BrainTrain

Generated by Doxygen 1.8.7

Thu Jun 19 2014 22:13:57

Contents

Chapter 1

BrainTrain Kurzanleitung

For english readers:
This OSG-Demo was done as part of the computer graphics I 2014 lecture of the University of Applies Sciences and Arts Hanover. This page is a short description of the project which was a requirement for passing the lecture.
Everything written here can either also be read in the source documentation or experienced by oneself by trying out this project (look for the hidden room!)
Everything here uses the GNU Public License. Use as you want, but give us credit =)!

Das Projekt "BrainTrain" wurde im Rahmen der Vorlesung CG1 an der HS Hannover im Sommersemester 2014 erstellt.
Beteiligt waren hierbei: Jonathan Spielvogel, Marcel Felix, Gleb Ostrowski, Phillip Sauer und Sebastian Huettermann.

1.1 Szene

Ziel war es eine alte, unfertige, verfallene U-Bahn-Station zu entwerfen, in der der "Spieler" sich im First-Person-↩ Shooter-Stil frei bewegen kann. Der Spieler startet am Kopf eines kleinen Niederganges, bestehend aus Treppe mit Gelaender und Rolltreppe. Direkt hinter der Startposition des Spielers befindet sich ein "geheimer Raum" (hier kann durch die Wand gelaufen werden). Der untere Teil besteht aus einem einzelnen, geschwungenen Bahnsteig (mit einem Gleis). Auf dem Bahnsteig selber befinden sich diverse Modelle, z.B: Kisten, ein altes Tickethaeuschen mit einer Folie darueber, Oelfaesser mit einer sowjetischen Flagge darueber, einem Fliesenspiegel und viele mehr. Es gibt also viel zu entdecken!

Nicht in Blender wurde hierbei folgendes erstellt:

- Die Sitzbaenke wurden in OSG modelliert, materialisiert und texturiert
- Die Bierflaschen sowie die Vase und Blume (auf dem Tickethaeuschen) und Figuren im geheimen Raum wurden als Rotationskoerper realisiert.
  - Die Bierflaschen wurdem zudem mit einem Partikelsystem aus Blender (zufaellig) im Raum verteilt. Die primaere Farbgebung (der "Cartoon Effekt" inklusive Cel Shading und Nebel) wurde hierbei ueber eigene Shader implementiert. Zudem verfuegt der Spieler ueber ein "WaffenHUD", das eine der weiteren Kamera darstellt.

1.2 Animationen

Mehrere Dinge sind animiert: Zum einen faehrt in regelmaessigen Abstaenden ein Zug das Gleis entlang. Diese Animation wurde als Animation Path in OSG realisiert. Und zum anderen weht die grosse, haengende Flagge "im Wind". Diese Animation wurde ueber den Shader realisiert. Ebenso ueber die Shader sind die Animationen der Figuren im geheimen Raum realisiert.

1.3 Elemente

In der gesamten Szene kann mit einigen Elementen interagiert werden:

- Am rechten Ende des Bahnhofs kann von einer Kiste eine "kaputte Portalgun" mittels Links-Klick aufgehoben werden. Sobald der Spieler mehr als eine Waffe traegt kann diese mittels Scrollen der des Mausrades gewechselt werden.
- Einige (drei) der herumliegenden Bierflaschen koennen getrunken werden. Hierzu muss der Spieler sich im geduckten Modus der Flasche naehern und kann diese - sofern der entsprechende Texthinweis erscheint - mit einem Links-Klick trinken. Hierbei handelt es sich um einen sehr starken Alkohol, der zwar schnell wirkt, seine Wirkung aber auch schnell wieder verliert.
  - Aus Sicherheitsgruenden darf leider nicht verraten werden, um welche Flaschen es sich handelt.
- Im Geheimraum am oberen Ende der Treppe stehen einige farbige Figuren. Naehert sich der Spieler diesen, so kann er mit einem Links-Klick u.a. Shader-Farben wechseln. Probieren Sie aus!

1.4 Szene

Die Bewegung in der Szene erfolgt im gewohnten FPS Stil. Hierbei ist sowohl eine Kollisionserkennung als auch eine "Clamp to Ground" Funktionalitaet implementiert (so dass der Spieler auf dem Boden laeuft). Die Tastaturbelegung ist hierbei die folgende:

- **W** Bewegung Vorwaerts, **S** Bewegung Rueckwaerts, **A** Nach links bewegen (nicht drehen), **D** Nach rechts bewegen (nicht drehen)
- **Maus:** Umschauen
- **Mausklick** links zur Interaktion (es erscheint immer ein Text der Interaktion "ankuendigt")
- **Leerstaste:** Springen, **L-Shift**: Sprinten, **L-Strg**: Gehen (langsamer gehen), **X:** Ducken (um z.B. an Bierflaschen heranzukommen)
- **Mausrad scrollen:** Wechseln der Waffe (sofern mehr als eine getragen wird) Mit der Taste **F** kann in den Flugmodus gewechselt werden. In diesem ist die Kollisionserkennung nicht mehr aktiv. Zu der o.g. Steuerung kommt Folgendes hinzu:
- **Q** senkrecht nach unten fliegen, **E** senkrecht nach oben fliegen

Weiteres:

- **C** aktiviert/deaktiviert den Polygon-Modus
- **L-Shift + 1** wechselt durch die Shader-Modi, die normalerweise im Geheimraum umgeschaltet werden koennen

1.5 Quellenverzeichnis

Sofern nicht anders dokumentiert (z.B. im Quellcode), handelt es sich bei allen Entwicklungen um Eigenentwicklungen. Insbesondere sind saemtliche Modelle Eigenentwicklungen.
Texturen kommen hierbei geschlossen von
http://www.cgtextures.com/
Ausnahmen sind hierbei:
UDSSR Flagge (auf den alten Oelfaessern liegend)
http://freestock.ca/soviet_union_ussr_grunge_flag_sjpg1191.jpg
Zuletzt geprueft/gesehen: 19.06.2014

Flagge mit Einhorn (in der Ecke haengend)
http://wallpoper.com/images/00/24/35/71/communism-soviet_00243571.jpg
Zuletzt geprueft/gesehen: 19.06.2014

Chapter 2

Bug List

Class brtr::FPSCameraManipulator

Jumping forward if there is no ground is not working

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

Chapter 7

Namespace Documentation

7.1 brtr Namespace Reference

Namespace for the whole BrainTrain Project.

Classes

- class AddInteractionCallbackToDrawableVisitor

  NodeVisitor for batch replacing all UserDataContainer of all Drawables.
- class AddPortalGunInteractionCallback

  InteractionCallback for adding the portal gun to the players inventar.
- class BaseInteractionCallback

  This is the TemplateClass for InteractionCallbacks.
- class Bench

  Bench class, creates a bench Object.
- struct BodyOfRotationFunction

  struct holding the function, which calculates the radius in dependece of the height. lambda (double)->double func, int end, BodyOfRotationFunction nextFunc if one wish to have more then one function then the end value and nextFunc pointer must be set accordingly the end+1 is the beginning x of the next function*
- class CelShading

  CelSading Effect, every child of this node will get the effect.
- class CelShadingTechnique

  The Technique for the cel-shading effect.
- class ControlRoom

  Control Room Class, derived from PositionAttitudeTransform, set ups the whole room as its own children.
- class DrunkenInteractionCallback

  Callback for the drunk effect.
- class FPSCameraManipulator

  A FPS style CameraManipulator with ground clamping and intersection.
- class GeometryPlacerVisitor

  NodeVisitor for batch replacing all Geometry in all visited Geodes.
- class KeyHandler

  Key Handler Class, handles all of our KeyFunctions, which do not belong to camera control (this are handled by FPSCameraManipulator)
- class ModifyMaterialVisitor

  Visitor for altering the material attributes, mainly used for objects craeted with blender.
- class ProgramSwitcherCallback

Callback for switching the postprocess programs.

- struct RenderingPipeline

  struct holding the camera for the multi-rendering passes. Also holds the program vector for the post process pass. pass0Color, pass0depth, passPostProcess, program array, count programArray The program vector is used by the KeyHandler and the InteractionItems for changing the postprocess programs

- class ToonTexSwitcherCallback

  Callback for switching the ToonTextures.

- class TrainSwitcherCallback

  Callback for switching the "trains".

- class WeaponHUD

  WeaponHUD class, provides the functions to add a HUD camera to the scene.

Functions

- void createRenderingPipeline (unsigned int width, unsigned int height, osg::Node &rootForToon, osgViewer↩ ::Viewer &viewer, RenderingPipeline &pipe, osg::Vec3f &fogColor)

  creates the rendering pipeline

- osg::ref_ptr< osg::LightSource > createLight (const osg::Vec3 &pos, int lightNum, int point=1, double spot↩ Cutoff=180, double spotExponent=0)

  creates a Light with a lightsource

- osg::ref_ptr< osg::Camera > createRTTCamera (osg::Camera::BufferComponent buffer, osg::Texture *tex, bool isAbsolute=false)

  creates a RTTCam

- osg::ref_ptr< osg::Geode > createScreenQuad (float width, float height, float scale=1.0f)

  creates a texture-ready screen quad for postprocessing

- osg::ref_ptr< osg::Camera > createHUDCamera (double left, double right, double bottom, double top)

  creates a HUD-Cam with a 2D-orthogonal projection matrix

- osg::ref_ptr< osgText::Text > createText (const osg::Vec3 &pos, const std::string &content, float size)

  creates a (arial) text object for use with a hud camera

- osg::ref_ptr< osg::Geometry > createBodyOfRotation (double height, int hsteps, int rsteps, const BodyOf↩ RotationFunction &function)

  Creates a body of rotation.

- osg::ref_ptr< osg::Geometry > createRectangle (double length, double width, int lsteps, int wsteps)

  Creates a Rectangle with TRIANGLE_STRIPS.

- osg::ref_ptr< osg::Geometry > createRectangleWithTexcoords (double length, double width, int lsteps, int wsteps)

  Creates a Rectangle with TRIANGLE_STRIPS.

- osg::ref_ptr< osg::Group > createCuboid (const double length, const double width, const double height, const double factor=6)

  Creates a Cubiod with TRIANGLE_STRIPS using the createRectangle function.

- osg::ref_ptr
  < osg::PositionAttitudeTransform > wrapInPositionAttitudeTransform (osg::Node *srcNode, const osg::Vec3d &pos)

  Return the given Node in a PositionAttitudeTransform with a given position.

- osg::ref_ptr< osg::Geometry > createBeerBottle ()

  Creates a BeerBottle with Material with the help of the BodyOfRotationFunction.

- osg::ref_ptr< osg::Geometry > createRealBottle ()

  Creates a Bottle with Material with the help of the BodyOfRotationFunction.

- osg::ref_ptr< osg::Geometry > createVase ()

  Creates a vase with Material with the help of the BodyOfRotationFunction.

- osg::ref_ptr< osg::Geometry > createStalk ()

  Creates a stalk with Material with the help of the BodyOfRotationFunction.

- osg::ref_ptr< osg::Geometry > createBud ()

  Creates a bud with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > createChessFigure ()

  Creates a "ChessFigure" with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr < osg::PositionAttitudeTransform > createVaseWithFlower ()

  combines the stalk, bud and vase in a postitionAttitudetransform
- osg::ref_ptr< osg::Geode > createCrosshair (unsigned int width, unsigned int height)

  creates a crosshair in the middle of the screen
- osg::ref_ptr< osg::Texture2D > createToonTex (std::string toonTex)

  creates a Texture2D object with the given toonTex
- osg::ref_ptr< osg::Material > createSimpleMaterial (osg::Material::Face face, const osg::Vec4 &diffuse, const osg::Vec4 &ambient, const osg::Vec4 &specular, const double shininess)

  creates a simple material
- osg::Vec3 getDimensionOfNode (osg::Node *source)

  return the dimension of a node (width, height, length)
- ref_ptr< LightSource > createLight (const Vec3 &pos, int lightNum, int point, double spotCutoff, double spotExponent)
- void createRenderingPipeline (unsigned int width, unsigned int height, osg::Node &rootForToon, osgViewer::Viewer &viewer, RenderingPipeline &pipe, Vec3f &fogColor)
- osg::Vec3 getDimensionOfNode (Node *source)

Variables

- const int collisionMask = 0x1
- const int interactionMask = 0x2
- const int interactionAndCollisionMask = collisionMask | interactionMask
- const int fakeWallMask = 0x4

7.1.1 Detailed Description

Namespace for the whole BrainTrain Project.

7.1.2 Function Documentation

7.1.2.1 osg::ref_ptr< osg::Geometry > brtr::createBeerBottle ( )

Creates a BeerBottle with Material with the help of the BodyOfRotationFunction.

Returns

ref_ptr containing the geometry

Definition at line 354 of file UtilFunctions.cpp.

7.1.2.2 osg::ref_ptr< osg::Geometry > brtr::createBodyOfRotation (double *height*, int *hsteps*, int *rsteps*, const BodyOfRotationFunction & *function*)

Creates a body of rotation.

Radius depends on height. (i.e. function x is height) Function is a modified createRectangle() from Chapter 7, CG1 Lecture Script by Frauke Sprengel

Parameters

height	the height of the body
hsteps	height resolution, more steps equals more triangles, hence better lightning, but more performance cost
rsteps	radius resolution, if rsteps value is too small, the cylinder may become a triangle or something else
function	a BodyOfRotationFunction, which determines the radius in dependence of the height

Returns

a ref_ptr<osg::Geometry> containing the body

Definition at line 153 of file UtilFunctions.cpp.

7.1.2.3 osg::ref_ptr< osg::Geometry > brtr::createBud ( )

Creates a bud with Material with the help of the BodyOfRotationFunction.

Function provided by Florian Wicke

Returns

ref_ptr containing the geometry

Definition at line 479 of file UtilFunctions.cpp.

7.1.2.4 osg::ref_ptr< osg::Geometry > brtr::createChessFigure ( )

Creates a "ChessFigure" with Material with the help of the BodyOfRotationFunction.

Function provided by Florian Wicke

Returns

ref_ptr containing the geometry

Definition at line 502 of file UtilFunctions.cpp.

7.1.2.5 osg::ref_ptr< osg::Geode > brtr::createCrosshair (unsigned int *width*, unsigned int *height*)

creates a crosshair in the middle of the screen

Parameters

width	screenwidth
height	screenheight

Returns

ref_ptr conatining the geode with the crosshair

Definition at line 557 of file UtilFunctions.cpp.

7.1.2.6 ref_ptr< osg::Group > brtr::createCuboid (const double *length*, const double *width*, const double *height*, const double *factor* = 6)

Creates a Cubiod with TRIANGLE_STRIPS using the createRectangle function.

Uses 6 Rectangles and creates a Cuboid of it.

Parameters

length	desired length of the Cuboid
width	desired length of the Cuboid
height	desired length of the Cuboid
factor	the higher the number the greater the resolution in all dimension

Returns

a osg::Group containing the Cuboid

Definition at line 247 of file UtilFunctions.cpp.

7.1.2.7 ref_ptr< Camera > brtr::createHUDCamera (double *left*, double *right*, double *bottom*, double *top*)

creates a HUD-Cam with a 2D-orthogonal projection matrix

Original Function by Rui Wang/Xuelei Qian from OSG 3 Cookbook, Packt Publishing, 2012

Parameters

left	left bound of the projection matrix
right	right bound of the projection matrix
bottom	bottom bound of the projection matrix
top	top bound of the projection matrix

Returns

the created HUD Camera in a ref_ptr

Definition at line 63 of file UtilFunctions.cpp.

7.1.2.8 osg::ref_ptr<osg::LightSource> brtr::createLight (const osg::Vec3 & *pos*, int *lightNum*, int *point* = `1`, double *spotCutoff* = `180`, double *spotExponent* = `0`)

creates a Light with a lightsource

Parameters

pos	light position
lightNum	gl_light num (must be 0 to 7)
point	1 = point light, 0 = directional light
spotCutoff	
spotExponent	

Returns

a ref_ptr containing the LightSource

7.1.2.9 ref_ptr<LightSource> brtr::createLight (const Vec3 & *pos*, int *lightNum*, int *point*, double *spotCutoff*, double *spotExponent*)

Definition at line 224 of file UtilFunctions.cpp.

7.1.2.10 osg::ref_ptr< osg::Geometry > brtr::createRealBottle ( )

Creates a Bottle with Material with the help of the BodyOfRotationFunction.

Function provided by Florian Wicke

Returns

ref_ptr containing the geometry

Definition at line 379 of file UtilFunctions.cpp.

7.1.2.11 ref_ptr< Geometry > brtr::createRectangle (double *length*, double *width*, int *lsteps*, int *wsteps*)

Creates a Rectangle with TRIANGLE_STRIPS.

Function is copy/pasted from Chapter 7, CG1 Lecture Script by Frauke Sprengel

Parameters

length	desired length of the rectangle
width	desired width of the rectangle
lsteps	the higher the number the greater the resolution in the length dimension
wsteps	the higher the number the greater the resolution in the width dimension

Returns

a ref_ptr<osg::Geometry> containing the rectangle

Definition at line 74 of file UtilFunctions.cpp.

7.1.2.12 ref_ptr< Geometry > brtr::createRectangleWithTexcoords (double *length*, double *width*, int *lsteps*, int *wsteps*)

Creates a Rectangle with TRIANGLE_STRIPS.

Function is copy/pasted from Chapter 7, CG1 Lecture Script by Frauke Sprengel Added the TexCoordArray. If the width/length ratio lower than 1:4 or 4:1 the texture coordiantes are streched to fit the Rectangle

Parameters

length	desired length of the rectangle
width	desired width of the rectangle
lsteps	the higher the number the greater the resolution in the length dimension
wsteps	the higher the number the greater the resolution in the width dimension

Returns

a ref_ptr<osg::Geometry> containing the rectangle

Definition at line 120 of file UtilFunctions.cpp.

7.1.2.13 void brtr::createRenderingPipeline (unsigned int *width*, unsigned int *height*, osg::Node & *rootForToon*, osgViewer::Viewer & *viewer*, RenderingPipeline & *pipe*, osg::Vec3f & *fogColor*)

creates the rendering pipeline

Creates the cameras and textures, attachs the textures to the cameras, set the projectionmatrix

Parameters

width	the width of the texture, should be screenwidth
height	the height of the texture, should be screenheight

rootForToon	Node which the CelShade effect will be applied to
viewer	clipping pane and projectionmatrix will be set on this viewers cam
pipe	pipe struct which should be filled

7.1.2.14 void brtr::createRenderingPipeline (unsigned int *width*, unsigned int *height*, osg::Node & *rootForToon*, osgViewer::Viewer & *viewer*, RenderingPipeline & *pipe*, Vec3f & *fogColor*)

Definition at line 285 of file UtilFunctions.cpp.

7.1.2.15 ref_ptr< osg::Camera > brtr::createRTTCamera (osg::Camera::BufferComponent *buffer*, osg::Texture * *tex*, bool *isAbsolute* = `false`)

creates a RTTCam

Original Function by Rui Wang/Xuelei Qian from OSG 3 Cookbook, Packt Publishing, 2012

Parameters

buffer	which buffer should be written to texture
tex	on this texture the buffer will be written to
isAbsolute	absolute or relative reference frame

Returns

a ref_ptr holding the camera

Definition at line 27 of file UtilFunctions.cpp.

7.1.2.16 ref_ptr< osg::Geode > brtr::createScreenQuad (float *width*, float *height*, float *scale* = `1.0f`)

creates a texture-ready screen quad for postprocessing

Original Function by Rui Wang/Xuelei Qian from OSG 3 Cookbook, Packt Publishing, 2012

Parameters

width	width of the quad
height	height of the quad
scale	scale of the quad

Returns

a ref_ptr caontaining the geode with the quad

Definition at line 49 of file UtilFunctions.cpp.

7.1.2.17 osg::ref_ptr< osg::Material > brtr::createSimpleMaterial (osg::Material::Face *face*, const osg::Vec4 & *diffuse*, const osg::Vec4 & *ambient*, const osg::Vec4 & *specular*, const double *shininess*)

creates a simple material

Parameters

diffuse	diffuse lighting
ambient	ambient lighting
specular	specular lighting
shininess	the shininess

Returns

the material as a osg::ref_ptr<osg::Material>

Definition at line 599 of file UtilFunctions.cpp.

7.1.2.18 osg::ref_ptr< osg::Geometry > brtr::createStalk ( )

Creates a stalk with Material with the help of the BodyOfRotationFunction.

Function provided by Florian Wicke

Returns

ref_ptr containing the geometry

Definition at line 453 of file UtilFunctions.cpp.

7.1.2.19 osg::ref_ptr< osgText::Text > brtr::createText (const osg::Vec3 & *pos*, const std::string & *content*, float *size*)

creates a (arial) text object for use with a hud camera

Original Function by Rui Wang/Xuelei Qian from OSG 3 Cookbook, Packt Publishing, 2012

Parameters

pos	postion of the text in x_y plane
content	
size	

Returns

a ref_ptr containing the osgText::Text object

Definition at line 212 of file UtilFunctions.cpp.

7.1.2.20 osg::ref_ptr< osg::Texture2D > brtr::createToonTex (std::string *toonTex*)

creates a Texture2D object with the given toonTex

Parameters

filename	of the toontex

Returns

ref_ptr containing the Texture2D

Definition at line 616 of file UtilFunctions.cpp.

7.1.2.21 osg::ref_ptr< osg::Geometry > brtr::createVase ( )

Creates a vase with Material with the help of the BodyOfRotationFunction.

Function provided by Florian Wicke

Returns

ref_ptr containing the geometry

Definition at line 414 of file UtilFunctions.cpp.

7.1.2.22 osg::ref_ptr< osg::PositionAttitudeTransform > brtr::createVaseWithFlower ( )

combines the stalk, bud and vase in a postitionAttitudetransform

Returns

a ref_ptr containing a positionAttitudeTransform containing the vase with a flower

Definition at line 583 of file UtilFunctions.cpp.

7.1.2.23 osg::Vec3 brtr::getDimensionOfNode (osg::Node * *source*)

return the dimension of a node (width, height, length)

Parameters

source	node, which dimension one want to know

Returns

vec3 holding the dimensions

7.1.2.24 osg::Vec3 brtr::getDimensionOfNode (Node * *source*)

Definition at line 608 of file UtilFunctions.cpp.

7.1.2.25 osg::ref_ptr< osg::PositionAttitudeTransform > brtr::wrapInPositionAttitudeTransform (osg::Node * *srcNode,* const osg::Vec3d & *pos*)

Return the given Node in a PositionAttitudeTransform with a given position.

Parameters

srcNode	the Node which should be moved
pos	the relative position change

Returns

a osg::PositionAttitudeTransform containing the Cuboid

Definition at line 240 of file UtilFunctions.cpp.

7.1.3 Variable Documentation

7.1.3.1 const int brtr::collisionMask = 0x1

Definition at line 27 of file UtilFunctions.h.

7.1.3.2 const int brtr::fakeWallMask = 0x4

Definition at line 30 of file UtilFunctions.h.

7.1.3.3 const int brtr::interactionAndCollisionMask = collisionMask | interactionMask

Definition at line 29 of file UtilFunctions.h.

7.1.3.4 const int brtr::interactionMask = 0x2

Definition at line 28 of file UtilFunctions.h.

Chapter 8

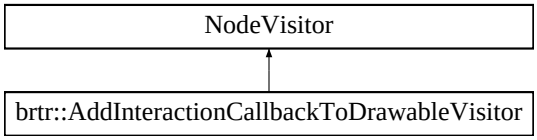
Class Documentation

8.1 brtr::AddInteractionCallbackToDrawableVisitor Class Reference

NodeVisitor for batch replacing all UserDataContainer of all Drawables.

`#include <AddInteractionCallbackToDrawableVisitor.h>`

Inheritance diagram for brtr::AddInteractionCallbackToDrawableVisitor:



Public Member Functions

- AddInteractionCallbackToDrawableVisitor (brtr::BaseInteractionCallback ∗callbackToAdd)
  Constructor.
- virtual void apply (osg::Geode &geode)

Private Attributes

- osg::ref_ptr
  < osg::DefaultUserDataContainer > _containerToAdd

8.1.1 Detailed Description

NodeVisitor for batch replacing all UserDataContainer of all Drawables.

New Container contains the provided InteractionCallback. Mainly used for making imported objects (e.g. from blender) interact-able.

Author

Gleb Ostrowski

Version

1.0

Date

2014

Precondition

needs a Node which will accept it. Should have some Geode's for this to work

Copyright

GNU Public License.

Definition at line 17 of file AddInteractionCallbackToDrawableVisitor.h.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 brtr::AddInteractionCallbackToDrawableVisitor::AddInteractionCallbackToDrawableVisitor (brtr::BaseInteractionCallback * *callbackToAdd*)

Constructor.

Parameters

callbackToAdd	the callback which should be add to all drawables in the object

Returns

Definition at line 5 of file AddInteractionCallbackToDrawableVisitor.cpp.

8.1.3 Member Function Documentation

8.1.3.1 void brtr::AddInteractionCallbackToDrawableVisitor::apply (osg::Geode & *geode*) [virtual]

Definition at line 11 of file AddInteractionCallbackToDrawableVisitor.cpp.

8.1.4 Member Data Documentation

8.1.4.1 osg::ref_ptr<osg::DefaultUserDataContainer> brtr::AddInteractionCallbackToDrawableVisitor::_containerToAdd [private]

Definition at line 28 of file AddInteractionCallbackToDrawableVisitor.h.

The documentation for this class was generated from the following files:

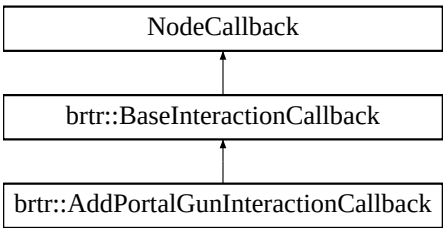
- header/AddInteractionCallbackToDrawableVisitor.h
- Util/AddInteractionCallbackToDrawableVisitor.cpp

8.2 brtr::AddPortalGunInteractionCallback Class Reference

InteractionCallback for adding the portal gun to the players inventar.

`#include <AddPortalGunInteractionCallback.h>`

Inheritance diagram for brtr::AddPortalGunInteractionCallback:



Public Member Functions

- AddPortalGunInteractionCallback (osg::Node ∗weaponHUD, osg::Camera ∗hudCam, osg::Switch ∗switcher, int width, int height)
  Constructor.
- virtual void setText ()
  sets the text on screen. Subclasses must override to set its own (info)text

Protected Member Functions

- virtual void interact (osg::Node ∗, osg::NodeVisitor ∗)
  the interaction logic must be implemented be the children in this method

Private Attributes

- osg::ref_ptr< osg::Switch > _switcher

Additional Inherited Members

8.2.1 Detailed Description

InteractionCallback for adding the portal gun to the players inventar.

Author

Gleb Ostrowski

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 14 of file AddPortalGunInteractionCallback.h.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 brtr::AddPortalGunInteractionCallback::AddPortalGunInteractionCallback (osg::Node ∗ *weaponHUD,* osg::Camera ∗ *hudCam,* osg::Switch ∗ *switcher,* int *width,* int *height*)

Constructor.

Parameters

weaponHUD	weaponHUD which provides the method to add the portalgun to it
hudCam	
switcher	switch-node which contains the portalGun object, will be switched to off upon interaction, removing the portal Gun from the world
width	screenWidth
height	screenHeight

Returns

Definition at line 5 of file AddPortalGunInteractionCallback.cpp.

8.2.3 Member Function Documentation

8.2.3.1 void brtr::AddPortalGunInteractionCallback::interact (osg::Node *, osg::NodeVisitor *) [protected], [virtual]

the interaction logic must be implemented be the children in this method

Implements brtr::BaseInteractionCallback.

Definition at line 13 of file AddPortalGunInteractionCallback.cpp.

8.2.3.2 void brtr::AddPortalGunInteractionCallback::setText ( ) [virtual]

sets the text on screen. Subclasses must override to set its own (info)text

Implements brtr::BaseInteractionCallback.

Definition at line 9 of file AddPortalGunInteractionCallback.cpp.

8.2.4 Member Data Documentation

8.2.4.1 osg::ref_ptr<osg::Switch> brtr::AddPortalGunInteractionCallback::_switcher [private]

Definition at line 32 of file AddPortalGunInteractionCallback.h.

The documentation for this class was generated from the following files:

- header/AddPortalGunInteractionCallback.h
- Callbacks/AddPortalGunInteractionCallback.cpp

8.3 AnimationCreator Class Reference

`#include <AnimationCreater.h>`

Public Member Functions

- double getAngleRad (osg::Vec3 pointA, osg::Vec3 pointB)

  Creator of the Animation Path for the Train-Simulation.
- osg::AnimationPath * createAnimationPath (float time)

  Creates the animation path.

8.3.1 Detailed Description

Definition at line 5 of file AnimationCreater.h.

8.3.2 Member Function Documentation

8.3.2.1 osg::AnimationPath * AnimationCreator::createAnimationPath (float *time*)

Creates the animation path.

The Method creates the Train AnimationPath. Each vector will be included in the AnimationPath, together with the correct rotation between two points.

Parameters

time	is the time that the train will take between two vectors, low time = fast train.

Returns

the complete AnimationPath for the train

Definition at line 40 of file AnimationCreater.cpp.

8.3.2.2 double AnimationCreator::getAngleRad (osg::Vec3 *pointA*, osg::Vec3 *pointB*)

Creator of the Animation Path for the Train-Simulation.

Author

Philip Sauer

Version

1.0

Date

2014 Calculate the angle between two Vectors

The Method calculates the dotproduct between two vectors and devides it with the length of both vectors. (vectorA * vectorB) / (|vectorA| * |vectorB|)

Parameters

pointA	the starting Vector
pointB	the end Vector

Returns

the angle in radian

Definition at line 16 of file AnimationCreater.cpp.

The documentation for this class was generated from the following files:

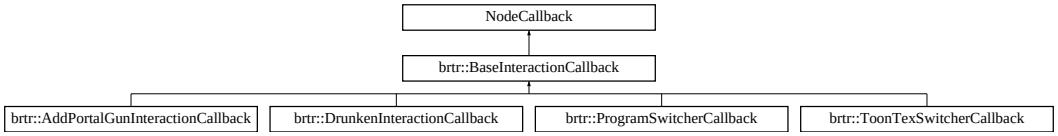
- header/AnimationCreater.h
- Animation/AnimationCreater.cpp

8.4 brtr::BaseInteractionCallback Class Reference

This is the TemplateClass for InteractionCallbacks.

`#include <BaseInteractionCallback.h>`

Inheritance diagram for brtr::BaseInteractionCallback:



Public Member Functions

- BaseInteractionCallback (osg::Node *attachTo, osg::Camera *hudCam, int width, int height)

  Constructor.
- virtual void operator() (osg::Node *node, osg::NodeVisitor *nv)
- virtual void setText ()=0

  sets the text on screen. Subclasses must override to set its own (info)text
- void clearText ()
- void reactivate ()
- osg::ref_ptr< osg::Node > getNode () const
- void setNode (osg::ref_ptr< osg::Node > val)

Protected Member Functions

- virtual void interact (osg::Node *, osg::NodeVisitor *)=0

  the interaction logic must be implemented be the children in this method

Protected Attributes

- osg::ref_ptr< osg::Node > _attachTo
- osg::ref_ptr< osg::Camera > _hudCam
- bool _done
- osg::ref_ptr< osgText::Text > _text

8.4.1 Detailed Description

This is the TemplateClass for InteractionCallbacks.

InteractionCallbacks are set as an UserObject in an UserDataContainer of a Geometry.
Furthermore, the right NodeMask (brtr::interactionMask) must be set.
Every subclass must override the setText() and interact() method.
After the child is finished with its work, it must set the done-flag to the value true
The client must check if there is a valid Geometry with a valid InteractionCallback and call
the setText Method to set the text on screen. If the user interacts (e.g by clicking a mouse button)
the client must attach the callback to the node with getNode()->addUpdateCallback(), if its not already attached.
In this case the client must call reactivate() to reactivate the callback. (which basicly sets the done flag back to false)
clearText() should be called, if the clients wants to remove the message from the screen (e.g. if the player no longer looks at the geometry).

Author

Gleb Ostrowski

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 24 of file BaseInteractionCallback.h.

8.4.2 Constructor & Destructor Documentation

8.4.2.1 brtr::BaseInteractionCallback::BaseInteractionCallback (osg::Node * *attachTo*, osg::Camera * *hudCam*, int *width*, int *height*)

Constructor.

Parameters

attachTo	the node the Callback will be attached to upon interaction
hudCam	the HUDCam, where the text will appear
width	screenWidth
height	screenHeight

Definition at line 6 of file BaseInteractionCallback.cpp.

8.4.3 Member Function Documentation

8.4.3.1 void brtr::BaseInteractionCallback::clearText ( )

Definition at line 31 of file BaseInteractionCallback.cpp.

8.4.3.2 osg::ref_ptr< osg::Node > brtr::BaseInteractionCallback::getNode ( ) const

Definition at line 27 of file BaseInteractionCallback.cpp.

8.4.3.3 virtual void brtr::BaseInteractionCallback::interact (osg::Node * , osg::NodeVisitor *) [protected], [pure virtual]

the interaction logic must be implemented be the children in this method

Implemented in brtr::ProgramSwitcherCallback, brtr::ToonTexSwitcherCallback, brtr::DrunkenInteractionCallback, and brtr::AddPortalGunInteractionCallback.

8.4.3.4 void brtr::BaseInteractionCallback::operator() (osg::Node * *node*, osg::NodeVisitor * *nv*) [virtual]

Definition at line 16 of file BaseInteractionCallback.cpp.

8.4.3.5 void brtr::BaseInteractionCallback::reactivate ( )

Definition at line 35 of file BaseInteractionCallback.cpp.

8.4.3.6 void brtr::BaseInteractionCallback::setNode (osg::ref_ptr< osg::Node > *val*)

Definition at line 23 of file BaseInteractionCallback.cpp.

8.4.3.7 virtual void brtr::BaseInteractionCallback::setText ( ) [pure virtual]

sets the text on screen. Subclasses must override to set its own (info)text

Implemented in brtr::ProgramSwitcherCallback, brtr::ToonTexSwitcherCallback, brtr::DrunkenInteractionCallback, and brtr::AddPortalGunInteractionCallback.

8.4.4 Member Data Documentation

8.4.4.1 osg::ref_ptr<osg::Node> brtr::BaseInteractionCallback::_attachTo [protected]

Definition at line 51 of file BaseInteractionCallback.h.

8.4.4.2 bool brtr::BaseInteractionCallback::_done [protected]

Definition at line 53 of file BaseInteractionCallback.h.

8.4.4.3 osg::ref_ptr<osg::Camera> brtr::BaseInteractionCallback::_hudCam [protected]

Definition at line 52 of file BaseInteractionCallback.h.

8.4.4.4 osg::ref_ptr<osgText::Text> brtr::BaseInteractionCallback::_text [protected]

Definition at line 54 of file BaseInteractionCallback.h.

The documentation for this class was generated from the following files:

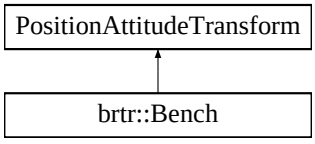
- header/BaseInteractionCallback.h
- Callbacks/BaseInteractionCallback.cpp

8.5 brtr::Bench Class Reference

Bench class, creates a bench Object.

`#include <Bench.h>`

Inheritance diagram for brtr::Bench:



Public Member Functions

- Bench (const Vec3 &pcenter=Vec3(0, 0, 0), const double plength=8)
- ref_ptr < PositionAttitudeTransform > getHitbox (const double alpha, double height=8)

  return the Hitbox of the Bench
- Bench (const Bench &, const CopyOp &copyop=CopyOp::SHALLOW_COPY)
- ~Bench ()

Private Member Functions

- void initBench (const double plength)

  initialize the bench
- ref_ptr< Material > createIronMaterial ()

  create the material for iron objects
- ref_ptr< Material > createWoodMaterial ()

  create the material for wood objects
- ref_ptr< Group > createLeg ()
- ref_ptr< Group > createBar ()
- ref_ptr< Group > createSeat (const double width)

  creates the seat
- ref_ptr< Group > createArmrest (double radius, double width, double length, double totalwidth)

  creates the armrest
- ref_ptr< Geometry > createArmrestSidesFrontBack (double radius, double width, int lsteps, int wsteps, bool flip=true)

  creates the front/back for the armrest
- ref_ptr< Geometry > createArmrestSidesLeftRight (double length, double width, int lsteps, int wsteps, bool flip=true)

  creates the left/rigth for the armrest
- ref_ptr< DrawElementsUInt > getPrimitiveSetforARectangle (int lsteps, int wsteps)

  creates a primitives set for the getRectangle function

Private Attributes

- Vec3 center
- double length
- ref_ptr< Group > bench

8.5.1 Detailed Description

Bench class, creates a bench Object.

creates a bench with a given length at a given position. The length has to be between 2 and 30

Author

Marcel Felix

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 24 of file Bench.h.

8.5.2 Constructor & Destructor Documentation

8.5.2.1 brtr::Bench::Bench (const Vec3 & *pcenter* = `Vec3(0, 0, 0)`, const double *plength* = `8`)

Definition at line 7 of file Bench.cpp.

8.5.2.2 brtr::Bench::Bench (const Bench & *copy*, const CopyOp & *copyop* = `CopyOp::SHALLOW_COPY`)

Definition at line 18 of file Bench.cpp.

8.5.2.3 brtr::Bench::~Bench ( )

Definition at line 21 of file Bench.cpp.

8.5.3 Member Function Documentation

8.5.3.1 ref_ptr< Group > brtr::Bench::createArmrest (double *radius*, double *width*, double *length*, double *totalwidth*) `[private]`

creates the armrest

Parameters

radius	the distance between bench and armrest
width	width of the armrest
length	length of the armrest
totalwidth	width of the bar on the armrest

Definition at line 252 of file Bench.cpp.

8.5.3.2 ref_ptr< Geometry > brtr::Bench::createArmrestSidesFrontBack (double *radius*, double *width*, int *lsteps*, int *wsteps*, bool *flip* = `true`) `[private]`

creates the front/back for the armrest

Parameters

radius	the distance between bench and armrest
width	width of the armrest
length	length of the armrest
flip	switch between front and back creation

Definition at line 140 of file Bench.cpp.

8.5.3.3 ref_ptr< Geometry > brtr::Bench::createArmrestSidesLeftRight (double *length,* double *width,* int *lsteps,* int *wsteps,* bool *flip* = `true`) [private]

creates the left/rigth for the armrest

Parameters

radius	the distance between bench and armrest
width	width of the armrest
length	length of the armrest
flip	switch between front and back creation

Definition at line 84 of file Bench.cpp.

8.5.3.4 ref_ptr< Group > brtr::Bench::createBar ( ) [private]

Definition at line 331 of file Bench.cpp.

8.5.3.5 ref_ptr< Material > brtr::Bench::createIronMaterial ( ) [private]

create the material for iron objects

Definition at line 53 of file Bench.cpp.

8.5.3.6 ref_ptr< Group > brtr::Bench::createLeg ( ) [private]

Definition at line 190 of file Bench.cpp.

8.5.3.7 ref_ptr< Group > brtr::Bench::createSeat (const double *width*) [private]

creates the seat

Parameters

width	the width/length of the Seat

Definition at line 212 of file Bench.cpp.

8.5.3.8 ref_ptr< Material > brtr::Bench::createWoodMaterial ( ) [private]

create the material for wood objects

Definition at line 62 of file Bench.cpp.

8.5.3.9 ref_ptr< PositionAttitudeTransform > brtr::Bench::getHitbox (const double *alpha*, double *height* = 8)

return the Hitbox of the Bench

Parameters

alpha	
height	the height of the hitbox. height < 0 will use the height of the bench

Returns

the hitbox as a PositionAttitudeTransform with the given alpha value

Definition at line 24 of file Bench.cpp.

8.5.3.10 ref_ptr< DrawElementsUInt > brtr::Bench::getPrimitiveSetforARectangle (int *lsteps,* int *wsteps*) [private]

creates a primitives set for the getRectangle function

parts of the function are copy/pasted from Chapter 7, CG1 Lecture Script by Frauke Sprengel

Parameters

lsteps	
wsteps	

Returns

a ref_ptr<DrawElementsUInt> containing the primitives set

Definition at line 71 of file Bench.cpp.

8.5.3.11 void brtr::Bench::initBench (const double *plength*) [private]

initialize the bench

Parameters

plength	the length of the bench

Definition at line 360 of file Bench.cpp.

8.5.4 Member Data Documentation

8.5.4.1 ref_ptr<Group> brtr::Bench::bench [private]

Definition at line 108 of file Bench.h.

8.5.4.2 Vec3 brtr::Bench::center [private]

Definition at line 105 of file Bench.h.

8.5.4.3 double brtr::Bench::length [private]

Definition at line 107 of file Bench.h.

The documentation for this class was generated from the following files:

- header/Bench.h
- Objects/Bench.cpp

8.6 brtr::BodyOfRotationFunction Struct Reference

struct holding the function, which calculates the radius in dependece of the height. lambda (double)->double func, int end, BodyOfRotationFunction* nextFunc if one wish to have more then one function then the end value and nextFunc pointer must be set accordingly the end+1 is the beginning x of the next function

`#include <UtilFunctions.h>`

Public Member Functions

- double derivation (double x) const

Public Attributes

- std::function< double(double)> func

  the function
- double end

  the end value of the function, should be less or equal createBodyOfRotation::height
- const BodyOfRotationFunction * nextFunc

  if end is less then createBodyOfRotation::height, must point towards the next function which shall be used from end

8.6.1 Detailed Description

struct holding the function, which calculates the radius in dependece of the height. lambda (double)->double func, int end, BodyOfRotationFunction* nextFunc if one wish to have more then one function then the end value and nextFunc pointer must be set accordingly the end+1 is the beginning x of the next function

Definition at line 40 of file UtilFunctions.h.

8.6.2 Member Function Documentation

8.6.2.1 double brtr::BodyOfRotationFunction::derivation (double *x*) const [inline]

Definition at line 44 of file UtilFunctions.h.

8.6.3 Member Data Documentation

8.6.3.1 double brtr::BodyOfRotationFunction::end

the end value of the function, should be less or equal createBodyOfRotation::height

Definition at line 42 of file UtilFunctions.h.

8.6.3.2 std::function<double(double)> brtr::BodyOfRotationFunction::func

the function

Definition at line 41 of file UtilFunctions.h.

8.6.3.3 const BodyOfRotationFunction* brtr::BodyOfRotationFunction::nextFunc

if end is less then createBodyOfRotation::height, must point towards the next function which shall be used from end

Definition at line 43 of file UtilFunctions.h.

The documentation for this struct was generated from the following file:

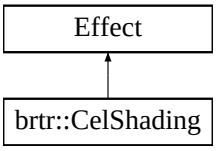
- header/UtilFunctions.h

8.7 brtr::CelShading Class Reference

CelSading Effect, every child of this node will get the effect.

`#include <CelShading.h>`

Inheritance diagram for brtr::CelShading:



Public Member Functions

- CelShading (bool secondPass=true, std::string vertSource="celShader.vert")

  Constructor.
- CelShading (const CelShading &copy, const osg::CopyOp &copyop=osg::CopyOp::SHALLOW_COPY)
- META_Effect (null, CelShading,"CelShading","This effect implements a technique called 'Cel-Shading' to produce a ""cartoon-style (non photorealistic) rendering. Two passes are required: ""the first one draws solid surfaces, the second one draws the outlines. ""Vertices Shader, Toon Texture pass can be customize upon creating.","Marco Jez; OGLSL port by Mike Weiblen, adaptions by Gleb Ostrowski ")

Protected Member Functions

- virtual ~CelShading ()
- bool define_techniques ()

Private Attributes

- osg::ref_ptr< osg::Material > _material
- osg::ref_ptr< osg::LineWidth > _lineWidth
- bool _secondPass
- std::string _vertSource

8.7.1 Detailed Description

CelSading Effect, every child of this node will get the effect.

This effect implements a technique called 'Cel-Shading' to produce a cartoon-style (non photorealistic) rendering. Two passes are required:
the first one draws solid surfaces, the second one draws the outlines.

Author

Gleb Ostrowski

Version

1.0

Date

2014

Precondition

In Texture Layer 1 (stateset) must be the ToonTexture set

Copyright

GNU Public License.

Definition at line 18 of file CelShading.h.

8.7.2 Constructor & Destructor Documentation

8.7.2.1 brtr::CelShading::CelShading (bool *secondPass* = `true`, std::string *vertSource* = `"celShader.vert"`)

Constructor.

Parameters

secondPass	if false, no outlines are being drawn
vertSource	one can set explicitly the vertex shader

Definition at line 97 of file CelShading.cpp.

8.7.2.2 brtr::CelShading::CelShading (const CelShading & *copy*, const osg::CopyOp & *copyop* = `osg::CopyOp::SHALLOW_COPY`)

Definition at line 104 of file CelShading.cpp.

8.7.2.3 virtual brtr::CelShading::~CelShading ( ) [inline], [protected], [virtual]

Definition at line 43 of file CelShading.h.

8.7.3 Member Function Documentation

8.7.3.1 bool brtr::CelShading::define_techniques ( ) [protected]

Definition at line 110 of file CelShading.cpp.

8.7.3.2 brtr::CelShading::META_Effect (null , CelShading , "CelShading" , "This effect implements a technique called 'Cel-Shading' to produce a ""cartoon-style (non photorealistic) rendering. Two passes are required: ""the first one draws solid *surfaces*, the second one draws the outlines.""Vertices *Shader*, Toon Texture pass can be customize upon creating." , "Marco Jez; OGLSL port by Mike *Weiblen*, adaptions by Gleb Ostrowski")

8.7.4 Member Data Documentation

8.7.4.1 osg::ref_ptr<osg::LineWidth> brtr::CelShading::_lineWidth [private]

Definition at line 49 of file CelShading.h.

8.7.4.2 osg::ref_ptr<osg::Material> brtr::CelShading::_material [private]

Definition at line 48 of file CelShading.h.

8.7.4.3 bool brtr::CelShading::_secondPass [private]

Definition at line 50 of file CelShading.h.

8.7.4.4 std::string brtr::CelShading::_vertSource [private]

Definition at line 51 of file CelShading.h.

The documentation for this class was generated from the following files:

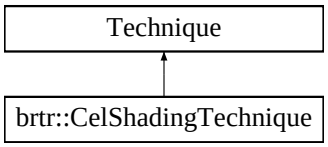
- header/CelShading.h

- Shader/CelShading.cpp

8.8 brtr::CelShadingTechnique Class Reference

The Technique for the cel-shading effect.

Inheritance diagram for brtr::CelShadingTechnique:



Public Member Functions

- CelShadingTechnique (osg::Material *material, osg::LineWidth *lineWidth, bool secondPass, std::string vertSource)

Protected Member Functions

- void define_passes ()

Private Attributes

- osg::ref_ptr< osg::Material > _material
- osg::ref_ptr< osg::LineWidth > _lineWidth
- std::string _toonTex
- bool _secondPass
- std::string _vertSource

8.8.1 Detailed Description

The Technique for the cel-shading effect.

Author

Gleb Ostrowski

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 22 of file CelShading.cpp.

8.8.2 Constructor & Destructor Documentation

8.8.2.1 brtr::CelShadingTechnique::CelShadingTechnique (osg::Material * *material,* osg::LineWidth * *lineWidth,* bool *secondPass,* std::string *vertSource*) [inline]

Definition at line 24 of file CelShading.cpp.

8.8.3 Member Function Documentation

8.8.3.1 void brtr::CelShadingTechnique::define_passes ( ) [inline], [protected]

Definition at line 33 of file CelShading.cpp.

8.8.4 Member Data Documentation

8.8.4.1 osg::ref_ptr<osg::LineWidth> brtr::CelShadingTechnique::_lineWidth [private]

Definition at line 89 of file CelShading.cpp.

8.8.4.2 osg::ref_ptr<osg::Material> brtr::CelShadingTechnique::_material [private]

Definition at line 88 of file CelShading.cpp.

8.8.4.3 bool brtr::CelShadingTechnique::_secondPass [private]

Definition at line 91 of file CelShading.cpp.

8.8.4.4 std::string brtr::CelShadingTechnique::_toonTex [private]

Definition at line 90 of file CelShading.cpp.

8.8.4.5 std::string brtr::CelShadingTechnique::_vertSource [private]

Definition at line 92 of file CelShading.cpp.

The documentation for this class was generated from the following file:

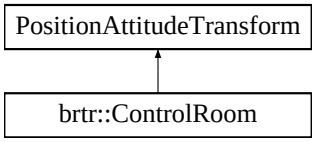
- Shader/CelShading.cpp

8.9 brtr::ControlRoom Class Reference

Control Room Class, derived from PositionAttitudeTransform, set ups the whole room as its own children.

`#include <ControlRoom.h>`

Inheritance diagram for brtr::ControlRoom:



Public Member Functions

- ControlRoom (double roomSize, int lod, brtr::ToonTexSwitcherCallback &toonCallback, brtr::ProgramSwitcherCallback &programCallback)

  Constructor.

Protected Member Functions

- ~ControlRoom ()

Private Member Functions

- osg::ref_ptr< osg::Group > createRoomSurrounding (double roomSize, int lod)
- osg::ref_ptr< osg::Group > createChessFigures (brtr::ToonTexSwitcherCallback &toonCallback, brtr::ProgramSwitcherCallback &programCallback)
- osg::ref_ptr< osg::Material > createMaterial (osg::Vec4 diffuse, osg::Vec4 ambient, osg::Vec4 specular=osg::Vec4(0.7, 0.7, 0.7, 1), double shininess=42.0)

8.9.1 Detailed Description

Control Room Class, derived from PositionAttitudeTransform, set ups the whole room as its own children.

sets always a light as light0, client should not use this light number any more the chess figures alongside with the provided interactioncallbacks are also set up

Author

Gleb Ostrowski

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 18 of file ControlRoom.h.

8.9.2 Constructor & Destructor Documentation

8.9.2.1 brtr::ControlRoom::ControlRoom (double *roomSize*, int *lod*, brtr::ToonTexSwitcherCallback & *toonCallback*, brtr::ProgramSwitcherCallback & *programCallback*)

Constructor.

Parameters

roomSize	size of the room, height is roomsize/2
lod	level of detail, the higher the more triangles are created
toonCallback	ToonTexSwitcherCallback, will be attached to first chess figure
programCallback	ProgramSwitcherCallback, will be attached to third chess figure

Definition at line 13 of file ControlRoom.cpp.

8.9.2.2 brtr::ControlRoom::~ControlRoom ( ) [inline], [protected]

Definition at line 31 of file ControlRoom.h.

8.9.3 Member Function Documentation

8.9.3.1 ref_ptr< Group > brtr::ControlRoom::createChessFigures (brtr::ToonTexSwitcherCallback & *toonCallback*, brtr::ProgramSwitcherCallback & *programCallback*) [private]

Definition at line 100 of file ControlRoom.cpp.

8.9.3.2 ref_ptr< Material > brtr::ControlRoom::createMaterial (osg::Vec4 *diffuse*, osg::Vec4 *ambient*, osg::Vec4 *specular* = osg::Vec4(0.7,0.7,0.7,1), double *shininess* = 42.0) [private]

Definition at line 145 of file ControlRoom.cpp.

8.9.3.3 ref_ptr< Group > brtr::ControlRoom::createRoomSurrounding (double *roomSize*, int *lod*) [private]

Definition at line 26 of file ControlRoom.cpp.

The documentation for this class was generated from the following files:

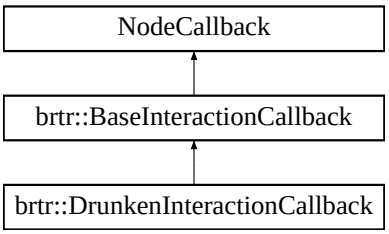
- header/ControlRoom.h
- Objects/ControlRoom.cpp

8.10 brtr::DrunkenInteractionCallback Class Reference

Callback for the drunk effect.

`#include <DrunkenInteractionCallback.h>`

Inheritance diagram for brtr::DrunkenInteractionCallback:



Public Member Functions

- DrunkenInteractionCallback (osg::Node *camera, osg::Camera *hudCam, osg::Switch *geometrySwitch, int width, int height)

Constructor.

- virtual void setText ()

  sets the text on screen. Subclasses must override to set its own (info)text

Protected Member Functions

- virtual void interact (osg::Node *, osg::NodeVisitor *)

  Drunk effect is simulated by changing the FOV of the projection matrix.

Private Attributes

- int _startTime
- osg::ref_ptr< osg::Switch > _geometrySwitch
- osg::ref_ptr < osgAnimation::LinearMotion > _motion
- bool _backwards

Additional Inherited Members

8.10.1 Detailed Description

Callback for the drunk effect.

Author

Gleb Ostrowski

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 15 of file DrunkenInteractionCallback.h.

8.10.2 Constructor & Destructor Documentation

8.10.2.1 brtr::DrunkenInteractionCallback::DrunkenInteractionCallback (osg::Node * *camera*, osg::Camera * *hudCam*, osg::Switch * *geometrySwitch*, int *width*, int *height*)

Constructor.

Parameters

camera	the camera, whichs projection matrix will be manipulated
hudCam	
geometrySwitch	switch containing the bottle, for removing it after the interaction
width	screenwidth
height	screenheight

Definition at line 5 of file DrunkenInteractionCallback.cpp.

8.10.3 Member Function Documentation

8.10.3.1 void brtr::DrunkenInteractionCallback::interact (osg::Node * *node*, osg::NodeVisitor * *nv*) [protected], [virtual]

Drunk effect is simulated by changing the FOV of the projection matrix.

Parameters

not	needed
not	needed

Implements brtr::BaseInteractionCallback.

Definition at line 13 of file DrunkenInteractionCallback.cpp.

8.10.3.2 void brtr::DrunkenInteractionCallback::setText ( ) [virtual]

sets the text on screen. Subclasses must override to set its own (info)text

Implements brtr::BaseInteractionCallback.

Definition at line 44 of file DrunkenInteractionCallback.cpp.

8.10.4 Member Data Documentation

8.10.4.1 bool brtr::DrunkenInteractionCallback::_backwards [private]

Definition at line 42 of file DrunkenInteractionCallback.h.

8.10.4.2 osg::ref_ptr<osg::Switch> brtr::DrunkenInteractionCallback::_geometrySwitch [private]

Definition at line 40 of file DrunkenInteractionCallback.h.

8.10.4.3 osg::ref_ptr<osgAnimation::LinearMotion> brtr::DrunkenInteractionCallback::_motion [private]

Definition at line 41 of file DrunkenInteractionCallback.h.

8.10.4.4 int brtr::DrunkenInteractionCallback::_startTime [private]

Definition at line 39 of file DrunkenInteractionCallback.h.

The documentation for this class was generated from the following files:

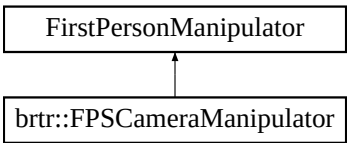
- header/DrunkenInteractionCallback.h
- Callbacks/DrunkenInteractionCallback.cpp

8.11 brtr::FPSCameraManipulator Class Reference

A FPS style CameraManipulator with ground clamping and intersection.

`#include <FPSCameraManipulator.h>`

Inheritance diagram for brtr::FPSCameraManipulator:



Public Member Functions

- FPSCameraManipulator (double movementSpeed, double zHeight, osg::Node *root, bool flightMode=false)

  Constructor.
- double getMovementSpeed () const
- FPSCameraManipulator & setMovementSpeed (double val)
- double getZHeight () const
- FPSCameraManipulator & setZHeight (double val)
- double getJumpHeight () const
- FPSCameraManipulator & setJumpHeight (double val)

Protected Member Functions

- ~FPSCameraManipulator ()
- virtual bool handleMouseMove (const osgGA::GUIEventAdapter &ea, osgGA::GUIActionAdapter &us)

  Handles the movement of the mouse.
- virtual bool handleFrame (const osgGA::GUIEventAdapter &ea, osgGA::GUIActionAdapter &us)

  Handles, what happens every frame.
- virtual bool handleKeyDown (const osgGA::GUIEventAdapter &ea, osgGA::GUIActionAdapter &us)

  Handle key down presses. For supported keys see the class desc.
- virtual bool handleKeyUp (const osgGA::GUIEventAdapter &ea, osgGA::GUIActionAdapter &us)

  Handle key up presses. For supported keys see the class desc.
- virtual bool performMovement ()

  MouseLook is implemented in this method.
- virtual bool performMovementLeftMouseButton (const double eventTimeDelta, const double dx, const double dy)
- virtual bool handleMouseWheel (const osgGA::GUIEventAdapter &ea, osgGA::GUIActionAdapter &us)

Private Member Functions

- bool performEyeMovement ()

  moves the cameraEye, checking various conditions
- bool intersect (const osg::Vec3d start, const osg::Vec3d end, double &distance)

  Finds the distance between start and end intersection, if there is any.
- bool groundIntersection (osg::Vec3d &newEye)

  checks, whether the newEye is still clamped to ground

Private Attributes

- osg::ref_ptr < osg::PositionAttitudeTransform > _body
- bool _flightMode
- bool _forwardMovement
- bool _backwardMovement
- bool _leftMovement
- bool _rightMovement
- bool _upMovement
- bool _downMovement
- bool _attachBody
- bool _shift
- bool _ctrl
- bool _jumpingUp
- bool _jumpingDown
- bool _crouch
- double _maxFallHeight
- double _movementSpeed
- double _zHeight
- double _savedzHeight
- double _intensity
- double _frameFactor
- double _bodyLength
- double _jumpHeight
- double _savedzHeightCrouch

8.11.1 Detailed Description

A FPS style CameraManipulator with ground clamping and intersection.

Controls:

```
  W       = Move forward.
A S D     = (A)=Move Left, (S)=Move backward, (D)=Move Right.
  F       = Toggle FlightMode on/off
Q / E     = Up/Down (in FlightMode)
  G       = Attach/Detach Body
  X       = Crouch
SPACE     = Jump (if not flying)
SHIFT     = Sprint
CTRL      = Walk
```

Inspiration for Intersection and Clamping Testing:
Official OSG Source (mostly DriveManipulator)
GameManipulator and Pod by Viggo Lovli, http://markmail.org/message/e6magjobl7fywbe6 , visited 26/05/2014
For Nodes, which should be passable regardless of FlightMode (e.g. FakeWalls) one must set the NodeMask to ~brtrcollisionMask
Body Code not used anymore (we did not like it). Not deleted, because it works.

Author

Gleb Ostrowski

Version

1.0

Date

06/2014

Precondition

Need to be attached to a viewer, so, create a viewer first

Bug Jumping forward if there is no ground is not working

Copyright

GNU Public License.

Definition at line 33 of file FPSCameraManipulator.h.

8.11.2 Constructor & Destructor Documentation

8.11.2.1 brtr::FPSCameraManipulator::FPSCameraManipulator (double *movementSpeed,* double *zHeight,* osg::Node * *root,* bool *flightMode* = `false`)

Constructor.

Parameters

movementSpeed	the player "movement" speed
zHeight	the camera height
root	root node for attaching the body to, not used anymore
flightMode	flightmoe true or false in the beginning

Definition at line 12 of file FPSCameraManipulator.cpp.

8.11.2.2 brtr::FPSCameraManipulator::~FPSCameraManipulator ( ) `[protected]`

Definition at line 54 of file FPSCameraManipulator.cpp.

8.11.3 Member Function Documentation

8.11.3.1 double brtr::FPSCameraManipulator::getJumpHeight ( ) const

Definition at line 303 of file FPSCameraManipulator.cpp.

8.11.3.2 double brtr::FPSCameraManipulator::getMovementSpeed ( ) const

Definition at line 280 of file FPSCameraManipulator.cpp.

8.11.3.3 double brtr::FPSCameraManipulator::getZHeight ( ) const

Definition at line 289 of file FPSCameraManipulator.cpp.

8.11.3.4 bool brtr::FPSCameraManipulator::groundIntersection (osg::Vec3d & *newEye*) [private]

checks, whether the newEye is still clamped to ground

Performs a LineIntersectionTest from newEye to -Z_AXIS * FPSCameraManipulator::_maxFallHeight if a ground is found (any geometry within _maxFallHeight), the z-Value of newEye will be corrected to be _zHeight above ground. some smoothing is applied for not-so-abrupt jumps

Parameters

newEye	the wannabe new cameraEye Position

Returns

true, if Position is valid, false otherwise

Definition at line 246 of file FPSCameraManipulator.cpp.

8.11.3.5 bool brtr::FPSCameraManipulator::handleFrame (const osgGA::GUIEventAdapter & *ea*, osgGA::GUIActionAdapter & *us*) [protected], [virtual]

Handles, what happens every frame.

Every frame there is a movement check, if the player holds one of the move buttons down the camera moves.

Parameters

ea	the GUIEventAdapter
us	the GUIActionAdapter

Returns

always false otherwise the method would block other frame handle methods

Definition at line 63 of file FPSCameraManipulator.cpp.

8.11.3.6 bool brtr::FPSCameraManipulator::handleKeyDown (const osgGA::GUIEventAdapter & *ea*, osgGA::GUIActionAdapter & *us*) [protected], [virtual]

Handle key down presses. For supported keys see the class desc.

Parameters

ea	
us	

Returns

true, if the keypress is handled, false otherwise

Definition at line 71 of file FPSCameraManipulator.cpp.

8.11.3.7 bool brtr::FPSCameraManipulator::handleKeyUp (const osgGA::GUIEventAdapter & *ea*, osgGA::GUIActionAdapter & *us*) [protected], [virtual]

Handle key up presses. For supported keys see the class desc.

Parameters

ea	ea the GUIEventAdapter
us	us the GUIActionAdapter

Returns

true, if the key-de-press is handled, false otherwise

Definition at line 123 of file FPSCameraManipulator.cpp.

8.11.3.8 bool brtr::FPSCameraManipulator::handleMouseMove (const osgGA::GUIEventAdapter & *ea*, osgGA::GUIActionAdapter & *us*) [protected], [virtual]

Handles the movement of the mouse.

Warp the mouse back to the center, adds the mouse movement to the event stack and calls performMovement(). If it was succesfull, request redraw.

Parameters

ea	the GUIEventAdapter
us	the GUIActionAdapter

Returns

always false otherwise the method would block other mouse movement handle methods

Definition at line 56 of file FPSCameraManipulator.cpp.

8.11.3.9 bool brtr::FPSCameraManipulator::handleMouseWheel (const osgGA::GUIEventAdapter & *ea*, osgGA::GUIActionAdapter & *us*) [protected], [virtual]

Definition at line 311 of file FPSCameraManipulator.cpp.

8.11.3.10 bool brtr::FPSCameraManipulator::intersect (const osg::Vec3d *start*, const osg::Vec3d *end*, double & *distance*) [private]

Finds the distance between start and end intersection, if there is any.

performs a LineIntersectionTest in Model coordinates, from intersect::start to intersect::end the distance to the nearest intersection point, if any exist, is stored in intersect::distance

Parameters

start	the start point of the line
end	the end point of the line
distance	var, which will hold the distance to the nearest intersection point, if any

Returns

true, if there is at least one intersection, false otherwise

Definition at line 263 of file FPSCameraManipulator.cpp.

8.11.3.11 bool brtr::FPSCameraManipulator::performEyeMovement ( ) [private]

moves the cameraEye, checking various conditions

directions + speed depends on the pressed keys. if flightmode is off, clamping to ground is performed, _zHeight is height of eye also, if flightmode is off, LineIntersection from the eye is performed so no walking trough walls again, with flightmode off, jumping request is also handled if flightmode is on, then just moves the cameraEye

Returns

true, if there was any movement, false otherwise

Definition at line 170 of file FPSCameraManipulator.cpp.

8.11.3.12 bool brtr::FPSCameraManipulator::performMovement ( ) [protected], [virtual]

MouseLook is implemented in this method.

Because the mouse is always at the center, it is enough to get the previous mouse position (_ga_t0, the current position is always the center = _ga_t1) and rotate the camera to that position.

Returns

true, if there was any movement, false otherwise

Definition at line 154 of file FPSCameraManipulator.cpp.

8.11.3.13 bool brtr::FPSCameraManipulator::performMovementLeftMouseButton (const double *eventTimeDelta,* const double *dx,* const double *dy*) [protected], [virtual]

Definition at line 307 of file FPSCameraManipulator.cpp.

8.11.3.14 FPSCameraManipulator & brtr::FPSCameraManipulator::setJumpHeight (double *val*)

Definition at line 298 of file FPSCameraManipulator.cpp.

8.11.3.15 FPSCameraManipulator & brtr::FPSCameraManipulator::setMovementSpeed (double *val*)

Definition at line 284 of file FPSCameraManipulator.cpp.

8.11.3.16 FPSCameraManipulator & brtr::FPSCameraManipulator::setZHeight (double *val*)

Definition at line 293 of file FPSCameraManipulator.cpp.

8.11.4 Member Data Documentation

8.11.4.1 bool brtr::FPSCameraManipulator::_attachBody [private]

Definition at line 158 of file FPSCameraManipulator.h.

8.11.4.2 bool brtr::FPSCameraManipulator::_backwardMovement [private]

Definition at line 153 of file FPSCameraManipulator.h.

8.11.4.3 osg::ref_ptr<osg::PositionAttitudeTransform> brtr::FPSCameraManipulator::_body [private]

Definition at line 150 of file FPSCameraManipulator.h.

8.11.4.4 double brtr::FPSCameraManipulator::_bodyLength [private]

Definition at line 170 of file FPSCameraManipulator.h.

8.11.4.5 bool brtr::FPSCameraManipulator::_crouch [private]

Definition at line 163 of file FPSCameraManipulator.h.

8.11.4.6 bool brtr::FPSCameraManipulator::_ctrl [private]

Definition at line 160 of file FPSCameraManipulator.h.

8.11.4.7 bool brtr::FPSCameraManipulator::_downMovement [private]

Definition at line 157 of file FPSCameraManipulator.h.

8.11.4.8 bool brtr::FPSCameraManipulator::_flightMode [private]

Definition at line 151 of file FPSCameraManipulator.h.

8.11.4.9 bool brtr::FPSCameraManipulator::_forwardMovement [private]

Definition at line 152 of file FPSCameraManipulator.h.

8.11.4.10 double brtr::FPSCameraManipulator::_frameFactor [private]

Definition at line 169 of file FPSCameraManipulator.h.

8.11.4.11 double brtr::FPSCameraManipulator::_intensity [private]

Definition at line 168 of file FPSCameraManipulator.h.

8.11.4.12 double brtr::FPSCameraManipulator::_jumpHeight [private]

Definition at line 171 of file FPSCameraManipulator.h.

8.11.4.13 bool brtr::FPSCameraManipulator::_jumpingDown [private]

Definition at line 162 of file FPSCameraManipulator.h.

8.11.4.14 bool brtr::FPSCameraManipulator::_jumpingUp [private]

Definition at line 161 of file FPSCameraManipulator.h.

8.11.4.15 bool brtr::FPSCameraManipulator::_leftMovement [private]

Definition at line 154 of file FPSCameraManipulator.h.

8.11.4.16 double brtr::FPSCameraManipulator::_maxFallHeight [private]

Definition at line 164 of file FPSCameraManipulator.h.

8.11.4.17 double brtr::FPSCameraManipulator::_movementSpeed [private]

Definition at line 165 of file FPSCameraManipulator.h.

8.11.4.18 bool brtr::FPSCameraManipulator::_rightMovement [private]

Definition at line 155 of file FPSCameraManipulator.h.

8.11.4.19 double brtr::FPSCameraManipulator::_savedzHeight [private]

Definition at line 167 of file FPSCameraManipulator.h.

8.11.4.20 double brtr::FPSCameraManipulator::_savedzHeightCrouch [private]

Definition at line 172 of file FPSCameraManipulator.h.

8.11.4.21 bool brtr::FPSCameraManipulator::_shift [private]

Definition at line 159 of file FPSCameraManipulator.h.

8.11.4.22 bool brtr::FPSCameraManipulator::_upMovement [private]

Definition at line 156 of file FPSCameraManipulator.h.

8.11.4.23 double brtr::FPSCameraManipulator::_zHeight [private]

Definition at line 166 of file FPSCameraManipulator.h.

The documentation for this class was generated from the following files:

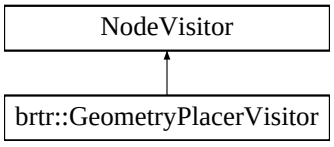
- header/FPSCameraManipulator.h
- Camera/FPSCameraManipulator.cpp

8.12 brtr::GeometryPlacerVisitor Class Reference

NodeVisitor for batch replacing all Geometry in all visited Geodes.

`#include <GeometryPlacerVisitor.h>`

Inheritance diagram for brtr::GeometryPlacerVisitor:



Public Member Functions

- GeometryPlacerVisitor (osg::Geometry *geometryToPlace)

  Constructor.
- virtual void apply (osg::Geode &geode)

  Change the Geometry of this Geode.
- osg::ref_ptr< osg::Geometry > getGeometryToPlace () const
- void setGeometryToPlace (osg::ref_ptr< osg::Geometry > val)

Private Attributes

- osg::ref_ptr< osg::Geometry > _geometryToPlace

8.12.1 Detailed Description

NodeVisitor for batch replacing all Geometry in all visited Geodes.

Takes a geometry as argument and replaces every geometry in the sub scene Useful for batch replacing a bunch of geometrys which were placed as dummys in Blender and then imported. Rotation and Scaling of the Geometry will persist.

Author

Gleb Ostrowski

Version

1.0

Date

2014

Precondition

needs a Node which will accept it. Should have some Geode's for this to work

Copyright

GNU Public License.

Definition at line 15 of file GeometryPlacerVisitor.h.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 brtr::GeometryPlacerVisitor::GeometryPlacerVisitor (osg::Geometry * *geometryToPlace*)

Constructor.

Parameters

geometryToPlace	geometry to replace the found drawables

Returns

Definition at line 6 of file GeometryPlacerVisitor.cpp.

8.12.3 Member Function Documentation

8.12.3.1 void brtr::GeometryPlacerVisitor::apply (osg::Geode & *geode*) [virtual]

Change the Geometry of this Geode.

Parameters

geode	the Geode which will be alternate

Definition at line 11 of file GeometryPlacerVisitor.cpp.

8.12.3.2 osg::ref_ptr< osg::Geometry > brtr::GeometryPlacerVisitor::getGeometryToPlace ( ) const

Definition at line 16 of file GeometryPlacerVisitor.cpp.

8.12.3.3 void brtr::GeometryPlacerVisitor::setGeometryToPlace (osg::ref_ptr< osg::Geometry > *val*)

Definition at line 20 of file GeometryPlacerVisitor.cpp.

8.12.4 Member Data Documentation

8.12.4.1 osg::ref_ptr<osg::Geometry> brtr::GeometryPlacerVisitor::_geometryToPlace [private]

Definition at line 34 of file GeometryPlacerVisitor.h.

The documentation for this class was generated from the following files:

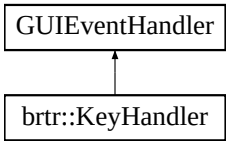
- header/GeometryPlacerVisitor.h
- Util/GeometryPlacerVisitor.cpp

8.13 brtr::KeyHandler Class Reference

Key Handler Class, handles all of our KeyFunctions, which do not belong to camera control (this are handled by FPSCameraManipulator)

`#include <KeyHandler.h>`

Inheritance diagram for brtr::KeyHandler:



Public Member Functions

- KeyHandler (osg::Node ∗, osg::Camera ∗postProcessCam, std::vector< osg::ref_ptr< osg::Program >> programs)

  Constructor.
- virtual bool handle (const osgGA::GUIEventAdapter &ea, osgGA::GUIActionAdapter &aa)

Protected Member Functions

- ∼KeyHandler ()

Private Member Functions

- bool handleKeyDown (const osgGA::GUIEventAdapter &ea, osgGA::GUIActionAdapter &aa)
- void mouseIntersection (osgGA::GUIActionAdapter &aa)

  Checks, if under the mouse (e.a center of screen) is an interact-able object (e.a geometry)
- brtr::BaseInteractionCallback ∗ modifyText (bool show)

  Shows the InteractionMessage on screen, if there is an InteractionObject beneath the mouse (e.a center of screen)

Private Attributes

- osg::ref_ptr< osg::Drawable > _curDrawable
- osg::ref_ptr< osg::Node > _rootNode
- osg::ref_ptr< osg::PolygonMode > _wireFrameMode
- osg::ref_ptr< osg::PolygonMode > _normaleMode
- osg::ref_ptr< osg::Camera > _postProcessCam
- std::vector< osg::ref_ptr < osg::Program > > _programs
- osg::ref_ptr< const osgGA::GUIEventAdapter > _mouseEvent
- bool _isWireFrame
- unsigned int _curProg

8.13.1 Detailed Description

Key Handler Class, handles all of our KeyFunctions, which do not belong to camera control (this are handled by FPSCameraManipulator)

Controls:

```
  C       = Toggle WireFrame Mode On/Off
LClick    = Interact
Shift+1   = Toggle programs
```

Author

Gleb Ostrowski

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 23 of file KeyHandler.h.

8.13.2 Constructor & Destructor Documentation

8.13.2.1 brtr::KeyHandler::KeyHandler (osg::Node * *rootNode*, osg::Camera * *postProcessCam*, std::vector< osg::ref_ptr< osg::Program >> *programs*)

Constructor.

Parameters

rootnode	rootnode of the scene, polygonmode will be activatd on all children
postProcess↩ Cam	node containing the postprocess programs
programs	vector with postprocess programs

Definition at line 8 of file KeyHandler.cpp.

8.13.2.2 brtr::KeyHandler::~KeyHandler ( ) [protected]

Definition at line 19 of file KeyHandler.cpp.

8.13.3 Member Function Documentation

8.13.3.1 bool brtr::KeyHandler::handle (const osgGA::GUIEventAdapter & *ea*, osgGA::GUIActionAdapter & *aa*) [virtual]

Definition at line 21 of file KeyHandler.cpp.

8.13.3.2 bool brtr::KeyHandler::handleKeyDown (const osgGA::GUIEventAdapter & *ea*, osgGA::GUIActionAdapter & *aa*) [private]

Definition at line 53 of file KeyHandler.cpp.

8.13.3.3 brtr::BaseInteractionCallback * brtr::KeyHandler::modifyText (bool *show*) [private]

Shows the InteractionMessage on screen, if there is an InteractionObject beneath the mouse (e.a center of screen)

Parameters

to	show or not show the text, that is the question

Returns

the BaseInteractionCallback associated with the interactionObject, if any, null else;

Definition at line 101 of file KeyHandler.cpp.

8.13.3.4 void brtr::KeyHandler::mouseIntersection (osgGA::GUIActionAdapter & *aa*) [private]

Checks, if under the mouse (e.a center of screen) is an interact-able object (e.a geometry)

Parameters

aa	GUIActionAdapter for getting the camera , to whom the LineIntersectionVisitor will be attached to

Definition at line 76 of file KeyHandler.cpp.

8.13.4 Member Data Documentation

8.13.4.1 osg::ref_ptr<osg::Drawable> brtr::KeyHandler::_curDrawable [private]

Definition at line 53 of file KeyHandler.h.

8.13.4.2 unsigned int brtr::KeyHandler::_curProg [private]

Definition at line 61 of file KeyHandler.h.

8.13.4.3 bool brtr::KeyHandler::_isWireFrame [private]

Definition at line 60 of file KeyHandler.h.

8.13.4.4 osg::ref_ptr< const osgGA::GUIEventAdapter > brtr::KeyHandler::_mouseEvent [private]

Definition at line 59 of file KeyHandler.h.

8.13.4.5 osg::ref_ptr<osg::PolygonMode> brtr::KeyHandler::_normaleMode [private]

Definition at line 56 of file KeyHandler.h.

8.13.4.6 osg::ref_ptr<osg::Camera> brtr::KeyHandler::_postProcessCam [private]

Definition at line 57 of file KeyHandler.h.

8.13.4.7 std::vector<osg::ref_ptr<osg::Program> > brtr::KeyHandler::_programs [private]

Definition at line 58 of file KeyHandler.h.

8.13.4.8 osg::ref_ptr<osg::Node> brtr::KeyHandler::_rootNode [private]

Definition at line 54 of file KeyHandler.h.

8.13.4.9 osg::ref_ptr<osg::PolygonMode> brtr::KeyHandler::_wireFrameMode [private]

Definition at line 55 of file KeyHandler.h.

The documentation for this class was generated from the following files:

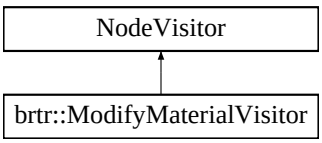
- header/KeyHandler.h
- GUI/KeyHandler.cpp

8.14 brtr::ModifyMaterialVisitor Class Reference

Visitor for altering the material attributes, mainly used for objects craeted with blender.

`#include <ModifyMaterialVisitor.h>`

Inheritance diagram for brtr::ModifyMaterialVisitor:



Public Member Functions

- ModifyMaterialVisitor ()
- void apply (osg::Geode &geode)
- osg::Vec4 getDiffuse () const
- ModifyMaterialVisitor & setDiffuse (osg::Vec4 val)
- osg::Vec4 getSpecular () const
- ModifyMaterialVisitor & setSpecular (osg::Vec4 val)
- osg::Vec4 getAmbient () const
- ModifyMaterialVisitor & setAmbient (osg::Vec4 val)
- double getShininess () const
- ModifyMaterialVisitor & setShininess (double val)

Private Attributes

- osg::Vec4 _diffuse
- osg::Vec4 _specular
- osg::Vec4 _ambient
- double _shininess
- bool _ambientFlag
- bool _specularFlag
- bool _shininessFlag
- bool _diffuseFlag

8.14.1 Detailed Description

Visitor for altering the material attributes, mainly used for objects craeted with blender.

before applying one must set the desired changes (setDiffuse, setAmbient, setSpecular oder setShininess)

Author

Gleb Ostrowski

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 14 of file ModifyMaterialVisitor.h.

8.14.2 Constructor & Destructor Documentation

8.14.2.1 brtr::ModifyMaterialVisitor::ModifyMaterialVisitor ( )

Definition at line 5 of file ModifyMaterialVisitor.cpp.

8.14.3 Member Function Documentation

8.14.3.1 void brtr::ModifyMaterialVisitor::apply (osg::Geode & *geode*)

Definition at line 14 of file ModifyMaterialVisitor.cpp.

8.14.3.2 osg::Vec4 brtr::ModifyMaterialVisitor::getAmbient ( ) const

Definition at line 53 of file ModifyMaterialVisitor.cpp.

8.14.3.3 osg::Vec4 brtr::ModifyMaterialVisitor::getDiffuse ( ) const

Definition at line 33 of file ModifyMaterialVisitor.cpp.

8.14.3.4 double brtr::ModifyMaterialVisitor::getShininess ( ) const

Definition at line 63 of file ModifyMaterialVisitor.cpp.

8.14.3.5 osg::Vec4 brtr::ModifyMaterialVisitor::getSpecular ( ) const

Definition at line 43 of file ModifyMaterialVisitor.cpp.

8.14.3.6 ModifyMaterialVisitor & brtr::ModifyMaterialVisitor::setAmbient (osg::Vec4 *val*)

Definition at line 57 of file ModifyMaterialVisitor.cpp.

8.14.3.7 ModifyMaterialVisitor & brtr::ModifyMaterialVisitor::setDiffuse (osg::Vec4 *val*)

Definition at line 37 of file ModifyMaterialVisitor.cpp.

8.14.3.8 ModifyMaterialVisitor & brtr::ModifyMaterialVisitor::setShininess (double *val*)

Definition at line 67 of file ModifyMaterialVisitor.cpp.

8.14.3.9 ModifyMaterialVisitor & brtr::ModifyMaterialVisitor::setSpecular (osg::Vec4 *val*)

Definition at line 47 of file ModifyMaterialVisitor.cpp.

8.14.4 Member Data Documentation

8.14.4.1 osg::Vec4 brtr::ModifyMaterialVisitor::_ambient [private]

Definition at line 31 of file ModifyMaterialVisitor.h.

8.14.4.2 bool brtr::ModifyMaterialVisitor::_ambientFlag [private]

Definition at line 33 of file ModifyMaterialVisitor.h.

8.14.4.3 osg::Vec4 brtr::ModifyMaterialVisitor::_diffuse [private]

Definition at line 29 of file ModifyMaterialVisitor.h.

8.14.4.4 bool brtr::ModifyMaterialVisitor::_diffuseFlag [private]

Definition at line 36 of file ModifyMaterialVisitor.h.

8.14.4.5 double brtr::ModifyMaterialVisitor::_shininess [private]

Definition at line 32 of file ModifyMaterialVisitor.h.

8.14.4.6 bool brtr::ModifyMaterialVisitor::_shininessFlag [private]

Definition at line 35 of file ModifyMaterialVisitor.h.

8.14.4.7 osg::Vec4 brtr::ModifyMaterialVisitor::_specular [private]

Definition at line 30 of file ModifyMaterialVisitor.h.

8.14.4.8 bool brtr::ModifyMaterialVisitor::_specularFlag [private]

Definition at line 34 of file ModifyMaterialVisitor.h.

The documentation for this class was generated from the following files:

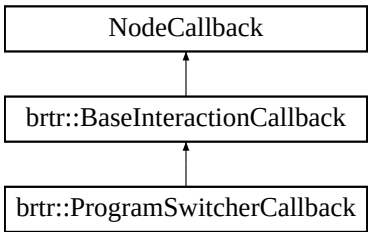
- header/ModifyMaterialVisitor.h
- Util/ModifyMaterialVisitor.cpp

8.15 brtr::ProgramSwitcherCallback Class Reference

Callback for switching the postprocess programs.

`#include <ProgramSwitcherCallback.h>`

Inheritance diagram for brtr::ProgramSwitcherCallback:



Public Member Functions

- ProgramSwitcherCallback (osg::Node ∗postprocessCam, osg::Camera ∗hudCam, int width, int height, std↩ ::vector< osg::ref_ptr< osg::Program >> programs)

  Constructor.
- virtual void setText ()

  sets the text on screen. Subclasses must override to set its own (info)text

Protected Member Functions

- virtual void interact (osg::Node ∗, osg::NodeVisitor ∗)

  each interact sets the next program

Private Attributes

- std::vector< osg::ref_ptr < osg::Program > > _programs
- unsigned int _curProg

Additional Inherited Members

8.15.1 Detailed Description

Callback for switching the postprocess programs.

Every click the next program in the vector is choosen postprocessCam is the node which stateset holds the programs

Author

Gleb Ostrowski

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 17 of file ProgramSwitcherCallback.h.

8.15.2 Constructor & Destructor Documentation

8.15.2.1 brtr::ProgramSwitcherCallback::ProgramSwitcherCallback (osg::Node * *postprocessCam,* osg::Camera * *hudCam,* int *width,* int *height,* std::vector< osg::ref_ptr< osg::Program >> *programs*)

Constructor.

Parameters

postprocessCam	node which stateset contains the programs
hudCam	
width	screenWidth
height	screenHeight
programs	vector with postProcessprograms

Definition at line 8 of file ProgramSwitcherCallback.cpp.

8.15.3 Member Function Documentation

8.15.3.1 void brtr::ProgramSwitcherCallback::interact (osg::Node *, osg::NodeVisitor *) [protected], [virtual]

each interact sets the next program

Parameters

not	used
not	used

Implements brtr::BaseInteractionCallback.

Definition at line 17 of file ProgramSwitcherCallback.cpp.

8.15.3.2 void brtr::ProgramSwitcherCallback::setText ( ) [virtual]

sets the text on screen. Subclasses must override to set its own (info)text

Implements brtr::BaseInteractionCallback.

Definition at line 13 of file ProgramSwitcherCallback.cpp.

8.15.4 Member Data Documentation

8.15.4.1 unsigned int brtr::ProgramSwitcherCallback::_curProg [private]

Definition at line 41 of file ProgramSwitcherCallback.h.

8.15.4.2 std::vector<osg::ref_ptr<osg::Program> > brtr::ProgramSwitcherCallback::_programs [private]

Definition at line 40 of file ProgramSwitcherCallback.h.

The documentation for this class was generated from the following files:

- header/ProgramSwitcherCallback.h
- Callbacks/ProgramSwitcherCallback.cpp

8.16 brtr::RenderingPipeline Struct Reference

struct holding the camera for the multi-rendering passes. Also holds the program vector for the post process pass. pass0Color, pass0depth, passPostProcess, program array, count programArray The program vector is used by the KeyHandler and the InteractionItems for changing the postprocess programs

`#include <UtilFunctions.h>`

Public Attributes

- osg::ref_ptr< osg::Camera > pass_0_color

  Camera for the first pass, renders the ColorBuffer to Texture.
- osg::ref_ptr< osg::Camera > pass_0_depth

  Camera for the first pass, renders the DepthBuffer to Texture.
- osg::ref_ptr< osg::Camera > pass_PostProcess

  PostProcess Camera, uses the texture from the first pass to create various effects.
- std::vector< osg::ref_ptr < osg::Program > > programs

  vector with the avaible postprocess programs

8.16.1 Detailed Description

struct holding the camera for the multi-rendering passes. Also holds the program vector for the post process pass. pass0Color, pass0depth, passPostProcess, program array, count programArray The program vector is used by the KeyHandler and the InteractionItems for changing the postprocess programs

Definition at line 56 of file UtilFunctions.h.

8.16.2 Member Data Documentation

8.16.2.1 osg::ref_ptr<osg::Camera> brtr::RenderingPipeline::pass_0_color

Camera for the first pass, renders the ColorBuffer to Texture.

Definition at line 57 of file UtilFunctions.h.

8.16.2.2 osg::ref_ptr<osg::Camera> brtr::RenderingPipeline::pass_0_depth

Camera for the first pass, renders the DepthBuffer to Texture.

Definition at line 58 of file UtilFunctions.h.

8.16.2.3 osg::ref_ptr<osg::Camera> brtr::RenderingPipeline::pass_PostProcess

PostProcess Camera, uses the texture from the first pass to create various effects.

Definition at line 59 of file UtilFunctions.h.

8.16.2.4 std::vector<osg::ref_ptr<osg::Program> > brtr::RenderingPipeline::programs

vector with the avaible postprocess programs

Definition at line 60 of file UtilFunctions.h.

The documentation for this struct was generated from the following file:

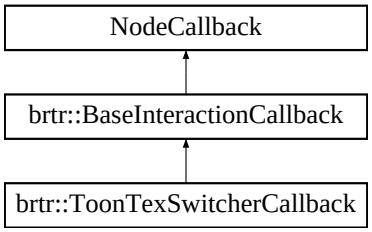
- header/UtilFunctions.h

8.17 brtr::ToonTexSwitcherCallback Class Reference

Callback for switching the ToonTextures.

`#include <ToonTexSwitcherCallback.h>`

Inheritance diagram for brtr::ToonTexSwitcherCallback:



Public Member Functions

- ToonTexSwitcherCallback (osg::Node ∗scenedata, osg::Camera ∗hudCam, int width, int height, std::vector< osg::ref_ptr< osg::Texture2D >> toonTexs)

  Constructor.
- virtual void setText ()

  sets the text on screen. Subclasses must override to set its own (info)text

Protected Member Functions

- virtual void interact (osg::Node ∗node, osg::NodeVisitor ∗)

  each interact sets the next texture

Private Attributes

- int _curTex
- std::vector< osg::ref_ptr < osg::Texture2D > > _toonTexs

Additional Inherited Members

8.17.1 Detailed Description

Callback for switching the ToonTextures.

Every click the next texture in the vector is choosen scenedata is the node which stateset holds the textures

Author

Gleb Ostrowski

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 17 of file ToonTexSwitcherCallback.h.

8.17.2 Constructor & Destructor Documentation

8.17.2.1 brtr::ToonTexSwitcherCallback::ToonTexSwitcherCallback (osg::Node * *scenedata*, osg::Camera * *hudCam*, int *width*, int *height*, std::vector< osg::ref_ptr< osg::Texture2D >> *toonTexs*)

Constructor.

Parameters

scenedata	node which stateset contains the ToonTextures
hudCam	
width	screenWidth
height	screenHeight
toonTexs	vector with ToonTextures

Definition at line 9 of file ToonTexSwitcherCallback.cpp.

8.17.3 Member Function Documentation

8.17.3.1 void brtr::ToonTexSwitcherCallback::interact (osg::Node * *node*, osg::NodeVisitor *) [protected], [virtual]

each interact sets the next texture

Parameters

not	used
not	used

Implements brtr::BaseInteractionCallback.

Definition at line 18 of file ToonTexSwitcherCallback.cpp.

8.17.3.2 void brtr::ToonTexSwitcherCallback::setText ( ) [virtual]

sets the text on screen. Subclasses must override to set its own (info)text

Implements brtr::BaseInteractionCallback.

Definition at line 14 of file ToonTexSwitcherCallback.cpp.

8.17.4 Member Data Documentation

8.17.4.1 int brtr::ToonTexSwitcherCallback::_curTex [private]

Definition at line 40 of file ToonTexSwitcherCallback.h.

8.17.4.2 std::vector<osg::ref_ptr<osg::Texture2D> > brtr::ToonTexSwitcherCallback::_toonTexs [private]

Definition at line 41 of file ToonTexSwitcherCallback.h.

The documentation for this class was generated from the following files:

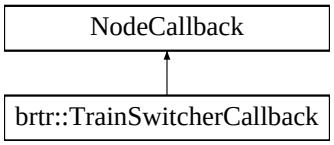
- header/ToonTexSwitcherCallback.h
- Callbacks/ToonTexSwitcherCallback.cpp

8.18 brtr::TrainSwitcherCallback Class Reference

Callback for switching the "trains".

`#include <TrainSwitcherCallback.h>`

Inheritance diagram for brtr::TrainSwitcherCallback:



Public Member Functions

- TrainSwitcherCallback ()
- virtual void operator() (osg::Node ∗node, osg::NodeVisitor ∗nv)

Private Attributes

- int _curActiveTrain
- int _deltaTime

8.18.1 Detailed Description

Callback for switching the "trains".

every ∼36 secs the "train" on the rails switched

Author

Gleb Ostrowski

Version

1.0

Date

2014

Precondition

needs to be attached to a switch node

Copyright

GNU Public License.

Definition at line 15 of file TrainSwitcherCallback.h.

8.18.2 Constructor & Destructor Documentation

8.18.2.1 brtr::TrainSwitcherCallback::TrainSwitcherCallback ( )

Definition at line 4 of file TrainSwitcherCallback.cpp.

8.18.3 Member Function Documentation

8.18.3.1 void brtr::TrainSwitcherCallback::operator() (osg::Node * *node*, osg::NodeVisitor * *nv*) [virtual]

Definition at line 8 of file TrainSwitcherCallback.cpp.

8.18.4 Member Data Documentation

8.18.4.1 int brtr::TrainSwitcherCallback::_curActiveTrain [private]

Definition at line 21 of file TrainSwitcherCallback.h.

8.18.4.2 int brtr::TrainSwitcherCallback::_deltaTime [private]

Definition at line 22 of file TrainSwitcherCallback.h.

The documentation for this class was generated from the following files:

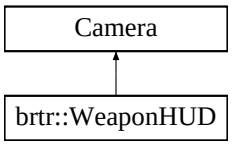
- header/TrainSwitcherCallback.h
- Callbacks/TrainSwitcherCallback.cpp

8.19 brtr::WeaponHUD Class Reference

WeaponHUD class, provides the functions to add a HUD camera to the scene.

`#include <WeaponHUD.h>`

Inheritance diagram for brtr::WeaponHUD:



Classes

- class WeaponSwitchHandler

  EventHandler for WeaponSwitching.

Public Member Functions

- WeaponHUD ()
- WeaponHUD (const WeaponHUD &, const CopyOp &copyop=CopyOp::SHALLOW_COPY)
- ref_ptr< WeaponSwitchHandler > getWeaponHandler ()
- void addPortalGun ()

  a portal gun is added to the weapon switch
- ~WeaponHUD ()

Private Member Functions

- void createWeaponHUD ()

  creates a weapon hud with the default weapon crowbar

Private Attributes

- ref_ptr< Switch > _switcher
- ref_ptr< WeaponSwitchHandler > _handler

8.19.1 Detailed Description

WeaponHUD class, provides the functions to add a HUD camera to the scene.

Use the mouse wheel to shift between weapons after picking up a second one

Author

Jonathan Spielvogel

Version

1.0

Date

2014

Precondition

create a root node and attach the scene to it, then add the HUD to root

Copyright

GNU Public License.

Definition at line 23 of file WeaponHUD.h.

8.19.2 Constructor & Destructor Documentation

8.19.2.1 brtr::WeaponHUD::WeaponHUD ( )

Definition at line 18 of file WeaponHUD.cpp.

8.19.2.2 brtr::WeaponHUD::WeaponHUD (const WeaponHUD & *copy*, const CopyOp & *copyop* = CopyOp::SHALLOW_COPY)

Definition at line 11 of file WeaponHUD.cpp.

8.19.2.3 brtr::WeaponHUD::~WeaponHUD ( )

Definition at line 65 of file WeaponHUD.cpp.

8.19.3 Member Function Documentation

8.19.3.1 void brtr::WeaponHUD::addPortalGun ( )

a portal gun is added to the weapon switch

Definition at line 73 of file WeaponHUD.cpp.

8.19.3.2 void brtr::WeaponHUD::createWeaponHUD ( ) [private]

creates a weapon hud with the default weapon crowbar

Definition at line 22 of file WeaponHUD.cpp.

8.19.3.3 ref_ptr< WeaponHUD::WeaponSwitchHandler > brtr::WeaponHUD::getWeaponHandler ( )

Definition at line 69 of file WeaponHUD.cpp.

8.19.4 Member Data Documentation

8.19.4.1 ref_ptr<WeaponSwitchHandler> brtr::WeaponHUD::_handler [private]

Definition at line 73 of file WeaponHUD.h.

8.19.4.2 ref_ptr<Switch> brtr::WeaponHUD::_switcher [private]

Definition at line 72 of file WeaponHUD.h.

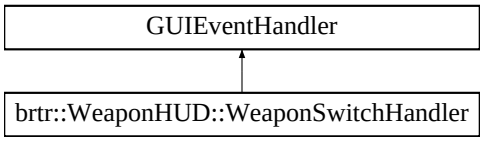
The documentation for this class was generated from the following files:

- header/WeaponHUD.h
- Camera/WeaponHUD.cpp

8.20 brtr::WeaponHUD::WeaponSwitchHandler Class Reference

EventHandler for WeaponSwitching.

Inheritance diagram for brtr::WeaponHUD::WeaponSwitchHandler:



Public Member Functions

- WeaponSwitchHandler (Switch *switchNode)
  initializes a switch handler to switch through weapons
- virtual bool handle (const osgGA::GUIEventAdapter &ea, osgGA::GUIActionAdapter &aa)
  When a mouse event is triggered, this function is called to switch between weapons.

Protected Member Functions

- ~WeaponSwitchHandler ()

Private Attributes

- ref_ptr< Switch > _switch
- int _curWeapon
- unsigned int _frameNumber

8.20.1 Detailed Description

EventHandler for WeaponSwitching.

can only be obtained trough WeaponHUD::getWeaponHandler()

Author

Jonathan Spielvogel

Version

1.0

Date

2014

Copyright

GNU Public License.

Definition at line 32 of file WeaponHUD.h.

8.20.2 Constructor & Destructor Documentation

8.20.2.1 brtr::WeaponHUD::WeaponSwitchHandler::WeaponSwitchHandler (Switch * *switchNode*)

initializes a switch handler to switch through weapons

Parameters

switchNode	pointer to Switch

Returns

WeaponSwitchHandler

Definition at line 101 of file WeaponHUD.cpp.

8.20.2.2 brtr::WeaponHUD::WeaponSwitchHandler::~WeaponSwitchHandler ( ) [inline],[protected]

Definition at line 49 of file WeaponHUD.h.

8.20.3 Member Function Documentation

8.20.3.1 bool brtr::WeaponHUD::WeaponSwitchHandler::handle (const osgGA::GUIEventAdapter & *ea*, osgGA::GUIActionAdapter & *aa*) [virtual]

When a mouse event is triggered, this function is called to switch between weapons.

Parameters

ea	GuiEventAdapter
aa	GuiActionAdapter

Returns

true, if the event was handled, otherwise false

Definition at line 106 of file WeaponHUD.cpp.

8.20.4 Member Data Documentation

8.20.4.1 int brtr::WeaponHUD::WeaponSwitchHandler::_curWeapon [private]

Definition at line 52 of file WeaponHUD.h.

8.20.4.2 unsigned int brtr::WeaponHUD::WeaponSwitchHandler::_frameNumber [private]

Definition at line 53 of file WeaponHUD.h.

8.20.4.3 ref_ptr<Switch> brtr::WeaponHUD::WeaponSwitchHandler::_switch [private]

Definition at line 51 of file WeaponHUD.h.

The documentation for this class was generated from the following files:

- header/WeaponHUD.h
- Camera/WeaponHUD.cpp

Chapter 9

File Documentation

9.1 Animation/AnimationCreater.cpp File Reference

```
#include "../header/AnimationCreater.h"
#include <osgViewer/Viewer>
#include <osg/Geometry>
#include <osgDB/ReadFile>
#include <osg/BlendFunc>
#include <osg/ValueObject>
#include <osgUtil/Optimizer>
#include <osg/AnimationPath>
#include <osg/MatrixTransform>
#include <cmath>
```

9.2 AnimationCreater.cpp

```
00001 #include "../header/AnimationCreater.h"
00002 #include <osgViewer/Viewer>
00003 #include <osg/Geometry>
00004 #include <osgDB/ReadFile>
00005 #include <osg/BlendFunc>
00006 #include <osg/ValueObject>
00007 #include <osgUtil/Optimizer>
00008 #include <osg/AnimationPath>
00009 #include <osg/MatrixTransform>
00010 #include <cmath>
00011 using namespace osg;
00012
00013 /*calculates the angle between two points and return the angle as radian.
00014  *not in use.
00015  */
00016 double AnimationCreator::getAngleRad(Vec3 pointA, Vec3 pointB) {
00017     int kurvenFaktor = 1; //a factor for bigger angles, if necessary.
00018
00019     double dotProd = pointA.x() * pointB.x() + pointA.y() * pointB.y() + pointA.z() * pointB.z();
00020     double lengthA = sqrt(pointA.x() * pointA.x() + pointA.y() * pointA.y() + pointA.z() * pointA.z());
00021     double lengthB = sqrt(pointB.x() * pointB.x() + pointB.y() * pointB.y() + pointB.z() * pointB.z());
00022
00023     //if (skalarProd == 0)
00024      //   return acos(0);
00025
00026     double result = dotProd / (lengthA * lengthB);
00027     return osg::DegreesToRadians(acos(result))*kurvenFaktor *  1;
00028 }
00029
00030 /*
00031
00032 Method construct's the AnimationPath for the Train.
00033 Time = time that the train will take between two points.
00034
00035 loading a file must be looking like this:
00036 osgDB::readNodeFile("..path../Train.ive.0,0,-48.rot");
```

```
00037
00038 */
00039
00040 osg::AnimationPath* AnimationCreator::createAnimationPath(float time)
    {
00041     int vectorCount = 10; //witch points will be used, in this case every 10th point.
00042     osg::ref_ptr<osg::AnimationPath> path = new osg::AnimationPath;
00043     path->setLoopMode(osg::AnimationPath::LOOP);
00044
00045     //array constructed with script.
00046     Vec3 pathArray[] = {
00047         Vec3(-187.90732, 43.37911, -4.13946),
00048         Vec3(-150.33286, -2.83142, -4.13946),
00049         Vec3(-149.98206, -3.25188, -4.13945),
00050         Vec3(-149.78517, -3.09980, -4.13945),
00051         Vec3(-150.13493, -2.68066, -4.13946),
00052         Vec3(-149.61302, -3.70120, -4.13945),
00053         Vec3(-149.41745, -3.54740, -4.13945),
00054         Vec3(-149.21957, -4.18303, -4.13944),
00055         Vec3(-149.02556, -4.02728, -4.13944),
00056         Vec3(-148.79701, -4.69947, -4.13943),
00057         Vec3(-148.60464, -4.54167, -4.13943),
00058         Vec3(-148.34061, -5.25264, -4.13943),
00059         Vec3(-148.14992, -5.09283, -4.13943),
00060         Vec3(-147.84575, -5.84474, -4.13942),
00061         Vec3(-147.65660, -5.68308, -4.13942),
00062         Vec3(-147.30782, -6.47796, -4.13940),
00063         Vec3(-147.12001, -6.31475, -4.13940),
00064         Vec3(-146.72226, -7.15456, -4.13939),
00065         Vec3(-146.53545, -6.99021, -4.13939),
00066         Vec3(-146.72226, -7.15456, -4.13939),
00067         Vec3(-146.08452, -7.87678, -4.13937),
00068         Vec3(-145.89822, -7.71182, -4.13937),
00069         Vec3(-146.53545, -6.99021, -4.13939),
00070         Vec3(-146.08452, -7.87678, -4.13937),
00071         Vec3(-145.45525, -8.47080, -4.13933),
00072         Vec3(-145.27003, -8.30459, -4.13933),
00073         Vec3(-145.89822, -7.71182, -4.13937),
00074         Vec3(-145.45525, -8.47080, -4.13933),
00075         Vec3(-144.95224, -9.02164, -4.13931),
00076         Vec3(-144.76845, -8.85387, -4.13931),
00077         Vec3(-145.27003, -8.30459, -4.13933),
00078         Vec3(-144.44533, -9.56906, -4.13929),
00079         Vec3(-144.26297, -9.39972, -4.13929),
00080         Vec3(-143.93442, -10.11294, -4.13927),
00081         Vec3(-143.75352, -9.94201, -4.13927),
00082         Vec3(-143.41940, -10.65314, -4.13925),
00083         Vec3(-143.24002, -10.48063, -4.13925),
00084         Vec3(-142.90018, -11.18955, -4.13923),
00085         Vec3(-142.72232, -11.01544, -4.13923),
00086         Vec3(-142.37665, -11.72202, -4.13921),
00087         Vec3(-142.20035, -11.54632, -4.13921),
00088         Vec3(-141.84869, -12.25044, -4.13919),
00089         Vec3(-141.67400, -12.07312, -4.13919),
00090         Vec3(-141.31621, -12.77466, -4.13917),
00091         Vec3(-141.14317, -12.59573, -4.13917),
00092         Vec3(-140.77911, -13.29456, -4.13915),
00093         Vec3(-140.60776, -13.11401, -4.13915),
00094         Vec3(-140.23730, -13.80999, -4.13913),
00095         Vec3(-140.06766, -13.62782, -4.13913),
00096         Vec3(-140.23730, -13.80999, -4.13913),
00097         Vec3(-139.69070, -14.32080, -4.13911),
00098         Vec3(-139.52281, -14.13699, -4.13911),
00099         Vec3(-140.06766, -13.62782, -4.13913),
00100         Vec3(-139.69070, -14.32080, -4.13911),
00101         Vec3(-139.13924, -14.82681, -4.13910),
00102         Vec3(-138.97314, -14.64136, -4.13910),
00103         Vec3(-139.52281, -14.13699, -4.13911),
00104         Vec3(-138.58282, -15.32785, -4.13909),
00105         Vec3(-138.41858, -15.14077, -4.13909),
00106         Vec3(-138.02141, -15.82376, -4.13908),
00107         Vec3(-137.85905, -15.63502, -4.13908),
00108         Vec3(-138.02141, -15.82376, -4.13908),
00109         Vec3(-137.45491, -16.31435, -4.13908),
00110         Vec3(-137.29449, -16.12397, -4.13908),
00111         Vec3(-137.85905, -15.63502, -4.13908),
00112         Vec3(-136.88326, -16.79946, -4.13908),
00113         Vec3(-136.72482, -16.60743, -4.13908),
00114         Vec3(-136.88326, -16.79946, -4.13908),
00115         Vec3(-136.30638, -17.27891, -4.13909),
00116         Vec3(-136.14998, -17.08523, -4.13909),
00117         Vec3(-136.72482, -16.60743, -4.13908),
00118         Vec3(-135.72421, -17.75254, -4.13910),
00119         Vec3(-135.56987, -17.55721, -4.13910),
00120         Vec3(-135.72421, -17.75254, -4.13910),
00121         Vec3(-135.13667, -18.22016, -4.13912),
00122         Vec3(-134.98445, -18.02318, -4.13912),
```

```
00123         Vec3(-135.56987, -17.55721, -4.13910),
00124         Vec3(-135.13667, -18.22016, -4.13912),
00125         Vec3(-134.54373, -18.68158, -4.13914),
00126         Vec3(-134.39369, -18.48295, -4.13914),
00127         Vec3(-134.98445, -18.02318, -4.13912),
00128         Vec3(-133.94537, -19.13657, -4.13917),
00129         Vec3(-133.79756, -18.93630, -4.13917),
00130         Vec3(-133.94537, -19.13657, -4.13917),
00131         Vec3(-133.34160, -19.58493, -4.13920),
00132         Vec3(-133.19604, -19.38301, -4.13920),
00133         Vec3(-133.79756, -18.93630, -4.13917),
00134         Vec3(-133.34160, -19.58493, -4.13920),
00135         Vec3(-132.73238, -20.02643, -4.13924),
00136         Vec3(-132.58914, -19.82288, -4.13924),
00137         Vec3(-133.19604, -19.38301, -4.13920),
00138         Vec3(-132.11769, -20.46085, -4.13927),
00139         Vec3(-131.97684, -20.25568, -4.13927),
00140         Vec3(-131.49753, -20.88799, -4.13931),
00141         Vec3(-131.35910, -20.68119, -4.13931),
00142         Vec3(-131.49753, -20.88799, -4.13931),
00143         Vec3(-130.87189, -21.30762, -4.13934),
00144         Vec3(-130.73592, -21.09921, -4.13934),
00145         Vec3(-131.35910, -20.68119, -4.13931),
00146         Vec3(-130.87189, -21.30762, -4.13934),
00147         Vec3(-130.24074, -21.71952, -4.13937),
00148         Vec3(-130.10728, -21.50951, -4.13937),
00149         Vec3(-130.73592, -21.09921, -4.13934),
00150         Vec3(-129.60408, -22.12349, -4.13939),
00151         Vec3(-129.47318, -21.91189, -4.13939),
00152         Vec3(-129.60408, -22.12349, -4.13939),
00153         Vec3(-128.96194, -22.51934, -4.13941),
00154         Vec3(-128.83365, -22.30616, -4.13941),
00155         Vec3(-129.47318, -21.91189, -4.13939),
00156         Vec3(-128.31442, -22.90694, -4.13943),
00157         Vec3(-128.18878, -22.69220, -4.13943),
00158         Vec3(-128.31442, -22.90694, -4.13943),
00159         Vec3(-127.66164, -23.28616, -4.13944),
00160         Vec3(-127.53868, -23.06987, -4.13944),
00161         Vec3(-128.18878, -22.69220, -4.13943),
00162         Vec3(-127.66164, -23.28616, -4.13944),
00163         Vec3(-127.00371, -23.65688, -4.13945),
00164         Vec3(-126.88348, -23.43906, -4.13945),
00165         Vec3(-127.53868, -23.06987, -4.13944),
00166         Vec3(-126.34073, -24.01897, -4.13945),
00167         Vec3(-126.22327, -23.79964, -4.13945),
00168         Vec3(-126.34073, -24.01897, -4.13945),
00169         Vec3(-125.67283, -24.37230, -4.13946),
00170         Vec3(-125.55818, -24.15150, -4.13946),
00171         Vec3(-126.22327, -23.79964, -4.13945),
00172         Vec3(-125.00011, -24.71677, -4.13946),
00173         Vec3(-124.88832, -24.49451, -4.13946),
00174         Vec3(-124.32271, -25.05223, -4.13946),
00175         Vec3(-124.21379, -24.82856, -4.13946),
00176         Vec3(-123.64072, -25.37859, -4.13946),
00177         Vec3(-123.53471, -25.15351, -4.13946),
00178         Vec3(-122.95428, -25.69571, -4.13946),
00179         Vec3(-122.85121, -25.46927, -4.13946),
00180         Vec3(-122.26353, -26.00356, -4.13946),
00181         Vec3(-122.16344, -25.77580, -4.13946),
00182         Vec3(-121.56863, -26.30214, -4.13946),
00183         Vec3(-121.47153, -26.07308, -4.13946),
00184         Vec3(-120.86975, -26.59141, -4.13946),
00185         Vec3(-120.77567, -26.36110, -4.13946),
00186         Vec3(-120.16704, -26.87136, -4.13946),
00187         Vec3(-120.07599, -26.63983, -4.13946),
00188         Vec3(-119.46065, -27.14198, -4.13946),
00189         Vec3(-119.37265, -26.90927, -4.13946),
00190         Vec3(-118.75076, -27.40325, -4.13946),
00191         Vec3(-118.66583, -27.16941, -4.13946),
00192         Vec3(-118.03752, -27.65516, -4.13946),
00193         Vec3(-117.95568, -27.42022, -4.13946),
00194         Vec3(-117.32111, -27.89770, -4.13946),
00195         Vec3(-117.24235, -27.66170, -4.13946),
00196         Vec3(-116.60166, -28.13085, -4.13946),
00197         Vec3(-116.52600, -27.89385, -4.13946),
00198         Vec3(-115.87935, -28.35463, -4.13946),
00199         Vec3(-115.80679, -28.11666, -4.13946),
00200         Vec3(-115.15433, -28.56903, -4.13946),
00201         Vec3(-115.08487, -28.33014, -4.13946),
00202         Vec3(-114.42673, -28.77409, -4.13946),
00203         Vec3(-114.36037, -28.53431, -4.13946),
00204         Vec3(-113.69670, -28.96981, -4.13946),
00205         Vec3(-113.63344, -28.72920, -4.13946),
00206         Vec3(-112.96441, -29.15620, -4.13946),
00207         Vec3(-112.90424, -28.91480, -4.13946),
00208         Vec3(-112.23000, -29.33329, -4.13946),
00209         Vec3(-112.17290, -29.09115, -4.13946),
```

```
00210         Vec3(-111.49360, -29.50111, -4.13946),
00211         Vec3(-111.43956, -29.25826, -4.13946),
00212         Vec3(-110.75537, -29.65965, -4.13946),
00213         Vec3(-110.70438, -29.41615, -4.13946),
00214         Vec3(-110.01547, -29.80896, -4.13946),
00215         Vec3(-109.96751, -29.56484, -4.13946),
00216         Vec3(-109.27403, -29.94905, -4.13946),
00217         Vec3(-109.22908, -29.70436, -4.13946),
00218         Vec3(-108.53119, -30.08001, -4.13946),
00219         Vec3(-108.48923, -29.83479, -4.13946),
00220         Vec3(-107.78709, -30.20189, -4.13946),
00221         Vec3(-107.74808, -29.95618, -4.13946),
00222         Vec3(-107.04185, -30.31478, -4.13946),
00223         Vec3(-107.00578, -30.06862, -4.13946),
00224         Vec3(-106.29562, -30.41875, -4.13946),
00225         Vec3(-106.26244, -30.17219, -4.13946),
00226         Vec3(-105.54852, -30.51389, -4.13946),
00227         Vec3(-105.51822, -30.26695, -4.13946),
00228         Vec3(-104.80069, -30.60025, -4.13946),
00229         Vec3(-104.77322, -30.35299, -4.13946),
00230         Vec3(-104.05226, -30.67794, -4.13946),
00231         Vec3(-104.02759, -30.43037, -4.13946),
00232         Vec3(-103.30337, -30.74701, -4.13946),
00233         Vec3(-103.28145, -30.49919, -4.13946),
00234         Vec3(-102.55413, -30.80756, -4.13946),
00235         Vec3(-102.53493, -30.55951, -4.13946),
00236         Vec3(-101.80469, -30.85971, -4.13946),
00237         Vec3(-101.78815, -30.61148, -4.13946),
00238         Vec3(-101.05508, -30.90365, -4.13946),
00239         Vec3(-101.04119, -30.65525, -4.13946),
00240         Vec3(-100.30545, -30.93952, -4.13946),
00241         Vec3(-100.29414, -30.69099, -4.13946),
00242         Vec3(-99.55585, -30.96750, -4.13946),
00243         Vec3(-99.54707, -30.71887, -4.13946),
00244         Vec3(-98.80640, -30.98776, -4.13946),
00245         Vec3(-98.80012, -30.73905, -4.13946),
00246         Vec3(-98.05717, -31.00046, -4.13946),
00247         Vec3(-98.05334, -30.75170, -4.13946),
00248         Vec3(-97.30827, -31.00576, -4.13946),
00249         Vec3(-97.30684, -30.75698, -4.13946),
00250         Vec3(-96.55977, -31.00385, -4.13946),
00251         Vec3(-96.56069, -30.75506, -4.13946),
00252         Vec3(-95.81177, -30.99488, -4.13946),
00253         Vec3(-95.81499, -30.74611, -4.13946),
00254         Vec3(-95.06435, -30.97901, -4.13946),
00255         Vec3(-95.06982, -30.73028, -4.13946),
00256         Vec3(-94.31756, -30.95643, -4.13946),
00257         Vec3(-94.32523, -30.70776, -4.13946),
00258         Vec3(-93.57141, -30.92729, -4.13946),
00259         Vec3(-93.58124, -30.67870, -4.13946),
00260         Vec3(-92.82597, -30.89177, -4.13946),
00261         Vec3(-92.83791, -30.64327, -4.13946),
00262         Vec3(-92.08128, -30.85004, -4.13946),
00263         Vec3(-92.09526, -30.60165, -4.13946),
00264         Vec3(-91.33737, -30.80227, -4.13946),
00265         Vec3(-91.35335, -30.55399, -4.13946),
00266         Vec3(-90.59427, -30.74861, -4.13946),
00267         Vec3(-90.61221, -30.50047, -4.13946),
00268         Vec3(-89.85204, -30.68924, -4.13946),
00269         Vec3(-89.87189, -30.44125, -4.13946),
00270         Vec3(-89.11073, -30.62434, -4.13946),
00271         Vec3(-89.13242, -30.37650, -4.13946),
00272         Vec3(-88.37035, -30.55405, -4.13946),
00273         Vec3(-88.39384, -30.30638, -4.13946),
00274         Vec3(-87.63091, -30.47854, -4.13946),
00275         Vec3(-87.65617, -30.23104, -4.13946),
00276         Vec3(-86.89246, -30.39795, -4.13946),
00277         Vec3(-86.91942, -30.15062, -4.13946),
00278         Vec3(-86.15496, -30.31242, -4.13946),
00279         Vec3(-86.18358, -30.06529, -4.13946),
00280         Vec3(-85.41846, -30.22211, -4.13946),
00281         Vec3(-85.44870, -29.97517, -4.13946),
00282         Vec3(-84.68295, -30.12716, -4.13946),
00283         Vec3(-84.71474, -29.88041, -4.13946),
00284         Vec3(-83.94843, -30.02770, -4.13946),
00285         Vec3(-83.98175, -29.78115, -4.13946),
00286         Vec3(-83.21492, -29.92390, -4.13946),
00287         Vec3(-83.24971, -29.67756, -4.13946),
00288         Vec3(-82.48243, -29.81588, -4.13946),
00289         Vec3(-82.51865, -29.56974, -4.13946),
00290         Vec3(-81.75096, -29.70380, -4.13946),
00291         Vec3(-81.78856, -29.45787, -4.13946),
00292         Vec3(-81.02052, -29.58780, -4.13946),
00293         Vec3(-81.05946, -29.34208, -4.13946),
00294         Vec3(-80.29109, -29.46799, -4.13946),
00295         Vec3(-80.33132, -29.22248, -4.13946),
00296         Vec3(-79.56268, -29.34452, -4.13946),
```

```
00297         Vec3(-79.60416, -29.09921, -4.13946),
00298         Vec3(-78.83524, -29.21750, -4.13946),
00299         Vec3(-78.87793, -28.97240, -4.13946),
00300         Vec3(-78.10879, -29.08706, -4.13946),
00301         Vec3(-78.15264, -28.84217, -4.13946),
00302         Vec3(-77.38332, -28.95333, -4.13946),
00303         Vec3(-77.42830, -28.70864, -4.13946),
00304         Vec3(-76.65880, -28.81644, -4.13946),
00305         Vec3(-76.70486, -28.57195, -4.13946),
00306         Vec3(-75.93525, -28.67651, -4.13946),
00307         Vec3(-75.98235, -28.43222, -4.13946),
00308         Vec3(-75.21263, -28.53366, -4.13946),
00309         Vec3(-75.26073, -28.28957, -4.13946),
00310         Vec3(-74.49094, -28.38802, -4.13946),
00311         Vec3(-74.54001, -28.14412, -4.13946),
00312         Vec3(-73.77016, -28.23971, -4.13946),
00313         Vec3(-73.82015, -27.99600, -4.13946),
00314         Vec3(-73.05026, -28.08883, -4.13946),
00315         Vec3(-73.10114, -27.84530, -4.13946),
00316         Vec3(-72.33125, -27.93550, -4.13946),
00317         Vec3(-72.38298, -27.69214, -4.13946),
00318         Vec3(-71.61310, -27.77981, -4.13946),
00319         Vec3(-71.66563, -27.53663, -4.13946),
00320         Vec3(-70.89577, -27.62188, -4.13946),
00321         Vec3(-70.94907, -27.37887, -4.13946),
00322         Vec3(-70.17924, -27.46183, -4.13946),
00323         Vec3(-70.23329, -27.21898, -4.13946),
00324         Vec3(-69.46352, -27.29975, -4.13946),
00325         Vec3(-69.51826, -27.05706, -4.13946),
00326         Vec3(-68.74855, -27.13575, -4.13946),
00327         Vec3(-68.80396, -26.89322, -4.13946),
00328         Vec3(-68.03434, -26.96996, -4.13946),
00329         Vec3(-68.09039, -26.72756, -4.13946),
00330         Vec3(-67.32086, -26.80245, -4.13946),
00331         Vec3(-67.37750, -26.56020, -4.13946),
00332         Vec3(-66.60809, -26.63334, -4.13946),
00333         Vec3(-66.66530, -26.39121, -4.13946),
00334         Vec3(-65.89600, -26.46270, -4.13946),
00335         Vec3(-65.95374, -26.22071, -4.13946),
00336         Vec3(-65.18456, -26.29065, -4.13946),
00337         Vec3(-65.24280, -26.04878, -4.13946),
00338         Vec3(-64.47375, -26.11728, -4.13946),
00339         Vec3(-64.53246, -25.87551, -4.13946),
00340         Vec3(-63.76357, -25.94267, -4.13946),
00341         Vec3(-63.82272, -25.70101, -4.13946),
00342         Vec3(-63.05398, -25.76692, -4.13946),
00343         Vec3(-63.11353, -25.52536, -4.13946),
00344         Vec3(-62.34494, -25.59013, -4.13946),
00345         Vec3(-62.40486, -25.34866, -4.13946),
00346         Vec3(-61.63643, -25.41239, -4.13946),
00347         Vec3(-61.69669, -25.17101, -4.13946),
00348         Vec3(-60.92844, -25.23379, -4.13946),
00349         Vec3(-60.98901, -24.99249, -4.13946),
00350         Vec3(-60.22095, -25.05441, -4.13946),
00351         Vec3(-60.28181, -24.81318, -4.13946),
00352         Vec3(-59.51392, -24.87435, -4.13946),
00353         Vec3(-59.57503, -24.63318, -4.13946),
00354         Vec3(-58.80734, -24.69367, -4.13946),
00355         Vec3(-58.86867, -24.45256, -4.13946),
00356         Vec3(-58.10118, -24.51248, -4.13946),
00357         Vec3(-58.16270, -24.27142, -4.13946),
00358         Vec3(-57.39542, -24.33084, -4.13946),
00359         Vec3(-57.45712, -24.08982, -4.13946),
00360         Vec3(-56.69004, -24.14885, -4.13946),
00361         Vec3(-56.75186, -23.90786, -4.13946),
00362         Vec3(-55.98500, -23.96658, -4.13946),
00363         Vec3(-56.04694, -23.72562, -4.13946),
00364         Vec3(-55.28028, -23.78412, -4.13946),
00365         Vec3(-55.34230, -23.54319, -4.13946),
00366         Vec3(-54.57589, -23.60155, -4.13946),
00367         Vec3(-54.63795, -23.36063, -4.13946),
00368         Vec3(-53.87176, -23.41895, -4.13946),
00369         Vec3(-53.93385, -23.17804, -4.13946),
00370         Vec3(-53.16788, -23.23639, -4.13946),
00371         Vec3(-53.22997, -22.99548, -4.13946),
00372         Vec3(-52.46425, -23.05395, -4.13946),
00373         Vec3(-52.52631, -22.81303, -4.13946),
00374         Vec3(-51.76084, -22.87171, -4.13946),
00375         Vec3(-51.82285, -22.63077, -4.13946),
00376         Vec3(-51.05760, -22.68973, -4.13946),
00377         Vec3(-51.11954, -22.44877, -4.13946),
00378         Vec3(-50.35458, -22.50809, -4.13946),
00379         Vec3(-50.41640, -22.26711, -4.13946),
00380         Vec3(-49.65166, -22.32687, -4.13946),
00381         Vec3(-49.71337, -22.08586, -4.13946),
00382         Vec3(-48.94891, -22.14615, -4.13946),
00383         Vec3(-49.01045, -21.90509, -4.13946),
```

```
00384         Vec3(-48.24623, -21.96599, -4.13946),
00385         Vec3(-48.30759, -21.72489, -4.13946),
00386         Vec3(-47.54365, -21.78647, -4.13946),
00387         Vec3(-47.60481, -21.54531, -4.13946),
00388         Vec3(-46.84115, -21.60765, -4.13946),
00389         Vec3(-46.90208, -21.36644, -4.13946),
00390         Vec3(-46.13870, -21.42962, -4.13946),
00391         Vec3(-46.19938, -21.18834, -4.13946),
00392         Vec3(-45.43626, -21.25243, -4.13946),
00393         Vec3(-45.49667, -21.01108, -4.13946),
00394         Vec3(-44.73387, -21.07615, -4.13946),
00395         Vec3(-44.79397, -20.83472, -4.13946),
00396         Vec3(-44.03144, -20.90084, -4.13946),
00397         Vec3(-44.09122, -20.65934, -4.13946),
00398         Vec3(-43.32903, -20.72659, -4.13946),
00399         Vec3(-43.38844, -20.48500, -4.13946),
00400         Vec3(-42.62653, -20.55345, -4.13946),
00401         Vec3(-42.68559, -20.31178, -4.13946),
00402         Vec3(-41.92402, -20.38150, -4.13946),
00403         Vec3(-41.98267, -20.13973, -4.13946),
00404         Vec3(-41.22142, -20.21080, -4.13946),
00405         Vec3(-41.27966, -19.96892, -4.13946),
00406         Vec3(-40.51873, -20.04142, -4.13946),
00407         Vec3(-40.57652, -19.79943, -4.13946),
00408         Vec3(-39.81596, -19.87341, -4.13946),
00409         Vec3(-39.87328, -19.63132, -4.13946),
00410         Vec3(-39.11304, -19.70685, -4.13946),
00411         Vec3(-39.16988, -19.46464, -4.13946),
00412         Vec3(-38.41002, -19.54178, -4.13946),
00413         Vec3(-38.46635, -19.29946, -4.13946),
00414         Vec3(-37.70686, -19.37829, -4.13946),
00415         Vec3(-37.76265, -19.13584, -4.13946),
00416         Vec3(-37.00352, -19.21642, -4.13946),
00417         Vec3(-37.05876, -18.97384, -4.13946),
00418         Vec3(-36.30002, -19.05623, -4.13946),
00419         Vec3(-36.35469, -18.81353, -4.13946),
00420         Vec3(-35.59637, -18.89780, -4.13946),
00421         Vec3(-35.65044, -18.65496, -4.13946),
00422         Vec3(-34.89249, -18.74119, -4.13946),
00423         Vec3(-34.94594, -18.49821, -4.13946),
00424         Vec3(-34.18842, -18.58644, -4.13946),
00425         Vec3(-34.24123, -18.34333, -4.13946),
00426         Vec3(-33.48412, -18.43363, -4.13946),
00427         Vec3(-33.53626, -18.19037, -4.13946),
00428         Vec3(-32.77962, -18.28280, -4.13946),
00429         Vec3(-32.83109, -18.03940, -4.13946),
00430         Vec3(-32.07490, -18.13402, -4.13946),
00431         Vec3(-32.12566, -17.89046, -4.13946),
00432         Vec3(-31.36991, -17.98733, -4.13946),
00433         Vec3(-31.41996, -17.74363, -4.13946),
00434         Vec3(-30.66471, -17.84280, -4.13946),
00435         Vec3(-30.71401, -17.59894, -4.13946),
00436         Vec3(-29.95924, -17.70047, -4.13946),
00437         Vec3(-30.00778, -17.45647, -4.13946),
00438         Vec3(-29.25352, -17.56042, -4.13946),
00439         Vec3(-29.30128, -17.31626, -4.13946),
00440         Vec3(-28.54752, -17.42268, -4.13946),
00441         Vec3(-28.59448, -17.17837, -4.13946),
00442         Vec3(-27.84126, -17.28733, -4.13946),
00443         Vec3(-27.88741, -17.04285, -4.13946),
00444         Vec3(-27.13472, -17.15440, -4.13946),
00445         Vec3(-27.18002, -16.90977, -4.13946),
00446         Vec3(-26.42790, -17.02394, -4.13946),
00447         Vec3(-26.47234, -16.77916, -4.13946),
00448         Vec3(-25.72080, -16.89602, -4.13946),
00449         Vec3(-25.76437, -16.65107, -4.13946),
00450         Vec3(-25.01343, -16.77067, -4.13946),
00451         Vec3(-25.05610, -16.52557, -4.13946),
00452         Vec3(-24.30576, -16.64793, -4.13946),
00453         Vec3(-24.34752, -16.40268, -4.13946),
00454         Vec3(-23.59780, -16.52788, -4.13946),
00455         Vec3(-23.63863, -16.28246, -4.13946),
00456         Vec3(-22.88956, -16.41054, -4.13946),
00457         Vec3(-22.92945, -16.16497, -4.13946),
00458         Vec3(-22.18101, -16.29597, -4.13946),
00459         Vec3(-22.21993, -16.05025, -4.13946),
00460         Vec3(-21.47219, -16.18422, -4.13946),
00461         Vec3(-21.51013, -15.93834, -4.13946),
00462         Vec3(-20.76305, -16.07533, -4.13946),
00463         Vec3(-20.80001, -15.82930, -4.13946),
00464         Vec3(-20.05367, -15.96935, -4.13946),
00465         Vec3(-20.08961, -15.72317, -4.13946),
00466         Vec3(-19.34396, -15.86630, -4.13946),
00467         Vec3(-19.37888, -15.61998, -4.13946),
00468         Vec3(-18.63399, -15.76624, -4.13946),
00469         Vec3(-18.66786, -15.51977, -4.13946),
00470         Vec3(-17.92374, -15.66920, -4.13946),
```

```
00471         Vec3(-17.95656, -15.42259, -4.13946),
00472         Vec3(-17.21323, -15.57522, -4.13946),
00473         Vec3(-17.24498, -15.32846, -4.13946),
00474         Vec3(-16.50244, -15.48434, -4.13946),
00475         Vec3(-16.53310, -15.23745, -4.13946),
00476         Vec3(-15.79137, -15.39660, -4.13946),
00477         Vec3(-15.82092, -15.14958, -4.13946),
00478         Vec3(-15.08005, -15.31205, -4.13946),
00479         Vec3(-15.10849, -15.06489, -4.13946),
00480         Vec3(-14.36845, -15.23071, -4.13946),
00481         Vec3(-14.39578, -14.98343, -4.13946),
00482         Vec3(-13.65663, -15.15262, -4.13946),
00483         Vec3(-13.68283, -14.90521, -4.13946),
00484         Vec3(-12.94453, -15.07780, -4.13946),
00485         Vec3(-12.96957, -14.83028, -4.13946),
00486         Vec3(-12.23227, -15.00630, -4.13946),
00487         Vec3(-12.25615, -14.75866, -4.13946),
00488         Vec3(-11.51973, -14.93812, -4.13946),
00489         Vec3(-11.54243, -14.69037, -4.13946),
00490         Vec3(-10.80698, -14.87331, -4.13946),
00491         Vec3(-10.82849, -14.62545, -4.13946),
00492         Vec3(-10.09402, -14.81189, -4.13946),
00493         Vec3(-10.11435, -14.56393, -4.13946),
00494         Vec3(-9.38087, -14.75389, -4.13946),
00495         Vec3(-9.40001, -14.50584, -4.13946),
00496         Vec3(-8.66753, -14.69934, -4.13946),
00497         Vec3(-8.68546, -14.45120, -4.13946),
00498         Vec3(-7.95398, -14.64827, -4.13946),
00499         Vec3(-7.97069, -14.40004, -4.13946),
00500         Vec3(-7.24028, -14.60071, -4.13946),
00501         Vec3(-7.25577, -14.35240, -4.13946),
00502         Vec3(-6.52640, -14.55670, -4.13946),
00503         Vec3(-6.54065, -14.30832, -4.13946),
00504         Vec3(-5.81241, -14.51627, -4.13946),
00505         Vec3(-5.82541, -14.26782, -4.13946),
00506         Vec3(-5.09825, -14.47946, -4.13946),
00507         Vec3(-5.11002, -14.23095, -4.13946),
00508         Vec3(-4.38396, -14.44630, -4.13946),
00509         Vec3(-4.39449, -14.19774, -4.13946),
00510         Vec3(-3.66957, -14.41683, -4.13946),
00511         Vec3(-3.67883, -14.16822, -4.13946),
00512         Vec3(-2.95511, -14.39108, -4.13946),
00513         Vec3(-2.96310, -14.14243, -4.13946),
00514         Vec3(-2.24052, -14.36910, -4.13946),
00515         Vec3(-2.24725, -14.12040, -4.13946),
00516         Vec3(-1.52588, -14.35090, -4.13946),
00517         Vec3(-1.53134, -14.10217, -4.13946),
00518         Vec3(-0.81116, -14.33652, -4.13946),
00519         Vec3(-0.81535, -14.08777, -4.13946),
00520         Vec3(-0.09637, -14.32595, -4.13946),
00521         Vec3(-0.09930, -14.07718, -4.13946),
00522         Vec3(0.61838, -14.31919, -4.13946),
00523         Vec3(0.61673, -14.07041, -4.13946),
00524         Vec3(1.33321, -14.31624, -4.13946),
00525         Vec3(1.33284, -14.06745, -4.13946),
00526         Vec3(2.04802, -14.31710, -4.13946),
00527         Vec3(2.04893, -14.06831, -4.13946),
00528         Vec3(2.76282, -14.32176, -4.13946),
00529         Vec3(2.76500, -14.07298, -4.13946),
00530         Vec3(3.47762, -14.33022, -4.13946),
00531         Vec3(3.48108, -14.08146, -4.13946),
00532         Vec3(4.19237, -14.34248, -4.13946),
00533         Vec3(4.19710, -14.09374, -4.13946),
00534         Vec3(4.90704, -14.35855, -4.13946),
00535         Vec3(4.91304, -14.10983, -4.13946),
00536         Vec3(5.62167, -14.37840, -4.13946),
00537         Vec3(5.62894, -14.12972, -4.13946),
00538         Vec3(6.33621, -14.40203, -4.13946),
00539         Vec3(6.34474, -14.15338, -4.13946),
00540         Vec3(7.05064, -14.42938, -4.13946),
00541         Vec3(7.06044, -14.18078, -4.13946),
00542         Vec3(7.76500, -14.46043, -4.13946),
00543         Vec3(7.77605, -14.21189, -4.13946),
00544         Vec3(8.47920, -14.49515, -4.13946),
00545         Vec3(8.49150, -14.24666, -4.13946),
00546         Vec3(9.19331, -14.53350, -4.13946),
00547         Vec3(9.20685, -14.28508, -4.13946),
00548         Vec3(9.90724, -14.57544, -4.13946),
00549         Vec3(9.92203, -14.32709, -4.13946),
00550         Vec3(10.62106, -14.62095, -4.13946),
00551         Vec3(10.63707, -14.37268, -4.13946),
00552         Vec3(11.33470, -14.66999, -4.13946),
00553         Vec3(11.35193, -14.42180, -4.13946),
00554         Vec3(12.04814, -14.72253, -4.13946),
00555         Vec3(12.06659, -14.47442, -4.13946),
00556         Vec3(12.76141, -14.77853, -4.13946),
00557         Vec3(12.78107, -14.53052, -4.13946),
```

```
00558         Vec3(13.47447, -14.83797, -4.13946),
00559         Vec3(13.49533, -14.59006, -4.13946),
00560         Vec3(14.18736, -14.90083, -4.13946),
00561         Vec3(14.20940, -14.65302, -4.13946),
00562         Vec3(14.90002, -14.96706, -4.13946),
00563         Vec3(14.92323, -14.71936, -4.13946),
00564         Vec3(15.61247, -15.03664, -4.13946),
00565         Vec3(15.63684, -14.78905, -4.13946),
00566         Vec3(16.32466, -15.10954, -4.13946),
00567         Vec3(16.35019, -14.86207, -4.13946),
00568         Vec3(17.03662, -15.18574, -4.13946),
00569         Vec3(17.06329, -14.93839, -4.13946),
00570         Vec3(17.74837, -15.26520, -4.13946),
00571         Vec3(17.77617, -15.01797, -4.13946),
00572         Vec3(18.45984, -15.34790, -4.13946),
00573         Vec3(18.48877, -15.10080, -4.13946),
00574         Vec3(19.17107, -15.43380, -4.13946),
00575         Vec3(19.20110, -15.18683, -4.13946),
00576         Vec3(19.88197, -15.52286, -4.13946),
00577         Vec3(19.91310, -15.27603, -4.13946),
00578         Vec3(20.59264, -15.61505, -4.13946),
00579         Vec3(20.62483, -15.36835, -4.13946),
00580         Vec3(21.30309, -15.71033, -4.13946),
00581         Vec3(21.33635, -15.46377, -4.13946),
00582         Vec3(22.01320, -15.80864, -4.13946),
00583         Vec3(22.04752, -15.56223, -4.13946),
00584         Vec3(22.72304, -15.90996, -4.13946),
00585         Vec3(22.75838, -15.66370, -4.13946),
00586         Vec3(23.43263, -16.01425, -4.13946),
00587         Vec3(23.46899, -15.76814, -4.13946),
00588         Vec3(24.14192, -16.12147, -4.13946),
00589         Vec3(24.17929, -15.87550, -4.13946),
00590         Vec3(24.85092, -16.23156, -4.13946),
00591         Vec3(24.88928, -15.98575, -4.13946),
00592         Vec3(25.55963, -16.34451, -4.13946),
00593         Vec3(25.59897, -16.09885, -4.13946),
00594         Vec3(26.26802, -16.46025, -4.13946),
00595         Vec3(26.30830, -16.21474, -4.13946),
00596         Vec3(26.97615, -16.57874, -4.13946),
00597         Vec3(27.01738, -16.33339, -4.13946),
00598         Vec3(27.68399, -16.69993, -4.13946),
00599         Vec3(27.72614, -16.45474, -4.13946),
00600         Vec3(28.39153, -16.82377, -4.13946),
00601         Vec3(28.43459, -16.57874, -4.13946),
00602         Vec3(29.09879, -16.95021, -4.13946),
00603         Vec3(29.14273, -16.70534, -4.13946),
00604         Vec3(29.80577, -17.07921, -4.13946),
00605         Vec3(29.85057, -16.83449, -4.13946),
00606         Vec3(30.51248, -17.21071, -4.13946),
00607         Vec3(30.55814, -16.96615, -4.13946),
00608         Vec3(31.21890, -17.34467, -4.13946),
00609         Vec3(31.26540, -17.10027, -4.13946),
00610         Vec3(31.92505, -17.48104, -4.13946),
00611         Vec3(31.97235, -17.23679, -4.13946),
00612         Vec3(32.63094, -17.61976, -4.13946),
00613         Vec3(32.67903, -17.37566, -4.13946),
00614         Vec3(33.33655, -17.76078, -4.13946),
00615         Vec3(33.38542, -17.51684, -4.13946),
00616         Vec3(34.04190, -17.90404, -4.13946),
00617         Vec3(34.09152, -17.66025, -4.13946),
00618         Vec3(34.74701, -18.04949, -4.13946),
00619         Vec3(34.79736, -17.80585, -4.13946),
00620         Vec3(35.45187, -18.19707, -4.13946),
00621         Vec3(35.50294, -17.95358, -4.13946),
00622         Vec3(36.15651, -18.34673, -4.13946),
00623         Vec3(36.20828, -18.10338, -4.13946),
00624         Vec3(36.86092, -18.49840, -4.13946),
00625         Vec3(36.91335, -18.25520, -4.13946),
00626         Vec3(37.56512, -18.65203, -4.13946),
00627         Vec3(37.61821, -18.40898, -4.13946),
00628         Vec3(38.26910, -18.80758, -4.13946),
00629         Vec3(38.32281, -18.56466, -4.13946),
00630         Vec3(38.97290, -18.96497, -4.13946),
00631         Vec3(39.02722, -18.72218, -4.13946),
00632         Vec3(39.67650, -19.12415, -4.13946),
00633         Vec3(39.73140, -18.88149, -4.13946),
00634         Vec3(40.37990, -19.28505, -4.13946),
00635         Vec3(40.43538, -19.04252, -4.13946),
00636         Vec3(41.08316, -19.44762, -4.13946),
00637         Vec3(41.13918, -19.20522, -4.13946),
00638         Vec3(41.78629, -19.61178, -4.13946),
00639         Vec3(41.84283, -19.36951, -4.13946),
00640         Vec3(42.48924, -19.77749, -4.13946),
00641         Vec3(42.54630, -19.53533, -4.13946),
00642         Vec3(43.19211, -19.94468, -4.13946),
00643         Vec3(43.24963, -19.70263, -4.13946),
00644         Vec3(43.89485, -20.11328, -4.13946),
```

```
00645         Vec3(43.95284, -19.87134, -4.13946),
00646         Vec3(44.59747, -20.28322, -4.13946),
00647         Vec3(44.65590, -20.04139, -4.13946),
00648         Vec3(45.30005, -20.45446, -4.13946),
00649         Vec3(45.35887, -20.21273, -4.13946),
00650         Vec3(46.00253, -20.62692, -4.13946),
00651         Vec3(46.06175, -20.38528, -4.13946),
00652         Vec3(46.70499, -20.80054, -4.13946),
00653         Vec3(46.76457, -20.55899, -4.13946),
00654         Vec3(47.40742, -20.97525, -4.13946),
00655         Vec3(47.46735, -20.73378, -4.13946),
00656         Vec3(48.10983, -21.15098, -4.13946),
00657         Vec3(48.17006, -20.90959, -4.13946),
00658         Vec3(48.81224, -21.32766, -4.13946),
00659         Vec3(48.87277, -21.08634, -4.13946),
00660         Vec3(49.51468, -21.50522, -4.13946),
00661         Vec3(49.57547, -21.26398, -4.13946),
00662         Vec3(50.21713, -21.68360, -4.13946),
00663         Vec3(50.27817, -21.44241, -4.13946),
00664         Vec3(50.91968, -21.86273, -4.13946),
00665         Vec3(50.98094, -21.62160, -4.13946),
00666         Vec3(51.62228, -22.04253, -4.13946),
00667         Vec3(51.68373, -21.80145, -4.13946),
00668         Vec3(52.32498, -22.22294, -4.13946),
00669         Vec3(52.38661, -21.98190, -4.13946),
00670         Vec3(53.02780, -22.40389, -4.13946),
00671         Vec3(53.08957, -22.16289, -4.13946),
00672         Vec3(53.73076, -22.58530, -4.13946),
00673         Vec3(53.79265, -22.34433, -4.13946),
00674         Vec3(54.43390, -22.76710, -4.13946),
00675         Vec3(54.49586, -22.52615, -4.13946),
00676         Vec3(55.13719, -22.94920, -4.13946),
00677         Vec3(55.19923, -22.70827, -4.13946),
00678         Vec3(55.84070, -23.13154, -4.13946),
00679         Vec3(55.90279, -22.89063, -4.13946),
00680         Vec3(56.54442, -23.31405, -4.13946),
00681         Vec3(56.60652, -23.07314, -4.13946),
00682         Vec3(57.24840, -23.49664, -4.13946),
00683         Vec3(57.31049, -23.25572, -4.13946),
00684         Vec3(57.95265, -23.67924, -4.13946),
00685         Vec3(58.01471, -23.43832, -4.13946),
00686         Vec3(58.65720, -23.86178, -4.13946),
00687         Vec3(58.71918, -23.62084, -4.13946),
00688         Vec3(59.36201, -24.04418, -4.13946),
00689         Vec3(59.42390, -23.80321, -4.13946),
00690         Vec3(60.06720, -24.22635, -4.13946),
00691         Vec3(60.12897, -23.98535, -4.13946),
00692         Vec3(60.77274, -24.40820, -4.13946),
00693         Vec3(60.83435, -24.16717, -4.13946),
00694         Vec3(61.47868, -24.58967, -4.13946),
00695         Vec3(61.54012, -24.34859, -4.13946),
00696         Vec3(62.18503, -24.77066, -4.13946),
00697         Vec3(62.24626, -24.52953, -4.13946),
00698         Vec3(62.89177, -24.95110, -4.13946),
00699         Vec3(62.95277, -24.70990, -4.13946),
00700         Vec3(63.59897, -25.13088, -4.13946),
00701         Vec3(63.65970, -24.88962, -4.13946),
00702         Vec3(64.30667, -25.30995, -4.13946),
00703         Vec3(64.36713, -25.06861, -4.13946),
00704         Vec3(65.01488, -25.48820, -4.13946),
00705         Vec3(65.07500, -25.24679, -4.13946),
00706         Vec3(65.72362, -25.66556, -4.13946),
00707         Vec3(65.78339, -25.42406, -4.13946),
00708         Vec3(66.43286, -25.84193, -4.13946),
00709         Vec3(66.49225, -25.60033, -4.13946),
00710         Vec3(67.14270, -26.01722, -4.13946),
00711         Vec3(67.20168, -25.77552, -4.13946),
00712         Vec3(67.85315, -26.19133, -4.13946),
00713         Vec3(67.91167, -25.94952, -4.13946),
00714         Vec3(68.56421, -26.36416, -4.13946),
00715         Vec3(68.62225, -26.12224, -4.13946),
00716         Vec3(69.27592, -26.53563, -4.13946),
00717         Vec3(69.33345, -26.29359, -4.13946),
00718         Vec3(69.98830, -26.70564, -4.13946),
00719         Vec3(70.04527, -26.46347, -4.13946),
00720         Vec3(70.70135, -26.87410, -4.13946),
00721         Vec3(70.75775, -26.63178, -4.13946),
00722         Vec3(71.41512, -27.04090, -4.13946),
00723         Vec3(71.47090, -26.79844, -4.13946),
00724         Vec3(72.12965, -27.20596, -4.13946),
00725         Vec3(72.18480, -26.96336, -4.13946),
00726         Vec3(72.84492, -27.36916, -4.13946),
00727         Vec3(72.89938, -27.12641, -4.13946),
00728         Vec3(73.56100, -27.53041, -4.13946),
00729         Vec3(73.61475, -27.28750, -4.13946),
00730         Vec3(74.27786, -27.68959, -4.13946),
00731         Vec3(74.33084, -27.44651, -4.13946),
```

```
00732         Vec3(74.99554, -27.84660, -4.13946),
00733         Vec3(75.04773, -27.60334, -4.13946),
00734         Vec3(75.71405, -28.00131, -4.13946),
00735         Vec3(75.76543, -27.75788, -4.13946),
00736         Vec3(76.43340, -28.15363, -4.13946),
00737         Vec3(76.48390, -27.91002, -4.13946),
00738         Vec3(77.15372, -28.30346, -4.13946),
00739         Vec3(77.20332, -28.05966, -4.13946),
00740         Vec3(77.87486, -28.45066, -4.13946),
00741         Vec3(77.92352, -28.20668, -4.13946),
00742         Vec3(78.59692, -28.59514, -4.13946),
00743         Vec3(78.64461, -28.35096, -4.13946),
00744         Vec3(79.31995, -28.73678, -4.13946),
00745         Vec3(79.36661, -28.49240, -4.13946),
00746         Vec3(80.04388, -28.87545, -4.13946),
00747         Vec3(80.08949, -28.63088, -4.13946),
00748         Vec3(80.76883, -29.01104, -4.13946),
00749         Vec3(80.81334, -28.76627, -4.13946),
00750         Vec3(81.49471, -29.14341, -4.13946),
00751         Vec3(81.53809, -28.89843, -4.13946),
00752         Vec3(82.22157, -29.27243, -4.13946),
00753         Vec3(82.26375, -29.02724, -4.13946),
00754         Vec3(82.94943, -29.39798, -4.13946),
00755         Vec3(82.99039, -29.15258, -4.13946),
00756         Vec3(83.67827, -29.51994, -4.13946),
00757         Vec3(83.71796, -29.27433, -4.13946),
00758         Vec3(84.40811, -29.63816, -4.13946),
00759         Vec3(84.44649, -29.39235, -4.13946),
00760         Vec3(85.13901, -29.75254, -4.13946),
00761         Vec3(85.17603, -29.50652, -4.13946),
00762         Vec3(85.87090, -29.86293, -4.13946),
00763         Vec3(85.90651, -29.61670, -4.13946),
00764         Vec3(86.60381, -29.96919, -4.13946),
00765         Vec3(86.63797, -29.72276, -4.13946),
00766         Vec3(87.33777, -30.07119, -4.13946),
00767         Vec3(87.37045, -29.82456, -4.13946),
00768         Vec3(88.07271, -30.16878, -4.13946),
00769         Vec3(88.10385, -29.92194, -4.13946),
00770         Vec3(88.80864, -30.26180, -4.13946),
00771         Vec3(88.83820, -30.01477, -4.13946),
00772         Vec3(89.54555, -30.35012, -4.13946),
00773         Vec3(89.57347, -30.10291, -4.13946),
00774         Vec3(90.28349, -30.43360, -4.13946),
00775         Vec3(90.30974, -30.18620, -4.13946),
00776         Vec3(91.02234, -30.51208, -4.13946),
00777         Vec3(91.04686, -30.26450, -4.13946),
00778         Vec3(91.76216, -30.58542, -4.13946),
00779         Vec3(91.78491, -30.33767, -4.13946),
00780         Vec3(92.50294, -30.65347, -4.13946),
00781         Vec3(92.52386, -30.40556, -4.13946),
00782         Vec3(93.24464, -30.71607, -4.13946),
00783         Vec3(93.26369, -30.46801, -4.13946),
00784         Vec3(93.98729, -30.77305, -4.13946),
00785         Vec3(94.00441, -30.52485, -4.13946),
00786         Vec3(94.73073, -30.82425, -4.13946),
00787         Vec3(94.74586, -30.57592, -4.13946),
00788         Vec3(95.47499, -30.86948, -4.13946),
00789         Vec3(95.48811, -30.62103, -4.13946),
00790         Vec3(96.22003, -30.90857, -4.13946),
00791         Vec3(96.23108, -30.66003, -4.13946),
00792         Vec3(96.96579, -30.94136, -4.13946),
00793         Vec3(96.97472, -30.69273, -4.13946),
00794         Vec3(97.71220, -30.96767, -4.13946),
00795         Vec3(97.71896, -30.71898, -4.13946),
00796         Vec3(98.45927, -30.98733, -4.13946),
00797         Vec3(98.46381, -30.73858, -4.13946),
00798         Vec3(99.20700, -31.00016, -4.13946),
00799         Vec3(99.20926, -30.75138, -4.13946),
00800         Vec3(99.95525, -31.00600, -4.13946),
00801         Vec3(99.95518, -30.75721, -4.13946),
00802         Vec3(100.70396, -31.00468, -4.13946),
00803         Vec3(100.70152, -30.75591, -4.13946),
00804         Vec3(101.45297, -30.99604, -4.13946),
00805         Vec3(101.44812, -30.74730, -4.13946),
00806         Vec3(102.20236, -30.97993, -4.13946),
00807         Vec3(102.19502, -30.73125, -4.13946),
00808         Vec3(102.95190, -30.95619, -4.13946),
00809         Vec3(102.94205, -30.70760, -4.13946),
00810         Vec3(103.70155, -30.92465, -4.13946),
00811         Vec3(103.68915, -30.67617, -4.13946),
00812         Vec3(104.45116, -30.88515, -4.13946),
00813         Vec3(104.43614, -30.63682, -4.13946),
00814         Vec3(105.20073, -30.83755, -4.13946),
00815         Vec3(105.18306, -30.58939, -4.13946),
00816         Vec3(105.95012, -30.78167, -4.13946),
00817         Vec3(105.92976, -30.53371, -4.13946),
00818         Vec3(106.69923, -30.71735, -4.13946),
```

```
00819         Vec3(106.67615, -30.46964, -4.13946),
00820         Vec3(107.44794, -30.64451, -4.13946),
00821         Vec3(107.42207, -30.39708, -4.13946),
00822         Vec3(108.19614, -30.56305, -4.13946),
00823         Vec3(108.16745, -30.31592, -4.13946),
00824         Vec3(108.94363, -30.47289, -4.13946),
00825         Vec3(108.91214, -30.22610, -4.13946),
00826         Vec3(109.69040, -30.37395, -4.13946),
00827         Vec3(109.65598, -30.12755, -4.13946),
00828         Vec3(110.43625, -30.26616, -4.13946),
00829         Vec3(110.39890, -30.02018, -4.13946),
00830         Vec3(111.18094, -30.14942, -4.13946),
00831         Vec3(111.14066, -29.90391, -4.13946),
00832         Vec3(111.92453, -30.02366, -4.13946),
00833         Vec3(111.88126, -29.77866, -4.13946),
00834         Vec3(112.66678, -29.88881, -4.13946),
00835         Vec3(112.62057, -29.64436, -4.13946),
00836         Vec3(113.40756, -29.74479, -4.13946),
00837         Vec3(113.35837, -29.50093, -4.13946),
00838         Vec3(114.14676, -29.59155, -4.13946),
00839         Vec3(114.09445, -29.34832, -4.13946),
00840         Vec3(114.88419, -29.42905, -4.13946),
00841         Vec3(114.82883, -29.18649, -4.13946),
00842         Vec3(115.61978, -29.25727, -4.13946),
00843         Vec3(115.56137, -29.01543, -4.13946),
00844         Vec3(116.35324, -29.07620, -4.13946),
00845         Vec3(116.29178, -28.83513, -4.13946),
00846         Vec3(117.08456, -28.88583, -4.13946),
00847         Vec3(117.01999, -28.64557, -4.13946),
00848         Vec3(117.81351, -28.68613, -4.13946),
00849         Vec3(117.74588, -28.44672, -4.13946),
00850         Vec3(118.54001, -28.47708, -4.13946),
00851         Vec3(118.46921, -28.23857, -4.13946),
00852         Vec3(119.26389, -28.25867, -4.13946),
00853         Vec3(119.19003, -28.02111, -4.13946),
00854         Vec3(119.98489, -28.03091, -4.13946),
00855         Vec3(119.90793, -27.79432, -4.13946),
00856         Vec3(120.70297, -27.79375, -4.13946),
00857         Vec3(120.62296, -27.55820, -4.13946),
00858         Vec3(121.41800, -27.54724, -4.13946),
00859         Vec3(121.33487, -27.31275, -4.13946),
00860         Vec3(122.12973, -27.29139, -4.13946),
00861         Vec3(122.04355, -27.05801, -4.13946),
00862         Vec3(122.83798, -27.02622, -4.13946),
00863         Vec3(122.74875, -26.79399, -4.13946),
00864         Vec3(123.54245, -26.75179, -4.13946),
00865         Vec3(123.45023, -26.52074, -4.13946),
00866         Vec3(124.24313, -26.46810, -4.13946),
00867         Vec3(124.14792, -26.23826, -4.13946),
00868         Vec3(124.93985, -26.17520, -4.13946),
00869         Vec3(124.84171, -25.94661, -4.13946),
00870         Vec3(125.63235, -25.87313, -4.13946),
00871         Vec3(125.53128, -25.64581, -4.13946),
00872         Vec3(126.32047, -25.56192, -4.13946),
00873         Vec3(126.21652, -25.33589, -4.13946),
00874         Vec3(127.00406, -25.24163, -4.13946),
00875         Vec3(126.89731, -25.01692, -4.13946),
00876         Vec3(127.68301, -24.91237, -4.13946),
00877         Vec3(127.57346, -24.68899, -4.13946),
00878         Vec3(128.35739, -24.57438, -4.13946),
00879         Vec3(128.24515, -24.35235, -4.13946),
00880         Vec3(129.02719, -24.22785, -4.13946),
00881         Vec3(128.91226, -24.00720, -4.13946),
00882         Vec3(129.69247, -23.87303, -4.13945),
00883         Vec3(129.57492, -23.65377, -4.13945),
00884         Vec3(129.69247, -23.87303, -4.13945),
00885         Vec3(130.35318, -23.51014, -4.13945),
00886         Vec3(130.23306, -23.29229, -4.13945),
00887         Vec3(129.57492, -23.65377, -4.13945),
00888         Vec3(130.35318, -23.51014, -4.13945),
00889         Vec3(131.00949, -23.13936, -4.13944),
00890         Vec3(130.88675, -22.92295, -4.13944),
00891         Vec3(130.23306, -23.29229, -4.13945),
00892         Vec3(131.66141, -22.76093, -4.13942),
00893         Vec3(131.53610, -22.54597, -4.13942),
00894         Vec3(131.66141, -22.76093, -4.13942),
00895         Vec3(132.30887, -22.37506, -4.13941),
00896         Vec3(132.18106, -22.16159, -4.13941),
00897         Vec3(131.53610, -22.54597, -4.13942),
00898         Vec3(132.95193, -21.98195, -4.13938),
00899         Vec3(132.82162, -21.76999, -4.13938),
00900         Vec3(132.95193, -21.98195, -4.13938),
00901         Vec3(133.59067, -21.58176, -4.13936),
00902         Vec3(133.45786, -21.37134, -4.13936),
00903         Vec3(132.82162, -21.76999, -4.13938),
00904         Vec3(134.22476, -21.17449, -4.13933),
00905         Vec3(134.08945, -20.96564, -4.13933),
```

```
00906         Vec3(134.22476, -21.17449, -4.13933),
00907         Vec3(134.85403, -20.76006, -4.13930),
00908         Vec3(134.71628, -20.55281, -4.13930),
00909         Vec3(134.08945, -20.96564, -4.13933),
00910         Vec3(135.47836, -20.33843, -4.13926),
00911         Vec3(135.33817, -20.13280, -4.13926),
00912         Vec3(135.47836, -20.33843, -4.13926),
00913         Vec3(136.09750, -19.90954, -4.13923),
00914         Vec3(135.95486, -19.70555, -4.13923),
00915         Vec3(135.33817, -20.13280, -4.13926),
00916         Vec3(136.09750, -19.90954, -4.13923),
00917         Vec3(136.71121, -19.47334, -4.13920),
00918         Vec3(136.56625, -19.27103, -4.13920),
00919         Vec3(135.95486, -19.70555, -4.13923),
00920         Vec3(136.71121, -19.47334, -4.13920),
00921         Vec3(137.31918, -19.02981, -4.13917),
00922         Vec3(137.17184, -18.82919, -4.13917),
00923         Vec3(136.56625, -19.27103, -4.13920),
00924         Vec3(137.92136, -18.57885, -4.13914),
00925         Vec3(137.77170, -18.37995, -4.13914),
00926         Vec3(137.92136, -18.57885, -4.13914),
00927         Vec3(138.51749, -18.12045, -4.13912),
00928         Vec3(138.36551, -17.92327, -4.13912),
00929         Vec3(137.77170, -18.37995, -4.13914),
00930         Vec3(138.51749, -18.12045, -4.13912),
00931         Vec3(139.10733, -17.65455, -4.13910),
00932         Vec3(138.95309, -17.45913, -4.13910),
00933         Vec3(138.36551, -17.92327, -4.13912),
00934         Vec3(139.10733, -17.65455, -4.13910),
00935         Vec3(139.69089, -17.18132, -4.13909),
00936         Vec3(139.53445, -16.98767, -4.13909),
00937         Vec3(138.95309, -17.45913, -4.13910),
00938         Vec3(139.69089, -17.18132, -4.13909),
00939         Vec3(140.26834, -16.70105, -4.13908),
00940         Vec3(140.10971, -16.50917, -4.13908),
00941         Vec3(139.53445, -16.98767, -4.13909),
00942         Vec3(140.26834, -16.70105, -4.13908),
00943         Vec3(140.83969, -16.21398, -4.13908),
00944         Vec3(140.67892, -16.02390, -4.13908),
00945         Vec3(140.10971, -16.50917, -4.13908),
00946         Vec3(141.40512, -15.72042, -4.13909),
00947         Vec3(141.24222, -15.53213, -4.13909),
00948         Vec3(141.40512, -15.72042, -4.13909),
00949         Vec3(141.96469, -15.22066, -4.13909),
00950         Vec3(141.79977, -15.03416, -4.13909),
00951         Vec3(141.24222, -15.53213, -4.13909),
00952         Vec3(142.51852, -14.71498, -4.13911),
00953         Vec3(142.35165, -14.53027, -4.13911),
00954         Vec3(143.06680, -14.20365, -4.13912),
00955         Vec3(142.89798, -14.02073, -4.13912),
00956         Vec3(143.60959, -13.68697, -4.13913),
00957         Vec3(143.43881, -13.50582, -4.13913),
00958         Vec3(143.60959, -13.68697, -4.13913),
00959         Vec3(144.14700, -13.16521, -4.13915),
00960         Vec3(143.97440, -12.98583, -4.13915),
00961         Vec3(143.43881, -13.50582, -4.13913),
00962         Vec3(144.67917, -12.63868, -4.13917),
00963         Vec3(144.50479, -12.46104, -4.13917),
00964         Vec3(145.20627, -12.10758, -4.13919),
00965         Vec3(145.03024, -11.93166, -4.13919),
00966         Vec3(145.72855, -11.57214, -4.13921),
00967         Vec3(145.55075, -11.39793, -4.13921),
00968         Vec3(146.24606, -11.03262, -4.13923),
00969         Vec3(146.06668, -10.86008, -4.13923),
00970         Vec3(146.75900, -10.48923, -4.13925),
00971         Vec3(146.57809, -10.31835, -4.13925),
00972         Vec3(147.26761, -9.94223, -4.13927),
00973         Vec3(147.08511, -9.77299, -4.13927),
00974         Vec3(147.77200, -9.39182, -4.13929),
00975         Vec3(147.58804, -9.22418, -4.13929),
00976         Vec3(148.27231, -8.83824, -4.13931),
00977         Vec3(148.08701, -8.67218, -4.13931),
00978         Vec3(148.76883, -8.28173, -4.13933),
00979         Vec3(148.58212, -8.11722, -4.13933),
00980         Vec3(148.76883, -8.28173, -4.13933),
00981         Vec3(149.45541, -7.59785, -4.13937),
00982         Vec3(149.26712, -7.43516, -4.13938),
00983         Vec3(148.58212, -8.11722, -4.13933),
00984         Vec3(149.45541, -7.59785, -4.13937),
00985         Vec3(150.07224, -6.88286, -4.13939),
00986         Vec3(149.88278, -6.72160, -4.13939),
00987         Vec3(149.26712, -7.43516, -4.13938),
00988         Vec3(150.07224, -6.88286, -4.13939),
00989         Vec3(150.63742, -6.21586, -4.13941),
00990         Vec3(150.44656, -6.05628, -4.13941),
00991         Vec3(149.88278, -6.72160, -4.13939),
00992         Vec3(151.15543, -5.59434, -4.13942),
```

```
00993         Vec3(150.96304, -5.43658, -4.13942),
00994         Vec3(151.63089, -5.01586, -4.13943),
00995         Vec3(151.43698, -4.85992, -4.13943),
00996         Vec3(152.06845, -4.47793, -4.13944),
00997         Vec3(151.87320, -4.32373, -4.13944),
00998         Vec3(152.47275, -3.97818, -4.13944),
00999         Vec3(152.27634, -3.82548, -4.13944),
01000         Vec3(152.84860, -3.51422, -4.13945),
01001         Vec3(152.65128, -3.36267, -4.13945),
01002         Vec3(153.20071, -3.08368, -4.13945),
01003         Vec3(153.00284, -2.93281, -4.13945),
01004         Vec3(254.86276, 117.97976, -4.13946)
01005     };
01006
01007     /*
01008      * method itterate's throw the Vec3 Array, insert the Points in the AnimationPath and calculate the
      correct angle between two points.
01009      * not in use anymore due to bugs.
01010      * This method was the idea of a good solution but becaused of not enough time I had to use the "brute
      force" solution, see below.
01011      *
01012     float j = 0.0f;
01013     float angleCount = 0;
01014     float angleLast = 0;
01015     float angleNew = 0;
01016     path->insert(0.0f, osg::AnimationPath::ControlPoint(pathArray[0]));
01017     for (int i = 10; i < (sizeof(pathArray) / sizeof(pathArray[0])); i += vectorCount) { //
     (sizeof(pathArray) / sizeof(pathArray[0])
01018         j += time; //time
01019         angleNew = getAngleRad(pathArray[i - vectorCount], pathArray[i]);
01020         if(angleNew > angleLast) {
01021           angleCount += angleNew;
01022         }
01023         else{
01024           angleCount -= angleNew;
01025         }
01026         path->insert((j), osg::AnimationPath::ControlPoint(pathArray[i], Quat(angleCount, osg::Z_AXIS)));
01027         angleLast = angleNew;
01028     }
01029     */
01030
01031
01032     //Insert every 10-th point from the Vec3 array. The time variable pinpoints the time that the Object
      will take between two points.
01033     //every insert command contains the angle around the Z-Axis.
01034     double t = 0.0f;
01035     int i = 1;
01036     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[0])); //start point with longer
      distance than the others
01037     path->insert((t = t + time + 2), osg::AnimationPath::ControlPoint(pathArray[i]));
01038     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.02
     8, osg::Z_AXIS)));
01039     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.04
     0, osg::Z_AXIS)));
01040     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.07
     9, osg::Z_AXIS)));
01041     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.14
     0, osg::Z_AXIS)));
01042     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.16
     5, osg::Z_AXIS)));
01043     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.21
     5, osg::Z_AXIS)));
01044     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.26
     5, osg::Z_AXIS)));
01045     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.30
     0, osg::Z_AXIS)));
01046     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.34
     0, osg::Z_AXIS)));
01047     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.38
     0, osg::Z_AXIS)));
01048     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.42
     5, osg::Z_AXIS)));
01049     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.47
     0, osg::Z_AXIS)));
01050     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.53
     0, osg::Z_AXIS)));
01051     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.59
     0, osg::Z_AXIS)));
01052     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.65
     0, osg::Z_AXIS)));
01053     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.71
     0, osg::Z_AXIS)));
01054     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.76
     5, osg::Z_AXIS)));
01055     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.82
     0, osg::Z_AXIS)));
01056     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.86
```

```
      0, osg::Z_AXIS)));
01057     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.92
      0, osg::Z_AXIS)));
01058     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.94
      0, osg::Z_AXIS)));
01059     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.96
      5, osg::Z_AXIS)));
01060     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.99
      0, osg::Z_AXIS)));
01061     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.02
      0, osg::Z_AXIS)));
01062     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.05
      0, osg::Z_AXIS)));
01063     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.07
      0, osg::Z_AXIS)));
01064     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.08
      0, osg::Z_AXIS)));
01065     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.08
      0, osg::Z_AXIS)));
01066     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.08
      5, osg::Z_AXIS)));
01067     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.09
      0, osg::Z_AXIS)));
01068     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.09
      0, osg::Z_AXIS)));
01069     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.09
      0, osg::Z_AXIS)));
01070     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.08
      5, osg::Z_AXIS)));
01071     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.08
      0, osg::Z_AXIS)));
01072     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.07
      0, osg::Z_AXIS)));
01073     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.06
      0, osg::Z_AXIS)));
01074     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.05
      0, osg::Z_AXIS)));
01075     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.05
      0, osg::Z_AXIS)));
01076     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.05
      0, osg::Z_AXIS)));
01077     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.04
      0, osg::Z_AXIS)));
01078     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.00
      0, osg::Z_AXIS)));
01079     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.98
      0, osg::Z_AXIS)));
01080     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.95
      0, osg::Z_AXIS)));
01081     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.91
      0, osg::Z_AXIS)));
01082     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.87
      0, osg::Z_AXIS)));
01083     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.83
      0, osg::Z_AXIS)));
01084     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.81
      0, osg::Z_AXIS)));
01085     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.79
      0, osg::Z_AXIS)));
01086     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.76
      0, osg::Z_AXIS)));
01087     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.74
      5, osg::Z_AXIS)));
01088     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.72
      5, osg::Z_AXIS)));
01089     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.70
      5, osg::Z_AXIS)));
01090     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.69
      0, osg::Z_AXIS)));
01091     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.67
      5, osg::Z_AXIS)));
01092     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.66
      0, osg::Z_AXIS)));
01093     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.64
      5, osg::Z_AXIS)));
01094     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.63
      0, osg::Z_AXIS)));
01095     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.62
      0, osg::Z_AXIS)));
01096     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.61
      0, osg::Z_AXIS)));
01097     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.60
      0, osg::Z_AXIS)));
01098     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.60
      0, osg::Z_AXIS)));
01099     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.59
      0, osg::Z_AXIS)));
```

```
01100     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.59
      0, osg::Z_AXIS)));
01101     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.60
      0, osg::Z_AXIS)));
01102     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.61
      0, osg::Z_AXIS)));
01103     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.62
      0, osg::Z_AXIS)));
01104     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.63
      0, osg::Z_AXIS)));
01105     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.64
      0, osg::Z_AXIS)));
01106     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.65
      0, osg::Z_AXIS)));
01107     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.66
      0, osg::Z_AXIS)));
01108     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.68
      0, osg::Z_AXIS)));
01109     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.70
      0, osg::Z_AXIS)));
01110     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.72
      0, osg::Z_AXIS)));
01111     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.75
      0, osg::Z_AXIS)));
01112     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.78
      0, osg::Z_AXIS)));
01113     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.80
      0, osg::Z_AXIS)));
01114     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.83
      0, osg::Z_AXIS)));
01115     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.88
      0, osg::Z_AXIS)));
01116     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.94
      0, osg::Z_AXIS)));
01117     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(0.99
      0, osg::Z_AXIS)));
01118     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.06
      0, osg::Z_AXIS)));
01119     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.12
      0, osg::Z_AXIS)));
01120     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.15
      0, osg::Z_AXIS)));
01121     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.19
      0, osg::Z_AXIS)));
01122     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.22
      0, osg::Z_AXIS)));
01123     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.26
      0, osg::Z_AXIS)));
01124     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.29
      0, osg::Z_AXIS)));
01125     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.35
      0, osg::Z_AXIS)));
01126     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.40
      0, osg::Z_AXIS)));
01127     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.46
      0, osg::Z_AXIS)));
01128     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.50
      0, osg::Z_AXIS)));
01129     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.52
      0, osg::Z_AXIS)));
01130     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.56
      0, osg::Z_AXIS)));
01131     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.58
      0, osg::Z_AXIS)));
01132     path->insert((t = t + time), osg::AnimationPath::ControlPoint(pathArray[i = i + vectorCount], Quat(1.60
      0, osg::Z_AXIS)));
01133
01134     return path.release();
01135 }
```

9.3 Callbacks/AddPortalGunInteractionCallback.cpp File Reference

```
#include "../header/AddPortalGunInteractionCallback.h"
#include "../header/WeaponHUD.h"
```

Namespaces

- brtr

Namespace for the whole BrainTrain Project.

9.4 AddPortalGunInteractionCallback.cpp

```
00001 #include "../header/AddPortalGunInteractionCallback.h"
00002 #include "../header/WeaponHUD.h"
00003
00004 namespace brtr{
00005     AddPortalGunInteractionCallback::AddPortalGunInteractionCallback
      (osg::Node* weaponHUD, osg::Camera* hudCam, osg::Switch* switcher, int width, int height):
00006     BaseInteractionCallback(weaponHUD,hudCam,width,height),
00007     _switcher(switcher){}
00008
00009     void AddPortalGunInteractionCallback::setText() {
00010         _text->setText("Defect Portal Gun. \nLeft Click to pick it up anyway.");
00011     }
00012
00013     void AddPortalGunInteractionCallback::interact(osg::Node* node
      , osg::NodeVisitor* nv) {
00014         _switcher->setAllChildrenOff();
00015         brtr::WeaponHUD* weaponHUD = static_cast<brtr::WeaponHUD*>(node);
00016         weaponHUD->addPortalGun();
00017         OSG_ALWAYS << "after add method" << std:: endl;
00018         _done = true;
00019     }
00020
00021 }
```

9.5 Callbacks/BaseInteractionCallback.cpp File Reference

```
#include "../header/BaseInteractionCallback.h"
#include <osg/Camera>
#include "../header/UtilFunctions.h"
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.6 BaseInteractionCallback.cpp

```
00001 #include "../header/BaseInteractionCallback.h"
00002 #include <osg/Camera>
00003 #include "../header/UtilFunctions.h"
00004 namespace brtr {
00005
00006     BaseInteractionCallback::BaseInteractionCallback(
      osg::Node* attachTo, osg::Camera* hudCam, int width, int height) :
00007         _attachTo(attachTo),
00008         _hudCam(hudCam),
00009         _done(false){
00010         _text = brtr::createText(osg::Vec3d(width / 2.0 - 320, height / 2.0 - 110, 0),
       "", width * 0.02);
00011         osg::ref_ptr<osg::Geode> textGeode = new osg::Geode;
00012         textGeode->addDrawable(_text);
00013         _hudCam->addChild(textGeode);
00014     }
00015
00016     void BaseInteractionCallback::operator()(osg::Node* node,
      osg::NodeVisitor* nv) {
00017         if (!_done) {
00018             interact(node, nv);
00019         }
00020         traverse(node, nv);
00021     }
00022
00023     void BaseInteractionCallback::setNode(osg::ref_ptr<osg::Node> val) {
00024         _attachTo = val;
```

```
00025     }
00026
00027     osg::ref_ptr<osg::Node> BaseInteractionCallback::getNode() const {
00028         return _attachTo;
00029     }
00030
00031     void BaseInteractionCallback::clearText() {
00032         _text->setText("");
00033     }
00034
00035     void BaseInteractionCallback::reactivate() {
00036         _done = false;
00037     }
00038
00039 }
00040
```

9.7 Callbacks/DrunkenInteractionCallback.cpp File Reference

```
#include "../header/DrunkenInteractionCallback.h"
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.8 DrunkenInteractionCallback.cpp

```
00001 #include "../header/DrunkenInteractionCallback.h"
00002
00003
00004 namespace brtr {
00005     DrunkenInteractionCallback::DrunkenInteractionCallback
      (osg::Node* camera,osg::Camera* hudCam, osg::Switch* geometrySwitch, int width, int height) :
00006     BaseInteractionCallback(camera, hudCam, width, height),
00007     _geometrySwitch(geometrySwitch),
00008     _startTime(0),
00009     _backwards(false){
00010         _motion = new osgAnimation::LinearMotion(70, 70, 70, osgAnimation::Motion::LOOP);
00011     }
00012
00013     void DrunkenInteractionCallback::interact(osg::Node* node,
      osg::NodeVisitor* nv) {
00014         //save starttime and switch off the bottle
00015         if (_startTime == 0) {
00016             _geometrySwitch->setAllChildrenOff();
00017             _startTime = nv->getFrameStamp()->getReferenceTime();
00018         }
00019         //linear motion for the effect
00020         _motion->update(1);
00021         osg::Camera* camera = static_cast<osg::Camera*>(node);
00022         if (camera) {
00023             OSG_NOTICE << "Motion Value " << _motion->getValue() << std::endl;
00024             //backwards after up to FOV 125
00025             if (_motion->getValue() >= 125)
00026                 _backwards = !_backwards;
00027             camera->setProjectionMatrixAsPerspective(_backwards ? 195 -
      _motion->getValue() : _motion->getValue(), 2, 0.01, 100000);
00028         }//camera
00029
00030         if (nv->getFrameStamp()->getReferenceTime() - _startTime >= 20) {
00031             if (camera) {
00032                 camera->setProjectionMatrixAsPerspective(70, 1.778, 0.01, 100000);
00033                 _done = true;
00034                 _geometrySwitch = nullptr;
00035                 _hudCam = nullptr;
00036                 _attachTo = nullptr;
00037                 _motion = nullptr;
00038                 _text = nullptr;
00039                 OSG_NOTICE << "DrunkenInteractionCallback: Should now be removed" << std::endl;
00040             }//camera
00041         }//30 sec over
00042     }
```

```
00043
00044     void DrunkenInteractionCallback::setText() {
00045         _text->setText("Click Left Mouse for a drink!");
00046     }
00047
00048 }
00049
```

9.9 Callbacks/ProgramSwitcherCallback.cpp File Reference

```
#include <osg/NodeCallback>
#include <osgViewer/Viewer>
#include "../header/ProgramSwitcherCallback.h"
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.10 ProgramSwitcherCallback.cpp

```
00001 #include <osg/NodeCallback>
00002 #include <osgViewer/Viewer>
00003 #include "../header/ProgramSwitcherCallback.h"
00004
00005 namespace brtr {
00006
00007
00008     ProgramSwitcherCallback::ProgramSwitcherCallback(
      osg::Node* postprocessCam, osg::Camera* hudCam, int width, int height, std::vector<osg::ref_ptr<osg::Program>>
      programs):
00009         BaseInteractionCallback(postprocessCam,hudCam,width,height),
00010         _programs(programs),
00011         _curProg(0){}
00012
00013     void ProgramSwitcherCallback::setText() {
00014         _text->setText("You feel a mysterious power\nfrom this strange device.\nA click will change
       the world...");
00015     }
00016
00017     void ProgramSwitcherCallback::interact(osg::Node*, osg::NodeVisitor*)
     {
00018         _done = true;
00019         _attachTo->getOrCreateStateSet()->removeAttribute(_programs[
      _curProg]);
00020         _curProg++;
00021         _curProg = _curProg % _programs.size();
00022         _attachTo->getOrCreateStateSet()->setAttributeAndModes(
      _programs[_curProg], osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON);
00023     }
00024
00025
00026
00027 }
```

9.11 Callbacks/ToonTexSwitcherCallback.cpp File Reference

```
#include <osg/NodeCallback>
#include <osgViewer/Viewer>
#include <osgDB/ReadFile>
#include "../header/ToonTexSwitcherCallback.h"
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.12 ToonTexSwitcherCallback.cpp

```
00001 #include <osg/NodeCallback>
00002 #include <osgViewer/Viewer>
00003 #include <osgDB/ReadFile>
00004 #include "../header/ToonTexSwitcherCallback.h"
00005
00006 namespace brtr {
00007
00008
00009     ToonTexSwitcherCallback::ToonTexSwitcherCallback(
      osg::Node* sceneData, osg::Camera* hudCam, int width, int height, std::vector<osg::ref_ptr<osg::Texture2D>>
      toonTexs) :
00010     BaseInteractionCallback(sceneData,hudCam,width,height),
00011     _curTex(0),
00012     _toonTexs(toonTexs){ }
00013
00014     void ToonTexSwitcherCallback::setText() {
00015         _text->setText("The colors of the world\nare hidden here.\nTouch them, if you dare.");
00016     }
00017
00018     void ToonTexSwitcherCallback::interact(osg::Node*, osg::NodeVisitor*)
      {
00019         _done = true;
00020         _attachTo->getOrCreateStateSet()->removeTextureAttribute(1,
      _toonTexs[_curTex]);
00021         _curTex++;
00022         _curTex = _curTex % _toonTexs.size();
00023         _attachTo->getOrCreateStateSet()->setTextureAttribute(1,
      _toonTexs[_curTex], osg::StateAttribute::ON | osg::StateAttribute::OVERRIDE);
00024     }
00025
00026
00027
00028 }
```

9.13 Callbacks/TrainSwitcherCallback.cpp File Reference

```
#include "../header/TrainSwitcherCallback.h"
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.14 TrainSwitcherCallback.cpp

```
00001 #include "../header/TrainSwitcherCallback.h"
00002
00003 namespace brtr{
00004     TrainSwitcherCallback::TrainSwitcherCallback():
00005         _curActiveTrain(0),
00006         _deltaTime(0){}
00007
00008     void TrainSwitcherCallback::operator()(osg::Node* node,
      osg::NodeVisitor* nv) {
00009         //deltatime == 0? So wee need a new timestamp
00010         if (_deltaTime == 0) {
00011             _deltaTime = nv->getFrameStamp()->getReferenceTime();
00012         }
00013         //it is attached to a switch, so the node is a switch
00014         osg::Switch* switcher = static_cast<osg::Switch*>(node);
```

```
00015         if (nv->getFrameStamp()->getReferenceTime() - _deltaTime > 36) {
00016             _curActiveTrain++;
00017             _curActiveTrain = _curActiveTrain % switcher->getNumChildren();
00018             switcher->setAllChildrenOff();
00019             switcher->setValue(_curActiveTrain, true);
00020             _deltaTime = 0;
00021         }
00022         traverse(node, nv);
00023     }
00024
00025 }
00026
```

9.15 Camera/FPSCameraManipulator.cpp File Reference

```
#include "../header/FPSCameraManipulator.h"
#include "../header/UtilFunctions.h"
#include <osgGA/GUIEventAdapter>
#include <osgViewer/Viewer>
#include <osg/PositionAttitudeTransform>
#include <osgDB/ReadFile>
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.16 FPSCameraManipulator.cpp

```
00001 #include "../header/FPSCameraManipulator.h"
00002 #include "../header/UtilFunctions.h"
00003 #include <osgGA/GUIEventAdapter>
00004 #include <osgViewer/Viewer>
00005 #include <osg/PositionAttitudeTransform>
00006 #include <osgDB/ReadFile>
00007
00008 using namespace osg;
00009 using namespace osgGA;
00010
00011 namespace brtr {
00012     FPSCameraManipulator::FPSCameraManipulator(double movementSpeed, double zHeight, Node* root, bool
      flightMode)
00013         :FirstPersonManipulator(),
00014         _forwardMovement(false),
00015         _backwardMovement(false),
00016         _leftMovement(false),
00017         _rightMovement(false),
00018         _upMovement(false),
00019         _downMovement(false),
00020         _attachBody(true),
00021         _shift(false),
00022         _ctrl(false),
00023         _jumpingUp(false),
00024         _jumpingDown(false),
00025         _crouch(false),
00026         _movementSpeed(movementSpeed),
00027         _zHeight(zHeight),
00028         _flightMode(flightMode),
00029         _savedzHeight(zHeight),
00030         _maxFallHeight(50),
00031         _frameFactor(30),
00032         _intensity(1.0),
00033         _bodyLength(0.0),
00034         _savedzHeightCrouch(0.0),
00035         _jumpHeight(4)
00036     {
00037         setNode(root);
00038         ref_ptr<Node> evaBody= osgDB::readNodeFile("../BlenderFiles/exports/BodyEva.ive");
00039         _body = new PositionAttitudeTransform();
00040         setHomePosition(Vec3(0, 134, 35 + zHeight + 5), Vec3(-1, 0, 26 + zHeight + 5), Z_AXIS);
```

```
00041         //_body->addChild(evaBody); body not used anymore
00042         _body->getOrCreateStateSet()->setAttribute(nullptr);
00043         Vec3d bodyPos = _homeEye;
00044         auto forward = _rotation * osg::Vec3d(0.0, 0.0, -1.0);
00045         bodyPos._v[2] =_homeEye._v[2]-1;
00046         _body->setPosition(bodyPos);
00047         _body->setNodeMask(~brtr::interactionAndCollisionMask);
00048         _body->getOrCreateStateSet()->addUniform(new Uniform("tex", false), StateAttribute::ON |
      StateAttribute::OVERRIDE);
00049         getNode()->asGroup()->addChild(_body);
00050         home(0);
00051
00052     }
00053
00054     FPSCameraManipulator::~FPSCameraManipulator() {}
00055
00056     bool FPSCameraManipulator::handleMouseMove(const
      osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& us) {
00057         addMouseEvent(ea);
00058         if (performMovement())
00059             us.requestRedraw();
00060         return false;
00061     }
00062
00063     bool FPSCameraManipulator::handleFrame(const osgGA::GUIEventAdapter&
      ea, osgGA::GUIActionAdapter& us) {
00064         FirstPersonManipulator::handleFrame(ea, us);
00065         if (performEyeMovement())
00066             us.requestRedraw();
00067         centerMousePointer(ea, us);
00068         return false;
00069     }
00070
00071     bool FPSCameraManipulator::handleKeyDown(const
      osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& us) {
00072         //FirstPersonManipulator::handleKeyDown(ea, us);
00073         switch (ea.getUnmodifiedKey()) {
00074         case GUIEventAdapter::KEY_W:
00075             _forwardMovement = true;
00076             return true;
00077         case GUIEventAdapter::KEY_A:
00078             _leftMovement = true;
00079             return true;
00080         case GUIEventAdapter::KEY_S:
00081             _backwardMovement = true;
00082             return true;
00083         case GUIEventAdapter::KEY_D:
00084             _rightMovement = true;
00085             return true;
00086         case GUIEventAdapter::KEY_E:
00087             if (_flightMode)
00088                 _upMovement = true;
00089             return true;
00090         case GUIEventAdapter::KEY_Q:
00091             if (_flightMode)
00092                 _downMovement = true;
00093             return true;
00094         case GUIEventAdapter::KEY_X:
00095             _crouch = !_crouch;
00096             _savedzHeightCrouch = _crouch ? _zHeight :
      _savedzHeightCrouch;
00097             _zHeight = _crouch ? _zHeight -3 :
      _savedzHeightCrouch;
00098             return true;
00099         case GUIEventAdapter::KEY_F:
00100             _flightMode = !_flightMode;
00101             return true;
00102         case GUIEventAdapter::KEY_G:
00103             _attachBody = !_attachBody;
00104             return true;
00105         case GUIEventAdapter::KEY_Space:
00106             if (!_jumpingUp && !_jumpingDown && !
      _flightMode) {
00107                 _savedzHeight = _zHeight;
00108                 _jumpingUp = true;
00109             }
00110             return true;
00111         case GUIEventAdapter::KEY_Shift_L:
00112            _shift = true;
00113             return true;
00114         case GUIEventAdapter::KEY_Control_L:
00115             _ctrl = true;
00116             return true;
00117         default:
00118             return false;
00119         }
00120
```

```
00121     }
00122
00123     bool FPSCameraManipulator::handleKeyUp(const osgGA::GUIEventAdapter&
      ea, osgGA::GUIActionAdapter& us) {
00124         switch (ea.getUnmodifiedKey()) {
00125         case GUIEventAdapter::KEY_W:
00126             _forwardMovement = false;
00127             return true;
00128         case GUIEventAdapter::KEY_A:
00129             _leftMovement = false;
00130             return true;
00131         case GUIEventAdapter::KEY_S:
00132             _backwardMovement = false;
00133             return true;
00134         case GUIEventAdapter::KEY_D:
00135             _rightMovement = false;
00136             return true;
00137         case GUIEventAdapter::KEY_E:
00138             _upMovement = false;
00139             return true;
00140         case GUIEventAdapter::KEY_Q:
00141             _downMovement = false;
00142             return true;
00143         case GUIEventAdapter::KEY_Shift_L:
00144             _shift = false;
00145             return true;
00146         case GUIEventAdapter::KEY_Control_L:
00147             _ctrl = false;
00148             return true;
00149         default:
00150             return false;
00151         }
00152     }
00153
00154     bool FPSCameraManipulator::performMovement() {
00155         if (_ga_t0.get() == NULL || _ga_t1.get() == NULL) return false;
00156         auto dy = _ga_t0->getYnormalized();
00157         auto dx = _ga_t0->getXnormalized();
00158         auto dt = _ga_t0->getTime() - _ga_t1->getTime();
00159
00160         // return if there is no movement.
00161         if (dx == 0. && dy == 0.)
00162             return false;
00163
00164         //rotate center
00165         Vec3d localUp = getUpVector(getCoordinateFrame(_eye));
00166         rotateYawPitch(_rotation, dx, dy, localUp);
00167         return true;
00168     }
00169
00170     bool FPSCameraManipulator::performEyeMovement() {
00171         double intensity = _intensity * _shift ? 2.0 : 1.0 *
      _ctrl ? 0.5 : 1;
00172         OSG_DEBUG << "Intensity= " << intensity << std::endl;
00173
00174         /*
00175         *  COLLISION DETECTION AND CLAMPING INSPIRED BY                *
00176         *  Viggo Lovli (http://markmail.org/message/e6magjobl7fywbe6)  *
00177         *  AND DRIVEMANIPULATOR                                        *
00178         */
00179
00180         //newEye tmp 'cos we do not know, if the new position is valid
00181         auto newEye = _eye;
00182         //up, right, forward dependent on the view
00183         auto up = _rotation * osg::Vec3d(0.0, 1.0, 0.0);
00184         auto forward = _rotation * osg::Vec3d(0.0, 0.0, -1.0);
00185         auto right = _rotation* osg::Vec3d(1.0, 0.0, 0.0);
00186         osg::Vec3d movement;
00187
00188         //where to move
00189         if (_rightMovement)
00190             movement += right;
00191         if (_leftMovement)
00192             movement += -right;
00193         if (_forwardMovement)
00194             movement += forward * 1.2;
00195         if (_backwardMovement)
00196             movement += -forward * 0.8;
00197         if (_upMovement)
00198             movement += up;
00199         if (_downMovement)
00200             movement += -up;
00201
00202         //_delta_frame_time -> not FPS depended, _frameFactor->'cos _deltaTime is so small
00203         movement *= _movementSpeed * intensity * _delta_frame_time *
      _frameFactor;
00204
```

```
00205         if (_flightMode) {
00206             newEye += movement;
00207             _eye = newEye;
00208         }//if(_flightMode)
00209         else {
00210             double eyeIntersectionDistance = 1000;
00211             if (intersect(newEye, newEye + movement * 10, eyeIntersectionDistance)) {
00212                 if (eyeIntersectionDistance < 1.75)
00213                     movement = Vec3d();
00214             }//if(intersect())
00215 
00216             if (_jumpingUp) {
00217                 if (_zHeight < _savedzHeight + _jumpHeight)
00218                     _zHeight += 0.4 * _delta_frame_time * _frameFactor * 2;
00219                 else {
00220                     _jumpingDown = true;
00221                     _jumpingUp = false;
00222                 } //else
00223             }//if (_jumpingUp)
00224 
00225             if (_jumpingDown) {
00226                 if (_zHeight > _savedzHeight )
00227                     _zHeight -= 0.2 * _delta_frame_time * _frameFactor * 2;
00228                 else {
00229                     _jumpingDown = false;
00230                     _jumpingUp = false;
00231                     _zHeight = _savedzHeight;
00232                 }//else
00233             }// if (_jumpingDown)
00234 
00235             newEye += movement;
00236             if (groundIntersection(newEye))
00237                 _eye = newEye;
00238             else return false;
00239         }
00240         OSG_DEBUG << "eyeZ: " << _eye._v[2]<< std::endl;
00241         if (_attachBody)
00242             _body->setPosition(Vec3d(_eye._v[0], _eye._v[1], _eye._v[2] -1));
00243         return true;
00244     }//performEyeMovement
00245 
00246     bool FPSCameraManipulator::groundIntersection(osg::Vec3d&
    newEye) {
00247         double dist = 0.0;
00248 
00249         if (!intersect(newEye, newEye + Vec3(0, 0, -1) * _maxFallHeight, dist))
00250             return false;
00251         dist -= _zHeight;
00252         OSG_DEBUG << "dDist: " << dist << std::endl;
00253         if (-dist > _zHeight -2.5 || dist > _maxFallHeight) {
00254              return false;
00255         }
00256 
00257 
00258         dist /=2;  //slide effect, smooth up/down
00259         newEye._v[2] -=  dist;
00260         return true;
00261     }//groundIntersection()
00262 
00263     bool FPSCameraManipulator::intersect(const osg::Vec3d start, const
    osg::Vec3d end, double& distance) {
00264         if ((start - end).length() < 1e-8) return false; //avoids termination of program, if start == end
00265         osg::ref_ptr<osgUtil::LineSegmentIntersector> intersector = new osgUtil::LineSegmentIntersector(
    start, end);
00266         intersector->setIntersectionLimit(osgUtil::Intersector::LIMIT_NEAREST);
00267         osgUtil::IntersectionVisitor iv(intersector);
00268 
00269         iv.setTraversalMask(collisionMask);
00270         _node->accept(iv);
00271 
00272         if (intersector->containsIntersections()) {
00273             auto intersection = intersector->getIntersections().begin();
00274             distance = (start - intersection->getWorldIntersectPoint()).length();
00275             return true;
00276         }//if (intersector->containsIntersections()
00277         return false;
00278     }//intersect()
00279 
00280     double FPSCameraManipulator::getMovementSpeed() const {
00281         return _movementSpeed;
00282     }
00283 
00284     FPSCameraManipulator&
    FPSCameraManipulator::setMovementSpeed(double val) {
00285         _movementSpeed = val;
00286         return *this;
00287     }
```

```
00288
00289     double FPSCameraManipulator::getZHeight() const {
00290         return _zHeight;
00291     }
00292
00293     FPSCameraManipulator& FPSCameraManipulator::setZHeight
     (double val) {
00294         _zHeight = val;
00295         return *this;
00296     }
00297
00298     FPSCameraManipulator&
      FPSCameraManipulator::setJumpHeight(double val) {
00299         _jumpHeight = val;
00300         return *this;
00301     }
00302
00303     double FPSCameraManipulator::getJumpHeight() const {
00304         return _jumpHeight;
00305     }
00306
00307     bool FPSCameraManipulator::performMovementLeftMouseButton
     (const double eventTimeDelta, const double dx, const double dy) {
00308         return false;
00309     }
00310
00311     bool FPSCameraManipulator::handleMouseWheel(const
     osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& us) {
00312         return false;
00313     }
00314
00315 }
```

9.17 Camera/WeaponHUD.cpp File Reference

```
#include "../header/WeaponHUD.h"
#include "../header/CelShading.h"
#include "../header/ModifyMaterialVisitor.h"
#include "../header/UtilFunctions.h"
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.18 WeaponHUD.cpp

```
00001 #include "../header/WeaponHUD.h"
00002 #include "../header/CelShading.h"
00003 #include "../header/ModifyMaterialVisitor.h"
00004 #include "../header/UtilFunctions.h"
00005
00006
00007 using namespace osg;
00008
00009 namespace brtr {
00010
00011     WeaponHUD::WeaponHUD(const WeaponHUD& copy,const CopyOp& copyop)
00012     : Camera(copy, copyop)
00013     {}
00014
00015     /*
00016      * The constructor initializes the WeaponHUD with a standard weapon (in this case it's a crowbar)
00017      */
00018     WeaponHUD::WeaponHUD(){
00019         createWeaponHUD();
00020     }
00021
00022     void WeaponHUD::createWeaponHUD() {
00023
00024         ref_ptr<Node> crow = osgDB::readNodeFile("../BlenderFiles/exports/Brecheisen.ive");
```

```
00025     /*
00026      * here a transformation is applied to the crowbar to show
00027      * it at the lower right of the screen as if it is held in
00028      * the right hand
00029      */
00030         ref_ptr<MatrixTransform> crowTransform = new MatrixTransform;
00031         crowTransform->setMatrix(
00032          Matrix::rotate(osg::DegreesToRadians(3.0f), Y_AXIS)
00033        * Matrix::rotate(DegreesToRadians(190.0f), Z_AXIS)
00034        * Matrix::translate(osg::Vec3(1.0f, 1.5f, -1.5f))
00035                                 );
00036         crowTransform->addChild(crow);
00037     /*
00038      * adjusting the shininess to give the crowbar a good appearance
00039      */
00040         brtr::ModifyMaterialVisitor mmv;
00041         mmv.setShininess(42).setSpecular(Vec4(0.4, 0.4, 0.4, 1));
00042         crow->accept(mmv);
00043     /*
00044      * Adding a weapon switch to enable multiple weapon use
00045      */
00046         ref_ptr<Switch> switcher = new Switch;
00047         switcher->addChild(crowTransform, true);
00048         _switcher = switcher;
00049         _handler = new WeaponSwitchHandler(switcher);
00050
00051         ref_ptr<brtr::CelShading> celshade = new brtr::CelShading;
00052         celshade->addChild(switcher);
00053         celshade->setNodeMask(~brtr::interactionAndCollisionMask);
00054
00055         setClearMask(GL_DEPTH_BUFFER_BIT);
00056         setRenderOrder(Camera::POST_RENDER);
00057         setReferenceFrame(Camera::ABSOLUTE_RF);
00058         setProjectionMatrixAsPerspective(100, 1, 0.001, 5);
00059
00060         setViewMatrixAsLookAt(Vec3(), Vec3(0,1,0), Z_AXIS);
00061         getOrCreateStateSet()->setMode(GL_LIGHTING, StateAttribute::OFF | StateAttribute::PROTECTED |
      StateAttribute::OVERRIDE);
00062         addChild(celshade);
00063     }
00064
00065     WeaponHUD::~WeaponHUD() {
00066
00067     }
00068
00069     ref_ptr<WeaponHUD::WeaponSwitchHandler> WeaponHUD::getWeaponHandler() {
00070         return _handler;
00071     }
00072
00073     void WeaponHUD::addPortalGun() {
00074         ref_ptr<Node> portalGun = osgDB::readNodeFile("../BlenderFiles/exports/Portalgun.ive");
00075     /*
00076      * rotating and translating the portal gun to the lower
00077      * right of the screen
00078      */
00079         ref_ptr<MatrixTransform> portalGunTransform = new MatrixTransform;
00080         portalGunTransform->setMatrix(
00081             Matrix::rotate(osg::DegreesToRadians(0.0), X_AXIS)
00082             *Matrix::rotate(DegreesToRadians(0.0), Y_AXIS)
00083             *Matrix::rotate(DegreesToRadians(90.0), Z_AXIS)
00084             *Matrix::translate(osg::Vec3(0.7f, 1.5f, -1.3f))
00085             );
00086         portalGunTransform->addChild(portalGun);
00087         /*
00088      * Set proper lighting and reflection
00089      */
00090     brtr::ModifyMaterialVisitor mmv;
00091         mmv.setAmbient(Vec4(1.3, 1.3, 1.3, 1)).setShininess(42).
      setSpecular(Vec4(0.4, 0.4, 0.4, 1));
00092         portalGun->accept(mmv);
00093         _switcher->setAllChildrenOff();
00094         _switcher->addChild(portalGunTransform, true);
00095
00096
00097     }
00098
00099 //WEAPON_SWITCH_HANDLER
00100
00101     WeaponHUD::WeaponSwitchHandler::WeaponSwitchHandler(
      Switch* switchNode) :
00102         _switch(switchNode),
00103         _curWeapon(0),
00104         _frameNumber(0){}
00105
00106     bool WeaponHUD::WeaponSwitchHandler::handle(const
      osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& aa) {
00107         switch (ea.getEventType()) {
```

```
00108         case osgGA::GUIEventAdapter::SCROLL:
00109         {
00110       /*
00111        * catches multiple activations of the weapon switching
00112        * per frame because linux triggers keys twice
00113        */
00114             if (_frameNumber == aa.asView()->getFrameStamp()->getFrameNumber()) {
00115                 return false;
00116             }
00117             _frameNumber = aa.asView()->getFrameStamp()->getFrameNumber();
00118         //Debug message
00119             OSG_ALWAYS << "SCROLL: " << aa.asView()->getFrameStamp()->getFrameNumber() << std::endl;
00120             osgGA::GUIEventAdapter::ScrollingMotion sm = ea.getScrollingMotion();
00121             switch (sm) {
00122             /*
00123          * switches backwards through weapon list when
00124          * the mouse wheel is scrolled down
00125          */
00126             case osgGA::GUIEventAdapter::SCROLL_DOWN:
00127         //Debug message
00128           OSG_ALWAYS << "SCROLL DOWN" << std::endl;
00129         _switch->setValue(_curWeapon, false);
00130                 _curWeapon--;
00131                 /*
00132          * if the first weapon is selected and another
00133          * scroll backwards is triggered, the last weapon
00134          * gets selected
00135          */
00136         if (_curWeapon < 0)
00137                     _curWeapon = _switch->getNumChildren() - 1;
00138                 _curWeapon = _curWeapon % _switch->getNumChildren();
00139                 _switch->setValue(_curWeapon, true);
00140                 return true;
00141         /*
00142          * switches forward through weapon list when
00143          * the mouse wheel is scrolled up
00144          */
00145         case osgGA::GUIEventAdapter::SCROLL_UP:
00146           //Debug message
00147                 OSG_ALWAYS << "SCROLL UP" << std::endl;
00148                 _switch->setValue(_curWeapon, false);
00149                 _curWeapon++;
00150                 _curWeapon = _curWeapon % _switch->getNumChildren();
00151                 _switch->setValue(_curWeapon, true);
00152                 return true;
00153             default:
00154                 return false;
00155             }
00156         }
00157         default:
00158             return false;
00159         }
00160     }
00161
00162 }
```

9.19 GUI/KeyHandler.cpp File Reference

```
#include "../header/KeyHandler.h"
#include "../header/UtilFunctions.h"
#include <osgUtil/CullVisitor>
#include <osgUtil/LineSegmentIntersector>
#include <osg/ValueObject>
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.20 KeyHandler.cpp

```
00001 #include "../header/KeyHandler.h"
00002 #include "../header/UtilFunctions.h"
00003 #include <osgUtil/CullVisitor>
00004 #include <osgUtil/LineSegmentIntersector>
00005 #include <osg/ValueObject>
00006
00007 namespace brtr {
00008     KeyHandler::KeyHandler(osg::Node* rootNode, osg::Camera* postProcessCam,
      std::vector<osg::ref_ptr<osg::Program>> programs) :
00009         _programs(programs),
00010         _postProcessCam(postProcessCam),
00011         _rootNode(rootNode),
00012         _isWireFrame(false),
00013         _curProg(0){
00014         _wireFrameMode = new osg::PolygonMode(osg::PolygonMode::FRONT_AND_BACK,
      osg::PolygonMode::LINE);
00015         _normaleMode = new osg::PolygonMode(osg::PolygonMode::FRONT_AND_BACK,
      osg::PolygonMode::FILL);
00016     }
00017
00018
00019     KeyHandler::~KeyHandler() {}
00020
00021     bool KeyHandler::handle(const osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& aa
      ) {
00022         switch (ea.getEventType()) {
00023         case osgGA::GUIEventAdapter::FRAME:
00024             mouseIntersection(aa);
00025             //return false;
00026         case osgGA::GUIEventAdapter::MOVE:
00027         case osgGA::GUIEventAdapter::DRAG:
00028         case osgGA::GUIEventAdapter::RELEASE:
00029             _mouseEvent = &ea;
00030             return false;
00031         case osgGA::GUIEventAdapter::PUSH:
00032             _mouseEvent = &ea;
00033             if (ea.getButton() == osgGA::GUIEventAdapter::LEFT_MOUSE_BUTTON) {
00034                 brtr::BaseInteractionCallback* callback =
      modifyText(false);
00035                 if (callback) {
00036                     OSG_NOTICE << "Attaching callback" << std::endl;
00037                     if (callback->getNode()->getUpdateCallback() == callback)
00038                         callback->reactivate();
00039                     else
00040                         callback->getNode()->addUpdateCallback(callback);
00041                 }// callback
00042
00043                 return true;
00044             }
00045             return false;
00046         case osgGA::GUIEventAdapter::KEYDOWN:
00047             return handleKeyDown(ea, aa);
00048         default:
00049             return false;
00050         }
00051     }//hanlde()
00052
00053     bool KeyHandler::handleKeyDown(const osgGA::GUIEventAdapter& ea,
      osgGA::GUIActionAdapter& aa) {
00054         switch (ea.getUnmodifiedKey()) {
00055             //Turn PolygonMode on/off
00056         case osgGA::GUIEventAdapter::KEY_C: {
00057             osg::PolygonMode* curMode = _isWireFrame ? _normaleMode :
      _wireFrameMode;
00058             _rootNode->getOrCreateStateSet()->setAttributeAndModes(curMode);
00059             _isWireFrame = !_isWireFrame;
00060             return true;
00061         }//case KEY_C
00062         case osgGA::GUIEventAdapter::KEY_1:{
00063             if (ea.getModKeyMask() == osgGA::GUIEventAdapter::MODKEY_LEFT_SHIFT) {
00064                 _postProcessCam->getOrCreateStateSet()->removeAttribute(
      _programs[_curProg]);
00065                 _curProg++;
00066                 _curProg = _curProg % _programs.size();
00067                 _postProcessCam->getOrCreateStateSet()->setAttributeAndModes(
      _programs[_curProg], osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON);
00068                 return true;
00069             }
00070         }   //KEY_1
00071         default:
00072             return false;
00073         }//switch
00074     }//if(KEYDOWN)
00075
```

```
00076     void KeyHandler::mouseIntersection(osgGA::GUIActionAdapter& aa) {
00077         osg::ref_ptr<osg::Camera> camera = aa.asView()->getCamera();
00078         if (!_mouseEvent || !camera)
00079             return;
00080 
00081         osg::Vec3d eyeInWorld = osg::Vec3d() *osg::Matrixd::inverse(camera->getViewMatrix());
00082         osg::ref_ptr<osgUtil::LineSegmentIntersector> lIntersector =
00083             new osgUtil::LineSegmentIntersector(osgUtil::Intersector::WINDOW,
    _mouseEvent->getX(), _mouseEvent->getY());
00084         lIntersector->setIntersectionLimit(osgUtil::Intersector::LIMIT_NEAREST);
00085         osgUtil::IntersectionVisitor iv(lIntersector);
00086         iv.setTraversalMask(interactionMask);
00087         camera->accept(iv);
00088         if (lIntersector->containsIntersections()) {
00089             auto intersection = lIntersector->getIntersections().begin();
00090             double curDistance = (eyeInWorld - intersection->getWorldIntersectPoint()).length();
00091             if (curDistance < 4.5 ) {
00092                 _curDrawable = intersection->drawable;
00093                 modifyText(true);
00094             }//if (curDistance < distance)
00095             else {//not the right distance, remove Text & drawable, if any were present
00096                 modifyText(false);
00097             }//else
00098         }//if (intersector->containsIntersections()
00099     }//intersect()
00100 
00101     brtr::BaseInteractionCallback*
    KeyHandler::modifyText(bool show) {
00102         brtr::BaseInteractionCallback* callback =nullptr;
00103         if (_curDrawable) {
00104             OSG_NOTICE << "Checking Container..." << std::endl;
00105             osg::UserDataContainer* container = _curDrawable->getUserDataContainer();
00106             if (container) {
00107                 OSG_NOTICE << "Container True" << std::endl;
00108                 callback = dynamic_cast<brtr::BaseInteractionCallback*>(
    container->getUserObject(0));
00109                 if (callback) {
00110                     if (show) callback->setText();
00111                     else callback->clearText();
00112                 }// callback
00113             }//container
00114         }//_curDrawable
00115         if (!show)
00116             _curDrawable = nullptr;
00117         return callback;
00118     }//modifyText()
00119 
00120 }//namespace
```

9.21 header/AddInteractionCallbackToDrawableVisitor.h File Reference

```
#include <osg/NodeVisitor>
#include <../header/BaseInteractionCallback.h>
#include <osg/ValueObject>
```

Classes

- class brtr::AddInteractionCallbackToDrawableVisitor

  NodeVisitor for batch replacing all UserDataContainer of all Drawables.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.22 AddInteractionCallbackToDrawableVisitor.h

```
00001 #pragma once
```

```
00002 #include <osg/NodeVisitor>
00003 #include <../header/BaseInteractionCallback.h>
00004 #include <osg/ValueObject>
00005
00006 namespace brtr{
00017     class AddInteractionCallbackToDrawableVisitor : public
      osg::NodeVisitor {
00018     public:
00025         AddInteractionCallbackToDrawableVisitor(
      brtr::BaseInteractionCallback* callbackToAdd);
00026         virtual void apply(osg::Geode& geode);
00027     private:
00028         osg::ref_ptr<osg::DefaultUserDataContainer> _containerToAdd;
00029     };
00030 }
```

9.23 header/AddPortalGunInteractionCallback.h File Reference

```
#include <osg/NodeCallback>
#include <osgViewer/Viewer>
#include "../header/BaseInteractionCallback.h"
```

Classes

- class brtr::AddPortalGunInteractionCallback

  InteractionCallback for adding the portal gun to the players inventar.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.24 AddPortalGunInteractionCallback.h

```
00001 #pragma once
00002 #include <osg/NodeCallback>
00003 #include <osgViewer/Viewer>
00004 #include "../header/BaseInteractionCallback.h"
00005
00006 namespace brtr {
00014     class AddPortalGunInteractionCallback : public
      BaseInteractionCallback {
00015     public:
00026         AddPortalGunInteractionCallback(osg::Node* weaponHUD, osg::Camera*
      hudCam, osg::Switch* switcher, int width, int height);
00027         //doc in parent
00028         virtual void setText();
00029     protected:
00030         virtual void interact(osg::Node*, osg::NodeVisitor*);
00031     private:
00032         osg::ref_ptr<osg::Switch> _switcher;
00033     };
00034 }
```

9.25 header/AnimationCreater.h File Reference

```
#include <osgViewer/Viewer>
#include <osg/AnimationPath>
```

Classes

- class AnimationCreator

9.26 AnimationCreater.h

```
00001 #include <osgViewer/Viewer>
00002 #include <osg/AnimationPath>
00003
00004
00005     class AnimationCreator {
00013     public:
00024         double getAngleRad(osg::Vec3 pointA, osg::Vec3 pointB);
00033         osg::AnimationPath* createAnimationPath(float time);
00034     private:
00035     };
```

9.27 header/BaseInteractionCallback.h File Reference

```
#include <osg/NodeCallback>
#include <osgViewer/Viewer>
#include <osgText/Text>
```

Classes

- class brtr::BaseInteractionCallback

  This is the TemplateClass for InteractionCallbacks.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.28 BaseInteractionCallback.h

```
00001 #pragma once
00002 #include <osg/NodeCallback>
00003 #include <osgViewer/Viewer>
00004 #include <osgText/Text>
00005
00006 namespace brtr {
00024     class BaseInteractionCallback: public osg::NodeCallback {
00025     public:
00026
00035         BaseInteractionCallback(osg::Node* attachTo, osg::Camera* hudCam, int width,
      int height);
00036         virtual void operator()(osg::Node* node, osg::NodeVisitor* nv);
00037
00041         virtual void setText() = 0;
00042         void clearText();
00043         void reactivate();
00044         osg::ref_ptr<osg::Node> getNode() const;
00045         void setNode(osg::ref_ptr<osg::Node> val);
00046     protected:
00050         virtual void interact(osg::Node*, osg::NodeVisitor*)=0;
00051         osg::ref_ptr<osg::Node> _attachTo;
00052         osg::ref_ptr<osg::Camera> _hudCam;
00053         bool _done;
00054         osg::ref_ptr<osgText::Text> _text;
00055
00056
00057     private:
00058     };
00059 }
```

9.29 header/Bench.h File Reference

```
#include <osg/ShapeDrawable>
#include <osg/Geometry>
#include <osg/Material>
#include <osg/BlendFunc>
#include <osgDB/ReadFile>
#include <osg/PositionAttitudeTransform>
#include <osg/MatrixTransform>
#include <osg/Texture2D>
#include <osg/ComputeBoundsVisitor>
```

Classes

- class brtr::Bench

  Bench class, creates a bench Object.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.30 Bench.h

```
00001 #include <osg/ShapeDrawable>
00002 #include <osg/Geometry>
00003 #include <osg/Material>
00004 #include <osg/BlendFunc>
00005 #include <osgDB/ReadFile>
00006 #include <osg/PositionAttitudeTransform>
00007 #include <osg/MatrixTransform>
00008 #include <osg/Texture2D>
00009 #include <osg/ComputeBoundsVisitor>
00010
00011 using namespace osg;
00012
00013 namespace brtr {
00024     class Bench : public PositionAttitudeTransform{
00025
00026
00027       public:
00028
00029           Bench(const Vec3& pcenter = Vec3(0, 0, 0), const double plength = 8);
00030
00038           ref_ptr<PositionAttitudeTransform> getHitbox(const double alpha, double height = 8);
00039
00040           Bench(const Bench&, const CopyOp& copyop = CopyOp::SHALLOW_COPY);
00041           ~Bench();
00042
00043       private:
00044
00050       void initBench(const double plength);
00051
00055       ref_ptr<Material> createIronMaterial();
00056
00060       ref_ptr<Material> createWoodMaterial();
00061
00062       ref_ptr<Group> createLeg();
00063
00064       ref_ptr<Group> createBar();
00065
00066
00072       ref_ptr<Group> createSeat(const double width);
00073
00082       ref_ptr<Group> createArmrest(double radius, double width, double length, double totalwidth);
00083
00092       ref_ptr<Geometry> createArmrestSidesFrontBack(double radius, double width, int lsteps, int wsteps,
      bool flip = true);
```

```
00093
00102       ref_ptr<Geometry> createArmrestSidesLeftRight(double length, double width, int lsteps, int wsteps,
      bool flip = true);
00103
00104
00105       Vec3 center;
00106
00107       double length;
00108       ref_ptr<Group>  bench;
00118       ref_ptr<DrawElementsUInt> getPrimitiveSetforARectangle(int lsteps, int wsteps);
00119     };
00120
00121
00122
00123
00124
00125 }
```

9.31 header/CelShading.h File Reference

```
#include <osgFX/Export>
#include <osgFX/Effect>
#include <osg/Material>
#include <osg/LineWidth>
```

Classes

- class brtr::CelShading

  CelSading Effect, every child of this node will get the effect.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.32 CelShading.h

```
00001 #pragma once
00002 #include <osgFX/Export>
00003 #include <osgFX/Effect>
00004 #include <osg/Material>
00005 #include <osg/LineWidth>
00006 namespace brtr {
00018     class CelShading : public osgFX::Effect {
00019     public:
00026         CelShading(bool secondPass = true, std::string vertSource = "celShader.vert");
00027         CelShading(const CelShading& copy, const osg::CopyOp& copyop =
      osg::CopyOp::SHALLOW_COPY);
00028
00029         META_Effect(
00030             null,
00031             CelShading,
00032
00033             "CelShading",
00034
00035             "This effect implements a technique called 'Cel-Shading' to produce a "
00036             "cartoon-style (non photorealistic) rendering. Two passes are required: "
00037             "the first one draws solid surfaces, the second one draws the outlines. "
00038             "Vertices Shader, Toon Texture pass can be customize upon creating."
00039              ,
00040             "Marco Jez; OGLSL port by Mike Weiblen, adaptions by Gleb Ostrowski ");
00041
00042     protected:
00043         virtual ~CelShading() {}
00044
00045         bool define_techniques();
00046
```

```
00047     private:
00048         osg::ref_ptr<osg::Material> _material;
00049         osg::ref_ptr<osg::LineWidth> _lineWidth;
00050         bool _secondPass;
00051         std::string _vertSource;
00052     };
00053 }
00054
```

9.33 header/ControlRoom.h File Reference

```
#include <osg/PositionAttitudeTransform>
#include <osg/Group>
#include <osg/Material>
#include "../header/ToonTexSwitcherCallback.h"
#include "../header/ProgramSwitcherCallback.h"
```

Classes

- class brtr::ControlRoom

  Control Room Class, derived from PositionAttitudeTransform, set ups the whole room as its own children.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.34 ControlRoom.h

```
00001 #pragma once
00002 #include <osg/PositionAttitudeTransform>
00003 #include <osg/Group>
00004 #include <osg/Material>
00005 #include "../header/ToonTexSwitcherCallback.h"
00006 #include "../header/ProgramSwitcherCallback.h"
00007
00008 namespace brtr{
00018     class ControlRoom : public osg::PositionAttitudeTransform {
00019     public:
00028         ControlRoom(double roomSize, int lod,
      brtr::ToonTexSwitcherCallback& toonCallback,
      brtr::ProgramSwitcherCallback& programCallback );
00029
00030     protected:
00031         ~ControlRoom() {}
00032     private:
00033         osg::ref_ptr<osg::Group> createRoomSurrounding(double roomSize, int lod);
00034         osg::ref_ptr<osg::Group> createChessFigures(
      brtr::ToonTexSwitcherCallback& toonCallback,
      brtr::ProgramSwitcherCallback& programCallback);
00035         osg::ref_ptr<osg::Material> createMaterial(osg::Vec4 diffuse, osg::Vec4 ambient,
      osg::Vec4 specular = osg::Vec4(0.7,0.7,0.7,1), double shininess = 42.0);
00036     };
00037 }
```

9.35 header/DrunkenInteractionCallback.h File Reference

```
#include <osg/NodeCallback>
#include <osgAnimation/EaseMotion>
#include <osgViewer/Viewer>
#include "../header/BaseInteractionCallback.h"
```

Classes

- class brtr::DrunkenInteractionCallback

  Callback for the drunk effect.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.36 DrunkenInteractionCallback.h

```
00001 #pragma once
00002 #include <osg/NodeCallback>
00003 #include <osgAnimation/EaseMotion>
00004 #include <osgViewer/Viewer>
00005 #include "../header/BaseInteractionCallback.h"
00006
00007 namespace brtr {
00015     class DrunkenInteractionCallback : public
      BaseInteractionCallback {
00016     public:
00017
00027         DrunkenInteractionCallback(osg::Node* camera, osg::Camera* hudCam,
      osg::Switch* geometrySwitch, int width, int height);
00028
00029         virtual void setText();
00030     protected:
00037         virtual void interact(osg::Node*, osg::NodeVisitor*);
00038     private:
00039         int _startTime;
00040         osg::ref_ptr<osg::Switch> _geometrySwitch;
00041         osg::ref_ptr<osgAnimation::LinearMotion> _motion;
00042         bool _backwards;
00043     };
00044 }
```

9.37 header/FPSCameraManipulator.h File Reference

```
#include <osgGA/FirstPersonManipulator>
```

Classes

- class brtr::FPSCameraManipulator

  A FPS style CameraManipulator with ground clamping and intersection.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.38 FPSCameraManipulator.h

```
00001 #pragma once
00002 #include <osgGA/FirstPersonManipulator>
00003 namespace brtr {
00033     class FPSCameraManipulator :
00034         public osgGA::FirstPersonManipulator {
00035     public:
00043         FPSCameraManipulator(double movementSpeed, double zHeight, osg::Node* root,
      bool flightMode = false);
00044         double getMovementSpeed() const;
00045         FPSCameraManipulator& setMovementSpeed(double val);
00046         double getZHeight() const;
00047         FPSCameraManipulator& setZHeight(double val);
00048         double getJumpHeight() const;
00049         FPSCameraManipulator& setJumpHeight(double val);
00050 
00051     protected:
00052         ~FPSCameraManipulator();
00063         virtual bool handleMouseMove(const osgGA::GUIEventAdapter& ea,
      osgGA::GUIActionAdapter& us) ;
00074         virtual bool handleFrame(const osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& us)
      ;
00083         virtual bool handleKeyDown(const osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter&
      us) ;
00092         virtual bool handleKeyUp(const osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& us)
      ;
00102         virtual bool performMovement() ;
00103 
00104 
00105         //Just  to kill the implementation
00106         virtual bool performMovementLeftMouseButton(const double
      eventTimeDelta, const double dx, const double dy) ;
00107        // virtual bool performMovementRightMouseButton(const double eventTimeDelta, const double dx, const
       double dy) ; //remove?
00108        // virtual bool performMovementMiddleMouseButton(const double eventTimeDelta, const double dx, const
       double dy) ; //remove?
00109         virtual bool handleMouseWheel(const osgGA::GUIEventAdapter& ea,
      osgGA::GUIActionAdapter& us) ;
00110 
00111     private:
00124         bool performEyeMovement();
00136         bool intersect(const osg::Vec3d start, const osg::Vec3d end, double& distance);
00148         bool groundIntersection(osg::Vec3d& newEye);
00149 
00150         osg::ref_ptr<osg::PositionAttitudeTransform> _body;
00151         bool _flightMode;
00152         bool _forwardMovement;
00153         bool _backwardMovement;
00154         bool _leftMovement;
00155         bool _rightMovement;
00156         bool _upMovement;
00157         bool _downMovement;
00158         bool _attachBody;
00159         bool _shift;
00160         bool _ctrl;
00161         bool _jumpingUp;
00162         bool _jumpingDown;
00163         bool _crouch;
00164         double _maxFallHeight;
00165         double _movementSpeed;
00166         double _zHeight;
00167         double _savedzHeight;
00168         double _intensity;
00169         double _frameFactor;
00170         double _bodyLength;
00171         double _jumpHeight;
00172         double _savedzHeightCrouch;
00173         };
00174 }
00175 
```

9.39 header/GeometryPlacerVisitor.h File Reference

```
#include <osgViewer/Viewer>
```

Classes

- class brtr::GeometryPlacerVisitor

  NodeVisitor for batch replacing all Geometry in all visited Geodes.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.40 GeometryPlacerVisitor.h

```
00001 #pragma once
00002 #include <osgViewer/Viewer>
00003 namespace brtr {
00015     class GeometryPlacerVisitor : public osg::NodeVisitor {
00016     public:
00023         GeometryPlacerVisitor(osg::Geometry* geometryToPlace);
00024
00029         virtual void apply(osg::Geode& geode);
00030
00031         osg::ref_ptr<osg::Geometry> getGeometryToPlace() const;
00032         void setGeometryToPlace(osg::ref_ptr<osg::Geometry> val);
00033     private:
00034         osg::ref_ptr<osg::Geometry> _geometryToPlace;
00035     };
00036 }
```

9.41 header/KeyHandler.h File Reference

```
#include <osgGA/GUIEventHandler>
#include <osgViewer/Viewer>
#include <osg/PolygonMode>
#include <osg/Program>
#include "../header/FPSCameraManipulator.h"
#include "../header/BaseInteractionCallback.h"
```

Classes

- class brtr::KeyHandler

  Key Handler Class, handles all of our KeyFunctions, which do not belong to camera control (this are handled by FPSCameraManipulator)

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.42 KeyHandler.h

```
00001 #pragma once
00002 #include <osgGA/GUIEventHandler>
00003 #include <osgViewer/Viewer>
00004 #include <osg/PolygonMode>
00005 #include <osg/Program>
00006 #include "../header/FPSCameraManipulator.h"
```

```
00007 #include "../header/BaseInteractionCallback.h"
00008 namespace brtr {
00023     class KeyHandler :
00024         public osgGA::GUIEventHandler {
00025     public:
00033         KeyHandler(osg::Node*, osg::Camera* postProcessCam, std::vector<
      osg::ref_ptr<osg::Program>> programs);
00034         virtual bool handle(const osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& aa);
00035
00036     protected:
00037         ~KeyHandler();
00038     private:
00039         bool handleKeyDown(const osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& aa);
00045         void mouseIntersection(osgGA::GUIActionAdapter& aa);
00052         brtr::BaseInteractionCallback* modifyText(bool show);
00053         osg::ref_ptr<osg::Drawable> _curDrawable;
00054         osg::ref_ptr<osg::Node> _rootNode;
00055         osg::ref_ptr<osg::PolygonMode> _wireFrameMode;
00056         osg::ref_ptr<osg::PolygonMode> _normaleMode;
00057         osg::ref_ptr<osg::Camera> _postProcessCam;
00058         std::vector<osg::ref_ptr<osg::Program>> _programs;
00059         osg::ref_ptr< const osgGA::GUIEventAdapter > _mouseEvent;
00060         bool _isWireFrame;
00061         unsigned int _curProg;
00062     };
00063 }
00064
```

9.43 header/ModifyMaterialVisitor.h File Reference

```
#include <osg/NodeCallback>
#include <osgViewer/Viewer>
```

Classes

- class brtr::ModifyMaterialVisitor

  Visitor for altering the material attributes, mainly used for objects craeted with blender.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.44 ModifyMaterialVisitor.h

```
00001 #pragma once
00002 #include <osg/NodeCallback>
00003 #include <osgViewer/Viewer>
00004
00005 namespace brtr {
00014     class ModifyMaterialVisitor : public osg::NodeVisitor {
00015     public:
00016
00017         ModifyMaterialVisitor();
00018         void apply(osg::Geode& geode);
00019
00020         osg::Vec4 getDiffuse() const;
00021         ModifyMaterialVisitor& setDiffuse(osg::Vec4 val);
00022         osg::Vec4 getSpecular() const;
00023         ModifyMaterialVisitor& setSpecular(osg::Vec4 val);
00024         osg::Vec4 getAmbient() const;
00025         ModifyMaterialVisitor& setAmbient(osg::Vec4 val);
00026         double getShininess() const;
00027         ModifyMaterialVisitor& setShininess(double val);
00028     private:
00029         osg::Vec4 _diffuse;
00030         osg::Vec4 _specular;
00031         osg::Vec4 _ambient;
```

```
00032         double _shininess;
00033         bool _ambientFlag;
00034         bool _specularFlag;
00035         bool _shininessFlag;
00036         bool _diffuseFlag;
00037     };
00038 }
00039
00040
00041
```

9.45 header/ProgramSwitcherCallback.h File Reference

```
#include <osg/NodeCallback>
#include <osgViewer/Viewer>
#include <osg/Program>
#include "../header/BaseInteractionCallback.h"
```

Classes

- class brtr::ProgramSwitcherCallback

  Callback for switching the postprocess programs.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.46 ProgramSwitcherCallback.h

```
00001 #pragma once
00002 #include <osg/NodeCallback>
00003 #include <osgViewer/Viewer>
00004 #include <osg/Program>
00005 #include "../header/BaseInteractionCallback.h"
00006
00007 namespace brtr {
00017     class ProgramSwitcherCallback : public
      BaseInteractionCallback {
00018     public:
00028         ProgramSwitcherCallback(osg::Node* postprocessCam, osg::Camera* hudCam, int
      width, int height, std::vector<osg::ref_ptr<osg::Program>> programs);
00029         //Doc in parent
00030         virtual void setText();
00031     protected:
00038         virtual void interact(osg::Node*, osg::NodeVisitor*);
00039     private:
00040         std::vector<osg::ref_ptr<osg::Program>> _programs;
00041         unsigned int _curProg;
00042     };
00043 }
```

9.47 header/ToonTexSwitcherCallback.h File Reference

```
#include <osg/NodeCallback>
#include <osgViewer/Viewer>
#include <osg/Texture2D>
#include "../header/BaseInteractionCallback.h"
```

Classes

- class brtr::ToonTexSwitcherCallback

  Callback for switching the ToonTextures.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.48 ToonTexSwitcherCallback.h

```
00001 #pragma once
00002 #include <osg/NodeCallback>
00003 #include <osgViewer/Viewer>
00004 #include <osg/Texture2D>
00005 #include "../header/BaseInteractionCallback.h"
00006
00007 namespace brtr {
00017     class ToonTexSwitcherCallback : public
      BaseInteractionCallback {
00018     public:
00028         ToonTexSwitcherCallback(osg::Node* scenedata, osg::Camera* hudCam, int width
      , int height, std::vector<osg::ref_ptr<osg::Texture2D>> toonTexs);
00029         //docu in parent
00030         virtual void setText();
00031     protected:
00038         virtual void interact(osg::Node* node, osg::NodeVisitor*);
00039     private:
00040         int _curTex;
00041         std::vector<osg::ref_ptr<osg::Texture2D>> _toonTexs;
00042     };
00043 }
```

9.49 header/TrainSwitcherCallback.h File Reference

```
#include <osg/NodeCallback>
#include <osgViewer/Viewer>
```

Classes

- class brtr::TrainSwitcherCallback

  Callback for switching the "trains".

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.50 TrainSwitcherCallback.h

```
00001 #pragma once
00002 #include <osg/NodeCallback>
00003 #include <osgViewer/Viewer>
00004
00005 namespace brtr {
00015     class TrainSwitcherCallback : public osg::NodeCallback {
00016     public:
```

```
00017
00018         TrainSwitcherCallback();
00019         virtual void operator()(osg::Node* node, osg::NodeVisitor* nv);
00020     private:
00021         int _curActiveTrain;
00022         int _deltaTime;
00023     };
00024 }
```

9.51 header/UtilFunctions.h File Reference

```
#include <osg/Camera>
#include <osg/Geometry>
#include <osg/MatrixTransform>
#include <osg/Texture2D>
#include <osgViewer/Viewer>
#include <osg/PositionAttitudeTransform>
#include <osgText/Text>
#include <osgDB/ReadFile>
#include <osg/Shader>
#include <osg/Material>
#include <osgParticle/ParticleSystem>
#include <cmath>
#include <functional>
```

Classes

- struct brtr::BodyOfRotationFunction

  struct holding the function, which calculates the radius in dependece of the height. lambda (double)->double func, int end, BodyOfRotationFunction nextFunc if one wish to have more then one function then the end value and nextFunc pointer must be set accordingly the end+1 is the beginning x of the next function*

- struct brtr::RenderingPipeline

  struct holding the camera for the multi-rendering passes. Also holds the program vector for the post process pass. pass0Color, pass0depth, passPostProcess, program array, count programArray The program vector is used by the KeyHandler and the InteractionItems for changing the postprocess programs

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

Macros

- #define _USE_MATH_DEFINES

Functions

- void brtr::createRenderingPipeline (unsigned int width, unsigned int height, osg::Node &rootForToon, osg↩ Viewer::Viewer &viewer, RenderingPipeline &pipe, osg::Vec3f &fogColor)

  creates the rendering pipeline

- osg::ref_ptr< osg::LightSource > brtr::createLight (const osg::Vec3 &pos, int lightNum, int point=1, double spotCutoff=180, double spotExponent=0)

  creates a Light with a lightsource

- osg::ref_ptr< osg::Camera > brtr::createRTTCamera (osg::Camera::BufferComponent buffer, osg::Texture *tex, bool isAbsolute=false)

  creates a RTTCam
- osg::ref_ptr< osg::Geode > brtr::createScreenQuad (float width, float height, float scale=1.0f)

  creates a texture-ready screen quad for postprocessing
- osg::ref_ptr< osg::Camera > brtr::createHUDCamera (double left, double right, double bottom, double top)

  creates a HUD-Cam with a 2D-orthogonal projection matrix
- osg::ref_ptr< osgText::Text > brtr::createText (const osg::Vec3 &pos, const std::string &content, float size)

  creates a (arial) text object for use with a hud camera
- osg::ref_ptr< osg::Geometry > brtr::createBodyOfRotation (double height, int hsteps, int rsteps, const BodyOfRotationFunction &function)

  Creates a body of rotation.
- osg::ref_ptr< osg::Geometry > brtr::createRectangle (double length, double width, int lsteps, int wsteps)

  Creates a Rectangle with TRIANGLE_STRIPS.
- osg::ref_ptr< osg::Geometry > brtr::createRectangleWithTexcoords (double length, double width, int lsteps, int wsteps)

  Creates a Rectangle with TRIANGLE_STRIPS.
- osg::ref_ptr< osg::Group > brtr::createCuboid (const double length, const double width, const double height, const double factor=6)

  Creates a Cubiod with TRIANGLE_STRIPS using the createRectangle function.
- osg::ref_ptr < osg::PositionAttitudeTransform > brtr::wrapInPositionAttitudeTransform (osg::Node *srcNode, const osg::Vec3d &pos)

  Return the given Node in a PositionAttitudeTransform with a given position.
- osg::ref_ptr< osg::Geometry > brtr::createBeerBottle ()

  Creates a BeerBottle with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createRealBottle ()

  Creates a Bottle with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createVase ()

  Creates a vase with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createStalk ()

  Creates a stalk with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createBud ()

  Creates a bud with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createChessFigure ()

  Creates a "ChessFigure" with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr < osg::PositionAttitudeTransform > brtr::createVaseWithFlower ()

  combines the stalk, bud and vase in a postitionAttitudetransform
- osg::ref_ptr< osg::Geode > brtr::createCrosshair (unsigned int width, unsigned int height)

  creates a crosshair in the middle of the screen
- osg::ref_ptr< osg::Texture2D > brtr::createToonTex (std::string toonTex)

  creates a Texture2D object with the given toonTex
- osg::ref_ptr< osg::Material > brtr::createSimpleMaterial (osg::Material::Face face, const osg::Vec4 &diffuse, const osg::Vec4 &ambient, const osg::Vec4 &specular, const double shininess)

  creates a simple material
- osg::Vec3 brtr::getDimensionOfNode (osg::Node *source)

  return the dimension of a node (width, height, length)

Variables

- const int brtr::collisionMask = 0x1
- const int brtr::interactionMask = 0x2
- const int brtr::interactionAndCollisionMask = collisionMask | interactionMask
- const int brtr::fakeWallMask = 0x4

9.51.1 Detailed Description

Author

Gleb Ostrowski & Marcel Felix

Definition in file UtilFunctions.h.

9.51.2 Macro Definition Documentation

9.51.2.1 #define _USE_MATH_DEFINES

Definition at line 14 of file UtilFunctions.h.

9.52 UtilFunctions.h

```
00001 #pragma once
00002
00003 #include <osg/Camera>
00004 #include <osg/Geometry>
00005 #include <osg/MatrixTransform>
00006 #include <osg/Texture2D>
00007 #include <osgViewer/Viewer>
00008 #include <osg/PositionAttitudeTransform>
00009 #include <osgText/Text>
00010 #include <osgDB/ReadFile>
00011 #include <osg/Shader>
00012 #include <osg/Material>
00013 #include <osgParticle/ParticleSystem>
00014 #define _USE_MATH_DEFINES
00015 #include <cmath>
00016 #include <functional>
00017
00026 namespace brtr {
00027     const int collisionMask = 0x1;
00028     const int interactionMask = 0x2;
00029     const int interactionAndCollisionMask =
      collisionMask | interactionMask;
00030     const int fakeWallMask = 0x4;
00031
00032
00040     struct  BodyOfRotationFunction {
00041         std::function<double(double)> func;
00042         double end;
00043         const BodyOfRotationFunction* nextFunc;
00044         double derivation(double x) const {
00045             double h = 1e-10; //very small
00046             return (func(x + h) - func(x)) / h;
00047         };
00048     };
00049
00056     struct RenderingPipeline {
00057         osg::ref_ptr<osg::Camera> pass_0_color;
00058         osg::ref_ptr<osg::Camera> pass_0_depth;
00059         osg::ref_ptr<osg::Camera> pass_PostProcess;
00060         std::vector<osg::ref_ptr<osg::Program>> programs;
00061     };
00062
00075     extern void createRenderingPipeline(unsigned int width, unsigned int height,
      osg::Node& rootForToon, osgViewer::Viewer &viewer, RenderingPipeline& pipe, osg::Vec3f&
      fogColor);
00076
00087     extern osg::ref_ptr<osg::LightSource> createLight(const osg::Vec3 &pos, int lightNum, int
      point = 1, double spotCutoff = 180, double spotExponent = 0);
```

```
00098     extern osg::ref_ptr<osg::Camera> createRTTCamera(osg::Camera::BufferComponent buffer,
      osg::Texture* tex, bool isAbsolute = false);
00109     extern osg::ref_ptr<osg::Geode> createScreenQuad(float width, float height, float scale
       = 1.0f);
00121     extern osg::ref_ptr<osg::Camera> createHUDCamera(double left, double right, double
      bottom, double top);
00132     extern osg::ref_ptr<osgText::Text> createText(const osg::Vec3& pos, const std::string&
      content, float size);
00145     extern osg::ref_ptr<osg::Geometry> createBodyOfRotation(double height, int hsteps,
      int rsteps, const BodyOfRotationFunction& function);
00157     extern osg::ref_ptr<osg::Geometry> createRectangle(double length, double width, int
      lsteps, int wsteps);
00158
00171     extern osg::ref_ptr<osg::Geometry> createRectangleWithTexcoords(double
      length, double width, int lsteps, int wsteps);
00172
00184     extern osg::ref_ptr<osg::Group> createCuboid(const double length, const double width, const
       double height, const double factor = 6);
00185
00193     extern osg::ref_ptr<osg::PositionAttitudeTransform>
      wrapInPositionAttitudeTransform(osg::Node * srcNode, const osg::Vec3d& pos);
00194
00199     extern osg::ref_ptr<osg::Geometry> createBeerBottle();
00207     extern osg::ref_ptr<osg::Geometry> createRealBottle();
00215     extern osg::ref_ptr<osg::Geometry> createVase();
00223     extern  osg::ref_ptr<osg::Geometry> createStalk();
00231     extern  osg::ref_ptr<osg::Geometry> createBud();
00239     extern osg::ref_ptr<osg::Geometry> createChessFigure();
00240
00246     extern osg::ref_ptr<osg::PositionAttitudeTransform> createVaseWithFlower();
00247
00255     extern osg::ref_ptr<osg::Geode> createCrosshair(unsigned int width, unsigned int height)
      ;
00262     extern osg::ref_ptr<osg::Texture2D> createToonTex(std::string toonTex);
00263
00275     extern osg::ref_ptr<osg::Material> createSimpleMaterial(osg::Material::Face face,
      const osg::Vec4& diffuse, const osg::Vec4& ambient, const osg::Vec4& specular, const double shininess);
00276
00277
00284     extern osg::Vec3 getDimensionOfNode(osg::Node * source);
00285 }
00286
```

9.53 header/WeaponHUD.h File Reference

```
#include <osg/Camera>
#include <osg/MatrixTransform>
#include <osg/PositionAttitudeTransform>
#include <osgGA/GUIEventHandler>
#include <osg/Switch>
```

Classes

- class brtr::WeaponHUD

  WeaponHUD class, provides the functions to add a HUD camera to the scene.
- class brtr::WeaponHUD::WeaponSwitchHandler

  EventHandler for WeaponSwitching.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.54 WeaponHUD.h

```
00001 #pragma once
```

```
00002 #include <osg/Camera>
00003 #include <osg/MatrixTransform>
00004 #include <osg/PositionAttitudeTransform>
00005 #include <osgGA/GUIEventHandler>
00006 #include <osg/Switch>
00007
00008 using namespace osg;
00009 namespace brtr {
00010
00023 class WeaponHUD : public Camera {
00032     class WeaponSwitchHandler : public osgGA::GUIEventHandler {
00033     public:
00039     WeaponSwitchHandler(Switch* switchNode);
00046         virtual bool handle(const osgGA::GUIEventAdapter& ea, osgGA::GUIActionAdapter& aa);
00047     protected:
00048
00049         ~WeaponSwitchHandler() {}
00050     private:
00051         ref_ptr<Switch> _switch;
00052         int _curWeapon;
00053         unsigned int _frameNumber;
00054
00055     };
00056
00057 public:
00058     WeaponHUD();
00059     WeaponHUD(const WeaponHUD&,const CopyOp& copyop=CopyOp::SHALLOW_COPY);
00060     ref_ptr<WeaponSwitchHandler> getWeaponHandler();
00064     void addPortalGun();
00065     ~WeaponHUD();
00066 protected:
00067 private:
00071     void createWeaponHUD();
00072     ref_ptr<Switch> _switcher;
00073     ref_ptr<WeaponSwitchHandler> _handler;
00074 };
00075 }
```

9.55 Main/Main.cpp File Reference

```
#include <osgViewer/Viewer>
#include <osg/Geometry>
#include <osgDB/ReadFile>
#include <osg/BlendFunc>
#include <osg/ValueObject>
#include <osgUtil/Optimizer>
#include <string>
#include <sstream>
#include <iostream>
#include "../header/UtilFunctions.h"
#include "../header/WeaponHUD.h"
#include "../header/FPSCameraManipulator.h"
#include "../header/GeometryPlacerVisitor.h"
#include "../header/DrunkenInteractionCallback.h"
#include "../header/ModifyMaterialVisitor.h"
#include "../header/Bench.h"
#include "../header/CelShading.h"
#include "../header/BaseInteractionCallback.h"
#include "../header/KeyHandler.h"
#include "../header/AnimationCreater.h"
#include "../header/AddPortalGunInteractionCallback.h"
#include "../header/AddInteractionCallbackToDrawableVisitor.h"
#include "../header/ControlRoom.h"
#include "../header/TrainSwitcherCallback.h"
```

Functions

- int main (void)

9.55.1 Function Documentation

9.55.1.1 int main (void)

Definition at line 30 of file Main.cpp.

9.56 Main.cpp

```
00001 #include <osgViewer/Viewer>
00002 #include <osg/Geometry>
00003 #include <osgDB/ReadFile>
00004 #include <osg/BlendFunc>
00005 #include <osg/ValueObject>
00006 #include <osgUtil/Optimizer>
00007 #include <string>
00008 #include <sstream>
00009 #include <iostream>
00010
00011 #include "../header/UtilFunctions.h"
00012 #include "../header/WeaponHUD.h"
00013 #include "../header/FPSCameraManipulator.h"
00014 #include "../header/GeometryPlacerVisitor.h"
00015 #include "../header/DrunkenInteractionCallback.h"
00016 #include "../header/ModifyMaterialVisitor.h"
00017 #include "../header/Bench.h"
00018 #include "../header/CelShading.h"
00019 #include "../header/BaseInteractionCallback.h"
00020 #include "../header/KeyHandler.h"
00021 #include "../header/AnimationCreater.h"
00022 #include "../header/AddPortalGunInteractionCallback.h"
00023 #include "../header/AddInteractionCallbackToDrawableVisitor.h"
00024 #include "../header/ControlRoom.h"
00025 #include "../header/TrainSwitcherCallback.h"
00026
00027
00028 using namespace osg;
00029
00030 int main(void){
00031     //some vars
00032     osg::setNotifyLevel(FATAL);
00033     Vec3f fogColor(.3219, 0.37, 0.3564);
00034     unsigned int width, height;
00035     unsigned int oldWidth, oldHeight;
00036     int screen = 0;  //for easy multimonitor switchning while debugging
00037     std::string inputLine = "";
00038     int choose = 0;
00039     std::vector<ref_ptr<Texture2D>> toonTexs;
00040     toonTexs.push_back(brtr::createToonTex("2d_toons_brown.png"));
00041     toonTexs.push_back(brtr::createToonTex("2d_toons_blue.png"));
00042     toonTexs.push_back(brtr::createToonTex("2d_toons_red.png"));
00043     toonTexs.push_back(brtr::createToonTex("2d_toons_violet.png"));
00044     toonTexs.push_back(brtr::createToonTex("2d_toons_yellow.png"));
00045     ref_ptr<GraphicsContext::WindowingSystemInterface> wsi = GraphicsContext::getWindowingSystemInterface()
    ;
00046
00047     OSG_ALWAYS << "Please choose the desired Display Resolution:" << std::endl;
00048     OSG_ALWAYS << "\t(1): Full HD 1920x1080 (only with a decent Graphic Card!)" << std::endl;
00049     OSG_ALWAYS << "\t(2): HD+ 1366x768 (should work with most Cards)" << std::endl;
00050     OSG_ALWAYS << "\t(3): HD 1280x720 (choose this for best performance, but worst quality)" <<std::endl;
00051     OSG_ALWAYS << "\t(4): Use Screen Resolution" <<std::endl;
00052     OSG_ALWAYS << "\t(5): quit the program without experiencing the forsaken station =(" << std::endl;
00053     std::getline(std::cin, inputLine);
00054     std::stringstream(inputLine) >> choose;
00055     while (!(choose == 1 || choose == 2 || choose == 3 || choose == 4 || choose == 5)) {
00056         OSG_ALWAYS << "Only (1), (2), (3), (4) or (5) are valid options!" <<std::endl;
00057         OSG_ALWAYS << choose << std::endl;
00058         std::getline(std::cin, inputLine);
00059         std::stringstream(inputLine) >> choose;
00060     }
00061     switch (choose) {
00062     case 1:
00063         width = 1920;
00064         height = 1080;
00065         break;
```

```
00066     case 2:
00067         width = 1366;
00068         height = 768;
00069         break;
00070     case 3:
00071         width = 1280;
00072         height = 720;
00073         break;
00074     case 4:
00075         wsi->getScreenResolution(GraphicsContext::ScreenIdentifier(screen), width, height);
00076         break;
00077     case 5:
00078         return EXIT_SUCCESS;
00079     }
00080
00081
00082     OSG_ALWAYS << "Setting some options which should help with performance (but probably do not)" <<
      std::endl;
00083     //this viewer will display our graph
00084     osgViewer::Viewer viewer;
00085     //Faster Intersection, hell yeah!
00086     osgDB::Registry::instance()->setBuildKdTreesHint(osgDB::Options::BUILD_KDTREES);
00087     //Get/Set Screen Resolution
00088     wsi->getScreenResolution(GraphicsContext::ScreenIdentifier(screen), oldWidth, oldHeight);
00089     OSG_ALWAYS << "This DisplaySettings will be used:" << std::endl;
00090     OSG_ALWAYS << width << "x" << height << std::endl;
00091     wsi->setScreenResolution(GraphicsContext::ScreenIdentifier(screen), width, height);
00092     //to make sure, we are using the right resolution, even if the set fails
00093     wsi->getScreenResolution(GraphicsContext::ScreenIdentifier(screen), width, height);
00094
00095     //Read IVEs, set Masks
00096     OSG_ALWAYS << "Reading IVE's, making cookies." << std::endl;
00097     ref_ptr<Node> trainStation = osgDB::readNodeFile("../BlenderFiles/exports/BrainTrain6_1p25E_Lights.ive"
      );
00098     trainStation->setNodeMask(brtr::collisionMask);
00099     ref_ptr<Node> trainStationHitbox = osgDB::readNodeFile("
      ../BlenderFiles/exports/BrainTrain6_1p25E_Lights_Hitbox.osgt");
00100     trainStationHitbox->setNodeMask(brtr::collisionMask);
00101     ref_ptr<Node> bottleEmitter = osgDB::readNodeFile("
      ../BlenderFiles/exports/BrainTrain_BottleParticles.osgt");
00102     bottleEmitter->setNodeMask(~brtr::interactionAndCollisionMask);
00103     ref_ptr<Node> drinkablebottleEmitter = osgDB::readNodeFile("
      ../BlenderFiles/exports/BrainTrain_BottleParticlesDrinkable.osgt");
00104     drinkablebottleEmitter->setNodeMask(brtr::interactionMask);
00105     ref_ptr<Node> trainModel = osgDB::readNodeFile("../BlenderFiles/exports/Train.ive.0,0,-48.rot");
00106     ref_ptr<Node> portalGunTrain = osgDB::readNodeFile("
      ../BlenderFiles/exports/Portalgun_Big.ive.0,0,-48.rot");
00107     //Position "Trains"
00108     ref_ptr<PositionAttitudeTransform> trainPosition = new PositionAttitudeTransform;
00109     trainPosition->setNodeMask(brtr::collisionMask);
00110     trainPosition->setPosition(Vec3(0, 0, -20));
00111     trainPosition->addChild(trainModel);
00112     trainPosition->setDataVariance(Object::DYNAMIC);
00113     ref_ptr<PositionAttitudeTransform> portalGuntrainPosition = new PositionAttitudeTransform;
00114     portalGuntrainPosition->setNodeMask(~brtr::interactionAndCollisionMask
      );
00115     portalGuntrainPosition->setPosition(Vec3(0, 0, -20));
00116     portalGuntrainPosition->addChild(portalGunTrain);
00117     portalGuntrainPosition->setDataVariance(Object::DYNAMIC);
00118
00119     //Animation for Train
00120     ref_ptr<AnimationPath> trainPath = AnimationCreator().
      createAnimationPath(0.1f);
00121     osg::ref_ptr<osg::AnimationPathCallback> trainAniCallback = new  osg::AnimationPathCallback;
00122     trainAniCallback->setAnimationPath(trainPath);
00123     trainPosition->setUpdateCallback(trainAniCallback);
00124     portalGuntrainPosition->setUpdateCallback(trainAniCallback);
00125
00126     //Switch for trains
00127     ref_ptr<Switch> train = new Switch;
00128     train->addChild(trainPosition, true);
00129     train->addChild(portalGuntrainPosition, false);
00130     train->addUpdateCallback(new brtr::TrainSwitcherCallback);
00131
00132     ref_ptr<Node> ponyFlagSourceNode = osgDB::readNodeFile("../BlenderFiles/exports/BrainTrain_Flag.ive");
00133     ref_ptr<brtr::CelShading> ponyFlag = new brtr::CelShading(false);
00134     ponyFlag->addChild(ponyFlagSourceNode);
00135     //let the flag move!
00136     ponyFlag->getOrCreateStateSet()->addUniform(new Uniform("zAnimation",true), StateAttribute::ON |
      StateAttribute::OVERRIDE);
00137     ref_ptr<Node> portalGunSource = osgDB::readNodeFile("../BlenderFiles/exports/Portalgun.ive");
00138     ref_ptr<PositionAttitudeTransform> portalGunPlacer = new PositionAttitudeTransform;
00139     portalGunPlacer->addChild(portalGunSource);
00140     portalGunPlacer->setPosition(Vec3(-76.54, 5.28, 3.82));
00141     //vase on top of the ticketcorner
00142     ref_ptr<PositionAttitudeTransform> vase = brtr::createVaseWithFlower();
00143     vase->setPosition(Vec3(-27.9, 17.4, 9.7));
```

```
00144
00145     OSG_ALWAYS << "Placing bottles (and making some them drinkable)" << std::endl;
00146     OSG_ALWAYS << "Do not drink and drive" << std::endl;
00147     OSG_ALWAYS << "Actually, this drink is bad, so do not drink it at all." << std::endl;
00148     //Create and make alpha Bottle
00149
00150     ref_ptr<Geometry> bottle = brtr::createRealBottle();
00151
00152     //Drinkable bottles
00153     ref_ptr<Geometry> drinkablebottle = brtr::createRealBottle();
00154
00155     //drunk one bottle, disable all! It's a Feature, not a bug ;)
00156     ref_ptr<Switch> drinableBottleSwitch = new Switch;
00157     drinableBottleSwitch->addChild(drinkablebottleEmitter, true);
00158     drinableBottleSwitch->setNodeMask(brtr::interactionMask);
00159
00160     //portalGunPicker
00161     ref_ptr<Switch> portalGunSwitch = new Switch;
00162     portalGunSwitch->addChild(portalGunPlacer, true);
00163     portalGunSwitch->setNodeMask(brtr::interactionMask);
00164
00165     //place bottle
00166     brtr::GeometryPlacerVisitor bottlePlacer(bottle);
00167     bottleEmitter->accept(bottlePlacer);
00168     //place drinkablebottle
00169     brtr::GeometryPlacerVisitor drinkablebottlePlacer(drinkablebottle);
00170     drinkablebottleEmitter->accept(drinkablebottlePlacer);
00171
00172     //Placing Benches
00173     OSG_ALWAYS << "Placing (uncomfortable) benches." << std::endl;
00174     OSG_ALWAYS << "Lying, they are great!" << std::endl;
00175     OSG_ALWAYS << "Na, that was a lie." << std::endl;
00176
00177     ref_ptr<PositionAttitudeTransform> leftBench = new brtr::Bench(Vec3(49.5, -2.3, -0.6), 8);
00178     leftBench->setAttitude(Quat(DegreesToRadians(167.0), Z_AXIS));
00179     leftBench->setNodeMask(brtr::collisionMask);
00180
00181     ref_ptr<PositionAttitudeTransform> rightBench = new brtr::Bench(Vec3(-50, 14.3, -0.6), 14);
00182     rightBench->setAttitude(Quat(DegreesToRadians(192.7), Z_AXIS));
00183     rightBench->setNodeMask(brtr::collisionMask);
00184
00185     //a group for the whole station
00186     //needed for the createPipeLine Function
00187     ref_ptr<Group> rootForToon = new Group;
00188     rootForToon->addChild(trainStation);
00189     rootForToon->addChild(leftBench);
00190     rootForToon->addChild(rightBench);
00191     rootForToon->addChild(train);
00192     rootForToon->addChild(bottleEmitter);
00193     rootForToon->addChild(drinableBottleSwitch);
00194     rootForToon->addChild(portalGunSwitch);
00195     rootForToon->addChild(vase);
00196     //just to make sure
00197     rootForToon->setDataVariance(Object::STATIC);
00198
00199
00200     OSG_ALWAYS << "Creating Lights. Nobody wants a creepy, dark station." << std::endl;
00201     OSG_ALWAYS << "Except for the creators." << std::endl;
00202     ref_ptr<LightSource> light1 = brtr::createLight(Vec3(-76.88403, -8.27441, 20.63965), 1
      );
00203     ref_ptr<LightSource> light2 = brtr::createLight(Vec3(-26.8972, 1.97552, 20.02043), 2);
00204     ref_ptr<LightSource> light3 = brtr::createLight(Vec3(24.33239, 2.49185, 21.58063), 3);
00205     ref_ptr<LightSource> light4 = brtr::createLight(Vec3(74.73347, -8.83866, 21.33362), 4)
      ;
00206     ref_ptr<LightSource> staircaseLight = brtr::createLight(Vec3(0, 110, 38), 5);
00207     staircaseLight->getLight()->setQuadraticAttenuation(0.005);
00208
00209
00210     rootForToon->addChild(light1);
00211     rootForToon->addChild(light2);
00212     rootForToon->addChild(light3);
00213     rootForToon->addChild(light4);
00214     rootForToon->addChild(staircaseLight);
00215
00216
00217     OSG_ALWAYS << "Creating RenderingPipeline. ToonyLoony!" << std::endl;
00218     brtr::RenderingPipeline pipe;
00219     brtr::createRenderingPipeline(width, height, *rootForToon, viewer, pipe,
      fogColor);
00220
00221
00222     //HUD Cams
00223     ref_ptr<brtr::WeaponHUD> weaponHUD = new brtr::WeaponHUD;
00224     ref_ptr<Camera> textHUD = brtr::createHUDCamera(0, width, 0, height);
00225     textHUD->addChild(brtr::createCrosshair(width, height));
00226     textHUD->getOrCreateStateSet()->setTextureMode(1, GL_TEXTURE_2D, StateAttribute::OFF);
00227     //making bottles drinkable
```

```
00228     drinkablebottle->getOrCreateUserDataContainer()->addUserObject(new
      brtr::DrunkenInteractionCallback(viewer.getCamera(), textHUD,
      drinableBottleSwitch, width, height));
00229     ref_ptr<brtr::AddPortalGunInteractionCallback> portalGunCallback = new
      brtr::AddPortalGunInteractionCallback(weaponHUD, textHUD,
      portalGunSwitch, width, height);
00230     brtr::AddInteractionCallbackToDrawableVisitor
      portalGunCallbackVisitor(portalGunCallback);
00231     portalGunSource->accept(portalGunCallbackVisitor);
00232
00233     OSG_ALWAYS << "Making coffee." << std::endl;
00234     brtr::ModifyMaterialVisitor mmv;
00235     mmv.setAmbient(Vec4(0.4, 0.4, 0.4, 4)).setDiffuse(Vec4(0.7,0.7,0.7,1.0)).
      setShininess(42*3); //.setShininess(42 * 3).setSpecular(Vec4(0.7, 0.7, 0.7, 1));
00236     //weaponHUD->accept(imv);
00237     trainStation->accept(mmv);
00238     trainPosition->accept(mmv);
00239     portalGunSource->accept(mmv);
00240
00241     //the root node, which holds the cams (pass and HUDs) as siblings
00242     ref_ptr<Group> sceneData = new Group;
00243     //add elements to sceneData
00244     OSG_ALWAYS << "Adding elements to scene root." << std::endl;
00245     OSG_ALWAYS << "I am soooo excited, we are nearly done!." << std::endl;
00246     sceneData->addChild(pipe.pass_0_color);
00247     sceneData->addChild(pipe.pass_0_depth);
00248     sceneData->addChild(pipe.pass_PostProcess);
00249     sceneData->addChild(trainStationHitbox);
00250     sceneData->addChild(weaponHUD);
00251     sceneData->addChild(textHUD);
00252     //safety
00253     //sceneData->getOrCreateStateSet()->setMode(GL_LIGHTING, StateAttribute::OFF |
       StateAttribute::OVERRIDE);
00254
00255     //Set toonTex
00256     sceneData->getOrCreateStateSet()->setTextureAttributeAndModes(1, toonTexs[0], osg::StateAttribute::ON);
00257
00258     //Control Room
00259     ref_ptr<brtr::ToonTexSwitcherCallback> toonCallback = new
      brtr::ToonTexSwitcherCallback(sceneData, textHUD, width, height, toonTexs);
00260     ref_ptr<brtr::ProgramSwitcherCallback> programCallback = new
      brtr::ProgramSwitcherCallback(pipe.pass_PostProcess, textHUD,
      width, height, pipe.programs);
00261     ref_ptr<brtr::ControlRoom> controlRoom = new brtr::ControlRoom(40, 50, *toonCallback,
      *programCallback);
00262     controlRoom->setPosition(Vec3(0, 170.3, 23.2));
00263
00264     //Adding "special-treatment nodes" (mainly no outlines) to first pass
00265     pipe.pass_0_color->addChild(ponyFlag);
00266     pipe.pass_0_depth->addChild(ponyFlag);
00267     pipe.pass_0_color->addChild(controlRoom);
00268     pipe.pass_0_depth->addChild(controlRoom);
00269
00270     viewer.setSceneData(sceneData);
00271     osgUtil::Optimizer optimizer;
00272     optimizer.optimize(sceneData,osgUtil::Optimizer::STATIC_OBJECT_DETECTION);
00273
00274
00275     //Manipulator and KeyHandler
00276     OSG_ALWAYS << "Adding Manipulator and KeyHandler. What could possible go wrong?." << std::endl;
00277     viewer.setCameraManipulator(new brtr::FPSCameraManipulator(0.25, 7,
      rootForToon));
00278     osg::ref_ptr<brtr::KeyHandler> keyHandler = new brtr::KeyHandler(sceneData, pipe.
      pass_PostProcess, pipe.programs);
00279     viewer.addEventHandler(weaponHUD->getWeaponHandler());
00280     viewer.addEventHandler(keyHandler);
00281
00282     OSG_ALWAYS << "Potato." << std::endl;
00283     OSG_ALWAYS << "Finished! Press Enter to start the fun!" <<std::endl;
00284
00285     getchar();
00286     OSG_ALWAYS << "The cake is a lie." << std::endl;
00287     viewer.setUpViewOnSingleScreen(screen);
00288     osgViewer::GraphicsWindow* window = dynamic_cast<osgViewer::GraphicsWindow*>(viewer.getCamera()->
      getGraphicsContext());
00289     if (window) {
00290         window->useCursor(false);
00291     }
00292     else {
00293         OSG_ALWAYS << "WARNING: COULD NOT HIDE MOUSE CURSOR" << std::endl << "PICTURE WILL SUCK A BIT" <<
      std::endl << "This just had to go wrong -.-*" << std::endl;
00294     }
00295
00296     while (!viewer.done())
00297         viewer.frame();
00298
00299     wsi->setScreenResolution(GraphicsContext::ScreenIdentifier(screen), oldWidth, oldHeight);
```

```
00300     return EXIT_SUCCESS;
00301 }
```

9.57 Objects/Bench.cpp File Reference

```
#include "../header/UtilFunctions.h"
#include "../header/Bench.h"
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.58 Bench.cpp

```
00001 #include "../header/UtilFunctions.h"
00002
00003 #include "../header/Bench.h"
00004 using namespace osg;
00005
00006 namespace brtr{
00007     Bench::Bench(const Vec3& pcenter, const double plength){
00008       if(plength < 2 || plength > 30) length = 8;
00009             else length = plength;
00010
00011         initBench(length); //initalize the bench and saves it into a private attribute
00012         center = pcenter;
00013
00014         addChild(bench.get());
00015         setPosition(center);
00016     }
00017
00018     Bench::Bench(const Bench& copy, const CopyOp& copyop)
00019         : PositionAttitudeTransform(copy, copyop){}
00020
00021     Bench::~Bench() {
00022     }
00023
00024     ref_ptr<PositionAttitudeTransform> Bench::getHitbox(const double alpha, double height){
00025         Vec3 size = getDimensionOfNode(bench);
00026
00027         if (height < 0) height = size.z();
00028         ref_ptr<Group> hitbox = brtr::createCuboid(size.x(), size.y(), height, 0);
00029
00030         //needed to make the hitbox transparent
00031         ref_ptr<BlendFunc> blendFunc = new BlendFunc;
00032         blendFunc->setFunction(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
00033
00034         StateSet* stateset = hitbox->getOrCreateStateSet();
00035         Material * newmaterial = new Material();
00036         newmaterial->setEmission(Material::FRONT, Vec4(0, 0, 0, alpha));
00037         newmaterial->setDiffuse(Material::FRONT, Vec4(0, 0, 0, alpha));
00038         newmaterial->setAmbient(Material::FRONT, Vec4(0, 0, 0, alpha));
00039         stateset->setAttributeAndModes(newmaterial, StateAttribute::OVERRIDE | StateAttribute::ON);
00040         stateset->setRenderingHint(StateSet::TRANSPARENT_BIN);
00041
00042         stateset->setAttributeAndModes(blendFunc);
00043
00044         //set the position of the hitbox
00045         ref_ptr<PositionAttitudeTransform> benchpos = new PositionAttitudeTransform;
00046         benchpos->setPosition(this->center);
00047         benchpos->addChild(hitbox);
00048
00049
00050         return benchpos;
00051     }
00052
00053     ref_ptr<Material> Bench::createIronMaterial(){
00054         ref_ptr<Material> mat = createSimpleMaterial(   osg::Material::FRONT_AND_BACK,
00055                                 osg::Vec4(.9f, .9f, .9f, 1.0f), //diffuse
00056                                 osg::Vec4(.7f, .7f, .7f, 1.0f), //ambient
00057                                 osg::Vec4(.3f, .3f, .3f, 1.0f), //specular
```

```
00058                                 128); //Shininess
00059         return mat;
00060     }
00061
00062     ref_ptr<Material> Bench::createWoodMaterial(){
00063         ref_ptr<Material> mat = createSimpleMaterial(osg::Material::FRONT_AND_BACK,
00064                                 osg::Vec4(.3f, .3f, .3f, 1.0f), //diffuse
00065                                 osg::Vec4(.7f, .7f, .7f, 1.0f), //ambient
00066                                 osg::Vec4(.1f, .1f, .1f, 1.0f), //specular
00067                                 3); //Shininess
00068         return mat;
00069     }
00070
00071     ref_ptr<DrawElementsUInt> Bench::getPrimitiveSetforARectangle(int
      lsteps, int wsteps){
00072         ref_ptr<DrawElementsUInt> indices = new DrawElementsUInt(GL_TRIANGLE_STRIP);
00073         for (int i = 0; i < lsteps; i++) {
00074             for (int j = 0; j <= wsteps; j++) {
00075                 indices->push_back(i*(wsteps + 1) + j);
00076                 indices->push_back((i + 1)*(wsteps + 1) + j);
00077             }
00078             indices->push_back((i + 1)*(wsteps + 1) + wsteps);
00079             indices->push_back((i + 1)*(wsteps + 1));
00080         }
00081         return indices;
00082     }
00083
00084     ref_ptr<Geometry> Bench::createArmrestSidesLeftRight(double radius,
      double radius2, int lsteps, int wsteps, bool flip){
00085         ref_ptr<Geometry> side = new Geometry;
00086         ref_ptr<Vec3Array> vertices = new Vec3Array();
00087         ref_ptr<Vec3Array> normals = new Vec3Array();
00088
00089         double xstep = (radius2 - radius) / lsteps;
00090         double ystep = radius / wsteps;
00091         // current vertex coordinates
00092         double x = 0.0;
00093         double y = 0.0;
00094         //  current normal coordinates
00095         double nx = 0.0;
00096         double ny = 1.0;
00097         double nz = 0.0;
00098
00099         double curRad = radius; //the current radius
00100         double radSteps = (radius2 - radius) / wsteps;
00101         double xstepalpha = DegreesToRadians(90.0 / wsteps); //steps for the radius
00102
00103         double alpha = DegreesToRadians(180.0);
00104         if (flip)  alpha -= DegreesToRadians(90.0); // when this should be the other side, the start radius
       is changed
00105
00106         ref_ptr<Vec2Array> texcoords = new Vec2Array;
00107         // set vertices and normals
00108         for (int i = 0; i <= lsteps; i++) {
00109             y = 0.0;
00110             curRad = radius;
00111             for (int j = 0; j <= wsteps; j++) {
00112                 vertices->push_back(Vec3d(cos(alpha)*curRad, 0, sin(alpha)*curRad));
00113
00114                 texcoords->push_back(Vec2(x / (radius2 - radius), y / radius));
00115
00116                 normals->push_back(Vec3d(nx, ny, nz));
00117                 y += ystep;
00118                 curRad += radSteps;
00119             }
00120             alpha += xstepalpha;
00121
00122             x += xstep;
00123         }
00124
00125
00126         //normalize the normals
00127         for (auto cnt = 0; cnt < normals->size(); cnt++) {
00128             normals->at(cnt).normalize();
00129         }
00130
00131         side->setVertexArray(vertices.get());
00132         side->addPrimitiveSet(getPrimitiveSetforARectangle(lsteps,wsteps));
00133         side->setNormalArray(normals.get());
00134         side->setTexCoordArray(0, texcoords.get());
00135         side->setNormalBinding(Geometry::BIND_PER_VERTEX);
00136
00137         return side;
00138     }
00139
00140     ref_ptr<Geometry> Bench::createArmrestSidesFrontBack(double radius,
      double width, int lsteps, int wsteps,bool flip){
```

```
00141         ref_ptr<Geometry> side = new Geometry;
00142         ref_ptr<Vec3Array> vertices = new Vec3Array();
00143         ref_ptr<Vec3Array> normals = new Vec3Array();
00144         ref_ptr<DrawElementsUInt> indices = new DrawElementsUInt(GL_TRIANGLE_STRIP);
00145 
00146         double ystep = width / wsteps;
00147 
00148         double xstepalpha = DegreesToRadians(90.0 / wsteps); //steps for the radius
00149 
00150         // current vertex coordinates
00151         double y = 0.0;
00152         //  current normal coordinates
00153         double ny = 0.0;
00154 
00155         double alpha = DegreesToRadians(180.0);
00156         if (flip)  alpha -= DegreesToRadians(90.0); // when this should be the other side, the start radius
      is changed
00157         // set vertices and normals
00158         ref_ptr<Vec2Array> texcoords = new Vec2Array;
00159         for (int i = 0; i <= lsteps; i++) {
00160             y = 0.0;
00161             for (int j = 0; j <= wsteps; j++) {
00162 
00163                 if (!flip) {
00164                     vertices->push_back(Vec3d(cos(alpha)*radius, y, sin(alpha)*radius));
00165                     normals->push_back(Vec3d(-cos(alpha)*radius, ny, -sin(alpha)*radius));
00166                 }else{ // flip the vertices and normales to fit
00167                     vertices->push_back(Vec3d(cos(alpha)*radius, y, -sin(alpha)*radius));
00168                     normals->push_back(Vec3d(cos(alpha)*radius, ny, -sin(alpha)*radius));
00169                 }
00170 
00171 
00172                 texcoords->push_back(Vec2(cos(alpha) / 1, y / width)); //calculate the texture coordinates
00173                 y += ystep;
00174             }
00175             alpha += xstepalpha;
00176 
00177         }
00178 
00179         //normalize the normals
00180         for (auto cnt = 0; cnt < normals->size(); cnt++) {
00181             normals->at(cnt).normalize();
00182         }
00183         side->setVertexArray(vertices.get());
00184         side->addPrimitiveSet(getPrimitiveSetforARectangle(lsteps,wsteps));
00185         side->setNormalArray(normals.get());
00186         side->setNormalBinding(Geometry::BIND_PER_VERTEX);
00187         side->setTexCoordArray(0, texcoords.get());
00188         return side;
00189     }
00190     ref_ptr<Group> Bench::createLeg(){
00191 
00192         double length = 0.5;
00193         double width = length;
00194         double height = 4 * length;
00195 
00196         ref_ptr<Group> leg = createCuboid(length, width, height);
00197 
00198         //set Textures
00199         ref_ptr<Texture2D> texture = new Texture2D;
00200         ref_ptr<Image> image = osgDB::readImageFile("../BlenderFiles/Texturen/iron.jpg");
00201         texture->setImage(image.get());
00202         texture->setWrap(Texture::WRAP_S, Texture::MIRROR);
00203         texture->setWrap(Texture::WRAP_T, Texture::MIRROR);
00204         osg::ref_ptr<osg::StateSet> legs(leg->getOrCreateStateSet());
00205 
00206         legs->setAttribute(createIronMaterial());
00207         legs->setTextureAttributeAndModes(0, texture.get());
00208 
00209         return leg;
00210     }
00211 
00212     ref_ptr<Group> Bench::createSeat(const double width){
00213         ref_ptr<Group> seat = new Group;
00214 
00215         double length = width; //ratio of the seat is 1:1
00216         double height = 0.1;
00217 
00218         ref_ptr<Group> base = createCuboid(length, width, height);
00219 
00220          //create the texture
00221         ref_ptr<Texture2D> texturewood = new Texture2D;
00222         ref_ptr<Image> imagewood = osgDB::readImageFile("../BlenderFiles/Texturen/wood.jpg");
00223         texturewood->setImage(imagewood.get());
00224         texturewood->setWrap(Texture::WRAP_S, Texture::MIRROR);
00225         texturewood->setWrap(Texture::WRAP_T, Texture::MIRROR);
00226 
```

```
00227
00228
00229         base->getChild(0)->getOrCreateStateSet()->setAttribute(
      createWoodMaterial());
00230         base->getChild(0)->getOrCreateStateSet()->setTextureAttributeAndModes(0, texturewood.get()); //
      Front
00231
00232         base->getChild(1)->getOrCreateStateSet()->setAttribute(
      createWoodMaterial());
00233         base->getChild(1)->getOrCreateStateSet()->setTextureAttributeAndModes(0, texturewood.get()); //Back
00234
00235
00236         //rotate the base to get the back of the seat
00237         ref_ptr<MatrixTransform> seatback = new MatrixTransform();
00238         seatback->setMatrix(Matrix::rotate(PI / 2 * 1.1, 1, 0, 0));
00239         seatback->addChild(base.get());
00240
00241         //group the components
00242         seat->addChild(base);
00243         seat->addChild(seatback);
00244
00245         //change the position
00246         ref_ptr<PositionAttitudeTransform> seatpos = new PositionAttitudeTransform;
00247         seatpos = wrapInPositionAttitudeTransform(seat,Vec3d(0, -(width -
      getDimensionOfNode(seat).y()), 0.0));
00248
00249         return seatpos;
00250
00251     }
00252     ref_ptr<Group> Bench::createArmrest(double radius, double width, double length,
      double totalwidth){
00253         ref_ptr<Group> armrest = new Group;
00254
00255         ref_ptr<Group> armrest_arch = new Group;
00256         //creates the components for the armrest_arch
00257         ref_ptr<Geode> front = new Geode;
00258         front->addDrawable(createArmrestSidesFrontBack(radius, width, 10, 10,
      false));
00259         ref_ptr<Geode> back = new Geode;
00260         back->addDrawable(createArmrestSidesFrontBack(radius + length, width, 10
      , 10, true));
00261         ref_ptr<Geode> leftside = new Geode;
00262         leftside->addDrawable(createArmrestSidesLeftRight(radius, radius +
      length, 10, 10, true));
00263         ref_ptr<Geode> rigthside = new Geode;
00264         rigthside->addDrawable(createArmrestSidesLeftRight(radius, radius +
      length, 10, 10, false));
00265
00266         ref_ptr<PositionAttitudeTransform> leftsiderotated = new PositionAttitudeTransform;
00267         leftsiderotated->setAttitude(Quat(DegreesToRadians(180.0), X_AXIS));
00268         leftsiderotated->addChild(leftside);
00269
00270         ref_ptr<PositionAttitudeTransform> rigthsiderotated = new PositionAttitudeTransform;
00271         rigthsiderotated->setPosition(Vec3d(0, width, 0));
00272         rigthsiderotated->addChild(rigthside);
00273
00274
00275
00276         ref_ptr<Geode> topshape = new Geode;
00277         topshape->addDrawable(brtr::createRectangleWithTexcoords(length,
      width, 20, 20));
00278         ref_ptr<PositionAttitudeTransform> top = wrapInPositionAttitudeTransform
      (topshape,Vec3d(-(radius + length), 0, 0));
00279
00280
00281         ref_ptr <MatrixTransform> bottom = new MatrixTransform;
00282         bottom->setMatrix(Matrix::rotate(PI / 2, 0, 1, 0)*Matrix::translate(0, 0, -radius));
00283         bottom->addChild(topshape);
00284
00285
00286
00287
00288         armrest_arch->addChild(front);
00289         armrest_arch->addChild(back);
00290         armrest_arch->addChild(leftsiderotated);
00291         armrest_arch->addChild(rigthsiderotated);
00292         armrest_arch->addChild(top);
00293         armrest_arch->addChild(bottom);
00294
00295         //Positioning the armrest_arch
00296         armrest_arch = wrapInPositionAttitudeTransform(armrest_arch, Vec3d(
      radius + length, 0, radius + length));
00297
00298         double height = 0.5;
00299         ref_ptr<Group> bar_up = brtr::createCuboid(length, width, height);
00300         ref_ptr<Group> barontop = brtr::createCuboid(width, totalwidth, width / 2);
00301
```

```
00302         bar_up = wrapInPositionAttitudeTransform(bar_up, Vec3d(0.0, 0.0,
      radius + length));
00303         barontop = wrapInPositionAttitudeTransform(barontop, Vec3d(-width /
      4, -(totalwidth) / 2 + width / 2, radius + length + height));
00304
00305         ref_ptr<Group> bars = new Group;
00306         bars->addChild(barontop);
00307         bars->addChild(bar_up);
00308
00309
00310
00311         armrest->addChild(armrest_arch);
00312         armrest->addChild(bars);
00313
00314
00315         ref_ptr<Texture2D> texture = new Texture2D;
00316         ref_ptr<Image> image = osgDB::readImageFile("../BlenderFiles/Texturen/iron.jpg");
00317         texture->setImage(image.get());
00318         texture->setWrap(Texture::WRAP_S, Texture::MIRROR);
00319         texture->setWrap(Texture::WRAP_T, Texture::MIRROR);
00320
00321         osg::ref_ptr<osg::StateSet> armrest_state(armrest->getOrCreateStateSet());
00322
00323         armrest_state->setAttribute(createIronMaterial());
00324         armrest_state->setTextureAttributeAndModes(0, texture.get());
00325
00326         return armrest;
00327     }
00328
00329
00330
00331     ref_ptr<Group> Bench::createBar(){
00332
00333         double width = 0.5;
00334         double height = width;
00335         double length = (this->length);
00336
00337
00338         ref_ptr<Group> bar = brtr::createCuboid(length, width, height);
00339
00340
00341         //set Textures
00342         ref_ptr<Texture2D> texture = new Texture2D;
00343         ref_ptr<Image> image =
00344             osgDB::readImageFile("../BlenderFiles/Texturen/iron.jpg");
00345         texture->setImage(image.get());
00346         texture->setWrap(Texture::WRAP_S, Texture::MIRROR);
00347         texture->setWrap(Texture::WRAP_T, Texture::MIRROR);
00348
00349         osg::ref_ptr<osg::StateSet> nodess(bar->getOrCreateStateSet());
00350
00351
00352         nodess->setAttribute(createIronMaterial());
00353         nodess->setTextureAttributeAndModes(0, texture.get());
00354
00355
00356         return bar;
00357     }
00358
00359
00360     void Bench::initBench(const double plength){
00361         ref_ptr<Group> leg = createLeg();
00362
00363         double legdistance = 0.05;
00364
00365         //create the legs
00366         ref_ptr<PositionAttitudeTransform> leg1 = new PositionAttitudeTransform;
00367         ref_ptr<PositionAttitudeTransform> leg2 = new PositionAttitudeTransform;
00368         leg1->addChild(leg.get());
00369         leg2->addChild(leg.get());
00370
00371         ref_ptr<Group> legs = new Group;
00372         legs->addChild(leg1.get());
00373         legs->addChild(leg2.get());
00374
00375         //save the dimension
00376         Vec3 legssize = getDimensionOfNode(legs);
00377
00378
00379         //create the bar
00380         ref_ptr<Group> bar = createBar();
00381         ref_ptr<PositionAttitudeTransform> bar1 = new PositionAttitudeTransform;
00382         bar1->addChild(bar.get());
00383         ref_ptr<Group> bars = new Group;
00384         ref_ptr<PositionAttitudeTransform> armrest1 = new PositionAttitudeTransform;
00385
00386
```

```
00387         Vec3 barssize = getDimensionOfNode(bar);
00388
00389
00390
00391         int anzahl_sitze = (int)(0.5*plength); // caluclate the number of seats
00392         double sitzspacebetween = 0.9;
00393         double seatwidth = (barssize.x()*sitzspacebetween) / anzahl_sitze;
00394
00395
00396
00397
00398         double radiusarmrest = 0.2;
00399
00400         ref_ptr<Group> armrest = createArmrest(radiusarmrest, barssize.y() / 2, barssize.z() /
       4, seatwidth);
00401
00402
00403
00404         armrest1->addChild(armrest);
00405
00406         armrest1->setPosition(Vec3d(-(radiusarmrest + barssize.z() / 4), (barssize.y() / 4), (barssize.z()
      / 2) - (barssize.z() / 8)));
00407
00408
00409         ref_ptr <MatrixTransform> armrest2 = new MatrixTransform;
00410         armrest2->setMatrix(Matrix::translate(-(radiusarmrest + (barssize.z() / 2) / 2), 0, 0)*
      Matrix::rotate(PI, 0, 0, 1));
00411         armrest2->addChild(armrest);
00412
00413         ref_ptr<PositionAttitudeTransform> armrest22 = new PositionAttitudeTransform;
00414         armrest22->setPosition(Vec3d(barssize.x(), (barssize.y() / 4) + barssize.y() / 2, (barssize.z() / 2
      ) - (barssize.z() / 8)));
00415         armrest22->addChild(armrest2);
00416
00417         bar1->addChild(armrest1);
00418         bar1->addChild(armrest22);
00419
00420         bars->addChild(bar1.get());
00421         Vec3 armrestsize = getDimensionOfNode(armrest);
00422
00423
00424
00425
00426         //creating the seats
00427         ref_ptr<Group> seat = createSeat(seatwidth);
00428
00429         Vec3 seatsize = getDimensionOfNode(seat);
00430         double middle = (seatsize.y() - seatwidth) + (seatwidth / 2) - barssize.y() / 2;
00431         //Position the stuff
00432         leg1->setPosition(Vec3d(legdistance*plength, middle, 0.0));
00433         leg2->setPosition(Vec3d((1 - legdistance)*plength - getDimensionOfNode(leg).x(),
      middle, 0.0));
00434         bar1->setPosition(Vec3d(0, middle, legssize.z()));
00435
00436         //duplicate the seats
00437         PositionAttitudeTransform *sitze[20]; //maximum of 20 seats
00438         ref_ptr<Group> sitze_all = new Group;
00439         for (int i = 0; i < anzahl_sitze; i++){
00440             sitze[i] = new PositionAttitudeTransform;
00441             double posx = (plength / anzahl_sitze) * i + ((plength / anzahl_sitze) - (seatsize.x())) / 2;
      //calcute the position for each seat
00442             sitze[i]->setPosition(Vec3d(posx, 0, legssize.z() + barssize.z()));
00443             sitze[i]->addChild(seat.get());
00444             sitze_all->addChild(sitze[i]);
00445         }
00446
00447         //bundle all components together into one group
00448         ref_ptr<Group> bench = new Group;
00449         bench->addChild(legs.get());
00450         bench->addChild(bars.get());
00451         bench->addChild(sitze_all.get());
00452
00453         //position
00454         ref_ptr<PositionAttitudeTransform> benchpos = new PositionAttitudeTransform;
00455         benchpos = wrapInPositionAttitudeTransform(bench,Vec3d(armrestsize.x
      (), 0, 0));
00456
00457
00458         this->bench = benchpos;
00459     }
00460
00461
00462
00463 }
00464
```

9.59 Objects/ControlRoom.cpp File Reference

```
#include "../header/ControlRoom.h"
#include "../header/UtilFunctions.h"
#include "../header/CelShading.h"
#include <osg/Geode>
#include <osg/MatrixTransform>
#include <osg/ValueObject>
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.60 ControlRoom.cpp

```
00001 #include "../header/ControlRoom.h"
00002 #include "../header/UtilFunctions.h"
00003 #include "../header/CelShading.h"
00004
00005 #include <osg/Geode>
00006 #include <osg/MatrixTransform>
00007 #include <osg/ValueObject>
00008
00009 using namespace osg;
00010
00011 namespace brtr{
00012
00013     ControlRoom::ControlRoom(double roomSize, int lod,
      brtr::ToonTexSwitcherCallback& toonCallback,
      brtr::ProgramSwitcherCallback& programCallback) {
00014         ref_ptr<Group> roomRoot = createRoomSurrounding(roomSize, lod);
00015         ref_ptr<Group> chessFigureRoot = createChessFigures(toonCallback,programCallback);
00016
00017         ref_ptr<brtr::CelShading> celShader = new brtr::CelShading(false);
00018         ref_ptr<LightSource> roomLight = brtr::createLight(Vec3(0, 0, roomSize / 2.0 /2.0
      +2), 0);
00019
00020         celShader->addChild(roomLight);
00021         celShader->addChild(roomRoot);
00022         celShader->addChild(chessFigureRoot);
00023         addChild(celShader);
00024     }
00025
00026     ref_ptr<Group> ControlRoom::createRoomSurrounding(double roomSize, int lod) {
00027         ref_ptr<Geode> floorGeode = new Geode;
00028         floorGeode->addDrawable(brtr::createRectangle(roomSize, roomSize, lod, lod));
00029         ref_ptr<MatrixTransform> floor = new MatrixTransform;
00030         floor->setMatrix(Matrix::translate(-roomSize / 2, -roomSize / 2, 0));
00031         floor->addChild(floorGeode);
00032
00033         ref_ptr<Geode> ceilingGeode = new Geode;
00034         ceilingGeode->addDrawable(brtr::createRectangle(roomSize, roomSize, lod, lod))
      ;
00035         ref_ptr<MatrixTransform> ceiling = new MatrixTransform;
00036         ceiling->setMatrix(Matrix::translate(-roomSize / 2, -roomSize / 2, -roomSize / 2)
00037             * Matrix::rotate(DegreesToRadians(180.0), X_AXIS));
00038         ceiling->addChild(ceilingGeode);
00039
00040         ref_ptr<Geode> firstWallGeode = new Geode;
00041         firstWallGeode->addDrawable(brtr::createRectangle(roomSize / 2, roomSize, lod,
       lod));
00042         ref_ptr<MatrixTransform> firstWall = new MatrixTransform;
00043         firstWall->setMatrix(
00044             Matrix::translate(-roomSize / 2, -roomSize / 2, 0)
00045             *Matrix::rotate(DegreesToRadians(90.0), Y_AXIS)
00046             *Matrix::translate(-roomSize / 2, 0, 0)
00047             );
00048         firstWall->addChild(firstWallGeode);
00049
00050         ref_ptr<Geode> secondWallGeode = new Geode;
00051         secondWallGeode->addDrawable(brtr::createRectangle(roomSize / 2, roomSize, lod
      , lod));
00052         ref_ptr<MatrixTransform> secondWall = new MatrixTransform;
```

```
00053         secondWall->setMatrix(
00054             Matrix::translate(-roomSize / 2, -roomSize / 2, 0)
00055             *Matrix::rotate(DegreesToRadians(-90.0), Y_AXIS)
00056             *Matrix::translate(roomSize / 2, 0, roomSize / 2)
00057             );
00058         secondWall->addChild(secondWallGeode);
00059 
00060         ref_ptr<Geode> thirdWallGeode = new Geode;
00061         thirdWallGeode->addDrawable(brtr::createRectangle(roomSize, roomSize / 2, lod,
      lod));
00062         ref_ptr<MatrixTransform> thirdWall = new MatrixTransform;
00063         thirdWall->setMatrix(
00064             Matrix::translate(-roomSize / 2, -roomSize / 2, 0)
00065             *Matrix::rotate(DegreesToRadians(-90.0), X_AXIS)
00066             *Matrix::translate(0, -roomSize / 2, 0)
00067             );
00068         thirdWall->addChild(thirdWallGeode);
00069 
00070         ref_ptr<Geode> fourthWallGeode = new Geode;
00071         fourthWallGeode->addDrawable(brtr::createRectangle(roomSize, roomSize / 2, lod
     , lod));
00072         ref_ptr<MatrixTransform> fourthWall = new MatrixTransform;
00073         fourthWall->setMatrix(
00074             Matrix::translate(-roomSize / 2, -roomSize / 2, 0)
00075             *Matrix::rotate(DegreesToRadians(90.0), X_AXIS)
00076             *Matrix::translate(0, roomSize / 2, roomSize / 2)
00077             );
00078         fourthWall->addChild(thirdWallGeode);
00079 
00080         ref_ptr<Group> roomRoot = new Group;
00081         roomRoot->addChild(floor);
00082         roomRoot->addChild(ceiling);
00083         roomRoot->addChild(firstWall);
00084         roomRoot->addChild(secondWall);
00085         roomRoot->addChild(thirdWall);
00086         roomRoot->addChild(fourthWall);
00087         roomRoot->setNodeMask(brtr::collisionMask);
00088         //this is the fakewall
00089         thirdWall->setNodeMask(~brtr::interactionAndCollisionMask);
00090 
00091         //material for the whole room
00092         ref_ptr<Material> roomMaterial = createMaterial(Vec4(0.3, 0.3, 0.3, 1.0), Vec4(0.4, 0.4, 0.4, 1.0),
      Vec4(0.9, 0.9, 0.9, 1.0), 42);
00093         roomRoot->getOrCreateStateSet()->setAttributeAndModes(roomMaterial, StateAttribute::ON);
00094         //shader should know that there is no texture
00095         roomRoot->getOrCreateStateSet()->addUniform(new Uniform("tex", false), StateAttribute::ON |
     StateAttribute::OVERRIDE);
00096 
00097         return roomRoot;
00098     }
00099 
00100     ref_ptr<Group> ControlRoom::createChessFigures(brtr::ToonTexSwitcherCallback
     & toonCallback, brtr::ProgramSwitcherCallback& programCallback)
00101 {
00102         ref_ptr<Geometry> chessFigure1Geometry = brtr::createChessFigure();
00103         ref_ptr<Geode> chessFigure1Source = new Geode;
00104         chessFigure1Source->addDrawable(chessFigure1Geometry);
00105 
00106         ref_ptr<MatrixTransform> chessFigure1 = new MatrixTransform;
00107         chessFigure1->setMatrix(
00108             Matrix::translate(-5, 0, 0)
00109             );
00110         chessFigure1->addChild(chessFigure1Source);
00111         chessFigure1->getOrCreateStateSet()->addUniform(new Uniform("xAnimation", true), StateAttribute::ON
      | StateAttribute::OVERRIDE);
00112 
00113         ref_ptr<Geode> chessFigure2Source = new Geode;
00114         chessFigure2Source->addDrawable(brtr::createChessFigure());
00115         ref_ptr<MatrixTransform> chessFigure2 = new MatrixTransform;
00116         chessFigure2->setMatrix(
00117             Matrix::translate(0, 5, 0)
00118             );
00119         chessFigure2->addChild(chessFigure2Source);
00120         chessFigure2->getOrCreateStateSet()->addUniform(new Uniform("zAnimation", true), StateAttribute::ON
      | StateAttribute::OVERRIDE);
00121         chessFigure2->getOrCreateStateSet()->setAttributeAndModes(createMaterial(Vec4(0.4583, 0.35, 1, 1),
      Vec4(0.4583, 0.35, 1, 1)), StateAttribute::ON | StateAttribute::OVERRIDE);
00122 
00123         ref_ptr<Geometry> chessFigure3Geometry = brtr::createChessFigure();
00124         ref_ptr<Geode> chessFigure3Source = new Geode;
00125         chessFigure3Source->addDrawable(chessFigure3Geometry);
00126         ref_ptr<MatrixTransform> chessFigure3 = new MatrixTransform;
00127         chessFigure3->setMatrix(
00128             Matrix::translate(5, 0, 0)
00129             );
00130         chessFigure3->addChild(chessFigure3Source);
00131         chessFigure3->getOrCreateStateSet()->addUniform(new Uniform("yAnimation", true), StateAttribute::ON
```

```
       | StateAttribute::OVERRIDE);
00132         chessFigure3->getOrCreateStateSet()->setAttributeAndModes(createMaterial(Vec4(0, 0.63, 0.084, 1),
      Vec4(0, 0.63, 0.084, 1)), StateAttribute::ON | StateAttribute::OVERRIDE);
00133
00134
00135         chessFigure1Geometry->getOrCreateUserDataContainer()->addUserObject(&toonCallback);
00136         chessFigure3Geometry->getOrCreateUserDataContainer()->addUserObject(&programCallback);
00137         ref_ptr<Group> chessFigureRoot = new Group;
00138         chessFigureRoot->addChild(chessFigure1);
00139         chessFigureRoot->addChild(chessFigure2);
00140         chessFigureRoot->addChild(chessFigure3);
00141
00142         return chessFigureRoot;
00143     }
00144
00145     ref_ptr<Material> ControlRoom::createMaterial(Vec4 diffuse, Vec4 ambient, Vec4 specular, double
      shininess) {
00146         ref_ptr<Material> mat = new Material;
00147         mat->setAmbient(Material::FRONT_AND_BACK, ambient);
00148         mat->setDiffuse(Material::FRONT_AND_BACK, diffuse);
00149         mat->setSpecular(Material::FRONT_AND_BACK, specular);
00150         mat->setShininess(Material::FRONT_AND_BACK, shininess);
00151         return mat;
00152     }
00153
00154 }
```

9.61 Shader/CelShading.cpp File Reference

```
#include "CelShading.h"
#include <osg/Texture2D>
#include <osgDB/ReadFile>
#include <osg/LineWidth>
#include <osg/Material>
#include <osg/Program>
#include <osg/Shader>
#include <osg/PolygonOffset>
#include "osg/TexEnv"
#include "osg/PolygonMode"
#include "osg/CullFace"
```

Classes

- class brtr::CelShadingTechnique

  The Technique for the cel-shading effect.

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.62 CelShading.cpp

```
00001 #include "CelShading.h"
00002
00003 #include <osg/Texture2D>
00004 #include <osgDB/ReadFile>
00005 #include <osg/LineWidth>
00006 #include <osg/Material>
00007 #include <osg/Program>
00008 #include <osg/Shader>
00009 #include <osg/PolygonOffset>
00010 #include "osg/TexEnv"
00011 #include "osg/PolygonMode"
```

```
00012 #include "osg/CullFace"
00013
00014 namespace brtr{
00022     class CelShadingTechnique : public osgFX::Technique {
00023     public:
00024         CelShadingTechnique(osg::Material* material, osg::LineWidth *lineWidth, bool
      secondPass, std::string vertSource)
00025             : Technique(),
00026             _material(material),
00027             _lineWidth(lineWidth),
00028             _secondPass(secondPass),
00029             _vertSource(vertSource){}
00030
00031     protected:
00032
00033         void define_passes() {
00034             // implement pass #1 (solid surfaces)
00035                 {
00036                     osg::ref_ptr<osg::Shader> toonFrag = osgDB::readShaderFile("../Shader/celShader.frag");
00037                     osg::ref_ptr<osg::Shader> toonVert = osgDB::readShaderFile("../Shader/" +
      _vertSource);
00038                     osg::ref_ptr<osg::Program> celShadingProgram = new osg::Program;
00039                     celShadingProgram->addShader(toonFrag);
00040                     celShadingProgram->addShader(toonVert);
00041
00042                     osg::ref_ptr<osg::StateSet> ss = new osg::StateSet;
00043
00044                     ss->addUniform(new osg::Uniform("toonTex", 1));
00045                     ss->setAttributeAndModes(celShadingProgram, osg::StateAttribute::OVERRIDE |
      osg::StateAttribute::ON);
00046
00047                     ss->addUniform(new osg::Uniform("tex", true));
00048                     ss->addUniform(new osg::Uniform("zAnimation", false));
00049                     ss->addUniform(new osg::Uniform("xAnimation", false));
00050                     ss->addUniform(new osg::Uniform("yAnimation", false));
00051                     addPass(ss);
00052                 }
00053
00054             // implement pass #2 (outlines) copy/paste from osgFX::Cartoon
00055             if(_secondPass){
00056                 osg::ref_ptr<osg::StateSet> ss = new osg::StateSet;
00057                 osg::ref_ptr<osg::PolygonMode> polymode = new osg::PolygonMode;
00058                 polymode->setMode(osg::PolygonMode::FRONT_AND_BACK, osg::PolygonMode::LINE);
00059                 ss->setAttributeAndModes(polymode.get(), osg::StateAttribute::OVERRIDE |
      osg::StateAttribute::ON);
00060
00061                 osg::ref_ptr<osg::CullFace> cf = new osg::CullFace;
00062                 cf->setMode(osg::CullFace::FRONT);
00063                 ss->setAttributeAndModes(cf.get(), osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON)
      ;
00064
00065                 ss->setAttributeAndModes(_lineWidth.get(), osg::StateAttribute::OVERRIDE |
      osg::StateAttribute::ON);
00066
00067                 _material->setColorMode(osg::Material::OFF);
00068                 _material->setDiffuse(osg::Material::FRONT_AND_BACK, osg::Vec4(0, 0, 0, 1));
00069                 _material->setAmbient(osg::Material::FRONT_AND_BACK, osg::Vec4(0, 0, 0, 1));
00070                 _material->setSpecular(osg::Material::FRONT_AND_BACK, osg::Vec4(0, 0, 0, 1));
00071
00072                 // set by outline colour so no need to set here.
00073                 _material->setEmission(osg::Material::FRONT_AND_BACK, osg::Vec4(0, 0, 0, 1));
00074
00075                 ss->setAttributeAndModes(_material.get(), osg::StateAttribute::OVERRIDE |
      osg::StateAttribute::ON);
00076
00077                 ss->setMode(GL_LIGHTING, osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON);
00078                 ss->setTextureMode(0, GL_TEXTURE_1D, osg::StateAttribute::OVERRIDE |
      osg::StateAttribute::OFF);
00079                 ss->setTextureMode(0, GL_TEXTURE_2D, osg::StateAttribute::OVERRIDE |
      osg::StateAttribute::OFF);
00080                 ss->setTextureMode(1, GL_TEXTURE_1D, osg::StateAttribute::OVERRIDE |
      osg::StateAttribute::OFF);
00081                 ss->setTextureMode(1, GL_TEXTURE_2D, osg::StateAttribute::OVERRIDE |
      osg::StateAttribute::OFF);
00082
00083                  addPass(ss.get());
00084             }
00085         }
00086
00087     private:
00088         osg::ref_ptr<osg::Material> _material;
00089         osg::ref_ptr<osg::LineWidth> _lineWidth;
00090         std::string _toonTex;
00091         bool _secondPass;
00092         std::string _vertSource;
00093     };
00094
```

```
00096
00097     CelShading::CelShading(bool secondPass, std::string vertSource)
00098         : Effect(),
00099         _material(new osg::Material),
00100         _lineWidth(new osg::LineWidth(3.0f)),
00101         _secondPass(secondPass),
00102         _vertSource(vertSource){}
00103
00104     CelShading::CelShading(const CelShading& copy, const osg::CopyOp&
      copyop /*= osg::CopyOp::SHALLOW_COPY*/):
00105         osgFX::Effect(copy, copyop),
00106         _material(static_cast<osg::Material*>(copyop(copy._material.get()))),
00107         _lineWidth(static_cast<osg::LineWidth *>(copyop(copy._lineWidth.get()))) {}
00108
00109
00110     bool CelShading::define_techniques() {
00111         addTechnique(new CelShadingTechnique(_material,
      _lineWidth, _secondPass, _vertSource));
00112         return true;
00113     }
00114 }
```

9.63 Util/AddInteractionCallbackToDrawableVisitor.cpp File Reference

```
#include "../header/AddInteractionCallbackToDrawableVisitor.h"
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.64 AddInteractionCallbackToDrawableVisitor.cpp

```
00001 #include "../header/AddInteractionCallbackToDrawableVisitor.h"
00002
00003 namespace brtr{
00004
00005
      AddInteractionCallbackToDrawableVisitor::AddInteractionCallbackToDrawableVisitor
      (brtr::BaseInteractionCallback* callbackToAdd) {
00006         setTraversalMode(osg::NodeVisitor::TRAVERSE_ALL_CHILDREN);
00007         _containerToAdd = new osg::DefaultUserDataContainer;
00008         _containerToAdd->addUserObject(callbackToAdd);
00009     }
00010
00011     void AddInteractionCallbackToDrawableVisitor::apply(
      osg::Geode& geode) {
00012         for (int i = 0; i < geode.getNumDrawables(); ++i) {
00013             geode.getDrawable(i)->setUserDataContainer(_containerToAdd);
00014         }
00015     }
00016
00017 }
```

9.65 Util/GeometryPlacerVisitor.cpp File Reference

```
#include "../header/GeometryPlacerVisitor.h"
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.66 GeometryPlacerVisitor.cpp

```
00001 #include "../header/GeometryPlacerVisitor.h"
00002
00003 using namespace osg;
00004
00005 namespace brtr {
00006     GeometryPlacerVisitor::GeometryPlacerVisitor(osg::Geometry* geometryToPlace) :
00007         _geometryToPlace(geometryToPlace) {
00008         setTraversalMode(NodeVisitor::TRAVERSE_ALL_CHILDREN);
00009     }
00010
00011     void GeometryPlacerVisitor::apply(osg::Geode& geode) {
00012         geode.removeDrawables(0, geode.getNumDrawables());
00013         geode.addDrawable(_geometryToPlace);
00014     }
00015
00016     osg::ref_ptr<osg::Geometry> GeometryPlacerVisitor::getGeometryToPlace
    () const {
00017         return _geometryToPlace;
00018     }
00019
00020     void GeometryPlacerVisitor::setGeometryToPlace(
    osg::ref_ptr<osg::Geometry> val) {
00021         _geometryToPlace = val;
00022     }
00023
00024 }
```

9.67 Util/ModifyMaterialVisitor.cpp File Reference

```
#include "../header/ModifyMaterialVisitor.h"
#include <osg/Material>
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

9.68 ModifyMaterialVisitor.cpp

```
00001 #include "../header/ModifyMaterialVisitor.h"
00002 #include <osg/Material>
00003 namespace brtr{
00004
00005     ModifyMaterialVisitor::ModifyMaterialVisitor() :
00006         _ambientFlag(false),
00007         _diffuseFlag(false),
00008         _specularFlag(false),
00009         _shininessFlag(false) {
00010         setTraversalMode(osg::NodeVisitor::TRAVERSE_ALL_CHILDREN);
00011
00012     }
00013
00014     void ModifyMaterialVisitor::apply(osg::Geode& geode) {
00015         for (unsigned int i = 0; i < geode.getNumDrawables(); ++i) {
00016             osg::Drawable* drawable = geode.getDrawable(i);
00017             osg::Material* materialPtr = dynamic_cast<osg::Material*>(drawable->getOrCreateStateSet()->
    getAttribute(osg::StateAttribute::Type::MATERIAL));
00018             if (materialPtr) {
00019                 osg::Material& material = *materialPtr;
00020                 if (_ambientFlag)
00021                     material.setAmbient(osg::Material::FRONT_AND_BACK, _ambient);
00022                 if (_specularFlag)
00023                     material.setSpecular(osg::Material::FRONT_AND_BACK,
    _specular);
00024                 if (_shininessFlag)
00025                     material.setShininess(osg::Material::FRONT_AND_BACK,
    _shininess);
00026                 if (_diffuseFlag)
00027                     material.setDiffuse(osg::Material::FRONT_AND_BACK, _diffuse);
```

```
00028             }
00029         }
00030
00031     }
00032
00033     osg::Vec4 ModifyMaterialVisitor::getDiffuse() const {
00034         return _diffuse;
00035     }
00036
00037     ModifyMaterialVisitor&
      ModifyMaterialVisitor::setDiffuse(osg::Vec4 val) {
00038         _diffuseFlag = true;
00039         _diffuse = val;
00040         return *this;
00041     }
00042
00043     osg::Vec4 ModifyMaterialVisitor::getSpecular() const {
00044         return _specular;
00045     }
00046
00047     ModifyMaterialVisitor&
      ModifyMaterialVisitor::setSpecular(osg::Vec4 val) {
00048         _specular = val;
00049         _specularFlag = true;
00050         return *this;
00051     }
00052
00053     osg::Vec4 ModifyMaterialVisitor::getAmbient() const {
00054         return _ambient;
00055     }
00056
00057     ModifyMaterialVisitor&
      ModifyMaterialVisitor::setAmbient(osg::Vec4 val) {
00058         _ambient = val;
00059         _ambientFlag = true;
00060         return *this;
00061     }
00062
00063     double ModifyMaterialVisitor::getShininess() const {
00064         return _shininess;
00065     }
00066
00067     ModifyMaterialVisitor&
      ModifyMaterialVisitor::setShininess(double val) {
00068         _shininess = val;
00069         _shininessFlag = true;
00070         return *this;
00071     }
00072
00073 }
00074
00075
```

9.69 Util/UtilFunctions.cpp File Reference

```
#include "../header/UtilFunctions.h"
#include "../header/CelShading.h"
#include <osgText/Text>
#include <osg/PolygonMode>
#include <osg/LightSource>
#include <osg/BlendFunc>
#include <osg/ComputeBoundsVisitor>
#include "osgFX/Outline"
#include <osg/Point>
#include <osg/PointSprite>
#include <osgParticle/ParticleSystem>
#include <osgParticle/ParticleSystemUpdater>
#include <osgParticle/Particle>
#include <osgParticle/ModularEmitter>
#include <osgParticle/ModularProgram>
#include <osgParticle/RandomRateCounter>
#include <osgParticle/MultiSegmentPlacer>
#include <osgParticle/RadialShooter>
#include <osgParticle/FluidFrictionOperator>
#include <osgParticle/AccelOperator>
```

Namespaces

- brtr

  Namespace for the whole BrainTrain Project.

Functions

- osg::ref_ptr< osg::Camera > brtr::createRTTCamera (osg::Camera::BufferComponent buffer, osg::Texture *tex, bool isAbsolute=false)

  creates a RTTCam
- osg::ref_ptr< osg::Geode > brtr::createScreenQuad (float width, float height, float scale=1.0f)

  creates a texture-ready screen quad for postprocessing
- osg::ref_ptr< osg::Camera > brtr::createHUDCamera (double left, double right, double bottom, double top)

  creates a HUD-Cam with a 2D-orthogonal projection matrix
- osg::ref_ptr< osg::Geometry > brtr::createRectangle (double length, double width, int lsteps, int wsteps)

  Creates a Rectangle with TRIANGLE_STRIPS.
- osg::ref_ptr< osg::Geometry > brtr::createRectangleWithTexcoords (double length, double width, int lsteps, int wsteps)

  Creates a Rectangle with TRIANGLE_STRIPS.
- osg::ref_ptr< osg::Geometry > brtr::createBodyOfRotation (double height, int hsteps, int rsteps, const BodyOfRotationFunction &function)

  Creates a body of rotation.
- osg::ref_ptr< osgText::Text > brtr::createText (const osg::Vec3 &pos, const std::string &content, float size)

  creates a (arial) text object for use with a hud camera
- ref_ptr< LightSource > brtr::createLight (const Vec3 &pos, int lightNum, int point, double spotCutoff, double spotExponent)
- osg::ref_ptr < osg::PositionAttitudeTransform > brtr::wrapInPositionAttitudeTransform (osg::Node *srcNode, const osg::Vec3d &pos)

  Return the given Node in a PositionAttitudeTransform with a given position.

- osg::ref_ptr< osg::Group > brtr::createCuboid (const double length, const double width, const double height, const double factor=6)

  Creates a Cubiod with TRIANGLE_STRIPS using the createRectangle function.
- void brtr::createRenderingPipeline (unsigned int width, unsigned int height, osg::Node &rootForToon, osg↩ Viewer::Viewer &viewer, RenderingPipeline &pipe, Vec3f &fogColor)
- osg::ref_ptr< osg::Geometry > brtr::createBeerBottle ()

  Creates a BeerBottle with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createRealBottle ()

  Creates a Bottle with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createVase ()

  Creates a vase with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createStalk ()

  Creates a stalk with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createBud ()

  Creates a bud with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geometry > brtr::createChessFigure ()

  Creates a "ChessFigure" with Material with the help of the BodyOfRotationFunction.
- osg::ref_ptr< osg::Geode > brtr::createCrosshair (unsigned int width, unsigned int height)

  creates a crosshair in the middle of the screen
- osg::ref_ptr
  < osg::PositionAttitudeTransform > brtr::createVaseWithFlower ()

  combines the stalk, bud and vase in a postitionAttitudetransform
- osg::ref_ptr< osg::Material > brtr::createSimpleMaterial (osg::Material::Face face, const osg::Vec4 &diffuse, const osg::Vec4 &ambient, const osg::Vec4 &specular, const double shininess)

  creates a simple material
- osg::Vec3 brtr::getDimensionOfNode (Node *source)
- osg::ref_ptr< osg::Texture2D > brtr::createToonTex (std::string toonTex)

  creates a Texture2D object with the given toonTex

9.70 UtilFunctions.cpp

```
00001 #include "../header/UtilFunctions.h"
00002 #include "../header/CelShading.h"
00003 #include <osgText/Text>
00004 #include <osg/PolygonMode>
00005 #include <osg/LightSource>
00006 #include <osg/BlendFunc>
00007 #include <osg/ComputeBoundsVisitor>
00008 #include "osgFX/Outline"
00009
00010 #include <osg/Point>
00011 #include <osg/PointSprite>
00012 #include <osgParticle/ParticleSystem>
00013 #include <osgParticle/ParticleSystemUpdater>
00014 #include <osgParticle/Particle>
00015 #include <osgParticle/ModularEmitter>
00016 #include <osgParticle/ModularProgram>
00017 #include <osgParticle/RandomRateCounter>
00018 #include <osgParticle/MultiSegmentPlacer>
00019 #include <osgParticle/RadialShooter>
00020 #include <osgParticle/FluidFrictionOperator>
00021 #include <osgParticle/AccelOperator>
00022
00023 using namespace osg;
00024
00025 namespace brtr{
00026
00027     ref_ptr<osg::Camera> createRTTCamera(osg::Camera::BufferComponent buffer, osg::Texture*
      tex, bool isAbsolute) {
00028         osg::ref_ptr<osg::Camera> camera = new osg::Camera;
00029         camera->setClearColor(osg::Vec4());
00030         camera->setClearMask(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
00031         camera->setRenderTargetImplementation(osg::Camera::FRAME_BUFFER_OBJECT);
00032         camera->setRenderOrder(osg::Camera::PRE_RENDER);
```

```
00033         if (tex) {
00034             tex->setFilter(osg::Texture2D::MIN_FILTER, osg::Texture2D::LINEAR);
00035             tex->setFilter(osg::Texture2D::MAG_FILTER, osg::Texture2D::LINEAR);
00036             camera->setViewport(0, 0, tex->getTextureWidth(), tex->getTextureHeight());
00037             camera->attach(buffer, tex);
00038         }
00039 
00040         if (isAbsolute) {
00041             camera->setReferenceFrame(osg::Transform::ABSOLUTE_RF);
00042             camera->setProjectionMatrix(osg::Matrix::ortho2D(0.0, 1.0, 0.0, 1.0));
00043             camera->setViewMatrix(osg::Matrix::identity());
00044             camera->addChild(createScreenQuad(1.0f, 1.0f,1.0f));
00045         }
00046         return camera;
00047     }
00048 
00049     ref_ptr<osg::Geode> createScreenQuad(float width, float height, float scale) {
00050         osg::Geometry* geom = osg::createTexturedQuadGeometry(
00051             osg::Vec3(), osg::Vec3(width, 0.0f, 0.0f), osg::Vec3(0.0f, height, 0.0f),
00052             0.0f, 0.0f, width*scale, height*scale);
00053         osg::ref_ptr<osg::Geode> quad = new osg::Geode;
00054         quad->addDrawable(geom);
00055 
00056         int values = osg::StateAttribute::OFF | osg::StateAttribute::PROTECTED;
00057         quad->getOrCreateStateSet()->setAttribute(
00058             new osg::PolygonMode(osg::PolygonMode::FRONT_AND_BACK, osg::PolygonMode::FILL), values);
00059         quad->getOrCreateStateSet()->setMode(GL_LIGHTING, values);
00060         return quad;
00061     }
00062 
00063     ref_ptr<Camera> createHUDCamera(double left, double right, double bottom, double top) {
00064         osg::ref_ptr<Camera> camera = new Camera;
00065         camera->setReferenceFrame(Transform::ABSOLUTE_RF);
00066         camera->setClearMask(GL_DEPTH_BUFFER_BIT);
00067         camera->setRenderOrder(Camera::POST_RENDER);
00068         camera->setAllowEventFocus(false);
00069         camera->setProjectionMatrix(Matrix::ortho2D(left, right, bottom, top));
00070         camera->getOrCreateStateSet()->setMode(GL_LIGHTING, StateAttribute::OFF);
00071         return camera;
00072     }
00073 
00074     ref_ptr<Geometry> createRectangle(double length, double width, int lsteps, int wsteps) {
00075         ref_ptr<Geometry> rect = new Geometry;
00076         ref_ptr<Vec3Array> vertices = new Vec3Array();
00077         ref_ptr<Vec3Array> normals = new Vec3Array();
00078         ref_ptr<DrawElementsUInt> indices = new DrawElementsUInt(GL_TRIANGLE_STRIP);
00079 
00080         double xstep = length / lsteps;
00081         double ystep = width / wsteps;
00082         // current vertex coordinates
00083         double x = 0.0;
00084         double y = 0.0;
00085         double z = 0.0;
00086         //  current normal coordinates
00087         double nx = 0.0;
00088         double ny = 0.0;
00089         double nz = 1.0;
00090 
00091         // set vertices and normals
00092         for (int i = 0; i <= lsteps; i++) {
00093             y = 0.0;
00094             for (int j = 0; j <= wsteps; j++) {
00095                 vertices->push_back(Vec3d(x, y, z));
00096                 normals->push_back(Vec3d(nx, ny, nz));
00097                 y += ystep;
00098             }
00099             x += xstep;
00100         }
00101 
00102         for (int i = 0; i < lsteps; i++) {
00103 
00104             for (int j = 0; j <= wsteps; j++) {
00105                 indices->push_back(i*(wsteps + 1) + j);
00106                 indices->push_back((i + 1)*(wsteps + 1) + j);
00107             }
00108             indices->push_back((i + 1)*(wsteps + 1) + wsteps);
00109             indices->push_back((i + 1)*(wsteps + 1));
00110         }
00111 
00112         rect->setVertexArray(vertices.get());
00113         rect->addPrimitiveSet(indices.get());
00114         rect->setNormalArray(normals.get());
00115         rect->setNormalBinding(Geometry::BIND_PER_VERTEX);
00116 
00117         return rect;
00118     }
00119 
```

```
00120     ref_ptr<Geometry> createRectangleWithTexcoords(double length, double width,
      int lsteps, int wsteps) {
00121         ref_ptr<Geometry> rect = createRectangle(length,  width,  lsteps,  wsteps);
00122 
00123         osg::ref_ptr<osg::Vec2Array> texcoords = new osg::Vec2Array;
00124         double xstep = length / lsteps;
00125         double ystep = width / wsteps;
00126         // current vertex coordinates
00127         double x = 0.0;
00128         double y = 0.0;
00129         double z = 0.0;
00130         //  current normal coordinates
00131         double nx = 0.0;
00132         double ny = 0.0;
00133         double nz = 1.0;
00134         double coordlen = length;
00135         double coordwid = width;
00136         if (width / length < 0.25) coordlen = 1;
00137         else coordlen = length;
00138         if (length / width < 0.25) coordwid = 1;
00139         else coordwid = width;
00140         // set vertices and normals
00141         for (int i = 0; i <= lsteps; i++) {
00142             y = 0.0;
00143             for (int j = 0; j <= wsteps; j++) {
00144                 texcoords->push_back(Vec2(x / coordlen, y / coordwid));
00145                 y += ystep;
00146             }
00147             x += xstep;
00148         }
00149         rect->setTexCoordArray(0, texcoords.get());
00150         return rect;
00151     }
00152 
00153     osg::ref_ptr<osg::Geometry> createBodyOfRotation(double height, int hsteps, int
     rsteps, const BodyOfRotationFunction& function) {
00154         ref_ptr<Geometry> rect = new Geometry;
00155         ref_ptr<Vec3Array> vertices = new Vec3Array();
00156         ref_ptr<Vec3Array> normals = new Vec3Array();
00157         ref_ptr<DrawElementsUInt> indices = new DrawElementsUInt(GL_TRIANGLE_STRIP);
00158         const BodyOfRotationFunction* curFunc = &function;
00159 
00160         double zstep = height / hsteps;
00161         double alphastep = DegreesToRadians(360.0 / rsteps);
00162         // current vertex coordinates
00163         double z = 0.0;
00164         double alpha = 2*PI;
00165         //normal coordinates are calculated on the fly
00166 
00167         // set vertices and normals
00168         for (int i = 0; i <= hsteps; i++) {
00169             alpha = 0;
00170             double curRadius = curFunc->func(z);
00171             for (int j = 0; j <= rsteps; j++) {
00172                 vertices->push_back(Vec3d(
00173                     curRadius * cos(alpha), //X
00174                     curRadius * sin(alpha),//Y
00175                     z));                   //Z
00176                 normals->push_back(Vec3d(
00177                     curRadius * cos(alpha), //X
00178                     curRadius * sin(alpha), //Y
00179                     -curRadius * curFunc->derivation(z))); //Z
00180                 alpha -= alphastep;
00181             }
00182             z += zstep;
00183             if (curFunc->end < z - 1e-8)
00184                 curFunc = curFunc->nextFunc;
00185         }
00186 
00187         //set strip connections
00188         for (int i = 0; i < hsteps; i++) {
00189 
00190             for (int j = 0; j <= rsteps; j++) {
00191                 indices->push_back(i*(rsteps + 1) + j);
00192                 indices->push_back((i + 1)*(rsteps + 1) + j);
00193             }
00194             indices->push_back((i + 1)*(rsteps + 1) + rsteps);
00195             indices->push_back((i + 1)*(rsteps + 1));
00196         }
00197 
00198         //normalize the normals
00199         for (auto cnt = 0; cnt < normals->size(); cnt++) {
00200             normals->at(cnt).normalize();
00201         }
00202 
00203         rect->setVertexArray(vertices);
00204         rect->addPrimitiveSet(indices);
```

```
00205         rect->setNormalArray(normals);
00206         //i know, deprecated, but osg 3.0.1
00207         rect->setNormalBinding(Geometry::BIND_PER_VERTEX);
00208 
00209         return rect;
00210     }
00211 
00212     osg::ref_ptr<osgText::Text> createText(const osg::Vec3& pos, const std::string& content,
    float size) {
00213         osg::ref_ptr<osgText::Font> g_font = osgText::readFontFile("../fonts/dirtydoz.ttf");
00214         osg::ref_ptr<osgText::Text> text = new osgText::Text;
00215         text->setDataVariance(osg::Object::DYNAMIC);
00216         text->setFont(g_font);
00217         text->setCharacterSize(size);
00218         text->setAxisAlignment(osgText::TextBase::XY_PLANE);
00219         text->setPosition(pos);
00220         text->setText(content);
00221         return text;
00222     }
00223 
00224     ref_ptr<LightSource> createLight(const Vec3 &pos, int lightNum, int point, double spotCutoff
    , double spotExponent) {
00225         ref_ptr<LightSource> light = new LightSource;
00226         light->getLight()->setPosition(Vec4(pos.x(), pos.y(), pos.z(), point));
00227         light->getLight()->setAmbient(Vec4(.2, .2, .2, 1));
00228         light->getLight()->setDiffuse(Vec4(.7, .7, .7, 1));
00229         light->getLight()->setSpecular(Vec4(0.7, 0.7, 0.7, 1));
00230         light->getLight()->setLightNum(lightNum);
00231         light->getLight()->setLinearAttenuation(0);
00232         light->getLight()->setQuadraticAttenuation(0.0008);
00233         light->getLight()->setConstantAttenuation(0.000);
00234         light->getLight()->setSpotCutoff(spotCutoff);
00235         light->getLight()->setSpotExponent(spotExponent);
00236         return light;
00237     }
00238 
00239 
00240   osg::ref_ptr<osg::PositionAttitudeTransform> wrapInPositionAttitudeTransform
    (osg::Node * srcNode, const osg::Vec3d& pos){
00241     osg::ref_ptr<osg::PositionAttitudeTransform> newpos = new osg::PositionAttitudeTransform;
00242     newpos->setPosition(pos);
00243     newpos->addChild(srcNode);
00244     return newpos;
00245   }
00246 
00247     ref_ptr<osg::Group> createCuboid(const double length, const double width, const double
    height, const double faktor){
00248 
00249         ref_ptr<Group> cube = new Group();
00250         Geode *cubeSides[6];
00251         MatrixTransform *cubeTrans[6];
00252         for (int i = 0; i < 6; i++){
00253             cubeSides[i] = new Geode();
00254             cubeTrans[i] = new MatrixTransform;
00255 
00256         }
00257         cubeSides[0]->addDrawable(createRectangleWithTexcoords(length, width, (
    int)(length * faktor) + 1, (int)(width * faktor) + 1));
00258         cubeSides[1]->addDrawable(createRectangleWithTexcoords(length, width, (
    int)(length * faktor) + 1, (int)(width * faktor) + 1));
00259 
00260         cubeSides[2]->addDrawable(createRectangleWithTexcoords(height, width, (
    int)(height * faktor) + 1, (int)(width * faktor) + 1));
00261         cubeSides[3]->addDrawable(createRectangleWithTexcoords(height, width, (
    int)(height * faktor) + 1, (int)(width * faktor) + 1));
00262 
00263         cubeSides[4]->addDrawable(createRectangleWithTexcoords(height, length,
    (int)(height * faktor) + 1, (int)(length * faktor) + 1));
00264         cubeSides[5]->addDrawable(createRectangleWithTexcoords(height, length,
    (int)(height * faktor) + 1, (int)(length * faktor) + 1));
00265 
00266         cubeTrans[0]->setMatrix(osg::Matrix::translate(0, 0, height));
00267         cubeTrans[1]->setMatrix(osg::Matrix::rotate((3 / 2)*PI, 1, 0, 0)*osg::Matrix::translate(0.0f, width
    , 0));
00268 
00269         cubeTrans[2]->setMatrix(osg::Matrix::rotate(PI / 2, 0, 1, 0)*osg::Matrix::translate(length, 0,
    height));
00270         cubeTrans[3]->setMatrix(osg::Matrix::rotate((PI / 2) * 3, 0, 1, 0));
00271 
00272         cubeTrans[4]->setMatrix(osg::Matrix::rotate((PI / 2), 0, 0, 1)*osg::Matrix::rotate((PI / 2), 1, 0,
    0)*osg::Matrix::translate(length, 0, 0));
00273         cubeTrans[5]->setMatrix(osg::Matrix::rotate((PI / 2) * 3, 0, 0, 1)*osg::Matrix::rotate((PI / 2) * 3
    , 1, 0, 0)*osg::Matrix::translate(0, width, 0));
00274 
00275         for (int i = 0; i < 6; i++){
00276             cubeTrans[i]->addChild(cubeSides[i]);
00277             cube->addChild(cubeTrans[i]);
```

```
00278         }
00279
00280         return cube.release();
00281
00282     }
00283
00284
00285     void createRenderingPipeline(unsigned int width, unsigned int height, osg::Node&
       rootForToon, osgViewer::Viewer &viewer, RenderingPipeline& pipe, Vec3f& fogColor) {
00286         osg::ref_ptr<brtr::CelShading> toonRoot = new brtr::CelShading;
00287         toonRoot->addChild(&rootForToon);
00288
00289         osg::ref_ptr<osg::Texture2D> toonAndOutline = new osg::Texture2D;
00290         toonAndOutline->setTextureSize(width, height);
00291         toonAndOutline->setInternalFormat(GL_RGBA);
00292         osg::ref_ptr<osg::Camera> rttCamToon = brtr::createRTTCamera(
      osg::Camera::COLOR_BUFFER, toonAndOutline);
00293         rttCamToon->addChild(toonRoot);
00294
00295         //taken from the OSG Beginners Guide
00296         osg::ref_ptr<osg::Texture2D> deepth = new osg::Texture2D;
00297         deepth->setTextureSize(width, height);
00298         deepth->setInternalFormat(GL_DEPTH_COMPONENT24);
00299         deepth->setSourceFormat(GL_DEPTH_COMPONENT);
00300         deepth->setSourceType(GL_FLOAT);
00301
00302         osg::ref_ptr<osg::Camera> rttCamDepth = brtr::createRTTCamera(
      osg::Camera::DEPTH_BUFFER, deepth);
00303         rttCamDepth->addChild(toonRoot);
00304
00305         osg::ref_ptr<Camera> postProcessCam = brtr::createHUDCamera(0, 1, 0, 1);
00306         postProcessCam->addChild(brtr::createScreenQuad(width, height));
00307
00308         osg::ref_ptr<osg::Shader> fogFrag = osgDB::readShaderFile("../Shader/fogShader.frag");
00309         osg::ref_ptr<osg::Shader> fogVert = osgDB::readShaderFile("../Shader/fogShader.vert");
00310         osg::ref_ptr<osg::Program> fogProgram = new osg::Program;
00311         fogProgram->addShader(fogFrag);
00312         fogProgram->addShader(fogVert);
00313
00314         osg::ref_ptr<osg::Shader> sepiaFogFrag = osgDB::readShaderFile("../Shader/sepiaFogShader.frag");
00315         osg::ref_ptr<osg::Program> sepiaFogProgram = new osg::Program;
00316         sepiaFogProgram->addShader(sepiaFogFrag);
00317         sepiaFogProgram->addShader(fogVert);
00318
00319         osg::ref_ptr<osg::Shader> wavesFrag = osgDB::readShaderFile("../Shader/sinShader.frag");
00320         osg::ref_ptr<osg::Program> wavesProgram = new osg::Program;
00321         wavesProgram->addShader(wavesFrag);
00322         wavesProgram->addShader(fogVert);
00323
00324         //creating Program vector, element 0 should be the active one
00325         std::vector<osg::ref_ptr<osg::Program>> programVector;
00326         programVector.push_back(fogProgram);
00327         programVector.push_back(sepiaFogProgram);
00328         programVector.push_back(wavesProgram);
00329
00330         //postprocess Attributs and Mods
00331         postProcessCam->getOrCreateStateSet()->setAttributeAndModes(programVector[0],
      osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON | osg::StateAttribute::PROTECTED);
00332         postProcessCam->getOrCreateStateSet()->setTextureAttributeAndModes(0, toonAndOutline,
      osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON | osg::StateAttribute::PROTECTED);
00333         postProcessCam->getOrCreateStateSet()->addUniform(new osg::Uniform("texture0", 0),
      osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON | osg::StateAttribute::PROTECTED);
00334         postProcessCam->getOrCreateStateSet()->setTextureAttributeAndModes(1, deepth,
      osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON | osg::StateAttribute::PROTECTED);
00335         postProcessCam->getOrCreateStateSet()->addUniform(new osg::Uniform("deepth", 1),
      osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON | osg::StateAttribute::PROTECTED);
00336         postProcessCam->getOrCreateStateSet()->addUniform(new osg::Uniform("fogColor", fogColor),
      osg::StateAttribute::OVERRIDE | osg::StateAttribute::ON | osg::StateAttribute::PROTECTED);
00337
00338         //setting Clipping Pane
00339         float zNear = 0.01, zFar = 100000;
00340         osg::ref_ptr<osg::Uniform> zNearUniform = new osg::Uniform("zNear", zNear);
00341         osg::ref_ptr<osg::Uniform> zFarUniform = new osg::Uniform("zFar", zFar);
00342         postProcessCam->getOrCreateStateSet()->addUniform(zNearUniform);
00343         postProcessCam->getOrCreateStateSet()->addUniform(zFarUniform);
00344         viewer.getCamera()->setComputeNearFarMode(osg::CullSettings::DO_NOT_COMPUTE_NEAR_FAR);
00345         viewer.getCamera()->setProjectionMatrixAsPerspective(70, 1.778, zNear, zFar);
00346
00347         //Setting Pipeline
00348         pipe.pass_0_color = rttCamToon;
00349         pipe.pass_0_depth = rttCamDepth;
00350         pipe.pass_PostProcess = postProcessCam;
00351         pipe.programs = programVector;
00352     }
00353
00354     osg::ref_ptr<osg::Geometry> createBeerBottle() {
00355         brtr::BodyOfRotationFunction seventh = { [](double x) {
```

```
00356             return -18.4*x + 17.02;
00357         }, 0.925, nullptr };
00358         brtr::BodyOfRotationFunction sixth = { [](double x) {
00359             return 0.19* exp(-0.77 * x);
00360         }, 0.92, &seventh };
00361         brtr::BodyOfRotationFunction fifth = { [](double x) {
00362             return 0.8*x - 0.6;
00363         }, 0.865, &sixth };
00364         brtr::BodyOfRotationFunction fourth = { [](double x) {
00365             return 0.092;
00366         }, 0.86, &fifth };
00367         brtr::BodyOfRotationFunction third = { [](double x) {
00368             return 0.22*exp(-1.49 * x);
00369         }, 0.6, &fourth };
00370         brtr::BodyOfRotationFunction second = { [](double x) {
00371             return 0.11;
00372         }, 0.48, &third };
00373         brtr::BodyOfRotationFunction first = { [](double x) {
00374             return 55 * x;
00375         }, 0.002, &second };
00376         return brtr::createBodyOfRotation(0.925, 1000, 50, first);
00377     }
00378
00379     osg::ref_ptr<osg::Geometry> createRealBottle() {
00380         brtr::BodyOfRotationFunction six = { [](double x) {
00381             return -1e3*x + 1280.06;
00382         }, 1.28002, nullptr };
00383         brtr::BodyOfRotationFunction fifth = { [](double x) {
00384             return 0.06;
00385         }, 1.28, &six };
00386         brtr::BodyOfRotationFunction fourth = { [](double x) {
00387             return 0.054;
00388         }, 1.22, &fifth };
00389         brtr::BodyOfRotationFunction third = { [](double x) {
00390             return -9594.216687 *x*x*x*x*x + 41734.84259*x*x*x*x
00391                 - 72461.92057 * x*x*x + 62769.20326*x*x - 27127.83957*x + 4679.866301;
00392         }, 0.94, &fourth };
00393         brtr::BodyOfRotationFunction second = { [](double x) {
00394             return 0.14;
00395         }, 0.8, &third };
00396         brtr::BodyOfRotationFunction first = { [](double x) {
00397             return 1e3 * x;
00398         }, 0.14e-3, &second };
00399
00400
00401         ref_ptr<Geometry> body = brtr::createBodyOfRotation(1.28002, 50, 25,
    first);
00402         ref_ptr<Material> bodyMat = new Material;
00403         bodyMat->setDiffuse(Material::FRONT, Vec4(0.33, 0.23, 0.15, 1));
00404         bodyMat->setAmbient(Material::FRONT, Vec4(0.33, 0.23, 0.15, 1));
00405         bodyMat->setSpecular(Material::FRONT, Vec4(0.7, 0.7, 0.7, 1));
00406         bodyMat->setShininess(Material::FRONT, 42.0);
00407
00408
00409         body->getOrCreateStateSet()->setAttributeAndModes(bodyMat, StateAttribute::ON);
00410         body->getOrCreateStateSet()->addUniform(new Uniform("tex",false), StateAttribute::ON |
    StateAttribute::OVERRIDE);
00411         return body;
00412     }
00413
00414     osg::ref_ptr<osg::Geometry> createVase() {
00415         brtr::BodyOfRotationFunction six = { [](double x) {
00416             return  13.14800902*x*x*x - 29.67693464*x*x + 22.22013524*x - 5.442163787;
00417         }, 0.92, nullptr };
00418         brtr::BodyOfRotationFunction fifth = { [](double x) {
00419             return -7723.99737*pow(x, 6) + 34421.33266*pow(x, 5) - 63445.68073*pow(x, 4)
00420                 + 61911.20624*x*x*x - 33732.12852*x*x + 9729.204284*x - 1160.328634;
00421         }, 0.7, &six };
00422         brtr::BodyOfRotationFunction fourth = { [](double x) {
00423             return -6.74046e-4*pow(x, 7) + 22500.00265*pow(x, 6) - 67500.00438*pow(x, 5)
00424                 + 83825.00396*pow(x, 4) - 55150.00212*x*x*x + 20268.00067*x*x - 3943.000117*x + 317.1600086
    ;
00425         }, 0.6, &fifth };
00426         brtr::BodyOfRotationFunction third = { [](double x) {
00427             return -2.46772e-4*pow(x, 7) - 22499.99949*pow(x, 6) + 40499.99956*pow(x, 5)
00428                 - 29824.99979*pow(x, 4) + 11489.99994*x*x*x - 2435.99999*x*x + 268.1999991*x - 11.79999997;
00429         }, 0.4, &fourth };
00430         brtr::BodyOfRotationFunction second = { [](double x) {
00431             return 91.2477433*pow(x, 7) + 22479.62608*pow(x, 6) - 13517.78749*pow(x, 5) + 2833.488659*pow(x
    , 4)
00432                 - 231.3692301*x*x*x + 0.0830483429*x*x + 0.999990044*x + 0.1198800004;
00433         }, 0.2, &third };
00434         brtr::BodyOfRotationFunction first = { [](double x) {
00435             return 1e3 * x;
00436         }, 0.12e-3, &second };
00437
00438
```

```
00439         ref_ptr<Geometry> body = brtr::createBodyOfRotation(0.92, 50, 25, first);
00440         ref_ptr<Material> bodyMat = new Material;
00441         bodyMat->setDiffuse(Material::FRONT, Vec4(0.0754, 0.3529, 0.58, 1));
00442         bodyMat->setAmbient(Material::FRONT, Vec4(0.0754, 0.3529, 0.58, 1));
00443         bodyMat->setSpecular(Material::FRONT, Vec4(0.7, 0.7, 0.7, 1));
00444         bodyMat->setShininess(Material::FRONT, 42.0);
00445
00446
00447         body->getOrCreateStateSet()->setAttributeAndModes(bodyMat, StateAttribute::ON);
00448         body->getOrCreateStateSet()->addUniform(new Uniform("tex", false), StateAttribute::ON |
      StateAttribute::OVERRIDE);
00449
00450         return body;
00451     }
00452
00453     osg::ref_ptr<osg::Geometry> createStalk() {
00454         brtr::BodyOfRotationFunction third = { [](double x) {
00455             return -1e3*x + 1200.02;
00456         }, 1.20002, nullptr };
00457         brtr::BodyOfRotationFunction second = { [](double x) {
00458             return 0.015;
00459         }, 1.2, &third };
00460         brtr::BodyOfRotationFunction first = { [](double x) {
00461             return 1e3 * x;
00462         }, 0.02e-3, &second };
00463
00464
00465         ref_ptr<Geometry> body = brtr::createBodyOfRotation(1.20002, 50, 25,
      first);
00466         ref_ptr<Material> bodyMat = new Material;
00467         bodyMat->setDiffuse(Material::FRONT, Vec4(0, 0.43, 0.0215, 1));
00468         bodyMat->setAmbient(Material::FRONT, Vec4(0, 0.43, 0.0215, 1));
00469         bodyMat->setSpecular(Material::FRONT, Vec4(0.1, 0.1, 0.1, 1));
00470         bodyMat->setShininess(Material::FRONT, 100.0);
00471
00472
00473         body->getOrCreateStateSet()->setAttributeAndModes(bodyMat, StateAttribute::ON);
00474         body->getOrCreateStateSet()->addUniform(new Uniform("tex", false), StateAttribute::ON |
      StateAttribute::OVERRIDE);
00475
00476         return body;
00477     }
00478
00479     osg::ref_ptr<osg::Geometry> createBud() {
00480         brtr::BodyOfRotationFunction second = { [](double x) {
00481             return  -1.5625 * x*x + 0.25 * x + 0.03;
00482         }, 0.24, nullptr };
00483         brtr::BodyOfRotationFunction first = { [](double x) {
00484             return -6.25 * x *x + x;
00485         }, 0.08, &second };
00486
00487
00488         ref_ptr<Geometry> body = brtr::createBodyOfRotation(0.24, 50, 25, first);
00489         ref_ptr<Material> bodyMat = new Material;
00490         bodyMat->setDiffuse(Material::FRONT, Vec4(0.8, 0.008, 0.4304, 1));
00491         bodyMat->setAmbient(Material::FRONT, Vec4(0.8, 0.008, 0.4304, 1));
00492         bodyMat->setSpecular(Material::FRONT, Vec4(0.1, 0.1, 0.1, 1));
00493         bodyMat->setShininess(Material::FRONT, 100.0);
00494
00495
00496         body->getOrCreateStateSet()->setAttributeAndModes(bodyMat, StateAttribute::ON);
00497         body->getOrCreateStateSet()->addUniform(new Uniform("tex", false), StateAttribute::ON |
      StateAttribute::OVERRIDE);
00498
00499         return body;
00500     }
00501
00502     osg::ref_ptr<osg::Geometry> createChessFigure() {
00503         brtr::BodyOfRotationFunction thirteen = { [](double x) {
00504             return  -1e3*x + 8000.44;
00505         }, 8.00044, nullptr };
00506         brtr::BodyOfRotationFunction twelve = { [](double x) {
00507             return sqrt(0.375 *0.375 - (x - 7.625)*(x - 7.625)) + 0.44;
00508         }, 8, &thirteen };
00509         brtr::BodyOfRotationFunction eleventh = { [](double x) {
00510             return 0.44;
00511         }, 7.25, &twelve };
00512         brtr::BodyOfRotationFunction tenth = { [](double x) {
00513             return 0.4*x - 2.2;
00514         }, 7, &eleventh };
00515         brtr::BodyOfRotationFunction ninth = { [](double x) {
00516             return  -(1 + 1 / 3)*x*x + (17.6 + 1 / 30)*x - 57.5;
00517         }, 7, &eleventh };
00518         brtr::BodyOfRotationFunction eight = { [](double x) {
00519             return   -2.25*x*x + 29.7*x - 97.2;
00520         }, 6.6, &tenth };
00521         brtr::BodyOfRotationFunction seventh = { [](double x) {
```

```
00522             return 0.44;
00523         }, 6.2, &eight };
00524         brtr::BodyOfRotationFunction six = { [](double x) {
00525             return  -1e3*x + 4200.96;
00526         }, 4.20052, &seventh };
00527         brtr::BodyOfRotationFunction fifth = { [](double x) {
00528             return  0.96;
00529         }, 4.2, &six };
00530         brtr::BodyOfRotationFunction fourth = { [](double x) {
00531             return 3.59724403e-58*exp(32.90396023*x) + 0.44;
00532         }, 4, &fifth };
00533         brtr::BodyOfRotationFunction third = { [](double x) {
00534             return 0.44;
00535         }, 3.6, &fourth };
00536         brtr::BodyOfRotationFunction second = { [](double x) {
00537             return sqrt(1.44*1.44 - x*x) + 0.44;
00538         }, 1.44, &third };
00539         brtr::BodyOfRotationFunction first = { [](double x) {
00540             return 1e3 * x;
00541         }, 0.00188, &second };
00542 
00543         ref_ptr<Geometry> body = brtr::createBodyOfRotation(8.00044, 1000, 50,
    first);
00544         ref_ptr<Material> bodyMat = new Material;
00545         bodyMat->setDiffuse(Material::FRONT, Vec4(0.84, 0.238, 0.0, 1));
00546         bodyMat->setAmbient(Material::FRONT, Vec4(0.84, 0.238, 0.0, 1));
00547         bodyMat->setSpecular(Material::FRONT, Vec4(0.7, 0.7, 0.7, 1));
00548         bodyMat->setShininess(Material::FRONT, 42.0);
00549 
00550 
00551         body->getOrCreateStateSet()->setAttributeAndModes(bodyMat, StateAttribute::ON);
00552         body->getOrCreateStateSet()->addUniform(new Uniform("tex", false), StateAttribute::ON |
    StateAttribute::OVERRIDE);
00553 
00554         return body;
00555     }
00556 
00557     extern osg::ref_ptr<osg::Geode> createCrosshair(unsigned int width, unsigned int height)
     {
00558         ref_ptr<osg::Geode>         geode = new osg::Geode;
00559         ref_ptr<osg::Geometry>    geom = new Geometry;;
00560         ref_ptr<osg::Vec3Array>   vertices =new osg::Vec3Array;
00561         geode->addDrawable(geom);
00562 
00563         vertices->push_back(osg::Vec3(width / 2.0 - 10, height / 2.0 - 1, 0.0));
00564         vertices->push_back(osg::Vec3(width / 2.0 - 10, height / 2.0 + 1, 0.0));
00565         vertices->push_back(osg::Vec3(width / 2.0 + 10, height / 2.0 + 1, 0.0));
00566         vertices->push_back(osg::Vec3(width / 2.0 + 10, height / 2.0 - 1, 0.0));
00567 
00568         vertices->push_back(osg::Vec3(width / 2.0 - 1, height / 2.0 - 10, 0.0));
00569         vertices->push_back(osg::Vec3(width / 2.0 - 1, height / 2.0 + 10, 0.0));
00570         vertices->push_back(osg::Vec3(width / 2.0 + 1, height / 2.0 + 10, 0.0));
00571         vertices->push_back(osg::Vec3(width / 2.0 + 1, height / 2.0 - 10, 0.0));
00572         geom->setVertexArray(vertices);
00573 
00574         // set colors
00575         ref_ptr<osg::Vec4Array> colors = new osg::Vec4Array;
00576         colors->push_back(osg::Vec4(0.55, 0.55, 0.55, 1.0));
00577         geom->setColorArray(colors);
00578         geom->setColorBinding(Geometry::BIND_OVERALL);
00579         geom->addPrimitiveSet(new osg::DrawArrays(GL_QUADS, 0, 8));
00580         return geode;
00581     }
00582 
00583     osg::ref_ptr<osg::PositionAttitudeTransform> createVaseWithFlower() {
00584         ref_ptr<Geode> bottom = new Geode;
00585         bottom->addDrawable(brtr::createVase());
00586         ref_ptr<Geode> inner = new Geode;
00587         inner->addDrawable(brtr::createStalk());
00588         ref_ptr<Geode> knospe = new Geode;
00589         knospe->addDrawable(brtr::createBud());
00590         ref_ptr<PositionAttitudeTransform> budTranslate = new PositionAttitudeTransform;
00591         budTranslate->addChild(knospe);
00592         budTranslate->setPosition(Vec3(0, 0, 1.0));
00593         ref_ptr<PositionAttitudeTransform> vaseFlower = new PositionAttitudeTransform;
00594         vaseFlower->addChild(bottom);
00595         vaseFlower->addChild(inner);
00596         vaseFlower->addChild(budTranslate);
00597         return vaseFlower;
00598     }
00599     osg::ref_ptr<osg::Material> createSimpleMaterial(osg::Material::Face face, const
    osg::Vec4& diffuse, const osg::Vec4& ambient, const osg::Vec4& specular, const double shininess) {
00600         ref_ptr<Material> mat = new osg::Material;
00601         mat->setDiffuse(face, osg::Vec4(.7f, .7f, .7f, 1.0f));
00602         mat->setAmbient(face, osg::Vec4(.3f, .3f, .3f, 1.0f));
00603         mat->setSpecular(face, osg::Vec4(.9f, .9f, .9f, 1.0f));
00604         mat->setShininess(face, shininess);
```

```
00605         return mat;
00606     }
00607
00608     osg::Vec3 getDimensionOfNode(Node * source) {
00609         ComputeBoundsVisitor cbbv;
00610         source->accept(cbbv);
00611         BoundingBox bb = cbbv.getBoundingBox();
00612         Vec3 size = bb._max - bb._min;
00613         return size;
00614     }
00615
00616     osg::ref_ptr<osg::Texture2D> createToonTex(std::string toonTex) {
00617         osg::ref_ptr<osg::Texture2D> toonTexture = new osg::Texture2D;
00618         toonTexture->setImage(osgDB::readImageFile("../BlenderFiles/Texturen/toons/" + toonTex));
00619         toonTexture->setFilter(osg::Texture::MIN_FILTER, osg::Texture::NEAREST);
00620         toonTexture->setFilter(osg::Texture::MAG_FILTER, osg::Texture::NEAREST);
00621         return toonTexture;
00622     }
00623
00624 }
```