Platforma 6DOF

1.0

Generated by Doxygen 1.9.8

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

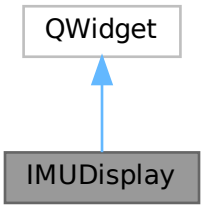
Class Documentation

4.1 IMUDisplay Class Reference

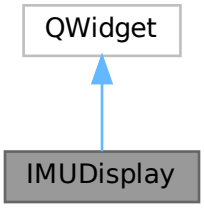
Widget for displaying IMU sensor data (accelerometer and gyroscope)

`#include <imudisplay.h>`

Inheritance diagram for IMUDisplay:



Collaboration diagram for IMUDisplay:



Public Member Functions

- IMUDisplay (QWidget *parent=nullptr)

  Constructs the IMU display widget.
- void updateValues (float ax, float ay, float az, float gx, float gy, float gz)

  Updates all displayed IMU values.

4.1.1 Detailed Description

Widget for displaying IMU sensor data (accelerometer and gyroscope)

Provides visual representation of 3-axis accelerometer and gyroscope data with proper units and formatting. Designed for real-time sensor monitoring.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 IMUDisplay()

```
IMUDisplay::IMUDisplay (
            QWidget * parent = nullptr )  [explicit]
```

Constructs the IMU display widget.

Constructs and initializes the IMU display.

Parameters

parent	Parent widget (optional)
parent	Parent widget (optional)

Sets up the UI layout with:

- Header title
- Separator lines
- 3-axis accelerometer display (m/s^2)
- 3-axis gyroscope display (°/s)
- Unified styling

4.1.3 Member Function Documentation

4.1.3.1 updateValues()

```
void IMUDisplay::updateValues (
            float ax,
            float ay,
```

```
            float az,
            float gx,
            float gy,
            float gz )
```

Updates all displayed IMU values.

Updates all displayed values with new sensor data.

Parameters

ax	Accelerometer X-axis value (raw sensor units)
ay	Accelerometer Y-axis value (raw sensor units)
az	Accelerometer Z-axis value (raw sensor units)
gx	Gyroscope X-axis value (raw sensor units)
gy	Gyroscope Y-axis value (raw sensor units)
gz	Gyroscope Z-axis value (raw sensor units)

Note

Accelerometer values are automatically converted to m/s^2

Parameters

ax	Raw accelerometer X value
ay	Raw accelerometer Y value
az	Raw accelerometer Z value
gx	Raw gyroscope X value
gy	Raw gyroscope Y value
gz	Raw gyroscope Z value

Performs:

- Accelerometer conversion (0.000565 factor to m/s^2)
- Value formatting (2 decimal places for accel, 0 for gyro)
- Immediate UI update

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

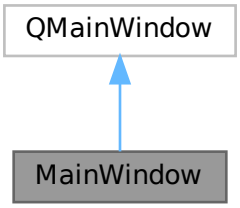
- Platform_app/imudisplay.h
- Platform_app/imudisplay.cpp

4.2 MainWindow Class Reference

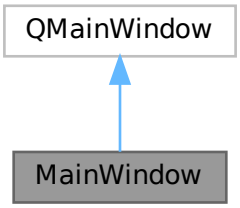
Central application window managing serial communication and visualization.

```
#include <mainwindow.h>
```

Inheritance diagram for MainWindow:



Collaboration diagram for MainWindow:



Signals

- void dataProcessed (int intValue, float floatValue)

  Signal emitted when data is processed.

Public Member Functions

- MainWindow (QWidget ∗parent=nullptr)

  Constructs the main application window.
- ∼MainWindow ()

  Destructor - ensures clean serial port closure.

4.2.1 Detailed Description

Central application window managing serial communication and visualization.

Handles:

- Serial port connection management
- IMU data reception and parsing
- 3D platform visualization
- IMU data display

4.2.2 Constructor & Destructor Documentation

4.2.2.1 MainWindow()

```
MainWindow::MainWindow (
            QWidget * parent = nullptr )
```

Constructs the main application window.

Parameters

parent	Parent widget (optional)
parent	Parent widget (optional)

Initializes:

- Serial port interface
- 3D visualization widget
- IMU data display
- Control panel with port selection

4.2.2.2 ∼MainWindow()

```
MainWindow::∼MainWindow ( )
```

Destructor - ensures clean serial port closure.

Destructor - ensures proper resource cleanup.

4.2.3 Member Function Documentation

4.2.3.1 dataProcessed

```
void MainWindow::dataProcessed (
            int intValue,
            float floatValue )  [signal]
```

Signal emitted when data is processed.

Parameters

intValue	Example integer value
floatValue	Example float value

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

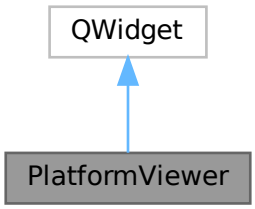
- Platform_app/mainwindow.h
- Platform_app/mainwindow.cpp

4.3 PlatformViewer Class Reference

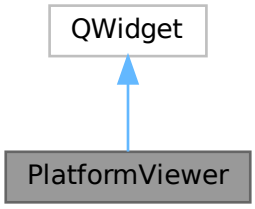
3D visualization widget that simulates platform movement based on IMU data

`#include <platformviewer.h>`

Inheritance diagram for PlatformViewer:



Collaboration diagram for PlatformViewer:



Public Member Functions

- PlatformViewer (QWidget *parent=nullptr)

  Constructs the 3D platform viewer.
- QSize sizeHint () const override

  Recommended widget size.
- void updatePlatformOrientation (int ax, int ay, int az)

  Updates platform orientation based on IMU data.

4.3.1 Detailed Description

3D visualization widget that simulates platform movement based on IMU data

Displays a 3D rectangular platform that tilts according to accelerometer inputs. Provides interactive camera controls for viewing from different angles.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 PlatformViewer()

```
PlatformViewer::PlatformViewer (
            QWidget * parent = nullptr )  [explicit]
```

Constructs the 3D platform viewer.

Constructs and initializes the 3D platform viewer.

Parameters

parent	Parent widget (optional)
parent	Parent widget (optional)

Creates a 3D scene containing:

- Green platform (5×0.5×3 units)
- Directional lighting
- Orbit camera controller
- Fixed size container (500×400px)

4.3.3 Member Function Documentation

4.3.3.1 sizeHint()

```
QSize PlatformViewer::sizeHint ( ) const  [inline], [override]
```

Recommended widget size.

Returns

Fixed size of 500x400 pixels

4.3.3.2 updatePlatformOrientation()

```
void PlatformViewer::updatePlatformOrientation (
            int ax,
            int ay,
            int az )
```

Updates platform orientation based on IMU data.

Updates platform orientation based on IMU accelerometer data.

Parameters

ax	Accelerometer X-axis raw value
ay	Accelerometer Y-axis raw value
az	Accelerometer Z-axis raw value

Note

Input values are normalized (17000 = 1g) and clamped

Parameters

ax	X-axis accelerometer value (raw)
ay	Y-axis accelerometer value (raw)
az	Z-axis accelerometer value (raw)

Converts raw accelerometer values to:

1. Normalized tilt values (-1.0 to 1.0)
2. Euler angles (pitch and roll)
3. Platform rotation quaternion

Note

Uses 17000 as normalization factor (1g)

Applies 25° scaling factor for visible tilt

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

- Platform_app/platformviewer.h
- Platform_app/platformviewer.cpp

Chapter 5

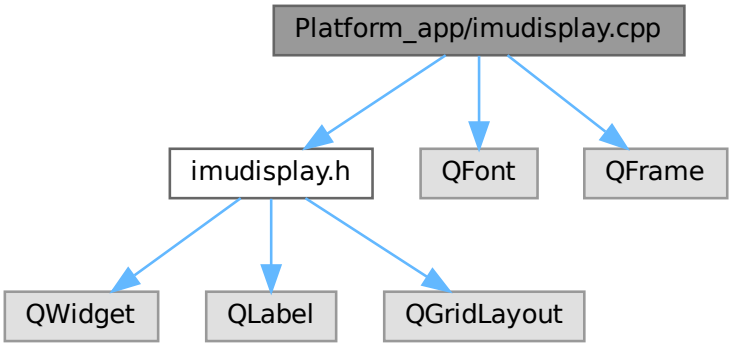
File Documentation

5.1 Platform_app/imudisplay.cpp File Reference

Implementation of IMU data display widget.

```
#include "imudisplay.h"
#include <QFont>
#include <QFrame>
```

Include dependency graph for imudisplay.cpp:



5.1.1 Detailed Description

Implementation of IMU data display widget.

Author

Piotr Siembab

Date

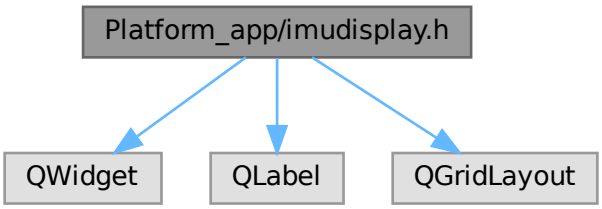
18.04.2025

5.2 Platform_app/imudisplay.h File Reference

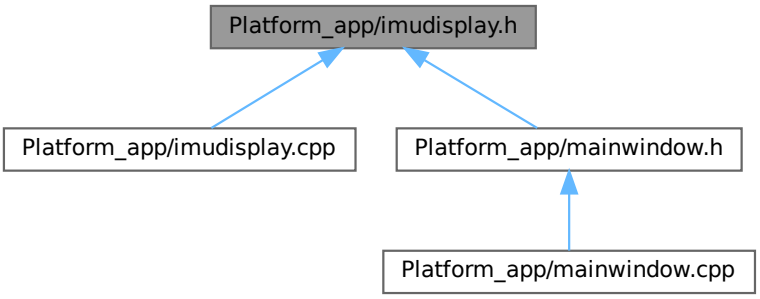
Header for IMU data visualization widget.

```
#include <QWidget>
#include <QLabel>
#include <QGridLayout>
```

Include dependency graph for imudisplay.h:



This graph shows which files directly or indirectly include this file:



Classes

- class IMUDisplay

  Widget for displaying IMU sensor data (accelerometer and gyroscope)

5.2.1 Detailed Description

Header for IMU data visualization widget.

Author

Piotr Siembab

Date

18.04.2025

5.3 imudisplay.h

Go to the documentation of this file.

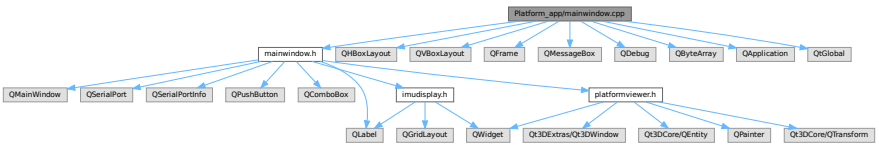
```
00001
00008 #ifndef IMUDISPLAY_H
00009 #define IMUDISPLAY_H
00010
00011 #include <QWidget>
00012 #include <QLabel>
00013 #include <QGridLayout>
00014
00022 class IMUDisplay : public QWidget {
00023     Q_OBJECT
00024 public:
00029     explicit IMUDisplay(QWidget *parent = nullptr);
00030
00041     void updateValues(float ax, float ay, float az, float gx, float gy, float gz);
00042
00043 private:
00048     QLabel *createValueLabel();
00049
00056     void setupAxisDisplay(QGridLayout *layout, const QString &name, int row);
00057
00062     struct AxisDisplay {
00063         QLabel *nameLabel;
00064         QLabel *valueLabel;
00065     };
00066
00067     AxisDisplay m_accelX, m_accelY, m_accelZ;
00068     AxisDisplay m_gyroX, m_gyroY, m_gyroZ;
00069 };
00070
00071 #endif // IMUDISPLAY_H
```

5.4 Platform_app/mainwindow.cpp File Reference

Implementation of main application window.

```
#include "mainwindow.h"
#include <QHBoxLayout>
#include <QVBoxLayout>
#include <QFrame>
#include <QMessageBox>
#include <QDebug>
#include <QByteArray>
#include <QApplication>
#include <QtGlobal>
```

Include dependency graph for mainwindow.cpp:



5.4.1 Detailed Description

Implementation of main application window.

Author

Piotr Siembab

Date

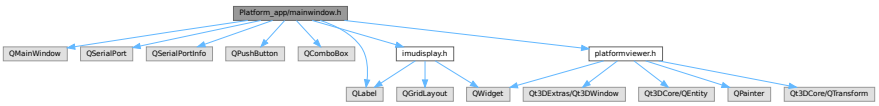
18.04.2025

5.5 Platform_app/mainwindow.h File Reference

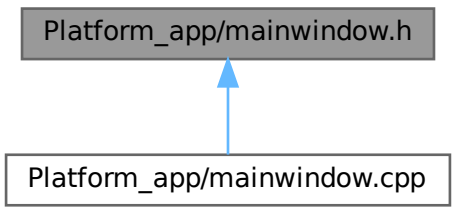
Main application window for IMU visualization system.

```
#include <QMainWindow>
#include <QSerialPort>
#include <QSerialPortInfo>
#include <QPushButton>
#include <QComboBox>
#include <QLabel>
#include "platformviewer.h"
#include "imudisplay.h"
```

Include dependency graph for mainwindow.h:



This graph shows which files directly or indirectly include this file:



Classes

- class MainWindow

  Central application window managing serial communication and visualization.

5.5.1 Detailed Description

Main application window for IMU visualization system.

Author

Piotr Siembab

Date

18.04.2025

5.6 mainwindow.h

Go to the documentation of this file.

```
00001
00008 #ifndef MAINWINDOW_H
00009 #define MAINWINDOW_H
00010
00011 #include <QMainWindow>
00012 #include <QSerialPort>
00013 #include <QSerialPortInfo>
00014 #include <QPushButton>
00015 #include <QComboBox>
00016 #include <QLabel>
00017 #include "platformviewer.h"
00018 #include "imudisplay.h"
00019
00030 class MainWindow : public QMainWindow
00031 {
00032     Q_OBJECT
00033 public:
00038     MainWindow(QWidget *parent = nullptr);
00039
00043     ~MainWindow();
00044
00045 private slots:
00049     void refreshPorts();
00050
00054     void toggleConnection();
00055
00059     void readSerialData();
00060
00066     uint8_t calculateCrc8(const QList<QByteArray>& data);
00067
00068 private:
00073     void updateConnectionStatus(bool connected);
00074
00075     QSerialPort *serial;
00076     QPushButton *refreshButton;
00077     QPushButton *connectButton;
00078     QComboBox *portComboBox;
00079     QLabel *statusLabel;
00080     PlatformViewer *platformViewer;
00081     IMUDisplay *imuDisplay;
00082
00083 signals:
00089     void dataProcessed(int intValue, float floatValue);
00090 };
00091
00092 #endif // MAINWINDOW_H
```

5.7 Platform_app/platformviewer.cpp File Reference

Implementation of 3D platform visualization.

```
#include "platformviewer.h"
#include <Qt3DExtras/Qt3DWindow>
```

```
#include <Qt3DCore/QEntity>
#include <Qt3DCore/QTransform>
#include <Qt3DExtras/QPhongMaterial>
#include <Qt3DExtras/QCuboidMesh>
#include <Qt3DExtras/QOrbitCameraController>
#include <QVBoxLayout>
#include <Qt3DRender/QCamera>
#include <QPointLight>
```

Include dependency graph for platformviewer.cpp:

5.7.1 Detailed Description

Implementation of 3D platform visualization.

Author

Piotr Siembab

Date

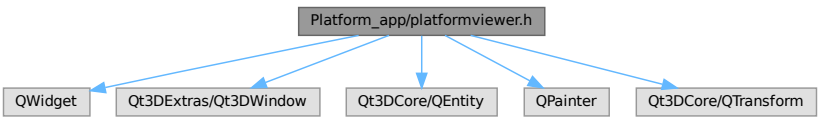
18.04.2025

5.8 Platform_app/platformviewer.h File Reference

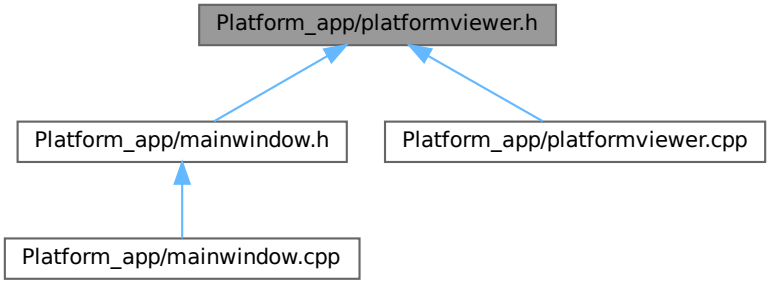
3D platform visualization widget for IMU data

```
#include <QWidget>
#include <Qt3DExtras/Qt3DWindow>
#include <Qt3DCore/QEntity>
#include <QPainter>
#include <Qt3DCore/QTransform>
```

Include dependency graph for platformviewer.h:



This graph shows which files directly or indirectly include this file:



Classes

- class PlatformViewer

  3D visualization widget that simulates platform movement based on IMU data

5.8.1 Detailed Description

3D platform visualization widget for IMU data

Author

Piotr Siembab

Date

18.04.2025

5.9 platformviewer.h

Go to the documentation of this file.

```
00001
00008 #ifndef PLATFORMVIEWER_H
00009 #define PLATFORMVIEWER_H
00010
00011 #include <QWidget>
00012 #include <Qt3DExtras/Qt3DWindow>
00013 #include <Qt3DCore/QEntity>
00014 #include <QPainter>
00015 #include <Qt3DCore/QTransform>
00016
00024 class PlatformViewer : public QWidget
00025 {
00026     Q_OBJECT
00027 public:
00032     explicit PlatformViewer(QWidget *parent = nullptr);
00033
00038     QSize sizeHint() const override { return QSize(500, 400); }
00039
00047     void updatePlatformOrientation(int ax, int ay, int az);
00048
00049 private:
00050     Qt3DExtras::Qt3DWindow *m_view;
00051     QWidget *m_container;
00052     Qt3DCore::QTransform *m_platformTransform;
00053 };
00054
00055 #endif // PLATFORMVIEWER_H
```

Index