PA9 Air Hockey - Ben Nicholes, Denis Morozov, Saharath Kleips

AUTHOR

Version

11/09/2015

Table of Contents

Table of contents

PA9 Air Hockey

This project showcases a physics-based air hockey video game.

Author:

Ben Nicholes, Denis Morozov, Saharath Kleips

Version:

1.00

Bug Encounters / Error Fixes

Return to README

Table of Contents

"Not Found in Mesa Table OpenGL Error"

Found on: VirtualBox 5.0.2r102096 Host: Windows 7 Ultimate x64 Guest: Ubuntu 14.04 LTS x64 (Guest Additions Installed)

VirtualBox 5.0.2r102096 Host: Windows 8.1 x64 Guest: Ubuntu 12.04 LTS x64 (Guest Additions Installed)

Error: Check `glxinfo` : ``` glxinfo | grep OpenGL `` `glxinfo requires` mesa-utils`installed using:` `` sudo apt-get install mesa-utils ``` The output should resemble the following and contain Chromium and not Mesa: ``` OpenGL vendor string: Humper OpenGL renderer string: Chromium OpenGL version string: 2.1 Chromium OpenGL shading language version string: 4.50 NVIDIA OpenGL extensions: ```

Solution: A fix is disabling 3D Hardware Acceleration on your VirtualBox. After disabling, your glxinfo should look similar to: ``` OpenGL vendor string: VMWare, Inc. OpenGL renderer string: Gallium 0.4 on llvmpipe (LLVM 3.6 256 bits) OpenGL version string: 3.0 Mesa 10.5.2 OpenGL shading language version string: 1.30 OpenGL context flags: (none) OpenGL extensions: ``` Note: The version string should now contain Mesa rather than Chromium.

Alternative Solution: Download and install VMWare Workstation Player 12. Migrate your current virtual machine image and boot from VMWare. Install VMWare tools.

Segmentation Fault after Team Select: A very rare occurence happens after team select resulting in a segmentation fault. The segmentation fault is in QSignalMapper::map(QObject*) () This is an rare occurence and running the program again should bypass the issue.

Build Instructions

Return to README

Table of Contents

Build Executable

Navigate to the `build` directory. ``` qmake make `` `**Note:**` qmake`does not need to be run before` make`everytime, only when the` .pro`file has been changed. Otherwise, simply` qmake`once and` make` from there on out.

Execute Program

Navigate to the `bin` directory. ``` ./AirHockey.exe ```

Clean Compilation Files

``` make clean ```
```

Installation Instructions

Return to README **It is recommended that you use Ubuntu 14.04 LTS.**

Table of Contents

Ubuntu 14.04 LTS

Tested with a clean install of Ubuntu 14.04 x64 LTS on VMWare Workstation Player with VMWare Tools installed and Windows 7 Ultimate x64 host.

g++ Install ``` sudo apt-get install build-essential ```

OpenGL Install ``` sudo apt-get install libgl1-mesa-dev ```

Qt 5.5 Install Add the Qt5.5 PPAs by Stephan Banner and then install the latest Qt files. ``` sudo apt-add-repository ppa:beineri/opt-qt55-trusty sudo apt-get update sudo apt-get install qt-latest ```

Set Up Development Environment Your `qtchooser` should now list opt-qt55 as an option. ``` qtchooser -list-versions ``` Add the environment variables to your bashrc file. ``` echo 'export QT_SELECT=opt-qt55' >> ~/.bashrc ``` Restart your terminal.

Assimp Install ``` sudo apt-get install libassimp-dev ```

Gstreamer Install sudo add-apt-repository ppa:gstreamer-developers/ppa sudo apt-get update sudo apt-get install gstreamer1.0

Ubuntu 12.04 LTS

Tested with a clean install of Ubuntu 12.04 x64 LTS on VirtualBox 5.0.2r102096 with Guest Additions installed and Windows 8.1 x64 host.

g++ Install ``` sudo apt-get install build-essential ```

OpenGL Install ``` sudo apt-get install libgl1-mesa-dev ```

Qt 5.5 Install Add the Qt5.5 PPAs by Stephan Banner and then install the latest Qt files. ``` sudo apt-add-repository ppa:beineri/opt-qt55 sudo apt-get update sudo apt-get install qt-latest ```

Set Up Development Environment Add the environment variables to your bashrc file. ``` cd /opt/qt55/bin ./qt55-env.sh ``` Restart your terminal.

WARNING: Potentially Unsupported Assimp Install ``` sudo apt-get install libassimp-dev ```

Other Linux Distributions

Qt can be built from source code, and is most likely available in your distribution's repositories. However, an easy way of getting the latest Qt is using the official installer:

Download the Qt Installer. Give it executable permissions and run as administrator. ``` chmod +x qt-unified-linux-x64-2.0.2-2-online.run sudo ./qt-unified-linux-x64-2.0.2-2-online.run ``` Add the

Qt binaries to PATH environment variable (assumes default install location of /opt/Qt/: ``` echo "PATH=\$PATH:/opt/Qt/Tools/QtCreator/bin:/opt/Qt/5.5/gcc_64/bin" >> ~/.bashrc source ~/.bashrc ```

... and that's it!

OSX

Install VirtualBox. Go get Ubuntu. Brew instructions unavailable.

Windows

Really?

PA9 - Air Hockey (Ben Nicholes, Denis Morozov, Saharath Kleips)

This project showcases a physics-based air hockey video game.

Notes

Single Player is disabled
GStreamer is required for audio to work, see Installation Instructions.

Extra Credit

Sound for Hockey Table, Puck Collisions, and Goals
Main Menu
Team Select
2 Players
Text2D display
Game Restart

Table of Contents

Instructions

When first starting up the game you are presented with the game's title screen as shown below. You may also change the resolution of the game by dragging the window at the corners. Here you are allowed to select from Single Player (disabled) and Versus game modes, as well as the option to exit the game. Upon selecting either "Single Player" or "Versus Mode" you will be presented with the Team Select Screen as shown below. All 30 of your favorite NHL teams may be selected to play as and against! Clicking on a team's logo will select the first player's team, another click on a team's logo will select the second player's team. The game will then begin with the default Player 2 Perspective shown below. Note there is a possiblity for the game to run into a segmentation fault after selecting two teams. Simply re-run the game until it works, trust us, it works. More information on the bug can be found in bugs.md. Here each player may control their respective paddles and attempt to hit the puck into the opposing player's goal. Upon scoring, the puck will reset and a new round will begin. There is no upper limit, so play to your heart's content!

Controls

Player 1 Controls

J Key
Move Player Two Puck Left
L Arrow
Move Player Two Puck Right
I Arrow
Move Player Two Puck Up
K Arrow
Move Player Two Puck Down </tbody>
Player 2 Controls

Left Arrow
Move Player One Puck Left
Right Arrow
Move Player One Puck Right
Up Arrow
Move Player One Puck Up
Down Arrow
Move Player One Puck Down </tbody>
Program Controls

Action
Description
RMB + mouse movement
Rotate camera
Q
Move camera down
E
Move camera up
W
Move camera forward
A
Move camera left
S
Move camera backwards
D
Move camera right
Ctrl + P
Pause Game
Ctrl + R
Restart Game
Ctrl + Q
Quit the Program
Ctrl + F1
Switch Camera to Player 1 Perspective
Ctrl + F2
Switch Camera to Player 2 Perspective

Ctrl + F3
Switch Camera to Side of the Table
Ctrl + F4
Switch Camera to Top of Table Perspective </tbody>

Namespace Index

Namespace List

Here is a list of all namespaces with brief descriptions:

Hierarchical Index

Class Hierarchy

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

Class Index

Class List

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

File Index

File List

Here is a list of all files with brief descriptions:

Namespace Documentation

anonymous_namespace{qrc_font_resource.cpp} Namespace Reference

Classes
struct initializer

Variables

struct anonymous_namespace{qrc_font_resource.cpp}::initializer dummy

---

Variable Documentation

struct anonymous_namespace{qrc_font_resource.cpp}::initializer anonymous_namespace{qrc_font_resource.cpp}::dummy

anonymous_namespace{qrc_shader_resource.cpp} Namespace Reference

Classes
struct initializer

Variables

struct anonymous_namespace{qrc_shader_resource.cpp}::initializer dummy

---

Variable Documentation

struct anonymous_namespace{qrc_shader_resource.cpp}::initializer
anonymous_namespace{qrc_shader_resource.cpp}::dummy

ModelLoader Namespace Reference

Functions

bool loadUVModel (QString filePath, UVVertex *&geometry, int &numVertices)

Loads a UV-based model based on a file path.

bool loadColorModel (QString filePath, ColorVertex *&geometry, int &numVertices)

Loads a material-based model based on a file path.

bool loadTriMesh (QString filePath, btTriangleMesh *&collisionMesh)

Function Documentation

bool ModelLoader::loadColorModel (QString *filePath*, ColorVertex *& *geometry*, int & *numVertices*)

Loads a material-based model based on a file path.

Parameters:

in	*filePath*	The full path to the model source.
out	*geometry*	The geometry data of the color model loaded.
out	*numVertices*	The number of vertices within the color model.

Returns:

True of the model loaded successfully, false otherwise.

bool ModelLoader::loadTriMesh (QString *filePath*, btTriangleMesh *& *collisionMesh*)

bool ModelLoader::loadUVModel (QString *filePath*, UVVertex *& *geometry*, int & *numVertices*)

Loads a UV-based model based on a file path.

Parameters:

in	*filePath*	The full path to the model source.
out	*geometry*	The geometry data of the uv model loaded.
out	*numVertices*	The number of vertices within the uv model.

Returns:

True of the model loaded successfully, false otherwise.

Class Documentation

Camera3D Class Reference

`#include <camera3d.h>`

Public Member Functions

Camera3D ()
Default constructor.

void translate (const QVector3D &dt)
Moves the matrix based on its current location.

void translate (float dx, float dy, float dz)
Overloaded translate function.

void rotate (const QQuaternion &dr)
Rotates the matrix based on its current rotation.

void rotate (float angle, const QVector3D &axis)
Overloaded rotate function.

void rotate (float angle, float ax, float ay, float az)
Overloaded rotate function.

void setTranslation (const QVector3D &t)
Sets the matrix to a new location.

void setTranslation (float x, float y, float z)
Overloaded setTranslation function.

void setRotation (const QQuaternion &r)
Sets the matrix to a new rotation.

void setRotation (float angle, const QVector3D &axis)
Overloaded setRotation function.

void setRotation (float angle, float ax, float ay, float az)
Overloaded setRotation function.

const QVector3D & translation () const
Getter function for m_translation.

const QQuaternion & rotation () const
Getter function for m_rotation.

const QMatrix4x4 & toMatrix ()
Getter function for m_world.

QVector3D forward () const
The relative forward position of this matrix.

QVector3D up () const
The relative up position of this matrix.

QVector3D right () const
The relative right position of this matrix.

Static Public Attributes

static const QVector3D LocalForward
static const QVector3D LocalUp
static const QVector3D LocalRight

Private Attributes

QVector3D m_translation
QQuaternion m_rotation
QMatrix4x4 m_world

Constructor & Destructor Documentation

Camera3D::Camera3D ()

Default constructor.

Member Function Documentation

QVector3D Camera3D::forward () const

The relative forward position of this matrix.

Returns:

A foward vector based on this matrix.

QVector3D Camera3D::right () const

The relative right position of this matrix.

Returns:

A right vector based on this matrix.

void Camera3D::rotate (const QQuaternion & *dr*)

Rotates the matrix based on its current rotation.

Parameters:

in	*dr*	The rotation to add onto the current matrix.

void Camera3D::rotate (float *angle*, const QVector3D & *axis*)

Overloaded rotate function.

See Also:

Camera3D::rotate(const QQuaternion& dr)

Parameters:

in	*angle*	The angle to add onto the current angle.
in	*axis*	The axis of rotation for the angle.

void Camera3D::rotate (float *angle*, float *ax*, float *ay*, float *az*)

Overloaded rotate function.

See Also:

Camera3D::rotate(const QQuaternion& dr)

Parameters:

in	*angle*	The angle to add onto the current angle.
in	*ax*	The x-axis of rotation.
in	*ay*	The y-axis of rotation.
in	*az*	The z-axis of rotation.

const QQuaternion & Camera3D::rotation () const

Getter function for m_rotation.

Returns:

The current rotation of the matrix.

void Camera3D::setRotation (const QQuaternion & *r*)

Sets the matrix to a new rotation.

Parameters:

in	*r*	The new rotation.

void Camera3D::setRotation (float *angle*, const QVector3D & *axis*)

Overloaded setRotation function.

See Also:

Camera3D::setRotation(const QQuaternion& r)

Parameters:

in	*angle*	The new angle of rotation.
in	*axis*	The axis of rotation for the new angle.

void Camera3D::setRotation (float *angle*, float *x*, float *y*, float *z*)

Overloaded setRotation function.

See Also:

Camera3D::setRotation(const QQuaternion& r)

Parameters:

in	*angle*	The new angle of rotation.
in	*x*	The x-axis of rotation for the new angle.
in	*y*	The y-axis of rotation for the new angle.
in	*z*	The z-axis of rotation for the new angle.

void Camera3D::setTranslation (const QVector3D & *t*)

Sets the matrix to a new location.

Parameters:

in	*t*	The new location.

void Camera3D::setTranslation (float *x*, float *y*, float *z*)

Overloaded setTranslation function.

See Also:

Camera3D::setTranslation(const QVector3D& t)

Parameters:

in	*x*	The new x-axis location.
in	*y*	The new y-axis location.
in	*z*	The new z-axis location.

const QMatrix4x4 & Camera3D::toMatrix ()

Getter function for m_world.

Returns:

The full matrix with current translations.

void Camera3D::translate (const QVector3D & *dt*)

Moves the matrix based on its current location.

Parameters:

in	*dt*	The location to add onto the current matrix.

void Camera3D::translate (float *dx*, float *dy*, float *dz*)

Overloaded translate function.

See Also:

Camera3D::translate(const QVector3D& dt)

Parameters:

in	*dx*	X-axis offset.
in	*dy*	Y-axis offset.
in	*dz*	Z-axis offset.

const QVector3D & Camera3D::translation () const

Getter function for m_translation.

Returns:

The current translation of the matrix.

QVector3D Camera3D::up () const

The relative up position of this matrix.

Returns:

An up vector based on this matrix.

Member Data Documentation

const QVector3D Camera3D::LocalForward[static]

const QVector3D Camera3D::LocalRight[static]

const QVector3D Camera3D::LocalUp[static]

QQuaternion Camera3D::m_rotation[private]

QVector3D Camera3D::m_translation[private]

QMatrix4x4 Camera3D::m_world[private]

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

1. src/Controls/camera3d.h
2. src/Controls/camera3d.cpp

ColorEntity Class Reference

`#include <colorEntity.h>`
Inheritance diagram for ColorEntity:

IMAGE

Public Attributes

Transform3D GTransform

Protected Member Functions

ColorEntity (QString pathToModel)
Constructor for ColorEntity.

~ColorEntity ()
Destructor for ColorEntity.

void initializeGL ()
Initializes the OpenGL data for drawing.

void paintGL (Camera3D &camera, QMatrix4x4 &projection)
Draws the object to the screen.

virtual void update ()
Virtual update function.

void teardownGL ()
Helper function to delete OpenGL data.

Protected Attributes

QOpenGLBuffer * m_vbo
QOpenGLVertexArrayObject * m_vao
QOpenGLShaderProgram * m_program
QString m_pathToModel
ColorVertex * m_model
int m_numVertices
const QString PATH_TO_V_SHADER = ":/shader/colorShader.vs"
const QString PATH_TO_F_SHADER = ":/shader/colorShader.fs"
int m_modelWorld
int m_worldEye
int m_eyeClip

Additional Inherited Members

Constructor & Destructor Documentation

ColorEntity::ColorEntity (QString *pathToModel*)`[protected]`

Constructor for ColorEntity.

Parameters:

in	*pathToModel*	The path the this object's model.

ColorEntity::~ColorEntity ()[protected]

Destructor for ColorEntity.

Member Function Documentation

void ColorEntity::initializeGL ()[protected], [virtual]

Initializes the OpenGL data for drawing.

Implements Renderable.

void ColorEntity::paintGL (Camera3D & *camera*, QMatrix4x4 & *projection*)[protected], [virtual]

Draws the object to the screen.

Parameters:

camera	The camera of the world.
projection	The projection of the world.

Implements Renderable.

Reimplemented in ColorPhysicsEntity.

void ColorEntity::teardownGL ()[protected], [virtual]

Helper function to delete OpenGL data.

Implements Renderable.

void ColorEntity::update ()[protected], [virtual]

Virtual update function.

Implements Renderable.

Reimplemented in ColorPhysicsEntity, and HockeyTable.

Member Data Documentation

Transform3D ColorEntity::GTransform

int ColorEntity::m_eyeClip[protected]

ColorVertex* ColorEntity::m_model[protected]

int ColorEntity::m_modelWorld[protected]

int ColorEntity::m_numVertices[protected]

QString ColorEntity::m_pathToModel[protected]

QOpenGLShaderProgram* ColorEntity::m_program[protected]

QOpenGLVertexArrayObject* ColorEntity::m_vao[protected]

QOpenGLBuffer* ColorEntity::m_vbo[protected]

int ColorEntity::m_worldEye[protected]

const QString ColorEntity::PATH_TO_F_SHADER = ":/shader/colorShader.fs"[protected]

const QString ColorEntity::PATH_TO_V_SHADER = ":/shader/colorShader.vs"[protected]

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

3 src/Entity/colorEntity.h
4 src/Entity/colorEntity.cpp

ColorPhysicsEntity Class Reference

`#include <colorPhysicsEntity.h>`
Inheritance diagram for ColorPhysicsEntity:
IMAGE

Public Attributes

btRigidBody * RigidBody
QMatrix4x4 BTransform

Protected Member Functions

ColorPhysicsEntity (btTransform startingState, btScalar mass, QString pathToModel)
Constructor for ColorPhysicsEntity.

~ColorPhysicsEntity ()
Destructor for ColorPhysicsEntity.

void paintGL (Camera3D &camera, QMatrix4x4 &projection)
Overloaded paintGL function.

void update ()
Updates the RigidBody based on the dynamics world it is in.

Private Attributes

btTriangleMesh * m_triMesh
btCollisionShape * m_collisionShape
btDefaultMotionState * m_motionState
btVector3 m_inertia
btScalar m_mass
btRigidBody::btRigidBodyConstructionInfo * m_rigidBodyCI

Additional Inherited Members

Constructor & Destructor Documentation

ColorPhysicsEntity::ColorPhysicsEntity (btTransform *startingState*, btScalar *mass*, QString *pathToModel*)`[protected]`

Constructor for ColorPhysicsEntity.

Parameters:

in	*startingState*	The location and rotation of the object.
in	*mass*	The mass of the object.
in	*pathToModel*	The path to the object's model.

ColorPhysicsEntity::~ColorPhysicsEntity ()[protected]

Destructor for ColorPhysicsEntity.

Member Function Documentation

void ColorPhysicsEntity::paintGL (Camera3D & *camera*, QMatrix4x4 & *projection*) [protected], [virtual]

Overloaded paintGL function.

Draws using a BTransform instead of a GTransform.

Parameters:

camera	The camera of the world.
projection	The projection of the world.

Reimplemented from ColorEntity.

void ColorPhysicsEntity::update ()[protected], [virtual]

Updates the RigidBody based on the dynamics world it is in.

Reimplemented from ColorEntity.

Reimplemented in HockeyTable.

Member Data Documentation

QMatrix4x4 ColorPhysicsEntity::BTransform

btCollisionShape* ColorPhysicsEntity::m_collisionShape`[private]`

btVector3 ColorPhysicsEntity::m_inertia`[private]`

btScalar ColorPhysicsEntity::m_mass`[private]`

btDefaultMotionState* ColorPhysicsEntity::m_motionState`[private]`

**btRigidBody::btRigidBodyConstructionInfo*
ColorPhysicsEntity::m_rigidBodyCI`[private]`**

btTriangleMesh* ColorPhysicsEntity::m_triMesh`[private]`

btRigidBody* ColorPhysicsEntity::RigidBody

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

5 src/Entity/colorPhysicsEntity.h
6 src/Entity/colorPhysicsEntity.cpp

ColorVertex Class Reference

`#include <colorVertex.h>`

Public Member Functions

ColorVertex ()
Default constructor for ColorVertex.

ColorVertex (const QVector3D &position)
Overloaded constructor for ColorVertex.

ColorVertex (const QVector3D &position, const QVector4D &color)
Overloaded constructor for ColorVertex.

const QVector3D & position () const
Gets the position.

const QVector4D & color () const
Gets the color.

void setPosition (const QVector3D &position)
Sets position to a new position.

void setColor (const QVector4D &color)
Sets color to a new color.

Static Public Member Functions

static int positionOffset ()
Calculates the position offset within ColorVertex.

static int colorOffset ()
Calcualtes the uv offset within ColorVertex.

static int stride ()
Calculates the stride of ColorVertex.

Static Public Attributes

static const int PositionTupleSize = 3
static const int ColorTupleSize = 4

Private Attributes

QVector3D m_position
QVector4D m_color

Constructor & Destructor Documentation

ColorVertex::ColorVertex ()

Default constructor for ColorVertex.

ColorVertex::ColorVertex (const QVector3D & *position*)[explicit]

Overloaded constructor for ColorVertex.

Parameters:

in	*position*	The position data for the ColorVertex.

ColorVertex::ColorVertex (const QVector3D & *position*, const QVector4D & *color*)

Overloaded constructor for ColorVertex.

Parameters:

in	*position*	The position data for the ColorVertex.
in	*color*	The color data for the ColorVertex.

Member Function Documentation

const QVector4D & ColorVertex::color () const

Gets the color.

Returns:

The color.

int ColorVertex::colorOffset ()[static]

Calcualtes the uv offset within ColorVertex.

Returns:

The uv offset.

const QVector3D & ColorVertex::position () const

Gets the position.

Returns:

The position.

int ColorVertex::positionOffset ()[static]

Calculates the position offset within ColorVertex.

Returns:

The position offset.

void ColorVertex::setColor (const QVector4D & *color*)

Sets color to a new color.

Parameters:

in	*color*	The new color.

void ColorVertex::setPosition (const QVector3D & *position*)

Sets position to a new position.

Parameters:

in	*position*	The new position.

int ColorVertex::stride ()[static]

Calculates the stride of ColorVertex.

Returns:

The stride of ColorVertex.

Member Data Documentation

const int ColorVertex::ColorTupleSize = 4[static]

QVector4D ColorVertex::m_color[private]

QVector3D ColorVertex::m_position[private]

const int ColorVertex::PositionTupleSize = 3[static]

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

7 src/3D/colorVertex.h

8 src/3D/colorVertex.cpp

OGLWidget::Goal2Callback Struct Reference

Inheritance diagram for OGLWidget::Goal2Callback:

IMAGE

Public Member Functions

Goal2Callback (OGLWidget *scopePtr)

btScalar addSingleResult (btManifoldPoint &cp, const btCollisionObjectWrapper *colObj0Wrap, int partId0, int index0, const btCollisionObjectWrapper *colObj1Wrap, int partId1, int index1)

Public Attributes

OGLWidget * context

Constructor & Destructor Documentation

OGLWidget::Goal2Callback::Goal2Callback (OGLWidget * *scopePtr*)`[inline]`

Member Function Documentation

btScalar OGLWidget::Goal2Callback::addSingleResult (btManifoldPoint & *cp*, const btCollisionObjectWrapper * *colObj0Wrap*, int *partId0*, int *index0*, const btCollisionObjectWrapper * *colObj1Wrap*, int *partId1*, int *index1*)`[inline]`

Member Data Documentation

OGLWidget* OGLWidget::Goal2Callback::context

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

9 src/UI/oglWidget.h

OGLWidget::GoalCallback Struct Reference

Inheritance diagram for OGLWidget::GoalCallback:

IMAGE

Public Member Functions

GoalCallback (OGLWidget *scopePtr)

btScalar addSingleResult (btManifoldPoint &cp, const btCollisionObjectWrapper *colObj0Wrap, int partId0, int index0, const btCollisionObjectWrapper *colObj1Wrap, int partId1, int index1)

Public Attributes

OGLWidget * context

Constructor & Destructor Documentation

OGLWidget::GoalCallback::GoalCallback (OGLWidget * *scopePtr*)`[inline]`

Member Function Documentation

btScalar OGLWidget::GoalCallback::addSingleResult (btManifoldPoint & *cp*, const btCollisionObjectWrapper * *colObj0Wrap*, int *partId0*, int *index0*, const btCollisionObjectWrapper * *colObj1Wrap*, int *partId1*, int *index1*)`[inline]`

Member Data Documentation

OGLWidget* OGLWidget::GoalCallback::context

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

10 src/UI/oglWidget.h

HockeyPaddle Class Reference

`#include <hockeyPaddle.h>`
Inheritance diagram for HockeyPaddle:

IMAGE

Public Member Functions

HockeyPaddle (const QString &color)
Default constructor for HockeyPaddle.

Additional Inherited Members

Constructor & Destructor Documentation

HockeyPaddle::HockeyPaddle (const QString & *color*)

Default constructor for HockeyPaddle.

Parameters:

in	*color*	The color of the paddle, Red or Blue.

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

11 src/GameObjects/hockeyPaddle.h
12 src/GameObjects/hockeyPaddle.cpp

HockeyPuck Class Reference

`#include <hockeyPuck.h>`
Inheritance diagram for HockeyPuck:

IMAGE

Public Member Functions

HockeyPuck ()
Default constructor for HockeyPuck.

Additional Inherited Members

Constructor & Destructor Documentation

HockeyPuck::HockeyPuck ()

Default constructor for HockeyPuck.

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

13 src/GameObjects/hockeyPuck.h
14 src/GameObjects/hockeyPuck.cpp

HockeyTable Class Reference

`#include <hockeyTable.h>`
Inheritance diagram for HockeyTable:

IMAGE

Public Member Functions

HockeyTable ()
Default constructor for Hockey Table.

void update ()
Overloaded update function.

Private Attributes

QMediaPlayer * player

Additional Inherited Members

Constructor & Destructor Documentation

HockeyTable::HockeyTable ()

Default constructor for Hockey Table.

Member Function Documentation

void HockeyTable::update ()[virtual]

Overloaded update function.

Plays a huming sound to simulate air.

Reimplemented from ColorPhysicsEntity.

Member Data Documentation

QMediaPlayer* HockeyTable::player[private]

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

15 src/GameObjects/hockeyTable.h
16 src/GameObjects/hockeyTable.cpp

anonymous_namespace{qrc_shader_resource.cpp}::initializer Struct Reference

Public Member Functions
initializer ()
~initializer ()

Constructor & Destructor Documentation

anonymous_namespace{qrc_shader_resource.cpp}::initializer::initializer ()[inline]

anonymous_namespace{qrc_shader_resource.cpp}::initializer::~initializer ()[inline]

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

17 build/qrc_shader_resource.cpp

anonymous_namespace{qrc_font_resource.cpp}::initializer Struct Reference

Public Member Functions
initializer ()
~initializer ()

Constructor & Destructor Documentation

anonymous_namespace{qrc_font_resource.cpp}::initializer::initializer ()[inline]

anonymous_namespace{qrc_font_resource.cpp}::initializer::~initializer ()[inline]

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

18 build/qrc_font_resource.cpp

Input Class Reference

`#include <input.h>`

Public Types

enum InputState { InputInvalid, InputRegistered, InputUnregistered, InputTriggered, InputPressed, InputReleased }

Static Public Member Functions

static InputState keyState (Qt::Key key)
static bool keyTriggered (Qt::Key key)
static bool keyPressed (Qt::Key key)
static bool keyReleased (Qt::Key key)
static InputState buttonState (Qt::MouseButton button)
static bool buttonTriggered (Qt::MouseButton button)
static bool buttonPressed (Qt::MouseButton button)
static bool buttonReleased (Qt::MouseButton button)
static QPoint mousePosition ()
static QPoint mouseDelta ()

Static Private Member Functions

static void update ()
static void registerKeyPress (int key)
static void registerKeyRelease (int key)
static void registerMousePress (Qt::MouseButton button)
static void registerMouseRelease (Qt::MouseButton button)
static void reset ()

Friends

class OGLWidget

Member Enumeration Documentation

enum Input::InputState

Enumerator

InputInvalid
InputRegistered
InputUnregistered
InputTriggered
InputPressed
InputReleased

Member Function Documentation

bool Input::buttonPressed (Qt::MouseButton *button*)[static]

bool Input::buttonReleased (Qt::MouseButton *button*)[static]

Input::InputState Input::buttonState (Qt::MouseButton *button*)[static]

bool Input::buttonTriggered (Qt::MouseButton *button*)[static]

bool Input::keyPressed (Qt::Key *key*)[static]

bool Input::keyReleased (Qt::Key *key*)[static]

Input::InputState Input::keyState (Qt::Key *key*)[static]

bool Input::keyTriggered (Qt::Key *key*)[static]

QPoint Input::mouseDelta ()[static]

QPoint Input::mousePosition ()[static]

void Input::registerKeyPress (int *key*)[static], [private]

void Input::registerKeyRelease (int *key*)[static], [private]

void Input::registerMousePress (Qt::MouseButton *button*)[static], [private]

void Input::registerMouseRelease (Qt::MouseButton *button*)[static], [private]

void Input::reset ()[static], [private]

void Input::update ()[static], [private]

Friends And Related Function Documentation

friend class OGLWidget[friend]

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

19 src/Controls/input.h
20 src/Controls/input.cpp

InputInstance< T > Struct Template Reference

Inheritance diagram for InputInstance< T >:

Public Types

typedef std::pair< T,
Input::InputState > base_class

Public Member Functions

InputInstance (T value)
InputInstance (T value, Input::InputState state)
bool operator== (const InputInstance &rhs) const

Member Typedef Documentation

template<typename T > typedef std::pair<T, Input::InputState> InputInstance< T >::base_class

---

Constructor & Destructor Documentation

template<typename T > InputInstance< T >::InputInstance (T *value*)[inline]

template<typename T > InputInstance< T >::InputInstance (T *value*, Input::InputState *state*) [inline]

Member Function Documentation

template<typename T > bool InputInstance< T >::operator== (const InputInstance< T > & *rhs*) const[inline]

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

21 src/Controls/input.cpp

MainMenuWidget Class Reference

`#include <mainMenuWidget.h>`
Inheritance diagram for MainMenuWidget:
IMAGE

Signals

void clickedSinglePlayer ()
void clickedTwoPlayer ()
void clickedExit ()

Public Member Functions

MainMenuWidget ()
Default constructor for MainMenuWidget.

~MainMenuWidget ()
Default destructor for MainMenuWidget.

Protected Member Functions

virtual void resizeEvent (QResizeEvent *event)
Overloaded function to resize elements within MainMenuWidget.

Private Member Functions

void resize ()
Helper function to resize the graphics and buttons within the main menu.

Private Attributes

QLabel * m_lbBackground
QLabel * m_lbLeftPlayer
QLabel * m_lbRightPlayer
QLabel * m_lbNhlLogo
QPushButton * m_btnSinglePlayer
QPushButton * m_btnTwoPlayer
QPushButton * m_btnExit
QPixmap * m_imgBackground
QPixmap * m_imgLeftPlayer
QPixmap * m_imgRightPlayer
QPixmap * m_imgNhlLogo

Constructor & Destructor Documentation

MainMenuWidget::MainMenuWidget ()

Default constructor for MainMenuWidget.

MainMenuWidget::~MainMenuWidget ()

Default destructor for MainMenuWidget.

Member Function Documentation

void MainMenuWidget::clickedExit ()[signal]

void MainMenuWidget::clickedSinglePlayer ()[signal]

void MainMenuWidget::clickedTwoPlayer ()[signal]

void MainMenuWidget::resize ()[private]

Helper function to resize the graphics and buttons within the main menu.

void MainMenuWidget::resizeEvent (QResizeEvent * *event*)[protected], [virtual]

Overloaded function to resize elements within MainMenuWidget.

Parameters:

event	The QResizeEvent.

Member Data Documentation

QPushButton* MainMenuWidget::m_btnExit[private]

QPushButton* MainMenuWidget::m_btnSinglePlayer[private]

QPushButton* MainMenuWidget::m_btnTwoPlayer[private]

QPixmap* MainMenuWidget::m_imgBackground[private]

QPixmap* MainMenuWidget::m_imgLeftPlayer[private]

QPixmap* MainMenuWidget::m_imgNhlLogo[private]

QPixmap* MainMenuWidget::m_imgRightPlayer[private]

QLabel* MainMenuWidget::m_lbBackground[private]

QLabel* MainMenuWidget::m_lbLeftPlayer[private]

QLabel* MainMenuWidget::m_lbNhlLogo[private]

QLabel* MainMenuWidget::m_lbRightPlayer[private]

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

22 src/UI/mainMenuWidget.h
23 build/moc_mainMenuWidget.cpp
24 src/UI/mainMenuWidget.cpp

MainWindow Class Reference

`#include <mainWindow.h>`
Inheritance diagram for MainWindow:

IMAGE

Public Slots

void swapToTeamSelect ()
Slot to swap to the Team Select window.

void swapToGame (QString team1, QString team2)
Slot to swap to the Game window.

void restartGame ()
Slot to restart the entire game.

Public Member Functions

MainWindow ()
Main window for OpenGL based projects.

Private Member Functions

void createActions ()
Creates the main window's actions.

void createMenus ()
Creates the main window's menus.

void createMenuBar ()
Creates the main window's menu bar.

Private Attributes

OGLWidget * oglWidget
MainMenuWidget * mainMenuWidget
TeamSelectWidget * teamSelectWidget
QMenuBar * menuBar
QMenu * menuFile
QAction * actionPauseProgram
QAction * actionRestartGame
QAction * actionExitProgram
QMenu * menuCamera
QSignalMapper * m_signalMapper
QAction * actionSideAngled
QAction * actionBehindPlayer1
QAction * actionBehindPlayer2
QAction * actionTopDown

Constructor & Destructor Documentation

MainWindow::MainWindow ()

Main window for OpenGL based projects.

Creates an OGLWidget with an OpenGL surface to draw on as the central widget. A menu bar will also be created for user interaction with the central widget.

Member Function Documentation

void MainWindow::createActions ()[private]

Creates the main window's actions.

void MainWindow::createMenuBar ()[private]

Creates the main window's menu bar.

void MainWindow::createMenus ()[private]

Creates the main window's menus.

void MainWindow::restartGame ()[slot]

Slot to restart the entire game.

void MainWindow::swapToGame (QString *team1*, QString *team2*)[slot]

Slot to swap to the Game window.

Parameters:

in	*team1*	Player 1's selected team.
in	*team2*	Player 2's selected team.

void MainWindow::swapToTeamSelect ()[slot]

Slot to swap to the Team Select window.

Member Data Documentation

QAction* MainWindow::actionBehindPlayer1 [private]

QAction* MainWindow::actionBehindPlayer2 [private]

QAction* MainWindow::actionExitProgram [private]

QAction* MainWindow::actionPauseProgram [private]

QAction* MainWindow::actionRestartGame [private]

QAction* MainWindow::actionSideAngled [private]

QAction* MainWindow::actionTopDown [private]

QSignalMapper* MainWindow::m_signalMapper [private]

MainMenuWidget* MainWindow::mainMenuWidget [private]

QMenuBar* MainWindow::menuBar [private]

QMenu* MainWindow::menuCamera [private]

QMenu* MainWindow::menuFile [private]

OGLWidget* MainWindow::oglWidget [private]

TeamSelectWidget* MainWindow::teamSelectWidget [private]

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

25 src/UI/mainWindow.h
26 src/UI/mainWindow.cpp

OGLWidget Class Reference

`#include <oglWidget.h>`
Inheritance diagram for OGLWidget:

IMAGE

Classes

struct Goal2Callback
struct GoalCallback
struct PuckCallback

Public Slots

void setPerspective (int perspective)
Sets the camera's perspective to predefined values. details 0: Behind player 1. 1: Behind player 2. 2: Side perspective. 3: Top-Down perspective.

Public Member Functions

OGLWidget ()
Default constructor for OGLWidget.

OGLWidget (QString team1, QString team2)
Overloaded constructor, sets custom team logos.

~OGLWidget ()
Destructor class to unallocate OpenGL information.

virtual void initializeGL ()
Initializes any OpenGL operations.

virtual void resizeGL (int width, int height)
Sets the prespective whenever the window is resized.

virtual void paintGL ()
OpenGL function to draw elements to the surface.

virtual void teardownGL ()
Destroys any OpenGL data.

Protected Slots

void pause ()
Public slot to invert the pause state of the game.

void update ()
Updates any user interactions and model transformations.

Protected Member Functions

void keyPressEvent (QKeyEvent *event)
Default slot for handling key press events.

void keyReleaseEvent (QKeyEvent *event)

Default slot for handling key release events.

void mousePressEvent (QMouseEvent *event)

Default slot for handling mouse press events.

void mouseReleaseEvent (QMouseEvent *event)

Default slot for handling mouse release events.

Private Member Functions

void initializeBullet ()

Helper function to initialize bullet data.

void teardownBullet ()

Helper function to delete bullet allocations.

void flyThroughCamera ()

Updates the main camera to behave like a Fly-Through Camera.

void controlObject ()

void printContextInfo ()

Helper function to print OpenGL Context information to the debug.

void processGoal ()

Resets the game state after a goal.

void puckContactSound ()

Plays a sound whenever a puck collides.

Private Attributes

QMap< QString, Renderable * > renderables
QMap< QString, Wall * > walls
QMatrix4x4 projection
Camera3D camera
btBroadphaseInterface * m_broadphase
btDefaultCollisionConfiguration * m_collisionConfig
btCollisionDispatcher * m_dispatcher
btSequentialImpulseConstraintSolver * m_solver
btDiscreteDynamicsWorld * m_dynamicsWorld
QMediaPlayer * player
const short COL_NOTHING = 0
const short COL_TABLE = 1 << 0
const short COL_PUCK = 1 << 1
const short COL_PADDLE = 1 << 2
const short COL_MIDDLE = 1 << 3
const short COL_GOAL = 1 << 4
const short m_TableCollidesWith = (COL_PUCK | COL_PADDLE)
const short m_PuckCollidesWith = (COL_TABLE | COL_PADDLE | COL_GOAL)
const short m_PaddleCollidesWith = (COL_TABLE | COL_PUCK | COL_MIDDLE)
const short m_MiddleCollidesWith = COL_PADDLE
const short m_GoalCollidesWith = (COL_PUCK)
bool isPaused
QString m_pathToTeam1 = "images/Team Logos/Avalanches.png"
QString m_pathToTeam2 = "images/Team Logos/Blackhawks.png"
QPixmap * m_imgTeam1
QPixmap * m_imgTeam2
QLabel * m_p1Team

QLabel * m_p2Team
int m_p1Score
int m_p2Score

Constructor & Destructor Documentation

OGLWidget::OGLWidget ()

Default constructor for OGLWidget.

OGLWidget::OGLWidget (QString *team1*, QString *team2*)

Overloaded constructor, sets custom team logos.

Parameters:

in	*team1*	Name of logo for team 1.
in	*team2*	Name of logo for team 2.

OGLWidget::~OGLWidget ()

Destructor class to unallocate OpenGL information.

Member Function Documentation

void OGLWidget::controlObject ()[private]

void OGLWidget::flyThroughCamera ()[private]

Updates the main camera to behave like a Fly-Through Camera.

void OGLWidget::initializeBullet ()[private]

Helper function to initialize bullet data.

void OGLWidget::initializeGL ()[virtual]

Initializes any OpenGL operations.

void OGLWidget::keyPressEvent (QKeyEvent * *event*)[protected]

Default slot for handling key press events.

Parameters:

event	The key event information.

void OGLWidget::keyReleaseEvent (QKeyEvent * *event*)`[protected]`

Default slot for handling key release events.

Parameters:

event	The key event information.

void OGLWidget::mousePressEvent (QMouseEvent * *event*)`[protected]`

Default slot for handling mouse press events.

Parameters:

event	The mouse event information.

void OGLWidget::mouseReleaseEvent (QMouseEvent * *event*)`[protected]`

Default slot for handling mouse release events.

Parameters:

event	The mouse event information.

void OGLWidget::paintGL ()`[virtual]`

OpenGL function to draw elements to the surface.

void OGLWidget::pause ()`[protected]`, `[slot]`

Public slot to invert the pause state of the game.

void OGLWidget::printContextInfo ()`[private]`

Helper function to print OpenGL Context information to the debug.

void OGLWidget::processGoal ()[private]

Resets the game state after a goal.

void OGLWidget::puckContactSound ()[private]

Plays a sound whenever a puck collides.

void OGLWidget::resizeGL (int *width*, int *height*)[virtual]

Sets the prespective whenever the window is resized.

Parameters:

in	*width*	The width of the new window.
in	*height*	The height of the new window.

void OGLWidget::setPerspective (int *perspective*)[slot]

Sets the camera's perspective to predefined values. details 0: Behind player 1. 1: Behind player 2. 2: Side perspective. 3: Top-Down perspective.

Parameters:

in	*perspective*	The index of the perspective desired.

void OGLWidget::teardownBullet ()[private]

Helper function to delete bullet allocations.

void OGLWidget::teardownGL ()[virtual]

Destroys any OpenGL data.

void OGLWidget::update ()[protected], [slot]

Updates any user interactions and model transformations.

Member Data Documentation

Camera3D OGLWidget::camera[private]

const short OGLWidget::COL_GOAL = 1 << 4[private]

const short OGLWidget::COL_MIDDLE = 1 << 3[private]

const short OGLWidget::COL_NOTHING = 0[private]

const short OGLWidget::COL_PADDLE = 1 << 2[private]

const short OGLWidget::COL_PUCK = 1 << 1[private]

const short OGLWidget::COL_TABLE = 1 << 0[private]

bool OGLWidget::isPaused[private]

btBroadphaseInterface* OGLWidget::m_broadphase[private]

btDefaultCollisionConfiguration* OGLWidget::m_collisionConfig[private]

btCollisionDispatcher* OGLWidget::m_dispatcher[private]

btDiscreteDynamicsWorld* OGLWidget::m_dynamicsWorld[private]

const short OGLWidget::m_GoalCollidesWith = (COL_PUCK)[private]

QPixmap* OGLWidget::m_imgTeam1[private]

QPixmap * OGLWidget::m_imgTeam2[private]

const short OGLWidget::m_MiddleCollidesWith = COL_PADDLE[private]

int OGLWidget::m_p1Score[private]

QLabel* OGLWidget::m_p1Team[private]

int OGLWidget::m_p2Score[private]

QLabel * OGLWidget::m_p2Team[private]

const short OGLWidget::m_PaddleCollidesWith = (COL_TABLE | COL_PUCK | COL_MIDDLE)[private]

QString OGLWidget::m_pathToTeam1 = "images/Team Logos/Avalanches.png"[private]

QString OGLWidget::m_pathToTeam2 = "images/Team Logos/Blackhawks.png"[private]

const short OGLWidget::m_PuckCollidesWith = (COL_TABLE | COL_PADDLE | COL_GOAL)[private]

btSequentialImpulseConstraintSolver* OGLWidget::m_solver[private]

const short OGLWidget::m_TableCollidesWith = (COL_PUCK | COL_PADDLE)[private]

QMediaPlayer* OGLWidget::player[private]

QMatrix4x4 OGLWidget::projection[private]

QMap<QString, Renderable*> OGLWidget::renderables[private]

QMap<QString, Wall*> OGLWidget::walls[private]

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

27 src/UI/oglWidget.h
28 src/UI/oglWidget.cpp

OGLWidget::PuckCallback Struct Reference

Inheritance diagram for OGLWidget::PuckCallback:

IMAGE

Public Member Functions

PuckCallback (OGLWidget *scopePtr)

btScalar addSingleResult (btManifoldPoint &cp, const btCollisionObjectWrapper *colObj0Wrap, int partId0, int index0, const btCollisionObjectWrapper *colObj1Wrap, int partId1, int index1)

Public Attributes

OGLWidget * context

Constructor & Destructor Documentation

OGLWidget::PuckCallback::PuckCallback (OGLWidget * *scopePtr*)`[inline]`

Member Function Documentation

btScalar OGLWidget::PuckCallback::addSingleResult (btManifoldPoint & *cp*, const btCollisionObjectWrapper * *colObj0Wrap*, int *partId0*, int *index0*, const btCollisionObjectWrapper * *colObj1Wrap*, int *partId1*, int *index1*)`[inline]`

Member Data Documentation

OGLWidget* OGLWidget::PuckCallback::context

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

29 src/UI/oglWidget.h

qt_meta_stringdata_MainMenuWidget_t Struct Reference

Public Attributes
QByteArrayData data [5]
char stringdata0 [65]

Member Data Documentation

QByteArrayData qt_meta_stringdata_MainMenuWidget_t::data[5]

char qt_meta_stringdata_MainMenuWidget_t::stringdata0[65]

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

30 build/moc_mainMenuWidget.cpp

qt_meta_stringdata_MainWindow_t Struct Reference

Public Attributes
QByteArrayData data [7]
char stringdata0 [64]

Member Data Documentation

QByteArrayData qt_meta_stringdata_MainWindow_t::data[7]

char qt_meta_stringdata_MainWindow_t::stringdata0[64]

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

31 build/moc_mainWindow.cpp

qt_meta_stringdata_OGLWidget_t Struct Reference

Public Attributes
QByteArrayData data [6]
char stringdata0 [51]

Member Data Documentation

QByteArrayData qt_meta_stringdata_OGLWidget_t::data[6]

char qt_meta_stringdata_OGLWidget_t::stringdata0[51]

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

32 build/moc_oglWidget.cpp

qt_meta_stringdata_TeamSelectWidget_t Struct Reference

Public Attributes
QByteArrayData data [8]
char stringdata0 [70]

Member Data Documentation

QByteArrayData qt_meta_stringdata_TeamSelectWidget_t::data[8]

char qt_meta_stringdata_TeamSelectWidget_t::stringdata0[70]

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

33 build/moc_teamSelectWidget.cpp

Renderable Class Reference

`#include <renderable.h>`
Inheritance diagram for Renderable:

IMAGE

Public Member Functions

virtual void initializeGL ()=0
virtual void paintGL (Camera3D &camera, QMatrix4x4 &projection)=0
virtual void update ()=0
virtual void teardownGL ()=0

Member Function Documentation

virtual void Renderable::initializeGL ()[pure virtual]

Implemented in ColorEntity, and UVEntity.

virtual void Renderable::paintGL (Camera3D & *camera*, QMatrix4x4 & *projection*)[pure virtual]

Implemented in ColorEntity, UVEntity, ColorPhysicsEntity, UVPhysicsEntity, and Skybox.

virtual void Renderable::teardownGL ()[pure virtual]

Implemented in ColorEntity, and UVEntity.

virtual void Renderable::update ()[pure virtual]

Implemented in ColorEntity, UVEntity, ColorPhysicsEntity, UVPhysicsEntity, and HockeyTable.

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

34 src/3D/renderable.h

Skybox Class Reference

`#include <skybox.h>`
Inheritance diagram for Skybox:

IMAGE

Public Member Functions

Skybox ()
Default constructor for Skybox.

void paintGL (Camera3D &camera, QMatrix4x4 &projection)
Overloaded paintGL function.

Additional Inherited Members

Constructor & Destructor Documentation

Skybox::Skybox ()

Default constructor for Skybox.

Member Function Documentation

void Skybox::paintGL (Camera3D & *camera*, QMatrix4x4 & *projection*) [virtual]

Overloaded paintGL function.

Parameters:

camera	The camera of the world.
projection	The current projection of the world.

Reimplemented from UVEntity.

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

35 src/GameObjects/skybox.h
36 src/GameObjects/skybox.cpp

TeamSelectWidget Class Reference

`#include <teamSelectWidget.h>`
Inheritance diagram for TeamSelectWidget:
IMAGE

Public Slots

void setTeams (QString team)
Sets the current player's team.

Signals

void selectedTeams (QString team1, QString team2)
void clickedTeam (QString team)

Public Member Functions

TeamSelectWidget ()
Default constructor for TeamSelectWidget.

~TeamSelectWidget ()
Destructor for TeamSelectWidget.

Protected Member Functions

virtual void resizeEvent (QResizeEvent *event)
Overloaded resizeEvent function.

Private Attributes

QPushButton * m_btnLogos [30]
QIcon * m_imgLogos [30]
QGridLayout * m_layout
QSignalMapper * m_signalMapper
QString m_team1
QString m_team2

Constructor & Destructor Documentation

TeamSelectWidget::TeamSelectWidget ()

Default constructor for TeamSelectWidget.

Creates a grid of 30 logos representing different NHL teams.

TeamSelectWidget::~TeamSelectWidget ()

Destructor for TeamSelectWidget.

Member Function Documentation

void TeamSelectWidget::clickedTeam (QString *team*)[signal]

void TeamSelectWidget::resizeEvent (QResizeEvent * *event*)[protected], [virtual]

Overloaded resizeEvent function.

Scales the logos based on the screen size.

Parameters:

event	The QResizeEvent.

void TeamSelectWidget::selectedTeams (QString *team1*, QString *team2*)[signal]

void TeamSelectWidget::setTeams (QString *team*)[slot]

Sets the current player's team.

Parameters:

in	*team*	The team to give to the current player.

Member Data Documentation

QPushButton* TeamSelectWidget::m_btnLogos[30][private]

QIcon* TeamSelectWidget::m_imgLogos[30][private]

QGridLayout* TeamSelectWidget::m_layout[private]

QSignalMapper* TeamSelectWidget::m_signalMapper[private]

QString TeamSelectWidget::m_team1[private]

QString TeamSelectWidget::m_team2[private]

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

37 src/UI/teamSelectWidget.h
38 build/moc_teamSelectWidget.cpp
39 src/UI/teamSelectWidget.cpp

Transform3D Class Reference

`#include <transform3d.h>`

Public Member Functions

Transform3D ()
Default constructor, sets the scale to 1.

void translate (const QVector3D &dt)
Moves the matrix based on its current location.

void translate (float dx, float dy, float dz)
Overloaded translate function.

void scale (const QVector3D &ds)
Scales the matrix based on its current size.

void scale (float dx, float dy, float dz)
Overloaded scale function.

void scale (float factor)
Overloaded scale function.

void rotate (const QQuaternion &dr)
Rotates the matrix based on its current rotation.

void rotate (float angle, const QVector3D &axis)
Overloaded rotate function.

void rotate (float angle, float ax, float ay, float az)
Overloaded rotate function.

void setTranslation (const QVector3D &t)
Sets the matrix to a new location.

void setTranslation (float x, float y, float z)
Overloaded setTranslation function.

void setScale (const QVector3D &s)
Sets the matrix to a new scale.

void setScale (float x, float y, float z)
Overloaded setScale function.

void setScale (float k)
Overloaded setScale function.

void setRotation (const QQuaternion &r)
Sets the matrix to a new rotation.

void setRotation (float angle, const QVector3D &axis)
Overloaded setRotation function.

void setRotation (float angle, float ax, float ay, float az)
Overloaded setRotation function.

const QVector3D & translation () const
Getter function for m_translation.

const QVector3D & scale () const
Getter function for m_scale.

const QQuaternion & rotation () const
Getter function for m_rotation.

const QMatrix4x4 & toMatrix ()
Getter function for m_world.

QVector3D forward () const
The relative forward position of this matrix.

QVector3D up () const
The relative up position of this matrix.

QVector3D right () const
The relative right position of this matrix.

Static Public Attributes

static const QVector3D LocalForward
static const QVector3D LocalUp
static const QVector3D LocalRight

Private Attributes

QVector3D m_translation
QVector3D m_scale
QQuaternion m_rotation
QMatrix4x4 m_world

Constructor & Destructor Documentation

Transform3D::Transform3D ()

Default constructor, sets the scale to 1.

Member Function Documentation

QVector3D Transform3D::forward () const

The relative forward position of this matrix.

Returns:
A foward vector based on this matrix.

QVector3D Transform3D::right () const

The relative right position of this matrix.

Returns:

A right vector based on this matrix.

void Transform3D::rotate (const QQuaternion & *dr*)

Rotates the matrix based on its current rotation.

Parameters:

in	*dr*	The rotation to add onto the current matrix.

void Transform3D::rotate (float *angle*, const QVector3D & *axis*)

Overloaded rotate function.

See Also:

Transform3D::rotate(const QQuaternion& dr)

Parameters:

in	*angle*	The angle to add onto the current angle.
in	*axis*	The axis of rotation for the angle.

void Transform3D::rotate (float *angle*, float *ax*, float *ay*, float *az*)

Overloaded rotate function.

See Also:

Transform3D::rotate(const QQuaternion& dr)

Parameters:

in	*angle*	The angle to add onto the current angle.
in	*ax*	The x-axis of rotation.
in	*ay*	The y-axis of rotation.
in	*az*	The z-axis of rotation.

const QQuaternion & Transform3D::rotation () const

Getter function for m_rotation.

Returns:

The current rotation of the matrix.

void Transform3D::scale (const QVector3D & *ds*)

Scales the matrix based on its current size.

Parameters:

in	*ds*	The scale to add onto the current matrix.

void Transform3D::scale (float *dx*, float *dy*, float *dz*)

Overloaded scale function.

See Also:

Transform3D::scale(const QVector3D& ds)

Parameters:

in	*dx*	X-axis scale.
in	*dy*	Y-axis scale.
in	*dz*	Z-axis scale.

void Transform3D::scale (float *factor*)

Overloaded scale function.

See Also:

Transform3D::scale(const QVector3D& ds)

Parameters:

in	*factor*	X-Y-Z-axis scale.

const QVector3D & Transform3D::scale () const

Getter function for m_scale.

Returns:

The current scale of the matrix.

void Transform3D::setRotation (const QQuaternion & *r*)

Sets the matrix to a new rotation.

Parameters:

in	*r*	The new rotation.

void Transform3D::setRotation (float *angle*, const QVector3D & *axis*)

Overloaded setRotation function.

See Also:

Transform3D::setRotation(const QQuaternion& r)

Parameters:

in	*angle*	The new angle of rotation.
in	*axis*	The axis of rotation for the new angle.

void Transform3D::setRotation (float *angle*, float *x*, float *y*, float *z*)

Overloaded setRotation function.

See Also:

Transform3D::setRotation(const QQuaternion& r)

Parameters:

in	*angle*	The new angle of rotation.
in	*x*	The x-axis of rotation for the new angle.
in	*y*	The y-axis of rotation for the new angle.
in	*z*	The z-axis of rotation for the new angle.

void Transform3D::setScale (const QVector3D & *s*)

Sets the matrix to a new scale.

Parameters:

in	*s*	The new scale.

void Transform3D::setScale (float *x*, float *y*, float *z*)

Overloaded setScale function.

See Also:

Transform3D::setScale(const QVector3D& s)

Parameters:

in	*x*	The new x-axis scale.
in	*y*	The new y-axis scale.
in	*z*	The new z-axis scale.

void Transform3D::setScale (float *k*)

Overloaded setScale function.

See Also:

Transform3D::setScale(const QVector3D& s)

Parameters:

in	*k*	The new scale based on a factor of k.

void Transform3D::setTranslation (const QVector3D & *t*)

Sets the matrix to a new location.

Parameters:

in	*t*	The new location.

void Transform3D::setTranslation (float *x*, float *y*, float *z*)

Overloaded setTranslation function.

See Also:

Transform3D::setTranslation(const QVector3D& t)

Parameters:

in	*x*	The new x-axis location.
in	*y*	The new y-axis location.
in	*z*	The new z-axis location.

const QMatrix4x4 & Transform3D::toMatrix ()

Getter function for m_world.

Returns:

The full matrix with current translations.

void Transform3D::translate (const QVector3D & *dt*)

Moves the matrix based on its current location.

Parameters:

in	*dt*	The location to add onto the current matrix.

void Transform3D::translate (float *dx*, float *dy*, float *dz*)

Overloaded translate function.

See Also:

Transform3D::translate(const QVector3D& dt)

Parameters:

in	*dx*	X-axis offset.
in	*dy*	Y-axis offset.
in	*dz*	Z-axis offset.

const QVector3D & Transform3D::translation () const

Getter function for m_translation.

Returns:

The current translation of the matrix.

QVector3D Transform3D::up () const

The relative up position of this matrix.

Returns:

An up vector based on this matrix.

Member Data Documentation

const QVector3D Transform3D::LocalForward[static]

const QVector3D Transform3D::LocalRight[static]

const QVector3D Transform3D::LocalUp[static]

QQuaternion Transform3D::m_rotation[private]

QVector3D Transform3D::m_scale[private]

QVector3D Transform3D::m_translation[private]

QMatrix4x4 Transform3D::m_world[private]

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

40 src/3D/transform3d.h
41 src/3D/transform3d.cpp

UVEntity Class Reference

`#include <uvEntity.h>`
Inheritance diagram for UVEntity:

IMAGE

Public Attributes

Transform3D GTransform

Protected Member Functions

UVEntity (QString pathToModel, QString pathToTexture)
Constructor for UVEntity.

~UVEntity ()
Destructor for UVEntity.

void initializeGL ()
Initializes the OpenGL data for drawing.

void paintGL (Camera3D &camera, QMatrix4x4 &projection)
Draws the object to the screen.

virtual void update ()
Virtual update function.

void teardownGL ()
Helper function to delete OpenGL data.

Protected Attributes

QOpenGLBuffer * m_vbo
QOpenGLVertexArrayObject * m_vao
QOpenGLShaderProgram * m_program
QString m_pathToModel
UVVertex * m_model
int m_numVertices
QString m_pathToTexture
QOpenGLTexture * m_texture
const QString PATH_TO_V_SHADER = ":/shader/uvShader.vs"
const QString PATH_TO_F_SHADER = ":/shader/uvShader.fs"
int m_modelWorld
int m_worldEye
int m_eyeClip

Additional Inherited Members

Constructor & Destructor Documentation

UVEntity::UVEntity (QString *pathToModel*, QString *pathToTexture*)`[protected]`

Constructor for UVEntity.

Parameters:

in	*pathToModel*	The path to this object's model.
in	*pathToTexture*	The path to this object's texture.

UVEntity::~UVEntity ()[protected]

Destructor for UVEntity.

Member Function Documentation

void UVEntity::initializeGL ()[protected], [virtual]

Initializes the OpenGL data for drawing.

Implements Renderable.

void UVEntity::paintGL (Camera3D & *camera*, QMatrix4x4 & *projection*)[protected], [virtual]

Draws the object to the screen.

Parameters:

camera	The camera of the world.
projection	The projection of the world.

Implements Renderable.

Reimplemented in UVPhysicsEntity, and Skybox.

void UVEntity::teardownGL ()[protected], [virtual]

Helper function to delete OpenGL data.

Implements Renderable.

void UVEntity::update ()[protected], [virtual]

Virtual update function.

Implements Renderable.

Reimplemented in UVPhysicsEntity.

Member Data Documentation

Transform3D UVEntity::GTransform

int UVEntity::m_eyeClip[protected]

UVVertex* UVEntity::m_model[protected]

int UVEntity::m_modelWorld[protected]

int UVEntity::m_numVertices[protected]

QString UVEntity::m_pathToModel[protected]

QString UVEntity::m_pathToTexture[protected]

QOpenGLShaderProgram* UVEntity::m_program[protected]

QOpenGLTexture* UVEntity::m_texture[protected]

QOpenGLVertexArrayObject* UVEntity::m_vao[protected]

QOpenGLBuffer* UVEntity::m_vbo[protected]

int UVEntity::m_worldEye[protected]

const QString UVEntity::PATH_TO_F_SHADER = ":/shader/uvShader.fs"[protected]

const QString UVEntity::PATH_TO_V_SHADER = ":/shader/uvShader.vs"[protected]

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

42 src/Entity/uvEntity.h
43 src/Entity/uvEntity.cpp

UVPhysicsEntity Class Reference

`#include <uvPhysicsEntity.h>`
Inheritance diagram for UVPhysicsEntity:
IMAGE

Public Attributes

btRigidBody * RigidBody
QMatrix4x4 BTransform

Protected Member Functions

UVPhysicsEntity (btTransform startingState, btScalar mass, QString pathToModel, QString pathToTexture)
Constructor for a UVPhysicsEntity.

~UVPhysicsEntity ()
Default destructor for UVPhysicsEntity.

void paintGL (Camera3D &camera, QMatrix4x4 &projection)
Overloaded paintGL function.

void update ()
Updates the RigidBody based on the dynamics world it is in.

Private Attributes

btTriangleMesh * m_triMesh
btCollisionShape * m_collisionShape
btDefaultMotionState * m_motionState
btVector3 m_inertia
btScalar m_mass
btRigidBody::btRigidBodyConstructionInfo * m_rigidBodyCI

Additional Inherited Members

Constructor & Destructor Documentation

UVPhysicsEntity::UVPhysicsEntity (btTransform *startingState*, btScalar *mass*, QString *pathToModel*, QString *pathToTexture*)`[protected]`

Constructor for a UVPhysicsEntity.

Parameters:

in	*startingState*	The starting location and rotation of the object.
in	*mass*	The mass of the object.
in	*pathToModel*	The path to the object's model.
in	*pathToTexture*	The path to the object's texture.

UVPhysicsEntity::~UVPhysicsEntity ()[protected]

Default destructor for UVPhysicsEntity.

Member Function Documentation

void UVPhysicsEntity::paintGL (Camera3D & *camera*, QMatrix4x4 & *projection*) [protected], [virtual]

Overloaded paintGL function.

Draws using a BTransform rather than a GTrasform.

Parameters:

camera	The camera of the world.
projection	The projection of the world.

Reimplemented from UVEntity.

void UVPhysicsEntity::update ()[protected], [virtual]

Updates the RigidBody based on the dynamics world it is in.

Reimplemented from UVEntity.

Member Data Documentation

QMatrix4x4 UVPhysicsEntity::BTransform

btCollisionShape* UVPhysicsEntity::m_collisionShape[private]

btVector3 UVPhysicsEntity::m_inertia[private]

btScalar UVPhysicsEntity::m_mass[private]

btDefaultMotionState* UVPhysicsEntity::m_motionState[private]

btRigidBody::btRigidBodyConstructionInfo* UVPhysicsEntity::m_rigidBodyCI[private]

btTriangleMesh* UVPhysicsEntity::m_triMesh[private]

btRigidBody* UVPhysicsEntity::RigidBody

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

44 src/Entity/uvPhysicsEntity.h
45 src/Entity/uvPhysicsEntity.cpp

UVVertex Class Reference

`#include <uvVertex.h>`

Public Member Functions

UVVertex ()
Default constructor for UVVertex.

UVVertex (const QVector3D &position)
Overloaded constructor for UVVertex.

UVVertex (const QVector3D &position, const QVector2D &uv)
Overloaded constructor for UVVertex.

const QVector3D & position () const
Gets the position.

const QVector2D & uv () const
Gets the uv.

void setPosition (const QVector3D &position)
Sets position to a new position.

void setUV (const QVector2D &uv)
Sets uv to a new uv.

Static Public Member Functions

static int positionOffset ()
Calculates the position offset within UVVertex.

static int uvOffset ()
Calcualtes the uv offset within UVVertex.

static int stride ()
Calculates the stride of UVVertex.

Static Public Attributes

static const int PositionTupleSize = 3
static const int UVTupleSize = 2

Private Attributes

QVector3D m_position
QVector2D m_uv

Constructor & Destructor Documentation

UVVertex::UVVertex ()

Default constructor for UVVertex.

UVVertex::UVVertex (const QVector3D & *position*)[explicit]

Overloaded constructor for UVVertex.

Parameters:

in	*position*	The position data for the UVVertex.

UVVertex::UVVertex (const QVector3D & *position*, const QVector2D & *uv*)

Overloaded constructor for UVVertex.

Parameters:

in	*position*	The position data for the UVVertex.
in	*uv*	The uv data for the UVVertex.

Member Function Documentation

const QVector3D & UVVertex::position () const

Gets the position.

Returns:

The position.

int UVVertex::positionOffset ()[static]

Calculates the position offset within UVVertex.

Returns:

The position offset.

void UVVertex::setPosition (const QVector3D & *position*)

Sets position to a new position.

Parameters:

in	*position*	The new position.

void UVVertex::setUV (const QVector2D & *uv*)

Sets uv to a new uv.

Parameters:

in	*uv*	The new uv.

int UVVertex::stride ()[static]

Calculates the stride of UVVertex.

Returns:

The stride of UVVertex.

const QVector2D & UVVertex::uv () const

Gets the uv.

Returns:

The uv.

int UVVertex::uvOffset ()[static]

Calcualtes the uv offset within UVVertex.

Returns:

The uv offset.

Member Data Documentation

QVector3D UVVertex::m_position[private]

QVector2D UVVertex::m_uv[private]

const int UVVertex::PositionTupleSize = 3[static]

const int UVVertex::UVTupleSize = 2[static]

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

46 src/3D/uvVertex.h

47 src/3D/uvVertex.cpp

Wall Class Reference

`#include <wall.h>`

Public Member Functions

Wall (btVector3 size, btVector3 location)
Constructor for Wall.

~Wall ()
Destructor for Wall.

Public Attributes

btRigidBody * RigidBody

Private Attributes

btTriangleMesh * m_triMesh
btCollisionShape * m_collisionShape
btDefaultMotionState * m_motionState
btVector3 m_inertia
btScalar m_mass
btRigidBody::btRigidBodyConstructionInfo * m_rigidBodyCI

Constructor & Destructor Documentation

Wall::Wall (btVector3 *size*, btVector3 *location*)

Constructor for Wall.

Parameters:

in	*size*	The size of the wall.
in	*location*	The location of the wall.

Wall::~Wall ()

Destructor for Wall.

Member Data Documentation

btCollisionShape* Wall::m_collisionShape `[private]`

btVector3 Wall::m_inertia `[private]`

btScalar Wall::m_mass `[private]`

btDefaultMotionState* Wall::m_motionState `[private]`

btRigidBody::btRigidBodyConstructionInfo* Wall::m_rigidBodyCI `[private]`

btTriangleMesh* Wall::m_triMesh `[private]`

btRigidBody* Wall::RigidBody

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

48 src/GameObjects/wall.h
49 src/GameObjects/wall.cpp

File Documentation

bugs.md File Reference

build.md File Reference

build/moc_mainMenuWidget.cpp File Reference

```
#include "../src/UI/mainMenuWidget.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
```

Classes

struct qt_meta_stringdata_MainMenuWidget_t

Macros

#define QT_MOC_LITERAL(idx, ofs, len)

Variables

static const
qt_meta_stringdata_MainMenuWidget_t qt_meta_stringdata_MainMenuWidget
static const uint qt_meta_data_MainMenuWidget []

Macro Definition Documentation

#define QT_MOC_LITERAL(idx, ofs, len)

Value:Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \

```
    qptrdiff(offsetof(qt_meta_stringdata_MainMenuWidget_t, stringdata0) + ofs \
        - idx * sizeof(QByteArrayData)) \
    )
```

Variable Documentation

const uint qt_meta_data_MainMenuWidget[][static]

Initial value:= {

```
    7,
    0,
    0,    0,
    3,   14,
    0,    0,
    0,    0,
    0,    0,
    0,
    3,


    1,    0,   29,    2, 0x06 ,
    3,    0,   30,    2, 0x06 ,
    4,    0,   31,    2, 0x06 ,


QMetaType::Void,
QMetaType::Void,
QMetaType::Void,

    0
```

```
}
```

const qt_meta_stringdata_MainMenuWidget_t
qt_meta_stringdata_MainMenuWidget[static]

Initial value:= {

```
    {
QT_MOC_LITERAL(0, 0, 14),
QT_MOC_LITERAL(1, 15, 19),
QT_MOC_LITERAL(2, 35, 0),
QT_MOC_LITERAL(3, 36, 16),
QT_MOC_LITERAL(4, 53, 11)

    },
    "MainMenuWidget\0clickedSinglePlayer\0\0"
    "clickedTwoPlayer\0clickedExit"
}
```

build/moc_mainWindow.cpp File Reference

```
#include "../src/UI/mainWindow.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
```

Classes

struct qt_meta_stringdata_MainWindow_t

Macros

#define QT_MOC_LITERAL(idx, ofs, len)

Variables

static const
qt_meta_stringdata_MainWindow_t qt_meta_stringdata_MainWindow
static const uint qt_meta_data_MainWindow []

Macro Definition Documentation

#define QT_MOC_LITERAL(idx, ofs, len)

Value:Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \

```
    qptrdiff(offsetof(qt_meta_stringdata_MainWindow_t, stringdata0) + ofs \
        - idx * sizeof(QByteArrayData)) \
    )
```

Variable Documentation

const uint qt_meta_data_MainWindow[][static]

Initial value:= {

```
    7,
    0,
    0,    0,
    3,   14,
    0,    0,
    0,    0,
    0,    0,
    0,
    0,


    1,    0,   29,    2, 0x0a ,
    3,    2,   30,    2, 0x0a ,
    6,    0,   35,    2, 0x0a ,


QMetaType::Void,
QMetaType::Void, QMetaType::QString, QMetaType::QString,    4,    5,
QMetaType::Void,

    0
```

```
}
```

const qt_meta_stringdata_MainWindow_t qt_meta_stringdata_MainWindow[static]

Initial value:= {

```
    {
QT_MOC_LITERAL(0, 0, 10),
QT_MOC_LITERAL(1, 11, 16),
QT_MOC_LITERAL(2, 28, 0),
QT_MOC_LITERAL(3, 29, 10),
QT_MOC_LITERAL(4, 40, 5),
QT_MOC_LITERAL(5, 46, 5),
QT_MOC_LITERAL(6, 52, 11)

    },
    "MainWindow\0swapToTeamSelect\0\0swapToGame\0"
    "team1\0team2\0restartGame"
}
```

build/moc_oglWidget.cpp File Reference

```
#include "../src/UI/oglWidget.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
```

Classes

struct qt_meta_stringdata_OGLWidget_t

Macros

#define QT_MOC_LITERAL(idx, ofs, len)

Variables

static const
qt_meta_stringdata_OGLWidget_t qt_meta_stringdata_OGLWidget
static const uint qt_meta_data_OGLWidget []

Macro Definition Documentation

#define QT_MOC_LITERAL(idx, ofs, len)

Value:Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \

```
    qptrdiff(offsetof(qt_meta_stringdata_OGLWidget_t, stringdata0) + ofs \
        - idx * sizeof(QByteArrayData)) \
    )
```

Variable Documentation

const uint qt_meta_data_OGLWidget[][static]

Initial value:= {

```
    7,
    0,
    0,    0,
    3,   14,
    0,    0,
    0,    0,
    0,    0,
    0,
    0,


    1,    1,   29,    2, 0x0a ,
    4,    0,   32,    2, 0x09 ,
    5,    0,   33,    2, 0x09 ,


QMetaType::Void, QMetaType::Int,    3,
QMetaType::Void,
QMetaType::Void,

    0
```

```
}
```

const qt_meta_stringdata_OGLWidget_t **qt_meta_stringdata_OGLWidget** `[static]`

Initial value:= {

```
    {
QT_MOC_LITERAL(0, 0, 9),
QT_MOC_LITERAL(1, 10, 14),
QT_MOC_LITERAL(2, 25, 0),
QT_MOC_LITERAL(3, 26, 11),
QT_MOC_LITERAL(4, 38, 5),
QT_MOC_LITERAL(5, 44, 6)

    },
    "OGLWidget\0setPerspective\0\0perspective\0"
    "pause\0update"
}
```

build/moc_teamSelectWidget.cpp File Reference

```
#include "../src/UI/teamSelectWidget.h"
#include <QtCore/qbytearray.h>
#include <QtCore/qmetatype.h>
```

Classes

struct qt_meta_stringdata_TeamSelectWidget_t

Macros

#define QT_MOC_LITERAL(idx, ofs, len)

Variables

static const
qt_meta_stringdata_TeamSelectWidget_t qt_meta_stringdata_TeamSelectWidget
static const uint qt_meta_data_TeamSelectWidget []

Macro Definition Documentation

#define QT_MOC_LITERAL(idx, ofs, len)

Value:Q_STATIC_BYTE_ARRAY_DATA_HEADER_INITIALIZER_WITH_OFFSET(len, \

```
qptrdiff(offsetof(qt_meta_stringdata_TeamSelectWidget_t, stringdata0) + ofs \
    - idx * sizeof(QByteArrayData)) \
)
```

Variable Documentation

const uint qt_meta_data_TeamSelectWidget[][static]

Initial value:= {

```
    7,
    0,
    0,    0,
    3,   14,
    0,    0,
    0,    0,
    0,    0,
    0,
    2,


    1,    2,   29,    2, 0x06 ,
    5,    1,   34,    2, 0x06 ,


    7,    1,   37,    2, 0x0a ,


QMetaType::Void, QMetaType::QString, QMetaType::QString,    3,    4,
QMetaType::Void, QMetaType::QString,    6,
```

```
    QMetaType::Void, QMetaType::QString,    6,

       0
}
```

const qt_meta_stringdata_TeamSelectWidget_t qt_meta_stringdata_TeamSelectWidget[static]

Initial value:= {

```
    {
QT_MOC_LITERAL(0, 0, 16),
QT_MOC_LITERAL(1, 17, 13),
QT_MOC_LITERAL(2, 31, 0),
QT_MOC_LITERAL(3, 32, 5),
QT_MOC_LITERAL(4, 38, 5),
QT_MOC_LITERAL(5, 44, 11),
QT_MOC_LITERAL(6, 56, 4),
QT_MOC_LITERAL(7, 61, 8)

    },
    "TeamSelectWidget\0selectedTeams\0\0team1\0"
    "team2\0clickedTeam\0team\0setTeams"
}
```

build/qrc_font_resource.cpp File Reference

Classes
struct anonymous_namespace{qrc_font_resource.cpp}::initializer

Namespaces

anonymous_namespace{qrc_font_resource.cpp}

Macros

#define QT_RCC_PREPEND_NAMESPACE(name) name
#define QT_RCC_MANGLE_NAMESPACE(name) name

Functions

bool qRegisterResourceData (int, const unsigned char *, const unsigned char *, const unsigned char *)
bool qUnregisterResourceData (int, const unsigned char *, const unsigned char *, const unsigned char *)
int QT_RCC_MANGLE_NAMESPACE() qInitResources_font_resource ()
int QT_RCC_MANGLE_NAMESPACE() qCleanupResources_font_resource ()

Variables

static const unsigned char qt_resource_data []
static const unsigned char qt_resource_name []
static const unsigned char qt_resource_struct []
struct anonymous_namespace{qrc_font_resource.cpp}::initializer
    anonymous_namespace{qrc_font_resource.cpp}::dummy

Macro Definition Documentation

#define QT_RCC_MANGLE_NAMESPACE(name) name

#define QT_RCC_PREPEND_NAMESPACE(name) name

Function Documentation

int QT_RCC_MANGLE_NAMESPACE() qCleanupResources_font_resource ()

int QT_RCC_MANGLE_NAMESPACE() qInitResources_font_resource ()

bool qRegisterResourceData (int , const unsigned char * , const unsigned char * , const unsigned char *)

bool qUnregisterResourceData (int , const unsigned char * , const unsigned char * , const unsigned char *)

Variable Documentation

const unsigned char qt_resource_data[][static]

const unsigned char qt_resource_name[][static]

Initial value:= {

```
  0x0,0x5,
  0x0,0x6d,0x65,0xb3,
  0x0,0x66,
  0x0,0x6f,0x0,0x6e,0x0,0x74,0x0,0x73,

  0x0,0x7,
  0x2,0xcf,0x5b,0x6,
  0x0,0x4e,
  0x0,0x48,0x0,0x4c,0x0,0x2e,0x0,0x74,0x0,0x74,0x0,0x66,

}
```

const unsigned char qt_resource_struct[][static]

Initial value:= {

```
  0x0,0x0,0x0,0x0,0x0,0x0,0x2,0x0,0x0,0x0,0x1,0x0,0x0,0x0,0x1,

  0x0,0x0,0x0,0x0,0x0,0x0,0x2,0x0,0x0,0x0,0x1,0x0,0x0,0x0,0x2,

  0x0,0x0,0x0,0x0,0x10,0x0,0x0,0x0,0x0,0x0,0x0,0x1,0x0,0x0,0x0,0x0,

}
```

build/qrc_shader_resource.cpp File Reference

Classes
struct anonymous_namespace{qrc_shader_resource.cpp}::initializer

Namespaces

anonymous_namespace{qrc_shader_resource.cpp}

Macros

#define QT_RCC_PREPEND_NAMESPACE(name) name
#define QT_RCC_MANGLE_NAMESPACE(name) name

Functions

bool qRegisterResourceData (int, const unsigned char *, const unsigned char *, const unsigned char *)
bool qUnregisterResourceData (int, const unsigned char *, const unsigned char *, const unsigned char *)
int QT_RCC_MANGLE_NAMESPACE() qInitResources_shader_resource ()
int QT_RCC_MANGLE_NAMESPACE() qCleanupResources_shader_resource ()

Variables

static const unsigned char qt_resource_data []
static const unsigned char qt_resource_name []
static const unsigned char qt_resource_struct []
struct anonymous_namespace{qrc_shader_resource.cpp}::initializer
    anonymous_namespace{qrc_shader_resource.cpp}::dummy

Macro Definition Documentation

#define QT_RCC_MANGLE_NAMESPACE(name) name

#define QT_RCC_PREPEND_NAMESPACE(name) name

Function Documentation

int QT_RCC_MANGLE_NAMESPACE() qCleanupResources_shader_resource ()

int QT_RCC_MANGLE_NAMESPACE() qInitResources_shader_resource ()

bool qRegisterResourceData (int , const unsigned char * , const unsigned char * , const unsigned char *)

bool qUnregisterResourceData (int , const unsigned char * , const unsigned char * , const unsigned char *)

Variable Documentation

const unsigned char qt_resource_data[][static]

const unsigned char qt_resource_name[][static]

Initial value:= {

```
  0x0,0x6,
  0x7,0x9e,0x7a,0xc2,
  0x0,0x73,
  0x0,0x68,0x0,0x61,0x0,0x64,0x0,0x65,0x0,0x72,

  0x0,0xe,
  0xb,0xc0,0xbc,0x93,
  0x0,0x63,

0x0,0x6f,0x0,0x6c,0x0,0x6f,0x0,0x72,0x0,0x53,0x0,0x68,0x0,0x61,0x0,0x64,0x0,0x65,0x0,0
x72,0x0,0x2e,0x0,0x76,0x0,0x73,

  0x0,0xb,
  0x7,0x55,0x27,0x13,
  0x0,0x75,

0x0,0x76,0x0,0x53,0x0,0x68,0x0,0x61,0x0,0x64,0x0,0x65,0x0,0x72,0x0,0x2e,0x0,0x66,0x0,0
x73,

  0x0,0xb,
  0x7,0x55,0x26,0x13,
  0x0,0x75,

0x0,0x76,0x0,0x53,0x0,0x68,0x0,0x61,0x0,0x64,0x0,0x65,0x0,0x72,0x0,0x2e,0x0,0x76,0x0,0
x73,

  0x0,0xe,
  0xb,0xc0,0xbd,0x93,
  0x0,0x63,
```

```
0x0,0x6f,0x0,0x6c,0x0,0x6f,0x0,0x72,0x0,0x53,0x0,0x68,0x0,0x61,0x0,0x64,0x0,0x65,0x0,0
x72,0x0,0x2e,0x0,0x66,0x0,0x73,

}
```

const unsigned char qt_resource_struct[][static]

Initial value:= {

```
  0x0,0x0,0x0,0x0,0x0,0x2,0x0,0x0,0x0,0x1,0x0,0x0,0x0,0x1,

  0x0,0x0,0x0,0x0,0x0,0x2,0x0,0x0,0x0,0x4,0x0,0x0,0x0,0x2,

  0x0,0x0,0x0,0x50,0x0,0x0,0x0,0x0,0x0,0x1,0x0,0x0,0x1,0x9a,

  0x0,0x0,0x0,0x34,0x0,0x0,0x0,0x0,0x0,0x1,0x0,0x0,0x1,0x19,

  0x0,0x0,0x0,0x12,0x0,0x0,0x0,0x0,0x0,0x1,0x0,0x0,0x0,0x0,

  0x0,0x0,0x0,0x6c,0x0,0x0,0x0,0x0,0x0,0x1,0x0,0x0,0x2,0xbd,

}
```

install.md File Reference

README.md File Reference

src/3D/colorVertex.cpp File Reference

```
#include "colorVertex.h"
```

src/3D/colorVertex.h File Reference

```
#include <QVector3D>
#include <QVector4D>
```

Classes

class ColorVertex

src/3D/modelLoader.cpp File Reference

```
#include "modelLoader.h"
```

src/3D/modelLoader.h File Reference

```
#include <QString>
#include <QDebug>
#include "uvVertex.h"
#include "colorVertex.h"
#include <assimp/Importer.hpp>
#include <assimp/scene.h>
#include <assimp/postprocess.h>
#include <btBulletDynamicsCommon.h>
```

Namespaces

ModelLoader

Functions

bool ModelLoader::loadUVModel (QString filePath, UVVertex *&geometry, int &numVertices)

Loads a UV-based model based on a file path.

bool ModelLoader::loadColorModel (QString filePath, ColorVertex *&geometry, int &numVertices)

Loads a material-based model based on a file path.

bool ModelLoader::loadTriMesh (QString filePath, btTriangleMesh *&collisionMesh)

src/3D/renderable.h File Reference

```
#include <QOpenGLFunctions>
#include <QMatrix4x4>
#include "Controls/camera3d.h"
```

Classes

class Renderable

src/3D/transform3d.cpp File Reference

`#include "transform3d.h"`

Functions

QDebug operator<< (QDebug dbg, const Transform3D &transform)

Overloaded operator<< for the Transform3D class.

Function Documentation

QDebug operator<< (QDebug *dbg*, const Transform3D & *transform*)

Overloaded operator<< for the Transform3D class.

Parameters:

in	*dbg*	The QDebug context to output to.
in	*transform*	The Transform3D class to output.

Returns:

The formatted QDebug output of Transform3D's properties

src/3D/transform3d.h File Reference

```
#include <QVector3D>
#include <QQuaternion>
#include <QMatrix4x4>
```

Classes

class Transform3D

Functions

QDebug operator<< (QDebug dbg, const Transform3D &transform)

Overloaded operator<< for the Transform3D class.

Function Documentation

QDebug operator<< (QDebug *dbg*, const Transform3D & *transform*)

Overloaded operator<< for the Transform3D class.

Parameters:

in	*dbg*	The QDebug context to output to.
in	*transform*	The Transform3D class to output.

Returns:

The formatted QDebug output of Transform3D's properties

src/3D/uvVertex.cpp File Reference

```
#include "uvVertex.h"
```

src/3D/uvVertex.h File Reference

```
#include <QVector3D>
#include <QVector2D>
```

Classes

class UVVertex

src/Controls/camera3d.cpp File Reference

`#include "camera3d.h"`

Functions

QDebug operator<< (QDebug dbg, const Camera3D &transform)

Overloaded operator<< for the Camera3D class.

Function Documentation

QDebug operator<< (QDebug *dbg*, const Camera3D & *transform*)

Overloaded operator<< for the Camera3D class.

Parameters:

in	*dbg*	The QDebug context to output to.
in	*transform*	The Camera3D class to output.

Returns:

The formatted QDebug output of Camera3D's properties

src/Controls/camera3d.h File Reference

```
#include <QVector3D>
#include <QQuaternion>
#include <QMatrix4x4>
```

Classes

class Camera3D

Functions

QDebug operator<< (QDebug dbg, const Camera3D &transform)

Overloaded operator<< for the Camera3D class.

Function Documentation

QDebug operator<< (QDebug *dbg*, const Camera3D & *transform*)

Overloaded operator<< for the Camera3D class.

Parameters:

in	*dbg*	The QDebug context to output to.
in	*transform*	The Camera3D class to output.

Returns:

The formatted QDebug output of Camera3D's properties

src/Controls/input.cpp File Reference

`#include "input.h"`

Classes

struct InputInstance< T >

Typedefs

typedef InputInstance< Qt::Key > KeyInstance
typedef InputInstance
< Qt::MouseButton > ButtonInstance
typedef std::vector< KeyInstance > KeyContainer
typedef std::vector
< ButtonInstance > ButtonContainer

Functions

static KeyContainer::iterator FindKey (Qt::Key value)
static ButtonContainer::iterator FindButton (Qt::MouseButton value)
template<typename TPair > static void UpdateStates (TPair &instance)
template<typename TPair > static bool CheckReleased (const TPair &instance)
template<typename Container > static void Update (Container &container)

Variables

static KeyContainer keyInstances
static ButtonContainer buttonInstances
static QPoint mouseCurrentPosition
static QPoint mousePreviousPosition
static QPoint mouseDeltaXy

Typedef Documentation

typedef std::vector<ButtonInstance> ButtonContainer

typedef InputInstance<Qt::MouseButton> ButtonInstance

typedef std::vector<KeyInstance> KeyContainer

typedef InputInstance<Qt::Key> KeyInstance

Function Documentation

template<typename TPair > static bool CheckReleased (const TPair & *instance*)`[inline]`, `[static]`

static ButtonContainer::iterator FindButton (Qt::MouseButton *value*)`[inline]`, `[static]`

static KeyContainer::iterator FindKey (Qt::Key *value*)`[inline]`, `[static]`

template<typename Container > static void Update (Container & *container*)`[inline]`, `[static]`

template<typename TPair > static void UpdateStates (TPair & *instance*)`[inline]`, `[static]`

Variable Documentation

ButtonContainer buttonInstances`[static]`

KeyContainer keyInstances`[static]`

QPoint mouseCurrentPosition`[static]`

QPoint mouseDeltaXy`[static]`

QPoint mousePreviousPosition`[static]`

src/Controls/input.h File Reference

```
#include <Qt>
#include <QPoint>
#include <QCursor>
#include <vector>
#include <algorithm>
```

Classes

class Input

src/Entity/colorEntity.cpp File Reference

```
#include "colorEntity.h"
```

src/Entity/colorEntity.h File Reference

```
#include <QOpenGLFunctions>
#include <QOpenGLBuffer>
#include <QOpenGLVertexArrayObject>
#include <QOpenGLTexture>
#include <QOpenGLShaderProgram>
#include "3D/renderable.h"
#include "3D/modelLoader.h"
#include "3D/transform3d.h"
#include "3D/colorVertex.h"
```

Classes

class ColorEntity

src/Entity/colorPhysicsEntity.cpp File Reference

```
#include "colorPhysicsEntity.h"
```

src/Entity/colorPhysicsEntity.h File Reference

```
#include "colorEntity.h"
#include <btBulletDynamicsCommon.h>
```

Classes

class ColorPhysicsEntity

src/Entity/uvEntity.cpp File Reference

```
#include "uvEntity.h"
```

src/Entity/uvEntity.h File Reference

```
#include <QOpenGLFunctions>
#include <QOpenGLBuffer>
#include <QOpenGLVertexArrayObject>
#include <QOpenGLTexture>
#include <QOpenGLShaderProgram>
#include "3D/renderable.h"
#include "3D/modelLoader.h"
#include "3D/transform3d.h"
#include "3D/uvVertex.h"
```

Classes

class UVEntity

src/Entity/uvPhysicsEntity.cpp File Reference

```
#include "uvPhysicsEntity.h"
```

src/Entity/uvPhysicsEntity.h File Reference

```
#include "uvEntity.h"
#include <btBulletDynamicsCommon.h>
```

Classes

class UVPhysicsEntity

src/GameObjects/hockeyPaddle.cpp File Reference

```
#include "hockeyPaddle.h"
```

Functions

RigidBody setAngularFactor (btVector3(0, 1, 0))

Function Documentation

RigidBody setAngularFactor (btVector3(0, 1, 0))

src/GameObjects/hockeyPaddle.h File Reference

```
#include "Entity/colorPhysicsEntity.h"
```

Classes

class HockeyPaddle

src/GameObjects/hockeyPuck.cpp File Reference

```
#include "hockeyPuck.h"
```

src/GameObjects/hockeyPuck.h File Reference

```
#include "Entity/colorPhysicsEntity.h"
```

Classes

class HockeyPuck

src/GameObjects/hockeyTable.cpp File Reference

```
#include "hockeyTable.h"
#include <QFileInfo>
```

src/GameObjects/hockeyTable.h File Reference

```
#include <QMediaPlayer>
#include "Entity/colorPhysicsEntity.h"
```

Classes

class HockeyTable

src/GameObjects/skybox.cpp File Reference

```
#include "skybox.h"
```

src/GameObjects/skybox.h File Reference

```
#include "Entity/uvEntity.h"
```

Classes

class Skybox

src/GameObjects/wall.cpp File Reference

```
#include "wall.h"
```

src/GameObjects/wall.h File Reference

```
#include <btBulletDynamicsCommon.h>
```

Classes

class Wall

src/main.cpp File Reference

```
#include <QApplication>
#include "UI/mainWindow.h"
```

Functions

int main (int argc, char **argv)

Function Documentation

int main (int *argc*, char ** *argv*)

src/UI/mainMenuWidget.cpp File Reference

```
#include "mainMenuWidget.h"
```

src/UI/mainMenuWidget.h File Reference

```
#include <QApplication>
#include <QOpenGLWidget>
#include <QBoxLayout>
#include <QPushButton>
#include <QFont>
#include <QPixmap>
#include <QLabel>
```

Classes

class MainMenuWidget

src/UI/mainWindow.cpp File Reference

```
#include "mainWindow.h"
```

src/UI/mainWindow.h File Reference

```
#include <QMainWindow>
#include <QMenuBar>
#include <QMenu>
#include <QAction>
#include <QActionGroup>
#include <QKeySequence>
#include <QSignalMapper>
#include "mainMenuWidget.h"
#include "teamSelectWidget.h"
#include "oglWidget.h"
```

Classes

class MainWindow

src/UI/oglWidget.cpp File Reference

```
#include "oglWidget.h"
```

src/UI/oglWidget.h File Reference

```
#include <QApplication>
#include <QOpenGLWidget>
#include <QOpenGLFunctions>
#include <QPainter>
#include <QFont>
#include <QRect>
#include <QFontDatabase>
#include <QLabel>
#include <QPixmap>
#include <QKeyEvent>
#include <QMouseEvent>
#include <QMap>
#include <QMatrix4x4>
#include <QMediaPlayer>
#include <QFileInfo>
#include <QDebug>
#include <QString>
#include <btBulletDynamicsCommon.h>
#include "Controls/input.h"
#include "Controls/camera3d.h"
#include "3D/renderable.h"
#include "GameObjects/hockeyTable.h"
#include "GameObjects/hockeyPuck.h"
#include "GameObjects/hockeyPaddle.h"
#include "GameObjects/skybox.h"
#include "GameObjects/wall.h"
```

Classes

class OGLWidget
struct OGLWidget::GoalCallback
struct OGLWidget::Goal2Callback
struct OGLWidget::PuckCallback

src/UI/teamSelectWidget.cpp File Reference

```
#include "teamSelectWidget.h"
```

src/UI/teamSelectWidget.h File Reference

```
#include <QApplication>
#include <QOpenGLWidget>
#include <QGridLayout>
#include <QPushButton>
#include <QIcon>
#include <QSignalMapper>
```

Classes

class TeamSelectWidget

Index

INDEX