ENTWURFSDOKUMENT

(V. 1.0)

$\begin{array}{c} \mathbf{KNOT^3} \\ \mathrm{PSE~WS~2013/14} \end{array}$

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

16. Dezember 2013

Inhaltsverzeichnis

1	Ein	leitung		4
2	Auf 2.1 2.2 2.3	Archite Klasser	ektur	5 6 6
3	Kla	ssenüb	ersicht	7
	3.1	Klassei	n	7
		3.1.1	Klasse Angles3	7
			Klasse ArrowModel	8
		3.1.3	Klasse ArrowModelInfo	8
		3.1.4	Klasse AudioSettingsScreen	9
		3.1.5	Klasse BooleanOptionInfo	9
		3.1.6	Klasse Camera	9
		3.1.7	Klasse CelShadingEffect	11
		3.1.8	Klasse Challenge	11
		3.1.9	Klasse ChallengeFileIO	13
		3.1.10	Klasse ChallengeLoadScreen	13
				13
				14
		3.1.13	Klasse CheckBoxItem	16
		3.1.14	Klasse Circle	16
				17
		3.1.16	Klasse ColorPickItem	17
		3.1.17	Klasse ConfigFile	18
		3.1.18	Klasse ConfirmDialog	18
				19
		3.1.20	Klasse CreativeLoadScreen	19
		3.1.21	Klasse CreativeModeScreen	19
		3.1.22	Klasse CreditsScreen	20
				20
		3.1.24	Klasse DistinctOptionInfo	21
				22
		3.1.26	Klasse DropDownEntry	22
			1	22
		3.1.28	Klasse Edge	23
		3.1.29	Klasse FadeEffect	24
		3.1.30	Klasse FileUtility	24

	Klasse GameModel	25
3.1.32	Klasse GameModelInfo	26
	Klasse GameObjectInfo	26
	Klasse GameScreen	27
3.1.35	Klasse GameScreenComponent	28
3.1.36	Klasse GraphicsSettingsScreen	29
3.1.37	Klasse Input	29
3.1.38	Klasse InputItem	30
3.1.39	Klasse KeyInputItem	30
	Klasse Knot	31
	Klasse Knot3Game	33
3.1.42	Klasse KnotFileIO	33
3.1.43	Klasse KnotInputHandler	34
3.1.44	Klasse KnotMetaData	34
3.1.45	Klasse KnotRenderer	35
3.1.46	Klasse KnotStringIO	36
3.1.47	Klasse Localizer	37
	Klasse Menu	37
	Klasse MenuButton	38
3.1.50	Klasse MenuItem	38
3.1.51	Klasse MenuScreen	39
	Klasse ModelFactory	39
	Klasse ModelMouseHandler	40
3.1.54	Klasse MousePointer	40
	Klasse NodeMap	40
	Klasse NodeModel	41
	Klasse NodeModelInfo	41
3.1.58	Klasse OptionInfo	42
	Klasse Options	43
	Klasse PauseDialog	43
	Klasse PipeModel	43
	Klasse PipeModelInfo	43
	Klasse PipeMovement	44
	Klasse PrinterIO	45
	Klasse ProfileSettingsScreen	45
	Klasse RenderEffect	46
	Klasse RenderEffectStack	
	Klasse SettingsScreen	
3.1.69	Klasse ShadowGameModel	48
	Klasse ShadowGameObject	48
	Klasse SliderItem	49
	Klasse StandardEffect	49
	Klasse StartScreen	50
	Klasse TextInputDialog	50
	Klasse TutorialChallengeMode	50
	Klasse VerticalMenu	50
	Klasse Widget	51
	Klasse WidgetKeyHandler	51
3.1.79	Klasse WidgetMouseHandler	52
	Klasse World	52
	Klasse XNA DrawableGameComponent	53

		3.1.82	Klasse XNA.Game	53
		3.1.83	Klasse XNA.GameComponent	53
	3.2	Schnit	tstellen	
		3.2.1	Schnittstelle IChallengeIO	54
		3.2.2	Schnittstelle ICloneable	54
		3.2.3	Schnittstelle IEnumerable	54
		3.2.4	Schnittstelle IEnumerable1	55
		3.2.5	Schnittstelle IEquatable	55
		3.2.6	Schnittstelle IEquatable 1	
		3.2.7	Schnittstelle IGameObject	
		3.2.8	Schnittstelle IGameScreenComponent	
		3.2.9	Schnittstelle IJunction	
		3.2.10	Schnittstelle IKeyEventListener	
		3.2.11	Schnittstelle IKnotIO	
			Schnittstelle IMouseEventListener	
			Schnittstelle IRenderEffect	
	3.3	Enume	erationen	59
4	Abl			60
	4.1	Sequer	zdiagramme	60
5	Klas	ssenind	lex	61
6	Ann	nerkun	gen	62
7	Glos	ssar		63

Kapitel 1

Einleitung

Das Knobel- und Konstruktionsspiel Knot^3 , 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 Render Effect und die von ihr abgeleiteten Klassen gekapselt. Ein Render Effect enthält ein Render target vom Typ Render Target 2D 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 Angles3

Beschreibung:

Diese Klasse repräsentiert die Rotationswinkel der drei Achsen.

Eigenschaften:

public float X

Der Rotationswinkel um die X-Achse.

public float Y

Der Rotationswinkel um die Y-Achse.

public float Z

Der Rotationswinkel um die Z-Achse.

public Angles3 Zero

Eine statische Property mit dem Wert X = 0, Y = 0, Z = 0.

Konstruktoren:

public Angles3 (float X, float Y, float Z)

Konstruiert ein neues Angles3-Objekt mit drei gegebenen Winkeln.

Methoden:

public Angles3 FromDegrees (float X, float Y, float Z)

Konvertiert Grad in Bogenmaß.

public void ToDegrees (float X, float Y, float Z)

Konvertiert Bogenmaß in Grad.

Angles3

- + X : float + Y : float
- + 7 . 110at
- + Zero : Angles3
- + FromDegrees (float X, float Y, float Z): Angles3
- + Angles3 (float X, float Y, float Z) : void
- + To Degrees (float X, float Y, float Z) : void

3.1.2 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.

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

Konstruktoren:

public ArrowModel (GameScreen screen, ArrowModelInfo info)

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

Methoden:

public void Draw (GameTime GameTime)

Zeichnet den Pfeil.

public GameObjectDistance Intersects (Ray ray)

Überprüft, ob der Mausstrahl den Pfeil schneidet.

public void Update (GameTime GameTime)

Wird für jeden Frame aufgerufen.

3.1.3 Klasse ArrowModelInfo

Beschreibung:

Ein Objekt dieser Klasse hält alle Informationen hält, 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 position, Vector3 direction) : void

Konstruktoren:

public ArrowModelInfo (Vector3 position, Vector3 direction)

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

3.1.4 Klasse AudioSettingsScreen

Beschreibung:

Der Spielzustand, der die Audio-Einstellungen darstellt.

Eigenschaften:

protected void settingsMenu

Das Menu, das die Einstellungen enthält.

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)

Fügt das Menu mit dem Einstellungen in die Spielkomponentenliste ein.

3.1.5 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 (String section, String name, String defaultValue, ConfigFile configFile) : void

AudioSettingsScreen

+ Entered (GameScreen previousScreen, GameTime GameTime): void

settingsMenu : void

+ Update (GameTime time) : void

Konstruktoren:

public BooleanOptionInfo (String section, String name, String defaultValue, ConfigFile configFile)

Erstellt eine neue Option, die die Werte "Wahr "oder "Falsch "annehmen kann, mit dem angegebenen Namen in dem angegebenen Abschnitt der angegebenen Einstellungsdatei.

3.1.6 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.

public Vector3 Position

Die Position der Kamera.

public Vector3 Target

Das Ziel der Kamera.

public float FoV

Das Sichtfeld.

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

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 BoundingFrustum ViewFrustum

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

private World World

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

public Angles3 Rotation

Die Rotationswinkel.

Konstruktoren:

public Camera (GameScreen screen, World world)

Erstellt eine neue Kamera in einem bestimmten GameScreen für eine bestimmte Spielwelt.

Camera

- + Position : Vector3
- + Target : Vector3 + FoV : float
- + View Matrix : Matrix
- + WorldMatrix: Matrix
- + ProjectionMatrix: Matrix
- + ArcballTarget : Vector3
- + View Frustum : BoundingFrustum
- World : World
- + Rotation : Angles3
- + TargetDirection (): Vector3
- + TargetDistance () : float
- + Camera (GameScreen screen, World world): void
- + Update (GameTime GameTime) : void
- + GetMouseRay (Vector2 mousePosition) : Ray

Methoden:

public Vector3 TargetDirection ()

Die Blickrichtung.

public float TargetDistance ()

Der Abstand zwischen der Kamera und dem Kamera-Ziel.

public void Update (GameTime GameTime)

Wird für jeden Frame aufgerufen.

public Ray GetMouseRay (Vector2 mousePosition)

Berechnet einen Strahl für die angegebenene 2D-Mausposition.

3.1.7 Klasse CelShadingEffect

Beschreibung:

Ein Cel-Shading-Effekt.

Konstruktoren:

public CelShadingEffect (GameScreen screen)

Erstellt einen neuen Cel-Shading-Effekt für den angegebenen GameScreen.

Methoden:

protected void DrawRenderTarget (GameTime)

Zeichnet das Rendertarget mithilfe eines Outline-Shaders.

public void DrawModel (GameTime, GameModel GameModel)

Zeichnet ein 3D-Modell auf das Rendertarget.

public void RemapModel (GameModel GameModel)

Weist dem 3D-Modell den Cel-Shader zu.

3.1.8 Klasse Challenge

Beschreibung:

Ein Objekt dieser Klasse repräsentiert eine Challenge.

CelShadingEffect

- # Draw RenderTarget (GameTime) : void
- + Draw Model (GameTime, GameModel GameModel) : void
- + RemapModel (GameModel GameModel) : void
- + CelShadingEffect (GameScreen screen) : void

public Knot Start

Der Ausgangsknoten, den der Spieler in den Referenzknoten transformieren muss.

public Knot Target

Der Referenzknoten, in den der Spieler den Ausgangsknoten transformieren muss.

private SortedList;Integer, String; highscore

Challenge

- + Start : Knot
- + Target : Knot
- highscore : SortedList<Integer, String>
- + Name : String
- file : IChallengelO
- + Highscore : IEnumerator<Key Value Pair<String, Integer>>
- + Info : ChallengeMetaData
- + Challenge (ChallengeInfo info): Challenge
- + Challenge (IChallangeIO file) : Challenge
- + CreateChallenge (Knot start, Knot target, String name, IChallengelO file): Boolean
- + AddToHighscore (String name, Integer time): void

Eine sortierte Bestenliste.

private IChallengeIO format

Das Speicherformat der Challenge.

public IEnumerator; Key Value Pair; String, Integer; Highscore

Ein öffentlicher Enumerator, der die Bestenliste unabhängig von der darunterliegenden Datenstruktur zugänglich macht.

public ChallengeMetaData MetaData

Die Metadaten der Challenge.

public String Name

Der Name der Challenge.

Konstruktoren:

public Challenge (ChallengeMetaData meta)

Erstellt ein Challenge-Objekt aus einem gegebenenen Challenge-Metadaten-Objekt. Erstellt ein Challenge-Objekt aus einer gegebenenen Challenge-Datei.

public Challenge (IChallangeIO format)

Erstellt ein Challenge-Objekt aus einem gegebenenen Challenge-Metadaten-Objekt. Erstellt ein Challenge-Objekt aus einer gegebenenen Challenge-Datei.

Methoden:

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

Erstellt eine neue Challenge mit einem gegebenen Ausgangs- und Referenzknoten und einem Namen

public void AddToHighscore (String name, Integer time)

Fügt eine neue Bestzeit eines bestimmten Spielers in die Bestenliste ein.

3.1.9 Klasse ChallengeFileIO

Beschreibung:

Implementiert das Speicherformat für Challenges.

Eigenschaften:

private KnotStringIO startParser

Der Parser für den Ausgangsknoten im Challenge-Format.

private KnotStringIO targetParser

Der Parser für den Referenzknoten im Challenge-Format.

Konstruktoren:

public ChallengeFileIO ()

Erstellt ein ChallengeFileIO-Objekt.

Methoden:

public void Save (Challenge challenge)

Speichert eine Challenge in dem Dateinamen, der in dem Challenge-Objekt enthalten ist.

public Challenge Load (String filename)

Lädt eine Challenge aus einer angebenenen Datei.

public ChallengeMetaData LoadMetaData (String filename)

Lädt die Metadaten einer Challenge aus einer angegebenen Datei.

3.1.10 Klasse ChallengeLoadScreen

Beschreibung:

Der Spielzustand, der den Ladebildschirm für Challenges darstellt.

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)

Fügt das Menu mit dem Spielständen in die Spielkomponentenliste ein.

3.1.11 Klasse ChallengeMetaData

ChallengeFileIO

+ Highscore : IEnumerable<KeyValuePair<String, Integer>>

+ Name : String + StartKnot : Knot + TargetKnot : Knot startParser : KnotStringle

startParser : KnotStringIOtargetParser : KnotStringIO

+ ChallengeFileIO (String path): void

+ Save (Challenge challenge) : void + ChallengeFileIO (Challenge challenge) : void

Beschreibung:

Enthält Metadaten zu einer Challenge.

Eigenschaften:

public String Name

Der Name der Challenge.

public KnotMetaData Start

Der Ausgangsknoten, den der Spieler in den Referenzknoten transformieren muss.

public KnotMetaData Target

Der Referenzknoten, in den der Spieler den Ausgangsknoten transformieren muss.

public IChallengeIO Format

Das Format, aus dem die Metadaten der Challenge gelesen wurden oder null.

public String Filename

Der Dateiname, aus dem die Metadaten der Challenge gelesen wurden oder in den sie abgespeichert werden sollen.

public IEnumerator; Key Value Pair; String, Integer;; Highscore

Ein öffentlicher Enumerator, der die Bestenliste unabhängig von der darunterliegenden Datenstruktur zugänglich macht.

Konstruktoren:

public ChallengeMetaData (String name, KnotMetaData start, KnotMetaData target)

Erstellt ein Challenge-Metadaten-Objekt mit einem gegebenen Namen und den Metadaten des Ausgangs- und Referenzknotens.

+ Name: String + Start: KnotMetaData + Target: KnotMetaData + File: IChallengeIO + ChallengeMetaData (IChallengeIO file): void

3.1.12 Klasse ChallengeModeScreen

Beschreibung:

Der Spielzustand, der während dem Spielen einer Challenge aktiv ist und für den Ausgangs- und Referenzknoten je eine 3D-Welt zeichnet.

ChallengeLoadScreen

- + Update (GameTime time) : void
- + Entered (GameScreen previousScreen, GameTime GameTime): void

public void PlayerKnot

Der Spielerknoten, der während der Transformation des Spielers aus dem Ausgangsknoten entsteht.

public void ChallengeKnot

Der Referenzknoten.

private World ChallengeWorld

Die Spielwelt in der die 3D-Modelle des dargestellten Referenzknotens enthalten sind.

private World PlayerWorld

Die Spielwelt in der die 3D-Modelle des dargestellten Spielerknotens enthalten sind.

${\bf private\ Knot Renderer\ Challenge Knot Renderer}$

Der Controller, der aus dem Referenzknoten die 3D-Modelle erstellt.

private KnotRenderer PlayerKnotRenderer

Der Controller, der aus dem Spielerknoten die 3D-Modelle erstellt.

private PipeMovement PlayerKnotMovement

Der Inputhandler, der die Kantenverschiebungen des Spielerknotens durchführt.

public Stack; Knot; Undo

Der Undo-Stack.

public Stack; Knot; Redo

Der Redo-Stack.

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)

Fügt die 3D-Welten und den Inputhandler in die Spielkomponentenliste ein.

3.1.13 Klasse CheckBoxItem

Beschreibung:

Ein Menueintrag, der einen Auswahlkasten darstellt.

Eigenschaften:

private BooleanOptionInfo option

Die Option, die mit dem Auswahlkasten verknüft ist.

CheckBoxItem - option : BooleanOptionInfo + CheckBoxItem (BooleanOptionInfo option) : void

Konstruktoren:

public CheckBoxItem (BooleanOptionInfo option)

Erstellt einen Menueintrag für die angegebenen Option.

3.1.14 Klasse Circle

Beschreibung:

Eine doppelt verkettete Liste.

Eigenschaften:

public T Content

Der Wert dieses Listeneintrags.

public Circle Next

Der nächste Listeneintrag.

public Circle Previous

Der vorherige Listeneintrag.

Konstruktoren:

public Circle (T content)

Erstellt einen neuen Listeneintrag.

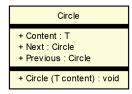
Methoden:

public void Remove ()

Entfernt diesen Listeneintrag und verknüft den vorherigen mit dem nächsten Eintrag.

public void InsertAfter (T next)

Fügt nach diesem Listeneintrag einen neuen Listeneintrag ein.



public void InsertBefore (T previous)

Fügt vor diesem Listeneintrag einen neuen Listeneintrag ein.

public IEnumerator; T; GetEnumerator ()

Gibt einen Enumerator über die Liste zurück.

public IEnumerator GetEnumerator ()

Gibt einen Enumerator über die Liste zurück.

3.1.15 Klasse ColorPicker

Beschreibung:

Ein Steuerelement der grafischen Benutzeroberfläche, das eine Auswahl von Farben ermöglicht.

Eigenschaften:

public Color Color

Die ausgewählte Farbe.

Methoden:

public void OnKeyEvent ()

Reagiert auf Tastatureingaben.

public Rectangle Bounds ()

Gibt die Ausmaße des ColorPickers zurück.

public void OnLeftClick (Vector2 position, ClickState state, GameTime time)

Bei einem Linksklick wird eine Farbe ausgewählt und im Attribut Color abgespeichert.

public void OnRightClick (Vector2 position, ClickState state, GameTime time)

Bei einem Rechtsklick geschieht nichts.

3.1.16 Klasse ColorPickItem

Beschreibung:

Ein Menueintrag, der eine aktuelle Farbe anzeigt und zum Ändern der Farbe per Mausklick einen ColorPicker öffnet.



public Color Color

Die aktuelle Farbe.

private ColorPicker picker

Der ColorPicker, der bei einem Mausklick auf den Menüeintrag geöffnet wird.

3.1.17 Klasse ConfigFile

Beschreibung:

Repräsentiert eine Einstellungsdatei.

angegebenen Wert.

Methoden:

public void SetOption (String section, String option, String value) ConfigFile

Setzt den Wert der Option mit dem angegebenen
Namen in den angegebenen Abschnitt auf den

+ SetOption (String section, String option, String value): void
+ GetOption (String section, String option, Boolean defaultValue): Boolean
+ GetOption (String section, String option, String defaultValue): String

public Boolean GetOption (String section, String option, Boolean defaultValue)

Gibt den aktuell in der Datei vorhandenen Wert für die angegebenen Option in dem angegebenen Abschnitt zurück.

public String GetOption (String section, String option, String defaultValue)

Gibt den aktuell in der Datei vorhandenen Wert für die angegebenen Option in dem angegebenen Abschnitt zurück.

public void SetOption (String section, String option, Boolean value)

Setzt den Wert der Option mit dem angegebenen Namen in den angegebenen Abschnitt auf den angegebenen Wert.

3.1.18 Klasse ConfirmDialog

Beschreibung:

Ein Dialog, der Schaltflächen zum Bestätigen einer Aktion anzeigt.



+ SetOption (String section, String option, Boolean _value) : void

private Menu buttons

Das Menü, das Schaltflächen enthält.

ConfirmDialog - buttons : Menu

3.1.19 Klasse ControlSettingsScreen

Beschreibung:

Der Spielzustand, der die Steuerungs-Einstellungen darstellt.

Eigenschaften:

protected void settingsMenu

Das Menu, das die Einstellungen enthält.

ControlSettingsScreen # settingsMenu : void + Update (GameTime time) : void + Entered (GameScreen previousScreen, GameTime GameTime) : void

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)

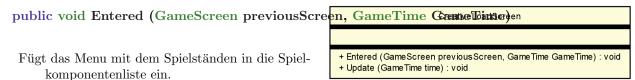
Fügt das Menu mit dem Einstellungen in die Spielkomponentenliste ein.

3.1.20 Klasse CreativeLoadScreen

Beschreibung:

Der Spielzustand, der den Ladebildschirm für Knoten darstellt.

Methoden:



public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

3.1.21 Klasse CreativeModeScreen

Beschreibung:

Der Spielzustand, der während dem Erstellen und Bearbeiten eines Knotens aktiv ist und für den Knoten eine 3D-Welt zeichnet.

public void Knot

Der Knoten, der vom Spieler bearbeitet wird.

private World World

Die Spielwelt in der die 3D-Objekte des dargestellten Knotens enthalten sind.

private KnotRenderer KnotRenderer

Der Controller, der aus dem Knoten die 3D-Modelle erstellt.

public Stack; Knot; Undo

Der Undo-Stack.

public Stack; Knot; Redo

Der Redo-Stack.

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime time)

Fügt die 3D-Welt und den Inputhandler in die Spielkomponentenliste ein.

3.1.22 Klasse CreditsScreen

Beschreibung:

Der Spielzustand, der die Auflistung der Mitwirkenden darstellt.

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

CreditsScreen

Fügt das Menu mit dem Mitwirkenden in die Spielkomponentenliste ein.

3.1.23 Klasse Dialog

Beschreibung:

Ein Dialog ist ein im Vordergrund erscheinendes Fenster, das auf Nutzerinteraktionen wartet.

public String Title

Der Fenstertitel.

public String Text

Der angezeigte Text.

Dialog + Name: String + Text: String + OnKeyEvent (): void + OnLeftClick (): void + OnRightClick (): void + Bounds (): Rectangle

Methoden:

public void OnKeyEvent ()

Durch Drücken der Entertaste wird die ausgewählte Aktion ausgeführt. Durch Drücken der Escape-Taste wird der Dialog abgebrochen. Mit Hilfe der Pfeiltasten kann zwischen den Aktionen gewechselt werden.

public Rectangle Bounds ()

Gibt die Ausmaße des Dialogs zurück.

public void OnLeftClick (Vector2 position, ClickState state, GameTime time)

Bei einem Linksklick geschieht nichts.

public void OnRightClick (Vector2 position, ClickState state, GameTime time)

Bei einem Rechtsklick geschieht nichts.

3.1.24 Klasse DistinctOptionInfo

Beschreibung:

Diese Klasse repräsentiert eine Option, die eine distinkte Werteliste annehmen kann.

Eigenschaften:

public HashSet;string; ValidValues

DistinctOptionInfo + ValidValues : HashSet<string> + Value : String + DistinctOptionInfo (String section, String name, String defaultValue, IEnumerable<string> validValue

public String Value

Ein Property, das den aktuell abgespeicherten Wert zurück gibt.

Konstruktoren:

public DistinctOptionInfo (String section, String name, String defaultValue, IEnumerable; string; validValue

Erstellt eine neue Option, die einen der angegebenen gültigen Werte annehmen kann, mit dem angegebenen Namen in dem angegebenen Abschnitt der angegebenen Einstellungsdatei.

3.1.25 Klasse DrawableGameScreenComponent

Beschreibung:

Eine zeichenbare Spielkomponente, die in einem angegebenen Spielzustand verwendet wird und eine bestimmte Priorität hat.

Eigenschaften:

public GameScreen Screen

Der zugewiesene Spielzustand.

public DisplayLayer Index

Die Zeichen- und Eingabepriorität.

Draw ableGameScreenComponent + Screen : GameScreen + Index : DisplayLayer

- + SubComponents (GameTime GameTime) : IEnumerable
- + Draw ableGameStateComponent (GameScreen screen, DisplayLayer index) : void

Methoden:

public IEnumerable SubComponents (GameTime GameTime)

Gibt Spielkomponenten zurück, die in dieser Spielkomponente enthalten sind.

public void DrawableGameStateComponent (GameScreen screen, DisplayLayer index)

Erstellt eine neue zeichenbare Spielkomponente in dem angegebenen Spielzustand mit der angegebenen Priorität.

3.1.26 Klasse DropDownEntry

Beschreibung:

Repräsentiert einen Eintrag in einem Dropdown-Menü.

Eigenschaften:

public String Text

Der Text des Eintrags.



3.1.27 Klasse DropDownMenuItem

Beschreibung:

Ein Menüeintrag, der den ausgewählten Wert anzeigt und bei einem Linksklick ein Dropdown-Menü zur Auswahl eines neuen Wertes ein- oder ausblendet.

private VerticalMenu dropdown

Das Dropdown-Menü, das ein- und ausgeblendet werden kann.

DropDow nMenultem - dropdow n : VerticalMenu + AddEntries (DistinctOptionInfo option) : void + AddEntries (DropDow nEntry enties) : void

Methoden:

public void AddEntries (DistinctOptionInfo option)

Fügt Einträge in das Dropdown-Menü ein, die auf Einstellungsoptionen basieren. Fügt Einträge in das Dropdown-Menü ein, die nicht auf Einstellungsoptionen basieren.

public void AddEntries (DropDownEntry enties)

Fügt Einträge in das Dropdown-Menü ein, die auf Einstellungsoptionen basieren. Fügt Einträge in das Dropdown-Menü ein, die nicht auf Einstellungsoptionen basieren.

3.1.28 Klasse Edge

Beschreibung:

Eine Kante eines Knoten, die aus einer Richtung und einer Farben sowie optional einer Liste von Flächennummern besteht.

Eigenschaften:

public Color Color

Die Farbe der Kante.

public Direction Direction

Die Richtung der Kante.

public List; int; Rectangles

Die Liste der Flächennummern, die an die Kante angrenzen.

Konstruktoren:

public Edge (Direction direction)

Erstellt eine neue Kante mit der angegebenen Richtung.

Methoden:

public Vector3 Get3DDirection ()

Gibt die Richtung als normalisierten Vektor3 zurück.

Edge + Color: Color + Direction: Direction + Rectangles: List<int> + Edge (Direction direction): void + Get3DDirection (): Vector3

3.1.29 Klasse FadeEffect

Beschreibung:

Ein Postprocessing-Effekt, der eine Überblendung zwischen zwei Spielzuständen darstellt.

Eigenschaften:

private Boolean IsFinished

Gibt an, ob die Überblendung abgeschlossen ist und das RenderTarget nur noch den neuen Spielzustand darstellt.

I	FadeEffect
	- lsFinished : Boolean - PreviousRenderTarget : RenderTarget2D
	+ FadeEffect (GameScreen new Screen, GameScreen oldScreen) : void # Draw RenderTarget (GameTime) : void

private RenderTarget2D PreviousRenderTarget

Der zuletzt gerenderte Frame im bisherigen Spielzustand.

Konstruktoren:

public FadeEffect (GameScreen newScreen, GameScreen oldScreen)

Erstellt einen Überblende-Effekt zwischen den angegebenen Spielzuständen.

Methoden:

protected void DrawRenderTarget (GameTime)

Zeichnet die Überblendung des zuletzt gerenderten Frames des bisherigen Spielzustands in das Rendertarget des aktuellen Spielzustands

3.1.30 Klasse FileUtility

Beschreibung:

Eine Hilfsklasse für Dateioperationen.

Eigenschaften:

public String SettingsDirectory

Das Einstellungsverzeichnis.

public String SavegameDirectory

Das Spielstandverzeichnis.

public String ScreenshotDirectory

Das Bildschirmfotoverzeichnis.



Methoden:

public String ConvertToFileName (String name)

Konvertiert einen Namen eines Knotens oder einer Challenge in einen gültigen Dateinamen durch Weglassen ungüliger Zeichen.

Klasse GameModel 3.1.31

Beschreibung:

Repräsentiert ein 3D-Modell in einer Spielwelt.

Eigenschaften:

public float Alpha

Die Transparenz des Modells.

public Color BaseColor

Die Farbe des Modells.

public Color HightlightColor

Die Auswahlfarbe des Modells.

public float HighlightIntensity

Die Intensität der Auswahlfarbe.

public GameModelInfo Info

Die Modellinformationen wie Position, Skalierung und der Dateiname des 3D-Modells.

public XNA.Model Model

Die Klasse des XNA-Frameworks, die ein 3D-Modell repräsentiert.

public World World

Die Spielwelt, in der sich das 3D-Modell befindet.

public Matrix WorldMatrix

Die Weltmatrix des 3D-Modells in der angegebenen Spielwelt.

Konstruktoren:

public GameModel (GameScreen screen, GameModelInfo info)

Erstellt ein neues 3D-Modell in dem angegebenen Spielzustand mit den angegebenen Modellinformationen.

GameModel

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

Methoden:

public Vector3 Center ()

Gibt die Mitte des 3D-Modells zurück.

public void Update (GameTime GameTime)

Wird für jeden Frame aufgerufen.

public void Draw (GameTime GameTime)

Zeichnet das 3D-Modell in der angegebenen Spielwelt mit dem aktuellen RenderEffekt der Spielwelt.

public GameObjectDistance Intersects (Ray Ray)

Überprüft, ob der Mausstrahl das 3D-Modell schneidet.

3.1.32 Klasse GameModelInfo

Beschreibung:

Enthält Informationen über ein 3D-Modell wie den Dateinamen, die Rotation und die Skalierung.

Eigenschaften:

public String Modelname

Der Dateiname des Modells.

public Angles3 Rotation

Die Rotation des Modells.

public Vector3 Scale

Die Skalierung des Modells.

Konstruktoren:

public GameModelInfo (String modelname, Angles3 rotation, Vector3 scale)

Erstellt ein neues Informations-Objekt eines 3D-Modells mit den angebenenen Informationen zu Dateiname, Rotation und Skalierung.

3.1.33 Klasse GameObjectInfo

Beschreibung:

Enthält Informationen über ein 3D-Objekt wie die Position, Sichtbarkeit, Verschiebbarkeit und Auswählbarkeit.

+ GameModelInfo (String modelname, Angles3 rotation) : void

public Boolean IsMovable

Die Verschiebbarkeit des Spielobjektes.

public Boolean IsSelectable

Die Auswählbarkeit des Spielobjektes.

public Boolean IsVisible

Die Sichtbarkeit des Spielobjektes.

public Vector3 Position

Die Position des Spielobjektes.

Methoden:

public Boolean Equals (GameObjectInfo GameObjectInfo)

Vergleicht zwei Informationsobjekte für Spielobjekte.

public Boolean Equals (T other)

Vergleicht zwei Informationsobjekte für Spielobjekte.

3.1.34 Klasse GameScreen

Beschreibung:

Ein Spielzustand, der zu einem angegebenen Spiel gehört und einen Inputhandler und Rendereffekte enthält.

Eigenschaften:

public Knot3Game Game

Das Spiel, zu dem der Spielzustand gehört.

public Input Input

Der Inputhandler des Spielzustands.

public RenderEffect PostProcessingEffect

Der aktuelle Postprocessing-Effekt des Spielzustands

${\bf public} \ {\bf Render Effect Stack} \ {\bf Current Render Effects}$

Ein Stack, der während dem Aufruf der Draw-Methoden der Spielkomponenten die jeweils aktuellen Rendereffekte enthält.

Game Object Info

- + IsMovable : Boolean + IsSelectable : Boolean + IsVisible : Boolean + Position : Vector3
- + Equals (GameObjectInfo GameObjectInfo) : Boolean
- + Equals (Tother): Boolean

GameScreen

- + Game : Knot3Game
- + Input : Input
- + PostProcessingEffect : RenderEffect
- + CurrentRenderEffects : RenderEffectStack
- + Entered (GameScreen previousScreen, GameTime time) : void
- + Before Exit (Game Screen next Screen, Game Time time) : void
- + Update (GameTime time) : void
- + GameScreen (Knot3Game game) : void
- $+ \, \mathsf{AddGameComponents} \, (\mathsf{IGameStateComponent[]} \, \mathsf{components}) : \mathsf{void} \,$
- + RemoveGameComponents (IGameStateComponent[] components) : void

Konstruktoren:

public GameScreen (Knot3Game game)

Methoden:

public void Entered (GameScreen previousScreen, GameTime time)

Beginnt mit dem Füllen der Spielkomponentenliste des XNA-Frameworks und fügt sowohl für Tastaturals auch für Mauseingaben einen Inputhandler für Widgets hinzu. Wird Unterklassen von GameScreen reimplementiert und fügt zursätzlich weitere Spielkomponenten hinzu.

public void BeforeExit (GameScreen nextScreen, GameTime time)

Leert die Spielkomponentenliste des XNA-Frameworks.

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void AddGameComponents (IGameStateComponent[] components)

Fügt die angegebenen Spielkomponenten und deren über die Methode SubComponents() ermittelten Unterkomponenten des Spielkomponentenliste des XNA-Frameworks hinzu.

public void RemoveGameComponents (IGameStateComponents)

Entfernt die angegebenen Spiekomponenten und deren Unterkomponenten von der Spielkomponentenliste des XNA-Frameworks.

3.1.35 Klasse GameScreenComponent

Beschreibung:

Eine Spielkomponente, die in einem angegebenen Spielzustand verwendet wird und eine bestimmte Priorität hat.

Eigenschaften:

public DisplayLayer Index

Die Zeichen- und Eingabepriorität.

public GameScreen Screen

Der zugewiesene Spielzustand.

GameScreenComponent

- + Index : DisplayLayer
- + Screen : GameScreen
- + SubComponents (GameTime GameTime) : lEnumerable
- + GameStateComponent (GameScreen screen, DisplayLayer index) : void

Methoden:

public IEnumerable SubComponents (GameTime GameTime)

Gibt Spielkomponenten zurück, die in dieser Spielkomponente enthalten sind.

public void GameStateComponent (GameScreen, DisplayLayer index)

Erstellt eine neue Spielkomponente in dem angegebenen Spielzustand mit der angegebenen Priorität.

3.1.36 Klasse GraphicsSettingsScreen

Beschreibung:

Der Spielzustand, der die Grafik-Einstellungen darstellt.

Eigenschaften:

protected void settingsMenu

Das Menu, das die Einstellungen enthält.

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)

Fügt das Menu mit dem Einstellungen in die Spielkomponentenliste ein.

3.1.37 Klasse Input

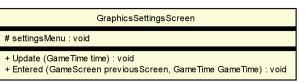
Beschreibung:

Eigenschaften:

public ClickState RightMouseButton

public ClickState LeftMouseButton

public MouseState CurrentMouseState



 ${\bf public} \ {\bf Keyboard State} \ {\bf Current Keyboard State}$

 ${\color{blue} \textbf{public MouseState PreviousMouseState}}$

 ${\color{blue} \textbf{public KeyboardState PreviousKeyboardState}}$

public Boolean GrabMouseMovement

Konstruktoren:

public Input (GameScreen screen)

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

3.1.38 Klasse InputItem

Beschreibung:

Eigenschaften:

public String InputText

3.1.39 Klasse KeyInputItem

Beschreibung:

Eigenschaften:

private OptionInfo option

Input + RightMouseButton: ClickState + LeftMouseButton: ClickState + CurrentMouseState: MouseState + CurrentKeyboardState: KeyboardState + PreviousMouseState: MouseState + PreviousKeyboardState: KeyboardState + GrabMouseMovement: Boolean + Input (GameScreen screen): void + Update (GameTime time): void

InputItem
+ InputText : String

```
Methoden:
public void OnKeyEvent ()
         Klasse Knot
3.1.40
Beschreibung:
{\bf Eigenschaften:}
public String Name
private Circle edges
public KnotMetaData MetaData
private IKnotIO format
public Action EdgesChanged
{\bf public\ IEnumerable}; {\bf Edge};\ {\bf SelectedEdges}
Konstruktoren:
public Knot ()
public Knot (KnotMetaData info)
```

public Knot (IKnotIO format)

KeyInputItem - option : OptionInfo + OnKeyEvent (): void

```
Methoden:
                                                                                      Knot
public void Save ()
                                                                   + Name : String
                                                                   - edges : Circle
                                                                   + MetaData : KnotMetaData
                                                                   - file : IKnotIO
public void ClearSelection ()
                                                                   + Edges Changed : Action
                                                                   + SelectedEdges : IEnumerable<Edge>
                                                                   + Knot (): void
                                                                   + Save () : void
public Boolean IsValidMove (Direction dir, Integer
                                                                 disclean Sete tion (): void
                                                                   + Knot (IKnotIO file) : void
                                                                   + Knot (KnotMetaData info) : void
                                                                   + IsValidMove (Direction dir, Integer distance): Boolean
                                                                   + Move (Direction dir, Integer distance) : Boolean
                                                                   + (Knot knotA, Knot knotB): Boolean
public Boolean Move (Direction dir, Integer distance) (Knot knot A, Knot knot B): Boolean
                                                                   + GetEnumerator (): IEnumerator<Edge>
                                                                   + Save (IKnotIO file): void
                                                                   + Clone () : Object
                                                                   + AddToSelection (Edge edge) : void
                                                                   + RemoveFromSelection (Edge edge) : void
public Boolean (Knot knotA, Knot knotB)
                                                                   + AddRangeToSelection (Edge edge): void
                                                                   + Is Selected (Edge edge) : Boolean
public Boolean (Knot knotA, Knot knotB)
public IEnumerator; Edge; GetEnumerator ()
public void Save (IKnotIO file)
public Object Clone ()
public void AddToSelection (Edge edge)
public void RemoveFromSelection (Edge edge)
public void AddRangeToSelection (Edge edge)
public Boolean IsSelected (Edge edge)
public IEnumerator GetEnumerator ()
```

3.1.41 Klasse Knot3Game

Beschreibung:

Eigenschaften: public Boolean IsFullScreen public Stack; GameScreen; Screens public Boolean VSync public GraphicsDeviceManager Graphics Methoden: public void Game () public void Initialize () public void LoadContent () public void UnloadContent () public void Draw (GameTime) public void Update (GameTime)

3.1.42 Klasse KnotFileIO

 ${\bf Beschreibung:}$

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

private KnotStringIO parser

Konstruktoren:

public KnotFileIO ()

Methoden:

public void Save (Knot knot)

public Knot Load (String filename)

public KnotMetaData LoadMetaData (String filename)

3.1.43 Klasse KnotInputHandler

Beschreibung:

Methoden:

public void Update (GameTime time)Wird für jeden Frame aufgerufen.

3.1.44 Klasse KnotMetaData

Beschreibung:

Eigenschaften:

public String Name

public IKnotIO Format

public Integer CountEdges

public String Filename

KnotInputHandler

KnotFile IO

+ Edges : IEnumerable<Edge>

+ KnotFileIO (String path) : void + Save (Knot knot) : void

+ KnotFileIO (Knot knot): void

+ Name : String - parser : KnotStringlO + CountEdges : Integer

+ Update (GameTime time) : void

KnotMeta Data

- + Name : String
- + File : IKnotlO
- + CountEdges : Integer
- $+ \ Knot Meta Data \ (String \ name, \ Integer \ count Edges, \ IKnot IO \ file) : Knot Meta Data$
- + KnotMetaData (IKnotIO file) : KnotMetaData

Konstruktoren:

public KnotMetaData (String name, Integer countEdges, IKnotIO file)

public KnotMetaData (IKnotIO file)

3.1.45 Klasse KnotRenderer

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

KnotRenderer

- + Info : GameObjectInfo
- + World : World
- arrow s : List<Arrow Model>
- nodes : List<NodeModel>
- pipes : List<PipeModel>
- + Knot : Knot
- pipeFactory : ModelFactory
- nodeFactory : ModelFactory
- arrow Factory : ModelFactory
- + Center (): Vector3
- + Intersects (Ray Ray) : GameObjectDistance
- + OnEdgesChanged (): void
- $+ \ Model Renderer \ (Game Screen \ screen, \ Game Object Info \ info) \ : \ void$
- + Update (GameTime GameTime) : void
- + Draw (GameTime GameTime) : void
- + GetEnumerator () : IEnumerator

```
Methoden:
public Vector3 Center ()
public GameObjectDistance Intersects (Ray Ray)
public void OnEdgesChanged ()
public void ModelRenderer (GameScreen screen, GameObjectInfo info)
public void Update (GameTime GameTime)
Wird für jeden Frame aufgerufen.
public void Draw (GameTime GameTime)
public IEnumerator GetEnumerator ()
         Klasse KnotStringIO
3.1.46
Beschreibung:
Konstruktoren:
public KnotStringIO ()
                                                                     KnotStringIO
                                                             + Name : String
                                                             + Edges : IErnumerable < Edge >
                                                             + Content : String
                                                             + CountEdges : Integer
Methoden:
                                                             + Save (Knot knot) : void
                                                              + KnotStringlO (String content) : void
public void Save (Knot knot)
public Knot Load (String filename)
public KnotMetaData LoadMetaData (String filename)
```

3.1.47 Klasse Localizer

Beschreibung:

Methoden:

public String Localize (String text)

Localizer + Localize (String text) : String

3.1.48 Klasse Menu

Beschreibung:

Eigenschaften:

public String Name

public Func;int, Vector2; RelativeItemSize

public Func;int, Vector2; RelativeItemPosition

Menu + Name: String + RelativeltemSize: Func<int, Vector2> + RelativeltemPosition: Func<int, Vector2> + ItemForgroundColor: Func<ItemState, Vector2> + ItemBackgroundColor: Func<ItemState, Vector2> + ItemAlignX: HorizontalAlignment + ItemAlignY: VerticalAlignment + Add (Menultem item): void + Delete (Menultem item): void + GetItem (Integer i): Menultem + Size (): Integer + GetEnumerator (): IEnumerator

 ${\bf public\ Func; Item State,\ Vector 2;\ Item Foreground Color}$

public Func; ItemState, Vector2; ItemBackgroundColor

public Horizontal Alignment Item Align X

public VerticalAlignment ItemAlignY

Methoden: public void Add (MenuItem item) public void Delete (MenuItem item) public MenuItem GetItem (Integer i) public Integer Size () public IEnumerator GetEnumerator () Klasse MenuButton 3.1.49 Beschreibung: Eigenschaften: public String Name Konstruktoren: public MenuButton (String name) 3.1.50 Klasse MenuItem Beschreibung: Eigenschaften: public ItemState ItemState public Integer ItemOrder

public String Text

MenuButton
+ Name : String
+ MenuButton (String name) : void

Methoden: public void OnKeyEvent () public void OnLeftClick (Vector2 position, ClickState state, CameTime time) public void OnRightClick (Vector2 position, ClickState state, GameTime time)

public void Officignt Click (vector 2 position, ClickState state, Gamerinie time)

public Rectangle Bounds ()

3.1.51 Klasse MenuScreen

Beschreibung:

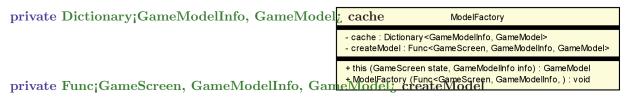
Methoden:



3.1.52 Klasse ModelFactory

Beschreibung:

Eigenschaften:



Konstruktoren:

public ModelFactory (Func;GameScreen, GameModelInfo,)

Methoden:

public GameModel this (GameScreen state, GameModelInfo info)

3.1.53 Klasse ModelMouseHandler

Beschreibung:

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

ModelMouseHandler + Update (GameTime time) : void

3.1.54 Klasse MousePointer

Beschreibung:

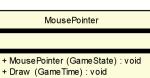
Konstruktoren:

public MousePointer (GameState)

Methoden:

public void Draw (GameTime)

3.1.55 Klasse NodeMap



Methoden: public Node From (Edge edge) + From (Edge edge): Node + To (Edge edge): Node + OnEdgesChanged (): void 3.1.56 Klasse NodeModel Beschreibung:

Konstruktoren:

Eigenschaften:

public NodeModelInfo Info

public NodeModel (GameScreen screen, NodeModelInfo info)

Methoden:

public void Draw (GameTime GameTime)

public void Update (GameTime GameTime)

Wird für jeden Frame aufgerufen.

3.1.57 Klasse NodeModelInfo

Beschreibung:

NodeModel

+ NodeModel (GameScreen screen, NodeModelInfo info) : void

+ Info : NodeModelInfo

+ Draw (GameTime GameTime) : void

+ Update (GameTime GameTime) : void

Eigenschaften: NodeModelInfo public Edge EdgeFrom + EdgeFrom : Edge + EdgeTo : Edge + Knot : Knot + Position : Vector3 public Edge EdgeTo + NodeModelInfo (Knot knot, Edge from, Edge to) : void public Knot Knot public Vector3 Position Konstruktoren: public NodeModelInfo (Knot knot, Edge from, Edge to) 3.1.58Klasse OptionInfo Beschreibung: Eigenschaften: private ConfigFile configFile OptionInfo - configFile : ConfigFile + Section : String + Name : String public String Section + DefaultValue : String + Value : String + OptionInfo (String section, String name, String defaultValue, ConfigFile configFile): void public String Name public String DefaultValue public String Value

public OptionInfo (String section, String name, String defaultValue, ConfigFile configFile)

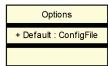
Konstruktoren:

3.1.59 Klasse Options

Beschreibung:

Eigenschaften:

public ConfigFile Default



3.1.60 Klasse PauseDialog

Beschreibung:

Eigenschaften:

private VerticalMenu pauseMenu

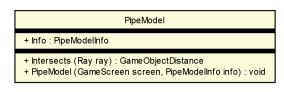


3.1.61 Klasse PipeModel

Beschreibung:

Eigenschaften:

public PipeModelInfo Info



Konstruktoren:

public PipeModel (GameScreen screen, PipeModelInfo info)

Methoden:

public GameObjectDistance Intersects (Ray ray)

3.1.62 Klasse PipeModelInfo

Eigenschaften:

public Edge Edge

public Knot Knot

public Vector3 PositionFrom

public Vector3 PositionTo

Konstruktoren:

public PipeModelInfo (Knot knot, Edge edge)

Klasse PipeMovement 3.1.63

Beschreibung:

Eigenschaften:

 ${\bf public~GameObjectInfo~Info}$

public Knot Knot

public World World

PipeMovement

PipeModelInfo

+ PipeModelInfo (Knot knot, Edge edge) : void

+ Edge : Edge + Knot : Knot

+ PositionFrom: Vector3 + PositionTo : Vector3

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

Konstruktoren:

public PipeMovement (GameState, World, GameObjectInfo)

```
Methoden:
public Vector3 Center ()
public GameObjectDistance Intersects (Ray Ray)
public void Update (GameTime GameTime)
Wird für jeden Frame aufgerufen.
public IEnumerator GetEnumerator ()
public void Draw (GameTime GameTime)
         Klasse PrinterIO
3.1.64
Beschreibung:
Konstruktoren:
public PrinterIO ()
                                                                      PrinterIO
                                                                + Edges : IEnumerable<Edge>
                                                                + Name : String
                                                                + CountEdges : Integer
Methoden:
                                                                + Save (Knot knot) : void
                                                                + PrinterIO (string path): void
public void Save (Knot knot)
public Knot Load (String filename)
public KnotMetaData LoadMetaData (String filename)
```

3.1.65 Klasse ProfileSettingsScreen

Eigenschaften:

protected void settingsMenu

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

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)

3.1.66 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 (GameTime, GameModel GameModel) : void
- + RemapModel (GameModel GameModel) : void # Draw RenderTarget (GameTime) : void

Methoden:

public void Begin (GameTime)

public void End (GameTime)

public void DrawModel (GameTime, GameModel GameModel)

public void RemapModel (GameModel GameModel)

protected void DrawRenderTarget (GameTime)

3.1.67 Klasse RenderEffectStack

Beschreibung:

Eigenschaften:

public IRenderEffect CurrentEffect

private IRenderEffect DefaultEffect

RenderEffectStack + CurrentEffect : IRenderEffect - DefaultEffect : IRenderEffect + Pop () : IRenderEffect + Push (IRenderEffect effect) : void + RenderEffectStack (IRenderEffect defaultEffect) : void

Konstruktoren:

public RenderEffectStack (IRenderEffect defaultEffect)

Methoden:

public IRenderEffect Pop ()

public void Push (IRenderEffect effect)

3.1.68 Klasse SettingsScreen

Beschreibung:

Eigenschaften:

protected void navigation

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

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime time)

3.1.69 Klasse ShadowGameModel

Beschreibung:

Eigenschaften:

public Color ShadowColor

public float ShadowAlpha

Shadow GameModel

- + Shadow Color : Color
- + Shadow Alpha: float
- + Shadow GameModel (GameScreen sreen, GameModel decoratedModel) : void
- + Draw (GameTime GameTime): void

Konstruktoren:

public ShadowGameModel (GameScreen sreen, GameModel decoratedModel)

Methoden:

public void Draw (GameTime GameTime)

3.1.70 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
- original collient vector
- + Center (): Vector3
- + Update (GameTime GameTime) : void
- + Draw (GameTime GameTime) : void
- + Intersects (Ray Ray) : GameObjectDistance
- + Shadow GameObject (GameScreen screen, IGameObject decoratedObj) : void

Konstruktoren:

public ShadowGameObject (GameScreen screen, IGameObject decoratedObj)

Methoden:

public Vector3 Center ()

public void Update (GameTime GameTime)

Wird für jeden Frame aufgerufen.

public void Draw (GameTime GameTime)

public GameObjectDistance Intersects (Ray Ray)

3.1.71 Klasse SliderItem

Beschreibung:

Eigenschaften:

public Integer Value

public Integer MinValue

public Integer MaxValue

SliderItem + Value : Integer + MinValue : Integer + MaxValue : Integer

3.1.72 Klasse StandardEffect

Beschreibung:

Konstruktoren:

public StandardEffect (GameScreen screen)

StandardEffect # Draw RenderTarget (GameTime) : void + StandardEffect (GameScreen screen) : void

Methoden:

protected void DrawRenderTarget (GameTime)

3.1.73 Klasse StartScreen

Beschreibung:

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen

StartScreen	
Update (GameTime time) : void	
Entered (Gaine Spreen Preyique Forman) Game Time Game Time) : void	d

3.1.74 Klasse TextInputDialog

Beschreibung:

Eigenschaften:

public String InputText

TextInputDialog
+ InputText : String

${\bf 3.1.75 \quad Klasse \quad Tutorial Challenge Mode}$ de

Beschreibung:

Methoden:

public void Entered (GameScreen previousScreen, GameTime Gternet) Diengelode

+ Entered (GameScreen previousScreen, GameTime GameTime) : void

3.1.76 Klasse VerticalMenu

Methoden: VerticalMenu public void AlignItems () + AlignItems (): void Klasse Widget 3.1.77Beschreibung: Eigenschaften: public Vector2 RelativeSize Widget + RelativeSize : Vector2 + RelativePosition : Vector2 + lsVisible : bool public Vector2 RelativePosition + BackgroundColor : Func<Color> + ForegroundColor : Func<Color> + A lign X: Horizontal A lignment+ AlignY: VerticalAlignment public bool IsVisible + BoundingBox () : Rectangle + Widget (GameScreen screen, DisplayLayer draw Order) : void public Func; Color; Background Color public Func; Color; Foreground Color public Horizontal Alignment AlignX public VerticalAlignment AlignY Konstruktoren: public Widget (GameScreen screen, DisplayLayer drawOrder) Methoden:

${\bf 3.1.78}\quad {\bf Klasse~WidgetKeyHandler}$

public Rectangle BoundingBox ()

Methoden: public void Update () 3.1.79Klasse WidgetMouseHandler Beschreibung: Methoden: public void Update () 3.1.80Klasse World Beschreibung: Eigenschaften: public void Camera public List; IGameObject; Objects public IGameObject SelectedObject public IRenderEffect CurrentEffect Konstruktoren: public World (GameScreen screen) Methoden: public void Update (GameTime GameTime)

Wird für jeden Frame aufgerufen.

public void Draw (GameTime GameTime)

public IEnumerator GetEnumerator ()



3.1.81 Klasse XNA.DrawableGameComponent

Beschreibung:

Methoden:

public void Draw (GameTime time)

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

3.1.82 Klasse XNA.Game

Beschreibung:

Methoden:

public void Game ()

public void Initialize ()

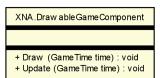
public void LoadContent ()

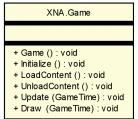
public void UnloadContent ()

public void Update (GameTime)

public void Draw (GameTime)

3.1.83 Klasse XNA.GameComponent

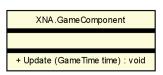




Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.



3.2 Schnittstellen

3.2.1 Schnittstelle IChallengeIO

Beschreibung:

Diese Schnittstelle enthält Methoden, die von Speicherformaten für Challenges implementiert werden müssen.

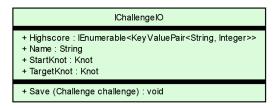
Methoden:

public void Save (Challenge challenge)

Speichert eine Challenge.

public Challenge Load (String filename)

Lädt eine Challenge.



public ChallengeMetaData LoadMetaData (String filename)

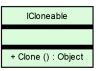
Lädt die Metadaten einer Challenge.

3.2.2 Schnittstelle ICloneable

Beschreibung:

Methoden:

public Object Clone ()



3.2.3 Schnittstelle IEnumerable

Methoden: |Enumerable<MenuItem> public IEnumerator GetEnumerator () Schnittstelle IEnumerable1 3.2.4Beschreibung: Methoden: public IEnumerator GetEnumerator () IEnumerable1 + GetEnumerator (): IEnumerator Schnittstelle IEquatable 3.2.5Beschreibung: Methoden: public Boolean Equals (T other) |Equatable<GameObjectInfo> 3.2.6Schnittstelle IEquatable1 Beschreibung: Methoden: public Boolean Equals (T other) IEquatable1

3.2.7 Schnittstelle IGameObject

Beschreibung:

Diese Schnittstelle repräsentiert ein Spielobjekt und enthält eine Referenz auf die Spielwelt, in der sich das Spielobjekt befindet, sowie Informationen zu dem Spielobjekt.

+ Equals (Tother) : Boolean

Eigenschaften:

public GameObjectInfo Info

Informationen über das Spielobjekt wie z.B. die Position.

public World World

Eine Referenz auf die Spielwelt, in der sich das Spielobjekt befindet.

Methoden:

public Vector3 Center ()

Die Mitte des Spielobjektes im 3D-Raum.

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Draw (GameTime time)

Zeichnet das Spielobjekt.

public GameObjectDistance Intersects (Ray ray)

Überprüft, ob der Mausstrahl das Spielobjekt schneidet.

3.2.8 Schnittstelle IGameScreenComponent

Beschreibung:

Eine Schnittstelle für eine Spielkomponente, die in einem angegebenen Spielzustand verwendet wird und eine bestimmte Priorität hat.

Eigenschaften:

public DisplayLayer Index

Die Zeichen- und Eingabepriorität.

public GameScreen Screen

Der zugewiesene Spielzustand.

Methoden:

public IEnumerable SubComponents (GameTime time)

Gibt Spielkomponenten zurück, die in dieser Spielkomponente enthalten sind.

IGameObject

- + Info : GameObjectInfo + World : World
- + vvoria : vvoria
- + Center (): Vector3
- + Update (GameTime time) : void
- + Draw (GameTime time) : void
- + Intersects (Ray ray) : GameObjectDistance



3.2.9 Schnittstelle IJunction

Beschreibung:

Repräsentiert einen Übergang zwischen zwei Kanten.

Eigenschaften:

public Edge EdgeFrom

Die Kante vor dem Übergang.

public Edge EdgeTo

Die Kante nach dem Übergang.

3.2.10 Schnittstelle IKeyEventListe-

Beschreibung:

ner

Eine Schnittstelle, die von Klassen inmplementiert wird, die auf Tastatureingaben reagieren.

Eigenschaften:

public DisplayLayer Index

Die Eingabepriorität.

public Boolean IsKeyEventEnabled

Ob die Klasse zur Zeit auf Tastatureingaben reagiert.

public List; Keys; ValidKeys

Die Tasten, auf die die Klasse reagiert.

Methoden:

public void OnKeyEvent ()

Die Reaktion auf eine Tasteneingabe.

3.2.11 Schnittstelle IKnotIO

${\bf Beschreibung:}$

Diese Schnittstelle enthält Methoden, die von Speicherformaten für Knoten implementiert werden müssen.





Methoden:

public void Save (Knot knot)

Speichert einen Knoten.

public Knot Load (String filename)

Lädt einen Knoten.

public KnotMetaData LoadMetaData (String filename)

Lädt die Metadaten eines Knotens.

3.2.12 Schnittstelle IMouseEventListener

Beschreibung:

Eine Schnittstelle, die von Klassen inmplementiert wird, die auf Maus-Klicks reagieren.

Eigenschaften:

public DisplayLayer Index

Die Eingabepriorität.

public Boolean IsMouseEventEnabled

Ob die Klasse zur Zeit auf Mausklicks reagiert.

Methoden:

public Rectangle Bounds ()

Die Ausmaße des von der Klasse repräsentierten Objektes.

public void OnLeftClick (Vector2 position, ClickState state, GameTime time)

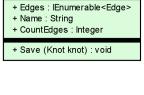
Die Reaktion auf einen Linksklick.

public void OnRightClick (Vector2 position, ClickState state, GameTime time)

Die Reaktion auf einen Rechtsklick.

3.2.13 Schnittstelle IRenderEffect

Beschreibung:



IKnotIO

- + OnLeftClick () : void
- + OnRightClick (): void
- + Bounds () : Rectangle

Eigenschaften: public RenderTarget2D RenderTarget + RenderTarget: RenderTarget2D + Begin (GameTime): void + End (GameTime): void + Draw Model (GameModel model): void + RemapModel (GameModel model): void public void Begin (GameTime) public void End (GameTime) public void DrawModel (GameTime, GameModel model) public void RemapModel (GameModel model)

3.3 Enumerationen

Abläufe

4.1 Sequenzdiagramme

Klassenindex

Anmerkungen

Glossar