

ENTWURFSDOKUMENT

(V. 1.0)

KNOT³

PSE WS 2013/14

Auftraggeber:

Karlsruher Institut für Technologie
Institut für Betriebs- und Dialogsysteme
Prof. Dr.-Ing. C. Dachsbacher

Betreuer:

Dipl.-Inf. Thorsten Schmidt
Dipl.-Inf. M. Retzlaff

Auftragnehmer:

Tobias Schulz, Maximilian Reuter, Pascal Knodel,
Gerd Augsburg, Christina Erler, Daniel Warzel

15. Dezember 2013

Inhaltsverzeichnis

1	Einleitung	4
2	Aufbau	5
2.1	Architektur	5
2.2	Klassendiagramm	6
2.3	Verwendete Entwurfsmuster	6
3	Klassenübersicht	7
3.1	Klassen	7
3.1.1	Klasse ArrowModel	7
3.1.2	Klasse ArrowModelInfo	8
3.1.3	Klasse AudioSettingsScreen	8
3.1.4	Klasse BooleanOptionInfo	8
3.1.5	Klasse Camera	9
3.1.6	Klasse CelShadingEffect	10
3.1.7	Klasse Challenge	11
3.1.8	Klasse ChallengeFileIO	11
3.1.9	Klasse ChallengeLoadScreen	12
3.1.10	Klasse ChallengeMetaData	13
3.1.11	Klasse ChallengeMode	13
3.1.12	Klasse CheckBoxItem	14
3.1.13	Klasse Circle	14
3.1.14	Klasse Class1	15
3.1.15	Klasse ColorPicker	15
3.1.16	Klasse ColorPickItem	16
3.1.17	Klasse ConfigFile	16
3.1.18	Klasse ControlSettingsScreen	16
3.1.19	Klasse CreativeLoadScreen	17
3.1.20	Klasse CreativeMode	17
3.1.21	Klasse CreditsScreen	17
3.1.22	Klasse Dialog	18
3.1.23	Klasse DistinctOptionInfo	18
3.1.24	Klasse DrawableGameStateComponent	19
3.1.25	Klasse DropDownMenuItem	19
3.1.26	Klasse Edge	20
3.1.27	Klasse Edge	20
3.1.28	Klasse FadeEffect	21
3.1.29	Klasse FileIO	21
3.1.30	Klasse Game	22

3.1.31	Klasse	GameModelInfo	22
3.1.32	Klasse	GameObjectInfo	23
3.1.33	Klasse	GameScreen	23
3.1.34	Klasse	GameStateComponent	24
3.1.35	Klasse	GraphicsSettingsScreen	25
3.1.36	Klasse	IGameObject	25
3.1.37	Klasse	InputItem	26
3.1.38	Klasse	KeyInputItem	27
3.1.39	Klasse	Knot	27
3.1.40	Klasse	Knot3Game	29
3.1.41	Klasse	KnotFileIO	30
3.1.42	Klasse	KnotMetaData	30
3.1.43	Klasse	KnotStringIO	31
3.1.44	Klasse	Localizer	31
3.1.45	Klasse	Menu	31
3.1.46	Klasse	MenuButton	32
3.1.47	Klasse	MenuItem	33
3.1.48	Klasse	MenuItem	33
3.1.49	Klasse	MenuScreen	34
3.1.50	Klasse	ModelFactory	34
3.1.51	Klasse	ModelkeyHandler	35
3.1.52	Klasse	ModelMouseHandler	35
3.1.53	Klasse	ModelRenderer	35
3.1.54	Klasse	MousePointer	36
3.1.55	Klasse	NodeModel	36
3.1.56	Klasse	NodeModelInfo	37
3.1.57	Klasse	OptionInfo	37
3.1.58	Klasse	Options	38
3.1.59	Klasse	PipeModel	38
3.1.60	Klasse	PipeModelInfo	39
3.1.61	Klasse	PipeMovement	39
3.1.62	Klasse	ProfileSettingsScreen	40
3.1.63	Klasse	RenderEffect	41
3.1.64	Klasse	RenderEffectStack	41
3.1.65	Klasse	SettingsScreen	42
3.1.66	Klasse	ShadowGameModel	42
3.1.67	Klasse	ShadowGameObject	43
3.1.68	Klasse	SliderItem	44
3.1.69	Klasse	StandardEffect	44
3.1.70	Klasse	TextInputDialog	44
3.1.71	Klasse	TutorialChallengeMode	45
3.1.72	Klasse	Widget	45
3.1.73	Klasse	WidgetKeyHandler	46
3.1.74	Klasse	WidgetMouseHandler	46
3.1.75	Klasse	World	46
3.1.76	Klasse	XNA.DrawableGameComponent	47
3.1.77	Klasse	XNA.Game	47
3.1.78	Klasse	XNA.GameComponent	48
3.2	Schnittstellen		48
3.2.1	Schnittstelle	ICallengeIO	48
3.2.2	Schnittstelle	ICloneable	49

3.2.3	Schnittstelle IEnumerable	49
3.2.4	Schnittstelle IEnumerable	49
3.2.5	Schnittstelle IEnumerable<IGameObject>	49
3.2.6	Schnittstelle IEnumerable<IGameObject>	50
3.2.7	Schnittstelle IGameObject	50
3.2.8	Schnittstelle IGameStateComponent	50
3.2.9	Schnittstelle IJunction	51
3.2.10	Schnittstelle IKeyEventListener	51
3.2.11	Schnittstelle IKnotIO	52
3.2.12	Schnittstelle IMouseEventListener	52
3.2.13	Schnittstelle IRenderEffect	53
3.2.14	Schnittstelle XNA.IGameComponent	53
3.3	Enumerationen	54
4	Abläufe	55
4.1	Sequenzdiagramme	55
5	Klassenindex	56
6	Anmerkungen	57
7	Glossar	58
7.1	Fachausdrücke	58
7.2	Abkürzungen	59

Kapitel 1

Einleitung

Das Knobel- und Konstruktionsspiel Knot³, welches im Auftrag des IBDS Dachsbacher ausgearbeitet und wie im Pflichtenheft spezifiziert angefertigt wird.

Kapitel 2

Aufbau

2.1 Architektur

Die grundlegende Architektur des Spiels basiert auf der Spielkomponenten-Infrastruktur des XNA-Frameworks, die mit Spielzuständen kombiniert wird. Die abstrakten Klassen `GameStateComponent` und `DrawableGameStateComponent` erben von den von XNA bereitgestellten Klassen `GameComponent` und `DrawableGameComponent` implementieren zusätzlich die Schnittstelle `IGameStateComponent`. Sie unterscheiden sich von den XNA-Basisklassen dadurch, dass sie immer eine Referenz auf einen bestimmten Spielzustand halten und nur in Kombination mit diesem zu verwenden sind.

Die Spielzustände erben von der abstrakten Basisklasse `GameScreen` und halten eine Liste von `IGameStateComponent`-Objekten. Wird ein Spielzustand aktiviert, indem von einem anderen Spielzustand aus zu ihm gewechselt wird oder indem er der Startzustand ist, dann weist er seine Liste von `IGameStateComponent`-Objekten dem `Components`-Attribut der `Game`-Klasse zu, die von der vom XNA-Framework bereitgestellten abstrakten Klasse `Game` erbt. So ist zu jedem Zeitpunkt während der Laufzeit des Spiels ein Spielzustand aktiv, der die aktuelle Liste von Spielkomponenten verwaltet.

Die Spielkomponenten, die nicht gezeichnet werden und nur auf Eingaben reagieren, haben nur eine `Update()`-Methode und erben von `GameStateComponent`. Dies sind vor allem verschiedene Input-Handler, welche Tastatur- und Mauseingaben verarbeiten und beispielsweise die Kameraposition und das Kameratarget ändern oder Spielobjekte bewegen.

Spielkomponenten, die neben der `Update()`-Methode auch eine `Draw()`-Methode besitzen, erben von `DrawableGameStateComponent`. Dies sind vor allem die Elemente, aus denen die grafische Benutzeroberfläche zusammengesetzt ist, deren abstrakte Basisklasse `Widget` darstellt. [weitere Erklärungen zu Widgets...]

Alle Spielobjekte implementieren die Schnittstelle `IGameObject`. Die abstrakte Klasse `GameModel` repräsentiert dabei ein Spielobjekt, das aus einem 3D-Modell besteht, und hält zu diesem Zweck eine Referenz auf ein Objekt der Klasse `Model` aus dem XNA-Framework sowie weitere Eigenschaften wie Position, Drehung und Skalierung.

Spielobjekte sind keine Komponenten, sondern werden in einer Spielwelt zusammengefasst, die durch die Klasse `World` repräsentiert wird. Die Spielwelt ist ein `DrawableGameStateComponent` und ruft in ihren `Update()`- und `Draw()`-Methoden jeweils die dazugehörigen Methoden aller in ihr enthaltenen Spielobjekte auf.

Shadereffekte werden durch die abstrakte Klasse `RenderEffect` und die von ihr abgeleiteten Klassen gekapselt. Ein `RenderEffect` enthält ein Rendertarget vom Typ `RenderTarget2D` als Attribut und implementiert jeweils eine `Begin()`- und eine `End()`-Methode. In der Methode `Begin()` wird das aktuell von XNA genutzte

Rendertarget auf einem Stack gesichert und das Rendertarget des Effekts wird als aktuelles Rendertarget gesetzt.

Nach dem Aufruf von `Begin()` werden alle `Draw()`-Aufrufe von XNA auf dem gesetzten Rendertarget ausgeführt. Es wird also in eine im `RenderTarget2D`-Objekt enthaltene Bitmap gezeichnet. Dabei wird von den `Draw()`-Methoden der `GameModels` die `DrawModel(GameModel)`-Methode des `RenderEffects` aufgerufen, der die Modelle mit bestimmten Shadereffekten in die Bitmap zeichnet.

In der `End()`-Methode wird schließlich das auf dem Stack gesicherte, vorher genutzte Rendertarget wiederhergestellt und das Rendertarget des `RenderEffects` wird, unter Umständen verändert durch Post-Processing-Effekte, auf dieses übergeordnete Rendertarget gezeichnet.

2.2 Klassendiagramm

2.3 Verwendete Entwurfsmuster

Kapitel 3

Klassenübersicht

3.1 Klassen

3.1.1 Klasse ArrowModel

Beschreibung:

Diese Klasse repräsentiert ein 3D-Modell für einen Pfeil, der an selektierten Kanten erscheinen soll.

Eigenschaften:

public ArrowModelInfo Info

Das Info-Objekt, das die Position und Richtung des Pfeils enthält.

Methoden:

public void Draw (GameTime)

Zeichnet den Pfeil.

public GameObjectDistance Intersects (Ray, GameObjectDistance Ray)

public void ArrowModel (GameScreen, ArrowModelInfo)

Erstellt ein neues Pfeilmodell in dem angegebenen GameScreen mit einem bestimmten Info-Objekt, das Position und Richtung des Pfeils festlegt.

public void Update (GameTime)

Arrow Model
+ Info : Arrow ModelInfo
+ Draw (GameTime) : void + Intersects (Ray, GameObjectDistance Ray) : GameObjectDistance + Arrow Model (GameScreen, Arrow ModelInfo) : void + Update (GameTime) : void

3.1.2 Klasse ArrowModelInfo

Beschreibung:

Ein Objekt dieser Klasse hält alle Informationen, die zur Erstellung eines Pfeil-3D-Modelles (ArrowModel) notwendig sind.

Eigenschaften:

public Vector3 Direction

Die Richtung, die der Pfeil zeigen soll.

Arrow ModelInfo
+ Direction : Vector3
+ Arrow ModelInfo (Vector3, Vector3) : void

Methoden:

public void ArrowModelInfo (**Vector3**, **Vector3**)

Erstellt ein neues ArrowModelInfo-Objekt an einer bestimmten Position im 3D-Raum, das in eine bestimmte Richtung zeigt.

3.1.3 Klasse AudioSettingsScreen

Beschreibung:

Eigenschaften:

protected void settingsMenu

AudioSettingsScreen
settingsMenu : void
+ Update () : void
+ Update (GameTime, GameScreen previousScreen) : GameScreen

Methoden:

public void Update ()

public GameScreen Update (**GameTime**, **GameScreen** previousScreen)

3.1.4 Klasse BooleanOptionInfo

Beschreibung:

Diese Klasse repräsentiert eine Option, die die Werte Wahr oder Falsch annehmen kann.

Eigenschaften:

public bool Value

Ein Property, das den aktuell abgespeicherten Wert zurück gibt.

BooleanOptionInfo
+ Value : bool
+ BooleanOptionInfo (, , ,) : void

Methoden:

public void BooleanOptionInfo (, , ,)

3.1.5 Klasse Camera

Beschreibung:

Jede Instanz der World-Klasse hält eine für diese Spielwelt verwendete Kamera als Attribut. Die Hauptfunktion der Kamera-Klasse ist das Berechnen der drei Matrizen, die für die Positionierung und Skalierung von 3D-Objekten in einer bestimmten Spielwelt benötigt werden, der View-, World- und Projection-Matrix. Um diese Matrizen zu berechnen, benötigt die Kamera unter Anderem Informationen über die aktuelle Kamera-Position, das aktuelle Kamera-Target und das Field of View.

Eigenschaften:

private void World

Eine Referenz auf die Spielwelt, für die die Kamera zuständig ist.

public Vector3 Position

Die Position der Kamera.

public Vector3 Target

Das Ziel der Kamera.

public float FoV

Das Field of View.

public Matrix ViewMatrix

Die View-Matrix wird über die statische Methode CreateLookAt der Klasse Matrix des XNA-Frameworks mit Matrix.CreateLookAt (Position, Target, Vector3.Up) berechnet.

public Matrix WorldMatrix

Die World-Matrix wird mit Matrix.CreateFromYawPitchRoll und den drei Rotationswinkeln berechnet.

public Matrix ProjectionMatrix

Camera
- World : void + Position : Vector3 + Target : Vector3 + FoV : float + View Matrix : Matrix + WorldMatrix : Matrix + ProjectionMatrix : Matrix + ArcballTarget : Vector3 + View Frustum : BoundingFrustum
+ TargetDirection (Vector3) : Vector3 + TargetDistance (float) : float + Camera (GameScreen, World) : void + GetMouseRay (Vector2, Ray) : Ray + Update (GameTime) : void

Die Projektionsmatrix wird über die statische XNA-Methode `Matrix.CreatePerspectiveFieldOfView` berechnet.

public Vector3 ArcballTarget

Eine Position, um die rotiert werden soll, wenn der User die rechte Maustaste gedrückt hält und die Maus bewegt.

public BoundingBox ViewFrustum

Berechnet ein Bounding-Frustum, das benötigt wird, um festzustellen, ob ein 3D-Objekt sich um Blickfeld des Spielers befindet.

Methoden:

public Vector3 TargetDirection (Vector3)

public float TargetDistance (float)

public void Camera (GameScreen, World)

public Ray GetMouseRay (Vector2, Ray)

public void Update (GameTime)

3.1.6 Klasse CelShadingEffect

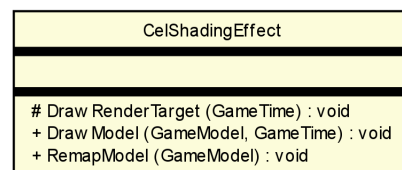
Beschreibung:

Methoden:

protected void DrawRenderTarget (GameTime)

public void DrawModel (GameModel, GameTime)

public void RemapModel (GameModel)



3.1.7 Klasse Challenge

Beschreibung:

Eigenschaften:

public Knot Start

public Knot Target

private SortedList<Integer, String> highscore

public String Name

private IChallengeIO file

public IEnumerable<KeyValuePair<String, Integer>> Highscore

public ChallengeMetaData Info

Methoden:

public ChallengeInfo Challenge (ChallengeInfo info, Challenge)

public Challenge Challenge (Challenge, IChallengeIO file)

public Boolean CreateChallenge (Boolean, Knot start, Knot target, String name, IChallengeIO file)

public String AddToHighscore (String name, Integer time)

Challenge
+ Start : Knot + Target : Knot - highscore : SortedList<Integer, String> + Name : String - file : IChallengeIO + Highscore : IEnumerable<KeyValuePair<String, Integer>> + Info : ChallengeMetaData
+ Challenge (ChallengeInfo info, Challenge) : ChallengeInfo + Challenge (Challenge, IChallengeIO file) : Challenge + CreateChallenge (Boolean, Knot start, Knot target, String name, IChallengeIO file) : Boolean + AddToHighscore (String name, Integer time) : String

3.1.8 Klasse ChallengeFileIO

Beschreibung:

Eigenschaften:

`public IEnumerable<KeyValuePair<String, Integer>> Highscore`

`public String Name`

`public Knot StartKnot`

`public Knot TargetKnot`

`private KnotStringIO startParser`

`private KnotStringIO targetParser`

`public ChallengeMetaData Meta`

Methoden:

`public String ChallengeFileIO (String path)`

`public Challenge Save (Challenge challenge)`

ChallengeFileIO
+ Highscore : IEnumerable<KeyValuePair<String, Integer>> + Name : String + StartKnot : Knot + TargetKnot : Knot - startParser : KnotStringIO - targetParser : KnotStringIO + Meta : ChallengeMetaData
+ ChallengeFileIO (String path) : String + Save (Challenge challenge) : Challenge

3.1.9 Klasse ChallengeLoadScreen

Beschreibung:

Methoden:

`public void Update ()`

`public GameScreen Update (GameTime, GameScreen previousScreen)`

ChallengeLoadScreen
+ Update () : void + Update (GameTime, GameScreen previousScreen) : GameScreen

3.1.10 Klasse ChallengeMetaData

Beschreibung:

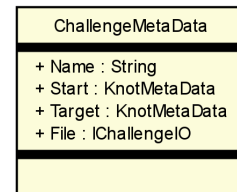
Eigenschaften:

`public String Name`

`public KnotMetaData Start`

`public KnotMetaData Target`

`public IChallengeIO File`



3.1.11 Klasse ChallengeMode

Beschreibung:

Eigenschaften:

`public void PlayerKnot`

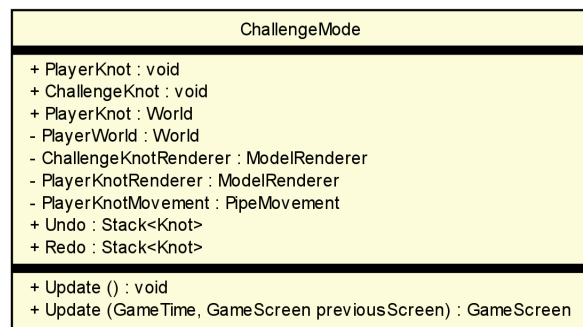
`public void ChallengeKnot`

`public World PlayerKnot`

`private World PlayerWorld`

`private ModelRenderer ChallengeKnotRenderer`

`private ModelRenderer PlayerKnotRenderer`



```
private PipeMovement PlayerKnotMovement
```

```
public Stack<Knot> Undo
```

```
public Stack<Knot> Redo
```

Methoden:

```
public void Update ()
```

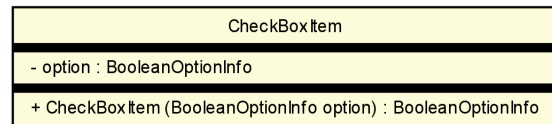
```
public GameScreen Update (GameTime, GameScreen previousScreen)
```

3.1.12 Klasse CheckBoxItem

Beschreibung:

Eigenschaften:

```
private BooleanOptionInfo option
```



Methoden:

```
public BooleanOptionInfo CheckBoxItem (BooleanOptionInfo option)
```

3.1.13 Klasse Circle

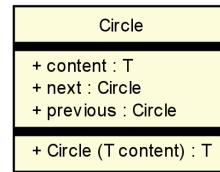
Beschreibung:

Eigenschaften:

`public T content`

`public Circle next`

`public Circle previous`



Methoden:

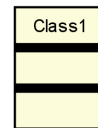
`public T Circle (T content)`

3.1.14 Klasse Class1

Beschreibung:

Eigenschaften:

`private VerticalMenu pauseMenu`



3.1.15 Klasse ColorPicker

Beschreibung:

Eigenschaften:

`public Color Color`

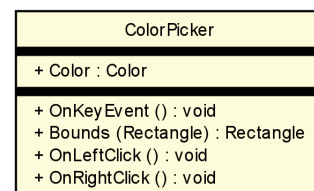
Methoden:

`public void OnKeyEvent ()`

`public Rectangle Bounds (Rectangle)`

`public void OnLeftClick ()`

`public void OnRightClick ()`

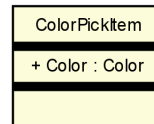


3.1.16 Klasse ColorPickItem

Beschreibung:

Eigenschaften:

`public Color Color`



3.1.17 Klasse ConfigFile

Beschreibung:

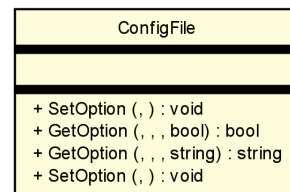
Methoden:

`public void SetOption (,)`

`public bool GetOption (, , bool)`

`public string GetOption (, , string)`

`public void SetOption (,)`



3.1.18 Klasse ControlSettingsScreen

Beschreibung:

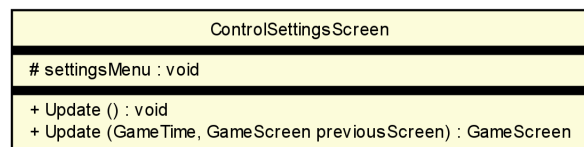
Eigenschaften:

`protected void settingsMenu`

Methoden:

`public void Update ()`

`public GameScreen Update (GameTime, GameScreen previousScreen)`



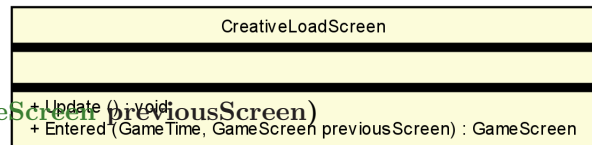
3.1.19 Klasse CreativeLoadScreen

Beschreibung:

Methoden:

`public void Update ()`

`public GameScreen Entered (GameTime, GameScreen previousScreen)`



3.1.20 Klasse CreativeMode

Beschreibung:

Eigenschaften:

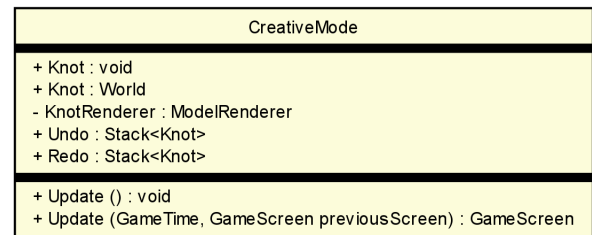
`public void Knot`

`public World Knot`

`private ModelRenderer KnotRenderer`

`public Stack<Knot> Undo`

`public Stack<Knot> Redo`



Methoden:

`public void Update ()`

`public GameScreen Update (GameTime, GameScreen previousScreen)`

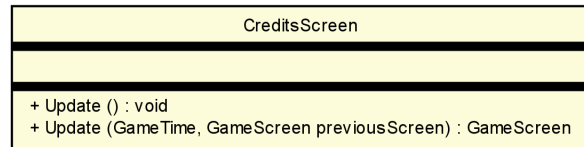
3.1.21 Klasse CreditsScreen

Beschreibung:

Methoden:

```
public void Update ()
```

```
public GameScreen Update (GameTime, GameScreen previousScreen)
```



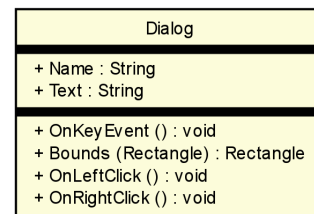
3.1.22 Klasse Dialog

Beschreibung:

Eigenschaften:

```
public String Name
```

```
public String Text
```



Methoden:

```
public void OnKeyEvent ()
```

```
public Rectangle Bounds (Rectangle)
```

```
public void OnLeftClick ()
```

```
public void OnRightClick ()
```

3.1.23 Klasse DistinctOptionInfo

Beschreibung:

Eigenschaften:

`public HashSet<string> ValidValues`

`public String Value`

DistinctOptionInfo
+ ValidValues : HashSet<string> + Value : String
+ DistinctOptionInfo (, , ,) : void

Methoden:

`public void DistinctOptionInfo (, , ,)`

3.1.24 Klasse DrawableGameStateComponent

Beschreibung:

Eigenschaften:

`public GameScreen State`

`public DisplayLayer Index`

DrawableGameStateComponent
+ State : GameScreen + Index : DisplayLayer
+ SubComponents (GameTime, IEnumerable GameTime) : IEnumerable + DrawableGameStateComponent (GameScreen, DisplayLayer) : void

Methoden:

`public IEnumerable SubComponents (GameTime, IEnumerable GameTime)`

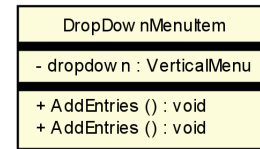
`public void DrawableGameStateComponent (GameScreen, DisplayLayer)`

3.1.25 Klasse DropDownMenuItem

Beschreibung:

Eigenschaften:

`private VerticalMenu dropdown`



Methoden:

`public void AddEntries ()`

`public void AddEntries ()`

3.1.26 Klasse Edge

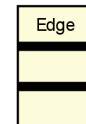
Beschreibung:

Eigenschaften:

`public IEnumerable Edges`

`public String Name`

`public KnotMetaData Meta`



Methoden:

`public IEnumerator<Edge> GetEnumerator (IEnumerator<Edge>)`

`public Knot Save (Knot knot)`

`public string PrinterIO (string path)`

3.1.27 Klasse Edge

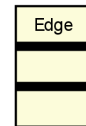
Beschreibung:

Eigenschaften:

`public Color EdgeColor`

`public Direction Dir`

`public List<int> Rectangles`



Methoden:

`public Direction Edge (Direction dir)`

`public Vector3 Get3DDirection (Vector3)`

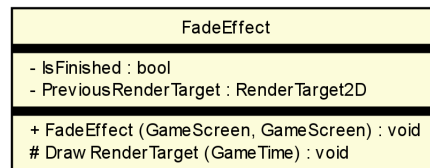
3.1.28 Klasse FadeEffect

Beschreibung:

Eigenschaften:

`private bool IsFinished`

`private RenderTarget2D PreviousRenderTarget`



Methoden:

`public void FadeEffect (GameScreen, GameScreen)`

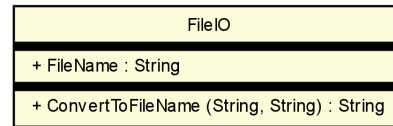
`protected void DrawRenderTarget (GameTime)`

3.1.29 Klasse FileIO

Beschreibung:

Eigenschaften:

`public String` FileName



Methoden:

`public String` ConvertToFileName (`String`, `String`)

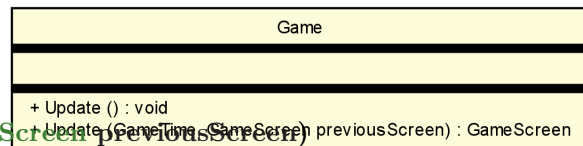
3.1.30 Klasse Game

Beschreibung:

Methoden:

`public void` Update ()

`public GameScreen` Update (`GameTime`, `GameScreen` previousScreen)



3.1.31 Klasse GameModelInfo

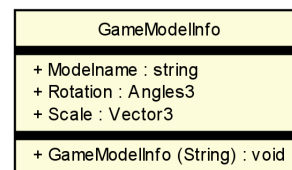
Beschreibung:

Eigenschaften:

`public string` Modelname

`public Angles3` Rotation

`public Vector3` Scale



Methoden:

`public void` GameModelInfo (`String`)

3.1.32 Klasse GameObjectInfo

Beschreibung:

Eigenschaften:

`public bool IsMovable`

`public bool IsSelectable`

`public bool IsVisible`

`public Vector3 Position`

GameObjectInfo
+ IsMovable : bool + IsSelectable : bool + IsVisible : bool + Position : Vector3
+ Equals (GameObjectInfo, bool GameObjectInfo) : bool

Methoden:

`public bool Equals (GameObjectInfo, bool GameObjectInfo)`

3.1.33 Klasse GameScreen

Beschreibung:

Eigenschaften:

`public Knot3Game Game`

`public InputHandler Input`

`public RenderEffect PostProcessingEffect`

GameScreen
+ Game : Knot3Game + Input : InputHandler + PostProcessingEffect : RenderEffect
+ Update () : void + Entered (GameTime, GameScreen previousScreen) : GameScreen + BeforeExit (GameTime, GameScreen nextScreen) : GameScreen + Update (Game) : void + AddGameComponents () : void + RemoveGameComponents () : void

Methoden:

```
public void Update ()
```

```
public GameScreen Entered (GameTime, GameScreen previousScreen)
```

```
public GameScreen BeforeExit (GameTime, GameScreen nextScreen)
```

```
public void Update (Game)
```

```
public void AddGameComponents ()
```

```
public void RemoveGameComponents ()
```

3.1.34 Klasse GameStateComponent

Beschreibung:

Eigenschaften:

```
public DisplayLayer Index
```

```
public GameScreen State
```

GameStateComponent
+ Index : DisplayLayer + State : GameScreen
+ SubComponents (GameTime, IEnumerable GameTime) : IEnumerable + GameStateComponent (GameScreen, DisplayLayer) : void

Methoden:

```
public IEnumerable SubComponents (GameTime, IEnumerable GameTime)
```

```
public void GameStateComponent (GameScreen, DisplayLayer)
```

3.1.35 Klasse GraphicsSettingsScreen

Beschreibung:

Eigenschaften:

`protected void settingsMenu`

Methoden:

`public void Update ()`

`public GameScreen Update (GameTime, GameScreen previousScreen)`

GraphicsSettingsScreen
settingsMenu : void
+ Update () : void + Update (GameTime, GameScreen previousScreen) : GameScreen

3.1.36 Klasse IGameObject

Beschreibung:

Eigenschaften:

`public GameObjectInfo Info`

`public World World`

`public float Alpha`

`public Color BaseColor`

`public Color HighlightColor`

`public float HighlightIntensity`

`public GameModelInfo Info`

IGameObject
+ Info : GameObjectInfo + World : World + Alpha : float + BaseColor : Color + HighlightColor : Color + HighlightIntensity : float + Info : GameModelInfo + Model : XNA.Model + World : World + WorldMatrix : Matrix
+ Center (Vector3) : Vector3 + Update (GameTime) : void + Draw (GameTime) : void + Intersects (Ray, GameObjectDistance Ray) : GameObjectDistance + Center (Vector3) : Vector3 + Update (GameTime) : void + Draw (GameTime) : void + Intersects (Ray, GameObjectDistance Ray) : GameObjectDistance + GameModel (GameScreen, GameModelInfo) : void

```
public XNA.Model Model
```

```
public World World
```

```
public Matrix WorldMatrix
```

Methoden:

```
public Vector3 Center (Vector3)
```

```
public void Update (GameTime)
```

```
public void Draw (GameTime)
```

```
public GameObjectDistance Intersects (Ray, GameObjectDistance Ray)
```

```
public Vector3 Center (Vector3)
```

```
public void Update (GameTime)
```

```
public void Draw (GameTime)
```

```
public GameObjectDistance Intersects (Ray, GameObjectDistance Ray)
```

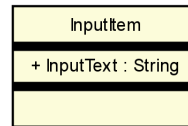
```
public void GameModel (GameScreen, GameModelInfo)
```

3.1.37 Klasse InputItem

Beschreibung:

Eigenschaften:

`public String InputText`

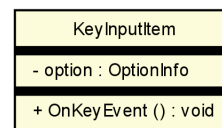


3.1.38 Klasse KeyInputItem

Beschreibung:

Eigenschaften:

`private OptionInfo option`



Methoden:

`public void OnKeyEvent ()`

3.1.39 Klasse Knot

Beschreibung:

Eigenschaften:

`public String Name`

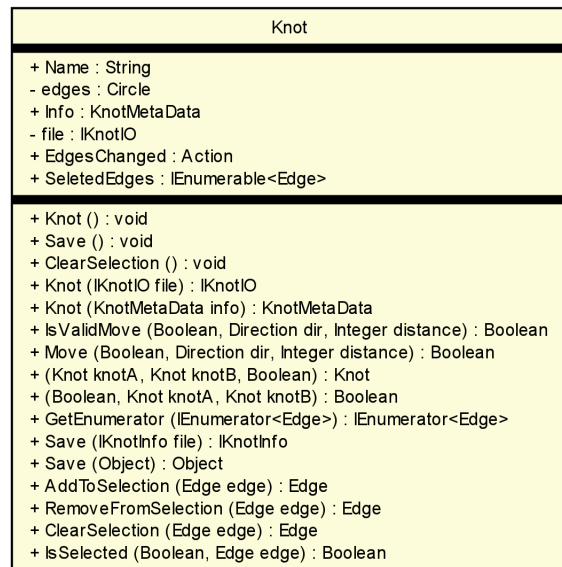
`private Circle edges`

`public KnotMetaData Info`

`private IKnotIO file`

`public Action EdgesChanged`

`public IEnumerable<Edge> SeletedEdges`



Methoden:

```
public void Knot ()
```

```
public void Save ()
```

```
public void ClearSelection ()
```

```
public IKnotIO Knot (IKnotIO file)
```

```
public KnotMetaData Knot (KnotMetaData info)
```

```
public Boolean IsValidMove (Boolean, Direction dir, Integer distance)
```

```
public Boolean Move (Boolean, Direction dir, Integer distance)
```

```
public Knot (Knot knotA, Knot knotB, Boolean)
```

```
public Boolean (Boolean, Knot knotA, Knot knotB)
```

```
public IEnumerator<Edge> GetEnumerator (IEnumerator<Edge>)
```

```
public IKnotInfo Save (IKnotInfo file)
```

```
public Object Save (Object)
```

```
public Edge AddToSelection (Edge edge)
```

```
public Edge RemoveFromSelection (Edge edge)
```

```
public Edge ClearSelection (Edge edge)
```

```
public Boolean IsSelected (Boolean, Edge edge)
```

3.1.40 Klasse Knot3Game

Beschreibung:

Eigenschaften:

```
public bool IsFullScreen
```

```
public Stack<GameScreen> Screens
```

```
public bool VSync
```

```
public GraphicsDeviceManager Graphics
```

Knot3Game
+ IsFullScreen : bool + Screens : Stack<GameScreen> + VSync : bool + Graphics : GraphicsDeviceManager
+ Game () : void + Initialize () : void + LoadContent () : void + UnloadContent () : void + Draw (GameTime) : void + Game (GameTime) : void

Methoden:

```
public void Game ()
```

```
public void Initialize ()
```

```
public void LoadContent ()
```

```
public void UnloadContent ()
```

```
public void Draw (GameTime)
```

```
public void Game (GameTime)
```

3.1.41 Klasse KnotFileIO

Beschreibung:

Eigenschaften:

`public IEnumerable<Edge> Edges`

`public String Name`

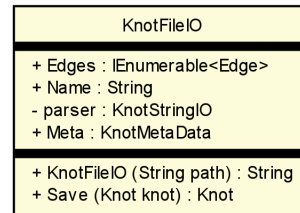
`private KnotStringIO parser`

`public KnotMetaData Meta`

Methoden:

`public String KnotFileIO (String path)`

`public Knot Save (Knot knot)`



3.1.42 Klasse KnotMetaData

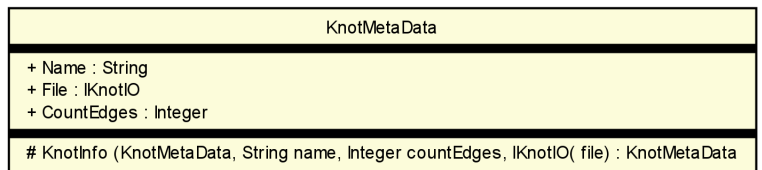
Beschreibung:

Eigenschaften:

`public String Name`

`public IKnotIO File`

`public Integer CountEdges`



Methoden:

`protected KnotMetaData KnotInfo (KnotMetaData, String name, Integer countEdges, IKnotIO(file)`

3.1.43 Klasse KnotStringIO

Beschreibung:

Eigenschaften:

`public String Name`

`public IEnumerable<Edge> Edges`

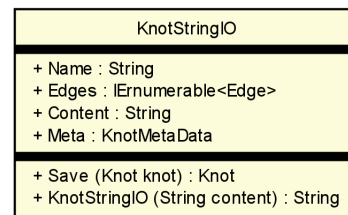
`public String Content`

`public KnotMetaData Meta`

Methoden:

`public Knot Save (Knot knot)`

`public String KnotStringIO (String content)`

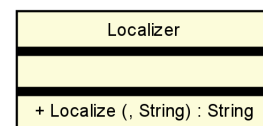


3.1.44 Klasse Localizer

Beschreibung:

Methoden:

`public String Localize (, String)`



3.1.45 Klasse Menu

Beschreibung:

Eigenschaften:

`public String Name`

`public Func<int, Vector2> RelativeItemSize`

`public Func<int, Vector2> RelativeItemPosition`

`public Func<ItemState, Vector2> ItemForegroundColor`

`public Func<ItemState, Vector2> ItemBackgroundColor`

`public HorizontalAlignment ItemAlignX`

`public VerticalAlignment ItemAlignY`

Methoden:

`public void Add (MenuItem, MenuItem)`

`public void Delete (MenuItem, MenuItem)`

`public void GetChild (int, int)`

`public int Size (int)`

`public IEnumerator<MenuItem> GetEnumerator (IEnumerator<MenuItem>)`

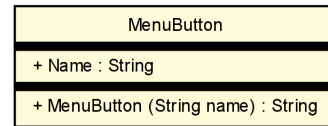
Menu
+ Name : String + RelativeItemSize : Func<int, Vector2> + RelativeItemPosition : Func<int, Vector2> + ItemForegroundColor : Func<ItemState, Vector2> + ItemBackgroundColor : Func<ItemState, Vector2> + ItemAlignX : HorizontalAlignment + ItemAlignY : VerticalAlignment
+ Add (MenuItem, MenuItem) : void + Delete (MenuItem, MenuItem) : void + GetChild (int, int) : void + Size (int) : int + GetEnumerator (IEnumerator<MenuItem>) : IEnumerator<MenuItem>

3.1.46 Klasse MenuButton

Beschreibung:

Eigenschaften:

`public String Name`



Methoden:

`public String MenuButton (String name)`

3.1.47 Klasse MenuItem

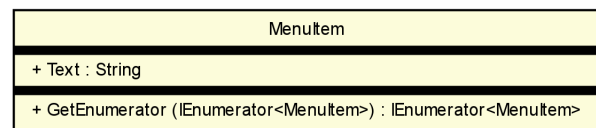
Beschreibung:

Eigenschaften:

`public ItemState ItemState`

`public int ItemOrder`

`public String Text`



Methoden:

`public void OnLeftClick ()`

`public void OnRightClick ()`

`public void OnKeyEvent ()`

`public Rectangle Bounds (Rectangle)`

3.1.48 Klasse MenuItem

Beschreibung:

Eigenschaften:

`public String Text`

MenuItem
+ Text : String
+ GetEnumerator (IEnumerator<MenuItem>) : IEnumerator<MenuItem>

Methoden:

`public IEnumerator<MenuItem> GetEnumerator (IEnumerator<MenuItem>)`

3.1.49 Klasse MenuScreen

Beschreibung:

Methoden:

`public void Update ()`

`public GameScreen Update (GameTime, GameState, GameScreen previousScreen) : GameScreen`

MenuScreen
+ Update () : void
+ Update (GameTime, GameState, GameScreen previousScreen) : GameScreen

3.1.50 Klasse ModelFactory

Beschreibung:

Eigenschaften:

`private Dictionary<GameModelInfo, GameModel> cache`

`private Func<GameState, GameModelInfo, GameModel> createModel`

ModelFactory
- cache : Dictionary<GameModelInfo, GameModel>
- createModel : Func<GameState, GameModelInfo, GameModel>
+ this (, , GameModel) : GameModel
+ ModelFactory (Func<GameState, GameModelInfo, >) : void

Methoden:

`public GameModel this (, , GameModel)`

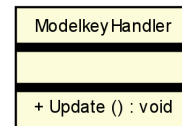
`public void ModelFactory (Func<GameState, GameModelInfo, >)`

3.1.51 Klasse ModelkeyHandler

Beschreibung:

Methoden:

`public void Update ()`

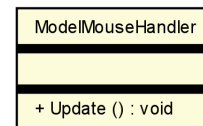


3.1.52 Klasse ModelMouseHandler

Beschreibung:

Methoden:

`public void Update ()`



3.1.53 Klasse ModelRenderer

Beschreibung:

Eigenschaften:

`public GameObjectInfo Info`

`public World World`

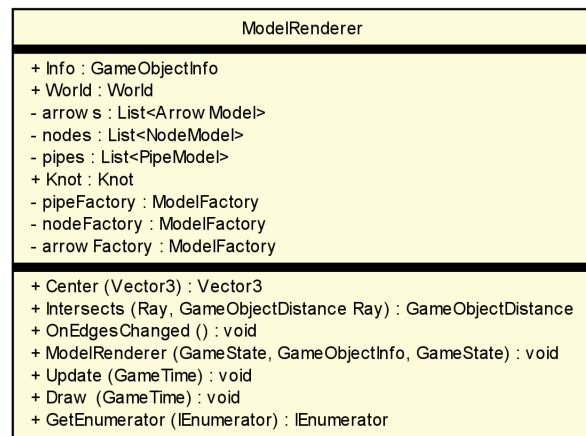
`private List<ArrowModel> arrows`

`private List<NodeModel> nodes`

`private List<PipeModel> pipes`

`public Knot Knot`

`private ModelFactory pipeFactory`



`private ModelFactory nodeFactory`

`private ModelFactory arrowFactory`

Methoden:

`public Vector3 Center (Vector3)`

`public GameObjectDistance Intersects (Ray, GameObjectDistance Ray)`

`public void OnEdgesChanged ()`

`public void ModelRenderer (GameState, GameObjectInfo, GameState)`

`public void Update (GameTime)`

`public void Draw (GameTime)`

`public IEnumerator GetEnumerator (IEnumerator)`

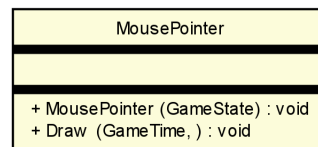
3.1.54 Klasse MousePointer

Beschreibung:

Methoden:

`public void MousePointer (GameState)`

`public void Draw (GameTime,)`



3.1.55 Klasse NodeModel

Beschreibung:

Eigenschaften:

`public NodeModelInfo Info`

Methoden:

`public void NodeModel (GameScreen, NodeModelInfo)`

`public void Draw (GameTime)`

`public void Update (GameTime)`

NodeModel
+ Info : NodeModelInfo
+ NodeModel (GameScreen, NodeModelInfo) : void + Draw (GameTime) : void + Update (GameTime) : void

3.1.56 Klasse NodeModelInfo

Beschreibung:

Eigenschaften:

`public void EdgeFrom`

`public void EdgeTo`

`public void Knot`

`public Vector3 EdgeFrom`

NodeModelInfo
+ EdgeFrom : void + EdgeTo : void + Knot : void + EdgeFrom : Vector3
+ NodeModelInfo (EdgeList, Edge, Edge) : void

Methoden:

`public void NodeModelInfo (EdgeList, Edge, Edge)`

3.1.57 Klasse OptionInfo

Beschreibung:

Eigenschaften:

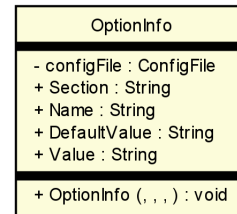
`private ConfigFile configFile`

`public String Section`

`public String Name`

`public String DefaultValue`

`public String Value`



Methoden:

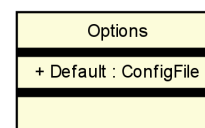
`public void OptionInfo (, , ,)`

3.1.58 Klasse Options

Beschreibung:

Eigenschaften:

`public ConfigFile Default`

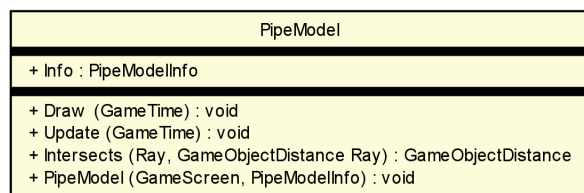


3.1.59 Klasse PipeModel

Beschreibung:

Eigenschaften:

`public PipeModelInfo Info`



Methoden:

```
public void Draw (GameTime)
```

```
public void Update (GameTime)
```

```
public GameObjectDistance Intersects (Ray, GameObjectDistance Ray)
```

```
public void PipeModel (GameScreen, PipeModelInfo)
```

3.1.60 Klasse PipeModelInfo

Beschreibung:

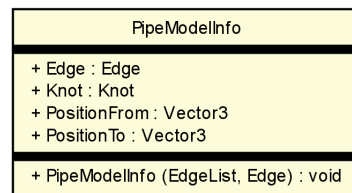
Eigenschaften:

```
public Edge Edge
```

```
public Knot Knot
```

```
public Vector3 PositionFrom
```

```
public Vector3 PositionTo
```



Methoden:

```
public void PipeModelInfo (EdgeList, Edge)
```

3.1.61 Klasse PipeMovement

Beschreibung:

Eigenschaften:

`public GameObjectInfo Info`

`public Knot Knot`

`public World World`

PipeMovement
+ Info : GameObjectInfo + Knot : Knot + World : World
+ Center (Vector3) : Vector3 + Intersects (Ray, GameObjectDistance Ray) : GameObjectDistance + Update (GameTime) : void + PipeMovement (GameState, World, GameObjectInfo) : void + GetEnumerator (IEnumerator) : IEnumerator + Draw (GameTime) : void

Methoden:

`public Vector3 Center (Vector3)`

`public GameObjectDistance Intersects (Ray, GameObjectDistance Ray)`

`public void Update (GameTime)`

`public void PipeMovement (GameState, World, GameObjectInfo)`

`public IEnumerator GetEnumerator (IEnumerator)`

`public void Draw (GameTime)`

3.1.62 Klasse ProfileSettingsScreen

Beschreibung:

Eigenschaften:

`protected void settingsMenu`

ProfileSettingsScreen
settingsMenu : void
+ Update () : void + Update (GameTime, GameScreen previousScreen) : GameScreen

Methoden:

```
public void Update ()
```

```
public GameScreen Update (GameTime, GameScreen previousScreen)
```

3.1.63 Klasse RenderEffect

Beschreibung:

Eigenschaften:

```
public RenderTarget2D RenderTarget
```

```
protected GameScreen screen
```

```
protected SpriteBatch spriteBatch
```

RenderEffect
+ RenderTarget : RenderTarget2D # screen : GameScreen # spriteBatch : SpriteBatch
+ Begin (GameTime) : void + End (GameTime) : void + Draw Model (GameModel, GameTime) : void + RemapModel (GameModel) : void # Draw RenderTarget (GameTime) : void

Methoden:

```
public void Begin (GameTime)
```

```
public void End (GameTime)
```

```
public void DrawModel (GameModel, GameTime)
```

```
public void RemapModel (GameModel)
```

```
protected void DrawRenderTarget (GameTime)
```

3.1.64 Klasse RenderEffectStack

Beschreibung:

Eigenschaften:

`public IRenderEffect CurrentEffect`

`private IRenderEffect DefaultEffect`

RenderEffectStack
+ CurrentEffect : IRenderEffect - DefaultEffect : IRenderEffect
+ () : void + (IRenderEffect) : void + RenderEffectStack () : void

Methoden:

`public void ()`

`public void (IRenderEffect)`

`public void RenderEffectStack ()`

3.1.65 Klasse SettingsScreen

Beschreibung:

Eigenschaften:

`protected void navigation`

SettingsScreen
navigation : void
+ Update () : void + Update (GameTime, GameScreen previousScreen) : GameScreen

Methoden:

`public void Update ()`

`public GameScreen Update (GameTime, GameScreen previousScreen)`

3.1.66 Klasse ShadowGameModel

Beschreibung:

Eigenschaften:

`public Color ShadowColor`

`public float ShadowAlpha`

Shadow GameModel
+ Shadow Color : Color + Shadow Alpha : float
+ Shadow GameModel (GameState, GameModel) : void + Draw (GameTime) : void

Methoden:

`public void ShadowGameModel (GameState, GameModel)`

`public void Draw (GameTime)`

3.1.67 Klasse ShadowGameObject

Beschreibung:

Eigenschaften:

`public GameObjectInfo Info`

`public World World`

`public Vector3 ShadowPosition`

`public Vector3 OriginalPosition`

Shadow GameObject
+ Info : GameObjectInfo + World : World + Shadow Position : Vector3 + OriginalPosition : Vector3
+ Center (Vector3) : Vector3 + Update (GameTime) : void + Draw (GameTime) : void + Intersects (Ray, GameObjectDistance Ray) : GameObjectDistance + Shadow GameObject (GameState, IGameObject) : void

Methoden:

`public Vector3 Center (Vector3)`

`public void Update (GameTime)`

`public void Draw (GameTime)`

```
public GameObjectDistance Intersects (Ray, GameObjectDistance Ray)
```

```
public void ShadowGameObject (GameState, IGameObject)
```

3.1.68 Klasse SliderItem

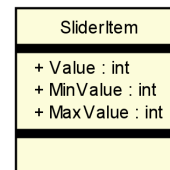
Beschreibung:

Eigenschaften:

```
public int Value
```

```
public int MinValue
```

```
public int MaxValue
```



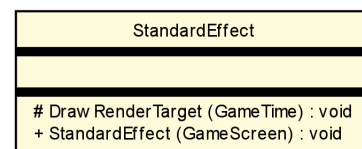
3.1.69 Klasse StandardEffect

Beschreibung:

Methoden:

```
protected void DrawRenderTarget (GameTime)
```

```
public void StandardEffect (GameScreen)
```

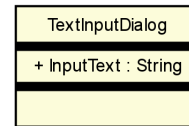


3.1.70 Klasse TextInputDialog

Beschreibung:

Eigenschaften:

`public String InputText`

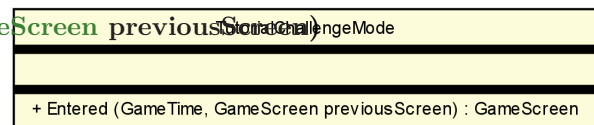


3.1.71 Klasse TutorialChallengeMode

Beschreibung:

Methoden:

`public GameScreen Entered (GameTime, GameScreen previousScreen)`



3.1.72 Klasse Widget

Beschreibung:

Eigenschaften:

`public Vector2 RelativeSize`

`public Vector2 RelativePosition`

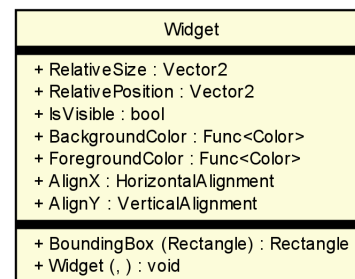
`public bool IsVisible`

`public Func<Color> BackgroundColor`

`public Func<Color> ForegroundColor`

`public HorizontalAlignment AlignX`

`public VerticalAlignment AlignY`



Methoden:

```
public Rectangle BoundingBox (Rectangle)
```

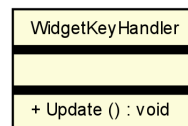
```
public void Widget (, )
```

3.1.73 Klasse WidgetKeyHandler

Beschreibung:

Methoden:

```
public void Update ()
```

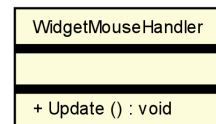


3.1.74 Klasse WidgetMouseHandler

Beschreibung:

Methoden:

```
public void Update ()
```



3.1.75 Klasse World

Beschreibung:

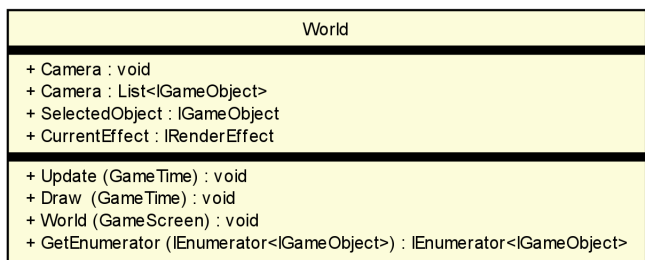
Eigenschaften:

```
public void Camera
```

```
public List<IGameObject> Camera
```

```
public IGameObject SelectedObject
```

```
public IRenderEffect CurrentEffect
```



Methoden:

```
public void Update (GameTime)
```

```
public void Draw (GameTime)
```

```
public void World (GameScreen)
```

```
public IEnumerator<IGameObject> GetEnumerator (IEnumerator<IGameObject>)
```

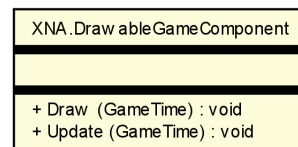
3.1.76 Klasse XNA.DrawableGameComponent

Beschreibung:

Methoden:

```
public void Draw (GameTime)
```

```
public void Update (GameTime)
```



3.1.77 Klasse XNA.Game

Beschreibung:

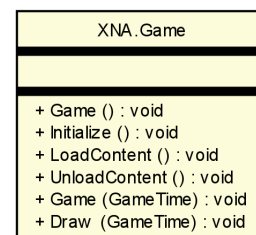
Methoden:

```
public void Game ()
```

```
public void Initialize ()
```

```
public void LoadContent ()
```

```
public void UnloadContent ()
```




```
public void Game (GameTime)
```

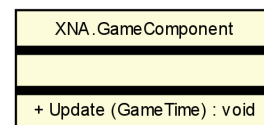
```
public void Draw (GameTime)
```

3.1.78 Klasse XNA.GameComponent

Beschreibung:

Methoden:

```
public void Update (GameTime)
```



3.2 Schnittstellen

3.2.1 Schnittstelle IChallengeIO

Beschreibung:

Eigenschaften:

```
public IEnumerable<KeyValuePair<String, Integer>> Highscore
```

```
public String Name
```

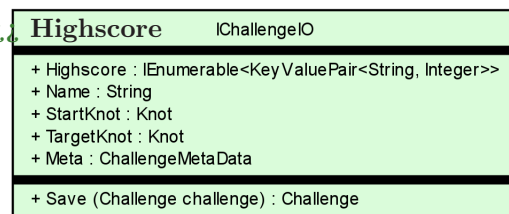
```
public Knot StartKnot
```

```
public Knot TargetKnot
```

```
public ChallengeMetaData Meta
```

Methoden:

```
public Challenge Save (Challenge challenge)
```

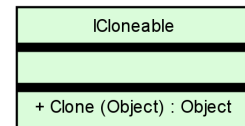


3.2.2 Schnittstelle ICloneable

Beschreibung:

Methoden:

```
public Object Clone (Object)
```

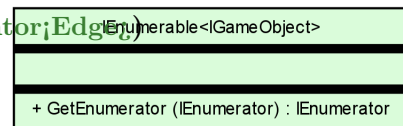


3.2.3 Schnittstelle IEnumerable

Beschreibung:

Methoden:

```
public IEnumerator<Edge> GetEnumerator (IEnumerator<Edge>)
```

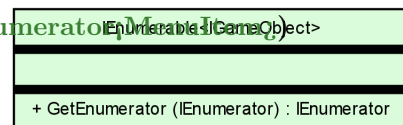


3.2.4 Schnittstelle IEnumerable

Beschreibung:

Methoden:

```
public IEnumerator<MenuItem> GetEnumerator (IEnumerator<MenuItem>)
```

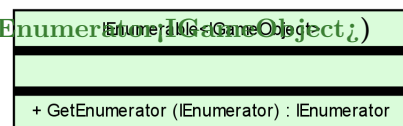


3.2.5 Schnittstelle IEnumerable<IGameObject>

Beschreibung:

Methoden:

```
public IEnumerator<IGameObject> GetEnumerator (IEnumerator<IGameObject>)
```



3.2.6 Schnittstelle IEnumerable<IGameObject>

Beschreibung:

Methoden:

public IEnumerator GetEnumerator (IEnumerator)

IEnumerable<IGameObject>
+ GetEnumerator (IEnumerator) : IEnumerator

3.2.7 Schnittstelle IGameObject

Beschreibung:

Eigenschaften:

public GameObjectInfo Info

public World World

Methoden:

public Vector3 Center (Vector3)

public void Update (GameTime)

public void Draw (GameTime)

public GameObjectDistance Intersects (Ray, GameObjectDistance Ray)

IGameObject
+ Info : GameObjectInfo + World : World + Alpha : float + BaseColor : Color + HighlightColor : Color + HighlightIntensity : float + Info : GameModelInfo + Model : XNA.Model + World : World + WorldMatrix : Matrix
+ Center (Vector3) : Vector3 + Update (GameTime) : void + Draw (GameTime) : void + Intersects (Ray, GameObjectDistance Ray) : GameObjectDistance + Center (Vector3) : Vector3 + Update (GameTime) : void + Draw (GameTime) : void + Intersects (Ray, GameObjectDistance Ray) : GameObjectDistance + GameModel (GameScreen, GameModelInfo) : void

3.2.8 Schnittstelle IGameStateComponent

Beschreibung:

Eigenschaften:

`public DisplayLayer Index`

`public GameScreen State`

IGameStateComponent
+ Index : DisplayLayer + State : GameScreen
+ SubComponents (GameTime, IEnumerable) : IEnumerable

Methoden:

`public IEnumerable SubComponents (GameTime, IEnumerable)`

3.2.9 Schnittstelle IJunction

Beschreibung:

Eigenschaften:

`public void EdgeFrom`

`public void EdgeTo`

Ijunction
+ EdgeFrom : void + EdgeTo : void

3.2.10 Schnittstelle IKeyEventListe- ner

Beschreibung:

Eigenschaften:

`public DisplayLayer Index`

`public bool IsKeyEventEnabled`

`public List ValidKeys`

IKeyListener
+ Index : DisplayLayer + IsKeyEventEnabled : bool + ValidKeys : List
+ OnKeyEvent () : void

Methoden:

```
public void OnKeyEvent ()
```

3.2.11 Schnittstelle IKnotIO

Beschreibung:

Eigenschaften:

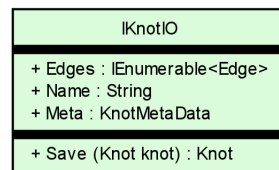
```
public IEnumerable<Edge> Edges
```

```
public String Name
```

```
public KnotMetaData Meta
```

Methoden:

```
public Knot Save (Knot knot)
```



3.2.12 Schnittstelle IMouseEventListener

Beschreibung:

Eigenschaften:

```
public DisplayLayer Index
```

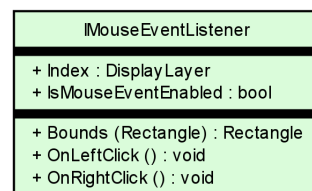
```
public bool IsMouseEventEnabled
```

Methoden:

```
public Rectangle Bounds (Rectangle)
```

```
public void OnLeftClick ()
```

```
public void OnRightClick ()
```



3.2.13 Schnittstelle IRenderEffect

Beschreibung:

Eigenschaften:

`public RenderTarget2D RenderTarget`

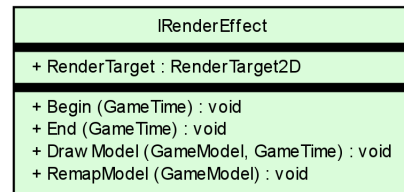
Methoden:

`public void Begin (GameTime)`

`public void End (GameTime)`

`public void DrawModel (GameModel, GameTime)`

`public void RemapModel (GameModel)`



3.2.14 Schnittstelle XNA.IGameComponent

Beschreibung:

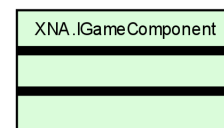
Eigenschaften:

`public bool IsMovable`

`public bool IsSelectable`

`public bool IsVisible`

`public Vector3 Position`



Methoden:

```
public bool Equals (GameObjectInfo, bool GameObjectInfo)
```

```
public bool Equals (GameObjectInfo, bool)
```

3.3 Enumerationen

Kapitel 4

Abläufe

4.1 Sequenzdiagramme

Kapitel 5

Klassenindex

Kapitel 6

Anmerkungen

Kapitel 7

Glossar

Test Test

7.1 Fachausdrücke

Test (Test-Beschreibung) ... 50

7.2 Abkürzungen

Test Test 50