

Spacetraveler

Milestone1.0

Erzeugt von Doxygen 1.8.11

Inhaltsverzeichnis

1	Ausstehende Aufgaben	2
2	Verzeichnis der Namensbereiche	2
2.1	Pakete	2
3	Hierarchie-Verzeichnis	2
3.1	Klassenhierarchie	2
4	Klassen-Verzeichnis	2
4.1	Auflistung der Klassen	2
5	Datei-Verzeichnis	3
5.1	Auflistung der Dateien	3
6	Dokumentation der Namensbereiche	3
6.1	Paket spacetraveler	3
7	Klassen-Dokumentation	4
7.1	spacetraveler.BlackHole Klassenreferenz	4
7.1.1	Ausführliche Beschreibung	5
7.1.2	Beschreibung der Konstruktoren und Destruktoren	5
7.1.3	Dokumentation der Datenelemente	6
7.2	spacetraveler.Game Klassenreferenz	6
7.2.1	Ausführliche Beschreibung	7
7.2.2	Dokumentation der Elementfunktionen	7
7.2.3	Dokumentation der Datenelemente	9
7.3	spacetraveler.Gravity Klassenreferenz	9
7.3.1	Ausführliche Beschreibung	10
7.3.2	Beschreibung der Konstruktoren und Destruktoren	10
7.3.3	Dokumentation der Elementfunktionen	11
7.3.4	Dokumentation der Datenelemente	11
7.4	spacetraveler.GravityModel Klassenreferenz	11

7.4.1	Ausführliche Beschreibung	12
7.4.2	Beschreibung der Konstruktoren und Destruktoren	12
7.4.3	Dokumentation der Elementfunktionen	12
7.4.4	Dokumentation der Datenelemente	12
7.5	spacetraveler.Level Klassenreferenz	13
7.5.1	Ausführliche Beschreibung	14
7.5.2	Beschreibung der Konstruktoren und Destruktoren	14
7.5.3	Dokumentation der Elementfunktionen	15
7.5.4	Dokumentation der Datenelemente	15
7.6	spacetraveler.SpaceObject Klassenreferenz	17
7.6.1	Ausführliche Beschreibung	18
7.6.2	Beschreibung der Konstruktoren und Destruktoren	18
7.6.3	Dokumentation der Elementfunktionen	18
7.6.4	Dokumentation der Datenelemente	19
7.7	spacetraveler.SpaceObjectModel Klassenreferenz	20
7.7.1	Ausführliche Beschreibung	21
7.7.2	Beschreibung der Konstruktoren und Destruktoren	21
7.7.3	Dokumentation der Elementfunktionen	22
7.7.4	Dokumentation der Datenelemente	23
7.8	spacetraveler.Tile Klassenreferenz	24
7.8.1	Ausführliche Beschreibung	24
7.8.2	Beschreibung der Konstruktoren und Destruktoren	24
7.8.3	Dokumentation der Datenelemente	25
8	Datei-Dokumentation	25
8.1	src/spacetraveler/BlackHole.java-Dateireferenz	25
8.2	src/spacetraveler/Game.java-Dateireferenz	26
8.3	src/spacetraveler/Gravity.java-Dateireferenz	26
8.4	src/spacetraveler/GravityModel.java-Dateireferenz	26
8.5	src/spacetraveler/Level.java-Dateireferenz	26
8.6	src/spacetraveler/SpaceObject.java-Dateireferenz	27
8.7	src/spacetraveler/SpaceObjectModel.java-Dateireferenz	27
8.8	src/spacetraveler/Tile.java-Dateireferenz	27

Index	29
-----------------------	----

1 Ausstehende Aufgaben

Element [spacetraveler.Game.I](#)

Die aktuelle Instanz der Levelklasse

2 Verzeichnis der Namensbereiche

2.1 Pakete

Hier folgen die Pakete mit einer Kurzbeschreibung (wenn verfügbar):

spacetraveler	3
-------------------------------	---

3 Hierarchie-Verzeichnis

3.1 Klassenhierarchie

Die Liste der Ableitungen ist -mit Einschränkungen- alphabetisch sortiert:

spacetraveler.Game	6
spacetraveler.Gravity	9
spacetraveler.BlackHole	4
spacetraveler.GravityModel	11
spacetraveler.Level	13
spacetraveler.SpaceObject	17
spacetraveler.SpaceObjectModel	20
spacetraveler.Tile	24

4 Klassen-Verzeichnis

4.1 Auflistung der Klassen

Hier folgt die Aufzählung aller Klassen, Strukturen, Varianten und Schnittstellen mit einer Kurzbeschreibung:

spacetraveler.BlackHole Klasse fuer Schwarze Loecher Klasse Gravity vererbt	4
--	---

spacetraveler.Game	
Gameklasse. Enthaeft die main() -Methode	6
spacetraveler.Gravity	
Gravitationsklasse	9
spacetraveler.GravityModel	
Klasse fuer rechnerische Eigenschaften der Gravitationspunkte	11
spacetraveler.Level	
Enthaeft alle fuer ein Level notwendige Objekte (Gravitationspunkte, etc...)	13
spacetraveler.SpaceObject	
Superklasse fuer alle Objekte, die sich auf dem Bildschirm bewegen koennen	17
spacetraveler.SpaceObjectModel	
Rechnerisches Modell fuer SpaceObjects	20
spacetraveler.Tile	
Darstellung der Tiles	24

5 Datei-Verzeichnis

5.1 Auflistung der Dateien

Hier folgt die Aufzählung aller Dateien mit einer Kurzbeschreibung:

src/spacetraveler/BlackHole.java	25
src/spacetraveler/Game.java	26
src/spacetraveler/Gravity.java	26
src/spacetraveler/GravityModel.java	26
src/spacetraveler/Level.java	26
src/spacetraveler/SpaceObject.java	27
src/spacetraveler/SpaceObjectModel.java	27
src/spacetraveler/Tile.java	27

6 Dokumentation der Namensbereiche

6.1 Paket spacetraveler

Klassen

- class [BlackHole](#)
Klasse fuer Schwarze Loecher Klasse [Gravity](#) vererbt.
- class [Game](#)
Gameklasse. Enthaeft die [main\(\)](#)-Methode.

- class [Gravity](#)

Gravitationsklasse.

- class [GravityModel](#)

Klasse fuer rechnerische Eigenschaften der Gravitationspunkte.

- class [Level](#)

Enthaelt alle fuer ein [Level](#) notwendige Objekte (Gravitationspunkte, etc...)

- class [SpaceObject](#)

Superklasse fuer alle Objekte, die sich auf dem Bildschirm bewegen koennen.

- class [SpaceObjectModel](#)

Rechnerisches Modell fuer SpaceObjects.

- class [Tile](#)

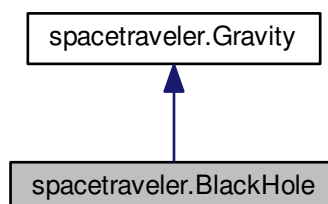
Darstellung der Tiles.

7 Klassen-Dokumentation

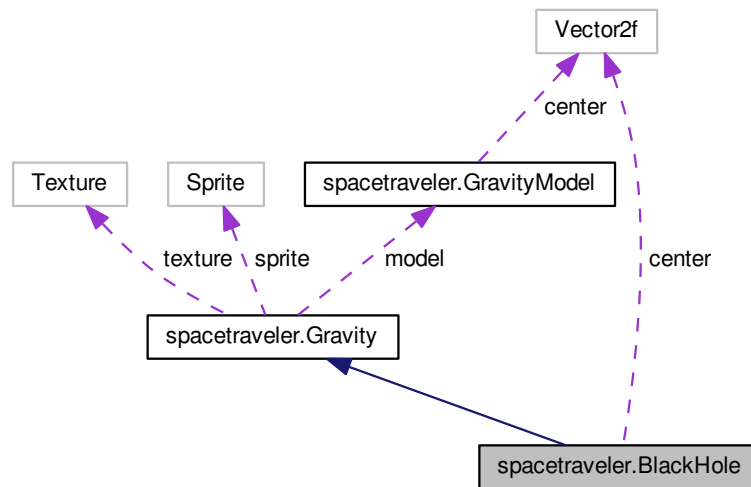
7.1 spacetraveler.BlackHole Klassenreferenz

Klasse fuer Schwarze Loecher Klasse [Gravity](#) vererbt.

Klassendiagramm für spacetraveler.BlackHole:



Zusammengehörigkeiten von spacetraveler.BlackHole:



Öffentliche Methoden

- **BlackHole** (Vector2f **center**, double m) throws IOException
Konstruktor fuer Schwarze Loecher.

Öffentliche Attribute

- Vector2f **center** = this.model.center

7.1.1 Ausführliche Beschreibung

Klasse fuer Schwarze Loecher Klasse **Gravity** vererbt.

7.1.2 Beschreibung der Konstruktoren und Destruktoren

7.1.2.1 spacetraveler.BlackHole.BlackHole (Vector2f **center**, double *m*) throws IOException

Konstruktor fuer Schwarze Loecher.

Parameter

<i>center</i>	Zentrum des Schwarzen Loches
<i>m</i>	Masse des Schwarzen Loches

Ausnahmebehandlung

<code>IOException</code>	wenn die Textur nicht geladen werden kann.
--------------------------	--

7.1.3 Dokumentation der Datenelemente

7.1.3.1 `Vector2f spacetraveler.BlackHole.center = this.model.center`

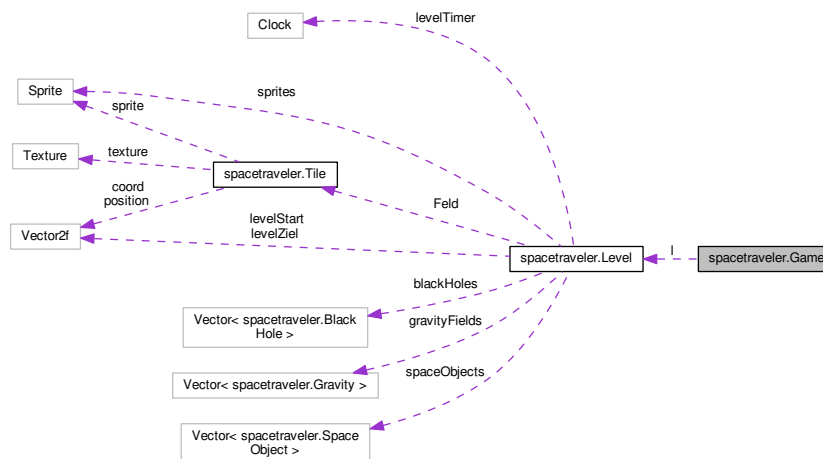
Die Dokumentation für diese Klasse wurde erzeugt aufgrund der Datei:

- `src/spacetraveler/BlackHole.java`

7.2 `spacetraveler.Game` Klassenreferenz

Gameklasse. Enthält die `main()`-Methode.

Zusammengehörigkeiten von `spacetraveler.Game`:



Öffentliche, statische Methoden

- static float `absVec` (Vector2f v)
Hilfsfunktion zum Berechnen der Laenge des Vectors.
- static float `skalar` (Vector2f a, Vector2f b)
Hilfsfunktion zum berechnen des Skalarproduktes.
- static boolean `contains` (FloatRect f, Vector2f P)
Funktion zum ueberpruefen ob ein Punkt in einem Floatrect ist.
- static boolean `intersection` (FloatRect a, FloatRect b)
Funktion die die ueberschneidung zweier Floatrects ueberpueft.
- static boolean `containsFloat` (FloatRect a, FloatRect b)
- static boolean `SpaceObjectsCollision` (SpaceObject A, SpaceObject B)
Ueberpruefen ob im naechsten Schritt zwei SpaceObjects kollidieren werden.
- static void `schneiden` (Vector< SpaceObject > spaceObjects)
Kollisionsueberpruefung und elastischer Stoss.
- static void `main` (String args[]) throws InterruptedException, IOException
Main-Methode des ganzen Spiels.

Statische, private Attribute

- static boolean `gravLeft` = false
Wird linke Maustaste gedrueckt?
- static boolean `gravRight` = false
Wird rechte Maustaste gedrueckt?
- static `Level` `l`

7.2.1 Ausführliche Beschreibung

Gameklasse. Enthaelte die `main()`-Methode.

7.2.2 Dokumentation der Elementfunktionen

7.2.2.1 static float spacetraveler.Game.absVec (Vector2f *v*) [static]

Hilfsfunktion zum Berechnen der Laenge des Vectors.

Parameter

<i>v</i>	ein Vektor
----------	------------

Rückgabe

Laenge des Vektors

7.2.2.2 static boolean spacetraveler.Game.contains (FloatRect *f*, Vector2f *P*) [static]

Funktion zum ueberpruefen ob ein Punkt in einem Floatrect ist.

Parameter

<i>f</i>	das Floatrect
<i>P</i>	der Punkt

Rückgabe

true wenn der Punkt enthalten ist, ansonsten false

7.2.2.3 static boolean spacetraveler.Game.containsFloat (FloatRect *a*, FloatRect *b*) [static]

7.2.2.4 static boolean spacetraveler.Game.intersection (FloatRect *a*, FloatRect *b*) [static]

Funktion die die ueberschneidung zweier Floatrects ueberpueft.

Parameter

<i>a</i>	FloatRect a muss erheblich kleiner als b sein
<i>b</i>	Das groessere Floatrect

Rückgabe

true wenn sie sich ueberschneiden, ansonnsten false

7.2.2.5 `static void spacetraveler.Game.main (String args[]) throws InterruptedException, IOException` [static]

Main-Methode des ganzen Spiels.

Parameter

<i>args</i>	Konsolenargumente, die dem Programm uebergeben werden. (Werden nicht ausgewertet)
-------------	---

<true, wenn der Spieler das Spiel verloren hat

< true, wenn der Spieler das Spiel gewonnen hat

<-1, wenn kein Gravitationszentrum gesetzt ist

Abschnitt zur Berechnung der Position der SpaceObjects im Feld Zuerst Berechnung der umliegenden Sprites und KOLLision Danach Ermittlung auf welchem [Tile](#) es momentan ist

ueberpruefen der Kollision unter den Objekten Bewegen aller Objekte

7.2.2.6 `static void spacetraveler.Game.schneiden (Vector< SpaceObject > spaceObjects)` [static]

Kollisionsueberpruefung und elastischer Stoss.

Parameter

<i>spaceObjects</i>	liste der Spaceobjects, um alle ueberpruefen zu kuennen
---------------------	---

7.2.2.7 `static float spacetraveler.Game.skalar (Vector2f a, Vector2f b)` [static]

Hilfsfunktion zum berechnen des Skalarproduktes.

Parameter

<i>a</i>	erster Vektor
<i>b</i>	zweiter Vektor

Rückgabe

Skalarprodukt der Beiden Vektoren

7.2.2.8 static boolean spacetraveler.Game.SpaceObjectsCollision (SpaceObject A, SpaceObject B) [static]

Ueberpruefen ob im naechsten Schritt zwei SpaceObjects kollidieren werden.

Parameter

A	erstes SpaceObject
B	zweites SpaceObject

Rückgabe

bei Kollision true, ansonsten false

7.2.3 Dokumentation der Datenelemente

7.2.3.1 boolean spacetraveler.Game.gravLeft = false [static],[private]

Wird linke Maustaste gedrueckt?

7.2.3.2 boolean spacetraveler.Game.gravRight = false [static],[private]

Wird rechte Maustaste gedrueckt?

7.2.3.3 Level spacetraveler.Game.l [static],[private]

Noch zu erledigen Die aktuelle Instanz der Levelklasse

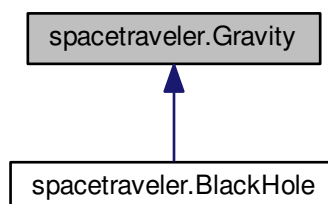
Die Dokumentation für diese Klasse wurde erzeugt aufgrund der Datei:

- src/spacetraveler/[Game.java](#)

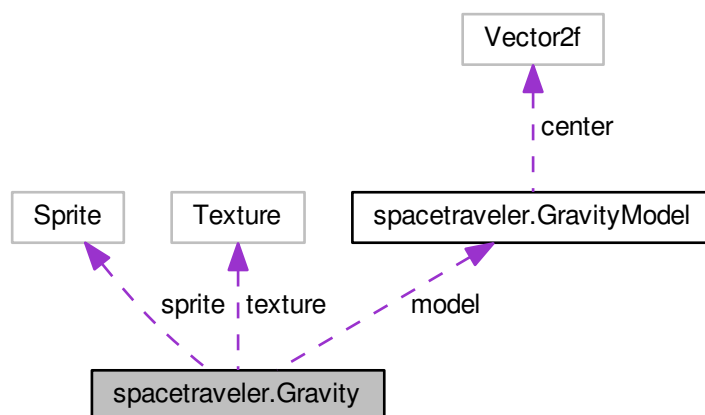
7.3 spacetraveler.Gravity Klassenreferenz

Gravitationsklasse.

Klassendiagramm für spacetraveler.Gravity:



Zusammengehörigkeiten von spacetraveler.Gravity:



Öffentliche Methoden

- [Gravity](#) (Vector2f center, double m) throws IOException
Konstruktor.
- Sprite [getSprite](#) ()
getter fuer Sprite

Öffentliche Attribute

- Texture [texture](#)
- Sprite [sprite](#)
- [GravityModel](#) [model](#)

7.3.1 Ausführliche Beschreibung

Gravitationsklasse.

Diese Klasse vereint unser Model ([GravityModel](#)) mit der SFML-Anzeige (Texture, Sprite, etc...)

7.3.2 Beschreibung der Konstruktoren und Destruktoren

7.3.2.1 spacetraveler.Gravity.Gravity (Vector2f center, double m) throws IOException

Konstruktor.

Parameter

<i>center</i>	Zentrum der Gravitation
<i>m</i>	Masse des Punktes (= Proportional zur Anziehungskraft)

Ausnahmebehandlung

<i>IOException</i>	Wenn Textur nicht geladen werden konnte
--------------------	---

7.3.3 Dokumentation der Elementfunktionen

7.3.3.1 Sprite spacetraveler.Gravity.getSprite ()

getter fuer Sprite

Rückgabe

Sprite der Klasse

7.3.4 Dokumentation der Datenelemente

7.3.4.1 GravityModel spacetraveler.Gravity.model

Rechnerisches Modell fuer die Gravitation

7.3.4.2 Sprite spacetraveler.Gravity.sprite

Sprite der Klasse

7.3.4.3 Texture spacetraveler.Gravity.texture

Textur der Klasse

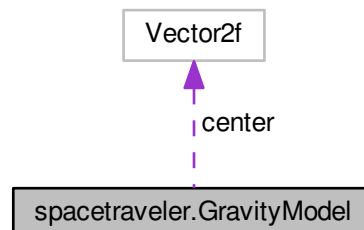
Die Dokumentation für diese Klasse wurde erzeugt aufgrund der Datei:

- src/spacetraveler/[Gravity.java](#)

7.4 spacetraveler.GravityModel Klassenreferenz

Klasse fuer rechnerische Eigenschaften der Gravitationspunkte.

Zusammengehörigkeiten von spacetraveler.GravityModel:



Öffentliche Methoden

- [GravityModel](#) (Vector2f [center](#), double [m](#))
Konstruktor.
- Vector2f [getEnergy](#) ([SpaceObject](#) [s](#))
Gibt die auf ein [SpaceObject](#) wirkende Energie aus Berechnet also die Auswirkungen der Gravitation auf das Objekt [s](#).

Öffentliche Attribute

- Vector2f [center](#)
- double [m](#)

7.4.1 Ausführliche Beschreibung

Klasse fuer rechnerische Eigenschaften der Gravitationspunkte.

7.4.2 Beschreibung der Konstruktoren und Destruktoren

7.4.2.1 spacetraveler.GravityModel.GravityModel (Vector2f [center](#), double [m](#))

Konstruktor.

Parameter

center	Zentrum der Gravitation
m	Masse des Gravitationspunktes (= Proportional zur Anziehungskraft)

7.4.3 Dokumentation der Elementfunktionen

7.4.3.1 Vector2f spacetraveler.GravityModel.getEnergy ([SpaceObject](#) [s](#))

Gibt die auf ein [SpaceObject](#) wirkende Energie aus Berechnet also die Auswirkungen der Gravitation auf das Objekt [s](#).

Parameter

s	SpaceObject fuer das die Energieauswirkungen berechnet werden sollen
-------------------	--

Rückgabe

Energievektor, der dem [SpaceObject](#) hinzugefuegt werden kann

7.4.4 Dokumentation der Datenelemente

7.4.4.1 Vector2f spacetraveler.GravityModel.center

Gravitationszentrum

7.4.4.2 double spacetraveler.GravityModel.m

Masse

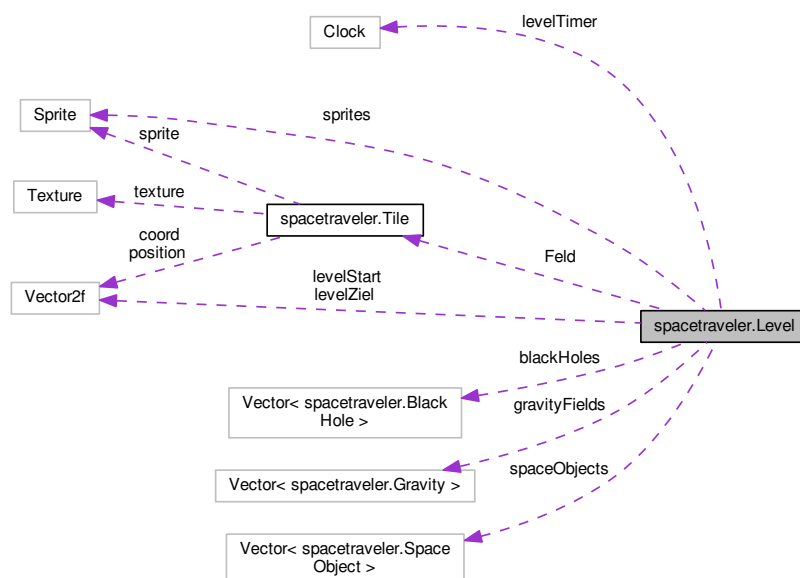
Die Dokumentation für diese Klasse wurde erzeugt aufgrund der Datei:

- src/spacetraveler/[GravityModel.java](#)

7.5 spacetraveler.Level Klassenreferenz

Enthaelt alle fuer ein [Level](#) notwendige Objekte (Gravitationspunkte, etc...)

Zusammengehörigkeiten von spacetraveler.Level:



Öffentliche Methoden

- [Level](#) (String levelId) throws IOException
Liest ein [Level](#) aus der Datei ein und erstellt die dazugehoerigen Objekte.
- void [loadTile](#) (Vector2f pos, int tileType) throws IOException
Laedt das Hintergrundbild eines Tiles und fuegt die Objekte des Tiles dem aktuellen [Level](#) hinzu.

Öffentliche Attribute

- `Tile[][] Feld`
Koordinatenfeld der indices.
- `Vector< SpaceObject > spaceObjects`
SpaceObjects im Level.
- `Vector< Gravity > gravityFields`
GravityFields im Level.
- `Vector< BlackHole > blackHoles`
BlackHoles im Level.
- `Sprite[] sprites`
zusätzliche Sprites fuer Start und Ziel im Level
- `Clock levelTimer`
Timer, der Zeit seit Beginn hochzaehlt.
- `float levelTimeAvailable`
Zeit, die fuer das Level zur Verfuegung steht.
- `Vector2f levelStart`
Level Startpunkt.
- `Vector2f levelZiel`
Level Zielpunkt.
- `int levelWidth`
Breite des Levels.
- `int levelHeight`
Hoehe des Levels.

7.5.1 Ausführliche Beschreibung

Enthaelt alle fuer ein `Level` notwendige Objekte (Gravitationspunkte, etc...)

Ausnahmebehandlung

<code>IOException</code>	Wenn Dateien nicht geladen werden koennen
--------------------------	---

7.5.2 Beschreibung der Konstruktoren und Destruktoren

7.5.2.1 `spacetraveler.Level.Level (String levelId) throws IOException`

Liest ein `Level` aus der Datei ein und erstellt die dazugehoerigen Objekte.

Parameter

<code>levelId</code>	Name des Levels (= Name der Datei in /spacetraveler/rsc/levels/)
----------------------	---

Ausnahmebehandlung

<code>IOException</code>	Wenn Leveldatei nicht geoeffnet werden kann
--------------------------	---

Struktur der Leveldatei:

Datentyp	Inhalt
int	levelTimeAvailable
floats	startX, startY
floats	zielX, zielY
ints	width, height
ints	id1,1 id2,1 id3,1 id4,1
ints	id2,1 id2,2 id3,2 id4,2
...	...

7.5.3 Dokumentation der Elementfunktionen

7.5.3.1 void spacetraveler.Level.loadTile (Vector2f pos, int tileType) throws IOException

Laedt das Hintergrundbild eines Tiles und fuegt die Objekte des Tiles dem aktuellen [Level](#) hinzu.

Parameter

<i>pos</i>	Position des Tiles (in absoluten Bildschirmkoordinaten)
<i>tileType</i>	Gibt den Typ/die Art des Tiles an

Ausnahmebehandlung

<i>IOException</i>	Dateizugriffsfehler
--------------------	---------------------

Einlesen der TileType-Datei Erstellen der Objekte und deren Speicherung in den Vektoren

Struktur einer Tile-Datei (Ohne Leerzeilen):

Datentyp	Inhalt
int	hintergrundId
int	anzahlSpaceObjects
string	texturePfad
float	m
floats	EX, EY
floats	posX, posY
boolean	gravityOn
int	anzahlGravityFields
floats	posX, posX
float	m
int	anzahlBlackHoles
floats	posX, posY
float	m

7.5.4 Dokumentation der Datenelemente

7.5.4.1 `Vector<BlackHole> spacetraveler.Level.blackHoles`

BlackHoles im [Level](#).

7.5.4.2 `Tile [][] spacetraveler.Level.Feld`

Koordinatenfeld der indices.

7.5.4.3 `Vector<Gravity> spacetraveler.Level.gravityFields`

GravityFields im [Level](#).

7.5.4.4 `int spacetraveler.Level.levelHeight`

Hoehe des Levels.

7.5.4.5 `Vector2f spacetraveler.Level.levelStart`

[Level](#) Startpunkt.

7.5.4.6 `float spacetraveler.Level.levelTimeAvailable`

Zeit, die fuer das [Level](#) zur Verfuegung steht.

7.5.4.7 `Clock spacetraveler.Level.levelTimer`

Timer, der Zeit seit Beginn hochzaehlt.

7.5.4.8 `int spacetraveler.Level.levelWidth`

Breite des Levels.

7.5.4.9 `Vector2f spacetraveler.Level.levelZiel`

[Level](#) Zielpunkt.

7.5.4.10 `Vector<SpaceObject> spacetraveler.Level.spaceObjects`

SpaceObjects im [Level](#).

7.5.4.11 `Sprite [] spacetraveler.Level.sprites`

zusaeztliche Sprites fuer Start und Ziel im [Level](#)

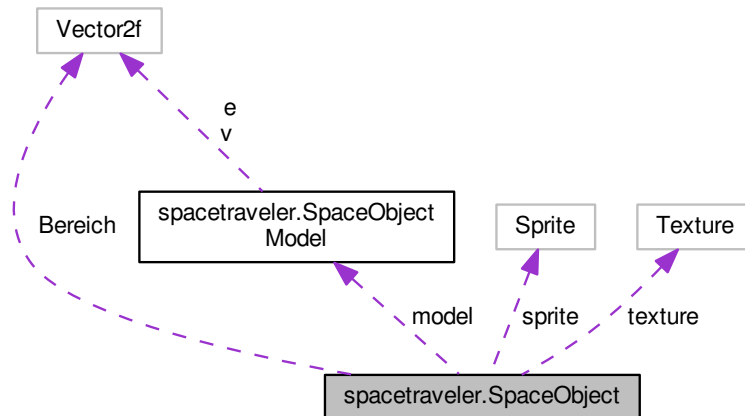
Die Dokumentation für diese Klasse wurde erzeugt aufgrund der Datei:

- `src/spacetraveler/Level.java`

7.6 spacetraveler.SpaceObject Klassenreferenz

Superklasse fuer alle Objekte, die sich auf dem Bildschirm bewegen koennen.

Zusammengehörigkeiten von spacetraveler.SpaceObject:



Öffentliche Methoden

- `SpaceObject` (String texturePath, float m, Vector2f energy, Vector2f coord, Vector2f pos, boolean gravityOn)
throws IOException
Konstruktor.
- void `bereichVerschieben` (Vector2f center)
neubesetzung von Bereich mit neuem Zentrum Ansatz: Um die Waende ueberpruefen zu koennen ueberpruefen wir die 4 Tiles, die um das aktuelle Tile sind auf index und Kollision. Sollte es Kollidieren und ist es eine wand also index = 1, prallt der spieler ab. Ansonnsten geht er weiter und sobald sein zentrum das Tile wechselt, werden die neuen 4 Tiles ueberprueft.
- void `move` ()
Bewegt das Objekt aufgrund seiner aktuellen Geschwindigkeit.
- Vector2f `getCenter` ()
- void `addAngularMomentum` (float am)
Fuegt dem Objekt an Rotationsgeschwindigkeit hinzu, d.h., beschleunigt das Objekt in der Rotation.
- float `getAngularMomentum` ()
getter fuer angularMomentum
- Sprite `getSprite` ()
getter fuer verwendetes Sprite

Öffentliche Attribute

- Texture `texture`
Textur des Spaceobjects.
- Sprite `sprite`
Sprite des Spaceobjects.

- [SpaceObjectModel model](#)

Model (fuer Berechnungen) des Spaceobjects.

- boolean [collided](#) = false
- boolean [elastisch](#) = false
- Vector2f[] [Bereich](#)

Array mit 5 Koordinaten zu Tiles.

Private Attribute

- float [angularMomentum](#)

Rotationsgeschwindigkeit.

7.6.1 Ausführliche Beschreibung

Superklasse fuer alle Objekte, die sich auf dem Bildschirm bewegen koennen.

Die Klasse vereint unser Model ([SpaceObjectModel](#)) mit den SFML-Darstellungsklassen wie Sprite und Texture. Im Code muss also nur ein [SpaceObject](#) erstellt werden, die immer dazugehoerigen anderen Klassen werden automatisch erzeugt.

7.6.2 Beschreibung der Konstruktoren und Destruktoren

7.6.2.1 `spacetraveler.SpaceObject.SpaceObject (String texturePath, float m, Vector2f energy, Vector2f coord, Vector2f pos, boolean gravityOn) throws IOException`

Konstruktor.

Parameter

<i>texturePath</i>	Pfad zur Textur, die fuer das SpaceObject verwendet werden soll
<i>m</i>	Masse des SpaceObjects
<i>energy</i>	Anfangsenergievektor des SpaceObjects
<i>coord</i>	position des SpaceObjects innerhalb des Koordinatenfeldes: Feld
<i>pos</i>	Position auf dem Spielfeld
<i>gravityOn</i>	true, wenn das Objekt von der Gravitation beeinflusst wird

Ausnahmebehandlung

<i>IOException</i>	Wenn die angegebene Textur nicht gefunden werden konnte
--------------------	---

7.6.3 Dokumentation der Elementfunktionen

7.6.3.1 `void spacetraveler.SpaceObject.addAngularMomentum (float am)`

Fuegt dem Objekt an Rotationsgeschwindigkeit hinzu, d.h., beschleunigt das Objekt in der Rotation.

Parameter

<i>am</i>	Zusaetzliche Rotationsgeschwindigkeit [Grad/s]
-----------	--

7.6.3.2 void spacetraveler.SpaceObject.bereichVerschieben (Vector2f center)

neubesetzung von Bereich mit neuem Zentrum Ansatz: Um die Waende ueberpruefen zu koennen ueberpruefen wir die 4 Tiles, die um das aktuelle [Tile](#) sind auf index und Kollision. Sollte es Kollidieren und ist es eine wand also index = 1, prallt der spieler ab. Ansonnsten geht er weiter und sobald sein zentrum das [Tile](#) wechselt, werden die neuen 4 Tiles ueberprueft.

Parameter

<i>center</i>	Position innerhalb des Koordinatenfeldes Feld
---------------	---

7.6.3.3 float spacetraveler.SpaceObject.getAngularMomentum ()

getter fuer angularMomentum

Rückgabe

Rotationsgeschwindigkeit [Grad/s]

7.6.3.4 Vector2f spacetraveler.SpaceObject.getCenter ()

Rückgabe

Globale Koordinaten der Mitte des SpaceObjects

7.6.3.5 Sprite spacetraveler.SpaceObject.getSprite ()

getter fuer verwendetes Sprite

Rückgabe

Verwendetes Spriteobjekt

7.6.3.6 void spacetraveler.SpaceObject.move ()

Bewegt das Objekt aufgrund seiner aktuellen Geschwindigkeit.

7.6.4 Dokumentation der Datenelemente

7.6.4.1 float spacetraveler.SpaceObject.angularMomentum [private]

Rotationsgeschwindigkeit.

7.6.4.2 `Vector2f [] spacetraveler.SpaceObject.Bereich`

Array mit 5 Koordinaten zu Tiles.

7.6.4.3 `boolean spacetraveler.SpaceObject.collided = false`

boolean um Doppelkollisionen mit der Wand zu minimieren

7.6.4.4 `boolean spacetraveler.SpaceObject.elastisch = false`

boolean um Doppelkollisionen mit anderenn Objekten zu minimieren

7.6.4.5 `SpaceObjectModel spacetraveler.SpaceObject.model`

Model (fuer Berechnungen) des Spaceobjects.

7.6.4.6 `Sprite spacetraveler.SpaceObject.sprite`

Sprite des Spaceobjects.

7.6.4.7 `Texture spacetraveler.SpaceObject.texture`

Textur des Spaceobjects.

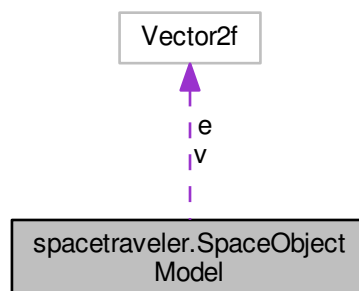
Die Dokumentation für diese Klasse wurde erzeugt aufgrund der Datei:

- `src/spacetraveler/SpaceObject.java`

7.7 `spacetraveler.SpaceObjectModel` Klassenreferenz

Rechnerisches Modell fuer SpaceObjects.

Zusammengehörigkeiten von `spacetraveler.SpaceObjectModel`:



Öffentliche Methoden

- void **addEnergy** (Vector2f energy)
Fuegt dem Objekt gerichtete Energie hinzu.
- void **setEnergy** (Vector2f energy)
Setzt die Energie des Objektes fest.
- void **setVelocity** (Vector2f velocity)
- float **getRadius** ()
getter des Radius
- Vector2f **getEnergy** ()
- Vector2f **getVelocity** ()
getter fuer v
- boolean **isGravityOn** ()
getter fuer gravityOn
- **SpaceObjectModel** (double **m**, Vector2f energy, boolean **gravityOn**, int Radius)
Konstruktor.

Private Methoden

- void **updateVelocity** ()
Berechnet den Geschwindigkeitsvektor anhand des Energievektors neu.

Private Attribute

- Vector2f **v**
Bewegungsrichtung / Geschwindigkeit.
- Vector2f **e**
Energievektor.
- double **m**
Masse.
- boolean **gravityOn**
Wirkt Gravitation auf dieses Objekt?
- int **radius**
radius des Kollisionskreises

7.7.1 Ausführliche Beschreibung

Rechnerisches Modell fuer SpaceObjects.

7.7.2 Beschreibung der Konstruktoren und Destruktoren

7.7.2.1 spacetraveler.SpaceObjectModel.SpaceObjectModel (double *m*, Vector2f *energy*, boolean *gravityOn*, int *Radius*)

Konstruktor.

Parameter

<i>m</i>	Masse des Objekts
<i>energy</i>	Anfangsenergievektor des Objekts
<i>gravityOn</i>	Gibt an ob das Objekt von Gravitationskraeften beeinflusst wird

Erzeugt von Doxygen

7.7.3 Dokumentation der Elementfunktionen

7.7.3.1 void spacetraveler.SpaceObjectModel.addEnergy (Vector2f energy)

Fuegt dem Objekt gerichtete Energie hinzu.

Parameter

<i>energy</i>	Die gerichtete Energie
---------------	------------------------

7.7.3.2 Vector2f spacetraveler.SpaceObjectModel.getEnergy ()

getter fuer Energy

Rückgabe

Energie des Objektes

7.7.3.3 float spacetraveler.SpaceObjectModel.getRadius ()

getter des Radius

Rückgabe

radius des Objekts

7.7.3.4 Vector2f spacetraveler.SpaceObjectModel.getVelocity ()

getter fuer v

Rückgabe

Geschwindigkeitsvektor v des Objekts

7.7.3.5 boolean spacetraveler.SpaceObjectModel.isGravityOn ()

getter fuer gravityOn

Rückgabe

Gravitation fuer dieses Objekt eingeschaltet?

7.7.3.6 void spacetraveler.SpaceObjectModel.setEnergy (Vector2f energy)

Setzt die Energie des Objektes fest.

Parameter

<i>energy</i>	Neue Energie des Objektes
---------------	---------------------------

7.7.3.7 void spacetraveler.SpaceObjectModel.setVelocity (Vector2f *velocity*)

Setzt die neue Geschwindigkeit des Objektes fest

Parameter

<i>velocity</i>	Neue Geschwindigkeit des Objektes
-----------------	-----------------------------------

7.7.3.8 void spacetraveler.SpaceObjectModel.updateVelocity () [private]

Berechnet den Geschwindigkeitsvektor anhand des Energievektors neu.

7.7.4 Dokumentation der Datenelemente

7.7.4.1 Vector2f spacetraveler.SpaceObjectModel.e [private]

Energievektor.

7.7.4.2 boolean spacetraveler.SpaceObjectModel.gravityOn [private]

Wirkt Gravitation auf dieses Objekt?

7.7.4.3 double spacetraveler.SpaceObjectModel.m [private]

Masse.

7.7.4.4 int spacetraveler.SpaceObjectModel.radius [private]

radius des Kollisionskreises

7.7.4.5 Vector2f spacetraveler.SpaceObjectModel.v [private]

Bewegungsrichtung / Geschwindigkeit.

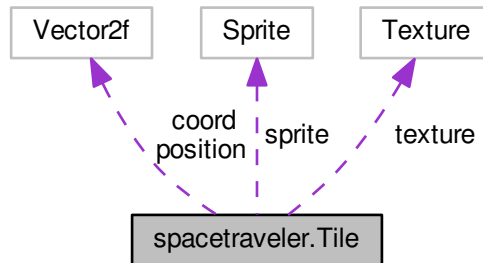
Die Dokumentation für diese Klasse wurde erzeugt aufgrund der Datei:

- src/spacetraveler/[SpaceObjectModel.java](#)

7.8 spacetraveler.Tile Klassenreferenz

Darstellung der Tiles.

Zusammengehörigkeiten von spacetraveler.Tile:



Öffentliche Methoden

- `Tile` (`Vector2f p`, `int index`) throws `IOException`
Konstruktor des Tiles: Lädt Texturdaten und erstellt das Sprite.

Öffentliche Attribute

- Sprite `sprite`
Sprite des Tiles.
- Texture `texture`
Texture des Tiles.
- Vector2f `position`
Position des Tiles.
- boolean `solid`
- int `index`
- Vector2f `coord`

7.8.1 Ausführliche Beschreibung

Darstellung der Tiles.

Diese Klasse dient dazu eine Sprite-Instanz, als auch die dazugehoerige Texture zu buendeln, um sie nicht einzeln verwalten zu muessen. Diese Klasse beinhaltet keine Methoden, sondern lediglich oeffentlich zugaeugliche Attribute. Der Konstruktor dient dazu, die fuer die Darstellung noetigen Daten einzulesen und zu laden.

7.8.2 Beschreibung der Konstruktoren und Destruktoren

7.8.2.1 spacetraveler.Tile.Tile (Vector2f p, int index) throws IOException

Konstruktor des Tiles: Lädt Texturdaten und erstellt das Sprite.

Parameter

<i>p</i>	Position des Tiles (in absoluten Bildschirmkoordinaten)
<i>index</i>	Hintergrundbild des Tiles (= Dateiname /spacetraveler/rsc/tiles/tile_bg***.png)

Ausnahmebehandlung

<i>IOException</i>	Dateizugriffsfehler
--------------------	---------------------

7.8.3 Dokumentation der Datenelemente

7.8.3.1 Vector2f spacetraveler.Tile.coord

7.8.3.2 int spacetraveler.Tile.index

7.8.3.3 Vector2f spacetraveler.Tile.position

Position des Tiles.

7.8.3.4 boolean spacetraveler.Tile.solid

7.8.3.5 Sprite spacetraveler.Tile.sprite

Sprite des Tiles.

7.8.3.6 Texture spacetraveler.Tile.texture

Texture des Tiles.

Die Dokumentation für diese Klasse wurde erzeugt aufgrund der Datei:

- src/spacetraveler/[Tile.java](#)

8 Datei-Dokumentation

8.1 src/spacetraveler/BlackHole.java-Dateireferenz

Klassen

- class [spacetraveler.BlackHole](#)
Klasse fuer Schwarze Loecher Klasse [Gravity](#) vererbt.

Pakete

- package [spacetraveler](#)

8.2 src/spacetraveler/Game.java-Dateireferenz

Klassen

- class [spacetraveler.Game](#)
Gameklasse. Enthält die [main\(\)](#)-Methode.

Pakete

- package [spacetraveler](#)

8.3 src/spacetraveler/Gravity.java-Dateireferenz

Klassen

- class [spacetraveler.Gravity](#)
Gravitationsklasse.

Pakete

- package [spacetraveler](#)

8.4 src/spacetraveler/GravityModel.java-Dateireferenz

Klassen

- class [spacetraveler.GravityModel](#)
Klasse fuer rechnerische Eigenschaften der Gravitationspunkte.

Pakete

- package [spacetraveler](#)

8.5 src/spacetraveler/Level.java-Dateireferenz

Klassen

- class [spacetraveler.Level](#)
Enthält alle fuer ein [Level](#) notwendige Objekte (Gravitationspunkte, etc...)

Pakete

- package [spacetraveler](#)

8.6 src/spacetraveler/SpaceObject.java-Dateireferenz

Klassen

- class [spacetraveler.SpaceObject](#)
Superklasse fuer alle Objekte, die sich auf dem Bildschirm bewegen koennen.

Pakete

- package [spacetraveler](#)

8.7 src/spacetraveler/SpaceObjectModel.java-Dateireferenz

Klassen

- class [spacetraveler.SpaceObjectModel](#)
Rechnerisches Modell fuer SpaceObjects.

Pakete

- package [spacetraveler](#)

8.8 src/spacetraveler/Tile.java-Dateireferenz

Klassen

- class [spacetraveler.Tile](#)
Darstellung der Tiles.

Pakete

- package [spacetraveler](#)

Index

- absVec
 - spacetraveler::Game, [7](#)
- addAngularMomentum
 - spacetraveler::SpaceObject, [18](#)
- addEnergy
 - spacetraveler::SpaceObjectModel, [22](#)
- angularMomentum
 - spacetraveler::SpaceObject, [19](#)
- Bereich
 - spacetraveler::SpaceObject, [19](#)
- bereichVerschieben
 - spacetraveler::SpaceObject, [19](#)
- BlackHole
 - spacetraveler::BlackHole, [5](#)
- blackHoles
 - spacetraveler::Level, [15](#)
- center
 - spacetraveler::BlackHole, [6](#)
 - spacetraveler::GravityModel, [12](#)
- collided
 - spacetraveler::SpaceObject, [20](#)
- contains
 - spacetraveler::Game, [7](#)
- containsFloat
 - spacetraveler::Game, [7](#)
- coord
 - spacetraveler::Tile, [25](#)
- e
 - spacetraveler::SpaceObjectModel, [23](#)
- elastisch
 - spacetraveler::SpaceObject, [20](#)
- Feld
 - spacetraveler::Level, [16](#)
- getAngularMomentum
 - spacetraveler::SpaceObject, [19](#)
- getCenter
 - spacetraveler::SpaceObject, [19](#)
- getEnergy
 - spacetraveler::GravityModel, [12](#)
 - spacetraveler::SpaceObjectModel, [22](#)
- getRadius
 - spacetraveler::SpaceObjectModel, [22](#)
- getSprite
 - spacetraveler::Gravity, [11](#)
 - spacetraveler::SpaceObject, [19](#)
- getVelocity
 - spacetraveler::SpaceObjectModel, [22](#)
- gravLeft
 - spacetraveler::Game, [9](#)
- gravRight
 - spacetraveler::Game, [9](#)

- Gravity
 - spacetraveler::Gravity, [10](#)
- gravityFields
 - spacetraveler::Level, [16](#)
- GravityModel
 - spacetraveler::GravityModel, [12](#)
- gravityOn
 - spacetraveler::SpaceObjectModel, [23](#)
- index
 - spacetraveler::Tile, [25](#)
- intersection
 - spacetraveler::Game, [7](#)
- isGravityOn
 - spacetraveler::SpaceObjectModel, [22](#)
- l
 - spacetraveler::Game, [9](#)
- Level
 - spacetraveler::Level, [14](#)
- levelHeight
 - spacetraveler::Level, [16](#)
- levelStart
 - spacetraveler::Level, [16](#)
- levelTimeAvailable
 - spacetraveler::Level, [16](#)
- levelTimer
 - spacetraveler::Level, [16](#)
- levelWidth
 - spacetraveler::Level, [16](#)
- levelZiel
 - spacetraveler::Level, [16](#)
- loadTile
 - spacetraveler::Level, [15](#)
- m
 - spacetraveler::GravityModel, [12](#)
 - spacetraveler::SpaceObjectModel, [23](#)
- main
 - spacetraveler::Game, [8](#)
- model
 - spacetraveler::Gravity, [11](#)
 - spacetraveler::SpaceObject, [20](#)
- move
 - spacetraveler::SpaceObject, [19](#)
- position
 - spacetraveler::Tile, [25](#)
- radius
 - spacetraveler::SpaceObjectModel, [23](#)
- schneiden
 - spacetraveler::Game, [8](#)
- setEnergy
 - spacetraveler::SpaceObjectModel, [22](#)
- setVelocity

- spacetraveler::SpaceObjectModel, 23
- skalar
 - spacetraveler::Game, 8
- solid
 - spacetraveler::Tile, 25
- SpaceObject
 - spacetraveler::SpaceObject, 18
- SpaceObjectModel
 - spacetraveler::SpaceObjectModel, 21
- spaceObjects
 - spacetraveler::Level, 16
- SpaceObjectsCollision
 - spacetraveler::Game, 8
- spacetraveler, 3
- spacetraveler.BlackHole, 4
- spacetraveler.Game, 6
- spacetraveler.Gravity, 9
- spacetraveler.GravityModel, 11
- spacetraveler.Level, 13
- spacetraveler.SpaceObject, 17
- spacetraveler.SpaceObjectModel, 20
- spacetraveler.Tile, 24
- spacetraveler::BlackHole
 - BlackHole, 5
 - center, 6
- spacetraveler::Game
 - absVec, 7
 - contains, 7
 - containsFloat, 7
 - gravLeft, 9
 - gravRight, 9
 - intersection, 7
 - l, 9
 - main, 8
 - schneiden, 8
 - skalar, 8
 - SpaceObjectsCollision, 8
- spacetraveler::Gravity
 - getSprite, 11
 - Gravity, 10
 - model, 11
 - sprite, 11
 - texture, 11
- spacetraveler::GravityModel
 - center, 12
 - getEnergy, 12
 - GravityModel, 12
 - m, 12
- spacetraveler::Level
 - blackHoles, 15
 - Feld, 16
 - gravityFields, 16
 - Level, 14
 - levelHeight, 16
 - levelStart, 16
 - levelTimeAvailable, 16
 - levelTimer, 16
 - levelWidth, 16
 - levelZiel, 16
 - loadTile, 15
 - spaceObjects, 16
 - sprites, 16
- spacetraveler::SpaceObject
 - addAngularMomentum, 18
 - angularMomentum, 19
 - Bereich, 19
 - bereichVerschieben, 19
 - collided, 20
 - elastisch, 20
 - getAngularMomentum, 19
 - getCenter, 19
 - getSprite, 19
 - model, 20
 - move, 19
 - SpaceObject, 18
 - sprite, 20
 - texture, 20
- spacetraveler::SpaceObjectModel
 - addEnergy, 22
 - e, 23
 - getEnergy, 22
 - getRadius, 22
 - getVelocity, 22
 - gravityOn, 23
 - isGravityOn, 22
 - m, 23
 - radius, 23
 - setEnergy, 22
 - setVelocity, 23
 - SpaceObjectModel, 21
 - updateVelocity, 23
 - v, 23
- spacetraveler::Tile
 - coord, 25
 - index, 25
 - position, 25
 - solid, 25
 - sprite, 25
 - texture, 25
 - Tile, 24
- sprite
 - spacetraveler::Gravity, 11
 - spacetraveler::SpaceObject, 20
 - spacetraveler::Tile, 25
- sprites
 - spacetraveler::Level, 16
- src/spacetraveler/BlackHole.java, 25
- src/spacetraveler/Game.java, 26
- src/spacetraveler/Gravity.java, 26
- src/spacetraveler/GravityModel.java, 26
- src/spacetraveler/Level.java, 26
- src/spacetraveler/SpaceObject.java, 27
- src/spacetraveler/SpaceObjectModel.java, 27
- src/spacetraveler/Tile.java, 27
- texture
 - spacetraveler::Gravity, 11

spacetraveler::SpaceObject, [20](#)

spacetraveler::Tile, [25](#)

Tile

spacetraveler::Tile, [24](#)

updateVelocity

spacetraveler::SpaceObjectModel, [23](#)

v

spacetraveler::SpaceObjectModel, [23](#)