ENTWURFSDOKUMENT

(V. 1.0)

KNOT^3

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

16. Dezember 2013

Inhaltsverzeichnis

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 RenderEffect und die von ihr abgeleiteten Klassen gekapselt. Ein RenderEffect enthält ein Rendertarget vom Typ RenderTarget2D als Attribut und im-

plementiert 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

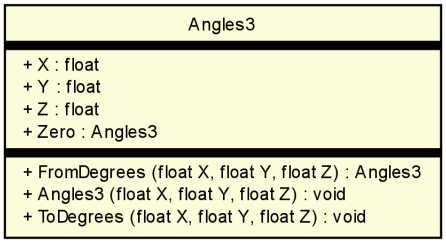
3.1 Klassen

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)

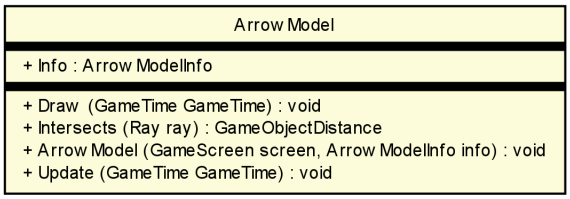
Konvertiert Bogenmaß in Grad.

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.

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)

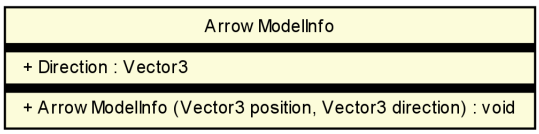
Wird für jeden Frame aufgerufen.

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 die der Pfeil zeigen soll.

Konstruktoren:

public ArrowModelInfo (Vector3 position, Vector3 direction)

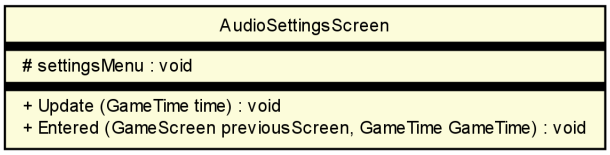
Erstellt ein neues ArrowModelInfo-Objekt an einer bestimmten Position im 3D-Raum, das in eine bestimmte Richtung zeigt.

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)

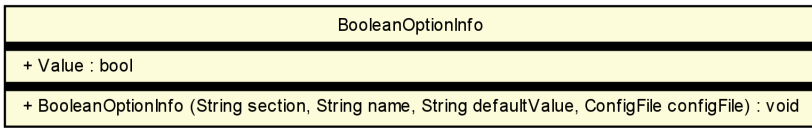
Fügt das Menu mit dem Einstellungen in die Spielkomponentenliste ein.

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.

Konstruktoren:

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

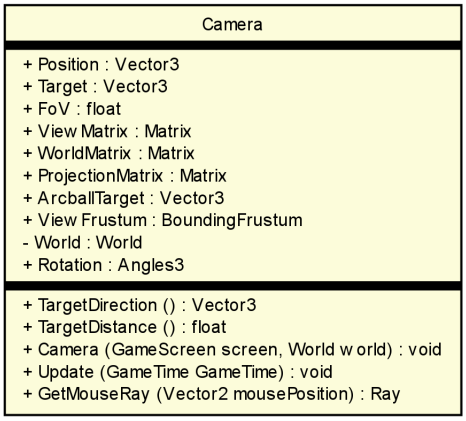
Erstellt eine neue Option, die die Werte „Wahr “oder „Falsch “annehmen kann, mit dem angegebenen Namen in dem angegebenen Abschnitt der angegebenen Einstellungsdatei.

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.

Eigenschaften:



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.

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

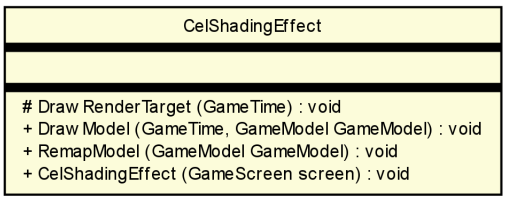
Beschreibung:

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)

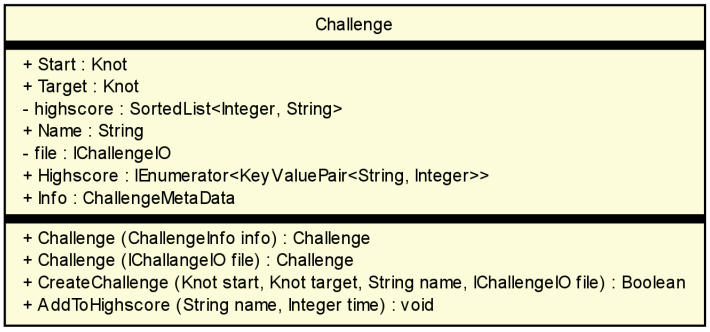
Weist dem 3D-Modell den Cel-Shader zu.

3.1.8 Klasse Challenge

Beschreibung:

Ein Objekt dieser Klasse repräsentiert eine Challenge.

Eigenschaften:



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

Eine sortierte Bestenliste.

private IChallengeIO format

Das Speicherformat der Challenge.

public IEnumerator¡KeyValuePair¡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)

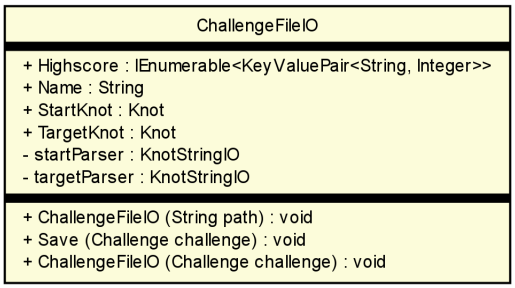
Fügt eine neue Bestzeit eines bestimmten Spielers in die Bestenliste ein.

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.

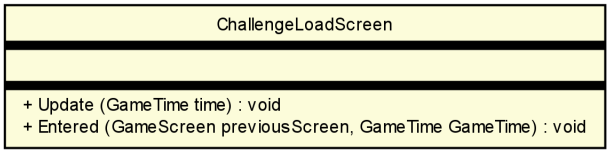
public void Entered (GameScreen previousScreen, GameTime GameTime)

Fügt das Menu mit dem Spielständen in die Spielkomponentenliste ein.

3.1.11 Klasse ChallengeMetaData

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.

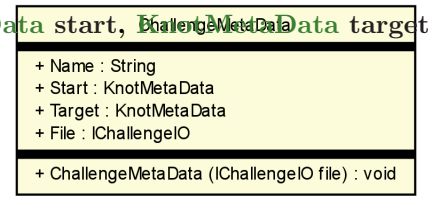
public IEnumerator¡KeyValuePair¡String, Integer¿¿ Highscore

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.



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.

Eigenschaften:

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.

private KnotRenderer ChallengeKnotRenderer

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)

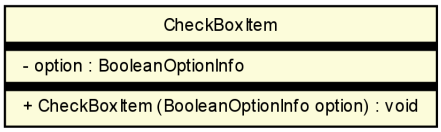
Fügt die 3D-Welten und den Inputhandler in die Spielkomponentenliste ein.

3.1.13 Klasse CheckBoxItem

Beschreibung:

Ein Menueintrag, der einen Auswahlkasten darstellt.

Eigenschaften:



private BooleanOptionInfo option

Die Option, die mit dem Auswahlkasten verknüft ist.

Konstruktoren:

public CheckBoxItem (BooleanOptionInfo option)

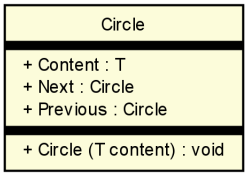
Erstellt einen Menueintrag für die angegebenen Option.

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

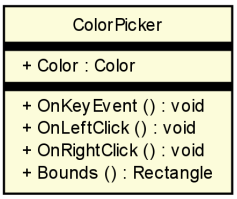
Beschreibung:

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)

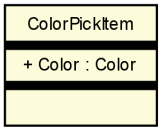
Bei einem Rechtsklick geschieht nichts.

3.1.16 Klasse ColorPickItem

Beschreibung:

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

Eigenschaften:



public Color Color

Die aktuelle Farbe.

private ColorPicker picker

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

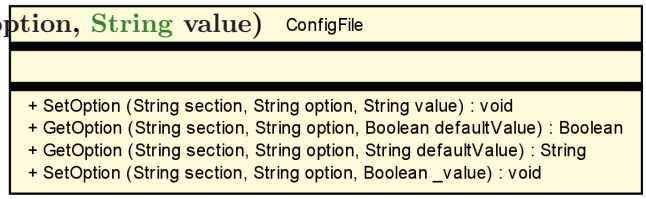
3.1.17 Klasse ConfigFile

Beschreibung:

Repräsentiert eine Einstellungsdatei.

Methoden:

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



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

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)

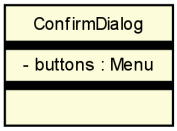
Setzt den Wert der Option mit dem angegebenen Namen in den angegebenen Abschnitt auf den angegebenen Wert.

3.1.18 Klasse ConfirmDialog

Beschreibung:

Ein Dialog, der Schaltflächen zum Bestätigen einer Aktion anzeigt.

Eigenschaften:



private Menu buttons

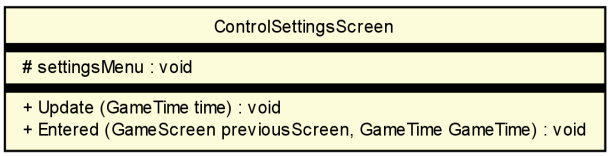
Das Menü, das Schaltflächen enthält.

3.1.19 Klasse ControlSettingsScreen

Beschreibung:

Der Spielzustand, der die Steuerungs-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.

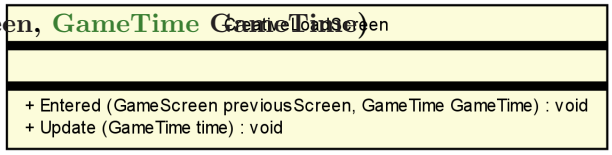
3.1.20 Klasse CreativeLoadScreen

Beschreibung:

Der Spielzustand, der den Ladebildschirm für Knoten darstellt.

Methoden:

public void Entered (GameScreen previousScreen, GameTime GameTime)



Fügt das Menu mit dem Spielständen in die Spielkomponentenliste ein.

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.

Eigenschaften:

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:

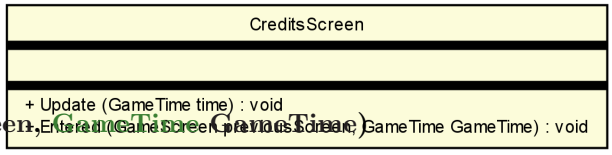
Der Spielzustand, der die Auflistung der Mitwirkenden darstellt.

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)



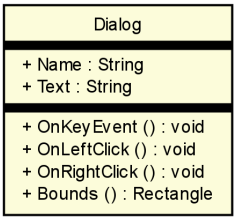
Fügt das Menu mit dem Mitwirkenden in die Spielkomponentenliste ein.

3.1.23 Klasse Dialog

Beschreibung:

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

Eigenschaften:



public String Title

Der Fenstertitel.

public String Text

Der angezeigte Text.

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)

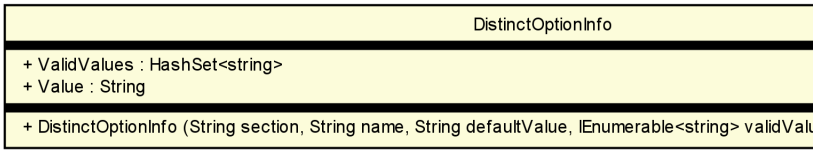
Bei einem Rechtsklick geschieht nichts.

3.1.24 Klasse DistinctOptionInfo

Beschreibung:

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

Eigenschaften:



public HashSet¡string¿ ValidValues

public String Value

Ein Property, das den aktuell abgespeicherten Wert zurück gibt.

Konstruktoren:

public DistinctOptionInfo (String section, String name, String defaultValue, IEnumerable¡string¿ validValu

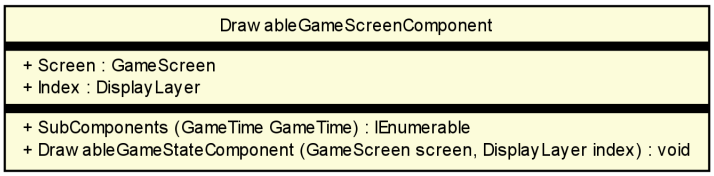
Erstellt eine neue Option, die einen der angegebenen gültigen Werte annehmen kann, mit dem angegebenen Namen in dem angegebenen Abschnitt der angegebenen Einstellungsdatei.

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.

Methoden:

public IEnumerable SubComponents (GameTime GameTime)

Gibt Spielkomponenten zurück, die in dieser Spielkomponente enthalten sind.

public void DrawableGameStateComponent (GameScreen screen, DisplayLayer index)

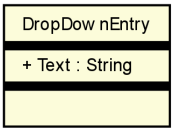
Erstellt eine neue zeichenbare Spielkomponente in dem angegebenen Spielzustand mit der angegebenen Priorität.

3.1.26 Klasse DropDownEntry

Beschreibung:

Repräsentiert einen Eintrag in einem Dropdown-Menü.

Eigenschaften:



public String Text

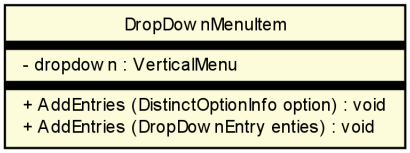
Der Text des Eintrags.

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.

Eigenschaften:



private VerticalMenu dropdown

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

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)

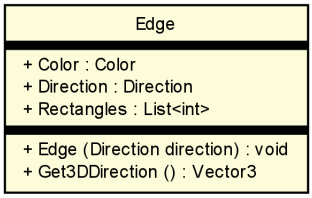
Fügt Einträge in das Dropdown-Menü ein, die auf Einstellungsoptionen basieren. Fügt Einträge in das Dropdown-Menü ein, die nicht auf Einstellungsoptionen basieren.

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 ()

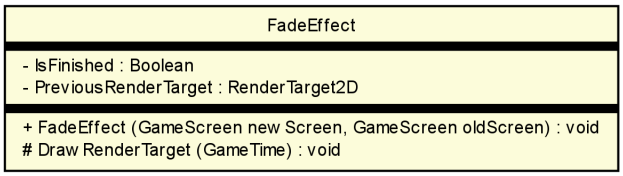
Gibt die Richtung als normalisierten Vektor3 zurück.

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.

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)

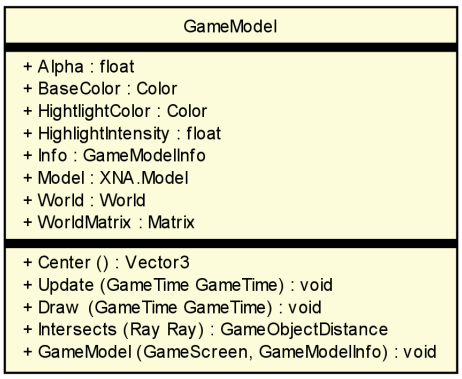
Konvertiert einen Namen eines Knotens oder einer Challenge in einen gültigen Dateinamen durch Weglassen ungüliger Zeichen.

3.1.31 Klasse GameModel

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.

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)

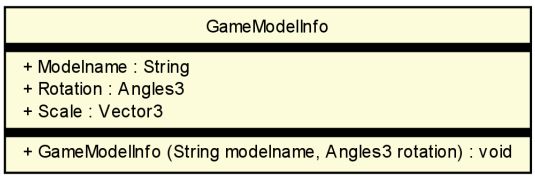
Überprüft, ob der Mausstrahl das 3D-Modell schneidet.

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)

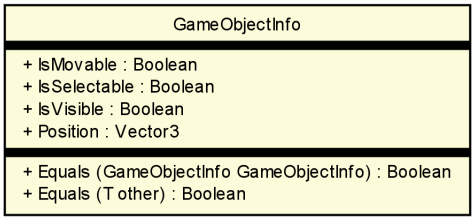
Erstellt ein neues Informations-Objekt eines 3D-Modells mit den angebenenen Informationen zu Dateiname, Rotation und Skalierung.

3.1.33 Klasse GameObjectInfo

Beschreibung:

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

Eigenschaften:



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)

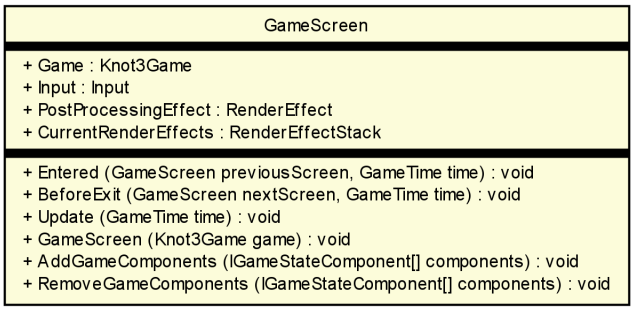
Vergleicht zwei Informationsobjekte für Spielobjekte.

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

public RenderEffectStack CurrentRenderEffects

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

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 Tastatur- als 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 (IGameStateComponent[] components)

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.

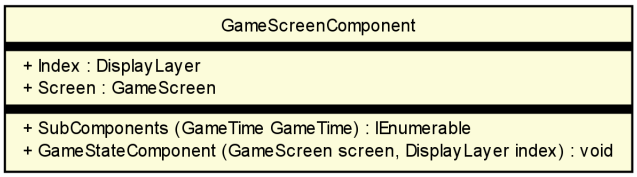
Eigenschaften:

public DisplayLayer Index

Die Zeichen- und Eingabepriorität.

public GameScreen Screen

Der zugewiesene Spielzustand.



Methoden:

public IEnumerable SubComponents (GameTime GameTime)

Gibt Spielkomponenten zurück, die in dieser Spielkomponente enthalten sind.

public void GameStateComponent (GameScreen screen, DisplayLayer index)

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

3.1.36 Klasse GraphicsSettingsScreen

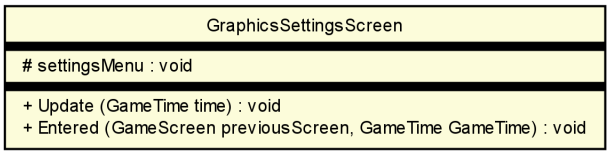
Beschreibung:

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

public KeyboardState CurrentKeyboardState

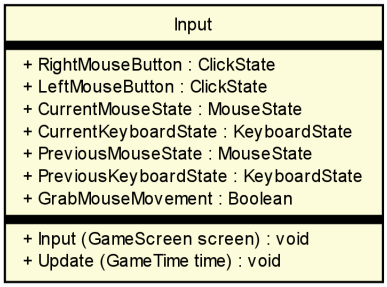
public MouseState PreviousMouseState

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

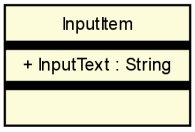
Beschreibung:

Eigenschaften:

public String InputText

3.1.39 Klasse KeyInputItem

Beschreibung:



Eigenschaften:

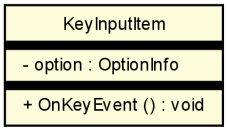
private OptionInfo option

Methoden:

public void OnKeyEvent ()

3.1.40 Klasse Knot

Beschreibung:



Eigenschaften:

public String Name

private Circle edges

public KnotMetaData MetaData

private IKnotIO format

public Action EdgesChanged

public IEnumerable¡Edge¿ SelectedEdges

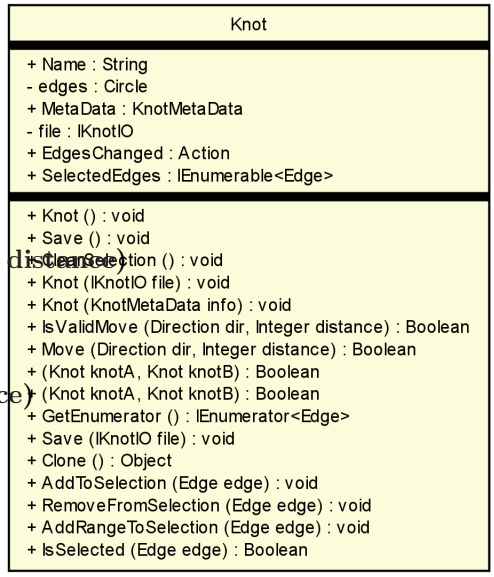
Konstruktoren:

public Knot ()

public Knot (KnotMetaData info)

public Knot (IKnotIO format)

Methoden:



public void Save ()

public void ClearSelection ()

public Boolean IsValidMove (Direction dir, Integer distance)

public Boolean Move (Direction dir, Integer distance)

public Boolean (Knot knotA, Knot knotB)

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)

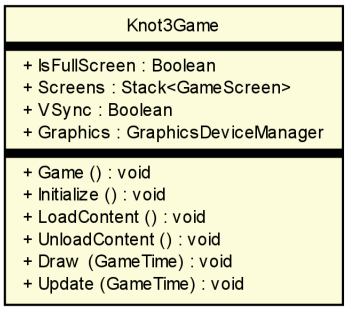
public Boolean IsSelected (Edge edge)

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)

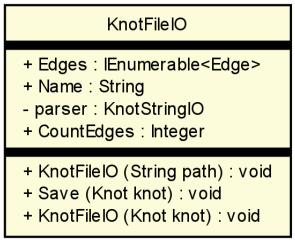
public void Update (GameTime)

3.1.42 Klasse KnotFileIO

Beschreibung:

Eigenschaften:

private KnotStringIO parser



Konstruktoren:

public KnotFileIO ()

Methoden:

public void Save (Knot knot)

public Knot Load (String filename)

public KnotMetaData LoadMetaData (String filename)

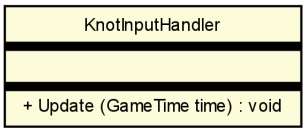
3.1.43 Klasse KnotInputHandler

Beschreibung:

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.



3.1.44 Klasse KnotMetaData

Beschreibung:

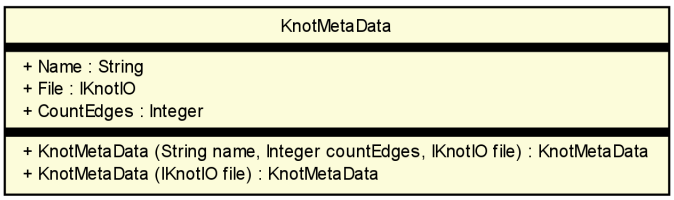
Eigenschaften:

public String Name

public IKnotIO Format

public Integer CountEdges

public String Filename



Konstruktoren:

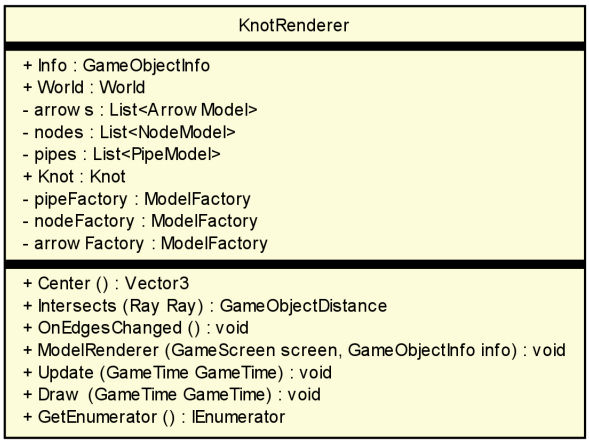
public KnotMetaData (String name, Integer countEdges, IKnotIO file)

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

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)

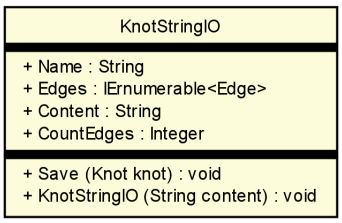
public IEnumerator GetEnumerator ()

3.1.46 Klasse KnotStringIO

Beschreibung:

Konstruktoren:

public KnotStringIO ()



Methoden:

public void Save (Knot knot)

public Knot Load (String filename)

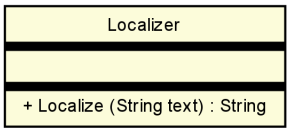
public KnotMetaData LoadMetaData (String filename)

3.1.47 Klasse Localizer

Beschreibung:

Methoden:

public String Localize (String text)



3.1.48 Klasse Menu

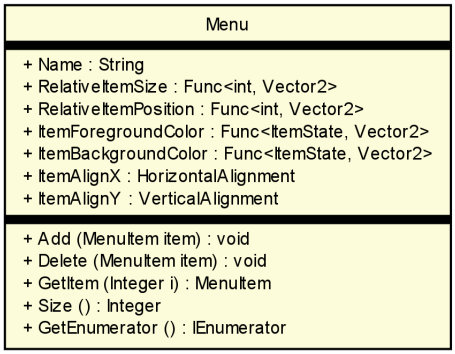
Beschreibung:

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 item)

public void Delete (MenuItem item)

public MenuItem GetItem (Integer i)

public Integer Size ()

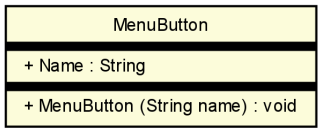
public IEnumerator GetEnumerator ()

3.1.49 Klasse MenuButton

Beschreibung:

Eigenschaften:

public String Name



Konstruktoren:

public MenuButton (String name)

3.1.50 Klasse MenuItem

Beschreibung:

Eigenschaften:

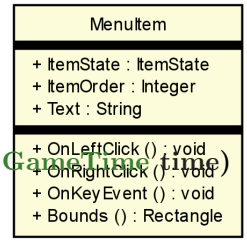
public ItemState ItemState

public Integer ItemOrder

public String Text

Methoden:

public void OnKeyEvent ()



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

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

public Rectangle Bounds ()

3.1.51 Klasse MenuScreen

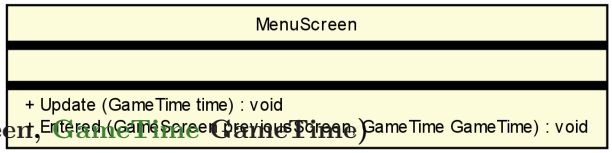
Beschreibung:

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)

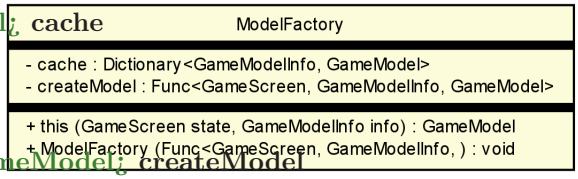


3.1.52 Klasse ModelFactory

Beschreibung:

Eigenschaften:

private Dictionary¡GameModelInfo, GameModel¿ cache



private Func¡GameScreen, GameModelInfo, GameModel¿ createModel

Konstruktoren:

public ModelFactory (Func¡GameScreen, GameModelInfo,)

Methoden:

public GameModel this (GameScreen state, GameModelInfo info)

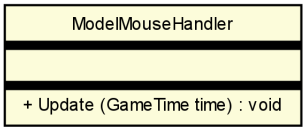
3.1.53 Klasse ModelMouseHandler

Beschreibung:

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

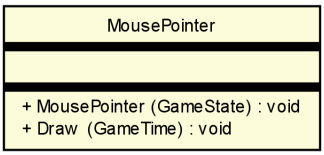


3.1.54 Klasse MousePointer

Beschreibung:

Konstruktoren:

public MousePointer (GameState)



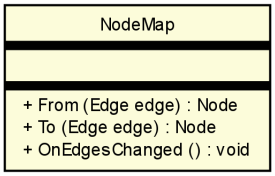
Methoden:

public void Draw (GameTime)

3.1.55 Klasse NodeMap

Beschreibung:

Methoden:



public Node From (Edge edge)

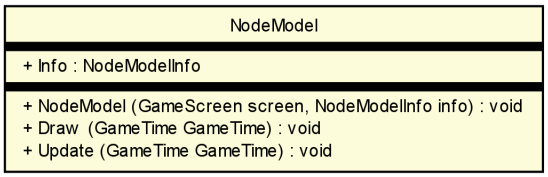
public Node To (Edge edge)

public void OnEdgesChanged ()

3.1.56 Klasse NodeModel

Beschreibung:

Eigenschaften:



public NodeModelInfo Info

Konstruktoren:

public NodeModel (GameScreen screen, NodeModelInfo info)

Methoden:

public void Draw (GameTime GameTime)

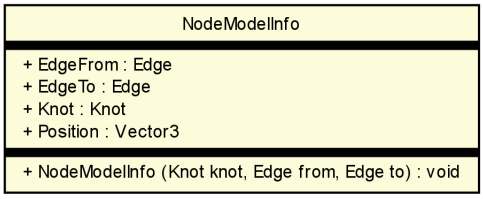
public void Update (GameTime GameTime)

Wird für jeden Frame aufgerufen.

3.1.57 Klasse NodeModelInfo

Beschreibung:

Eigenschaften:



public Edge EdgeFrom

public Edge EdgeTo

public Knot Knot

public Vector3 Position

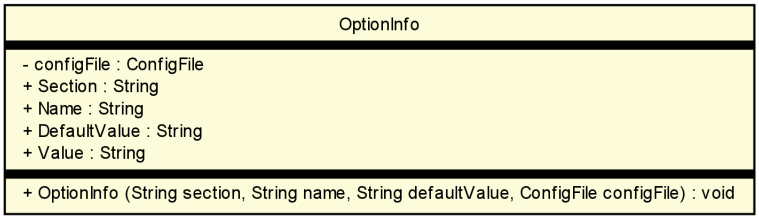
Konstruktoren:

public NodeModelInfo (Knot knot, Edge from, Edge to)

3.1.58 Klasse OptionInfo

Beschreibung:

Eigenschaften:



private ConfigFile configFile

public String Section

public String Name

public String DefaultValue

public String Value

Konstruktoren:

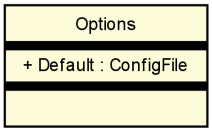
public OptionInfo (String section, String name, String defaultValue, ConfigFile configFile)

3.1.59 Klasse Options

Beschreibung:

Eigenschaften:

public ConfigFile Default

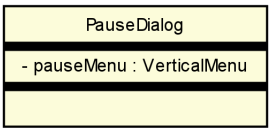


3.1.60 Klasse PauseDialog

Beschreibung:

Eigenschaften:

private VerticalMenu pauseMenu

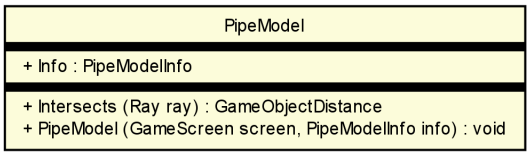


3.1.61 Klasse PipeModel

Beschreibung:

Eigenschaften:

public PipeModelInfo Info



Konstruktoren:

public PipeModel (GameScreen screen, PipeModelInfo info)

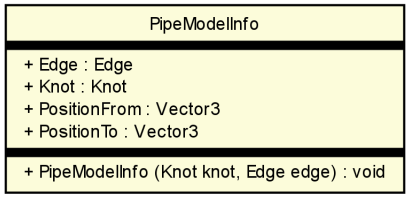
Methoden:

public GameObjectDistance Intersects (Ray ray)

3.1.62 Klasse PipeModelInfo

Beschreibung:

Eigenschaften:



public Edge Edge

public Knot Knot

public Vector3 PositionFrom

public Vector3 PositionTo

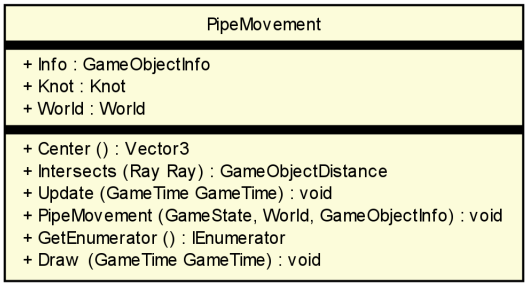
Konstruktoren:

public PipeModelInfo (Knot knot, Edge edge)

3.1.63 Klasse PipeMovement

Beschreibung:

Eigenschaften:



public GameObjectInfo Info

public Knot Knot

public World World

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 ()

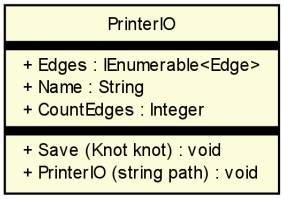
public void Draw (GameTime GameTime)

3.1.64 Klasse PrinterIO

Beschreibung:

Konstruktoren:

public PrinterIO ()



Methoden:

public void Save (Knot knot)

public Knot Load (String filename)

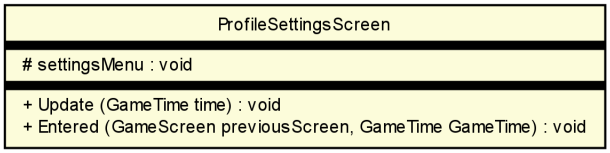
public KnotMetaData LoadMetaData (String filename)

3.1.65 Klasse ProfileSettingsScreen

Beschreibung:

Eigenschaften:

protected void settingsMenu



Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)

3.1.66 Klasse RenderEffect

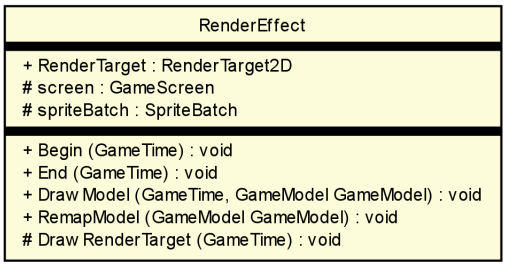
Beschreibung:

Eigenschaften:

public RenderTarget2D RenderTarget

protected GameScreen screen

protected SpriteBatch spriteBatch



Methoden:

public void Begin (GameTime)

public void End (GameTime)

public void DrawModel (GameTime, GameModel GameModel)

public void RemapModel (GameModel GameModel)

protected void DrawRenderTarget (GameTime)

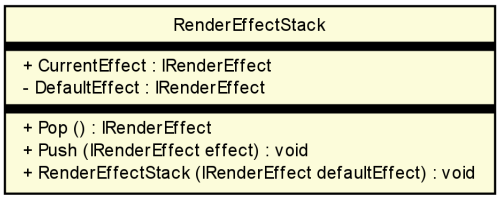
3.1.67 Klasse RenderEffectStack

Beschreibung:

Eigenschaften:

public IRenderEffect CurrentEffect

private IRenderEffect DefaultEffect



Konstruktoren:

public RenderEffectStack (IRenderEffect defaultEffect)

Methoden:

public IRenderEffect Pop ()

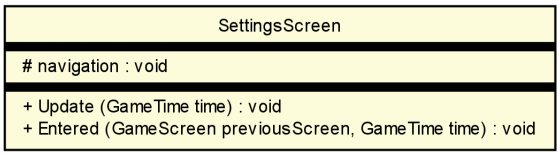
public void Push (IRenderEffect effect)

3.1.68 Klasse SettingsScreen

Beschreibung:

Eigenschaften:

protected void navigation



Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime time)

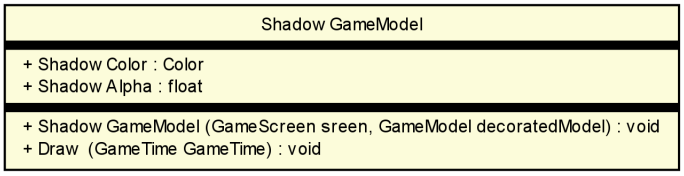
3.1.69 Klasse ShadowGameModel

Beschreibung:

Eigenschaften:

public Color ShadowColor

public float ShadowAlpha



Konstruktoren:

public ShadowGameModel (GameScreen sreen, GameModel decoratedModel)

Methoden:

public void Draw (GameTime GameTime)

3.1.70 Klasse ShadowGameObject

Beschreibung:

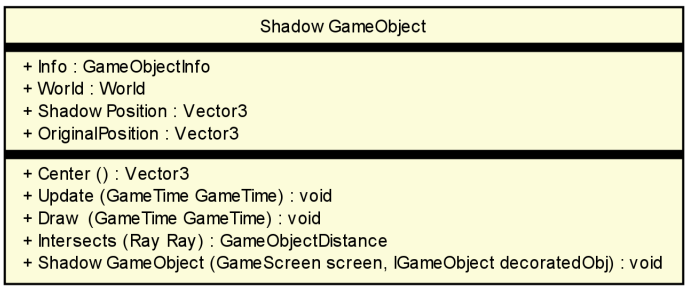
Eigenschaften:

public GameObjectInfo Info

public World World

public Vector3 ShadowPosition

public Vector3 OriginalPosition



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)

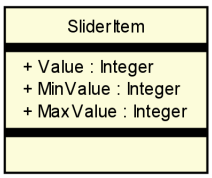
public GameObjectDistance Intersects (Ray Ray)

3.1.71 Klasse SliderItem

Beschreibung:

Eigenschaften:

public Integer Value



public Integer MinValue

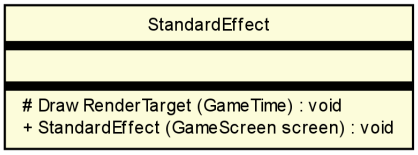
public Integer MaxValue

3.1.72 Klasse StandardEffect

Beschreibung:

Konstruktoren:

public StandardEffect (GameScreen screen)



Methoden:

protected void DrawRenderTarget (GameTime)

3.1.73 Klasse StartScreen

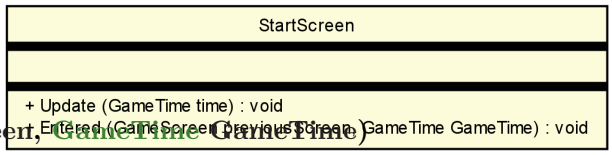
Beschreibung:

Methoden:

public void Update (GameTime time)

Wird für jeden Frame aufgerufen.

public void Entered (GameScreen previousScreen, GameTime GameTime)

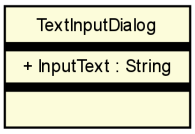


3.1.74 Klasse TextInputDialog

Beschreibung:

Eigenschaften:

public String InputText

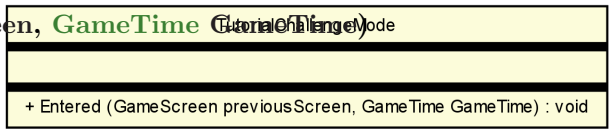


3.1.75 Klasse TutorialChallengeMode

Beschreibung:

Methoden:

public void Entered (GameScreen previousScreen, GameTime GameTime)

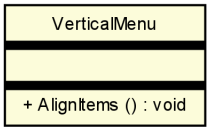


3.1.76 Klasse VerticalMenu

Beschreibung:

Methoden:

public void AlignItems ()



3.1.77 Klasse Widget

Beschreibung:

Eigenschaften:

public Vector2 RelativeSize

public Vector2 RelativePosition

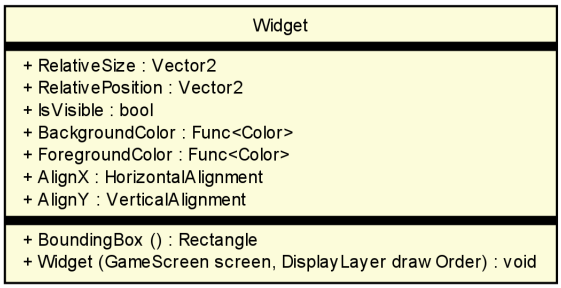
public bool IsVisible

public Func¡Color¿ BackgroundColor

public Func¡Color¿ ForegroundColor

public HorizontalAlignment AlignX

public VerticalAlignment AlignY



Konstruktoren:

public Widget (GameScreen screen, DisplayLayer drawOrder)

Methoden:

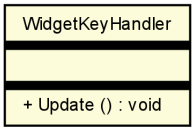
public Rectangle BoundingBox ()

3.1.78 Klasse WidgetKeyHandler

Beschreibung:

Methoden:

public void Update ()

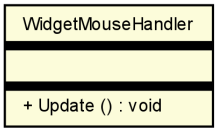


3.1.79 Klasse WidgetMouseHandler

Beschreibung:

Methoden:

public void Update ()



3.1.80 Klasse World

Beschreibung:

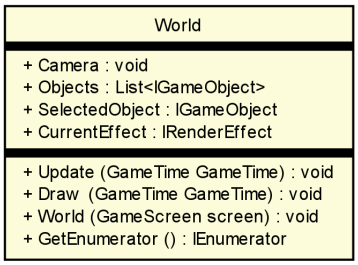
Eigenschaften:

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)

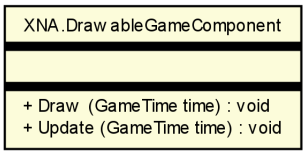
public IEnumerator GetEnumerator ()

3.1.81 Klasse XNA.DrawableGameComponent

Beschreibung:

Methoden:

public void Draw (GameTime time)



public void Update (GameTime time)

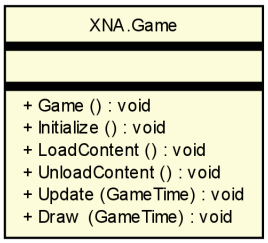
Wird für jeden Frame aufgerufen.

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)

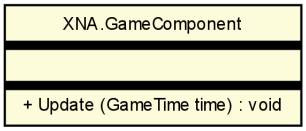
3.1.83 Klasse XNA.GameComponent

Beschreibung:

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.

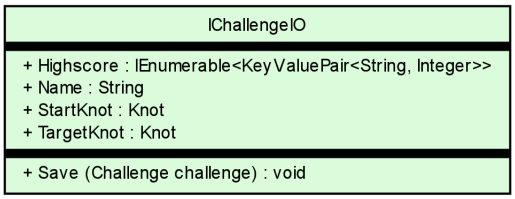
Methoden:

public void Save (Challenge challenge)

Speichert eine Challenge.

public Challenge Load (String filename)

Lädt eine Challenge.



public ChallengeMetaData LoadMetaData (String filename)

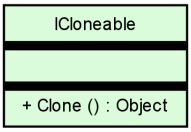
Lädt die Metadaten einer Challenge.

3.2.2 Schnittstelle ICloneable

Beschreibung:

Methoden:

public Object Clone ()

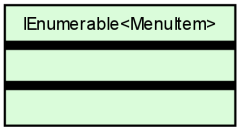


3.2.3 Schnittstelle IEnumerable

Beschreibung:

Methoden:

public IEnumerator GetEnumerator ()

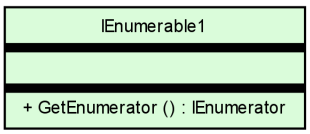


3.2.4 Schnittstelle IEnumerable1

Beschreibung:

Methoden:

public IEnumerator GetEnumerator ()

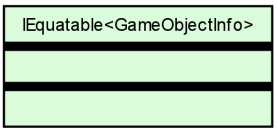


3.2.5 Schnittstelle IEquatable

Beschreibung:

Methoden:

public Boolean Equals (T other)

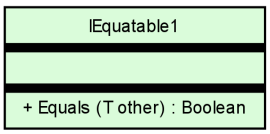


3.2.6 Schnittstelle IEquatable1

Beschreibung:

Methoden:

public Boolean Equals (T other)

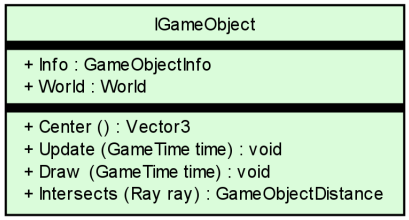


3.2.7 Schnittstelle IGameObject

Beschreibung:

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

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)

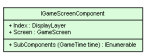
Überprüft, ob der Mausstrahl das Spielobjekt schneidet.

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)

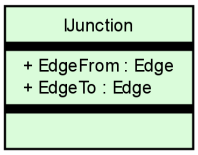
Gibt Spielkomponenten zurück, die in dieser Spielkomponente enthalten sind.

3.2.9 Schnittstelle IJunction

Beschreibung:

Repräsentiert einen Übergang zwischen zwei Kanten.

Eigenschaften:



public Edge EdgeFrom

Die Kante vor dem Übergang.

public Edge EdgeTo

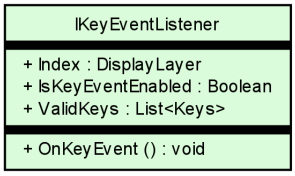
Die Kante nach dem Übergang.

3.2.10 Schnittstelle IKeyEventListener

Beschreibung:

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 ()

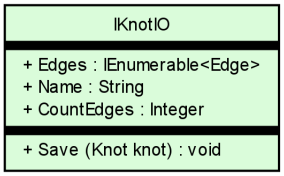
Die Reaktion auf eine Tasteneingabe.

3.2.11 Schnittstelle IKnotIO

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)

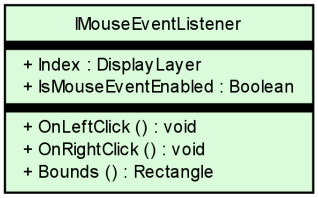
Lädt die Metadaten eines Knotens.

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.

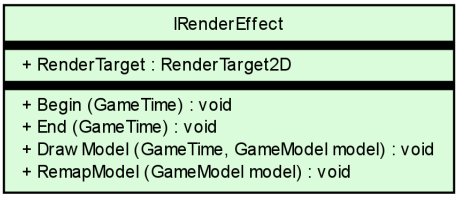
public void OnRightClick (Vector2 position, ClickState state, GameTime time)

Die Reaktion auf einen Rechtsklick.

3.2.13 Schnittstelle IRenderEffect

Beschreibung:

Eigenschaften:



public RenderTarget2D RenderTarget

Methoden:

public void Begin (GameTime)

public void End (GameTime)

public void DrawModel (GameTime, GameModel model)

public void RemapModel (GameModel model)

3.3 Enumerationen

Kapitel 4

Abläufe

4.1 Sequenzdiagramme

Kapitel 5

Klassenindex

Kapitel 6

Anmerkungen

Kapitel 7

Glossar