, { Right, Axis.X },
92         { Up, Axis.Y }, { Down, Axis.Y },
93         { Forward, Axis.Z }, { Backward, Axis.Z },
94         { Zero, Axis.Zero }
95     };
96
2657 97     public Vector3 Vector { get; private set; }
98
375  99     public string Description { get; private set; }
100
183  101     public Direction Reverse { get { return ReverseMap [this]; } }
102
33   103     public Axis Axis { get { return AxisMap[this]; } }
104
7    105     private Direction (Vector3 vector, string description)
7    106     {
7    107         Vector = vector;
7    108         Description = description;
7    109     }
110
111     public static Direction FromAxis (Axis axis)
112     {
3    113         return axis == Axis.X ? Right : axis == Axis.Y ? Up : axis == Axis
3    114     }
115
116     public static Direction FromString (string str)
117     {
7    118         foreach (Direction direction in Values) {
96   119             if (str.ToLower () == direction.Description.ToLower ()) {
33   120                 return direction;
6    121             }
21   122         }
1    123         return null;
7    124     }
125
126     [ExcludeFromCodeCoverageAttribute]
127     public override string ToString ()
128     {
129         return Description;
130     }
131
132     public static Vector3 operator + (Vector3 v, Direction d)
415  133     {
415  134         return v + d.Vector;
415  135     }
136
137     public static Vector3 operator - (Vector3 v, Direction d)
1    138     {
1    139         return v - d.Vector;
1    140     }
141
142     public static Vector3 operator / (Direction d, float i)
796  143     {
796  144         return d.Vector / i;
796  145     }

```

	146	
	147	public static Vector3 operator * (Direction d, float i)
1	148	{
1	149	return d.Vector * i;
1	150	}
	151	
	152	public static bool operator == (Direction a, Direction b)
1098	153	{
	154	// If both are null, or both are same instance, return true.
1498	155	if (System.Object.ReferenceEquals (a, b)) {
400	156	return true;
	157	}
	158	
	159	// If one is null, but not both, return false.
776	160	if (((object)a == null) ((object)b == null)) {
78	161	return false;
	162	}
	163	
	164	// Return true if the fields match:
620	165	return a.Vector == b.Vector;
1098	166	}
	167	
	168	public static bool operator != (Direction d1, Direction d2)
110	169	{
110	170	return !(d1 == d2);
110	171	}
	172	
	173	public bool Equals (Direction other)
78	174	{
78	175	return other != null && Vector == other.Vector;
78	176	}
	177	
	178	public override bool Equals (object other)
7	179	{
8	180	if (other == null) {
1	181	return false;
	182	}
7	183	else if (other is Direction) {
1	184	return Equals (other as Direction);
	185	}
6	186	else if (other is Vector3) {
1	187	return Vector.Equals ((Vector3)other);
	188	}
6	189	else if (other is string) {
2	190	return Description.Equals ((string)other);
	191	}
2	192	else {
2	193	return false;
	194	}
7	195	}
	196	
	197	public static implicit operator string (Direction direction)
1	198	{
1	199	return direction.Description;
1	200	}
	201	
	202	public static implicit operator Vector3 (Direction direction)
23	203	{
23	204	return direction.Vector;
23	205	}
	206	

```
207      [ExcludeFromCodeCoverageAttribute]
208      public override int GetHashCode ()
209      {
210          return Description.GetHashCode ();
211      }
212  }
213
214      public enum Axis {
215          X, Y, Z, Zero
216      }
217  }
```

Knot3.Game.Data.Edge

Summary

Class:	Knot3.Game.Data.Edge
Assembly:	Knot3
File(s):	c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Edge.cs
Coverage:	100%
Covered lines:	80
Uncovered lines:	0
Coverable lines:	80
Total lines:	208

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
.ctor(...)	1	100	100
op_Equality(...)	4	100	85.71
op_Inequality(...)	1	100	100
Equals(...)	2	100	100
Equals(...)	6	100	100
op_Implicit(...)	1	100	100
op_Implicit(...)	1	100	100
op_Implicit(...)	1	100	100
RandomColor()	1	100	100
RandomColor(...)	1	100	100
RandomEdge()	6	100	18.18
Clone()	1	100	100
.cctor()	1	100	100

File(s)

c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Edge.cs

```

#   Line Coverage
    1  /*
    2  * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3  *                               Gerd Augsburg, Christina Erler, Daniel Warzel
    4  *
    5  * This source code file is part of Knot3. Copying, redistribution and
    6  * use of the source code in this file in source and binary forms,
    7  * with or without modification, are permitted provided that the conditions
    8  * of the MIT license are met:
    9  *
   10  *   Permission is hereby granted, free of charge, to any person obtaining a c
   11  *   of this software and associated documentation files (the "Software"), to
   12  *   in the Software without restriction, including without limitation the rig
   13  *   to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14  *   copies of the Software, and to permit persons to whom the Software is
   15  *   furnished to do so, subject to the following conditions:
   16  *
   17  *   The above copyright notice and this permission notice shall be included i
   18  *   copies or substantial portions of the Software.
   19  *
   20  *   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21  *   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22  *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
   23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER

```

```

24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Collections.Generic;
33 using System.Diagnostics.CodeAnalysis;
34
35 using Microsoft.Xna.Framework;
36
37 namespace Knot3.Game.Data
38 {
39     /// <summary>
40     /// Eine Kante eines Knotens, die aus einer Richtung und einer Farbe, sowi
41     /// </summary>
42     public sealed class Edge : IEquatable<Edge>, ICloneable
43     {
44         /// <summary>
45         /// Die Farbe der Kante.
46         /// </summary>
687     47         public Color Color { get; set; }
48
49         /// <summary>
50         /// Die Richtung der Kante.
51         /// </summary>
2122    52         public Direction Direction { get; private set; }
53
54         /// <summary>
55         /// Die Liste der Flchennummern, die an die Kante angrenzen.
56         /// </summary>
651    57         public HashSet<int> Rectangles { get; private set; }
58
59         private int id;
1    60         private static int previousId = 0;
61
62         /// <summary>
63         /// Erstellt eine neue Kante mit der angegebenen Richtung.
64         /// </summary>
282    65         public Edge (Direction direction)
282    66         {
282    67             Direction = direction;
282    68             Color = DefaultColor;
282    69             id = ++previousId;
282    70             Rectangles = new HashSet<int> ();
282    71         }
72
73         /// <summary>
74         /// Erstellt eine neue Kante mit der angegebenen Richtung und Farbe.
75         /// </summary>
19    76         public Edge (Direction direction, Color color)
19    77         {
19    78             Direction = direction;
19    79             Color = color;
19    80             id = ++previousId;
19    81             Rectangles = new HashSet<int>();
19    82         }
83
84         public static bool operator == (Edge a, Edge b)

```



```

76      85      {
      86          // If both are null, or both are same instance, return true.
78      87          if (System.Object.ReferenceEquals (a, b)) {
2      88              return true;
      89          }
      90
      91          // If one is null, but not both, return false.
145    92          if (((object)a == null) || ((object)b == null)) {
71    93              return false;
      94          }
      95
      96          // Return true if the fields match:
3      97          return a.id == b.id;
76    98      }
      99
      100      public static bool operator != (Edge a, Edge b)
74    101      {
74    102          return !(a == b);
74    103      }
      104
      105      public bool Equals (Edge other)
72    106      {
72    107          return other != null && this.id == other.id;
72    108      }
      109
      110      public override bool Equals (object other)
37    111      {
38    112          if (other == null) {
1    113              return false;
      114          }
65    115          else if (other is Edge) {
29    116              return Equals (other as Edge);
      117          }
8    118          else if (other is Direction) {
1    119              return Direction.Equals (other as Direction);
      120          }
7    121          else if (other is Vector3) {
1    122              return Direction.Vector.Equals ((Vector3)other);
      123          }
6    124          else if (other is Color) {
1    125              return Color.Equals ((Color)other);
      126          }
4    127          else {
4    128              return false;
      129          }
37    130      }
      131
      132      [ExcludeFromCodeCoverageAttribute]
      133      public override int GetHashCode ()
      134      {
      135          return id;
      136      }
      137
      138      [ExcludeFromCodeCoverageAttribute]
      139      public override string ToString ()
      140      {
      141          return Direction + "/" + id.ToString ();
      142      }
      143
      144      public static implicit operator Direction (Edge edge)
9    145      {

```

```

9 146         return edge.Direction;
9 147     }
148
149     public static implicit operator Vector3 (Edge edge)
7 150     {
7 151         return edge.Direction;
7 152     }
153
154     public static implicit operator Color (Edge edge)
18 155     {
18 156         return edge.Color;
18 157     }
158
159     private static Random r = new Random ();
160
161     public static Color RandomColor ()
1 162     {
1 163         return Colors [r.Next () % Colors.Count];
1 164     }
165
166     public static Color RandomColor (GameTime time)
1 167     {
1 168         return Colors [(int)time.TotalGameTime.TotalSeconds % Colors.Count
1 169     }
170
171     public static Edge RandomEdge ()
1 172     {
1 173         int i = r.Next () % 6;
1 174         return i == 0 ? Left : i == 1 ? Right : i == 2 ? Up : i == 3 ? Dow
1 175     }
176
177     public object Clone ()
1 178     {
1 179         return new Edge (Direction, Color);
1 180     }
181
182     public static List<Color> Colors = new List<Color> ()
183     {
184         Color.Red, Color.Green, Color.Blue, Color.Yellow, Color.Orange
185     };
1 186     public static Color DefaultColor = RandomColor ();
187
42 188     public static Edge Zero { get { return new Edge (Direction.Zero); } }
189
3 190     public static Edge UnitX { get { return new Edge (Direction.Right); } }
191
3 192     public static Edge UnitY { get { return new Edge (Direction.Up); } }
193
3 194     public static Edge UnitZ { get { return new Edge (Direction.Backward);
195
186 196     public static Edge Up { get { return new Edge (Direction.Up); } }
197
171 198     public static Edge Down { get { return new Edge (Direction.Down); } }
199
183 200     public static Edge Right { get { return new Edge (Direction.Right); } }
201
186 202     public static Edge Left { get { return new Edge (Direction.Left); } }
203
33 204     public static Edge Forward { get { return new Edge (Direction.Forward)
205
36 206     public static Edge Backward { get { return new Edge (Direction.Backwar

```

207 }

208 }

Knot3.Game.Data.Knot

Summary

Class: Knot3.Game.Data.Knot
Assembly: Knot3
File(s): c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Knot.cs
Coverage: 84.8%
Covered lines: 263
Uncovered lines: 47
Coverable lines: 310
Total lines: 614

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor()	5	100	55.56
.ctor(...)	8	100	86.67
.ctor(...)	5	100	88.89
IsValidStructure(...	6	83.33	72.73
IsValidStructure(...	2	100	100
TryMove(...)	21	87.30	87.80
MoveCenterToZero()	8	100	86.67
Move(...)	2	100	100
IsValidDirection(...	16	81.25	77.42
onEdgesChanged()	1	0	0
GetEnumerator()	1	100	100
Save()	3	0	0
Clone()	2	100	100
OnSelectionChanged()	1	100	100
AddToSelection(...)	3	100	60
RemoveFromSelection(2	100	100
ClearSelection()	1	100	100
AddRangeToSelection(9	100	88.24
IsSelected(...)	1	0	0
System.Collections.I	1	100	100
Save(...)	1	0	0
Equals(...)	8	51.43	46.67
Charakteristic()	9	100	100
.ctor(...)	1	100	100

File(s)

c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Knot.cs

```

#   Line  Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *   Permission is hereby granted, free of charge, to any person obtaining a c
11  *   of this software and associated documentation files (the "Software"), to
12  *   in the Software without restriction, including without limitation the rig
13  *   to use, copy, modify, merge, publish, distribute, sublicense, and/or sell

```

```

14  *   copies of the Software, and to permit persons to whom the Software is
15  *   furnished to do so, subject to the following conditions:
16  *
17  *   The above copyright notice and this permission notice shall be included i
18  *   copies or substantial portions of the Software.
19  *
20  *   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31  using System;
32  using System.Collections;
33  using System.Collections.Generic;
34  using System.Diagnostics.CodeAnalysis;
35  using System.IO;
36  using System.Linq;
37
38  using Microsoft.Xna.Framework;
39
40  using Knot3.Framework.Platform;
41  using Knot3.Framework.Utilities;
42
43  namespace Knot3.Game.Data
44  {
45      /// <summary>
46      /// Diese Klasse repräsentiert einen Knoten, bestehend aus einem Knoten-Me
47      /// </summary>
48      public sealed class Knot : ICloneable, IEnumerable<Edge>, IEquatable<Knot>
49      {
50          /// <summary>
51          /// Der Name des Knotens, welcher auch leer sein kann.
52          /// Beim Speichern muss der Nutzer in diesem Fall zwingend einen nicht
53          /// Der Wert dieser Eigenschaft wird aus der \glqq Name\grqq -Eigensch
54          /// und bei Änderungen wieder in diesem gespeichert.
55          /// Beim Ändern dieser Eigenschaft wird automatisch auch der im Metada
56          /// </summary>
57          public string Name
58          {
45      59              get { return MetaData.Name; }
3      60              set { MetaData.Name = value; }
61          }
62
63          /// <summary>
64          /// Das Startelement der doppelt-verketteten Liste, in der die Kanten
65          /// </summary>
66          private CircleEntry<Edge> startElement;
67
68          /// <summary>
69          /// Die Metadaten des Knotens.
70          /// </summary>
61      71          public KnotMetaData MetaData { get; private set; }
72
73          /// <summary>
74          /// Ein Ereignis, das in der Move-Methode ausgelöst wird, wenn sich di

```

```

75      /// </summary>
33 76      public Action EdgesChanged = () => {};
77
78      /// <summary>
79      /// Enthlt die aktuell vom Spieler selektierten Kanten in der Reihenfolge
80      /// </summary>
24 81      public IEnumerable<Edge> SelectedEdges { get { return selectedEdges; } }
82
83      /// <summary>
84      /// Enthlt die selektierten Kanten.
85      /// </summary>
86      private HashSet<Edge> selectedEdges;
87
88      /// <summary>
89      /// WTF?!
90      /// </summary>
91      public int debugId;
92
93      /// <summary>
94      /// Wird aufgerufen, wenn sich die Selektion gendert hat.
95      /// </summary>
44 96      public Action SelectionChanged = () => {};
97
98      /// <summary>
99      /// Enthlt die zuletzt selektierte Kante.
100     /// </summary>
101     private CircleEntry<Edge> lastSelected;
102
103     /// <summary>
104     /// Wird aufgerufen, wenn sich die Startkante gendert hat.
105     /// </summary>
33 106     public Action<Vector3> StartEdgeChanged = (v) => {};
107
108     /// <summary>
109     /// Der Cache fr die Knotencharakteristik.
110     /// </summary>
33 111     private KnotCharakteristic? CharakteristicCache = null;
112
40 113     public Vector3 OffSet { get; private set;}
114
115     /// <summary>
116     /// Erstellt einen minimalen Standardknoten. Das Metadaten-Objekt enth
117     /// lt die das Speicherformat und den Dateinamen beinhalten, den Wert \gl
118     /// </summary>
1 119     public Knot ()
1 120     {
1 121         debugId++;
1 122         MetaData = new KnotMetaData (String.Empty, () => startElement.Count)
1 123         {
124             startElement = new CircleEntry<Edge> (new Edge[] {
125                 // Edge.Up, Edge.Right, Edge.Right, Edge.Down, Edge.Backward,
126                 // Edge.Up, Edge.Left, Edge.Left, Edge.Down, Edge.Forward
127                 Edge.Up, Edge.Right, Edge.Down, Edge.Left
128             });
129         selectedEdges = new HashSet<Edge> ();
130         OffSet = Vector3.Zero;
131     }
132
133     /// <summary>
134     /// Erstellt einen neuen Knoten mit dem angegebenen Metadaten-Objekt und
135     /// die in der doppelt verketteten Liste gespeichert werden.

```

```

136      /// Die Eigenschaft des Metadaten-Objektes, die die Anzahl der Kanten
137      /// wird auf ein Delegate gesetzt, welches jeweils die aktuelle Anzahl
138      /// </summary>
29 139      public Knot (KnotMetaData metaData, IEnumerable<Edge> edges)
29 140      {
29 141          debugId++;
29 142          Stack<Direction> structure = new Stack<Direction> ();
1035 143          foreach (Edge edge in edges) {
316 144              structure.Push (edge.Direction);
316 145          }
30 146          if (!IsValidStructure (structure)) {
1 147              throw new InvalidDataException ();
148          }
28 149          MetaData = new KnotMetaData (
150              name: metaData.Name,
1 151              countEdges: () => this.startElement.Count,
152              format: metaData.Format,
153              filename: metaData.Filename
154          );
28 155          this.startElement = new CircleEntry<Edge> (edges);
28 156          selectedEdges = new HashSet<Edge> ();
28 157          OffSet = Vector3.Zero;
28 158      }
159
3 160      private Knot (KnotMetaData metaData, CircleEntry<Edge> start, HashSet<
3 161      {
3 162          startElement = start;
3 163          MetaData = new KnotMetaData (
164              name: metaData.Name,
0 165              countEdges: () => this.startElement.Count,
166              format: metaData.Format,
167              filename: metaData.Filename
168          );
3 169          selectedEdges = selected;
3 170          OffSet = offset;
3 171      }
172
173      /// <summary>
174      /// Prft ob die gegeben Struktur einen gltigen Knoten darstellt.
175      /// </summary>
176      public bool IsValidStructure (IEnumerable<Direction> knot)
32 177      {
32 178          Vector3 position3D = Vector3.Zero;
32 179          HashSet<Vector3> occupancy = new HashSet<Vector3> ();
32 180          if (knot.Count () < 4) {
0 181              return false;
182          }
1104 183          foreach (Direction peek in knot) {
336 184              if (occupancy.Contains (position3D + (peek / 2))) {
0 185                  return false;
186              }
336 187              else {
336 188                  occupancy.Add (position3D + (peek / 2));
336 189                  position3D += peek;
336 190              }
336 191          }
33 192          if (position3D.DistanceTo (Vector3.Zero) > 0.00001f) {
1 193              return false;
194          }
31 195          return true;
32 196      }

```

```

197
198 private bool IsValidStructure (IEnumerable<Edge> edges)
3 199 {
43 200     return IsValidStructure (from e in edges select e.Direction);
3 201 }
202
203 /// <summary>
204 /// Verschiebt die aktuelle Kantenauswahl in die angegebene Richtung u
205 /// </summary>
206 public bool TryMove (Direction direction, int distance, out Knot newkn
9 207 {
12 208     if (direction == Direction.Zero || distance == 0) {
3 209         newknot = this;
3 210         return true;
211     }
212
213     Log.Debug ("TryMove: direction = ", direction, ", distance = ", di
38 214     Log.Debug ("Current Knot #", startElement.Count, " = ", string.Joi
215
6 216     HashSet<Edge> selected = new HashSet<Edge> (selectedEdges);
6 217     CircleEntry<Edge> newCircle = CircleEntry<Edge>.Empty;
218
114 219     foreach (Tuple<Edge, Edge, Edge> triple in startElement.Triples) {
32 220         Edge previousEdge = triple.Item1;
32 221         Edge currentEdge = triple.Item2;
32 222         Edge nextEdge = triple.Item3;
223
38 224         if (selectedEdges.Contains (currentEdge) && !selectedEdges.Con
15 225             distance.Repeat (i => newCircle.Add (new Edge (direction:
6 226         })
227
32 228         newCircle.Add (currentEdge);
229
38 230         if (selectedEdges.Contains (currentEdge) && !selectedEdges.Con
15 231             distance.Repeat (i => newCircle.Add (new Edge (direction:
6 232         })
32 233     }
234
56 235     Log.Debug ("New Knot #", newCircle.Count, " = ", string.Join ("", "
236
6 237     Vector3 localOffset = OffSet;
6 238     CircleEntry<Edge> current = newCircle;
45 239     do {
55 240         if (current [- 1].Direction == current [- 2].Direction.Reverse
241             // Selektierte nicht lschen
13 242         if (selected.Contains (current [- 1]) || selected.Contains
3 243             Log.Debug ("Error: Selektierte nicht lschen");
3 244             newknot = null;
3 245             return false;
246         }
7 247         if (newCircle == current - 1) {
0 248             localOffset += (current - 1).Value;
0 249             newCircle = current;
0 250         }
8 251         else if (newCircle == current - 2) {
1 252             localOffset += (current - 1).Value.Direction + (curren
1 253             newCircle = current;
1 254         }
7 255         (current - 2).Remove ();
7 256         (current - 1).Remove ();
7 257     }

```



```

42 258         ++ current;
42 259     }
42 260     while (current != newCircle);
    261
23 262     Log.Debug ("New Knot after Remove #", newCircle.Count, " = ", stri
    263
    3 264     if (!IsValidStructure (newCircle)) {
0 265         Log.Debug ("Error: newCircle ist keine valide Struktur");
0 266         newknot = null;
0 267         return false;
    268     }
    3 269     newknot = new Knot (MetaData, newCircle, selected, localOffset);
    3 270     return true;
    9 271 }
    272
    273 public Vector3 MoveCenterToZero ()
    1 274 {
    1 275     Vector3 position3D = Vector3.Zero;
    1 276     Dictionary<Vector3, Edge> occupancy = new Dictionary<Vector3, Edge>
21 277     foreach (Edge edge in startElement) {
    6 278         occupancy.Add (position3D + (edge.Direction / 2), edge);
    6 279         position3D += edge;
    6 280     }
    1 281     Vector3 mid = Vector3.Zero;
21 282     foreach (KeyValuePair<Vector3,Edge> pos in occupancy) {
    6 283         mid += pos.Key;
    6 284     }
    1 285     mid /= startElement.Count;
    1 286     float minDistance = mid.Length ();
    1 287     Edge newStart = startElement.Value;
21 288     foreach (KeyValuePair<Vector3,Edge> pos in occupancy) {
    6 289         float testDistance = pos.Key.DistanceTo (mid);
    8 290         if (testDistance < minDistance) {
    2 291             newStart = pos.Value;
    2 292             minDistance = testDistance;
    2 293         }
    6 294     }
    1 295     Vector3 offset = Vector3.Zero;
    6 296     foreach (Edge edge in startElement.WayTo (newStart)) {
    1 297         offset += edge;
    1 298     }
    1 299     startElement.Contains (newStart, out startElement);
    1 300     offset += OffSet;
    1 301     OffSet = Vector3.Zero;
    1 302     return offset;
    1 303 }
    304
    305 /// <summary>
    306 /// Verschiebt die aktuelle Kantenauswahl in die angegebene Richtung u
    307 /// </summary>
    308 public bool Move (Direction direction, int distance)
    9 309 {
    310     Knot newKnot;
15 311     if (TryMove (direction, distance, out newKnot)) {
    6 312         startElement = newKnot.startElement;
    6 313         selectedEdges = newKnot.selectedEdges;
    6 314         return true;
    315     }
    3 316     else {
    3 317         return false;
    318     }

```

```

9 319     }
      320
      321     ///

```

```

380      /// oder beide den Wert \glqq null\grqq, dann wird eine IOException ge
381      /// </summary>
382      public void Save ()
383      {
384          if (MetaData.Format == null) {
385              throw new IOException ("Error: Knot: MetaData.Format is null!")
386          }
387          else if (MetaData.Filename == null) {
388              throw new IOException ("Error: Knot: MetaData.Filename is null
389          }
390          else {
391              MetaData.Format.Save (this);
392          }
393      }
394
395      /// <summary>
396      /// Erstellt eine vollstndige Kopie des Knotens, inklusive der Kanten
397      /// </summary>
398      public object Clone ()
399      {
400          CircleEntry<Edge> newCircle = new CircleEntry<Edge> (startElement
401          KnotMetaData metaData = new KnotMetaData (
402              name: MetaData.Name,
403              countEdges: () => 0,
404              format: MetaData.Format,
405              filename: MetaData.Filename
406          );
407          return new Knot (metaData: metaData, edges: newCircle) {
408              selectedEdges = new HashSet<Edge> (selectedEdges),
409              EdgesChanged = null,
410              SelectionChanged = null,
411          };
412      }
413
414      private void OnSelectionChanged ()
415      {
416          SelectionChanged ();
417      }
418
419      /// <summary>
420      /// Fgt die angegebene Kante zur aktuellen Kantenauswahl hinzu.
421      /// </summary>
422      public void AddToSelection (Edge edge)
423      {
424          IEnumerable<CircleEntry<Edge>> found = startElement.Find (edge);
425          if (found.Any ()) {
426              if (!selectedEdges.Contains (edge)) {
427                  selectedEdges.Add (edge);
428              }
429              lastSelected = found.ElementAt (0);
430          }
431          OnSelectionChanged ();
432      }
433
434      /// <summary>
435      /// Entfernt die angegebene Kante von der aktuellen Kantenauswahl.
436      /// </summary>
437      public void RemoveFromSelection (Edge edge)
438      {
439          selectedEdges.Remove (edge);
440          if (lastSelected.Value == edge) {

```

```

1 441         lastSelected = null;
1 442     }
2 443     OnSelectionChanged ();
2 444 }
445
446     /// <summary>
447     /// Hebt die aktuelle Kantenauswahl auf.
448     /// </summary>
449     public void ClearSelection ()
2 450     {
2 451         selectedEdges.Clear ();
2 452         lastSelected = null;
2 453         OnSelectionChanged ();
2 454     }
455
456     /// <summary>
457     /// Fgt alle Kanten auf dem krzesten Weg zwischen der zuletzt ausgew
458     /// zur aktuellen Kantenauswahl hinzu. Sind beide Wege gleich lang,
459     /// wird der Weg in Richtung der ersten Kante ausgewhlt.
460     /// </summary>
461     public void AddRangeToSelection (Edge selectedEdge)
3 462     {
4 463         if (lastSelected == null) {
1 464             AddToSelection (selectedEdge);
1 465             return;
466         }
2 467         CircleEntry<Edge> selectedCircle = null;
4 468         if (startElement.Contains (selectedEdge, out selectedCircle) && se
2 469             List<Edge> forward = new List<Edge> (lastSelected.RangeTo (sel
2 470             List<Edge> backward = new List<Edge> (selectedCircle.RangeTo (
471
472             if (forward.Count < backward.Count) {
12 473                 foreach (Edge e in forward) {
5 474                     if (!selectedEdges.Contains (e)) {
2 475                         selectedEdges.Add (e);
2 476                     }
3 477                 }
1 478             }
1 479             else {
9 480                 foreach (Edge e in backward) {
3 481                     if (!selectedEdges.Contains (e)) {
1 482                         selectedEdges.Add (e);
1 483                     }
2 484                 }
1 485             }
2 486             lastSelected = selectedCircle;
2 487         }
2 488         OnSelectionChanged ();
3 489     }
490
491     /// <summary>
492     /// Prft, ob die angegebene Kante in der aktuellen Kantenauswahl enth
493     /// </summary>
494     public Boolean IsSelected (Edge edge)
0 495     {
0 496         return selectedEdges.Contains (edge);
0 497     }
498
499     /// <summary>
500     /// Gibt die doppelt-verkettete Kantenliste als Enumerator zurck.
501     /// [name=IEnumerable.GetEnumerator]

```

```

502      /// [keywords= ]
503      /// </summary>
504      IEnumerator IEnumerable.GetEnumerator ()
2 505      {
2 506          return GetEnumerator (); // just return the generic version
2 507      }
508
509      /// <summary>
510      /// Speichert den Knoten unter dem angegebenen Dateinamen in dem angeg
511      /// </summary>
512      public void Save (IKnotIO format, string filename)
0 513      {
0 514          KnotMetaData metaData = new KnotMetaData (MetaData.Name, () => Met
0 515          Knot knotToSave = new Knot (metaData, startElement);
0 516          format.Save (knotToSave);
0 517      }
518
519      /// <summary>
520      /// Prft, ob die rumliche Struktur identisch ist, unabngig von dem
521      /// [parameters=Knot other]
522      /// </summary>
523      public bool Equals (Knot other)
12 524      {
12 525          KnotCharakteristik thisCharakteristik = Charakteristic ();
12 526          KnotCharakteristik otherCharakteristik = other.Charakteristic ();
18 527          if (thisCharakteristik.CountEdges != otherCharakteristik.CountEdge
6 528              return false;
529      }
530      // Bei Struktur im gleicher Richtung
12 531      if (thisCharakteristik.CharacteristicalEdge.Value.Direction == oth
6 532          CircleEntry<Edge> currentThisElement = thisCharakteristik.Char
6 533          CircleEntry<Edge> currentOtherElement = otherCharakteristik.Ch
66 534          while (currentThisElement != thisCharakteristik.Characteristic
30 535              if (currentThisElement.Value.Direction != currentOtherElem
0 536                  return false;
537              }
30 538              currentThisElement++;
30 539              currentOtherElement++;
30 540          }
6 541          return true;
542      }
543      // Bei Struktur in entgegengesetzter Richtung
0 544      else if (thisCharakteristik.CharacteristicalEdge.Value.Direction =
0 545          CircleEntry<Edge> currentThisElement = thisCharakteristik.Char
0 546          CircleEntry<Edge> currentOtherElement = otherCharakteristik.Ch
0 547          while (currentThisElement != thisCharakteristik.Characteristic
0 548              if (currentThisElement.Value.Direction != currentOtherElem
0 549                  return false;
550              }
0 551              currentThisElement++;
0 552              currentOtherElement++;
0 553          }
0 554          return true;
555      }
0 556      else {
0 557          return false;
558      }
12 559      }
560
561      /// <summary>
562      /// Gibt chrakteristische Werte zurck, die bei gleichen Knoten gleich

```

```

563      /// Einmal als Key ein eindeutiges Circle\<Edge\> Element und als Valu
564      /// einen Charakteristischen Integer. Momentan die Anzahl der Kanten.
565      /// </summary>
566      private KnotCharakteristic Charakteristic ()
24 567      {
35 568          if (CharakteristicCache.HasValue) {
11 569              return CharakteristicCache.Value;
570          }
571
13 572          CircleEntry<Edge> charakteristikElement = startElement;
13 573          Vector3 position3D = startElement.Value.Direction;
13 574          Vector3 bestPosition3D = startElement.Value.Direction / 2;
13 575          CircleEntry<Edge> edgePointer = startElement.Next;
576
13 577          int edgeCount = 1;
257 578          for (edgeCount = 1; edgePointer != startElement; edgePointer++, ed
77 579              Vector3 nextPosition3D = position3D + edgePointer.Value.Direct
77 580              if ((nextPosition3D.X < bestPosition3D.X)
581                  || (nextPosition3D.X == bestPosition3D.X && nextPositi
27 582                  || (nextPosition3D.X == bestPosition3D.X && nextPositi
27 583                  bestPosition3D = position3D + edgePointer.Value.Direction
27 584                  charakteristikElement = edgePointer;
27 585              }
77 586              position3D += edgePointer.Value.Direction;
77 587          }
588
13 589          CharakteristicCache = new KnotCharakteristic (charakteristikElemen
13 590          return CharakteristicCache.Value;
24 591      }
592
593      [ExcludeFromCodeCoverageAttribute]
594      public override string ToString ()
595      {
596          return "Knot (name=" + Name + ",#edgecount=" + startElement.Count.
597              + ",format=" + (MetaData.Format != null ? MetaData.ToString
598              + ")";
599      }
600
601      private struct KnotCharakteristic {
602          public CircleEntry<Edge> CharacteristicalEdge { get; private set;
603
604          public int CountEdges { get; private set; }
605
606          public KnotCharakteristic (CircleEntry<Edge> characteristicalEdge,
13 607              : this ()
13 608              {
13 609                  CharacteristicalEdge = characteristicalEdge;
13 610                  CountEdges = countEdges;
13 611              }
612          }
613      }
614  }

```

Knot3.Game.Data.KnotFileIO

Summary

Class:	Knot3.Game.Data.KnotFileIO
Assembly:	Knot3
File(s):	c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\KnotFileIO.cs
Coverage:	27.7%
Covered lines:	10
Uncovered lines:	26
Coverable lines:	36
Total lines:	124

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor()	1	100	100
Save(...)	2	0	0
Load(...)	2	0	0
LoadMetaData(...)	2	0	0
MoveNext()	5	100	80

File(s)

c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\KnotFileIO.cs

```

#   Line Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System.Collections.Generic;
32 using System.Diagnostics.CodeAnalysis;

```

```

33 using System.IO;
34
35 using Knot3.Framework.Platform;
36 using Knot3.Framework.Storage;
37
38 namespace Knot3.Game.Data
39 {
40     /// <summary>
41     /// Implementiert das Speicherformat fr Knoten.
42     /// </summary>
43     public sealed class KnotFileIO : IKnotIO
44     {
45         /// <summary>
46         /// Die fr eine Knoten-Datei gltigen Dateieindungen.
47         /// </summary>
86 48         public IEnumerable<string> FileExtensions
49         {
94 50             get {
94 51                 yield return ".knot";
10 52                 yield return ".knt";
10 53             }
54         }
55
44 56         private Dictionary<string, Knot> KnotCache = new Dictionary<string, Kn
44 57         private Dictionary<string, KnotMetaData> KnotMetaDataCache = new Dicti
58
59         /// <summary>
60         /// Erstellt ein KnotFileIO-Objekt.
61         /// </summary>
44 62         public KnotFileIO ()
44 63         {
44 64         }
65
66         /// <summary>
67         /// Speichert einen Knoten in dem Dateinamen, der in dem Knot-Objekt e
68         /// </summary>
69         public void Save (Knot knot)
0 70         {
0 71             KnotStringIO parser = new KnotStringIO (knot);
0 72             Log.Debug ("KnotFileIO.Save (", knot, ") = #", parser.Content.Leng
0 73             if (knot.MetaData.Filename == null) {
0 74                 throw new IOException ("Error! knot has no filename: " + knot)
75             }
0 76             else {
0 77                 File.WriteAllText (knot.MetaData.Filename, parser.Content);
0 78             }
0 79         }
80
81         /// <summary>
82         /// Ldt eines Knotens aus einer angegebenen Datei.
83         /// </summary>
84         public Knot Load (string filename)
0 85         {
0 86             if (KnotCache.ContainsKey (filename)) {
0 87                 return KnotCache [filename];
88             }
0 89             else {
0 90                 Log.Debug ("Load knot from ", filename);
0 91                 KnotStringIO parser = new KnotStringIO (content: string.Join (
0 92                 return KnotCache [filename] = new Knot (
0 93                     new KnotMetaData (parser.Name, () => parser.CountEdges, th

```



```

    94         parser.Edges
    95     );
    96 }
0  97 }
    98
    99     /// <summary>
   100     /// Ldt die Metadaten eines Knotens aus einer angegebenen Datei.
   101     /// </summary>
   102     public KnotMetaData LoadMetaData (string filename)
0  103     {
0  104         if (KnotMetaDataCache.ContainsKey (filename)) {
0  105             return KnotMetaDataCache [filename];
   106         }
0  107         else {
0  108             KnotStringIO parser = new KnotStringIO (content: string.Join (
0  109             return KnotMetaDataCache [filename] = new KnotMetaData (
   110                 name: parser.Name,
0  111                 countEdges: () => parser.CountEdges,
   112                 format: this,
   113                 filename: filename
   114             );
   115         }
0  116     }
   117
   118     [ExcludeFromCodeCoverageAttribute]
   119     public override string ToString ()
   120     {
   121         return "KnotFileIO";
   122     }
   123 }
124 }
```

Knot3.Game.Data.KnotMetaData

Summary

Class:	Knot3.Game.Data.KnotMetaData
Assembly:	Knot3
File(s):	\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\KnotMetaData.cs
Coverage:	64%
Covered lines:	32
Uncovered lines:	18
Coverable lines:	50
Total lines:	165

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	3	100	100
.ctor(...)	1	100	100
Equals(...)	3	0	0
Equals(...)	2	0	0
op_Equality(...)	4	0	0
op_Inequality(...)	1	0	0

File(s)

\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\KnotMetaData.cs

```

#   Line Coverage
    1  /*
    2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
    4   *
    5   * This source code file is part of Knot3. Copying, redistribution and
    6   * use of the source code in this file in source and binary forms,
    7   * with or without modification, are permitted provided that the conditions
    8   * of the MIT license are met:
    9   *
   10   * Permission is hereby granted, free of charge, to any person obtaining a c
   11   * of this software and associated documentation files (the "Software"), to
   12   * in the Software without restriction, including without limitation the rig
   13   * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14   * copies of the Software, and to permit persons to whom the Software is
   15   * furnished to do so, subject to the following conditions:
   16   *
   17   * The above copyright notice and this permission notice shall be included i
   18   * copies or substantial portions of the Software.
   19   *
   20   * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21   * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22   * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
   23   * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
   24   * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
   25   * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
   26   * SOFTWARE.
   27   *
   28   * See the LICENSE file for full license details of the Knot3 project.
   29   */
   30
   31 using System;
```

```

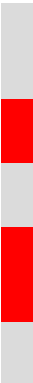
32 using System.Diagnostics.CodeAnalysis;
33 using System.Linq;
34
35 using Knot3.Framework.Platform;
36 using Knot3.Framework.Storage;
37
38 namespace Knot3.Game.Data
39 {
40     /// <summary>
41     /// Enthlt Metadaten eines Knotens, die aus einer Spielstand-Datei schnell
42     /// als der vollstndige Knoten. Dieses Objekt enthlt keine Datenstruktur
43     /// sondern nur Informationen ber den Namen des Knoten und die Anzahl sei
44     /// dazugehriges Knoten-Objekt existieren, aber jedes Knoten-Objekt enth
45     /// </summary>
46     public class KnotMetaData : IEquatable<KnotMetaData>
47     {
48         /// <summary>
49         /// Der Anzeigename des Knotens, welcher auch leer sein kann.
50         /// Beim Speichern muss der Spieler in diesem Fall zwingend
51         /// einen nichtleeren Namen whlen. Wird ein neuer Anzeigename festgel
52         /// dann wird der Dateiname ebenfalls auf einen neuen Wert gesetzt, un
53         /// ob er bereits einen Wert enthlt oder \glqq null\grqq ist.
54         /// Diese Eigenschaft kann ffentlich gelesen und gesetzt werden.
55         /// </summary>
56         public string Name
57         {
49 58             get {
49 59                 return name;
49 60             }
43 61             set {
43 62                 name = value;
85 63                 if (Format == null) {
42 64                     Format = new KnotFileIO ();
42 65                 }
85 66                 if (name != null && name.Length > 0) {
67                     string extension;
84 68                     if (Format.FileExtensions.Any ()) {
42 69                         extension = Format.FileExtensions.ElementAt (0);
42 70                     }
0 71                     else {
0 72                         throw new ArgumentException ("Every implementation of
73                     }
42 74                     Filename = SystemInfo.SavegameDirectory + SystemInfo.PathS
42 75                 }
43 76             }
77         }
78
79         private string name;
80
81         /// <summary>
82         /// Das Format, aus dem die Metadaten geladen wurden.
83         /// Es ist genau dann \glqq null\grqq, wenn die Metadaten nicht aus ei
84         /// </summary>
280 85         public IKnotIO Format { get; private set; }
86
87         /// <summary>
88         /// Ein Delegate, das die Anzahl der Kanten zurckliefert.
89         /// Falls dieses Metadaten-Objekt Teil eines Knotens ist, gibt es dyna
90         /// Kanten des Knoten-Objektes zurck. Anderenfalls gibt es eine stati
91         /// die beim Einlesen der Metadaten vor dem Erstellen dieses Objektes
92         /// </summary>

```

```

3    93    public int CountEdges { get { return countEdges (); } }
    94
    95    private Func<int> countEdges;
    96
    97    /// <summary>
    98    /// Falls die Metadaten aus einer Datei eingelesen wurden, enthl t die
    99    /// sonst \glqq null\grqq.
100    /// </summary>
152  101    public string Filename { get; private set; }
    102
    103    /// <summary>
    104    /// Erstellt ein neues Knoten-Metadaten-Objekt mit einem angegebenen K
    105    /// und einer angegebenen Funktion, welche eine Kantenanzahl zur ck gi
    106    /// Zust tzlich wird der Dateiname oder das Speicherformat angegeben, a
    107    /// </summary>
    42  108    public KnotMetaData (string name, Func<int> countEdges, IKnotIO format
    42  109    {
    42  110        Name = name;
    42  111        this.countEdges = countEdges;
    42  112        Format = format ?? Format;
    42  113        Filename = filename ?? Filename;
    42  114    }
    115
    116    /// <summary>
    117    /// Erstellt ein neues Knoten-Metadaten-Objekt mit einem angegebenen K
    118    /// und einer angegebenen Funktion, welche eine Kantenanzahl zur ck gi
    119    /// </summary>
    7   120    public KnotMetaData (string name, Func<int> countEdges)
    7   121    {
    7   122        this.name = name;
    7   123        this.countEdges = countEdges;
    7   124        Format = null;
    7   125        Filename = null;
    7   126    }
    127
    128    public bool Equals (KnotMetaData other)
    0   129    {
    0   130        return other != null && name == other.name && countEdges () == oth
    0   131    }
    132
    133    public override bool Equals (object other)
    0   134    {
    0   135        return other != null && Equals (other as KnotMetaData);
    0   136    }
    137
    138    [ExcludeFromCodeCoverageAttribute]
    139    public override int GetHashCode ()
    140    {
    141        return (countEdges ().ToString () + (name ?? String.Empty)).GetHas
    142    }
    143
    144    public static bool operator == (KnotMetaData a, KnotMetaData b)
    0   145    {
    146        // If both are null, or both are same instance, return true.
    0   147        if (System.Object.ReferenceEquals (a, b)) {
    0   148            return true;
    149        }
    150
    151        // If one is null, but not both, return false.
    0   152        if (((object)a == null) || ((object)b == null)) {
    0   153            return false;

```



```
154         }
155
156         // Return true if the fields match:
0 157         return a.Equals (b);
0 158     }
159
160     public static bool operator != (KnotMetaData a, KnotMetaData b)
0 161     {
0 162         return !(a == b);
0 163     }
164 }
165 }
```

Knot3.Game.Data.KnotStringIO

Summary

Class:	Knot3.Game.Data.KnotStringIO
Assembly:	Knot3
File(s):	\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\KnotStringIO.cs
Coverage:	73.1%
Covered lines:	90
Uncovered lines:	33
Coverable lines:	123
Total lines:	242

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
.ctor(...)	1	63.64	100
DecodeEdge(...)	8	100	100
EncodeEdge(...)	7	90.91	92.31
EncodeColor(...)	1	100	100
DecodeColor(...)	4	53.57	42.86
MoveNext()	8	72.73	60
MoveNext()	5	100	71.43

File(s)

\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\KnotStringIO.cs

```

#   Line Coverage
    1  /*
    2  * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3  *                               Gerd Augsburg, Christina Erler, Daniel Warzel
    4  *
    5  * This source code file is part of Knot3. Copying, redistribution and
    6  * use of the source code in this file in source and binary forms,
    7  * with or without modification, are permitted provided that the conditions
    8  * of the MIT license are met:
    9  *
   10  * Permission is hereby granted, free of charge, to any person obtaining a c
   11  * of this software and associated documentation files (the "Software"), to
   12  * in the Software without restriction, including without limitation the rig
   13  * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14  * copies of the Software, and to permit persons to whom the Software is
   15  * furnished to do so, subject to the following conditions:
   16  *
   17  * The above copyright notice and this permission notice shall be included i
   18  * copies or substantial portions of the Software.
   19  *
   20  * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21  * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22  * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
   23  * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
   24  * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
   25  * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
   26  * SOFTWARE.
   27  *
   28  * See the LICENSE file for full license details of the Knot3 project.
   29  */

```

```

30
31 using System;
32 using System.Collections.Generic;
33 using System.Diagnostics.CodeAnalysis;
34 using System.IO;
35 using System.Linq;
36
37 using Microsoft.Xna.Framework;
38
39 using Knot3.Framework.Platform;
40
41 namespace Knot3.Game.Data
42 {
43     /// <summary>
44     /// Diese Klasse repräsentiert einen Parser für das Knoten-Austauschformat
45     /// eingelesenen Informationen wie den Namen des Knotens und die Kantenlis
46     /// </summary>
47     public sealed class KnotStringIO
48     {
49         /// <summary>
50         /// Der Name der eingelesenen Knotendatei oder des zugewiesenen Knoten
51         /// </summary>
22 52         public string Name { get; set; }
53
54         private IEnumerable<string> edgeLines;
55
56         /// <summary>
57         /// Die Kanten der eingelesenen Knotendatei oder des zugewiesenen Knot
58         /// </summary>
5 59         public IEnumerable<Edge> Edges
60         {
61             get {
62                 Log.Debug ("KnotStringIO.Edges[get] = ", edgeLines.Count ());
67 63                 foreach (string _line in edgeLines) {
18 64                     string line = _line;
18 65                     Edge edge = DecodeEdge (line [0]);
16 66                     line = line.Substring (1);
32 67                     if (line.StartsWith("#")) {
16 68                         line = line.Substring (1);
16 69                     }
16 70                     edge.Color = DecodeColor (line.Substring (0, 8));
16 71                     line = line.Substring (8);
32 72                     if (line.StartsWith("#")) {
16 73                         line = line.Substring (1);
16 74                     }
16 75                     if (line.Length > 0) {
0 76                         foreach (int rect in line.Split (',' ).Select (int.Pars
0 77                             edge.Rectangles.Add (rect);
0 78                     }
0 79                 }
16 80                 yield return edge;
16 81             }
3 82         }
0 83         set {
0 84             Log.Debug ("KnotStringIO.Edges[set] = #", value.Count ());
0 85             try {
0 86                 edgeLines = ToLines (value);
0 87             }
0 88             catch (Exception ex) {
0 89                 Log.Debug (ex);
0 90             }

```

```

0    91      }
      92      }
      93
      94      ///

```



```

13 152 {
1089 153     foreach (Edge edge in edges) {
350 154         yield return EncodeEdge (edge) + "#" + EncodeColor (edge.Color
350 155     }
13 156 }
    157
    158 private static Edge DecodeEdge (char c)
    159 {
18 160     switch (c) {
18 161         case 'X':
2 162             return Edge.Right;
    163         case 'x':
2 164             return Edge.Left;
    165         case 'Y':
3 166             return Edge.Up;
    167         case 'y':
3 168             return Edge.Down;
    169         case 'Z':
3 170             return Edge.Backward;
    171         case 'z':
3 172             return Edge.Forward;
    173         default:
2 174             throw new IOException ("Failed to decode Edge: '" + c + "'!");
    175     }
16 176 }
    177
    178 private static char EncodeEdge (Edge edge)
    179 {
350 180     if (edge.Direction == Direction.Right) {
435 181         return 'X';
85 182     }
350 183     else if (edge.Direction == Direction.Left) {
85 184         return 'x';
    185     }
265 186     else if (edge.Direction == Direction.Up) {
85 187         return 'Y';
    188     }
180 189     else if (edge.Direction == Direction.Down) {
85 190         return 'y';
    191     }
15 192     else if (edge.Direction == Direction.Backward) {
5 193         return 'Z';
    194     }
10 195     else if (edge.Direction == Direction.Forward) {
5 196         return 'z';
    197     }
0 198     else {
0 199         throw new IOException ("Failed to encode Edge: '" + edge + "'!");
    200     }
350 201 }
    202
    203 private static String EncodeColor (Color c)
    204 {
350 205     return c.R.ToString ("X2") + c.G.ToString ("X2") + c.B.ToString ("
350 206 }
    207
    208 private static Color DecodeColor (string hexString)
    209 {
16 210     if (hexString.StartsWith("#")) {
16 211         hexString = hexString.Substring (1);
0 212     }
0

```

```

16 213         uint hex = unchecked ( uint.Parse (hexString, System.Globalization
16 214         Color color = Color.White;
32 215         if (hexString.Length == 8) {
16 216             unchecked {
16 217                 color.R = (byte)(hex >> 24);
16 218                 color.G = (byte)(hex >> 16);
16 219                 color.B = (byte)(hex >> 8);
16 220                 color.A = (byte)(hex);
16 221             }
16 222         }
0 223         else if (hexString.Length == 6) {
0 224             unchecked {
0 225                 color.R = (byte)(hex >> 16);
0 226                 color.G = (byte)(hex >> 8);
0 227                 color.B = (byte)(hex);
0 228             }
0 229         }
0 230         else {
0 231             throw new IOException ("Invalid hex representation of an ARGB o
232         }
16 233         return color;
16 234     }
235
236     [ExcludeFromCodeCoverageAttribute]
237     public override string ToString ()
238     {
239         return "KnotStringIO (length=" + Content.Length + ")";
240     }
241 }
242 }

```

Knot3.Game.Data.Node

Summary

Class:	Knot3.Game.Data.Node
Assembly:	Knot3
File(s):	c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Node.cs
Coverage:	73.7%
Covered lines:	45
Uncovered lines:	16
Coverable lines:	61
Total lines:	180

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
op_Implicit(...)	1	100	100
CenterBetween(...)	1	100	100
op_Addition(...)	1	100	100
op_Subtraction(...)	1	100	100
op_Addition(...)	1	0	0
op_Subtraction(...)	1	0	0
op_Addition(...)	1	0	0
op_Subtraction(...)	1	0	0
Clone()	1	100	100
op_Equality(...)	6	55.56	54.55
op_Inequality(...)	1	100	100
Equals(...)	3	100	60
Equals(...)	2	71.43	66.67
.cctor()	1	100	100

File(s)

c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\Node.cs

```

#   Line Coverage
    1  /*
    2  * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3  *                               Gerd Augsburg, Christina Erler, Daniel Warzel
    4  *
    5  * This source code file is part of Knot3. Copying, redistribution and
    6  * use of the source code in this file in source and binary forms,
    7  * with or without modification, are permitted provided that the conditions
    8  * of the MIT license are met:
    9  *
   10  * Permission is hereby granted, free of charge, to any person obtaining a c
   11  * of this software and associated documentation files (the "Software"), to
   12  * in the Software without restriction, including without limitation the rig
   13  * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14  * copies of the Software, and to permit persons to whom the Software is
   15  * furnished to do so, subject to the following conditions:
   16  *
   17  * The above copyright notice and this permission notice shall be included i
   18  * copies or substantial portions of the Software.
   19  *
   20  * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21  * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22  * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T

```

```

23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Diagnostics.CodeAnalysis;
33
34 using Microsoft.Xna.Framework;
35
36 namespace Knot3.Game.Data
37 {
38     /// <summary>
39     /// Eine Position im 3D-Raster. Die Werte fr alle drei Koordinaten sind I
40     /// Eine Skalierung auf Koordinaten im 3D-Raum und damit einhergehend eine
41     /// </summary>
42     public sealed class Node : IEquatable<Node>, ICloneable
43     {
44         /// <summary>
45         /// X steht fr eine x-Koordinate im dreidimensionalen Raster.
46         /// </summary>
116     47         public int X { get; private set; }
48
49         /// <summary>
50         /// Y steht fr eine y-Koordinate im dreidimensionalen Raster.
51         /// </summary>
116     52         public int Y { get; private set; }
53
54         /// <summary>
55         /// Z steht fr eine z-Koordinate im dreidimensionalen Raster.
56         /// </summary>
116     57         public int Z { get; private set; }
58
59         /// <summary>
60         /// Ein Skalierungswert.
61         /// </summary>
1     62         public static readonly int Scale = 100;
63
64         /// <summary>
65         /// Erzeugt eine neue Instanz eines Node-Objekts und initialisiert die
66         /// fr die x-, y- und z-Koordinate.
67         /// </summary>
22 68         public Node (int x, int y, int z)
22 69         {
22 70             X = x;
22 71             Y = y;
22 72             Z = z;
22 73         }
74
75         /// <summary>
76         /// Liefert die x-, y- und z-Koordinaten im 3D-Raum als ein Vektor3 de
77         /// </summary>
78         public Vector3 Vector
79         {
22 80             get {
22 81                 return new Vector3 (X * Scale, Y * Scale, Z * Scale);
22 82             }
83         }

```

```

84
85     public static implicit operator Vector3 (Node node)
14 86     {
14 87         return node.Vector;
14 88     }
89
90     public Vector3 CenterBetween (Node other)
4 91     {
4 92         Vector3 positionFrom = this.Vector;
4 93         Vector3 positionTo = other.Vector;
4 94         return positionFrom + (positionTo - positionFrom) / 2;
4 95     }
96
97     public static Node operator + (Node a, Vector3 b)
1 98     {
1 99         return new Node (a.X + (int)b.X, a.Y + (int)b.Y, a.Z + (int)b.Z);
1 100    }
101
102     public static Vector3 operator - (Node a, Node b)
1 103     {
1 104         return new Vector3 (a.X - b.X, a.Y - b.Y, a.Z - b.Z);
1 105     }
106
107     public static Node operator + (Node a, Direction b)
0 108     {
0 109         return new Node (a.X + (int)b.Vector.X, a.Y + (int)b.Vector.Y, a.Z
0 110     }
111
112     public static Node operator - (Node a, Direction b)
0 113     {
0 114         return new Node (a.X - (int)b.Vector.X, a.Y - (int)b.Vector.Y, a.Z
0 115     }
116
117     public static Node operator + (Direction a, Node b)
0 118     {
0 119         return b+a;
0 120     }
121
122     public static Node operator - (Direction a, Node b)
0 123     {
0 124         return b-a;
0 125     }
126
127     [ExcludeFromCodeCoverageAttribute]
128     public override int GetHashCode ()
129     {
130         return X * 10000 + Y * 100 + Z;
131     }
132
133     [ExcludeFromCodeCoverageAttribute]
134     public override string ToString ()
135     {
136         return "(" + X.ToString () + "," + Y.ToString () + "," + Z.ToStrin
137     }
138
139     public object Clone ()
1 140     {
1 141         return new Node (X, Y, Z);
1 142     }
143
144     public static bool operator == (Node a, Node b)

```

2	145	{
	146	// If both are null, or both are same instance, return true.
2	147	if (System.Object.ReferenceEquals (a, b)) {
0	148	return true;
	149	}
	150	
	151	// If one is null, but not both, return false.
2	152	if (((object)a == null) ((object)b == null)) {
0	153	return false;
	154	}
	155	
	156	// Return true if the fields match:
2	157	return a.X == b.X && a.Y == b.Y && a.Z == b.Z;
2	158	}
	159	
	160	public static bool operator != (Node a, Node b)
	161	{
1	162	return !(a == b);
1	163	}
	164	
	165	public bool Equals (Node other)
20	166	{
20	167	return this.X == other.X && this.Y == other.Y && this.Z == other.Z
20	168	}
	169	
	170	public override bool Equals (object obj)
2	171	{
4	172	if (obj is Node) {
2	173	return Equals ((Node)obj);
	174	}
0	175	else {
0	176	return false;
	177	}
2	178	}
	179	}
	180	}

Knot3.Game.Data.NodeMap

Summary

Class:	Knot3.Game.Data.NodeMap
Assembly:	Knot3
File(s):	c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\NodeMap.cs
Coverage:	88.5%
Covered lines:	54
Uncovered lines:	7
Coverable lines:	61
Total lines:	151

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor()	2	100	66.67
.ctor(...)	1	100	100
NodeBeforeEdge(...)	1	100	100
NodeAfterEdge(...)	1	100	100
JunctionsAtNode(...)	1	100	100
JunctionsBeforeEdge(1	100	100
JunctionsAfterEdge(.	1	100	100
OnEdgesChanged()	1	0	0
BuildIndex()	5	100	77.78

File(s)

c:\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\NodeMap.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
```

```

29  */
30
31  using System;
32  using System.Collections;
33  using System.Collections.Generic;
34  using System.Diagnostics.CodeAnalysis;
35  using System.Linq;
36
37  using Microsoft.Xna.Framework;
38
39  using Knot3.Framework.Utilities;
40
41  using Knot3.Game.Models;
42
43  namespace Knot3.Game.Data
44  {
45      /// <summary>
46      /// Eine Zuordnung zwischen Kanten und den dreidimensionalen Rasterpunkten
47      /// </summary>
48      public sealed class NodeMap : INodeMap
49      {
50          private Hashtable fromMap = new Hashtable ();
51          private Hashtable toMap = new Hashtable ();
52          private Dictionary<Node, List<IJunction>> junctionMap = new Dictionary
53
54          /// <summary>
55          /// Die Skalierung, die bei einer Konvertierung in einen Vector3 des X
56          /// </summary>
57          public int Scale { get; set; }
58
59          public IEnumerable<Edge> Edges { get; set; }
60
61          public Vector3 Offset { get; set; }
62
63          public Action IndexRebuilt { get; set; }
64
65          public NodeMap ()
66          {
67              IndexRebuilt = () => {};
68          }
69
70          public NodeMap (IEnumerable<Edge> edges)
71              : this ()
72          {
73              Edges = edges;
74              BuildIndex ();
75          }
76
77          /// <summary>
78          /// Gibt die Rasterposition des bergangs am Anfang der Kante zurck.
79          /// </summary>
80          public Node NodeBeforeEdge (Edge edge)
81          {
82              return (Node)fromMap [edge];
83          }
84
85          /// <summary>
86          /// Gibt die Rasterposition des bergangs am Ende der Kante zurck.
87          /// </summary>
88          public Node NodeAfterEdge (Edge edge)
89          {

```



```

20 90         return (Node)toMap [edge];
20 91     }
    92
    93     public List<IJunction> JunctionsAtNode (Node node)
    94     {
    95         return junctionMap [node];
    96     }
    97
    98     public List<IJunction> JunctionsBeforeEdge (Edge edge)
    99     {
   100         return junctionMap [NodeBeforeEdge (edge)];
   101     }
   102
   103     public List<IJunction> JunctionsAfterEdge (Edge edge)
   104     {
   105         return junctionMap [NodeAfterEdge (edge)];
   106     }
   107
   108     public IEnumerable<Node> Nodes
   109     {
   110         get {
   111             return junctionMap.Keys;
   112         }
   113     }
   114
   115     /// <summary>
   116     /// Aktualisiert die Zuordnung, wenn sich die Kanten gendert haben.
   117     /// </summary>
   118     public void OnEdgesChanged ()
   119     {
   120         BuildIndex ();
   121     }
   122
   123     private void BuildIndex ()
   124     {
   125         fromMap.Clear ();
   126         toMap.Clear ();
   127         float x = Offset.X, y = Offset.Y, z = Offset.Z;
   128         foreach (Edge edge in Edges) {
   129             fromMap [edge] = new Node ((int)x, (int)y, (int)z);
   130             Vector3 v = edge.Direction.Vector;
   131             x += v.X;
   132             y += v.Y;
   133             z += v.Z;
   134             toMap [edge] = new Node ((int)x, (int)y, (int)z);
   135         }
   136
   137         IndexRebuilt = () => {};
   138         junctionMap.Clear ();
   139         List<Edge> EdgeList = Edges.ToList ();
   140         for (int n = 0; n < EdgeList.Count; n++) {
   141             Edge edgeA = Edges.At (n);
   142             Edge edgeB = Edges.At (n + 1);
   143             Node node = NodeAfterEdge (edgeA);
   144             IJunction junction = new Junction (nodeMap: this, from: edgeA,
   145             junctionMap.Add (node, junction);
   146         }
   147
   148         IndexRebuilt ();
   149     }
   150 }

```

151 }

Knot3.Game.Data.RectangleMap

Summary

Class:	Knot3.Game.Data.RectangleMap
Assembly:	Knot3
File(s):	\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\RectangleMap.cs
Coverage:	0%
Covered lines:	0
Uncovered lines:	53
Coverable lines:	53
Total lines:	144

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	0	0
AddEdge(...)	1	0	0
AddEdge(...)	3	0	0
ContainsEdge(...)	7	0	0
MoveNext()	13	0	0

File(s)

\Users\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Data\RectangleMap.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System.Collections.Generic;
32 using System.Diagnostics.CodeAnalysis;

```

```

33 using System.Linq;
34
35 using Microsoft.Xna.Framework;
36
37 using Knot3.Framework.Utilities;
38
39 namespace Knot3.Game.Data
40 {
41     public sealed class RectangleMap
42     {
43         private INodeMap NodeMap;
44         private Dictionary<Vector3, List<PossibleRectanglePosition>> positions
45             = new Dictionary<Vector3, List<PossibleRectanglePosition>> ();
46
47         public RectangleMap (INodeMap nodeMap)
48         {
49             NodeMap = nodeMap;
50         }
51
52         public void AddEdge (Edge edge, bool isVirtual)
53         {
54             Node a = NodeMap.NodeBeforeEdge (edge);
55             Node b = NodeMap.NodeAfterEdge (edge);
56             AddEdge (edge, a, b, isVirtual);
57         }
58
59         public void AddEdge (Edge edge, Node nodeA, Node nodeB, bool isVirtual)
60         {
61             Vector3 edgeCenter = nodeA.CenterBetween (nodeB);
62             foreach (Direction direction in Direction.Values) {
63                 if (direction.Axis != edge.Direction.Axis) {
64                     Vector3 rectangleCenter = edgeCenter + direction * Node.Sc
65                     PossibleRectanglePosition rectanglePosition = new Possible
66                         Edge = edge,
67                         NodeA = nodeA,
68                         NodeB = nodeB,
69                         Position = rectangleCenter,
70                         IsVirtual = isVirtual
71                 };
72                 positions.Add (rectangleCenter, rectanglePosition);
73             }
74         }
75     }
76
77     public bool ContainsEdge (Node a, Node b)
78     {
79         foreach (List<PossibleRectanglePosition> many in positions.Values)
80             foreach (PossibleRectanglePosition position in many) {
81                 if ((position.NodeA == a && position.NodeB == b) || (posit
82                     return true;
83             }
84         }
85     }
86     return false;
87 }
88
89 public IEnumerable<ValidRectanglePosition> ValidPositions ()
90 {
91     foreach (List<PossibleRectanglePosition> many in positions.Values)
92         foreach (var pair in many.SelectMany ((value, index) => many.S
93             (first, second) => new { first, second })) {

```

```

0 94 List<PossibleRectanglePosition> pos
0 95 = new PossibleRectanglePosition[] { pair.first, pair.s
0 96 if (pos.Count == 2) {
0 97     for (int i = 0; i <= 1; ++i) {
0 98         int first = i % 2;
0 99         int second = (i + 1) % 2;
0 100         Edge edgeAB = pos [first].Edge;
0 101         Edge edgeCD = pos [second].Edge;
0 102         Node nodeA = pos [first].NodeA;
0 103         Node nodeB = pos [first].NodeB;
0 104         Node nodeC = pos [second].NodeA;
0 105         Node nodeD = pos [second].NodeB;
0 106         if (nodeB == nodeC || (nodeA-nodeB) == (nodeC-node
0 107             var valid = new ValidRectanglePosition {
0 108                 EdgeAB = edgeAB,
0 109                 EdgeCD = edgeCD,
0 110                 NodeA = nodeA,
0 111                 NodeB = nodeB,
0 112                 NodeC = nodeC,
0 113                 NodeD = nodeD,
0 114                 Position = pos[first].Position,
0 115                 IsVirtual = pos[first].IsVirtual || pos[se
0 116             };
0 117         yield return valid;
0 118     }
0 119 }
0 120 }
0 121 }
0 122 }
0 123 }
0 124 }
0 125
0 126 public struct PossibleRectanglePosition {
0 127     public Edge Edge;
0 128     public Node NodeA;
0 129     public Node NodeB;
0 130     public Vector3 Position;
0 131     public bool IsVirtual;
0 132 }
0 133
0 134 public struct ValidRectanglePosition {
0 135     public Edge EdgeAB;
0 136     public Edge EdgeCD;
0 137     public Node NodeA;
0 138     public Node NodeB;
0 139     public Node NodeC;
0 140     public Node NodeD;
0 141     public Vector3 Position;
0 142     public bool IsVirtual;
0 143 }
0 144 }

```

Knot3.Game.Utilities.FileIndex

Summary

Class:	Knot3.Game.Utilities.FileIndex
Assembly:	Knot3
File(s):	sers\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Utilities\FileIndex.cs
Coverage:	44.4%
Covered lines:	12
Uncovered lines:	15
Coverable lines:	27
Total lines:	80

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	93.33	100
Add(...)	1	0	0
Remove(...)	1	0	0
Contains(...)	1	0	0
Save()	1	0	0

File(s)

sers\Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Utilities\FileIndex.cs

```

#   Line Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System.Collections.Generic;
32 using System.Diagnostics.CodeAnalysis;

```

```

33 using System.IO;
34
35 using Knot3.Framework.Storage;
36
37 namespace Knot3.Game.Utilities
38 {
39     public class FileIndex
40     {
41         private HashSet<string> hashes;
42         private string filename;
43
44         public FileIndex (string filename)
45         {
46             this.filename = filename;
47             try {
48                 hashes = new HashSet<string> (FileUtility.ReadFrom (filename))
49             }
50             catch (System.ArgumentException) {
51                 hashes = new HashSet<string> ();
52             }
53             catch (IOException) {
54                 hashes = new HashSet<string> ();
55             }
56         }
57
58         public void Add (string hash)
59         {
60             hashes.Add (hash);
61             Save ();
62         }
63
64         public void Remove (string hash)
65         {
66             hashes.Remove (hash);
67             Save ();
68         }
69
70         public bool Contains (string hash)
71         {
72             return hashes.Contains (hash);
73         }
74
75         private void Save ()
76         {
77             File.WriteAllText (filename, string.Join ("\n", hashes));
78         }
79     }
80 }

```

Knot3.Game.Utilities.SavegameLoader'2

Summary

Class:	Knot3.Game.Utilities.SavegameLoader'2
Assembly:	Knot3
File(s):	Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Utilities\SavegameLoader.cs
Coverage:	38.7%
Covered lines:	12
Uncovered lines:	19
Coverable lines:	31
Total lines:	111

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
FindSavegames(...)	1	100	100
AddFileToList(...)	3	0	0

File(s)

Pascal\Documents\GitHub\knot3-code\Game\Knot3.Game\Utilities\SavegameLoader.cs

```

#   Line Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Diagnostics.CodeAnalysis;
33
34 using Knot3.Framework.Platform;

```



```

35 using Knot3.Framework.Storage;
36
37 using Knot3.Game.Data;
38
39 namespace Knot3.Game.Utilities
40 {
41     public class SavegameLoader<Savegame, SavegameMetaData>
42     {
43         public ISavegameIO<Savegame, SavegameMetaData> FileFormat { get; set; }
44
45         public FileIndex fileIndex { get; private set; }
46
47         public string IndexName;
48         private Action<string, SavegameMetaData> OnSavegameFound;
49
50         public SavegameLoader (ISavegameIO<Savegame, SavegameMetaData> fileFor
51         {
52             FileFormat = fileFormat;
53             IndexName = indexName;
54         }
55
56         public void FindSavegames (Action<string, SavegameMetaData> onSavegame
57         {
58             // Erstelle einen neuen Index, der eine Datei mit dem angegebenen In
59             fileIndex = new FileIndex (SystemInfo.SavegameDirectory + SystemIn
60
61             // Diese Verzeichnisse werden nach Spielstnden durchsucht
62             string[] searchDirectories = new string[] {
63                 SystemInfo.BaseDirectory,
64                 SystemInfo.SavegameDirectory
65             };
66             Log.Debug ("Search for Savegames: ", string.Join ("", searchDire
67
68             // Suche nach Spielstanddateien und fllle das Men auf
69             OnSavegameFound = onSavegameFound;
70             FileUtility.SearchFiles (searchDirectories, FileFormat.FileExtensi
71         }
72
73         /// <summary>
74         /// Diese Methode wird fr jede gefundene Spielstanddatei aufgerufen
75         /// </summary>
76         private void AddFileToList (string filename)
77         {
78             // Lese die Datei ein und erstelle einen Hashcode
79             string hashCode = FileUtility.GetHash (filename);
80
81             // Ist dieser Hashcode im Index enthalten?
82             // Dann wre der Spielstand gltig, sonst ungltig oder unbekannt.
83             bool isValid = fileIndex.Contains (hashCode);
84
85             // Wenn der Spielstand ungltig oder unbekannt ist...
86             if (!isValid) {
87                 try {
88                     // Lade den Knoten und prfe, ob Exceptions auftreten
89                     FileFormat.Load (filename);
90                     // Keine Exceptions? Dann ist enthlt die Datei einen glt
91                     isValid = true;
92                     fileIndex.Add (hashCode);
93                 }
94                 catch (Exception ex) {
95                     // Es ist eine Exception aufgetreten, der Knoten ist offen

```

```
0 96          Log.Debug (ex);
0 97          isValid = false;
0 98      }
0 99  }
100
101      // Falls der Knoten gltig ist, entweder laut Index oder nach ber
0 102  if (isValid) {
103      // Lade die Metadaten
0 104      SavegameMetaData meta = FileFormat.LoadMetaData (filename);
105
106      // Rufe die Callback-Funktion auf
0 107      OnSavegameFound (filename, meta);
0 108  }
0 109  }
110  }
111 }
```

Knot3.Framework.Audio.AudioManager

Summary

Class:	Knot3.Framework.Audio.AudioManager
Assembly:	Knot3.Framework
File(s):	al\Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\AudioManager.cs
Coverage:	15.5%
Covered lines:	14
Uncovered lines:	76
Coverable lines:	90
Total lines:	202

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
Initialize()	1	0	0
Initialize(...)	3	13.33	20
Reset()	1	100	100
AddOggAudioFile(...)	3	0	0
LoadOggAudioFile(...)	1	0	0
StartBackgroundMusic	3	0	0
PlaySound(...)	2	0	0
Volume(...)	1	0	0
SetVolume(...)	1	0	0
ValidVolume(...)	1	0	0
.cctor()	1	100	100

File(s)

al\Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\AudioManager.cs

#	Line	Coverage
	1	/*
	2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
	3	* Gerd Augsburg, Christina Erler, Daniel Warzel
	4	*
	5	* This source code file is part of Knot3. Copying, redistribution and
	6	* use of the source code in this file in source and binary forms,
	7	* with or without modification, are permitted provided that the conditions
	8	* of the MIT license are met:
	9	*
	10	Permission is hereby granted, free of charge, to any person obtaining a c
	11	of this software and associated documentation files (the "Software"), to
	12	in the Software without restriction, including without limitation the rig
	13	to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
	14	copies of the Software, and to permit persons to whom the Software is
	15	furnished to do so, subject to the following conditions:
	16	*
	17	The above copyright notice and this permission notice shall be included i
	18	copies or substantial portions of the Software.
	19	*
	20	THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
	21	IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
	22	FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
	23	AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
	24	LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
	25	OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN

```

26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31  using System;
32  using System.Collections.Generic;
33  using System.Diagnostics.CodeAnalysis;
34  using System.IO;
35
36  using Microsoft.Xna.Framework;
37  using Microsoft.Xna.Framework.Audio;
38
39  using Knot3.Framework.Core;
40  using Knot3.Framework.Platform;
41  using Knot3.Framework.Storage;
42  using Knot3.Framework.Utilities;
43
44  namespace Knot3.Framework.Audio
45  {
46      public abstract class AudioManager : DrawableGameComponent
47      {
48          /// <summary>
49          /// Eine Zuordnung zwischen dem Typ der Audiodateien und den Ordnern u
50          /// in denen sich die Audiodateien befinden.
51          /// </summary>
18 52          public Dictionary<Sound, string> AudioDirectories { get; private set; }
53
54          // Enthlt alle gefunden Audiodateien, sortiert nach ihrem Zweck
3 55          private Dictionary<Sound, HashSet<IAudioFile>> AudioFiles = new Dictio
56
57          /// <summary>
58          /// Die aktuell verwendete Hintergrundmusik.
59          /// </summary>
60          public Sound BackgroundMusic
61          {
0 62              get {
0 63                  return _backgroundMusic;
0 64              }
0 65              set {
0 66                  if (value != Sound.None && value != _backgroundMusic) {
0 67                      _backgroundMusic = value;
0 68                      StartBackgroundMusic ();
0 69                  }
0 70              }
71          }
72
3 73          private Sound _backgroundMusic = Sound.None;
74
75          /// <summary>
76          /// Enthlt die Playlist, die aktuell abgespielt wird,
77          /// oder null, falls keine Playlist abgespielt wird.
78          /// </summary>
0 79          public IPlaylist Playlist { get; set; }
80
1 81          private static Dictionary<Sound, float> VolumeMap = new Dictionary<Sou
82
83          /// <summary>
84          /// Erstellt einen neuen AudioManager fr den angegebenen Spielzustand
85          /// </summary>
3 86          public AudioManager (Game game)

```

```

87         : base (game)
3 88     {
3 89         AudioDirectories = new Dictionary<Sound, string> ();
3 90     }
91
92     public override void Initialize ()
0 93     {
0 94         Initialize (SystemInfo.RelativeContentDirectory);
0 95     }
96
97     public virtual void Initialize (string directory)
3 98     {
3 99         base.Initialize ();
100
0 101         if (AudioFiles.Count == 0) {
102             // Erstelle fr alle Enum-Werte von Sound ein HashSet
0 103             foreach (Sound soundType in Sound.Values) {
0 104                 AudioFiles [soundType] = new HashSet<IAudioFile> ();
0 105                 VolumeMap [soundType] = ValidVolume (Config.Default ["volu
0 106             }
107
108             // Suche nach OGG-Dateien
0 109             FileUtility.SearchFiles (directory, new string[] {".ogg"}, Add
0 110         }
0 111     }
112
113     public void Reset ()
3 114     {
3 115         AudioFiles.Clear ();
3 116         VolumeMap.Clear ();
3 117     }
118
119     private void AddOggAudioFile (string filepath)
0 120     {
0 121         filepath = filepath.Replace (@"\", "/");
122
0 123         foreach (KeyValuePair<Sound,string> pair in AudioDirectories) {
0 124             Sound soundType = pair.Key;
0 125             string directory = pair.Value;
0 126             if (filepath.ToLower ().Contains (directory.ToLower ())) {
0 127                 string name = Path.GetFileName (filepath);
0 128                 LoadOggAudioFile (filepath, name, soundType);
0 129                 break;
130             }
0 131         }
0 132     }
133
134     private void LoadOggAudioFile (string filepath, string name, Sound sou
0 135     {
0 136         try {
137             // erstelle ein OggVorbisFile-Objekt
0 138             AudioFiles [soundType].Add (new OggVorbisFile (name, filepath,
0 139         }
0 140         catch (Exception ex) {
141             // egal, warum das laden nicht klappt; mehr als die Fehlermeld
142             // macht wegen einer fehlenden Musikdatei keinen Sinn
0 143             Log.Debug ("Failed to load ogg audio file (" , soundType, "): "
0 144                 Log.Debug (ex);
0 145         }
0 146     }
147

```

```

148     private void StartBackgroundMusic ()
149     {
150         if (Playlist != null) {
151             Playlist.Stop ();
152         }
153         Log.Debug ("Background Music: ", BackgroundMusic);
154         if (AudioFiles.ContainsKey (BackgroundMusic)) {
155             Playlist = new LoopPlaylist (AudioFiles [BackgroundMusic]);
156             Playlist.Shuffle ();
157             Playlist.Start ();
158         }
159         else {
160             Log.Message ("Warning: ", BackgroundMusic, ": no sound files a
161         }
162     }
163
164     public void PlaySound (Sound sound)
165     {
166         Log.Debug ("Sound: ", sound);
167         if (AudioFiles [sound].Count > 0) {
168             AudioFiles [sound].RandomElement ().Play ();
169         }
170         else {
171             Log.Debug ("There are no audio files for: ", sound);
172         }
173     }
174
175     [ExcludeFromCodeCoverageAttribute]
176     public override void Update (GameTime time)
177     {
178         if (Playlist != null) {
179             Playlist.Update (time);
180         }
181         base.Update (time);
182     }
183
184     public static float Volume (Sound soundType)
185     {
186         return VolumeMap [soundType];
187     }
188
189     public static void SetVolume (Sound soundType, float volume)
190     {
191         volume = ValidVolume (volume);
192         VolumeMap [soundType] = volume;
193         Config.Default ["volume", soundType.ToString (), 1] = volume;
194         Log.Debug ("Set Volume (" , soundType, "): ", volume);
195     }
196
197     public static float ValidVolume (float volume)
198     {
199         return MathHelper.Clamp (volume, 0.0f, 2.0f);
200     }
201 }
202 }

```

Knot3.Framework.Audio.LoopPlaylist

Summary

Class:	Knot3.Framework.Audio.LoopPlaylist
Assembly:	Knot3.Framework
File(s):	al\Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\LoopPlaylist.cs
Coverage:	0%
Covered lines:	0
Uncovered lines:	26
Coverable lines:	26
Total lines:	116

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	2	0	0
Shuffle()	1	0	0
Start()	2	0	0
Stop()	2	0	0

File(s)

al\Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\LoopPlaylist.cs

#	Line	Coverage
	1	/*
	2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
	3	* Gerd Augsburg, Christina Erler, Daniel Warzel
	4	*
	5	* This source code file is part of Knot3. Copying, redistribution and
	6	* use of the source code in this file in source and binary forms,
	7	* with or without modification, are permitted provided that the conditions
	8	* of the MIT license are met:
	9	*
	10	* Permission is hereby granted, free of charge, to any person obtaining a c
	11	* of this software and associated documentation files (the "Software"), to
	12	* in the Software without restriction, including without limitation the rig
	13	* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
	14	* copies of the Software, and to permit persons to whom the Software is
	15	* furnished to do so, subject to the following conditions:
	16	*
	17	* The above copyright notice and this permission notice shall be included i
	18	* copies or substantial portions of the Software.
	19	*
	20	* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
	21	* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
	22	* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
	23	* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
	24	* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
	25	* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
	26	* SOFTWARE.
	27	*
	28	* See the LICENSE file for full license details of the Knot3 project.
	29	*/
	30	
	31	using System.Collections.Generic;
	32	using System.Diagnostics.CodeAnalysis;
	33	using System.Linq;

```
34
35 using Microsoft.Xna.Framework;
36 using Microsoft.Xna.Framework.Audio;
37
38 using Knot3.Framework.Platform;
39 using Knot3.Framework.Utilities;
40
41 namespace Knot3.Framework.Audio
42 {
43     /// <summary>
44     /// Diese Klasse repräsentiert eine Playlist, deren Audiodateien der Reihe
45     /// Endlosschleife abgespielt werden.
46     /// </summary>
47     public class LoopPlaylist : IPlaylist
48     {
49         private List<IAudioFile> Sounds;
50         private int index;
51
52         public SoundState State { get; private set; }
53
54         /// <summary>
55         /// Erstellt eine neue Playlist.
56         /// </summary>
57         /// <param name='sounds'>
58         /// Die abzuspielenden Audiodateien.
59         /// </param>
60         public LoopPlaylist (IEnumerable<IAudioFile> sounds)
61         {
62             Sounds = sounds.ToList ();
63             index = 0;
64             State = SoundState.Stopped;
65
66             Log.Debug ("Created new playlist (", Sounds.Count, " songs)");
67             foreach (IAudioFile sound in Sounds) {
68                 Log.Debug (" - ", sound.Name);
69             }
70         }
71
72         public void Shuffle ()
73         {
74             Sounds = Sounds.Shuffle ().ToList ();
75         }
76
77         /// <summary>
78         /// Starte die Wiedergabe.
79         /// </summary>
80         public void Start ()
81         {
82             if (Sounds.Count > 0) {
83                 State = SoundState.Playing;
84                 Sounds .At (index).Play ();
85             }
86         }
87
88         /// <summary>
89         /// Stoppe die Wiedergabe.
90         /// </summary>
91         public void Stop ()
92         {
93             if (Sounds.Count > 0) {
94                 State = SoundState.Stopped;
```



```
0 95         Sounds.At (index).Stop ();
0 96     }
0 97 }
98
99     /// <summary>
100     /// Wird fr jeden Frame aufgerufen.
101     /// </summary>
102     [ExcludeFromCodeCoverageAttribute]
103     public void Update (GameTime time)
104     {
105         if (Sounds.Count > 0) {
106             if (State == SoundState.Playing && Sounds.At (index).State !=
107                 ++index;
108                 Sounds.At (index).Play ();
109             }
110         }
111         if (index >= 0 && index < Sounds.Count) {
112             Sounds.At (index).Update (time);
113         }
114     }
115 }
116 }
```

Knot3.Framework.Audio.OggVorbisFile

Summary

Class:	Knot3.Framework.Audio.OggVorbisFile
Assembly:	Knot3.Framework
File(s):	l\Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\OggVorbisFile.cs
Coverage:	0%
Covered lines:	0
Uncovered lines:	53
Coverable lines:	53
Total lines:	128

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	7	0	0
Play()	1	0	0
Stop()	1	0	0
WriteWave(...)	1	0	0

File(s)

l\Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\OggVorbisFile.cs

#	Line	Coverage
	1	/*
	2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
	3	* Gerd Augsburg, Christina Erler, Daniel Warzel
	4	*
	5	* This source code file is part of Knot3. Copying, redistribution and
	6	* use of the source code in this file in source and binary forms,
	7	* with or without modification, are permitted provided that the conditions
	8	* of the MIT license are met:
	9	*
	10	* Permission is hereby granted, free of charge, to any person obtaining a c
	11	* of this software and associated documentation files (the "Software"), to
	12	* in the Software without restriction, including without limitation the rig
	13	* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
	14	* copies of the Software, and to permit persons to whom the Software is
	15	* furnished to do so, subject to the following conditions:
	16	*
	17	* The above copyright notice and this permission notice shall be included i
	18	* copies or substantial portions of the Software.
	19	*
	20	* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
	21	* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
	22	* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
	23	* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
	24	* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
	25	* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
	26	* SOFTWARE.
	27	*
	28	* See the LICENSE file for full license details of the Knot3 project.
	29	*/
	30	
	31	using System;
	32	using System.Diagnostics.CodeAnalysis;
	33	using System.IO;

```

34 using System.Linq;
35
36 using Microsoft.Xna.Framework;
37 using Microsoft.Xna.Framework.Audio;
38
39 using OggSharp;
40
41 using Knot3.Framework.Platform;
42
43 namespace Knot3.Framework.Audio
44 {
45     public class OggVorbisFile : IAudioFile
46     {
0 47         public string Name { get; private set; }
48
0 49         public SoundState State { get { return internalFile.State; } }
50
51         private SoundEffectFile internalFile;
52
0 53         public OggVorbisFile (string name, string filepath, Sound soundType)
0 54         {
0 55             Name = name;
0 56             string cacheFile = SystemInfo.DecodedMusicCache
57                 + SystemInfo.PathSeparator.ToString ()
58                 + soundType.ToString ()
59                 + "-"
60                 + name.GetHashCode ().ToString ()
61                 + ".wav";
62
0 63             Log.BlockList (id: 33, before: " - ", after: "", begin: "Load ogg
0 64             Log.BlockList (id: 34, before: " - ", after: "", begin: "Decode o
65
66             byte[] data;
67             try {
0 68                 Log.ListElement (33, "[", soundType, "]", name);
0 69                 data = File.ReadAllBytes (cacheFile);
0 70             }
0 71             catch (Exception) {
0 72                 Log.ListElement (34, "[", soundType, "]", name);
0 73                 OggDecoder decoder = new OggDecoder ();
0 74                 decoder.Initialize (TitleContainer.OpenStream (filepath));
0 75                 data = decoder.SelectMany (chunk => chunk.Bytes.Take (chunk.Le
0 76                 using (MemoryStream stream = new MemoryStream ())
0 77                 using (BinaryWriter writer = new BinaryWriter (stream)) {
0 78                     WriteWave (writer, decoder.Stereo ? 2 : 1, decoder.SampleR
0 79                     stream.Position = 0;
0 80                     data = stream.ToArray ();
0 81                 }
0 82                 File.WriteAllBytes (cacheFile, data);
0 83             }
84
0 85             using (MemoryStream stream = new MemoryStream (data)) {
0 86                 stream.Position = 0;
0 87                 SoundEffect soundEffect = SoundEffect.FromStream (stream);
0 88                 internalFile = new SoundEffectFile (name, soundEffect, soundTy
0 89             }
0 90         }
91
92         public void Play ()
93         {
0 94             internalFile.Play ();

```

```
0 95      }
96
97      public void Stop ()
0 98      {
0 99          internalFile.Stop ();
0 100     }
101
102     [ExcludeFromCodeCoverageAttribute]
103     public void Update (GameTime time)
104     {
105         internalFile.Update (time);
106     }
107
108     private static void WriteWave (BinaryWriter writer, int channels, int
0 109     {
0 110         writer.Write (new char[4] { 'R', 'I', 'F', 'F' });
0 111         writer.Write ((int)(36 + data.Length));
0 112         writer.Write (new char[4] { 'W', 'A', 'V', 'E' });
113
0 114         writer.Write (new char[4] { 'f', 'm', 't', ' ' });
0 115         writer.Write ((int)16);
0 116         writer.Write ((short)1);
0 117         writer.Write ((short)channels);
0 118         writer.Write ((int)rate);
0 119         writer.Write ((int)(rate * ((16 * channels) / 8)));
0 120         writer.Write ((short)((16 * channels) / 8));
0 121         writer.Write ((short)16);
122
0 123         writer.Write (new char[4] { 'd', 'a', 't', 'a' });
0 124         writer.Write ((int)data.Length);
0 125         writer.Write (data);
0 126     }
127 }
128 }
```

Knot3.Framework.Audio.Sound

Summary

Class:	Knot3.Framework.Audio.Sound
Assembly:	Knot3.Framework
File(s):	rs\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\Sound.cs
Coverage:	28.5%
Covered lines:	8
Uncovered lines:	20
Coverable lines:	28
Total lines:	108

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
op_Implicit(...)	1	0	0
op_Equality(...)	4	0	0
op_Inequality(...)	1	0	0
Equals(...)	2	0	0
Equals(...)	2	0	0
.cctor()	1	100	100

File(s)

rs\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\Sound.cs

```

#   Line  Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30

```

```

31 using System;
32 using System.Collections.Generic;
33 using System.Diagnostics.CodeAnalysis;
34 using System.Linq;
35
36 namespace Knot3.Framework.Audio
37 {
38     public class Sound : IEquatable<Sound>
39     {
40         /// <summary>
41         /// Gibt alle Sound-Werte zurck.
42         /// </summary>
0 43         public static Sound[] Values { get { return _values.Select (name => ne
44
1 45         private static HashSet<string> _values = new HashSet<string> ();
46
47         /// <summary>
48         /// Kein Sound.
49         /// </summary>
1 50         public static readonly Sound None = new Sound ("None");
51
21 52         public string Name { get; private set; }
53
6 54         public Sound (string name)
6 55         {
6 56             Name = name;
6 57             _values.Add (name);
6 58         }
59
60         [ExcludeFromCodeCoverageAttribute]
61         public override string ToString ()
62         {
63             return Name;
64         }
65
66         public static implicit operator string (Sound layer)
0 67         {
0 68             return layer.Name;
0 69         }
70
71         public static bool operator == (Sound a, Sound b)
0 72         {
73             // If both are null, or both are same instance, return true.
0 74             if (System.Object.ReferenceEquals (a, b)) {
0 75                 return true;
76             }
77
78             // If one is null, but not both, return false.
0 79             if (((object)a == null) || ((object)b == null)) {
0 80                 return false;
81             }
82
83             // Return true if the fields match:
0 84             return a.Name == b.Name;
0 85         }
86
87         public static bool operator != (Sound d1, Sound d2)
0 88         {
0 89             return !(d1 == d2);
0 90         }
91

```

```
92         public bool Equals (Sound other)
0 93         {
0 94             return other != null && Name == other.Name;
0 95         }
96
97         public override bool Equals (object other)
0 98         {
0 99             return other != null && Equals (other as Sound);
0 100        }
101
102         [ExcludeFromCodeCoverageAttribute]
103         public override int GetHashCode ()
104         {
105             return Name.GetHashCode ();
106         }
107     }
108 }
```

Knot3.Framework.Audio.SoundEffectFile

Summary

Class:	Knot3.Framework.Audio.SoundEffectFile
Assembly:	Knot3.Framework
File(s):	Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\SoundEffectFile.cs
Coverage:	0%
Covered lines:	0
Uncovered lines:	19
Coverable lines:	19
Total lines:	94

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	0	0
Play()	1	0	0
Stop()	1	0	0

File(s)

Documents\GitHub\knot3-code\framework\Knot3.Framework\Audio\SoundEffectFile.cs

#	Line	Coverage
	1	/*
	2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
	3	* Gerd Augsburg, Christina Erler, Daniel Warzel
	4	*
	5	* This source code file is part of Knot3. Copying, redistribution and
	6	* use of the source code in this file in source and binary forms,
	7	* with or without modification, are permitted provided that the conditions
	8	* of the MIT license are met:
	9	*
	10	* Permission is hereby granted, free of charge, to any person obtaining a c
	11	* of this software and associated documentation files (the "Software"), to
	12	* in the Software without restriction, including without limitation the rig
	13	* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
	14	* copies of the Software, and to permit persons to whom the Software is
	15	* furnished to do so, subject to the following conditions:
	16	*
	17	* The above copyright notice and this permission notice shall be included i
	18	* copies or substantial portions of the Software.
	19	*
	20	* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
	21	* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
	22	* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
	23	* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
	24	* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
	25	* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
	26	* SOFTWARE.
	27	*
	28	* See the LICENSE file for full license details of the Knot3 project.
	29	*/
	30	
	31	using System.Diagnostics.CodeAnalysis;
	32	
	33	using Microsoft.Xna.Framework;
	34	using Microsoft.Xna.Framework.Audio;


```

35
36 using Knot3.Framework.Platform;
37
38 namespace Knot3.Framework.Audio
39 {
40     /// <summary>
41     /// Ein Wrapper um die SoundEffect-Klasse des XNA-Frameworks.
42     /// </summary>
43     public class SoundEffectFile : IAudioFile
44     {
45         /// <summary>
46         /// Der Anzeigename des SoundEffects.
47         /// </summary>
0 48         public string Name { get; private set; }
49
50         /// <summary>
51         /// Gibt an, ob die Wiedergabe luft oder gestoppt bzw. pausiert ist.
52         /// </summary>
0 53         public SoundState State { get { return Instance.State; } }
54
0 55         public SoundEffect SoundEffect { get; private set; }
56
57         private SoundEffectInstance Instance;
58
59         private Sound SoundType;
60         private float volume;
61
62         /// <summary>
63         /// Erstellt eine neue SoundEffect-Datei mit dem angegebenen Anzeigena
64         /// </summary>
0 65         public SoundEffectFile (string name, SoundEffect soundEffect, Sound so
0 66         {
0 67             Name = name;
0 68             SoundEffect = soundEffect;
0 69             Instance = soundEffect.CreateInstance ();
0 70             SoundType = soundType;
0 71         }
72
73         public void Play ()
0 74         {
0 75             Log.Debug ("Play: ", Name);
0 76             Instance.Volume = volume = AudioManager.Volume (SoundType);
0 77             Instance.Play ();
0 78         }
79
80         public void Stop ()
0 81         {
0 82             Log.Debug ("Stop: ", Name);
0 83             Instance.Stop ();
0 84         }
85
86         [ExcludeFromCodeCoverageAttribute]
87         public void Update (GameTime time)
88         {
89             if (volume != AudioManager.Volume (SoundType)) {
90                 Instance.Volume = volume = AudioManager.Volume (SoundType);
91             }
92         }
93     }
94 }

```

Knot3.Framework.Core.Camera

Summary

Class:	Knot3.Framework.Core.Camera
Assembly:	Knot3.Framework
File(s):	rs\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Core\Camera.cs
Coverage:	65.3%
Covered lines:	83
Uncovered lines:	44
Coverable lines:	127
Total lines:	352

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	2	100	100
UpdateMatrices(...)	1	100	100
GetMouseRay(...)	1	0	0
ResetCamera()	1	100	100
StartSmoothMove(...)	2	0	0
UpdateSmoothMove(...)	2	0	0
To3D(...)	2	100	100
To2D(...)	1	100	100

File(s)

rs\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Core\Camera.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */

```

```

30
31 using System;
32 using System.Diagnostics.CodeAnalysis;
33
34 using Microsoft.Xna.Framework;
35 using Microsoft.Xna.Framework.Graphics;
36
37 using Knot3.Framework.Math;
38 using Knot3.Framework.Storage;
39 using Knot3.Framework.Utilities;
40
41 namespace Knot3.Framework.Core
42 {
43     /// <summary>
44     /// Jede Instanz der World-Klasse hlt eine fr diese Spielwelt verwendete
45     /// Die Hauptfunktion der Kamera-Klasse ist das Berechnen der drei Matrize
46     /// und Skalierung von 3D-Objekten in einer bestimmten Spielwelt bentigt
47     /// Um diese Matrizen zu berechnen, bentigt die Kamera unter Anderem Info
48     /// das aktuelle Kamera-Ziel und das Field of View.
49     /// </summary>
50     public sealed class Camera : ScreenComponent
51     {
52         private Vector3 _position;
53
54         /// <summary>
55         /// Die Position der Kamera.
56         /// </summary>
57         public Vector3 Position
58         {
75 59             get { return _position; }
16 60             set {
16 61                 OnViewChanged ();
16 62                 if ((value.X.Abs () <= MaxPositionDistance && value.Y.Abs () <
16 63                     && value.Z.Abs () <= MaxPositionDistance) || MaxPositi
16 64                     _position = value;
16 65             }
16 66         }
67     }
68
69     private Vector3 _target;
70
71     /// <summary>
72     /// Das Ziel der Kamera.
73     /// </summary>
74     public Vector3 Target
75     {
81 76         get { return _target; }
12 77         set {
12 78             OnViewChanged ();
12 79             _target = value;
12 80         }
81     }
82
83     private float _foV;
84
85     /// <summary>
86     /// Das Sichtfeld.
87     /// </summary>
88     public float FoV
89     {
36 90         get { return _foV; }

```

```

11 91         set {
11 92             _foV = MathHelper.Clamp (value, 10, 70);
11 93             OnViewChanged ();
11 94         }
11 95     }
11 96
11 97     /// <summary>
11 98     /// Die View-Matrix wird ber die statische Methode CreateLookAt der K
11 99     /// mit Matrix.CreateLookAt (Position, Target, Vector3.Up) berechnet.
11 100    /// </summary>
23 101    public Matrix ViewMatrix { get; private set; }
102
103    /// <summary>
104    /// Die World-Matrix wird mit Matrix.CreateFromYawPitchRoll und den dr
105    /// </summary>
12 106    public Matrix WorldMatrix { get; private set; }
107
108    /// <summary>
109    /// Die Projektionsmatrix wird ber die statische XNA-Methode Matrix.C
110    /// </summary>
23 111    public Matrix ProjectionMatrix { get; private set; }
112
113    /// <summary>
114    /// Berechnet ein Bounding-Frustum, das bentigt wird, um festzustellen
115    /// </summary>
9 116    public BoundingBox ViewFrustum { get; private set; }
117
118    /// <summary>
119    /// Eine Referenz auf die Spielwelt, fr welche die Kamera zustndig i
120    /// </summary>
29 121    private World World { get; set; }
122
123    /// <summary>
124    /// Die Rotationswinkel.
125    /// </summary>
34 126    public Angles3 Rotation { get; set; }
127
16 128    public Vector3 UpVector { get; private set; }
129
56 130    public float MaxPositionDistance { get; set; }
131
47 132    public Action OnViewChanged = () => {};
133    private float aspectRatio;
134    private float nearPlane;
135    private float farPlane;
8 136    private Vector3 defaultPosition = new Vector3 (400, 400, 700);
137
138    /// <summary>
139    /// Erstellt eine neue Kamera in einem bestimmten IGameScreen fr eine
140    /// </summary>
8 141    public Camera (IScreen screen, World world)
142    : base (screen, DisplayLayer.None)
8 143    {
8 144        World = world;
8 145        Position = defaultPosition;
8 146        Target = Vector3.Zero;
8 147        UpVector = Vector3.Up;
8 148        Rotation = Angles3.Zero;
8 149        MaxPositionDistance = 5000;
150
8 151        FoV = 60;

```

```

8 152         nearPlane = 0.5f;
8 153         farPlane = 15000.0f;
      154
8 155         UpdateMatrices (null);
8 156     }
      157
      158     /// <summary>
      159     /// Die Blickrichtung.
      160     /// </summary>
      161     public Vector3 PositionToTargetDirection
      162     {
0 163         get {
0 164             return Vector3.Normalize (Target - Position);
0 165         }
      166     }
      167
      168     public Vector3 PositionToArcballTargetDirection
      169     {
0 170         get {
0 171             return Vector3.Normalize (ArcballTarget - Position);
0 172         }
      173     }
      174
      175     /// <summary>
      176     /// Der Abstand zwischen der Kamera und dem Kamera-Ziel.
      177     /// </summary>
      178     public float PositionToTargetDistance
      179     {
4 180         get {
4 181             return Position.DistanceTo (Target);
4 182         }
4 183         set {
4 184             Position = Position.SetDistanceTo (Target, value);
4 185         }
      186     }
      187
      188     public float PositionToArcballTargetDistance
      189     {
0 190         get {
0 191             return Position.DistanceTo (ArcballTarget);
0 192         }
0 193         set {
0 194             Position = Position.SetDistanceTo (ArcballTarget, value);
0 195         }
      196     }
      197
      198     /// <summary>
      199     /// Wird fr jeden Frame aufgerufen.
      200     /// </summary>
      201     [ExcludeFromCodeCoverageAttribute]
      202     public override void Update (GameTime time)
      203     {
      204         // Setze den Viewport auf den der aktuellen Spielwelt
      205         Viewport original = Screen.Viewport;
      206         Screen.Viewport = World.Viewport;
      207
      208         UpdateMatrices (time);
      209         UpdateSmoothMove (time);
      210
      211         // Setze den Viewport wieder auf den ganzen Screen
      212         Screen.Viewport = original;

```

```

213     }
214
215     private void UpdateMatrices (GameTime time)
216     {
217         aspectRatio = Screen.Viewport.AspectRatio;
218         farPlane = MaxPositionDistance * 4;
219         ViewMatrix = Matrix.CreateLookAt (Position, Target, UpVector);
220         WorldMatrix = Matrix.CreateFromYawPitchRoll (Rotation.Y, Rotation.
221         ProjectionMatrix = Matrix.CreatePerspectiveFieldOfView (MathHelper
222         ViewFrustum = new BoundingFrustum (ViewMatrix * ProjectionMatrix);
223     }
224
225     /// <summary>
226     /// Berechnet einen Strahl fr die angegebenen 2D-Mausposition.
227     /// </summary>
228     public Ray GetMouseRay (ScreenPoint mousePosition)
229     {
230         Viewport viewport = World.Viewport;
231
232         Vector3 nearPoint = new Vector3 (mousePosition.AbsoluteVector, 0);
233         Vector3 farPoint = new Vector3 (mousePosition.AbsoluteVector, 1);
234
235         nearPoint = viewport.Unproject (nearPoint, ProjectionMatrix, ViewM
236         farPoint = viewport.Unproject (farPoint, ProjectionMatrix, ViewMat
237
238         Vector3 direction = farPoint - nearPoint;
239         direction.Normalize ();
240
241         return new Ray (nearPoint, direction);
242     }
243
244     /// <summary>
245     /// Eine Position, um die rotiert werden soll, wenn der User die recht
246     /// </summary>
247     public Vector3 ArcballTarget
248     {
249         get {
250             if (World.SelectedObject != null) {
251                 return World.SelectedObject.Center ();
252             }
253             else {
254                 return Vector3.Zero;
255             }
256         }
257     }
258
259     public void ResetCamera ()
260     {
261         Position = defaultPosition;
262         Target = new Vector3 (0, 0, 0);
263         Rotation = Angles3.Zero;
264         FoV = 45;
265     }
266
267     private Vector3? smoothTarget = null;
268     private float smoothDistance = 0f;
269     private float smoothProgress = 0f;
270
271     public void StartSmoothMove (Vector3 target, GameTime time)
272     {
273         if (!InSmoothMove) {

```

```

0 274         smoothTarget = target;
0 275         smoothDistance = System.Math.Abs (Target.DistanceTo (target));
0 276         smoothProgress = 0f;
0 277     }
0 278 }
0 279
0 280 public bool InSmoothMove { get { return smoothTarget.HasValue && smoot
0 281
0 282 private void UpdateSmoothMove (GameTime time)
0 283 {
0 284     if (InSmoothMove) {
0 285         float distance = MathHelper.SmoothStep (0, smoothDistance, smo
0 286
0 287         smoothProgress += 0.05f;
0 288
0 289         //Log.Debug ("distance = ", distance);
0 290         Target = Target.SetDistanceTo (target: smoothTarget.Value, dis
0 291         World.Redraw = true;
0 292     }
0 293 }
0 294
0 295 /// <summary>
0 296 /// Berechne aus einer 2D-Positon (z.b. Mausposition) die entsprechend
0 297 /// Fr die fehlende dritte Koordinate wird eine Angabe einer weiteren
0 298 /// mit der die 3D-(Maus-)Position auf der selben Ebene liegen soll.
0 299 /// </summary>
0 300 public Vector3 To3D (ScreenPoint position, Vector3 nearTo)
0 301 {
2 302     if (Config.Default ["debug", "unproject", "SelectedObject"] == "Ne
3 303         Vector3 nearScreenPoint = new Vector3 (position.AbsoluteVector
1 304         Vector3 farScreenPoint = new Vector3 (position.AbsoluteVector,
1 305         Vector3 nearWorldPoint = World.Viewport.Unproject (
0 306             source: nearScreenPoint,
0 307             projection: World.Camera.Projecti
0 308             view: World.Camera.ViewMatrix,
0 309             world: Matrix.Identity
0 310         );
1 311         Vector3 farWorldPoint = World.Viewport.Unproject (
0 312             source: farScreenPoint,
0 313             projection: World.Camera.Projectio
0 314             view: World.Camera.ViewMatrix,
0 315             world: Matrix.Identity
0 316         );
0 317
1 318         Vector3 direction = farWorldPoint - nearWorldPoint;
0 319
1 320         float zFactor = -nearWorldPoint.Y / direction.Y;
1 321         Vector3 zeroWorldPoint = nearWorldPoint + direction * zFactor;
1 322         return zeroWorldPoint;
0 323     }
1 324     else {
1 325         Vector3 screenLocation = World.Viewport.Project (
0 326             source: nearTo,
0 327             projection: World.Camera.Projecti
0 328             view: World.Camera.ViewMatrix,
0 329             world: World.Camera.WorldMatrix
0 330         );
1 331         Vector3 currentMousePosition = World.Viewport.Unproject (
0 332             source: new Vector3 (positi
0 333             projection: World.Camera.Pr
0 334             view: World.Camera.ViewMatr

```

```
335                                     world: Matrix.Identity
336                                     );
1 337         return currentPosition;
338     }
2 339 }
340
341 public Vector2 To2D (Vector3 position)
2 342 {
2 343     Vector3 screenLocation = World.Viewport.Project (
344         source: position,
345         projection: World.Camera.ProjectionMa
346         view: World.Camera.ViewMatrix,
347         world: World.Camera.WorldMatrix
348     );
2 349     return new Vector2 (screenLocation.X, screenLocation.Y);
2 350 }
351 }
352 }
```


Knot3.Framework.Core.DisplayLayer

Summary

Class:	Knot3.Framework.Core.DisplayLayer
Assembly:	Knot3.Framework
File(s):	cal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Core\DisplayLayer.cs
Coverage:	98.1%
Covered lines:	52
Uncovered lines:	1
Coverable lines:	53
Total lines:	169

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
.ctor(...)	1	100	100
op_Addition(...)	1	100	100
op_Addition(...)	1	100	100
op_Multiply(...)	1	100	100
op_Equality(...)	4	88.89	71.43
op_Inequality(...)	1	100	100
Equals(...)	2	100	100
Equals(...)	2	100	66.67
op_Implicit(...)	1	100	100
op_Implicit(...)	1	100	100
.cctor()	1	100	100

File(s)

cal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Core\DisplayLayer.cs

```

#   Line Coverage
    1  /*
    2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3   *                                     Gerd Augsburg, Christina Erler, Daniel Warzel
    4   *
    5   * This source code file is part of Knot3. Copying, redistribution and
    6   * use of the source code in this file in source and binary forms,
    7   * with or without modification, are permitted provided that the conditions
    8   * of the MIT license are met:
    9   *
   10   *   Permission is hereby granted, free of charge, to any person obtaining a c
   11   *   of this software and associated documentation files (the "Software"), to
   12   *   in the Software without restriction, including without limitation the rig
   13   *   to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14   *   copies of the Software, and to permit persons to whom the Software is
   15   *   furnished to do so, subject to the following conditions:
   16   *
   17   *   The above copyright notice and this permission notice shall be included i
   18   *   copies or substantial portions of the Software.
   19   *
   20   *   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21   *   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22   *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
   23   *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
   24   *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
   25   *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN

```

```

26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31  using System;
32  using System.Diagnostics.CodeAnalysis;
33
34  using Knot3.Framework.Widgets;
35
36  namespace Knot3.Framework.Core
37  {
38      /// <summary>
39      /// Die Zeichenreihenfolge der Elemente der grafischen Benutzeroberfläche.
40      /// </summary>
41      public class DisplayLayer : IEquatable<DisplayLayer>
42      {
43          /// <summary>
44          /// Steht fr die hinterste Ebene bei der Zeichenreihenfolge.
45          /// </summary>
46          public static readonly DisplayLayer None = new DisplayLayer (0, "None"
47          /// <summary>
48          /// Steht fr eine Ebene hinter der Spielwelt, z.B. um
49          /// Hintergrundbilder darzustellen.
50          /// </summary>
51          public static readonly DisplayLayer Background = new DisplayLayer (10,
52          /// <summary>
53          /// Steht fr die Ebene in der die Spielwelt dargestellt wird.
54          /// </summary>
55          public static readonly DisplayLayer GameWorld = new DisplayLayer (20,
56          public static readonly DisplayLayer ScreenUI = new DisplayLayer (30, "
57          /// <summary>
58          /// Steht fr die Ebene in der die Dialoge dargestellt werden.
59          /// Dialoge werden vor der Spielwelt gezeichnet, damit der Spieler dam
60          /// </summary>
61          public static readonly DisplayLayer Dialog = new DisplayLayer (50, "Di
62          /// <summary>
63          /// Steht fr die Ebene in der Mens gezeichnet werden. Mens werden i
64          /// </summary>
65          public static readonly DisplayLayer Menu = new DisplayLayer (10, "Menu
66          /// <summary>
67          /// Steht fr die Ebene in der Meneintrge gezeichnet werden. Menein
68          /// </summary>
69          public static readonly DisplayLayer MenuItem = new DisplayLayer (20, "
70          /// <summary>
71          /// Zum Anzeigen zustzlicher Informationen bei der (Weiter-)Entwicklu
72          /// </summary>
73          public static readonly DisplayLayer Overlay = new DisplayLayer (300, "
74          /// <summary>
75          /// Die Maus ist das Hauptinteraktionswerkzeug, welches der Spieler
76          /// stndig verwendet. Daher muss die Maus bei der Interaktion immer
77          /// im Vordergrund sein. Cursor steht fr die vorderste Ebene.
78          /// </summary>
79          public static readonly DisplayLayer Cursor = new DisplayLayer (500, "C
80
81          public static readonly DisplayLayer[] Values = {
82              None, Background, GameWorld, ScreenUI, Dialog, Menu, MenuItem, Ove
83          };
84
85          public int Index { get; private set; }
86

```

```

1001      87      public string Description { get; private set; }
           88
           89      private DisplayLayer (int index, string description)
18         90      {
18         91          Index = index;
18         92          Description = description;
18         93      }
           94
324        95      private DisplayLayer (DisplayLayer layer1, DisplayLayer layer2)
324        96      {
324        97          Index = layer1.Index + layer2.Index;
324        98          Description = layer1.Description + "+" + layer2.Description;
324        99      }
           100
           101      [ExcludeFromCodeCoverageAttribute]
           102      public override string ToString ()
           103      {
           104          return Description;
           105      }
           106
           107      public static DisplayLayer operator + (DisplayLayer layer1, DisplayLay
243       108      {
243       109          return new DisplayLayer (layer1, layer2);
243       110      }
           111
           112      public static DisplayLayer operator + (DisplayLayer layer, Widget widg
81        113      {
81        114          return new DisplayLayer (widget.Index, layer);
81        115      }
           116
           117      public static DisplayLayer operator * (DisplayLayer layer, int i)
9         118      {
9         119          return new DisplayLayer (layer.Index * i, "(" + layer + "*" + i +
9         120      }
           121
           122      public static bool operator == (DisplayLayer a, DisplayLayer b)
27        123      {
           124          // If both are null, or both are same instance, return true.
36        125          if (System.Object.ReferenceEquals (a, b)) {
9         126              return true;
           127          }
           128
           129          // If one is null, but not both, return false.
36        130          if (((object)a == null) || ((object)b == null)) {
18        131              return false;
           132          }
           133
           134          // Return true if the fields match:
0         135          return a.Index == b.Index;
27        136      }
           137
           138      public static bool operator != (DisplayLayer d1, DisplayLayer d2)
27        139      {
27        140          return !(d1 == d2);
27        141      }
           142
           143      public bool Equals (DisplayLayer other)
27        144      {
27        145          return other != null && Index == other.Index;
27        146      }
           147

```

```
148         public override bool Equals (object other)
149         {
150             return other != null && Equals (other as DisplayLayer);
151         }
152
153         public static implicit operator string (DisplayLayer layer)
154         {
155             return layer.Description;
156         }
157
158         public static implicit operator int (DisplayLayer layer)
159         {
160             return layer.Index;
161         }
162
163         [ExcludeFromCodeCoverageAttribute]
164         public override int GetHashCode ()
165         {
166             return Description.GetHashCode ();
167         }
168     }
169 }
```

Knot3.Framework.Core.World

Summary

Class:	Knot3.Framework.Core.World
Assembly:	Knot3.Framework
File(s):	ers\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Core\World.cs
Coverage:	15.9%
Covered lines:	23
Uncovered lines:	121
Coverable lines:	144
Total lines:	383

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	5	80	66.67
.ctor(...)	2	0	0
DefaultEffect(...)	1	0	0
Add(...)	2	0	0
Remove(...)	2	0	0
System.Collections.I	1	0	0
MoveNext()	5	0	0
MoveNext()	8	0	0
MoveNext()	7	0	0
MoveNext()	9	0	0
MoveNext()	9	0	0

File(s)

ers\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Core\World.cs

#	Line	Coverage
	1	/*
	2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
	3	* Gerd Augsburg, Christina Erler, Daniel Warzel
	4	*
	5	* This source code file is part of Knot3. Copying, redistribution and
	6	* use of the source code in this file in source and binary forms,
	7	* with or without modification, are permitted provided that the conditions
	8	* of the MIT license are met:
	9	*
	10	* Permission is hereby granted, free of charge, to any person obtaining a c
	11	* of this software and associated documentation files (the "Software"), to
	12	* in the Software without restriction, including without limitation the rig
	13	* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
	14	* copies of the Software, and to permit persons to whom the Software is
	15	* furnished to do so, subject to the following conditions:
	16	*
	17	* The above copyright notice and this permission notice shall be included i
	18	* copies or substantial portions of the Software.
	19	*
	20	* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
	21	* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
	22	* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
	23	* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
	24	* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
	25	* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
	26	* SOFTWARE.

```

27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31  using System;
32  using System.Collections;
33  using System.Collections.Generic;
34  using System.Diagnostics.CodeAnalysis;
35  using System.Linq;
36
37  using Microsoft.Xna.Framework;
38  using Microsoft.Xna.Framework.Graphics;
39
40  using Knot3.Framework.Effects;
41  using Knot3.Framework.Math;
42  using Knot3.Framework.Models;
43  using Knot3.Framework.Storage;
44
45  namespace Knot3.Framework.Core
46  {
47      /// <summary>
48      /// Repräsentiert eine Spielwelt, in der sich 3D-Modelle befinden und geze
49      /// </summary>
50      public sealed class World : DrawableScreenComponent, IEnumerable<IGameObje
51      {
52          /// <summary>
53          /// Die Kamera dieser Spielwelt.
54          /// </summary>
55          public Camera Camera
56          {
57              get {
58                  return _camera;
59              }
60              set {
61                  _camera = value;
62                  useInternalCamera = false;
63              }
64          }
65
66          private Camera _camera;
67          private bool useInternalCamera = true;
68
69          /// <summary>
70          /// Die Liste von Spielobjekten.
71          /// </summary>
72          public HashSet<IGameObject> Objects { get; set; }
73
74          private IGameObject _selectedObject;
75
76          /// <summary>
77          /// Das aktuell ausgewählte Spielobjekt.
78          /// </summary>
79          public IGameObject SelectedObject
80          {
81              get {
82                  return _selectedObject;
83              }
84              set {
85                  if (_selectedObject != value) {
86                      _selectedObject = value;
87                      SelectionChanged (_selectedObject);

```

```

0 88             Redraw = true;
0 89         }
0 90     }
    91 }
    92
    93     /// <summary>
    94     /// Der Abstand von der aktuellen Kameraposition und dem ausgewhlten
    95     /// Gibt 0 zurck, wenn kein Spielobjekt ausgewhlt ist.
    96     /// </summary>
    97     public float SelectedObjectDistance
    98     {
0 99         get {
0 100             if (SelectedObject != null) {
0 101                 Vector3 toTarget = SelectedObject.Center () - Camera.Posit
0 102                 return toTarget.Length ();
0 103             }
0 104             else {
0 105                 return 0;
0 106             }
0 107         }
0 108     }
    109
    110     /// <summary>
    111     /// Der aktuell angewendete Rendereffekt.
    112     /// </summary>
4 113     public IRenderEffect CurrentEffect { get; set; }
    114
    115     /// <summary>
    116     /// Wird ausgelst, wenn das selektierte Spielobjekt gendert wurde.
    117     /// </summary>
4 118     public Action<IGameObject> SelectionChanged = (o) => {};
    119
    120     /// <summary>
    121     /// Gibt an, ob die Spielwelt im folgenden Frame neu gezeichnet wird
    122     /// oder nur der letzte Frame wiedergegeben wird.
    123     /// </summary>
0 124     public bool Redraw { get; set; }
    125
    126     /// <summary>
    127     /// Wird ausgelst, wenn die Spielwelt neu gezeichnet wird.
    128     /// </summary>
4 129     public Action OnRedraw = () => {};
    130
    131     /// <summary>
    132     /// Die Ausmae der Welt auf dem Screen.
    133     /// </summary>
4 134     public Bounds Bounds { get; private set; }
    135
    136     /// <summary>
    137     /// Erstellt eine neue Spielwelt im angegebenen Spielzustand, dem ange
    138     /// angegebenen Bounds-Objekt.
    139     /// </summary>
4 140     public World (IScreen screen, DisplayLayer drawOrder, IRenderEffect ef
    141     : base (screen, drawOrder)
4 142     {
    143         // die Kamera fr diese Spielwelt
4 144         _camera = new Camera (screen, this);
    145
    146         // die Liste der Spielobjekte
4 147         Objects = new HashSet<IGameObject> ();
    148

```

```

149         // der Rendereffekt
150         CurrentEffect = effect;
151
152         // die relative Standard-Position und Gre
153         Bounds = bounds;
154
155         if (Screen.Game != null) {
156             Screen.Game.FullScreenChanged += () => viewportCache.Clear ();
157         }
158     }
159
160     /// <summary>
161     /// Erstellt eine neue Spielwelt im angegebenen Spielzustand und dem
162     /// angegebenen Bounds-Objekt.
163     /// Als Rendereffekt wird der in der Konfigurationsdatei festgelegte S
164     /// Falls whrend der Existenz dieses Objektes der Standardeffekt gen
165     /// wird der neue Effekt bernommen.
166     /// </summary>
167     public World (IScreen screen, DisplayLayer drawOrder, Bounds bounds)
168     : this (screen: screen, drawOrder: drawOrder, effect: DefaultEffect (s
169     {
170         RenderEffectLibrary.RenderEffectChanged += (newEffectName, time) =
171             CurrentEffect = RenderEffectLibrary.CreateEffect (screen: scre
172     };
173     }
174
175     private static IRenderEffect DefaultEffect (IScreen screen)
176     {
177         // suche den eingestellten Standardeffekt heraus
178         string effectName = Config.Default ["video", "knot-shader", "defau
179         IRenderEffect effect = RenderEffectLibrary.CreateEffect (screen: s
180         return effect;
181     }
182
183     /// <summary>
184     /// Fge ein Spielobjekt zur Spielwelt hinzu.
185     /// </summary>
186     public void Add (IGameObject obj)
187     {
188         if (obj != null) {
189             Objects.Add (obj);
190             obj.World = this;
191         }
192         Redraw = true;
193     }
194
195     /// <summary>
196     /// Entferne ein Spielobjekt aus der Spielwelt.
197     /// </summary>
198     public void Remove (IGameObject obj)
199     {
200         if (obj != null) {
201             Objects.Remove (obj);
202         }
203         Redraw = true;
204     }
205
206     /// <summary>
207     /// Ruft auf allen Spielobjekten die Update ()-Methode auf.
208     /// </summary>
209     [ExcludeFromCodeCoverageAttribute]

```



```

210     public override void Update (GameTime time)
211     {
212         if (!Config.Default ["video", "selectiveRendering", false]) {
213             Redraw = true;
214         }
215         if (!Screen.PostProcessingEffect.SelectiveRendering) {
216             Redraw = true;
217         }
218
219         // run the update method on all game objects
220         foreach (IGameObject obj in Objects) {
221             obj.Update (time);
222         }
223     }
224
225     private Dictionary<Point,Dictionary<Vector4, Viewport>> viewportCache
226         = new Dictionary<Point,Dictionary<Vector4, Viewport>> ();
227
228     /// <summary>
229     /// Gibt den aktuellen Viewport zuruck.
230     /// </summary>
231     public Viewport Viewport
232     {
233         get {
234             // when we have a graphics device
235             if (Screen.GraphicsDevice != null) {
236                 PresentationParameters pp = Screen.GraphicsDevice.Presenta
237                 Point resolution = new Point (pp.BackBufferWidth, pp.BackB
238                 Vector4 key = Bounds.Vector4;
239                 if (!viewportCache.ContainsKey (resolution)) {
240                     viewportCache [resolution] = new Dictionary<Vector4, V
241                 }
242                 if (!viewportCache [resolution].ContainsKey (key)) {
243                     Rectangle subScreen = Bounds.Rectangle;
244                     viewportCache [resolution] [key] = new Viewport (subSc
245                         MinDepth = 0,
246                         MaxDepth = 1
247                 };
248             }
249             return viewportCache [resolution] [key];
250         }
251         // for unit tests
252         else {
253             return Screen.Viewport;
254         }
255     }
256 }
257
258 /// <summary>
259 /// Ruft auf allen Spielobjekten die Draw ()-Methode auf.
260 /// </summary>
261 [ExcludeFromCodeCoverageAttribute]
262 public override void Draw (GameTime time)
263 {
264     if (Redraw) {
265         OnRedraw ();
266         Redraw = false;
267
268         //Screen.BackgroundColor = CurrentEffect is CelShadingEffect ?
269
270         // begin the knot render effect

```

```

271         CurrentEffect.Begin (time);
272
273         foreach (IGameObject obj in Objects) {
274             obj.World = this;
275             obj.Draw (time);
276         }
277
278         // end of the knot render effect
279         CurrentEffect.End (time);
280     }
281     else {
282         CurrentEffect.DrawLastFrame (time);
283     }
284 }
285
286 /// <summary>
287 /// Liefert einen Enumerator ber die Spielobjekte dieser Spielwelt.
288 /// [returntype=IEnumerator<IGameObject>]
289 /// </summary>
290 public IEnumerator<IGameObject> GetEnumerator ()
0 291 {
0 292     foreach (IGameObject obj in flat (Objects)) {
0 293         yield return obj;
0 294     }
0 295 }
296
297 private IEnumerable<IGameObject> flat (IEnumerable<IGameObject> enumer
0 298 {
0 299     foreach (IGameObject obj in enumerable) {
0 300         if (obj is IEnumerable<IGameObject>) {
0 301             foreach (IGameObject subobj in flat (obj as IEnumerable<IG
0 302                 yield return subobj;
0 303             }
0 304         }
0 305         else {
0 306             yield return obj;
0 307         }
0 308     }
0 309 }
310 // Explicit interface implementation for nongeneric interface
311 IEnumerator IEnumerable.GetEnumerator ()
0 312 {
0 313     return GetEnumerator (); // Just return the generic version
0 314 }
315
316 public override IEnumerable<IScreenComponent> SubComponents (GameTime
0 317 {
0 318     foreach (DrawableScreenComponent component in base.SubComponents (
0 319         yield return component;
0 320     }
0 321     if (useInternalCamera) {
0 322         yield return Camera;
0 323     }
0 324 }
325
326 /// <summary>
327 /// Gibt einen Iterator ber alle Spielobjekte zurck, der so sortiert
328 /// Spielobjekte, die der angegebenen 2D-Position am nchsten sind, am
329 /// Dazu wird die 2D-Position in eine 3D-Position konvertiert.
330 /// </summary>
331 public IEnumerable<IGameObject> FindNearestObjects (ScreenPoint nearTo

```

```

0 332      {
0 333          Dictionary<float, IGameObject> distances = new Dictionary<float, I
0 334      foreach (IGameObject obj in this) {
0 335          if (obj.Info.IsSelectable) {
0 336              // Berechne aus der angegebenen 2D-Position eine 3D-Positi
0 337              Vector3 position3D = Camera.To3D (
0 338                  position: nearTo,
0 339                  nearTo: obj.Center ()
0 340              );
0 341              // Berechne die Distanz zwischen 3D-Mausposition und dem S
0 342              float distance = System.Math.Abs ((position3D - obj.Center
0 343              distances [distance] = obj;
0 344          }
0 345      }
0 346      if (distances.Count > 0) {
0 347          IEnumerable<float> sorted = distances.Keys.OrderBy (k => k);
0 348          foreach (float where in sorted) {
0 349              yield return distances [where];
0 350              // Log.Debug ("where=", where, " = ", distances [where].Ce
0 351          }
0 352      }
0 353      else {
0 354          yield break;
0 355      }
0 356  }
0 357
0 358  /// <summary>
0 359  /// Gibt einen Iterator ber alle Spielobjekte zurck, der so sortiert
0 360  /// Spielobjekte, die der angegebenen 3D-Position am nchsten sind, am
0 361  /// </summary>
0 362  public IEnumerable<IGameObject> FindNearestObjects (Vector3 nearTo)
0 363  {
0 364      Dictionary<float, IGameObject> distances = new Dictionary<float, I
0 365      foreach (IGameObject obj in this) {
0 366          if (obj.Info.IsSelectable) {
0 367              // Berechne die Distanz zwischen 3D-Mausposition und dem S
0 368              float distance = System.Math.Abs ((nearTo - obj.Center ())
0 369              distances [distance] = obj;
0 370          }
0 371      }
0 372      if (distances.Count > 0) {
0 373          IEnumerable<float> sorted = distances.Keys.OrderBy (k => k);
0 374          foreach (float where in sorted) {
0 375              yield return distances [where];
0 376          }
0 377      }
0 378      else {
0 379          yield break;
0 380      }
0 381  }
0 382  }
0 383  }

```

Knot3.Framework.Math.Angles3

Summary

Class:	Knot3.Framework.Math.Angles3
Assembly:	Knot3.Framework
File(s):	s\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\Angles3.cs
Coverage:	100%
Covered lines:	62
Uncovered lines:	0
Coverable lines:	62
Total lines:	192

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
.ctor(...)	1	100	100
FromDegrees(...)	1	100	100
ToDegrees(...)	1	100	100
Equals(...)	3	100	66.67
Equals(...)	1	100	100
op_Equality(...)	3	100	80
op_Inequality(...)	1	100	100
op_Addition(...)	1	100	100
op_UnaryNegation(...)	1	100	100
op_Subtraction(...)	1	100	100
op_Multiply(...)	1	100	100
op_Multiply(...)	1	100	100
op_Multiply(...)	1	100	100
op_Division(...)	1	100	100
op_Division(...)	1	100	100

File(s)

s\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\Angles3.cs

#	Line	Coverage
	1	/*
	2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
	3	* Gerd Augsburg, Christina Erler, Daniel Warzel
	4	*
	5	* This source code file is part of Knot3. Copying, redistribution and
	6	* use of the source code in this file in source and binary forms,
	7	* with or without modification, are permitted provided that the conditions
	8	* of the MIT license are met:
	9	*
	10	* Permission is hereby granted, free of charge, to any person obtaining a c
	11	* of this software and associated documentation files (the "Software"), to
	12	* in the Software without restriction, including without limitation the rig
	13	* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
	14	* copies of the Software, and to permit persons to whom the Software is
	15	* furnished to do so, subject to the following conditions:
	16	*
	17	* The above copyright notice and this permission notice shall be included i
	18	* copies or substantial portions of the Software.
	19	*
	20	* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
	21	* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,

```

22  *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Diagnostics.CodeAnalysis;
33
34 using Microsoft.Xna.Framework;
35
36 namespace Knot3.Framework.Math
37 {
38     /// <summary>
39     /// Diese Klasse repräsentiert die Rollwinkel der drei Achsen X, Y und Z.
40     /// Sie bietet Möglichkeit vordefinierte Winkelwerte zu verwenden, z.B. st
41     /// Die Umwandlung zwischen verschiedenen Winkelmaßen wie Grad- und Bogenm
42     /// </summary>
43     public sealed class Angles3 : IEquatable<Angles3>
44     {
45         /// <summary>
46         /// Der Winkel im Bogenmaß für das Rollen um die X-Achse. Siehe statis
47         /// </summary>
158 48         public float X { get; set; }
49
50         /// <summary>
51         /// Der Winkel im Bogenmaß für das Rollen um die Y-Achse. Siehe statis
52         /// </summary>
154 53         public float Y { get; set; }
54
55         /// <summary>
56         /// Der Winkel im Bogenmaß für das Rollen um die Z-Achse. Siehe statis
57         /// </summary>
154 58         public float Z { get; set; }
59
60         /// <summary>
61         /// Eine statische Eigenschaft mit dem Wert X = 0, Y = 0, Z = 0.
62         /// </summary>
63         public static Angles3 Zero
64         {
54 65             get { return new Angles3 (0f, 0f, 0f); }
66         }
67
68         /// <summary>
69         /// Konstruiert ein neues Angles3-Objekt mit drei gegebenen Winkeln im
70         /// </summary>
91 71         public Angles3 (float x, float y, float z)
91 72         {
91 73             X = x;
91 74             Y = y;
91 75             Z = z;
91 76         }
77
5 78         public Angles3 (Vector3 v)
5 79         {
5 80             X = v.X;
5 81             Y = v.Y;
5 82             Z = v.Z;

```

```

5      83      }
      84
      85      ///

```

```

1 144     }
145
146     public static Angles3 operator - (Angles3 value1, Angles3 value2)
1 147     {
1 148         return new Angles3 (value1.X - value2.X, value1.Y - value2.Y, valu
1 149     }
150
151     public static Angles3 operator * (Angles3 value1, Angles3 value2)
1 152     {
1 153         return new Angles3 (value1.X * value2.X, value1.Y * value2.Y, valu
1 154     }
155
156     public static Angles3 operator * (Angles3 value, float scaleFactor)
1 157     {
1 158         return new Angles3 (value.X * scaleFactor, value.Y * scaleFactor,
1 159     }
160
161     public static Angles3 operator * (float scaleFactor, Angles3 value)
1 162     {
1 163         return new Angles3 (value.X * scaleFactor, value.Y * scaleFactor,
1 164     }
165
166     public static Angles3 operator / (Angles3 value1, Angles3 value2)
1 167     {
1 168         return new Angles3 (value1.X / value2.X, value1.Y / value2.Y, valu
1 169     }
170
171     public static Angles3 operator / (Angles3 value, float divider)
1 172     {
1 173         float scaleFactor = 1 / divider;
1 174         return new Angles3 (value.X * scaleFactor, value.Y * scaleFactor,
1 175     }
176
177     [ExcludeFromCodeCoverageAttribute]
178     public override string ToString ()
179     {
180         float x, y, z;
181         ToDegrees (out x, out y, out z);
182
183         return  "Angles3 ("
184                 + x.ToString ()
185                 + ","
186                 + y.ToString ()
187                 + ","
188                 + z.ToString ()
189                 + ")";
190     }
191 }
192 }

```

Knot3.Framework.Math.BoundingCylinder

Summary

Class:	Knot3.Framework.Math.BoundingCylinder
Assembly:	Knot3.Framework
File(s):	Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\BoundingCylinder.cs
Coverage:	90.4%
Covered lines:	19
Uncovered lines:	2
Coverable lines:	21
Total lines:	85

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
op_Equality(...)	4	55.56	57.14
op_Inequality(...)	1	100	100
Equals(...)	5	100	77.78
Equals(...)	2	100	100

File(s)

Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\BoundingCylinder.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Diagnostics.CodeAnalysis;

```



```

33
34 using Microsoft.Xna.Framework;
35
36 namespace Knot3.Framework.Math
37 {
38     public struct BoundingCylinder : IEquatable<BoundingCylinder> {
39         public Vector3 SideA;
40         public Vector3 SideB;
41         public float Radius;
42
43         public BoundingCylinder (Vector3 sideA, Vector3 sideB, float radius)
44         {
45             SideA = sideA;
46             SideB = sideB;
47             Radius = radius;
48         }
49
50         public static bool operator == (BoundingCylinder a, BoundingCylinder b)
51         {
52             if (System.Object.ReferenceEquals (a, b)) {
53                 return true;
54             }
55             if (((object)a == null) || ((object)b == null)) {
56                 return false;
57             }
58             return a.Equals (b);
59         }
60
61         public static bool operator != (BoundingCylinder a, BoundingCylinder b)
62         {
63             return !(a == b);
64         }
65
66         public bool Equals (BoundingCylinder other)
67         {
68             return ((SideA == other.SideA && SideB == other.SideB)
69                 || (SideA == other.SideB && SideB == other.SideA))
70                 && Radius == other.Radius;
71         }
72
73         public override bool Equals (object other)
74         {
75             return other != null && Equals ((BoundingCylinder)other);
76         }
77
78         [ExcludeFromCodeCoverageAttribute]
79         public override int GetHashCode ()
80         {
81             // irgendwas mglichst eindeutiges
82             return (Radius * (SideA + SideB)).GetHashCode ();
83         }
84     }
85 }

```

Knot3.Framework.Math.Bounds

Summary

Class:	Knot3.Framework.Math.Bounds
Assembly:	Knot3.Framework
File(s):	rs\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\Bounds.cs
Coverage:	94.5%
Covered lines:	87
Uncovered lines:	5
Coverable lines:	92
Total lines:	229

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
.ctor(...)	1	100	100
.ctor(...)	1	100	100
Contains(...)	1	100	100
Contains(...)	1	100	100
Zero(...)	1	100	100
FromLeft(...)	1	100	100
FromRight(...)	2	100	66.67
FromTop(...)	1	100	100
FromBottom(...)	2	100	66.67
FromLeft(...)	1	100	100
FromRight(...)	1	100	100
FromTop(...)	1	100	100
FromBottom(...)	1	100	100
In(...)	1	100	100
Grow(...)	1	100	100
Shrink(...)	1	100	100
Grow(...)	1	100	100
Shrink(...)	1	100	100
op_Implicit(...)	1	0	0

File(s)

rs\Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\Bounds.cs

```

#   Line Coverage
    1  /*
    2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
    4   *
    5   * This source code file is part of Knot3. Copying, redistribution and
    6   * use of the source code in this file in source and binary forms,
    7   * with or without modification, are permitted provided that the conditions
    8   * of the MIT license are met:
    9   *
   10   * Permission is hereby granted, free of charge, to any person obtaining a c
   11   * of this software and associated documentation files (the "Software"), to
   12   * in the Software without restriction, including without limitation the rig
   13   * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14   * copies of the Software, and to permit persons to whom the Software is
   15   * furnished to do so, subject to the following conditions:
   16   *
   17   * The above copyright notice and this permission notice shall be included i

```

```

18  *   copies or substantial portions of the Software.
19  *
20  *   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Diagnostics.CodeAnalysis;
33
34 using Microsoft.Xna.Framework;
35
36 using Knot3.Framework.Core;
37
38 namespace Knot3.Framework.Math
39 {
40     public class Bounds
41     {
42         /// <summary>
43         /// Die von der Auflsung unabhnigige Position in Prozent.
44         /// </summary>
45         public ScreenPoint Position
46         {
132         get { return _position; }
174         set { _position.Assign (value); }
48         }
49
50         private ScreenPoint _position;
51
52         /// <summary>
53         /// Die von der Auflsung unabhnigige Gre in Prozent.
54         /// </summary>
55         public ScreenPoint Size
56         {
57         get { return _size; }
58         set { _size.Assign (value); }
59         }
60
61         private ScreenPoint _size;
62
63         /// <summary>
64         /// Der von der Auflsung unabhnigige Abstand in Prozent.
65         /// </summary>
66         public ScreenPoint Padding
67         {
30         get { return _padding; }
30         set { _padding.Assign (value); }
69         }
70
71         private ScreenPoint _padding;
72
73         /// <summary>
74         /// Gibt ein auf die Auflsujng skaliertes Rechteck zurck, das in den
75         /// </summary>
76         public Rectangle Rectangle

```

```

79      {
80          get {
81              Point pos = Position.Absolute;
82              Point size = Size.Absolute;
83              return new Rectangle (pos.X, pos.Y, size.X, size.Y);
84          }
85      }
86
87      public Vector4 Vector4
88      {
89          get {
90              Point pos = Position.Absolute;
91              Point size = Size.Absolute;
92              return new Vector4 (pos.X, pos.Y, size.X, size.Y);
93          }
94      }
95
96      public Bounds (ScreenPoint position, ScreenPoint size, ScreenPoint pad
97      {
98          _position = position;
99          _size = size;
100         _padding = padding;
101     }
102
103     public Bounds (ScreenPoint position, ScreenPoint size)
104     {
105         _position = position;
106         _size = size;
107         _padding = new ScreenPoint (position.Screen, Vector2.Zero);
108     }
109
110     public Bounds (IScreen screen, float relX, float relY, float relWidth,
111     {
112         _position = new ScreenPoint (screen, relX, relY);
113         _size = new ScreenPoint (screen, relWidth, relHeight);
114         _padding = new ScreenPoint (screen, Vector2.Zero);
115     }
116
117     public bool Contains (Point point)
118     {
119         return Rectangle.Contains (point);
120     }
121
122     public bool Contains (ScreenPoint point)
123     {
124         return Rectangle.Contains ((Point)point);
125     }
126
127     public static Bounds Zero (IScreen screen)
128     {
129         return new Bounds (
130             position: ScreenPoint.Zero (screen),
131             size: ScreenPoint.Zero (screen),
132             padding: ScreenPoint.Zero (screen)
133         );
134     }
135
136     public Bounds FromLeft (Func<float> percent)
137     {
138         return new Bounds (
139             position: Position,

```

```

4 140             size: new ScreenPoint (Size.Screen, () => Size.Relative
141             padding: Padding
142             );
3 143     }
144
145     public Bounds FromRight (Func<float> percent)
1 146     {
1 147         return new Bounds (
4 148             position: Position + new ScreenPoint (Size.Screen, () =
4 149             size: new ScreenPoint (Size.Screen, () => Size.Relative
150             padding: Padding
151             );
1 152     }
153
154     public Bounds FromTop (Func<float> percent)
3 155     {
3 156         return new Bounds (
157             position: Position,
4 158             size: new ScreenPoint (Size.Screen, () => Size.Relative
159             padding: Padding
160             );
3 161     }
162
163     public Bounds FromBottom (Func<float> percent)
1 164     {
1 165         return new Bounds (
4 166             position: Position + new ScreenPoint (Size.Screen, () =
4 167             size: new ScreenPoint (Size.Screen, () => Size.Relative
168             padding: Padding
169             );
1 170     }
171
172     public Bounds FromLeft (float percent)
3 173     {
5 174         return FromLeft (() => percent);
3 175     }
176
177     public Bounds FromRight (float percent)
1 178     {
5 179         return FromRight (() => percent);
1 180     }
181
182     public Bounds FromTop (float percent)
3 183     {
5 184         return FromTop (() => percent);
3 185     }
186
187     public Bounds FromBottom (float percent)
1 188     {
5 189         return FromBottom (() => percent);
1 190     }
191
192     public Bounds In (Bounds container)
1 193     {
1 194         return new Bounds (Position + container.Position, Size, Padding);
1 195     }
196
197     public Bounds Grow (int x, int y)
4 198     {
4 199         ScreenPoint diff = ScreenPoint.FromAbsolute (x, y, Position.Screen
4 200         return new Bounds (Position - diff, Size + diff * 2);

```

```

4    201      }
      202
      203      public Bounds Shrink (int x, int y)
1    204      {
1    205          return Grow (-x, -y);
1    206      }
      207
      208      public Bounds Grow (int xy)
1    209      {
1    210          return Grow (xy, xy);
1    211      }
      212
      213      public Bounds Shrink (int xy)
1    214      {
1    215          return Grow (-xy, -xy);
1    216      }
      217
      218      public static implicit operator Rectangle (Bounds bounds)
0    219      {
0    220          return bounds.Rectangle;
0    221      }
      222
      223      [ExcludeFromCodeCoverageAttribute]
      224      public override string ToString ()
      225      {
      226          return "(" + Position.Relative.X + "x" + Position.Relative.Y + ","
      227      }
      228      }
      229  }
```

Knot3.Framework.Math.RayExtensions

Summary

Class:	Knot3.Framework.Math.RayExtensions
Assembly:	Knot3.Framework
File(s):	al\Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\RayExtensions.cs
Coverage:	68%
Covered lines:	34
Uncovered lines:	16
Coverable lines:	50
Total lines:	142

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
Intersects(...)	21	61.90	53.66

File(s)

al\Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\RayExtensions.cs

#	Line	Coverage
1	/*	
2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,	
3	* Gerd Augsburg, Christina Erler, Daniel Warzel	
4	*	
5	* This source code file is part of Knot3. Copying, redistribution and	
6	* use of the source code in this file in source and binary forms,	
7	* with or without modification, are permitted provided that the conditions	
8	* of the MIT license are met:	
9	*	
10	* Permission is hereby granted, free of charge, to any person obtaining a c	
11	* of this software and associated documentation files (the "Software"), to	
12	* in the Software without restriction, including without limitation the rig	
13	* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell	
14	* copies of the Software, and to permit persons to whom the Software is	
15	* furnished to do so, subject to the following conditions:	
16	*	
17	* The above copyright notice and this permission notice shall be included i	
18	* copies or substantial portions of the Software.	
19	*	
20	* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O	
21	* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,	
22	* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T	
23	* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER	
24	* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F	
25	* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN	
26	* SOFTWARE.	
27	*	
28	* See the LICENSE file for full license details of the Knot3 project.	
29	*/	
30		
31	using System.Diagnostics.CodeAnalysis;	
32		
33	using Microsoft.Xna.Framework;	
34		
35	namespace Knot3.Framework.Math	
36	{	
37	public static class RayExtensions	

```

38      {
39          public static float? Intersects (this Ray ray, BoundingCylinder cylind
40      {
41          Vector3 dirAB = cylinder.SideB - cylinder.SideA;
42          // Raystart innerhalb des Zylinders
43          if (Vector3.Cross ((ray.Position - cylinder.SideA), ray.Direction)
44              return 0.0f;
45      }
46          Vector3 perpendicular = Vector3.Cross (dirAB, ray.Direction);
47          // if !(Ray Parallel zum Zylinder)
48          if (perpendicular.Length () > 0.0000001f) {
49              perpendicular.Normalize ();
50              if (Vector3.Dot (perpendicular, ray.Direction) > 0) {
51                  perpendicular = -perpendicular;
52              }
53              Vector3 perpendicular2 = Vector3.Cross (dirAB, perpendicular);
54              // If (Ray Senkrecht zum Zylinder)
55              if (perpendicular2.Length () < 0.0000001f) {
56                  if (Vector3.Dot (dirAB, ray.Position - cylinder.SideA) < 0
57                      return null;
58                  }
59                  float? result = Vector3.Cross ((ray.Position - cylinder.Si
60                  if (result < 0) {
61                      result = 0.0f;
62                  }
63                  return result;
64              }
65              if (Vector3.Dot (perpendicular2, ray.Direction) > 0) {
66                  perpendicular2 = -perpendicular2;
67              }
68              perpendicular2.Normalize ();
69              float minDist = System.Math.Abs (Vector3.Dot (cylinder.SideA -
70              if (minDist > cylinder.Radius) {
71                  return null;
72              }
73              Vector3 plainNorm = perpendicular * minDist + (float)System.Ma
74              plainNorm.Normalize ();
75              float? other_result = ray.Intersects (new Plane (plainNorm, Ve
76              if (other_result == null) {
77                  return null;
78              }
79              Vector3 cutA = ray.Position + ray.Direction * (float)other_res
80              Vector3 cutB = ray.Position + ray.Direction * (float)other_res
81              if (Vector3.Dot (dirAB, cutA) > 0 && Vector3.Dot (-dirAB, cutB
82                  return other_result;
83              }
84          }
85          if (Vector3.Distance (ray.Position, cylinder.SideA) < Vector3.Dist
86              dirAB.Normalize ();
87              float? result = ray.Intersects (new Plane (dirAB, Vector3.Dot
88              if (result == null || Vector3.Distance (ray.Position + ray.Dir
89                  return null;
90              }
91              return result;
92          }
93          else {
94              dirAB.Normalize ();
95              dirAB = -dirAB;
96              float? result = ray.Intersects (new Plane (dirAB, Vector3.Dot
97              if (result == null || Vector3.Distance (ray.Position + ray.Dir
98                  return null;

```



```

    99         }
0 100     return result;
101     }
102     /*
103     Vector3 diffA = capsule.CornerA - ray.Position;
104     Vector3 diffB = capsule.CornerB - ray.Position;
105     float diffASquared = diffA.LengthSquared ();
106     float diffBSquared = diffB.LengthSquared ();
107     float radiusSquared = capsule.Radius * capsule.Radius;
108     // Startpunkt innerhalb der Eckkugeln
109     if (diffASquared < radiusSquared || diffBSquared < radiusSquared)
110     {
111         return 0.0f;
112     }
113     Vector3 dirBA = (capsule.CornerA - capsule.CornerB);
114     float distAlongAB = Vector3.Dot (diffA, dirBA) / dirBA.Length ();
115     // Startpunkt innerhalb des Zylinders
116     if (distAlongAB > 0 && distAlongAB < dirBA.Length () && (distAlong
117     {
118         return 0.0f;
119     }
120     float distAlongRayA = Vector3.Dot (ray.Direction, diffA);
121     float distAlongRayB = Vector3.Dot (ray.Direction, diffB);
122     // Richtung geht weg von der Kapsel
123     if (distAlongRayA < 0 && distAlongRayB < 0)
124         return null;
125     Vector3 perpendicular = Vector3.Cross (ray.Direction, dirBA);
126     perpendicular.Normalize ();
127     float minDistance = Math.Abs (Vector3.Dot (diffA, perpendicular));
128     // Kommt selbst der Geraden nie nahe genug.
129     if (minDistance > capsule.Radius)
130     {
131         return null;
132     }
133     Vector3 normDirAB = -dirBA;
134     normDirAB.Normalize ();
135     Vector3 extensionToBase = Vector3.Cross (normDirAB, perpendicular)
136     extensionToBase.Normalize ();
137     Matrix transformation = new Matrix (normDirAB.X, normDirAB.Y, norm
138     transformation = Matrix.Invert (transformation);
139     */
10 140     }
141     }
142 }

```

Knot3.Framework.Math.ScreenPoint

Summary

Class:	Knot3.Framework.Math.ScreenPoint
Assembly:	Knot3.Framework
File(s):	scal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\ScreenPoint.cs
Coverage:	51.8%
Covered lines:	69
Uncovered lines:	64
Coverable lines:	133
Total lines:	278

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
.ctor(...)	1	100	100
.ctor(...)	1	100	100
.ctor(...)	2	100	66.67
.ctor(...)	1	0	0
Assign(...)	1	100	100
FromAbsolute(...)	1	100	100
FromAbsolute(...)	1	0	0
Zero(...)	1	100	100
TopLeft(...)	1	0	0
BottomRight(...)	1	0	0
Centered(...)	1	0	0
op_Implicit(...)	1	0	0
op_Implicit(...)	1	0	0
op_Implicit(...)	1	100	100
op_Implicit(...)	1	0	0
op_Multiply(...)	1	100	100
op_Multiply(...)	1	0	0
op_Division(...)	1	0	0
op_Addition(...)	1	100	100
op_Subtraction(...)	1	100	100
ScaleX(...)	1	0	0
ScaleY(...)	1	0	0
op_Equality(...)	4	66.67	57.14
op_Inequality(...)	1	100	100
Equals(...)	3	100	60
Equals(...)	6	0	0

File(s)

scal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Math\ScreenPoint.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *   Permission is hereby granted, free of charge, to any person obtaining a c

```

```

11  *   of this software and associated documentation files (the "Software"), to
12  *   in the Software without restriction, including without limitation the rig
13  *   to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *   copies of the Software, and to permit persons to whom the Software is
15  *   furnished to do so, subject to the following conditions:
16  *
17  *   The above copyright notice and this permission notice shall be included i
18  *   copies or substantial portions of the Software.
19  *
20  *   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  *   See the LICENSE file for full license details of the Knot3 project.
29  */
30
31  using System;
32  using System.Diagnostics.CodeAnalysis;
33
34  using Microsoft.Xna.Framework;
35
36  using Knot3.Framework.Core;
37  using Knot3.Framework.Platform;
38  using Knot3.Framework.Utilities;
39
40  namespace Knot3.Framework.Math
41  {
42      public class ScreenPoint : IEquatable<ScreenPoint>
43      {
406      44          public IScreen Screen { get; private set; }
45
46          public Vector2 Relative
47          {
169      48              get {
169      49                  return RelativeFunc ();
169      50              }
321      51              set {
448      52                  RelativeFunc = () => value;
321      53              }
54          }
55
517      56          public Func<Vector2> RelativeFunc
57          {
58              set;
59              private get;
60          }
61
62          public Point Absolute
63          {
8      64              get {
8      65                  Vector2 scaled = Relative.Scale (Screen.Viewport);
8      66                  return new Point ((int)scaled.X, (int)scaled.Y);
8      67              }
68          }
69
70          public Vector2 AbsoluteVector
71          {

```

```

5      72      get {
5      73          return Relative.Scale (Screen.Viewport);
5      74      }
      75      }
      76
      77      public ScreenPoint OnlyX
      78      {
0      79          get {
0      80              return new ScreenPoint (Screen, () => new Vector2 (RelativeFun
0      81          }
      82      }
      83
      84      public ScreenPoint OnlyY
      85      {
0      86          get {
0      87              return new ScreenPoint (Screen, () => new Vector2 (0, Relative
0      88          }
      89      }
      90
      91      public ScreenPoint Const
      92      {
0      93          get {
0      94              return new ScreenPoint (Screen, Relative.X, Relative.Y);
0      95          }
      96      }
      97
0      98      public bool IsEmpty { get { return Relative.Length () == 0; } }
      99
15     100      public ScreenPoint (IScreen screen, Func<Vector2> func)
15     101      {
15     102          Screen = screen;
15     103          RelativeFunc = func;
15     104      }
      105
270    106      public ScreenPoint (IScreen screen, Vector2 vector)
270    107      {
270    108          Screen = screen;
270    109          Relative = vector;
270    110      }
      111
51     112      public ScreenPoint (IScreen screen, float x, float y)
51     113      {
51     114          Screen = screen;
51     115          Relative = new Vector2 (x, y);
51     116      }
      117
10     118      public ScreenPoint (IScreen screen, Func<float> x, Func<float> y)
10     119      {
10     120          Screen = screen;
22     121          RelativeFunc = () => new Vector2 (x (), y ());
10     122      }
      123
0      124      public ScreenPoint (IScreen screen, float xy)
0      125      {
0      126          Screen = screen;
0      127          Relative = new Vector2 (xy, xy);
0      128      }
      129
      130      public void Assign (ScreenPoint other)
1      131      {
1      132          Screen = other.Screen;

```

```

1 133         RelativeFunc = other.RelativeFunc;
1 134     }
    135
    136     public static ScreenPoint FromAbsolute (float x, float y, IScreen scre
4 137     {
4 138         return new ScreenPoint (screen, x / screen.Viewport.Width, y / scr
4 139     }
    140
    141     public static ScreenPoint FromAbsolute (Point point, IScreen screen)
0 142     {
0 143         return FromAbsolute ((float)point.X, (float)point.Y, screen);
0 144     }
    145
    146     public static ScreenPoint Zero (IScreen screen)
243 147     {
243 148         return new ScreenPoint (screen, Vector2.Zero);
243 149     }
    150
    151     public static ScreenPoint TopLeft (IScreen screen)
0 152     {
0 153         return new ScreenPoint (screen, Vector2.Zero);
0 154     }
    155
    156     public static ScreenPoint BottomRight (IScreen screen)
0 157     {
0 158         return new ScreenPoint (screen, Vector2.One);
0 159     }
    160
    161     public static ScreenPoint Centered (IScreen screen, Bounds sizeOf)
0 162     {
0 163         return new ScreenPoint (screen, () => (ScreenPoint.BottomRight (sc
0 164     }
    165
    166     public static implicit operator Vector2 (ScreenPoint point)
0 167     {
0 168         return point.Relative;
0 169     }
    170
    171     public static implicit operator Func<Vector2> (ScreenPoint point)
0 172     {
0 173         return point.RelativeFunc;
0 174     }
    175
    176     public static implicit operator Point (ScreenPoint point)
2 177     {
2 178         return point.Absolute;
2 179     }
    180
    181     public static implicit operator bool (ScreenPoint point)
0 182     {
0 183         return !point.IsEmpty;
0 184     }
    185
    186     [ExcludeFromCodeCoverageAttribute]
    187     public override string ToString ()
    188     {
    189         return "(" + Relative.X + "x" + Relative.Y + ")";
    190     }
    191
    192     public static ScreenPoint operator * (ScreenPoint a, float b)
4 193     {

```

```

12 194         return new ScreenPoint (a.Screen, () => a.Relative * b);
4 195     }
196
197     public static ScreenPoint operator * (ScreenPoint a, ScreenPoint b)
0 198     {
0 199         return new ScreenPoint (a.Screen, () => new Vector2 (a.Relative.X
0 200     }
201
202     public static ScreenPoint operator / (ScreenPoint a, float b)
0 203     {
0 204         return new ScreenPoint (a.Screen, () => a.Relative / b);
0 205     }
206
207     public static ScreenPoint operator + (ScreenPoint a, ScreenPoint b)
7 208     {
21 209         return new ScreenPoint (a.Screen, () => a.Relative + b.Relative);
7 210     }
211
212     public static ScreenPoint operator - (ScreenPoint a, ScreenPoint b)
4 213     {
12 214         return new ScreenPoint (a.Screen, () => a.Relative - b.Relative);
4 215     }
216
217     public ScreenPoint ScaleX (float percent)
0 218     {
0 219         return new ScreenPoint (Screen, () => new Vector2 (Relative.X * pe
0 220     }
221
222     public ScreenPoint ScaleY (float percent)
0 223     {
0 224         return new ScreenPoint (Screen, () => new Vector2 (Relative.X, Rel
0 225     }
226
227     public static bool operator == (ScreenPoint a, ScreenPoint b)
21 228     {
21 229         if (System.Object.ReferenceEquals (a, b)) {
0 230             return true;
231         }
42 232         if (((object)a == null) || ((object)b == null)) {
21 233             return false;
234         }
0 235         return a.Equals (b);
21 236     }
237
238     public static bool operator != (ScreenPoint d1, ScreenPoint d2)
21 239     {
21 240         return !(d1 == d2);
21 241     }
242
243     public bool Equals (ScreenPoint other)
21 244     {
21 245         float epsilon = 0.000001f;
246
21 247         return other != null && System.Math.Abs (Relative.X - other.Relati
21 248     }
249
250     public override bool Equals (object other)
0 251     {
0 252         if (other == null) {
0 253             return false;
254         }

```

```
0 255         else if (other is ScreenPoint) {
0 256             return Equals ((ScreenPoint)other);
257         }
0 258         else if (other is Vector2) {
0 259             return Relative.Equals ((Vector2)other);
260         }
0 261         else if (other is Point) {
0 262             return Absolute.Equals ((Point)other);
263         }
0 264         else if ((other = other as string) != null) {
0 265             return ToString ().Equals (other);
266         }
0 267         else {
0 268             return false;
269         }
0 270     }
271
272     [ExcludeFromCodeCoverageAttribute]
273     public override int GetHashCode ()
274     {
275         return Relative.GetHashCode ();
276     }
277 }
278 }
```

Knot3.Framework.Platform.SystemInfo

Summary

Class: Knot3.Framework.Platform.SystemInfo
Assembly: Knot3.Framework
File(s): I:\Documents\GitHub\knot3-code\framework\Knot3.Framework\Platform\SystemInfo.cs
Coverage: 100%
Covered lines: 16
Uncovered lines: 0
Coverable lines: 16
Total lines: 245

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
IsRunningOnMono()	1	100	100
IsRunningOnMonogame()	1	100	100
IsRunningOnLinux()	1	100	100
IsRunningOnWindows()	1	100	100
.cctor()	1	100	100

File(s)

I:\Documents\GitHub\knot3-code\framework\Knot3.Framework\Platform\SystemInfo.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *   Permission is hereby granted, free of charge, to any person obtaining a c
11  *   of this software and associated documentation files (the "Software"), to
12  *   in the Software without restriction, including without limitation the rig
13  *   to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *   copies of the Software, and to permit persons to whom the Software is
15  *   furnished to do so, subject to the following conditions:
16  *
17  *   The above copyright notice and this permission notice shall be included i
18  *   copies or substantial portions of the Software.
19  *
20  *   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
  
```



```
32 using System.Diagnostics.CodeAnalysis;
33 using System.IO;
34
35 namespace Knot3.Framework.Platform
36 {
37     public static partial class SystemInfo
38     {
39         /// <summary>
40         /// Das Einstellungsverzeichnis.
41         /// </summary>
42         [ExcludeFromCodeCoverageAttribute]
43         public static string SettingsDirectory
44         {
45             get {
46                 if (settingsDirectory != null) {
47                     return settingsDirectory;
48                 }
49                 else {
50                     string directory;
51                     if (SystemInfo.IsRunningOnLinux ()) {
52                         directory = Environment.GetEnvironmentVariable ("HOME"
53                     }
54                     else {
55                         directory = Environment.GetFolderPath (System.Environm
56                     }
57                     Directory.CreateDirectory (directory);
58                     return settingsDirectory = directory;
59                 }
60             }
61             set {
62                 settingsDirectory = value;
63             }
64         }
65
66         private static string settingsDirectory = null;
67
68         /// <summary>
69         /// Das Spielstandverzeichnis.
70         /// </summary>
71         [ExcludeFromCodeCoverageAttribute]
72         public static string SavegameDirectory
73         {
74             get {
75                 string directory = SettingsDirectory + "Savegames";
76                 Directory.CreateDirectory (directory);
77                 return directory;
78             }
79         }
80
81         /// <summary>
82         /// Das Bildschirmfotoverzeichnis.
83         /// </summary>
84         [ExcludeFromCodeCoverageAttribute]
85         public static string ScreenshotDirectory
86         {
87             get {
88                 string directory;
89                 if (SystemInfo.IsRunningOnLinux ()) {
90                     directory = Environment.GetEnvironmentVariable ("HOME");
91                 }
92                 else {
```

1

```
93         directory = Environment.GetFolderPath (System.Environment.
94     }
95     Directory.CreateDirectory (directory);
96     return directory;
97 }
98 }
99
100 [ExcludeFromCodeCoverageAttribute]
101 public static string DecodedMusicCache
102 {
103     get {
104         string directory;
105         if (SystemInfo.IsRunningOnLinux ()) {
106             directory = "/var/tmp/knot3/";
107         }
108         else {
109             directory = Environment.GetFolderPath (System.Environment.
110         }
111         Directory.CreateDirectory (directory);
112         return directory;
113     }
114 }
115
116 [ExcludeFromCodeCoverageAttribute]
117 public static string BaseDirectory
118 {
119     get {
120         if (baseDirectory != null) {
121             return baseDirectory;
122         }
123         else {
124             findBaseDirectory ();
125             return baseDirectory;
126         }
127     }
128 }
129
130 [ExcludeFromCodeCoverageAttribute]
131 public static string RelativeBaseDirectory
132 {
133     get {
134         if (relativeBaseDirectory != null) {
135             return relativeBaseDirectory;
136         }
137         else {
138             findBaseDirectory ();
139             return relativeBaseDirectory;
140         }
141     }
142     set {
143         Log.Debug ("Set Base directory: ", value);
144         baseDirectory = value;
145         Log.Debug ("Set Base directory (relative): ", value);
146         relativeBaseDirectory = value;
147     }
148 }
149
150 [ExcludeFromCodeCoverageAttribute]
151 private static void findBaseDirectory ()
152 {
153     string baseDir = Directory.GetCurrentDirectory ();
```

```

154         string relBaseDir = "." + PathSeparator;
155         string[] binDirectories = new string[] {
156             "Debug",
157             "Release",
158             "x86",
159             "bin",
160             "Game",
161             "ModelEditor",
162             "Tools",
163         };
164         foreach (string dir in binDirectories) {
165             if (baseDir.ToLower ().EndsWith (dir.ToLower ())) {
166                 baseDir = baseDir.Substring (0, baseDir.Length - dir.Lengt
167                 relBaseDir += ".." + PathSeparator;
168             }
169         }
170         Log.Debug ("Base directory: ", baseDir);
171         baseDirectory = baseDir;
172         Log.Debug ("Base directory (relative): ", relBaseDir);
173         relativeBaseDirectory = relBaseDir;
174     }
175
176     private static string relativeBaseDirectory = null;
177     private static string baseDirectory = null;
178     public readonly static char PathSeparator = Path.DirectorySeparatorCha
179
180     [ExcludeFromCodeCoverageAttribute]
181     public static string RelativeContentDirectory
182     {
183         get {
184             return SystemInfo.RelativeBaseDirectory + "Content" + PathSepa
185         }
186     }
187 }
188 }

```

cuments\GitHub\knot3-code\framework\Knot3.Framework\Platform\SystemInfo-XNA.cs

#	Line	Coverage
	1	/*
	2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
	3	* Gerd Augsburg, Christina Erler, Daniel Warzel
	4	*
	5	* This source code file is part of Knot3. Copying, redistribution and
	6	* use of the source code in this file in source and binary forms,
	7	* with or without modification, are permitted provided that the conditions
	8	* of the MIT license are met:
	9	*
	10	* Permission is hereby granted, free of charge, to any person obtaining a c
	11	* of this software and associated documentation files (the "Software"), to
	12	* in the Software without restriction, including without limitation the rig
	13	* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
	14	* copies of the Software, and to permit persons to whom the Software is
	15	* furnished to do so, subject to the following conditions:
	16	*
	17	* The above copyright notice and this permission notice shall be included i
	18	* copies or substantial portions of the Software.
	19	*
	20	* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
	21	* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
	22	* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T

```
23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System.Diagnostics.CodeAnalysis;
32
33 namespace Knot3.Framework.Platform
34 {
35     public static partial class SystemInfo
36     {
37         public static bool IsRunningOnMono ()
38         {
39             return false;
40         }
41
42         public static bool IsRunningOnMonogame ()
43         {
44             return false;
45         }
46
47         public static bool IsRunningOnLinux ()
48         {
49             return false;
50         }
51
52         public static bool IsRunningOnWindows ()
53         {
54             return true;
55         }
56     }
57 }
```

Knot3.Framework.Storage.BooleanOption

Summary

Class:	Knot3.Framework.Storage.BooleanOption
Assembly:	Knot3.Framework
File(s):	Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\BooleanOption.cs
Coverage:	100%
Covered lines:	10
Uncovered lines:	0
Coverable lines:	10
Total lines:	68

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	2	100	66.67
.cctor()	1	100	100

File(s)

Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\BooleanOption.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                                     Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System.Diagnostics.CodeAnalysis;
32
33 namespace Knot3.Framework.Storage
34 {
35     /// <summary>

```

```
36    /// Diese Klasse repräsentiert eine Option, welche die Werte \glqq Wahr\gr
37    /// </summary>
38    public sealed class BooleanOption : DistinctOption
39    {
40        /// <summary>
41        /// Eine Eigenschaft, die den aktuell abgespeicherten Wert zurückgibt.
42        /// </summary>
43        public new bool Value
44        {
45            get {
46                return base.Value == ConfigFile.True ? true : false;
47            }
48            set {
49                base.Value = value ? ConfigFile.True : ConfigFile.False;
50            }
51        }
52
53        public new static string[] ValidValues = new string[] {
54            ConfigFile.True,
55            ConfigFile.False
56        };
57
58        /// <summary>
59        /// Erstellt eine neue Option, welche die Werte \glqq Wahr\grqq oder \
60        /// angegebenen Abschnitt der angegebenen Einstellungsdatei.
61        /// [base=section, name, defaultValue?ConfigFile.True:ConfigFile.False
62        /// </summary>
63        public BooleanOption (string section, string name, bool defaultValue,
64            : base (section, name, defaultValue?ConfigFile.True:ConfigFile.False,
65            {
66            }
67        }
68    }
```

Knot3.Framework.Storage.Config

Summary

Class:	Knot3.Framework.Storage.Config
Assembly:	Knot3.Framework
File(s):	Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\Config.cs
Coverage:	66.6%
Covered lines:	10
Uncovered lines:	5
Coverable lines:	15
Total lines:	72

File(s)

Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\Config.cs

#	Line	Coverage
1	/*	
2	* Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,	
3	* Gerd Augsburg, Christina Erler, Daniel Warzel	
4	*	
5	* This source code file is part of Knot3. Copying, redistribution and	
6	* use of the source code in this file in source and binary forms,	
7	* with or without modification, are permitted provided that the conditions	
8	* of the MIT license are met:	
9	*	
10	* Permission is hereby granted, free of charge, to any person obtaining a c	
11	* of this software and associated documentation files (the "Software"), to	
12	* in the Software without restriction, including without limitation the rig	
13	* to use, copy, modify, merge, publish, distribute, sublicense, and/or sell	
14	* copies of the Software, and to permit persons to whom the Software is	
15	* furnished to do so, subject to the following conditions:	
16	*	
17	* The above copyright notice and this permission notice shall be included i	
18	* copies or substantial portions of the Software.	
19	*	
20	* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O	
21	* IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,	
22	* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T	
23	* AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER	
24	* LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F	
25	* OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN	
26	* SOFTWARE.	
27	*	
28	* See the LICENSE file for full license details of the Knot3 project.	
29	*/	
30		
31	using System.Diagnostics.CodeAnalysis;	
32		
33	using Knot3.Framework.Platform;	
34		
35	namespace Knot3.Framework.Storage	
36	{	
37	/// <summary>	
38	/// Eine statische Klasse, die eine Referenz auf die zentrale Einstellungs	
39	/// </summary>	
40	public static class Config	
41	{	
42	/// <summary>	

```
43      /// Die zentrale Einstellungsdatei des Spiels.
44      /// </summary>
45      public static ConfigFile Default
46      {
47          get {
48              if (_default == null) {
49                  _default = new ConfigFile (SystemInfo.SettingsDirectory +
50              }
51              return _default;
52          }
53          set {
54              _default = value;
55          }
56      }
57
58      private static ConfigFile _default;
59
60      public static ConfigFile Models
61      {
62          get {
63              if (_models == null) {
64                  _models = new ConfigFile (SystemInfo.RelativeContentDirect
65              }
66              return _models;
67          }
68      }
69
70      private static ConfigFile _models;
71  }
72 }
```


Knot3.Framework.Storage.ConfigFile

Summary

Class:	Knot3.Framework.Storage.ConfigFile
Assembly:	Knot3.Framework
File(s):	al\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\ConfigFile.cs
Coverage:	100%
Covered lines:	55
Uncovered lines:	0
Coverable lines:	55
Total lines:	155

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	2	100	66.67
SetOption(...)	1	100	100
GetOption(...)	1	100	100
SetOption(...)	2	100	100
GetOption(...)	4	100	80
SetOption(...)	1	100	100
GetOption(...)	1	100	100
floatToString(...)	1	100	100
stringToFloat(...)	2	100	100

File(s)

al\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\ConfigFile.cs

```

#   Line Coverage
    1  /*
    2  * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3  *                               Gerd Augsburg, Christina Erler, Daniel Warzel
    4  *
    5  * This source code file is part of Knot3. Copying, redistribution and
    6  * use of the source code in this file in source and binary forms,
    7  * with or without modification, are permitted provided that the conditions
    8  * of the MIT license are met:
    9  *
   10  * Permission is hereby granted, free of charge, to any person obtaining a c
   11  * of this software and associated documentation files (the "Software"), to
   12  * in the Software without restriction, including without limitation the rig
   13  * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14  * copies of the Software, and to permit persons to whom the Software is
   15  * furnished to do so, subject to the following conditions:
   16  *
   17  * The above copyright notice and this permission notice shall be included i
   18  * copies or substantial portions of the Software.
   19  *
   20  * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21  * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22  * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
   23  * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
   24  * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
   25  * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
   26  * SOFTWARE.
   27  *
   28  * See the LICENSE file for full license details of the Knot3 project.
```

```

29  */
30
31  using System;
32  using System.Diagnostics.CodeAnalysis;
33  using System.IO;
34
35  namespace Knot3.Framework.Storage
36  {
37      /// <summary>
38      /// Repräsentiert eine Einstellungsdatei.
39      /// </summary>
40      public sealed class ConfigFile
41      {
42          /// <summary>
43          /// Die Representation des Wahrheitswerts "wahr" als String in einer E
44          /// </summary>
489 45          public static string True { get { return "true"; } }
46
47          /// <summary>
48          /// Die Representation des Wahrheitswerts "falsch" als String in einer
49          /// </summary>
450 50          public static string False { get { return "false"; } }
51
52          private string Filename;
53          private IniFile ini;
54
55          public ConfigFile (string filename)
56          {
57              // load ini file
58              Filename = filename;
59
60              // create a new ini parser
61              using (StreamWriter w = File.AppendText (Filename)) {
62              }
63              ini = new IniFile (Filename);
64          }
65
66          /// <summary>
67          /// Setzt den Wert der Option mit dem angegebenen Namen in den angegeb
68          /// </summary>
69          public void SetOption (string section, string option, string _value)
70          {
71              ini [section, option] = _value;
72          }
73
74          /// <summary>
75          /// Gibt den aktuell in der Datei vorhandenen Wert fr die angegebene
76          /// </summary>
77          public string GetOption (string section, string option, string default
553 78          {
553 79              return ini [section, option, defaultValue];
553 80          }
81
82          /// <summary>
83          /// Setzt den Wert der Option mit dem angegebenen Namen in den angegeb
84          /// </summary>
85          public void SetOption (string section, string option, bool _value)
100 86          {
100 87              SetOption (section, option, _value ? True : False);
100 88          }
89

```

```

90      /// <summary>
91      /// Gibt den aktuell in der Datei vorhandenen Wert fr die angegebene
92      /// </summary>
93      public bool GetOption (string section, string option, bool defaultValu
100  94      {
100  95          return GetOption (section, option, defaultValue ? True : False) ==
100  96      }
97
98      public void SetOption (string section, string option, float _value)
100  99      {
100  100          SetOption (section, option, floatToString (_value));
100  101      }
102
103      public float GetOption (string section, string option, float defaultVa
200  104      {
200  105          return stringToFloat (GetOption (section, option, floatToString (d
200  106      }
107
108      private string floatToString (float f)
300  109      {
300  110          return String.Empty + ((int) (f * 1000)).ToString ();
300  111      }
112
113      private float stringToFloat (string s)
200  114      {
115          int i;
200  116          bool result = Int32.TryParse (s, out i);
300  117          if (true == result) {
100  118              return ((float)i) / 1000f;
119          }
100  120          else {
100  121              return 0;
122          }
200  123      }
124
125      public bool this [string section, string option, bool defaultValue = f
126      {
100  127          get {
100  128              return GetOption (section, option, defaultValue);
100  129          }
100  130          set {
100  131              SetOption (section, option, value);
100  132          }
133      }
134
135      public float this [string section, string option, float defaultValue =
136      {
200  137          get {
200  138              return GetOption (section, option, defaultValue);
200  139          }
100  140          set {
100  141              SetOption (section, option, value);
100  142          }
143      }
144
145      public string this [string section, string option, string defaultValue
146      {
253  147          get {
253  148              return GetOption (section, option, defaultValue);
253  149          }
112  150          set {

```


Knot3.Framework.Storage.DistinctOption

Summary

Class:	Knot3.Framework.Storage.DistinctOption
Assembly:	Knot3.Framework
File(s):	ocuments\GitHub\knot3-code\framework\Knot3.Framework\Storage\DistinctOption.cs
Coverage:	100%
Covered lines:	22
Uncovered lines:	0
Coverable lines:	22
Total lines:	85

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	3	100	100

File(s)

ocuments\GitHub\knot3-code\framework\Knot3.Framework\Storage\DistinctOption.cs

```

#   Line   Coverage
    1   /*
    2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3   *                                     Gerd Augsburg, Christina Erler, Daniel Warzel
    4   *
    5   * This source code file is part of Knot3. Copying, redistribution and
    6   * use of the source code in this file in source and binary forms,
    7   * with or without modification, are permitted provided that the conditions
    8   * of the MIT license are met:
    9   *
   10   * Permission is hereby granted, free of charge, to any person obtaining a c
   11   * of this software and associated documentation files (the "Software"), to
   12   * in the Software without restriction, including without limitation the rig
   13   * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14   * copies of the Software, and to permit persons to whom the Software is
   15   * furnished to do so, subject to the following conditions:
   16   *
   17   * The above copyright notice and this permission notice shall be included i
   18   * copies or substantial portions of the Software.
   19   *
   20   * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21   * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22   * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
   23   * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
   24   * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
   25   * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
   26   * SOFTWARE.
   27   *
   28   * See the LICENSE file for full license details of the Knot3 project.
   29   */
   30
   31 using System.Collections.Generic;
   32 using System.Diagnostics.CodeAnalysis;
   33 using System.Linq;
   34
   35 namespace Knot3.Framework.Storage
   36 {
   37     /// <summary>
```

```

38     /// Diese Klasse repräsentiert eine Option, die einen Wert aus einer disti
39     /// </summary>
40     public class DistinctOption : Option
41     {
42         /// <summary>
43         /// Eine Menge von Texten, welche die fr die Option gltigen Werte be
44         /// </summary>
20    45         public HashSet<string> ValidValues { get; private set; }
46
7    47         public virtual Dictionary<string,string> DisplayValidValues { get; pri
48         /// <summary>
49         /// Eine Eigenschaft, die den aktuell abgespeicherten Wert zurck gibt
50         /// </summary>
51         public override string Value
52         {
25    53             get {
25    54                 return base.Value;
25    55             }
9    56             set {
14   57                 if (ValidValues.Contains (value)) {
5    58                     base.Value = value;
5    59                 }
4    60                 else {
4    61                     base.Value = DefaultValue;
4    62                 }
9    63             }
64         }
65         public virtual string DisplayValue
66         {
7    67             get {
7    68                 return Value;
7    69             }
70         }
71
72         /// <summary>
73         /// Erstellt eine neue Option, die einen der angegebenen Werte aus val
74         /// angegebenen Abschnitt der angegebenen Einstellungsdatei.
75         /// [base=section, name, defaultValue, configFile]
76         /// </summary>
3    77         public DistinctOption (string section, string name, string defaultValu
3    78         : base (section, name, defaultValue, configFile)
3    79         {
3    80             ValidValues = new HashSet<string> (validValues);
3    81             ValidValues.Add (defaultValue);
337  82             DisplayValidValues = new Dictionary<string,string> (ValidValues.To
3    83         }
84     }
85 }

```

Knot3.Framework.Storage.FileUtility

Summary

Class:	Knot3.Framework.Storage.FileUtility
Assembly:	Knot3.Framework
File(s):	l\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\FileUtility.cs
Coverage:	50%
Covered lines:	18
Uncovered lines:	18
Coverable lines:	36
Total lines:	107

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
ConvertToFileName(..)	2	100	100
GetHash(...)	1	0	0
ToMD5Hash(...)	2	0	0
SearchFiles(...)	3	100	80
SearchFiles(...)	3	63.64	60
MoveNext()	5	22.22	28.57

File(s)

l\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\FileUtility.cs


```

#   Line   Coverage
    1   /*
    2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
    4   *
    5   * This source code file is part of Knot3. Copying, redistribution and
    6   * use of the source code in this file in source and binary forms,
    7   * with or without modification, are permitted provided that the conditions
    8   * of the MIT license are met:
    9   *
   10   * Permission is hereby granted, free of charge, to any person obtaining a c
   11   * of this software and associated documentation files (the "Software"), to
   12   * in the Software without restriction, including without limitation the rig
   13   * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14   * copies of the Software, and to permit persons to whom the Software is
   15   * furnished to do so, subject to the following conditions:
   16   *
   17   * The above copyright notice and this permission notice shall be included i
   18   * copies or substantial portions of the Software.
   19   *
   20   * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21   * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22   * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
   23   * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
   24   * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
   25   * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
   26   * SOFTWARE.
   27   *
   28   * See the LICENSE file for full license details of the Knot3 project.
   29   */
   30
   31 using System;
```

```

32 using System.Collections.Generic;
33 using System.Diagnostics.CodeAnalysis;
34 using System.IO;
35 using System.Linq;
36 using System.Security.Cryptography;
37 using System.Text;
38
39 namespace Knot3.Framework.Storage
40 {
41     /// <summary>
42     /// Eine Hilfsklasse fr Dateiooperationen.
43     /// </summary>
44     public static class FileUtility
45     {
46         /// <summary>
47         /// Konvertiert einen Namen eines Knotens oder einer Challenge in eine
48         /// </summary>
49         public static string ConvertToFileName (string name)
50         {
51             char[] arr = name.ToCharArray ();
52             arr = Array.FindAll<char> (arr, (c => (char.IsLetterOrDigit (c)
53                 || char.IsWhiteSpace (c)
54                 || c == '-'))
55             );
56             return new string (arr);
57         }
58
59         /// <summary>
60         /// Liefert einen Hash-Wert zu der durch filename spezifizierten Datei
61         /// </summary>
62         public static string GetHash (string filename)
63         {
64             return string.Join ("\n", FileUtility.ReadFrom (filename)).ToMD5Ha
65         }
66
67         public static string ToMD5Hash (this string TextToHash)
68         {
69             if (string.IsNullOrEmpty (TextToHash)) {
70                 return string.Empty;
71             }
72
73             MD5 md5 = new MD5CryptoServiceProvider ();
74             byte[] textToHash = Encoding.Default.GetBytes (TextToHash);
75             byte[] result = md5.ComputeHash (textToHash);
76
77             return System.BitConverter.ToString (result);
78         }
79
80         public static IEnumerable<string> ReadFrom (string file)
81         {
82             string line;
83             using (var reader = File.OpenText (file)) {
84                 while ((line = reader.ReadLine ()) != null) {
85                     yield return line;
86                 }
87             }
88         }
89
90         public static void SearchFiles (IEnumerable<string> directories, IEnum
91         {
92             foreach (string directory in directories) {

```

```
4      93          SearchFiles (directory, extensions, add);
4      94      }
2      95      }
        96
        97      public static void SearchFiles (string directory, IEnumerable<string>
4      98      {
4      99          Directory.CreateDirectory (directory);
4     100          var files = Directory.GetFiles (directory, ".*", SearchOption.All
30    101              .Where (s => extensions.Any (e => s.EndsWith (e))));
12    102          foreach (string file in files) {
0     103              add (file);
0     104          }
4     105      }
        106      }
107 }
```

Knot3.Framework.Storage.FloatOption

Summary

Class:	Knot3.Framework.Storage.FloatOption
Assembly:	Knot3.Framework
File(s):	l\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\FloatOption.cs
Coverage:	92%
Covered lines:	23
Uncovered lines:	2
Coverable lines:	25
Total lines:	96

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
convertToString(...)	1	100	100
stringToFloat(...)	2	75	66.67

File(s)

l\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\FloatOption.cs

```

#   Line Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Collections.Generic;
33 using System.Diagnostics.CodeAnalysis;
34 using System.Linq;

```

```

35
36 namespace Knot3.Framework.Storage
37 {
38     /// <summary>
39     /// Diese Klasse repräsentiert eine Option, welche die Werte \glqq Wahr\gr
40     /// </summary>
41     public sealed class FloatOption : DistinctOption
42     {
43         /// <summary>
44         /// Eine Eigenschaft, die den aktuell abgespeicherten Wert zurckgibt.
45         /// </summary>
46         public new float Value
47         {
48             get {
49                 return stringToFloat (base.Value);
50             }
51             set {
52                 base.Value = convertToString (value);
53             }
54         }
55
56         public override string DisplayValue
57         {
58             get {
59                 return String.Empty + stringToFloat (base.Value);
60             }
61         }
62
63         public override Dictionary<string,string> DisplayValidValues
64         {
65             get {
66                 return new Dictionary<string, string>(base.ValidValues.ToDicti
67             }
68         }
69
70         /// <summary>
71         /// Erstellt eine neue Option, welche die Werte \glqq Wahr\grqq oder \
72         /// angegebenen Abschnitt der angegebenen Einstellungsdatei.
73         /// [base=section, name, defaultValue?ConfigFile.True:ConfigFile.False
74         /// </summary>
75         public FloatOption (string section, string name, float defaultValue, I
76         : base (section, name, convertToString ( defaultValue),validValues.Sel
77         {
78         }
79
80         private static string convertToString (float f)
81         {
82             return (String.Empty + (int)(f * 1000f));
83         }
84         private static float stringToFloat (string s)
85         {
86             int i;
87             bool result = Int32.TryParse (s, out i);
88             if (true == result) {
89                 return ((float)i) / 1000f;
90             }
91             else {
92                 return 0;
93             }
94         }
95     }

```

96 }

Knot3.Framework.Storage.IniFile

Summary

Class:	Knot3.Framework.Storage.IniFile
Assembly:	Knot3.Framework
File(s):	ascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\IniFile.cs
Coverage:	60.6%
Covered lines:	40
Uncovered lines:	26
Coverable lines:	66
Total lines:	140

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	9	40.62	23.53
Save()	8	100	80
StripComments(...)	3	0	0
Encode(...)	1	100	100
Decode(...)	1	0	0

File(s)

ascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\IniFile.cs

```

#   Line Coverage
    1  /*
    2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
    3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
    4   *
    5   * This source code file is part of Knot3. Copying, redistribution and
    6   * use of the source code in this file in source and binary forms,
    7   * with or without modification, are permitted provided that the conditions
    8   * of the MIT license are met:
    9   *
   10   * Permission is hereby granted, free of charge, to any person obtaining a c
   11   * of this software and associated documentation files (the "Software"), to
   12   * in the Software without restriction, including without limitation the rig
   13   * to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
   14   * copies of the Software, and to permit persons to whom the Software is
   15   * furnished to do so, subject to the following conditions:
   16   *
   17   * The above copyright notice and this permission notice shall be included i
   18   * copies or substantial portions of the Software.
   19   *
   20   * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
   21   * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
   22   * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
   23   * AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
   24   * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
   25   * OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
   26   * SOFTWARE.
   27   *
   28   * See the LICENSE file for full license details of the Knot3 project.
   29   */
   30
   31 using System;
   32 using System.Collections.Generic;

```

```

33 using System.Diagnostics.CodeAnalysis;
34 using System.IO;
35 using System.Linq;
36
37 namespace Knot3.Framework.Storage
38 {
39     public sealed class IniFile : IDisposable
40     {
41         private string Filename;
42         public Dictionary<string, Dictionary<string, string>> Data;
43
44         public IniFile (string filename)
45         {
46             Data = new Dictionary<string, Dictionary<string, string>> ();
47             Filename = filename;
48             if (File.Exists (filename)) {
49                 using (StreamReader reader = new StreamReader (filename)) {
50                     string section = null;
51                     while (reader.Peek () != -1) {
52                         string line = StripComments (reader.ReadLine ().Trim ());
53                         if (line.StartsWith ("[" && line.EndsWith ("]")) {
54                             section = line.Substring (1, line.Length - 2);
55                             if (!Data.ContainsKey (section)) {
56                                 Data [section] = new Dictionary<string,string> ();
57                             }
58                         }
59                         else if (line.Contains ("=")) {
60                             string[] parts = line.Split ('=');
61                             if (section != null) {
62                                 Data [section] [Decode (parts [0].Trim ())] =
63                                     parts [1].Trim ();
64                             }
65                         }
66                     }
67                 }
68             }
69
70             [ExcludeFromCodeCoverageAttribute]
71             public void Dispose ()
72             {
73                 Dispose (true);
74                 GC.SuppressFinalize (this);
75             }
76
77             [ExcludeFromCodeCoverageAttribute]
78             private void Dispose (bool disposing)
79             {
80                 if (disposing) {
81                     Save ();
82                 }
83             }
84
85             public void Save ()
86             {
87                 using (StreamWriter writer = new StreamWriter (Filename)) {
88                     foreach (string section in Data.Keys.OrderBy (x => x)) {
89                         writer.WriteLine ("[" + section + "]");
90                         foreach (string key in Data[section].Keys.OrderBy (x => x))
91                             writer.WriteLine (Encode (key) + "=" + Encode (Data [s
92                                     ] [key]));
93                     }
94                 }
95             }
96         }
97     }
98 }

```

```

416      94      }
416      95      }
      96
      97      private static string StripComments (string line)
0      98      {
0      99          if (line != null) {
0      100              if (line.Contains ("//")) {
0      101                  return line.Remove (line.IndexOf ("//")).Trim ();
      102              }
0      103              return line.Trim ();
      104          }
0      105          return string.Empty;
0      106      }
      107
      108      public string this [string section, string key, string defaultValue =
      109      {
553      110          get {
567      111              if (!Data.ContainsKey (section)) {
14      112                  Data [section] = new Dictionary<string,string> ();
14      113              }
657      114              if (!Data [section].ContainsKey (key)) {
104      115                  Data [section] [key] = defaultValue;
104      116                  Save ();
104      117              }
553      118              string value = Data [section] [key];
553      119              return value;
553      120          }
312      121          set {
312      122              if (!Data.ContainsKey (section)) {
0      123                  Data [section] = new Dictionary<string,string> ();
0      124              }
312      125              Data [section] [key] = value;
312      126              Save ();
312      127          }
      128      }
      129
      130      private string Encode (string text)
61132      131      {
61132      132          return text.Replace ("\r", "").Replace ("\n", "\\n");
61132      133      }
      134
      135      private string Decode (string text)
0      136      {
0      137          return text.Replace ("\\n", "\n");
0      138      }
      139      }
      140  }

```

Knot3.Framework.Storage.KeyOption

Summary

Class:	Knot3.Framework.Storage.KeyOption
Assembly:	Knot3.Framework
File(s):	cal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\KeyOption.cs
Coverage:	100%
Covered lines:	10
Uncovered lines:	0
Coverable lines:	10
Total lines:	62

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	100	100
.cctor()	1	100	100

File(s)

cal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\KeyOption.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31  using System.Collections.Generic;
32  using System.Diagnostics.CodeAnalysis;
33
34  using Microsoft.Xna.Framework.Input;
35

```



```
36 using Knot3.Framework.Utilities;
37
38 namespace Knot3.Framework.Storage
39 {
40     public class KeyOption : DistinctOption
41     {
42         /// <summary>
43         /// Eine Eigenschaft, die den aktuell abgespeicherten Wert zurckgibt.
44         /// </summary>
45         public new Keys Value
46         {
47             get {
48                 return base.Value.ToEnumValue<Keys> ();
49             }
50             set {
51                 base.Value = value.ToEnumDescription<Keys> ();
52             }
53         }
54
55         public new static IEnumerable<string> ValidValues = typeof (Keys).ToEn
56
57         public KeyOption (string section, string name, Keys defaultValue, Conf
58             : base (section, name, defaultValue.ToEnumDescription<Keys> (), ValidV
59         {
60         }
61     }
62 }
```

Knot3.Framework.Storage.Language

Summary

Class:	Knot3.Framework.Storage.Language
Assembly:	Knot3.Framework
File(s):	scal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\Language.cs
Coverage:	0%
Covered lines:	0
Uncovered lines:	34
Coverable lines:	34
Total lines:	111

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	1	0	0
op_Inequality(...)	1	0	0
op_Equality(...)	4	0	0
Equals(...)	2	0	0
Equals(...)	3	0	0
op_Implicit(...)	1	0	0

File(s)

scal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\Language.cs

```
#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *   Permission is hereby granted, free of charge, to any person obtaining a c
11  *   of this software and associated documentation files (the "Software"), to
12  *   in the Software without restriction, including without limitation the rig
13  *   to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *   copies of the Software, and to permit persons to whom the Software is
15  *   furnished to do so, subject to the following conditions:
16  *
17  *   The above copyright notice and this permission notice shall be included i
18  *   copies or substantial portions of the Software.
19  *
20  *   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *   FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *   AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *   LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *   OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *   SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System.Diagnostics.CodeAnalysis;
```

```

32 using System.IO;
33
34 namespace Knot3.Framework.Storage
35 {
36     public class Language
37     {
38         /// <summary>
39         /// Der Sprachcode der Sprache.
40         /// </summary>
0 41         public string Code { get; private set; }
42
43         /// <summary>
44         /// Der Anzeigename der Sprache.
45         /// </summary>
0 46         public string DisplayName { get; private set; }
47
48         /// <summary>
49         /// Die Datei, welche Informationen fr die Lokalisierung enthlt.
50         /// </summary>
0 51         public ConfigFile Localization { get; private set; }
52
0 53         public Language (string file)
0 54         {
0 55             Code = Path.GetFileNameWithoutExtension (file).ToLower ();
0 56             file = Localizer.LanguageDirectory + Code + ".ini";
0 57             Localization = new ConfigFile (file);
0 58             DisplayName = Localization ["language", "displayname", Code];
0 59         }
60
61         public static bool operator != (Language a, Language b)
0 62         {
0 63             return !(a == b);
0 64         }
65
66         public static bool operator == (Language a, Language b)
0 67         {
68             // If both are null, or both are same instance, return true.
0 69             if (System.Object.ReferenceEquals (a, b)) {
0 70                 return true;
71             }
72
73             // If one is null, but not both, return false.
0 74             if (((object)a == null) || ((object)b == null)) {
0 75                 return false;
76             }
77
78             // Return true if the fields match:
0 79             return a.Code == b.Code;
0 80         }
81
82         public bool Equals (Language other)
0 83         {
0 84             return other != null && Code == other.Code;
0 85         }
86
87         public override bool Equals (object other)
0 88         {
0 89             if (other == null) {
0 90                 return false;
91             }
0 92             else if (other is Language) {

```

```
0 93         return Equals (other as Language);
94     }
0 95     else {
0 96         return false;
97     }
0 98     }
99
100    public static implicit operator string (Language language)
0 101    {
0 102        return language.Code;
0 103    }
104
105    [ExcludeFromCodeCoverageAttribute]
106    public override int GetHashCode ()
107    {
108        return Code.GetHashCode ();
109    }
110    }
111 }
```

Knot3.Framework.Storage.LanguageOption

Summary

Class:	Knot3.Framework.Storage.LanguageOption
Assembly:	Knot3.Framework
File(s):	ocuments\GitHub\knot3-code\framework\Knot3.Framework\Storage\LanguageOption.cs
Coverage:	0%
Covered lines:	0
Uncovered lines:	39
Coverable lines:	39
Total lines:	98

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	2	0	0
toDisplayName(...)	3	0	0
fromDisplayName(...)	3	0	0

File(s)

ocuments\GitHub\knot3-code\framework\Knot3.Framework\Storage\LanguageOption.cs

```

#   Line Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31  using System.Collections.Generic;
32  using System.Diagnostics.CodeAnalysis;
33  using System.Linq;
34

```

```

35 namespace Knot3.Framework.Storage
36 {
37     public class LanguageOption: DistinctOption
38     {
39         public new Language Value
40         {
41             get {
42                 string code = base.Value;
43                 foreach (Language lang in Localizer.ValidLanguages) {
44                     if (lang.Code == code) {
45                         return lang;
46                     }
47                 }
48                 return Localizer.CurrentLanguage;
49             }
50             set {
51                 base.Value = value.Code;
52             }
53         }
54
55         public override string DisplayValue
56         {
57             get {
58                 return toDisplayName (Value);
59             }
60         }
61
62         public override Dictionary<string,string> DisplayValidValues
63         {
64             get {
65                 Dictionary<string, string> dict = new Dictionary<string, string>();
66                 foreach (string value in base.ValidValues) {
67                     dict [toDisplayName (value)] = value;
68                 }
69                 return dict;
70             }
71         }
72
73         public LanguageOption (string section, string name, ConfigFile configFile
74             : base (section, name, Localizer.DefaultLanguageCode, from lang in Localizer.ValidLanguages)
75         {
76         }
77
78         private string toDisplayName (string code)
79         {
80             foreach (Language lang in Localizer.ValidLanguages) {
81                 if (lang.Code == code) {
82                     return lang.DisplayName;
83                 }
84             }
85             return Localizer.CurrentLanguage.DisplayName;
86         }
87
88         private string fromDisplayName (string displayName)
89         {
90             foreach (Language lang in Localizer.ValidLanguages) {
91                 if (lang.DisplayName == displayName) {
92                     return lang.Code;
93                 }
94             }
95             return Localizer.CurrentLanguage.Code;

```

<div><div></div><div></div></div>	0	96	}
		97	}
		98	}

Knot3.Framework.Storage.Localizer

Summary

Class:	Knot3.Framework.Storage.Localizer
Assembly:	Knot3.Framework
File(s):	cal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\Localizer.cs
Coverage:	0%
Covered lines:	0
Uncovered lines:	56
Coverable lines:	56
Total lines:	142

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
Localize(...)	9	0	0
ToUnicode(...)	1	0	0
.cctor()	1	0	0

File(s)

cal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\Localizer.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                               Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System;
32 using System.Collections.Generic;
33 using System.Diagnostics.CodeAnalysis;
34 using System.IO;

```



```

35 using System.Linq;
36
37 using Knot3.Framework.Platform;
38
39 namespace Knot3.Framework.Storage
40 {
41     /// <summary>
42     /// Eine statische Klasse, die Bezeichner in lokalisierten Text umsetzen k
43     /// </summary>
44     public static class Localizer
45     {
0 46         public static readonly string DefaultLanguageCode = "en";
47
48         /// <summary>
49         /// Gibt die zur Zeit in der zentralen Konfigurationsdatei eingestellt
50         /// </summary>
51         private static Option CurrentLanguageCode
52         {
0 53             get {
0 54                 if (_currentLanguageCode == null) {
0 55                     _currentLanguageCode = new Option ("language", "current",
56                     { Verbose = false });
0 57                 }
0 58                 return _currentLanguageCode;
0 59             }
60         }
61
62         private static Option _currentLanguageCode;
63
64         /// <summary>
65         /// Die aktuell geladene Sprache.
66         /// </summary>
0 67         public static Language CurrentLanguage { get; private set; }
68
69         public static string LanguageDirectory
70         {
0 71             get {
0 72                 string directory = SystemInfo.RelativeContentDirectory + "Lang
0 73                 Directory.CreateDirectory (directory);
0 74                 return directory;
0 75             }
76         }
77
78         private static Language[] _validLanguages;
79
80         public static Language[] ValidLanguages
81         {
0 82             get {
0 83                 if (_validLanguages != null) {
0 84                     return _validLanguages;
85                 }
0 86                 else {
0 87                     string[] files = Directory.GetFiles (LanguageDirectory);
0 88                     List<Language> languages = new List<Language> ();
0 89                     foreach (string file in files) {
0 90                         try {
0 91                             Log.Debug ("Language file: ", file);
0 92                             languages.Add (new Language (file: file));
0 93                         }
0 94                         catch (Exception ex) {
0 95                             Log.Error (ex);

```

```

0 96         }
0 97     }
0 98     if (languages.Count == 0) {
0 99         languages.Add (new Language (file: LanguageDirectory +
0 100     })
0 101     Log.Message ("Valid Languages: " + string.Join ("", ", ", from
0 102     return _validLanguages = languages.ToArray ());
0 103     }
0 104     }
0 105 }
0 106
0 107 /// <summary>
0 108 /// Liefert zu dem angegebenen Bezeichner den zugehörigen Text aus der
0 109 /// aktuellen Sprache zurück, die dabei aus der Einstellungsdatei des
0 110 /// </summary>
0 111 public static string Localize (this string text)
0 112 {
0 113     if (text == null) {
0 114         return "";
0 115     }
0 116     else if (text == string.Empty || text.Contains ("Exception") || te
0 117         return text;
0 118     }
0 119     else {
0 120         if (CurrentLanguage == null || CurrentLanguage.Code != Current
0 121             _validLanguages = null;
0 122         foreach (Language lang in ValidLanguages) {
0 123             if (lang.Code == CurrentLanguageCode.Value) {
0 124                 CurrentLanguage = lang;
0 125             }
0 126         }
0 127         CurrentLanguageCode.Value = CurrentLanguage.Code;
0 128     }
0 129
0 130     string trimmed = text.Trim ('\\r', '\\n', ' ', '\\t', ':');
0 131     string localized = CurrentLanguage.Localization ["text", trimm
0 132     return ToUnicode (text.Replace (trimmed, localized));
0 133 }
0 134 }
0 135
0 136 public static string ToUnicode (string text)
0 137 {
0 138     return text.Replace ("&auml;", "\\u00E4").Replace ("&ouml;", "\\u00F
0 139         .Replace ("&Auml;", "\\u00C4").Replace ("&Ouml;", "\\u00D6").
0 140 }
0 141 }
0 142 }

```

Knot3.Framework.Storage.Option

Summary

Class:	Knot3.Framework.Storage.Option
Assembly:	Knot3.Framework
File(s):	Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\Option.cs
Coverage:	100%
Covered lines:	22
Uncovered lines:	0
Coverable lines:	22
Total lines:	93

Metrics

Method	Cyclomatic Complexity	Sequence Coverage	Branch Coverage
.ctor(...)	2	100	66.67

File(s)

Pascal\Documents\GitHub\knot3-code\framework\Knot3.Framework\Storage\Option.cs

```

#   Line   Coverage
1   /*
2   * Copyright (c) 2013-2014 Tobias Schulz, Maximilian Reuter, Pascal Knodel,
3   *                                     Gerd Augsburg, Christina Erler, Daniel Warzel
4   *
5   * This source code file is part of Knot3. Copying, redistribution and
6   * use of the source code in this file in source and binary forms,
7   * with or without modification, are permitted provided that the conditions
8   * of the MIT license are met:
9   *
10  *  Permission is hereby granted, free of charge, to any person obtaining a c
11  *  of this software and associated documentation files (the "Software"), to
12  *  in the Software without restriction, including without limitation the rig
13  *  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
14  *  copies of the Software, and to permit persons to whom the Software is
15  *  furnished to do so, subject to the following conditions:
16  *
17  *  The above copyright notice and this permission notice shall be included i
18  *  copies or substantial portions of the Software.
19  *
20  *  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS O
21  *  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
22  *  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL T
23  *  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
24  *  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING F
25  *  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
26  *  SOFTWARE.
27  *
28  * See the LICENSE file for full license details of the Knot3 project.
29  */
30
31 using System.Diagnostics.CodeAnalysis;
32
33 using Knot3.Framework.Platform;
34
35 namespace Knot3.Framework.Storage
36 {
37     /// <summary>

```

```

38     /// Enthlt Informationen ber einen Eintrag in einer Einstellungsdatei.
39     /// </summary>
40     public class Option
41     {
42         /// <summary>
43         /// Die Einstellungsdatei.
44         /// </summary>
45         private ConfigFile ConfigFile;
46
47         /// <summary>
48         /// Der Abschnitt der Einstellungsdatei.
49         /// </summary>
96     50         public string Section { get; private set; }
51
52         /// <summary>
53         /// Der Name der Option.
54         /// </summary>
96     55         public string Name { get; private set; }
56
57         /// <summary>
58         /// Der Standardwert der Option.
59         /// </summary>
66     60         public string DefaultValue { get; private set; }
61
62     62         public bool Verbose { get; set; }
63
64         /// <summary>
65         /// Der Wert der Option.
66         /// </summary>
67         public virtual string Value
68         {
25     69             get {
50     70                 if (Verbose) {
25     71                     Log.Debug ("Option: ", Section, ".", Name, " => ", ConfigF
25     72                 }
25     73                 return ConfigFile [Section, Name, DefaultValue];
25     74             }
9    75             set {
9    76                 Log.Debug ("Option: ", Section, ".", Name, " <= ", value);
9    77                 ConfigFile [Section, Name, DefaultValue] = value;
9    78             }
79         }
80
81         /// <summary>
82         /// Erstellt ein neues OptionsInfo-Objekt aus den bergegebenen Werten
83         /// </summary>
3    84         public Option (string section, string name, string defaultValue, Confi
3    85         {
3    86             Section = section;
3    87             Name = name;
3    88             DefaultValue = defaultValue;
3    89             ConfigFile = configFile != null ? configFile : Config.Default;
3    90             Verbose = true;
3    91         }
92     }
93 }

```