HeartRate2Go - Qt Anwendung

Generated by Doxygen 1.8.8

Thu Jan 15 2015 11:01:15

Contents

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

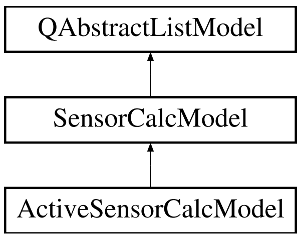
Class Documentation

4.1 ActiveSensorCalcModel Class Reference

The ActiveSensorCalcModel class This class represents a model for active calc values.

```
#include <activesensorcalcmodel.h>
```

Inheritance diagram for ActiveSensorCalcModel:



Public Types

- enum SensorCalcRoles { ACTIVE_SENSOR_CALC_VALUE_ROLE = 0, ACTIVE_SENSOR_CALC_DES↩ CRIPTION_ROLE, ACTIVE_SENSOR_CALC_UNITOFMEASUREMENT_ROLE }

  The SensorCalcRoles enum Inlcude all roles from model.

Public Member Functions

- ActiveSensorCalcModel (SensorModel &aModel)

  ActiveSensorCalcModel Constructor to init all attributes.
- QVariant data (const QModelIndex &aIndex, int aRole=Qt::DisplayRole) const

  data Return to a index and role a QVariant value for view
- int rowCount (const QModelIndex &aParent=QModelIndex()) const

  rowCount Actual count of rows in model
- void updateCalcValues (const SensorModel &aModel)

  updateCalcValues Update all calc values when data from model change

Protected Member Functions

- QHash< int, QByteArray > roleNames () const

  roleNames Connect Roles on view with roles in model

Additional Inherited Members

4.1.1 Detailed Description

The ActiveSensorCalcModel class This class represents a model for active calc values.

4.1.2 Member Enumeration Documentation

4.1.2.1 enum ActiveSensorCalcModel::SensorCalcRoles

The SensorCalcRoles enum Inlcude all roles from model.

Enumerator

ACTIVE_SENSOR_CALC_VALUE_ROLE Role for a single active calc value

ACTIVE_SENSOR_CALC_DESCRIPTION_ROLE Role for a single active calc description

ACTIVE_SENSOR_CALC_UNITOFMEASUREMENT_ROLE Role for a single active calc unit of measurement

4.1.3 Constructor & Destructor Documentation

4.1.3.1 ActiveSensorCalcModel::ActiveSensorCalcModel (SensorModel & *aModel*)

ActiveSensorCalcModel Constructor to init all attributes.

Parameters

aModel	Reference to a SensorModel to store all data to make some calculations

4.1.4 Member Function Documentation

4.1.4.1 QVariant ActiveSensorCalcModel::data (const QModelIndex & *aIndex*, int *aRole* = `Qt::DisplayRole`) const

data Return to a index and role a QVariant value for view

Parameters

aIndex	Index of model
aRole	Current Role

Returns

QVariant value with value and role

This function is used by model/view on QT

4.1.4.2 QHash< int, QByteArray > ActiveSensorCalcModel::roleNames ( ) const `[protected]`

roleNames Connect Roles on view with roles in model

Returns

QHash with the connected roles

This function is used by model/view on QT

4.1.4.3 int ActiveSensorCalcModel::rowCount (const QModelIndex & *aParent* = `QModelIndex()`) const

rowCount Actual count of rows in model

Parameters

aParent	-

Returns

Count of rows in model

This function is used by model/view on QT

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

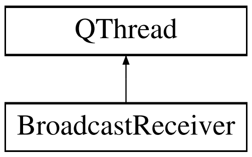
- Model/activesensorcalcmodel.h
- Model/activesensorcalcmodel.cpp

4.2 BroadcastReceiver Class Reference

The BroadcastReceiver class implements an UDP-Server for Server-Discovery.

```
#include <BroadcastReceiver.h>
```

Inheritance diagram for BroadcastReceiver:



Public Slots

- void readyRead ()

  readyRead is called on incoming datagram and handles it

Signals

- void error (QUdpSocket::SocketError socketerror)

  error signal on socket errors

Public Member Functions

- BroadcastReceiver (QObject ∗aParent=0)

  BroadcastReceiver Constructor initializes all attributes of class.
- virtual ∼BroadcastReceiver ()

  BroadcastReceiver destructor marks UDP-socket as deleteable.
- void run ()

  run implements the QThread Threadloop

4.2.1 Detailed Description

The BroadcastReceiver class implements an UDP-Server for Server-Discovery.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 BroadcastReceiver::BroadcastReceiver (QObject ∗ *aParent* = `0`) `[explicit]`

BroadcastReceiver Constructor initializes all attributes of class.

Parameters

aParent	Pointer to QObject parent class

4.2.3 Member Function Documentation

4.2.3.1 void BroadcastReceiver::error (QUdpSocket::SocketError *socketerror*) `[signal]`

error signal on socket errors

Parameters

socketerror	exception object

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

- Connection/BroadcastReceiver.h
- Connection/BroadcastReceiver.cpp

4.3 CalcSensorData Class Reference

The CalcSensorData class Store all calculate values in a single class.

`#include <calcsensordata.h>`

Public Member Functions

- CalcSensorData (const QString aDescription, const double aValue, const QString aUnit)

  CalcSensorData Constructor to init all attributes.
- QString getUnitOfMeasurement () const

  getUnitOfMeasurement GETTER-Method to get the current unit of measurement
- QString getDescription () const

  getDescription GETTER-Method to get the current description
- double getCalcValue () const

  getCalcValue GETTER-Method to get the current calculate value
- void setUnitOfMeasurement (const QString aUnit)

  setUnitOfMeasurement SETTER-Method to set the current unit of measurement
- void setDescription (const QString aDescription)

  setDescription SETTER-Method to set the current description
- void setCalcValue (const double aValue)

  setCalcValue SETTER-Method to set the current value

4.3.1 Detailed Description

The CalcSensorData class Store all calculate values in a single class.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 CalcSensorData::CalcSensorData (const QString *aDescription*, const double *aValue*, const QString *aUnit*)

CalcSensorData Constructor to init all attributes.

Parameters

aDescription	Description of calculate data
aValue	Value from the calculate data
aUnit	Value from current unit

4.3.3 Member Function Documentation

4.3.3.1 double CalcSensorData::getCalcValue ( ) const

getCalcValue GETTER-Method to get the current calculate value

Returns

Current value

4.3.3.2 QString CalcSensorData::getDescription ( ) const

getDescription GETTER-Method to get the current description

Returns

Current description

4.3.3.3 QString CalcSensorData::getUnitOfMeasurement ( ) const

getUnitOfMeasurement GETTER-Method to get the current unit of measurement

Returns

Current unit of measurement

4.3.3.4 void CalcSensorData::setCalcValue (const double *aValue*)

setCalcValue SETTER-Method to set the current value

Parameters

aValue	Double Type with the new value

4.3.3.5 void CalcSensorData::setDescription (const QString *aDescription*)

setDescription SETTER-Method to set the current description

Parameters

aDescription	QString with the new description

4.3.3.6 void CalcSensorData::setUnitOfMeasurement (const QString *aUnit*)

setUnitOfMeasurement SETTER-Method to set the current unit of measurement

Parameters

aUnit	QString with new unit of measurement

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

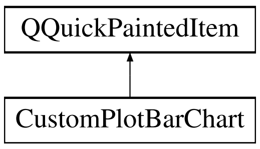
- Model/Data/calcsensordata.h
- Model/Data/calcsensordata.cpp

4.4 CustomPlotBarChart Class Reference

The CustomPlotBarChart class Paint a bar chart on view. This class in include in qml code.

`#include <customplotbarchart.h>`

Inheritance diagram for CustomPlotBarChart:



Public Member Functions

- void setData (SensorModel ∗)

  setData SETTER-Method to set the data
- SensorModel ∗ getData ()

  getData GETTER-Method to get the current data
- CustomPlotBarChart (QQuickItem ∗aParent=0)

  CustomPlotBarChart Constructor to init attributes and parent class.
- CustomPlotBarChart (const CustomPlotBarChart &aOther)=delete

  CustomPlotBarChart Copy-Constructor is not allowed.
- CustomPlotBarChart & operator= (const CustomPlotBarChart &aRhs)=delete

  operator = Copy-Assigment Operator is not allowed
- virtual ∼CustomPlotBarChart ()

  ∼CustomPlotBarChart Destructor of class
- Q_INVOKABLE void initCustomPlot ()

  initCustomPlot Set range and size of diagram
- Q_INVOKABLE void updateDataAndGUI ()

  updateDataAndGUI delete old diagram and repaint a new diagram
- void paint (QPainter ∗aPainter)

  paint paint with the current data a diagram

Properties

- SensorModel data

4.4.1 Detailed Description

The CustomPlotBarChart class Paint a bar chart on view. This class in include in qml code.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 CustomPlotBarChart::CustomPlotBarChart (QQuickItem * *aParent* = 0)

CustomPlotBarChart Constructor to init attributes and parent class.

Parameters

aParent	Pointer to QQuickItem parent class

4.4.2.2 CustomPlotBarChart::CustomPlotBarChart (const **CustomPlotBarChart** & *aOther*) `[delete]`

CustomPlotBarChart Copy-Constructor is not allowed.

Parameters

aOther	Reference to a other CustomPlotBarChart to init Object

4.4.3 Member Function Documentation

4.4.3.1 **SensorModel** ∗ CustomPlotBarChart::getData ( )

getData GETTER-Method to get the current data

Returns

current data of diagram

4.4.3.2 void CustomPlotBarChart::initCustomPlot ( )

initCustomPlot Set range and size of diagram

This function is called from view to init diagram

4.4.3.3 **CustomPlotBarChart**& CustomPlotBarChart::operator= (const **CustomPlotBarChart** & *aRhs*) `[delete]`

operator = Copy-Assigment Operator is not allowed

Parameters

aRhs	Right side of Copy-Assigment Operator

Returns

4.4.3.4 void CustomPlotBarChart::paint (QPainter ∗ *aPainter*)

paint paint with the current data a diagram

Parameters

aPainter	QPainter object to paint diagram

This function must be implement from developer

4.4.3.5 void CustomPlotBarChart::updateDataAndGUI ( )

updateDataAndGUI delete old diagram and repaint a new diagram

This function is called from view to update diagram

4.4.4 Property Documentation

4.4.4.1 SensorModel CustomPlotBarChart::data [read], [write]

A new define qml attribute to get the current data to diagram

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

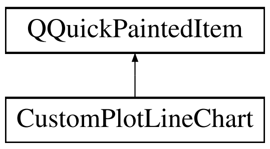
- Diagram/customplotbarchart.h
- Diagram/customplotbarchart.cpp

4.5 CustomPlotLineChart Class Reference

The CustomPlotLineChart class Paint a line chart on view. This class in include in qml code.

`#include <customplotlinechart.h>`

Inheritance diagram for CustomPlotLineChart:



Public Member Functions

- void setData (SensorModel ∗)

  setData SETTER-Method to set the data
- SensorModel ∗ getData ()

  getData GETTER-Method to get the current data
- CustomPlotLineChart (QQuickItem ∗aParent=0)

  CustomPlotLineChart Constructor to init attributes and parent class.
- CustomPlotLineChart (const CustomPlotLineChart &aOther)=delete

  CustomPlotLineChart Copy-Constructor is not allowed.
- CustomPlotLineChart & operator= (const CustomPlotLineChart &aRhs)=delete

  operator = Copy-Assigment Operator is not allowed
- virtual ∼CustomPlotLineChart ()

  ∼CustomPlotLineChart Destructor of class
- void paint (QPainter ∗aPainter)

  paint paint with the current data a diagram
- Q_INVOKABLE void initCustomPlot ()

  initCustomPlot Set range and size of diagram
- Q_INVOKABLE void updateDataAndGUI ()

  updateDataAndGUI delete old diagram and repaint a new diagram

Properties

- SensorModel data

4.5.1 Detailed Description

The CustomPlotLineChart class Paint a line chart on view. This class in include in qml code.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 CustomPlotLineChart::CustomPlotLineChart (QQuickItem ∗ *aParent* = 0)

CustomPlotLineChart Constructor to init attributes and parent class.

Parameters

aParent	Pointer to QQuickItem parent class

4.5.2.2 CustomPlotLineChart::CustomPlotLineChart (const **CustomPlotLineChart** & *aOther*) [delete]

CustomPlotLineChart Copy-Constructor is not allowed.

Parameters

aOther	Reference to a other CustomPlotLineChart to init Object

4.5.3 Member Function Documentation

4.5.3.1 **SensorModel** ∗ CustomPlotLineChart::getData ( )

getData GETTER-Method to get the current data

Returns

current data of diagram

4.5.3.2 void CustomPlotLineChart::initCustomPlot ( )

initCustomPlot Set range and size of diagram

This function is called from view to init diagram

4.5.3.3 **CustomPlotLineChart**& CustomPlotLineChart::operator= (const **CustomPlotLineChart** & *aRhs*) [delete]

operator = Copy-Assigment Operator is not allowed

Parameters

aRhs	Right side of Copy-Assigment Operator

Returns

4.5.3.4 void CustomPlotLineChart::paint (QPainter ∗ *aPainter*)

paint paint with the current data a diagram

Parameters

aPainter	QPainter object to paint diagram

This function must be implement from developer

4.5.3.5 void CustomPlotLineChart::updateDataAndGUI ( )

updateDataAndGUI delete old diagram and repaint a new diagram

This function is called from view to update diagram

4.5.4 Property Documentation

4.5.4.1 SensorModel CustomPlotLineChart::data `[read], [write]`

A new define qml attribute to get the current data to diagram

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

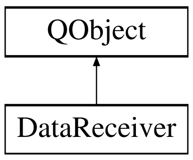
- Diagram/customplotlinechart.h
- Diagram/customplotlinechart.cpp

4.6 DataReceiver Class Reference

Implements the parser for incoming measurements.

`#include <DataReceiver.h>`

Inheritance diagram for DataReceiver:



Signals

- void updateStorage (QList< rawData > &, quint8, quint8, quint16)

  updateStorage
- void signalizeController (quint8)

  signalizeController

Public Member Functions

- bool validateData (const quint8 ∗buffer, const quint64 aLen)

  validateData triggers the parsing mechanism

Static Public Member Functions

- static DataReceiver & getInstance ()

  getInstance Creation of single instance

4.6.1 Detailed Description

Implements the parser for incoming measurements.

Is implemented as singleton class and used for parsing the received byte-vector. On successful parsing, the list of parsed measurement information is passed to ImportExport-class

4.6.2 Member Function Documentation

4.6.2.1 DataReceiver & DataReceiver::getInstance ( ) [static]

getInstance Creation of single instance

Returns

reference to single instance

4.6.2.2 void DataReceiver::signalizeController (quint8) [signal]

signalizeController

Is signalizing to the SelectionController that new data is available via ImportExport

4.6.2.3 void DataReceiver::updateStorage (QList< rawData > & , quint8 , quint8 , quint16) [signal]

updateStorage

Is passing the parsed data to ImportExport class

4.6.2.4 bool DataReceiver::validateData (const quint8 ∗ *buffer,* const quint64 *aLen*)

validateData triggers the parsing mechanism

Parameters

buffer	byte-vector received by TcpConnection
aLen	length of byte-vector

Returns

true on valid parsing, false on any error

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

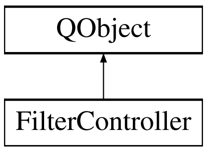
- Connection/DataReceiver.h
- Connection/DataReceiver.cpp

4.7 FilterController Class Reference

The FilterController class Manage all filter interaction from view.

`#include <filtercontroller.h>`

Inheritance diagram for FilterController:



Public Slots

- void validateUserInputSlot ()

  validateUserInputSlot Slot to validate given user input
- void newDataFromDeviceSlot (quint8 aType)

  newDataFromDeviceSlot Slot to update view if new data from device is available

Public Member Functions

- FilterController (QObject ∗aParent, SensorModel &aModel, InactiveSensorCalcModel &aCalcModel, Import↩ Export &aStorage)

  FilterController Constructor Init all attributes on class.
- FilterController (const FilterController &aOther)=delete

  FilterController Copy-Constructor is not allowed.
- FilterController & operator= (const FilterController &aRhs)=delete

  operator = Copy-Assigment Operator is not allowed

4.7.1 Detailed Description

The FilterController class Manage all filter interaction from view.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 FilterController::FilterController (QObject ∗ *aParent,* SensorModel & *aModel,* InactiveSensorCalcModel & *aCalcModel,* ImportExport & *aStorage*)

FilterController Constructor Init all attributes on class.

Parameters

aParent	Pointer to QObject parent class
aModel	Reference to a SensorModel
aCalcModel	Reference to a InactiveSensorCalcModel
aStorage	Reference to database with data to update model and view

4.7.2.2 FilterController::FilterController (const FilterController & *aOther*) [delete]

FilterController Copy-Constructor is not allowed.

Parameters

aOther	Reference to a other FilterController to init Object

4.7.3 Member Function Documentation

4.7.3.1 void FilterController::newDataFromDeviceSlot (quint8 *aType*) `[slot]`

newDataFromDeviceSlot Slot to update view if new data from device is available

Parameters

aType	Type of measurement values

4.7.3.2 FilterController& FilterController::operator= (const FilterController & *aRhs*) [delete]

operator = Copy-Assigment Operator is not allowed

Parameters

aRhs	Right side of Copy-Assigment Operator

Returns

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

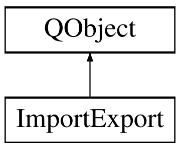
- Controller/filtercontroller.h
- Controller/filtercontroller.cpp

4.8 ImportExport Class Reference

The ImportExport class.

```
#include <ImportExport.h>
```

Inheritance diagram for ImportExport:



Public Slots

- void insertMeasurement (QList< rawData > &dataList, quint8 type, quint8 mood, quint16 average)
  insertMeasurement slot called by DataReceiver

Public Member Functions

- ImportExport (QObject ∗parent=0)
  ImportExport constructor is initializing the database connection.
- virtual ∼ImportExport ()
  ∼ImportExport destructor is safely closing database connection
- QList< QString > years (quint8 aType)
  years looks for all years for the passed measurement-type
- QList< QString > months (quint8 aType, const QDate &year)
  months looks for all available months containing data
- QList< const SensorData ∗ > dataByMeasurementId (quint64)
  dataByMeasurementId get a list of detailed data-points by id
- QList< const SensorData ∗ > measurements (quint8 aType)

measurements get a list of measurements by type
- QList< const SensorData ∗ > measurementsFromTo (quint8 aType, const QDate &aStart, const QDate &s↩ End)

  measurementsFromTo get a list of measurements by start and end date
- operator bool () const

  operator bool get information about the state of this object

4.8.1 Detailed Description

The ImportExport class.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 ImportExport::ImportExport (QObject ∗ *parent* = 0) [explicit]

ImportExport constructor is initializing the database connection.

Parameters

parent	QObject of parent class

4.8.3 Member Function Documentation

4.8.3.1 QList< const SensorData ∗ > ImportExport::dataByMeasurementId (quint64 *aId*)

dataByMeasurementId get a list of detailed data-points by id

Parameters

aId	the id of the measurement

Returns

List of SendorData objects, each containing a data-point of the desired measurement

4.8.3.2 void ImportExport::insertMeasurement (QList< rawData > & *dataList*, quint8 *type*, quint8 *mood*, quint16 *average*) [slot]

insertMeasurement slot called by DataReceiver

Parameters

dataList	contains raw data-points for database insertion
type	of the measurement
mood	of the measurement
average	heartrate of measurement

4.8.3.3 QList< const SensorData ∗ > ImportExport::measurements (quint8 *aType*)

measurements get a list of measurements by type

Parameters

aType	the type of measurement

Returns

List of SensorData with general information about the found measurements

4.8.3.4 QList< const SensorData ∗ > ImportExport::measurementsFromTo (quint8 *aType,* const QDate & *aStart,* const QDate & *sEnd*)

measurementsFromTo get a list of measurements by start and end date

Parameters

aType	the type of measurement
aStart	start date of the measurements (inclusive)
sEnd	end date of the measurements (inclusive)

Returns

List of SensorData with general information about the found measurements

4.8.3.5 QList< QString > ImportExport::months (quint8 *aType,* const QDate & *year*)

months looks for all available months containing data

Parameters

aType	the type of measurement
year	the year to search in

Returns

List of string with months strings (english)

4.8.3.6 ImportExport::operator bool ( ) const [explicit]

operator bool get information about the state of this object

Returns

true when object initialization and database setup was successful, false otherwise

4.8.3.7 QList< QString > ImportExport::years (quint8 *aType*)

years looks for all years for the passed measurement-type

Parameters

aType	the type of measurement

Returns

List of strings with year numbers in the form of YYYY

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

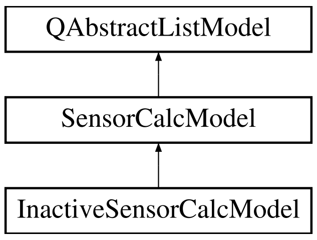
- ImportExport/ImportExport.h
- ImportExport/ImportExport.cpp

4.9 InactiveSensorCalcModel Class Reference

The InactiveSensorCalcModel class This class represents a model for inactive calc values.

`#include <inactivesensorcalcmodel.h>`

Inheritance diagram for InactiveSensorCalcModel:



Public Types

- enum SensorCalcRoles { INACTIVE_SENSOR_CALC_VALUE_ROLE = 0, INACTIVE_SENSOR_CALC_↩ DESCRIPTION_ROLE }

  The SensorCalcRoles enum Inlcude all roles from model.

Public Member Functions

- InactiveSensorCalcModel (SensorModel &aModel)

  InactiveSensorCalcModel Constructor to init all attributes.
- QVariant data (const QModelIndex &aIndex, int aRole=Qt::DisplayRole) const

  data Return to a index and role a QVariant value for view
- int rowCount (const QModelIndex &aParent=QModelIndex()) const

  rowCount Actual count of rows in model

Protected Member Functions

- QHash< int, QByteArray > roleNames () const

  roleNames Connect Roles on view with roles in model

Additional Inherited Members

4.9.1 Detailed Description

The InactiveSensorCalcModel class This class represents a model for inactive calc values.

4.9.2 Member Enumeration Documentation

4.9.2.1 enum **InactiveSensorCalcModel::SensorCalcRoles**

The SensorCalcRoles enum Inlcude all roles from model.

Enumerator

INACTIVE_SENSOR_CALC_VALUE_ROLE	Role for a single inactive calc value
INACTIVE_SENSOR_CALC_DESCRIPTION_ROLE	Role for a single inactive calc description

4.9.3 Constructor & Destructor Documentation

4.9.3.1 InactiveSensorCalcModel::InactiveSensorCalcModel (**SensorModel** & *aModel*)

InactiveSensorCalcModel Constructor to init all attributes.

Parameters

aModel	Reference to a SensorModel to calculate data

4.9.4 Member Function Documentation

4.9.4.1 QVariant InactiveSensorCalcModel::data (const QModelIndex & *aIndex,* int *aRole* = `Qt::DisplayRole`) const

data Return to a index and role a QVariant value for view

Parameters

aIndex	Index of model
aRole	Current Role

Returns

QVariant value with value and role

This function is used by model/view on QT

4.9.4.2 QHash< int, QByteArray > InactiveSensorCalcModel::roleNames ( ) const `[protected]`

roleNames Connect Roles on view with roles in model

Returns

QHash with the connected roles

This function is used by model/view on QT

4.9.4.3 int InactiveSensorCalcModel::rowCount (const QModelIndex & *aParent* = `QModelIndex()`) const

rowCount Actual count of rows in model

Parameters

aParent	-

Returns

Count of rows in model

This function is used by model/view on QT

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

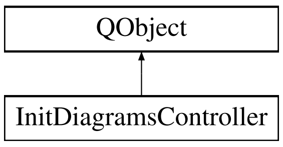
- Model/inactivesensorcalcmodel.h
- Model/inactivesensorcalcmodel.cpp

4.10 InitDiagramsController Class Reference

The InitDiagramsController class Init diagrams on view with actual data from database.

```
#include <initdiagramscontroller.h>
```

Inheritance diagram for InitDiagramsController:



Public Member Functions

- InitDiagramsController (QObject ∗aParent, SensorModel &aInactiveModel, SensorModel &aActiveModel)

  InitDiagramsController Constructor to init attributes.
- InitDiagramsController (const InitDiagramsController &aOther)=delete

  InitDiagramsController Copy-Constructor is not allowed.
- InitDiagramsController & operator= (const InitDiagramsController &aRhs)

  operator = Copy-Assigment Operator is not allowed

4.10.1 Detailed Description

The InitDiagramsController class Init diagrams on view with actual data from database.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 InitDiagramsController::InitDiagramsController (QObject ∗ *aParent*, SensorModel & *aInactiveModel*, SensorModel & *aActiveModel*)

InitDiagramsController Constructor to init attributes.

Parameters

aParent	Pointer to QObject parent class

aInactiveModel	Reference to a SensorModel with inactive sensor data
aActiveModel	Reference to a SensorModel with active sensor data

4.10.2.2 InitDiagramsController::InitDiagramsController (const **InitDiagramsController** & *aOther*) `[delete]`

InitDiagramsController Copy-Constructor is not allowed.

Parameters

aOther	Reference to a other InitDiagramsController to init Object

4.10.3 Member Function Documentation

4.10.3.1 **InitDiagramsController&** InitDiagramsController::operator= (const **InitDiagramsController** & *aRhs*)

operator = Copy-Assigment Operator is not allowed

Parameters

aRhs	Right side of Copy-Assigment Operator

Returns

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

- Controller/initdiagramscontroller.h
- Controller/initdiagramscontroller.cpp

4.11 MeasureType Class Reference

The MeasureType class is a container for static const strings of measurement types.

`#include <MeasureType.h>`

Static Public Attributes

- static const quint8 numOfTypes = 2

  numOfTypes number of available types
- static const char ∗ typeName []

  typeName static const string of earch type name

4.11.1 Detailed Description

The MeasureType class is a container for static const strings of measurement types.

4.11.2 Member Data Documentation

4.11.2.1 const char ∗ MeasureType::typeName `[static]`

Initial value:

```
= {
    "activity",
    "rest"
}
```

typeName static const string of earch type name

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

- ImportExport/MeasureType.h
- ImportExport/MeasureType.cpp

4.12 MoodType Class Reference

The MoodType class is a container for static const strings of mood types.

`#include <MoodType.h>`

Static Public Attributes

- static const quint8 numOfTypes = 3

  numOfTypes number of available mood types
- static const char ∗ typeName []

  typeName static const string of earch type name

4.12.1 Detailed Description

The MoodType class is a container for static const strings of mood types.

4.12.2 Member Data Documentation

4.12.2.1 const char ∗ MoodType::typeName `[static]`

Initial value:

```
= {
    "good",
    "average",
    "bad"
}
```

typeName static const string of earch type name

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

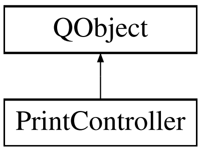
- ImportExport/MoodType.h
- ImportExport/MoodType.cpp

4.13 PrintController Class Reference

The PrintController class Print all selected inactive and active sensor data.

`#include <printcontroller.h>`

Inheritance diagram for PrintController:



Public Slots

- void clickPrintButtonSlot ()

  clickPrintButtonSlot Slot to print all data

Public Member Functions

- PrintController (QObject ∗aParent, SensorModel &aModelForInactiveData, SensorModel &aModelFor↩ ActiveData)

  PrintController Constructor to init all attributes.
- PrintController (const PrintController &aOther)=delete

  PrintButtonController Copy-Constructor is not allowed.
- PrintController & operator= (const PrintController &aRhs)=delete

  operator = Copy-Assigment Operator is not allowed

4.13.1 Detailed Description

The PrintController class Print all selected inactive and active sensor data.

This class use a QTextDocument to format print output. The current format is given through html code

4.13.2 Constructor & Destructor Documentation

4.13.2.1 PrintController::PrintController (QObject ∗ *aParent*, **SensorModel** & *aModelForInactiveData*, **SensorModel** & *aModelForActiveData*)

PrintController Constructor to init all attributes.

Parameters

aParent	Pointer to qObject parent class
aModelFor↩ InactiveData	Reference to SensorModel with inactvie sensor data
aModelFor↩ ActiveData	Reference to SensorModel with active sensor data

4.13.2.2 PrintController::PrintController (const **PrintController** & *aOther*) [delete]

PrintButtonController Copy-Constructor is not allowed.

Parameters

aOther	Reference to a other PrintButtonController to init Object

4.13.3 Member Function Documentation

4.13.3.1 PrintController& PrintController::operator= (const PrintController & *aRhs*) `[delete]`

operator = Copy-Assigment Operator is not allowed

Parameters

aRhs	Right side of Copy-Assigment Operator

Returns

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

- Controller/printcontroller.h
- Controller/printcontroller.cpp

4.14 rawData Struct Reference

simple data structure for received measurement datapoints

```
#include <DataReceiver.h>
```

Public Attributes

- quint64 **timeStamp**
- quint16 **heartRate**
- quint16 **steps**

4.14.1 Detailed Description

simple data structure for received measurement datapoints

is used for building a list containing all measurement values for a measurement. this list is used by ImportExport class

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

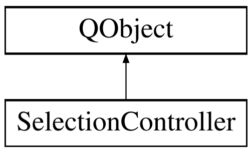
- Connection/DataReceiver.h

4.15 SelectionController Class Reference

The SelectionController class This calss manage the interaction with the comboboxes on "inactive" Tab.

```
#include <selectioncontroller.h>
```

Inheritance diagram for SelectionController:



Public Slots

- void selectYearSlot (QString aCurrentText)

  selectYearSlot Slot to get current year from view

- void selectMonthSlot (QString aCurrentText)

  selectMonthSlot Slot to get current month from view
- void newDataFromDeviceSlot (quint8 aType)

  newDataFromDeviceSlot Slot to update view if new data from device is available

Public Member Functions

- SelectionController (QObject ∗aParent, SelectionModel &aYearModel, SelectionModel &aMonthModel, SensorModel &aInactiveModel, SensorModel &aRunModel, ActiveSensorCalcModel &aCalcModel, Import←Export &aStorage)

  SelectionController Constructor to init all attributes.
- SelectionController (const SelectionController &aOther)=delete

  SelectionController Copy-Constructor is not allowed.
- SelectionController & operator= (const SelectionController &aRhs)=delete

  operator = Copy-Assigment Operator is not allowed

4.15.1 Detailed Description

The SelectionController class This calss manage the interaction with the comboboxes on "inactive" Tab.

4.15.2 Constructor & Destructor Documentation

4.15.2.1 SelectionController::SelectionController (QObject ∗ *aParent*, **SelectionModel** & *aYearModel*, **SelectionModel** & *aMonthModel*, **SensorModel** & *aInactiveModel*, **SensorModel** & *aRunModel*, **ActiveSensorCalcModel** & *aCalcModel*, **ImportExport** & *aStorage*)

SelectionController Constructor to init all attributes.

Parameters

aParent	Pointer to QObject parent calss
aYearModel	Reference to SelectionModel with data for year combobox
aMonthModel	Reference to SelectionModel with data for month combobox
aInactiveModel	Reference to SensorModel with all inactive sensor data
aRunModel	Reference to SensorModel with a overview of all actvie sensor data
aCalcModel	Reference to SensorCalcModel with all calculate actvie sensor data
aStorage	Reference with all querys to get data from database

4.15.2.2 SelectionController::SelectionController (const **SelectionController** & *aOther*) [delete]

SelectionController Copy-Constructor is not allowed.

Parameters

aOther	Reference to a other SelectionController to init Object

4.15.3 Member Function Documentation

4.15.3.1 void SelectionController::newDataFromDeviceSlot (quint8 *aType*) [slot]

newDataFromDeviceSlot Slot to update view if new data from device is available

Parameters

aType	Type of measurement values

4.15.3.2 SelectionController& SelectionController::operator= (const SelectionController & *aRhs*) [delete]

operator = Copy-Assigment Operator is not allowed

Parameters

aRhs	Right side of Copy-Assigment Operator

Returns

4.15.3.3 void SelectionController::selectMonthSlot (QString *aCurrentText*) [slot]

selectMonthSlot Slot to get current month from view

Parameters

aIndex	Selected text from month combobox

4.15.3.4 void SelectionController::selectYearSlot (QString *aCurrentText*) [slot]

selectYearSlot Slot to get current year from view

Parameters

aCurrentText	Selected text from year combobox

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

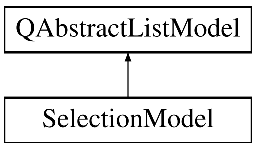
- Controller/selectioncontroller.h
- Controller/selectioncontroller.cpp

4.16 SelectionModel Class Reference

The SelectionModel class This class represents a model for comboboxes.

`#include <selectionmodel.h>`

Inheritance diagram for SelectionModel:



Public Types

- enum SelectionRoles { SELECTION_VALUE_ROLE }

  The SelectionRoles enum Inlcude all roles from model.

Public Member Functions

- SelectionModel (QObject ∗aParent=0)

  SelectionModel Constructor to init all attributes.
- QVariant data (const QModelIndex &aIndex, int aRole=Qt::DisplayRole) const

  data Return to a index and role a QVariant value for view
- int rowCount (const QModelIndex &aParent=QModelIndex()) const

  rowCount Actual count of rows in model
- void setNewSelectionModel (QList< QString > &aSelectionModel)

  setNewSelection Model Add a new list of data objects to model

Protected Member Functions

- QHash< int, QByteArray > roleNames () const

  roleNames Connect Roles on view with roles in model

4.16.1 Detailed Description

The SelectionModel class This class represents a model for comboboxes.

All values from comboboxes are store in a data list from this class.

4.16.2 Member Enumeration Documentation

4.16.2.1 enum **SelectionModel::SelectionRoles**

The SelectionRoles enum Inlcude all roles from model.

Enumerator

SELECTION_VALUE_ROLE	Role for a single value in combobox

4.16.3 Constructor & Destructor Documentation

4.16.3.1 SelectionModel::SelectionModel (QObject ∗ *aParent* = `0`) `[explicit]`

SelectionModel Constructor to init all attributes.

Parameters

aParent	Pointer to QAbstractListModel parent class

4.16.4 Member Function Documentation

4.16.4.1 QVariant SelectionModel::data (const QModelIndex & *aIndex*, int *aRole* = `Qt::DisplayRole`) const

data Return to a index and role a QVariant value for view

Parameters

aIndex	Index of model

aRole	Current Role

Returns

QVariant value with value and role

This function is used by model/view on QT

4.16.4.2 QHash< int, QByteArray > SelectionModel::roleNames ( ) const [protected]

roleNames Connect Roles on view with roles in model

Returns

QHash with the connected roles

This function is used by model/view on QT

4.16.4.3 int SelectionModel::rowCount (const QModelIndex & *aParent* = `QModelIndex()`) const

rowCount Actual count of rows in model

Parameters

aParent	-

Returns

Count of rows in model

This function is used by model/view on QT

4.16.4.4 void SelectionModel::setNewSelectionModel (QList< QString > & *aSelectionModel*)

setNewSelection Model Add a new list of data objects to model

Parameters

aSensorModel	List with new data objects

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

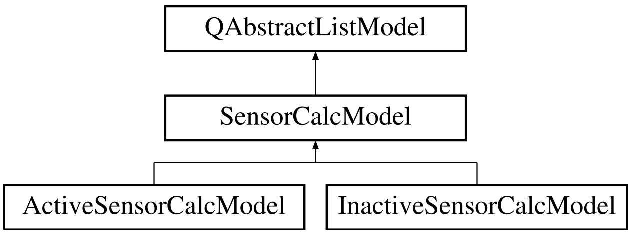
- Model/selectionmodel.h
- Model/selectionmodel.cpp

4.17 SensorCalcModel Class Reference

The SensorCalcModel class This class represents a model for calculate data to show on view.

`#include <sensorcalcmodel.h>`

Inheritance diagram for SensorCalcModel:



Public Member Functions

- SensorCalcModel (SensorModel &aModel, QObject ∗aParent=0)

    SensorCalcModel Constructor to init all attributes on class.
- void setNewSensorCalcModel (const QList< CalcSensorData > &aSensorModel)

    setNewSensorCalcModel Add a new sensor data list with calculate values to model
- void updateCalcValues (const SensorModel &aModel)

    updateCalcValues Update all calc values when data from model change

Protected Attributes

- QList< CalcSensorData > m_calcSensorList

    m_calcSensorList QList with sensor data objects (Data from Model)
- SensorModel & m_Model

    m_Model Reference from SensorModel to calculate data

4.17.1 Detailed Description

The SensorCalcModel class This class represents a model for calculate data to show on view.

This is the parent class of InactiveSensorCalcModel and ActiveSensorCalcModel. Duplicated code from both sub classes is abstract to this class

4.17.2 Constructor & Destructor Documentation

4.17.2.1 SensorCalcModel::SensorCalcModel (SensorModel & *aModel*, QObject ∗ *aParent* = 0)

SensorCalcModel Constructor to init all attributes on class.

Parameters

aModel	Reference to a SensorModel with inactvie or active sensor data
aParent	Pointer to QAbstractLisModel parent class

4.17.3 Member Function Documentation

4.17.3.1 void SensorCalcModel::setNewSensorCalcModel (const QList< CalcSensorData > & *aSensorModel*)

setNewSensorCalcModel Add a new sensor data list with calculate values to model

Parameters

aSensorModel	

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

- Model/sensorcalcmodel.h
- Model/sensorcalcmodel.cpp

4.18 SensorData Class Reference

The SensorData class Store all sensor values from a single measurement.

`#include <sensordata.h>`

Public Member Functions

- const QDateTime & getDate () const

  getDate GETTER-METHOD to get current time
- quint16 getHeartRate () const

  getDate GETTER-METHOD to get current heart rate
- quint64 getStepCount () const

  getStepCount GETTER-Method to get current step count
- void setDate (const QDateTime &aDate)

  setDate SETTER-Method to set new time
- void setHeartRate (quint16 aHeartRate)

  setHeartRate SETTER-Method to set new heart rate
- void setStepCount (quint16 aStepCount)

  setStepCount SETTER-Method to set new step count
- quint64 getId () const

  getId GETTER-Method to get the id of a single measurement
- SensorData (const QDateTime &aDate, quint16 aHeartRate, quint64 aStepLength, quint64 aId)

  SensorData Constructor to init all attributes.

4.18.1 Detailed Description

The SensorData class Store all sensor values from a single measurement.

4.18.2 Constructor & Destructor Documentation

4.18.2.1 SensorData::SensorData (const QDateTime & *aDate,* quint16 *aHeartRate,* quint64 *aStepLength,* quint64 *aId*)

SensorData Constructor to init all attributes.

Parameters

aDate	New time value
aHeartRate	New heart rate value
aStepLength	New step length value
aId	New id value

4.18.3 Member Function Documentation

4.18.3.1 const QDateTime & SensorData::getDate ( ) const

getDate GETTER-METHOD to get current time

Returns

Current time

4.18.3.2 quint16 SensorData::getHeartRate ( ) const

getDate GETTER-METHOD to get current heart rate

Returns

Current heart rate

4.18.3.3 quint64 SensorData::getId ( ) const

getId GETTER-Method to get the id of a single measurement

Returns

Current ID of a single measurement

4.18.3.4 quint64 SensorData::getStepCount ( ) const

getStepCount GETTER-Method to get current step count

Returns

Current step count

4.18.3.5 void SensorData::setDate (const QDateTime & *aDate*)

setDate SETTER-Method to set new time

Parameters

aDate	QDateTime with new time value

4.18.3.6 void SensorData::setHeartRate (quint16 *aHeartRate*)

setHeartRate SETTER-Method to set new heart rate

Parameters

aHeartRate	New heart rate value

4.18.3.7 void SensorData::setStepCount (quint16 *aStepCount*)

setStepCount SETTER-Method to set new step count

Parameters

aStepCount	New step count value

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

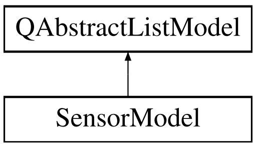
- Model/Data/sensordata.h
- Model/Data/sensordata.cpp

4.19 SensorModel Class Reference

The SensorModel class This class represents a model for inactvie and active sensor data.

```
#include <sensormodel.h>
```

Inheritance diagram for SensorModel:



Public Types

- enum SensorRoles {
  SENSOR_MEASUREPOINT, SENSOR_DATE_ROLE, SENSOR_TIME_ROLE, SENSOR_HEART_RATE↩
  _ROLE,
  SENSOR_STEPS_ROLE, SENSOR_DURATION_ROLE }

  The SensorRoles enum.

Public Member Functions

- ∼SensorModel ()

  ∼SensorModel Destructor is needed to declare a new QVariant type
- SensorModel ()

  SensorModel Constructor is needed to declare a new QVariant type.
- SensorModel (const SensorModel &aOther)

  SensorModel Copy-Constructor is needed to declare a new QVariant type.
- QList< const SensorData ∗ > getDataList () const

  getDataList GETTER-Method for data list
- void addSensorData (const SensorData ∗aSensorData)

  addSensorData Add a new data object to model
- void setNewSensorModel (QList< const SensorData ∗ > &aSensorModel)

  setNewSensorModel Add a new list of data objects to model
- int getSensorModelCount () const

  getSensorModelCount GETTER-Method for the size of the data list
- const SensorData ∗ getSingleSensorData (const int aIndex) const

  getSingleSensorData GETTER-Method for a single data object
- QVariant data (const QModelIndex &aIndex, int aRole=Qt::DisplayRole) const

  data Return to a index and role a QVariant value for view
- int rowCount (const QModelIndex &aParent=QModelIndex()) const

  rowCount Actual count of rows in model

Protected Member Functions

- QHash< int, QByteArray > roleNames () const

  roleNames Connect Roles on view with roles in model

Friends

- class SensorCalcModel

4.19.1 Detailed Description

The SensorModel class This class represents a model for inactvie and active sensor data.

The model is needed to update automatically view when the model data get change. This model is used in diagrams

4.19.2 Member Enumeration Documentation

4.19.2.1 enum SensorModel::SensorRoles

The SensorRoles enum.

This enum store all Roles wich are needed for communcation with view

Enumerator

SENSOR_MEASUREPOINT	Role for seconds since start
SENSOR_DATE_ROLE	Role for date stamp
SENSOR_TIME_ROLE	Role for time stamp
SENSOR_HEART_RATE_ROLE	Role for heart rate
SENSOR_STEPS_ROLE	Role for step count
SENSOR_DURATION_ROLE	Role for duration

4.19.3 Constructor & Destructor Documentation

4.19.3.1 SensorModel::SensorModel (const SensorModel & *aOther*)

SensorModel Copy-Constructor is needed to declare a new QVariant type.

Parameters

aOther	

4.19.4 Member Function Documentation

4.19.4.1 void SensorModel::addSensorData (const SensorData ∗ *aSensorData*)

addSensorData Add a new data object to model

Parameters

aSensorData	New data object

4.19.4.2 QVariant SensorModel::data (const QModelIndex & *aIndex*, int *aRole* = `Qt::DisplayRole`) const

data Return to a index and role a QVariant value for view

Parameters

aIndex	Index of model
aRole	Current Role

Returns

QVariant value with value and role

This function is used by model/view on QT

4.19.4.3 QList< const SensorData ∗ > SensorModel::getDataList ( ) const

getDataList GETTER-Method for data list

Returns

4.19.4.4 int SensorModel::getSensorModelCount ( ) const

getSensorModelCount GETTER-Method for the size of the data list

Returns

Size of data list

4.19.4.5 const SensorData ∗ SensorModel::getSingleSensorData (const int *aIndex*) const

getSingleSensorData GETTER-Method for a single data object

Parameters

aIndex	Index in model to find the data object

Returns

Selected data object

4.19.4.6 QHash< int, QByteArray > SensorModel::roleNames ( ) const `[protected]`

roleNames Connect Roles on view with roles in model

Returns

QHash with the connected roles

This function is used by model/view on QT

4.19.4.7 int SensorModel::rowCount (const QModelIndex & *aParent* = `QModelIndex()`) const

rowCount Actual count of rows in model

Parameters

aParent	-

Returns

Count of rows in model

This function is used by model/view on QT

4.19.4.8 void SensorModel::setNewSensorModel (QList< const SensorData * > & *aSensorModel*)

setNewSensorModel Add a new list of data objects to model

Parameters

aSensorModel	List with new data objects

4.19.5 Friends And Related Function Documentation

4.19.5.1 friend class SensorCalcModel `[friend]`

fried declaration to allow SensorCalcModel to acces on attributes in the class

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

- Model/sensormodel.h
- Model/sensormodel.cpp

4.20 Settings Class Reference

Singleton class container for settings configuration.

`#include <Settings.h>`

Public Member Functions

- operator bool () const

  operator bool get information about the state of this object

Static Public Member Functions

- static Settings & getInstance ()

  getInstance creates single instance

Public Attributes

- const QString mDateFormat

  mDataFormat representing the operating system defined DateFormat
- const QString mTimeFormat

  mTimeFormat representing the operating system defined TimeFormat
- QString mDataDirectory

  mDataDirectory string of the HeartRate2Go user directory

4.20.1 Detailed Description

Singleton class container for settings configuration.

4.20.2 Member Function Documentation

4.20.2.1 Settings & Settings::getInstance ( ) `[static]`

getInstance creates single instance

Returns

object reference to the single instance

4.20.2.2 Settings::operator bool ( ) const `[explicit]`

operator bool get information about the state of this object

Returns

true when object initialization was successful, false otherwise

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

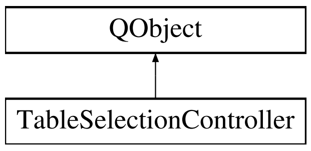
- Settings/Settings.h
- Settings/Settings.cpp

4.21 TableSelectionController Class Reference

The TableSelectionController class Manange interaction with overview table to show current run on diagram.

`#include <tableselectioncontroller.h>`

Inheritance diagram for TableSelectionController:



Public Slots

- void selectSingleRunSlot (int aIndex)

  selectSingleRunSlot Slot to get all data from a selected run

Public Member Functions

- TableSelectionController (QObject ∗aParent, SensorModel &aRunModel, SensorModel &aActiveSensor↩ Model, ActiveSensorCalcModel &aActiveSensorCalcModel, ImportExport &aStorage)

  TableSelectionController Constructor to init all attributes.
- TableSelectionController (const TableSelectionController &aOther)=delete

  TableSelectionController Copy-Constructor is not allowed.
- TableSelectionController & operator= (const TableSelectionController &aRhs)=delete

  operator = Copy-Assigment Operator is not allowed

4.21.1 Detailed Description

The TableSelectionController class Manange interaction with overview table to show current run on diagram.

4.21.2 Constructor & Destructor Documentation

4.21.2.1 TableSelectionController::TableSelectionController (QObject ∗ *aParent,* SensorModel & *aRunModel,* SensorModel & *aActiveSensorModel,* ActiveSensorCalcModel & *aActiveSensorCalcModel,* ImportExport & *aStorage*)

TableSelectionController Constructor to init all attributes.

Parameters

aParent	Pointer to QObject parent class
aRunModel	Reference to SensorModel with a overview of single runs
aActiveSensor↩ Model	Reference to SensorModel with active sensor data
aActiveSensor↩ CalcModel	Reference to ActiveSensorCalcModel with calculate data
aStorage	Reference to database to get data

4.21.2.2 TableSelectionController::TableSelectionController (const TableSelectionController & *aOther*) [delete]

TableSelectionController Copy-Constructor is not allowed.

Parameters

aOther	Reference to a other TableSelectionController to init Object

4.21.3 Member Function Documentation

4.21.3.1 TableSelectionController& TableSelectionController::operator= (const TableSelectionController & *aRhs*) [delete]

operator = Copy-Assigment Operator is not allowed

Parameters

aRhs	Right side of Copy-Assigment Operator

Returns

4.21.3.2 void TableSelectionController::selectSingleRunSlot (int *aIndex*) [slot]

selectSingleRunSlot Slot to get all data from a selected run

Parameters

aIndex	Index of model

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

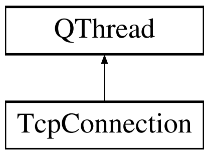
- Controller/tableselectioncontroller.h
- Controller/tableselectioncontroller.cpp

4.22 TcpConnection Class Reference

Implementation of established connection handler.

`#include <TcpConnection.h>`

Inheritance diagram for TcpConnection:



Public Slots

- void readyRead ()

  readyRead is called on incoming datagram and handles it
- void disconnected ()

  disconnected is called on TCP-connection disconnect

Signals

- void error (QTcpSocket::SocketError socketerror)

  error signal on socket errors

Public Member Functions

- TcpConnection (qintptr aSocketDescriptor, QObject ∗aParent=0)

  TcpConnection.
- void **run** ()

4.22.1 Detailed Description

Implementation of established connection handler.

4.22.2 Constructor & Destructor Documentation

4.22.2.1 TcpConnection::TcpConnection (qintptr *aSocketDescriptor,* QObject ∗ *aParent* = 0)

TcpConnection.

Parameters

aSocket↩ Descriptor	
aParent	

4.22.3 Member Function Documentation

4.22.3.1 void TcpConnection::disconnected ( ) [slot]

disconnected is called on TCP-connection disconnect

On disconnect of remote host and successful received data, this method is calling ImportExport

4.22.3.2 void TcpConnection::error (QTcpSocket::SocketError *socketerror*) `[signal]`

error signal on socket errors

Parameters

socketerror	exception object

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

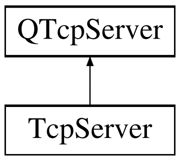
- Connection/TcpConnection.h
- Connection/TcpConnection.cpp

4.23 TcpServer Class Reference

The TcpServer class is implementing the TcpServer Threadloop.

`#include <TcpServer.h>`

Inheritance diagram for TcpServer:



Public Member Functions

- TcpServer (QObject ∗aParent=0)

  TcpServer constructor.
- virtual ∼TcpServer ()

  ∼TcpServer destructor
- void startServer ()

  startServer switches socket into listen-mode

Protected Member Functions

- virtual void incomingConnection (qintptr aSocketDescriptor)

  incomingConnection implementation of QTcpSocket pure virtual method

4.23.1 Detailed Description

The TcpServer class is implementing the TcpServer Threadloop.

4.23.2 Constructor & Destructor Documentation

4.23.2.1 TcpServer::TcpServer (QObject ∗ *aParent* = 0) `[explicit]`

TcpServer constructor.

Parameters

aParent	QObject of parent class

4.23.2.2 TcpServer::∼TcpServer ( ) [virtual]

∼TcpServer destructor

Is closing socket and marking it for release

4.23.3 Member Function Documentation

4.23.3.1 void TcpServer::incomingConnection (qintptr *aSocketDescriptor*) [protected], [virtual]

incomingConnection implementation of QTcpSocket pure virtual method

Parameters

aSocket↩ Descriptor	filedescriptor of new established connection

Is calld by QTcpSocket on new connection

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

- Connection/TcpServer.h
- Connection/TcpServer.cpp

Chapter 5

File Documentation

5.1 Connection/BroadcastReceiver.cpp File Reference

Implementation file of BroadcastReceiver class.

```
#include "BroadcastReceiver.h"
```

5.1.1 Detailed Description

Implementation file of BroadcastReceiver class.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.2 Connection/BroadcastReceiver.h File Reference

Class definition of BroadcastReceiver.

```
#include <QThread>
#include <QUdpSocket>
```

Classes

- class BroadcastReceiver

  The BroadcastReceiver class implements an UDP-Server for Server-Discovery.

5.2.1 Detailed Description

Class definition of BroadcastReceiver.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.3 Connection/DataReceiver.cpp File Reference

Implementation file of DataReceiver class.

```
#include "DataReceiver.h"
#include "Model/inactivesensorcalcmodel.h"
#include "Model/activesensorcalcmodel.h"
#include <QDebug>
#include <QList>
#include <QDateTime>
#include <QUrlQuery>
#include <QtEndian>
```

5.3.1 Detailed Description

Implementation file of DataReceiver class.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.4 Connection/DataReceiver.h File Reference

Class definition of DataReceiver.

```
#include <QString>
#include <QList>
#include <QObject>
#include "Model/Data/sensordata.h"
```

Classes

- struct rawData

  simple data structure for received measurement datapoints
- class DataReceiver

  Implements the parser for incoming measurements.

5.4.1 Detailed Description

Class definition of DataReceiver.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.5 Connection/TcpConnection.cpp File Reference

Implementation file of TcpConnection class.

```
#include "TcpConnection.h"
#include "DataReceiver.h"
```

5.5.1 Detailed Description

Implementation file of TcpConnection class.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.6 Connection/TcpConnection.h File Reference

Class definition of TcpConnection.

```
#include <QThread>
#include <QTcpSocket>
```

Classes

- class TcpConnection

  Implementation of established connection handler.

5.6.1 Detailed Description

Class definition of TcpConnection.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.7 Connection/TcpServer.cpp File Reference

Implementation file of TcpServer class.

```
#include "TcpServer.h"
#include "TcpConnection.h"
```

5.7.1 Detailed Description

Implementation file of TcpServer class.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.8 Connection/TcpServer.h File Reference

Class definition of TcpServer.

```
#include <QTcpServer>
```

Classes

- class TcpServer
  The TcpServer class is implementing the TcpServer Threadloop.

5.8.1 Detailed Description

Class definition of TcpServer.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.9 Controller/filtercontroller.cpp File Reference

Implementation file of FilterController class.

```
#include "filtercontroller.h"
```

5.9.1 Detailed Description

Implementation file of FilterController class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

21.12.2014 13:13:05 GMT

5.10 Controller/filtercontroller.h File Reference

Include all declarations from Filtercontroller.

```
#include <QObject>
#include "Model/sensormodel.h"
#include "Model/inactivesensorcalcmodel.h"
#include "ImportExport/ImportExport.h"
#include <QDate>
```

Classes

- class FilterController
  The FilterController class Manage all filter interaction from view.

5.10.1 Detailed Description

Include all declarations from Filtercontroller.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

21.12.2014 15:30:00 GMT

5.11 Controller/initdiagramscontroller.cpp File Reference

Implementation file of InitDiagramsController class.

```
#include "initdiagramscontroller.h"
```

5.11.1 Detailed Description

Implementation file of InitDiagramsController class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

20.12.2014 12:56:00 GMT

5.12 Controller/printcontroller.cpp File Reference

Implementation file of PrintController class.

```
#include "printcontroller.h"
```

5.12.1 Detailed Description

Implementation file of PrintController class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

20.12.2014 14:56:00 GMT

5.13 Controller/printcontroller.h File Reference

Inlcude all declarations from PrintController.

```
#include <QObject>
#include <QDebug>
#include <QPrinter>
#include <QPrintDialog>
#include "Model/sensormodel.h"
#include <QTextDocument>
```

Classes

- class PrintController

  The PrintController class Print all selected inactive and active sensor data.

5.13.1 Detailed Description

Inlcude all declarations from PrintController.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

20.12.2014 14:55:00 GMT

5.14 Controller/selectioncontroller.cpp File Reference

Implementation file of SelectionController class.

```
#include "selectioncontroller.h"
```

5.14.1 Detailed Description

Implementation file of SelectionController class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

20.12.2014 11:56:00 GMT

5.15 Controller/selectioncontroller.h File Reference

Include all declarations from SelectionController.

```
#include <QObject>
#include "Model/selectionmodel.h"
#include "Model/sensormodel.h"
#include "Model/activesensorcalcmodel.h"
#include "ImportExport/ImportExport.h"
```

Classes

- class SelectionController

  The SelectionController class This calss manage the interaction with the comboboxes on "inactive" Tab.

5.15.1 Detailed Description

Include all declarations from SelectionController.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

20.12.2014 11:56:00 GMT

5.16 Controller/tableselectioncontroller.cpp File Reference

Implementation file of TableSelectionController class.

```
#include "tableselectioncontroller.h"
```

5.16.1 Detailed Description

Implementation file of TableSelectionController class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

22.12.2014 10:31:00 GMT

5.17 Controller/tableselectioncontroller.h File Reference

Include all declarations from TableSelectionController.

```
#include <QObject>
#include "Model/sensormodel.h"
#include "Model/activesensorcalcmodel.h"
#include "ImportExport/ImportExport.h"
```

Classes

- class TableSelectionController

  The TableSelectionController class Manange interaction with overview table to show current run on diagram.

5.17.1 Detailed Description

Include all declarations from TableSelectionController.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

22.12.2014 10:30:00 GMT

5.18 Diagram/customplotbarchart.cpp File Reference

Implementation file of CustomPlotBarChart class.

```
#include "customplotbarchart.h"
```

5.18.1 Detailed Description

Implementation file of CustomPlotBarChart class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

15.12.2014 18:46:00 GMT

5.19 Diagram/customplotbarchart.h File Reference

Include all declarations from CustomPlotBarChart.

```
#include "Thirdparty/qcustomplot.h"
#include "Model/sensormodel.h"
#include <QtQuick>
#include <QPainter>
#include <QColor>
#include <QVector>
```

Classes

- class CustomPlotBarChart

  The CustomPlotBarChart class Paint a bar chart on view. This class in include in qml code.

Variables

- const int MAX_HEARTRATE = 230

5.19.1 Detailed Description

Include all declarations from CustomPlotBarChart.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

15.12.2014 18:45:00 GMT

5.19.2 Variable Documentation

5.19.2.1 const int MAX_HEARTRATE = 230

Fix attribute for heart rate

5.20 Diagram/customplotlinechart.cpp File Reference

Implementation file of CustomPlotLineChart class.

```
#include "customplotlinechart.h"
```

5.20.1 Detailed Description

Implementation file of CustomPlotLineChart class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

16.12.2014 15:50:00 GMT

5.21 Diagram/customplotlinechart.h File Reference

Include all declarations from CustomPlotBarChart.

```
#include <QPainter>
#include <QtQuick>
#include "Thirdparty/qcustomplot.h"
#include "Model/sensormodel.h"
#include "Model/Data/sensordata.h"
```

Classes

- class CustomPlotLineChart

  The CustomPlotLineChart class Paint a line chart on view. This class in include in qml code.

5.21.1 Detailed Description

Include all declarations from CustomPlotBarChart.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

16.12.2014 15:50:00 GMT

5.22 ImportExport/ImportExport.cpp File Reference

Implementation file of ImportExport class.

```
#include "ImportExport.h"
#include "MeasureType.h"
#include "MoodType.h"
#include "Settings/Settings.h"
```

5.22.1 Detailed Description

Implementation file of ImportExport class.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.23 ImportExport/ImportExport.h File Reference

Class definition of ImportExport.

```
#include <QObject>
#include <QDateTime>
#include <QtSql/QtSql>
#include "Model/Data/sensordata.h"
#include "Connection/DataReceiver.h"
```

Classes

- class ImportExport

  The ImportExport class.

5.23.1 Detailed Description

Class definition of ImportExport.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.24 ImportExport/MeasureType.cpp File Reference

Implementation file of MeasureType class.

```
#include "MeasureType.h"
```

5.24.1 Detailed Description

Implementation file of MeasureType class.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.25 ImportExport/MeasureType.h File Reference

Class definition of MeasureType.

```
#include <QtGlobal>
```

Classes

- class MeasureType

  The MeasureType class is a container for static const strings of measurement types.

5.25.1 Detailed Description

Class definition of MeasureType.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.26 ImportExport/MoodType.cpp File Reference

Implementation file of MoodType class.

```
#include "MoodType.h"
```

5.26.1 Detailed Description

Implementation file of MoodType class.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.27 ImportExport/MoodType.h File Reference

Class definition of MoodType.

```
#include <QtGlobal>
```

Classes

- class MoodType

  The MoodType class is a container for static const strings of mood types.

5.27.1 Detailed Description

Class definition of MoodType.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

5.28 main.cpp File Reference

Implementation file of main.cpp.

```
#include <QApplication>
#include <QGuiApplication>
#include <QQmlApplicationEngine>
#include <QTranslator>
#include <QDebug>
#include "Model/Data/sensordata.h"
#include "Model/sensormodel.h"
#include "Model/activesensorcalcmodel.h"
#include "Model/inactivesensorcalcmodel.h"
#include "Model/selectionmodel.h"
#include "Controller/printcontroller.h"
#include "Controller/selectioncontroller.h"
#include "Controller/initdiagramscontroller.h"
#include "Controller/filtercontroller.h"
#include "Controller/tableselectioncontroller.h"
#include "Diagram/customplotbarchart.h"
#include "Diagram/customplotlinechart.h"
#include "RessourceFilePaths.h"
#include "Connection/BroadcastReceiver.h"
#include "Connection/TcpServer.h"
#include "Settings/Settings.h"
#include "ImportExport/ImportExport.h"
```

Functions

- int main (int argc, char ∗argv[])

  main main method to create application

5.28.1 Detailed Description

Implementation file of main.cpp.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

17.11.2014 15:05:00 GMT

In this file all language files are load. QML engine is definded and view is showed. All models and controllers are init. Data Receiver and database get installed.

5.28.2 Function Documentation

5.28.2.1 int main (int *argc*, char ∗ *argv[ ]*)

main main method to create application

declare a new QVariant typ for view

Parameters

argc	count of elements in argv
argv	include all commando line parameter

Returns

Return 0 if ok otherwise a error code not equal 0

5.29 Model/activesensorcalcmodel.cpp File Reference

TImplementation file of ActiveSensorCalcModel class.

```
#include "activesensorcalcmodel.h"
```

5.29.1 Detailed Description

TImplementation file of ActiveSensorCalcModel class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

16.12.2014 13:16:00 GMT

5.30 Model/activesensorcalcmodel.h File Reference

Include all declarations from ActiveSensorCalcModel.

```
#include "Model/sensorcalcmodel.h"
```

Classes

- class ActiveSensorCalcModel

  The ActiveSensorCalcModel class This class represents a model for active calc values.

5.30.1 Detailed Description

Include all declarations from ActiveSensorCalcModel.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

16.12.2014 13:15:00 GMT

5.31 Model/Data/calcsensordata.cpp File Reference

Implementation file of CalcSensorData class.

```
#include "calcsensordata.h"
```

5.31.1 Detailed Description

Implementation file of CalcSensorData class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

14.12.2014 13:01:00 GMT

5.32 Model/Data/calcsensordata.h File Reference

Include all declarations from CalcSensorData.

```
#include <QString>
```

Classes

- class CalcSensorData

  The CalcSensorData class Store all calculate values in a single class.

5.32.1 Detailed Description

Include all declarations from CalcSensorData.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

14.12.2014 13:00:00 GMT

5.33 Model/Data/sensordata.cpp File Reference

Implementation file of SensorData class.

```
#include "sensordata.h"
```

5.33.1 Detailed Description

Implementation file of SensorData class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

16.12.2014 12:02:00 GMT

5.34 Model/Data/sensordata.h File Reference

Include all declarations from SensorData.

```
#include <QString>
#include <QDateTime>
```

Classes

- class SensorData

  The SensorData class Store all sensor values from a single measurement.

5.34.1 Detailed Description

Include all declarations from SensorData.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

16.12.2014 12:00:00 GMT

5.35 Model/inactivesensorcalcmodel.cpp File Reference

Implementation file of InactiveSensorCalcModel class.

```
#include "inactivesensorcalcmodel.h"
```

5.35.1 Detailed Description

Implementation file of InactiveSensorCalcModel class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

16.12.2014 13:35:00 GMT

5.36 Model/inactivesensorcalcmodel.h File Reference

Include all declarations from InactiveSensorCalcModel.

```
#include "Model/sensorcalcmodel.h"
```

Classes

- class InactiveSensorCalcModel

    The InactiveSensorCalcModel class This class represents a model for inactive calc values.

5.36.1 Detailed Description

Include all declarations from InactiveSensorCalcModel.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

16.12.2014 13:30:00 GMT

5.37 Model/selectionmodel.cpp File Reference

Implementation file of SelectionModel class.

```
#include "selectionmodel.h"
```

5.37.1 Detailed Description

Implementation file of SelectionModel class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

19.12.2014 09:11:00 GMT

5.38 Model/sensorcalcmodel.cpp File Reference

Implementation file of SensorCalcModel class.

```
#include "sensorcalcmodel.h"
```

5.38.1 Detailed Description

Implementation file of SensorCalcModel class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

15.12.2014 21:52:00 GMT

5.39 Model/sensorcalcmodel.h File Reference

Include all declarations from SensorCalcModel.

```
#include <QAbstractListModel>
#include "Model/Data/calcsensordata.h"
#include "Model/sensormodel.h"
#include <QDebug>
```

Classes

- class SensorCalcModel

  The SensorCalcModel class This class represents a model for calculate data to show on view.

5.39.1 Detailed Description

Include all declarations from SensorCalcModel.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

15.12.2014 21:50:00 GMT

5.40 Model/sensormodel.cpp File Reference

Implementation file of SensorModel class.

```
#include "sensormodel.h"
#include "Settings/Settings.h"
```

5.40.1 Detailed Description

Implementation file of SensorModel class.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

27.11.2014 17:52:00 GMT

5.41 Model/sensormodel.h File Reference

Include all declarations from SensorModel.

```
#include <QAbstractListModel>
#include "Model/Data/sensordata.h"
#include <QDebug>
```

Classes

- class SensorModel

  The SensorModel class This class represents a model for inactvie and active sensor data.

5.41.1 Detailed Description

Include all declarations from SensorModel.

Author

Patrick Mathias, Markus Nebel
responsible: Patrick Mathias

Date

27.11.2014 17:50:00 GMT

5.42 Settings/Settings.cpp File Reference

Implementation file of Settings class.

```
#include "Settings.h"
#include <QDebug>
```

5.42.1 Detailed Description

Implementation file of Settings class.

Author

Patrick Mathias, Markus Nebel
responsible: Markus Nebel

Date

12.12.2014 14:33:34 GMT

Index