

RodSystemEstimator

Generated by Doxygen 1.9.3

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	9
4.1 RSE::Core::AbstractDataObject Class Reference	9
4.1.1 Detailed Description	10
4.1.2 Member Function Documentation	10
4.1.2.1 addItem()	11
4.1.2.2 clone()	11
4.1.2.3 deserialize()	11
4.1.2.4 import()	11
4.2 RSE::Core::Array< T > Class Template Reference	11
4.2.1 Detailed Description	13
4.3 RSE::Core::Cable Struct Reference	13
4.3.1 Detailed Description	13
4.4 RSE::Viewers::ConvergenceViewer Class Reference	14
4.4.1 Detailed Description	14
4.5 RSE::Core::Damper Class Reference	15
4.5.1 Detailed Description	15
4.6 RSE::Core::DataBaseCables Class Reference	16
4.6.1 Detailed Description	16
4.7 RSE::Models::DoubleSpinBoxItemDelegate Class Reference	16
4.7.1 Detailed Description	17
4.8 KLP::EnergyFrame Struct Reference	17
4.8.1 Detailed Description	17
4.9 KLP::FrameCollection Struct Reference	18
4.9.1 Detailed Description	18
4.10 KLP::FrameObject< T > Class Template Reference	18
4.11 KLP::FrameObjectIterator< T > Class Template Reference	19
4.11.1 Detailed Description	20
4.12 KLP::Index Struct Reference	20
4.12.1 Detailed Description	21
4.13 KLP::IndexData Struct Reference	21
4.13.1 Detailed Description	21
4.14 RSE::Core::IO Class Reference	21
4.14.1 Detailed Description	22
4.15 RSE::Viewers::KLPGraphViewer Class Reference	22

4.15.1 Detailed Description	23
4.16 RSE::App::MainWindow Class Reference	23
4.16.1 Detailed Description	26
4.17 RSE::Core::Project Class Reference	26
4.18 KLP::Result Class Reference	27
4.18.1 Detailed Description	28
4.19 KLP::ResultInfo Struct Reference	28
4.20 RSE::Core::RodSystem Class Reference	29
4.21 RSE::Core::RodSystemParameters Struct Reference	29
4.21.1 Detailed Description	30
4.22 RSE::Models::RodSystemTableModel Class Reference	30
4.22.1 Detailed Description	31
4.23 RSE::Core::Array< T >::Row< U > Class Template Reference	31
4.23.1 Detailed Description	31
4.24 RSE::Core::ScalarDataObject Class Reference	31
4.24.1 Detailed Description	32
4.24.2 Member Function Documentation	32
4.24.2.1 addItem()	32
4.24.2.2 clone()	33
4.24.2.3 import()	33
4.25 RSE::Solution::SolutionManager Class Reference	33
4.25.1 Detailed Description	34
4.26 RSE::Solution::SolutionOptions Class Reference	34
4.27 RSE::Core::Spans Struct Reference	35
4.27.1 Detailed Description	35
4.28 KLP::StateFrame Struct Reference	36
4.28.1 Detailed Description	36
4.29 RSE::Core::Support Class Reference	36
4.29.1 Detailed Description	37
4.30 RSE::Core::VectorDataObject Class Reference	37
4.30.1 Detailed Description	38
4.30.2 Member Function Documentation	38
4.30.2.1 addItem()	38
4.30.2.2 clone()	38
4.30.2.3 import()	38
5 File Documentation	39
5.1 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/doublespinboxitemdelegate.cpp	
File Reference	39
5.1.1 Detailed Description	39
5.2 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/doublespinboxitemdelegate.h	
File Reference	39
5.2.1 Detailed Description	40

5.3 doublespinboxitemdelegate.h	40
5.4 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/mainwindow.cpp File Reference	40
5.4.1 Detailed Description	41
5.5 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/mainwindow.h File Reference	41
5.5.1 Detailed Description	42
5.6/mainwindow.h	42
5.7 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/rodsystemtablemodel.cpp File Reference	44
5.7.1 Detailed Description	44
5.8 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/rodsystemtablemodel.h File Reference	44
5.8.1 Detailed Description	44
5.9 rodsystemtablemodel.h	45
5.10 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/uiconstants.h File Reference	45
5.10.1 Detailed Description	46
5.11 uiconstants.h	46
5.12 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/abstractdataobject.cpp File Reference	46
5.12.1 Detailed Description	46
5.13 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/abstractdataobject.h File Reference	47
5.13.1 Detailed Description	47
5.14 abstractdataobject.h	48
5.15 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/aliasdata.h File Reference	48
5.15.1 Detailed Description	49
5.16 aliasdata.h	49
5.17 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/array.cpp File Reference	49
5.17.1 Detailed Description	50
5.18 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/array.h File Reference	50
5.18.1 Detailed Description	51
5.19 array.h	51
5.20 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/constants.h File Reference	52
5.20.1 Detailed Description	53
5.21 constants.h	53
5.22 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/damper.cpp File Reference	53
5.22.1 Detailed Description	53
5.23 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/damper.h File Reference	54
5.23.1 Detailed Description	54
5.24 damper.h	54
5.25 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.cpp File Reference	55
5.25.1 Detailed Description	55

5.26	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.h File Reference	55
5.26.1	Detailed Description	55
5.27	databasecables.h	56
5.28	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/fileutilities.cpp File Reference	56
5.28.1	Detailed Description	56
5.29	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/fileutilities.h File Reference	57
5.29.1	Detailed Description	57
5.30	fileutilities.h	57
5.31	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.cpp File Reference	58
5.31.1	Detailed Description	58
5.32	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.h File Reference	58
5.32.1	Detailed Description	59
5.33	io.h	59
5.34	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/project.cpp File Reference	59
5.34.1	Detailed Description	60
5.35	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/project.h File Reference	60
5.35.1	Detailed Description	61
5.36	project.h	61
5.37	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.cpp File Reference	62
5.37.1	Detailed Description	62
5.38	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.h File Reference	63
5.38.1	Detailed Description	63
5.39	rodsystem.h	63
5.40	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.cpp File Reference	64
5.40.1	Detailed Description	64
5.41	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.h File Reference	64
5.41.1	Detailed Description	65
5.42	scalardataobject.h	65
5.43	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionmanager.cpp File Reference	65
5.43.1	Detailed Description	66
5.44	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionmanager.h File Reference	66
5.44.1	Detailed Description	66
5.45	solutionmanager.h	67
5.46	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionoptions.cpp File Reference	67
5.46.1	Detailed Description	67
5.47	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionoptions.h File Reference	68
5.47.1	Detailed Description	68

5.48 solutionoptions.h	68
5.49 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.cpp File Reference	69
5.49.1 Detailed Description	69
5.50 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.h File Reference	69
5.50.1 Detailed Description	70
5.51 support.h	70
5.52 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.cpp File Reference	70
5.52.1 Detailed Description	71
5.53 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.h File Reference	71
5.53.1 Detailed Description	71
5.54 vectordataobject.h	71
5.55 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/framecollection.h File Reference	72
5.55.1 Detailed Description	72
5.56 framecollection.h	73
5.57 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobject.cpp File Reference	73
5.57.1 Detailed Description	73
5.58 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobject.h File Reference	74
5.58.1 Detailed Description	74
5.59 frameobject.h	74
5.60 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.cpp File Reference	75
5.60.1 Detailed Description	75
5.61 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.h File Reference	75
5.61.1 Detailed Description	76
5.62 frameobjectiterator.h	76
5.63 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/index.h File Reference	76
5.63.1 Detailed Description	77
5.64 index.h	77
5.65 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/result.cpp File Reference	77
5.65.1 Detailed Description	78
5.66 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/result.h File Reference	78
5.66.1 Detailed Description	78
5.67 result.h	79
5.68 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/types.h File Reference	79
5.68.1 Detailed Description	80
5.69 types.h	80
5.70 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/main/main.cpp File Reference	81
5.70.1 Detailed Description	81
5.71 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/apputilities.cpp File Reference	81

5.71.1 Detailed Description	82
5.72 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/apputilities.h File Reference	82
5.72.1 Detailed Description	82
5.73 apputilities.h	83
5.74 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.cpp File Reference	83
5.74.1 Detailed Description	83
5.75 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.h File Reference	83
5.75.1 Detailed Description	84
5.76 convergenceviewer.h	84
5.77 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.cpp File Reference	85
5.77.1 Detailed Description	85
5.78 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.h File Reference	85
5.78.1 Detailed Description	86
5.79 klpgraphviewer.h	86

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

RSE::Core::Array< T >	11
RSE::Core::Array< double >	11
RSE::Core::Cable	13
RSE::Core::Damper	15
RSE::Core::DataBaseCables	16
KLP::EnergyFrame	17
KLP::FrameCollection	18
KLP::FrameObject< T >	18
KLP::FrameObject< float >	18
KLP::FrameObjectIterator< T >	19
KLP::Index	20
KLP::IndexData	21
RSE::Core::IO	21
RSE::Core::Project	26
QDialog	
RSE::Viewers::KLPGraphViewer	22
QMainWindow	
RSE::App::MainWindow	23
QObject	
RSE::Core::AbstractDataObject	9
RSE::Core::ScalarDataObject	31
RSE::Core::VectorDataObject	37
RSE::Solution::SolutionManager	33
QStandardItemModel	
RSE::Models::KLPResultListModel	??
RSE::Models::RodSystemTableModel	30
QStyledItemDelegate	
RSE::Models::DoubleSpinBoxItemDelegate	16
QWidget	
RSE::Viewers::ConvergenceViewer	14
KLP::Result	27
KLP::ResultInfo	28
RSE::Core::RodSystem	29
RSE::Core::RodSystemParameters	29
RSE::Core::Array< T >::Row< U >	31

RSE::Solution::SolutionOptions	34
RSE::Core::Spans	35
KLP::StateFrame	36
RSE::Core::Support	36

Chapter 2

Class Index

2.1 Class List

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

RSE::Core::AbstractDataObject	
Data object which is designed in the way to be represented in a table easily	9
RSE::Core::Array< T >	
Numerical array class	11
RSE::Core::Cable	
Mechanical properties of a cable	13
RSE::Viewers::ConvergenceViewer	
Class to represent convergence of viscosities	14
RSE::Core::Damper	
Class to compute and collect properties of a damper	15
RSE::Core::DataBaseCables	
Aggregate data of cables	16
RSE::Models::DoubleSpinBoxItemDelegate	
Class to specify how table values can be edited	16
KLP::EnergyFrame	
Energy quantities associated with a frame	17
KLP::FrameCollection	
Set of all quantities belonged to a frame	18
KLP::FrameObject< T >	18
KLP::FrameObjectIterator< T >	
Class to iterate through data of a record	19
KLP::Index	
Structure to navigate through records	20
KLP::IndexData	
Data of each record	21
RSE::Core::IO	
Class to save the project and solution data	21
RSE::Viewers::KLPGraphViewer	
Class to graphically represent content of KLP output files	22
RSE::Models::KLPResultListModel	??
RSE::App::MainWindow	
Central window of the program	23
RSE::Core::Project	26
KLP::Result	
Class to aggregate all the records	27

KLP::ResultInfo	28
RSE::Core::RodSystem	29
RSE::Core::RodSystemParameters	
Parameters of a rod system	29
RSE::Models::RodSystemTableModel	
Table model to set and represent data of a rod system	30
RSE::Core::Array< T >::Row< U >	
Proxy class to acquire a row by index	31
RSE::Core::ScalarDataObject	
Scalar data object	31
RSE::Solution::SolutionManager	
Class to control the solution process	33
RSE::Solution::SolutionOptions	34
RSE::Core::Spans	
Computed parameters of spans	35
KLP::StateFrame	
Kinematic and dynamic quantities associated with a frame	36
RSE::Core::Support	
Class to aggregate data of supports	36
RSE::Core::VectorDataObject	
Vector data object	37

Chapter 3

File Index

3.1 File List

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

/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ doublespinboxitemdelegate.cpp	
DoubleSpinBoxItemDelegate	39
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ doublespinboxitemdelegate.h	
DoubleSpinBoxItemDelegate	39
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ mainwindow.cpp	
Definition of the MainWindow class	40
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ mainwindow.h	
Declaration of the MainWindow class	41
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ rodsystemtablemodel.cpp	
Definition of the RodSystemTableModel class	44
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ rodsystemtablemodel.h	
Declaration of the RodSystemTableModel class	44
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ uiconstants.h	
Graphical constants shared between several widgets	45
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ abstractdataobject.cpp	
Implementation of the AbstractDataObject class	46
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ abstractdataobject.h	
Declaration of the AbstractDataObject class	47
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ aliasdata.h	
Specification of data types used in a project	48
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ array.cpp	
Implementation of the Array class	49
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ array.h	
Declaration of the Array class	50
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ constants.h	
Computational constants	52
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ damper.cpp	
Definition of the Damper class	53
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ damper.h	
Declaration the Damper class	54
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ databasecables.cpp	
Definition of the DataBaseCables class	55
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ databasecables.h	
Declaration of the DataBaseCables class	55
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ fileutilities.cpp	
Definition of utilites targeted to working with files	56

/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/fileutilities.h	
Declaration of utilities targeted to working with files	57
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.cpp	
Definition of the IO class	58
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.h	
Declaration of the IO class	58
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/project.cpp	
Definition of the Project class	59
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/project.h	
Declaration of the Project class	60
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.cpp	
Definition of the RodSystem class	62
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.h	
Declaration of the RodSystem class	63
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.cpp	
Implementation of the ScalarDataObject class	64
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.h	
Declaration of the ScalarDataObject class	64
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionmanager.cpp	
Definition of the SolutionManager class	65
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionmanager.h	
Declaration of the SolutionManager class	66
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionoptions.cpp	
Definition of the SolutionOptions class	67
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionoptions.h	
Declaration of the SolutionOptions class	68
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.cpp	
Definition of the Support class	69
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.h	
Declaration of the Support class	69
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.cpp	
Implementation of the VectorDataObject class	70
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.h	
Declaration of the VectorDataObject class	71
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/framecollection.h	
Collection of the data associated with the specified frame	72
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobject.cpp	
Definition of the FrameObject class	73
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobject.h	
Declaration of the FrameObject class	74
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.cpp	
Definition of the FrameObjectIterator class	75
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.h	
Declaration of the FrameObjectIterator class	75
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/index.h	
Specification of a structure to index records	76
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/result.cpp	
Definition of the Result class	77
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/result.h	
Declaration of the Result class	78
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/types.h	
Specification of data types in a KLP file	79
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/main/main.cpp	
Startup	81
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/appaliases.h	
Declaration of aliases used in the application	??
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/apputilities.cpp	
Definition of utilities targeted to working with application data	81

/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/apputilities.h	
Declaration of utilities targeted to working with application data	82
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.cpp	
Definition of the ConvergenceViewer class	83
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.h	
Declaration of the ConvergenceViewer class	83
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.cpp	
Definition of the KLPGraphViewer class	85
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.h	
Declaration of the KLPGraphViewer class	85
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpresultlistmodel.cpp	
Definition of the KLPResultListModel class	??
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpresultlistmodel.h	
Declaration of the KLPResultListModel class	??

Chapter 4

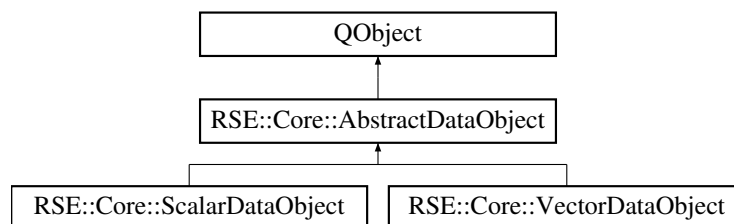
Class Documentation

4.1 RSE::Core::AbstractDataObject Class Reference

Data object which is designed in the way to be represented in a table easily.

```
#include <abstractdataobject.h>
```

Inheritance diagram for RSE::Core::AbstractDataObject:



Public Types

- enum **ObjectType** { **kScalar** , **kVector** , **kMatrix** , **kSurface** }

Public Member Functions

- **AbstractDataObject** (ObjectType type, QString const &name)
Base constructor.
- virtual **AbstractDataObject** * **clone** () const =0
- virtual **DataItemType** & **addItem** (DataKeyType key)=0
- void **removeItem** (DataValueType key)
Remove the entity paired to the specified key.
- bool **changeItemKey** (DataKeyType oldKey, DataKeyType newKey, DataHolder *items=nullptr)
Modify an existing key.
- bool **setArrayValue** (DataKeyType key, DataValueType newValue, IndexType iRow=0, IndexType iColumn=0)
Set an array value with the specified indices.
- DataValueType **arrayValue** (DataKeyType key, IndexType iRow=0, IndexType iColumn=0)
Retrieve a value from an array.

- `std::vector< DataKeyType > keys () const`
Retrieve all the keys.
- `quint32 numberItems () const`
- `DataHolder const & getItems ()`
- `DataIDType id () const`
- `ObjectType type () const`
- `QString const & name () const`
- `void setName (QString const &name)`
- `virtual void serialize (QDataStream &stream) const`
Serialize an abstract data object.
- `virtual void deserialize (QDataStream &stream)`
Partly deserialize an abstract data object.
- `virtual void import (QTextStream &stream)=0`
- `void write (QTextStream &stream) const`
Write an abstract data object to a file.

Static Public Member Functions

- `static DataIDType maxObjectID ()`
- `static void setMaxObjectID (DataIDType iMaxObjectID)`

Protected Attributes

- `const ObjectType mkType`
- `QString mName`
- `DataIDType mID`
- `DataHolder mItems`

Static Private Attributes

- `static DataIDType smMaxObjectID = 0`

Friends

- `QDataStream & operator<< (QDataStream &stream, AbstractDataObject const &obj)`
Print a data object to a binary stream.

4.1.1 Detailed Description

Data object which is designed in the way to be represented in a table easily.

4.1.2 Member Function Documentation

4.1.2.1 addItem()

```
virtual DataItemType & RSE::Core::AbstractDataObject::addItem (
    DataKeyType key ) [pure virtual]
```

Implemented in [RSE::Core::ScalarDataObject](#), and [RSE::Core::VectorDataObject](#).

4.1.2.2 clone()

```
virtual AbstractDataObject * RSE::Core::AbstractDataObject::clone ( ) const [pure virtual]
```

Implemented in [RSE::Core::ScalarDataObject](#), and [RSE::Core::VectorDataObject](#).

4.1.2.3 deserialize()

```
void AbstractDataObject::deserialize (
    QDataStream & stream ) [virtual]
```

Partly deserialize an abstract data object.

It is assumed that a type and name have already been assigned. So, only an identifier and items need to be set.

4.1.2.4 import()

```
virtual void RSE::Core::AbstractDataObject::import (
    QTextStream & stream ) [pure virtual]
```

Implemented in [RSE::Core::ScalarDataObject](#), and [RSE::Core::VectorDataObject](#).

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

- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/abstractdataobject.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/abstractdataobject.cpp](#)

4.2 RSE::Core::Array< T > Class Template Reference

Numerical array class.

```
#include <array.h>
```

Classes

- class [Row](#)
Proxy class to acquire a row by index.

Public Member Functions

- **Array** (IndexType numRows=0, IndexType numCols=0)
- **Array** ([Array](#)< T > const &another)
Copy constructor.
- **Array** ([Array](#)< T > &&another)
Move constructor.
- T * **data** ()
- void **resize** (IndexType numRows, IndexType numCols)
Resize and copy previous values if possible.
- void **removeColumn** (IndexType iRemoveColumn)
Remove a column by index.
- void **swapColumns** (IndexType iFirstColumn, IndexType iSecondColumn)
Swap two columns.
- void **clear** ()
Remove all the values.
- IndexType **rows** () const
- IndexType **cols** () const
- IndexType **size** () const
- [Row](#)< T > **operator[]** (IndexType iRow)
- [Row](#)< T > **operator[]** (IndexType iRow) const
- [Array](#) & **operator=** ([Array](#)< T > const &another)
Assignment operator.

Private Attributes

- IndexType **mNumRows**
Number of rows.
- IndexType **mNumCols**
Number of columns.
- T * **mpData** = nullptr
Pointer to the data stored.

Friends

- template<typename K >
QDebug **operator**<< (QDebug stream, [Array](#)< K > &array)
Print all array values using the matrix format.
- template<typename K >
QDataStream & **operator**<< (QDataStream &stream, [Array](#)< K > const &array)
Write an array to a binary stream.
- template<typename K >
QDataStream & **operator**>> (QDataStream &stream, [Array](#)< K > &array)
Read an array from a stream.
- template<typename K >
QTextStream & **operator**<< (QTextStream &stream, [Array](#)< K > const &array)
Write an array to a text stream.

4.2.1 Detailed Description

```
template<typename T>
class RSE::Core::Array< T >
```

Numerical array class.

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

- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/array.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/array.cpp](#)

4.3 RSE::Core::Cable Struct Reference

Mechanical properties of a cable.

```
#include <databasecables.h>
```

Public Attributes

- `std::string name`
Name of a cable.
- `double bendingStiffness`
Bending stiffness, N.
- `double torsionalStiffness`
Torsional stiffness, N.
- `double massPerLength`
Mass per length, kg/m.
- `double youngsModulus`
Youngs modulus, Pa.
- `double area`
Area of a cross-section, m².

4.3.1 Detailed Description

Mechanical properties of a cable.

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

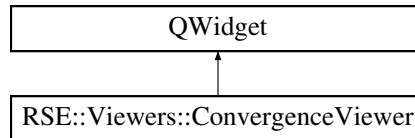
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.h](#)

4.4 RSE::Viewers::ConvergenceViewer Class Reference

Class to represent convergence of viscosities.

```
#include <convergenceviewer.h>
```

Inheritance diagram for RSE::Viewers::ConvergenceViewer:



Public Member Functions

- **ConvergenceViewer** (QString const &pathFile, QWidget *pParent=nullptr)
- void **plot** ()
Represent the convergence.

Private Member Functions

- void **initialize** ()
Initialize the widget.
- bool **read** ()
Read the file contained viscosities of dampers.

Private Attributes

- QString const **mkPathFile**
- QCustomPlot * **mpFigure**
- QStringList **mAvailableColors**
- QVector< QCPScatterStyle::ScatterShape > **mAvailableShapes**
- QVector< int > **mCalcModes**
- [Core::Array](#)< double > **mDampingValues**

4.4.1 Detailed Description

Class to represent convergence of viscosities.

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

- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.cpp](#)

4.5 RSE::Core::Damper Class Reference

Class to compute and collect properties of a damper.

```
#include <damper.h>
```

Public Member Functions

- **Damper** (double massCable, double massLoadedCable, double workingLength, double bouncerLength, double springLength=0, double springStiffness=0)
- double **massCable** () const
- double **massLoadedCable** () const
- double **workingLength** () const
- double **bouncerLength** () const
- double **springLength** () const
- double **springStiffness** () const
- void **setMassCable** (double massCable)
- void **setMassLoadedCable** (double massLoadedCable)
- void **setWorkingLength** (double workingLength)
- void **setBouncerLength** (double bouncerLength)
- void **setSpringLength** (double springLength)
- void **setSpringStiffness** (double springStiffness)
- void **computeSpring** ()

Compute parameters of a spring belonged to a damper.

Private Attributes

- double **mMassCable**
Mass of a cable, kg.
- double **mMassLoadedCable**
Mass of a cable with ice on it, kg.
- double **mWorkingLength**
Working length, m.
- double **mBouncerLength**
Length of a bouncer, m.
- double **mSpringLength** = 0.0
Length of a spring, m.
- double **mSpringStiffness** = 0.0
Spring stiffness, N/m.

4.5.1 Detailed Description

Class to compute and collect properties of a damper.

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

- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/damper.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/damper.cpp](#)

4.6 RSE::Core::DataBaseCables Class Reference

Aggregate data of cables.

```
#include <databasecables.h>
```

Public Member Functions

- **DataBaseCables** (QString const &directory, QString const &fileName)
- std::vector< std::string > **names** () const
Names of available cables.
- **Cable** const & **getItem** (std::string const &name) const

Private Member Functions

- bool **readDataBase** (QString const &pathFile)
Read a database from a file.

Private Attributes

- std::unordered_map< std::string, **Cable** > **mData**

4.6.1 Detailed Description

Aggregate data of cables.

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

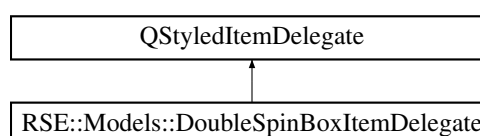
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.h
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.cpp

4.7 RSE::Models::DoubleSpinBoxItemDelegate Class Reference

Class to specify how table values can be edited.

```
#include <doublespinboxitemdelegate.h>
```

Inheritance diagram for RSE::Models::DoubleSpinBoxItemDelegate:



Public Member Functions

- **DoubleSpinBoxItemDelegate** (QObject *parent=nullptr)
- QWidget * **createEditor** (QWidget *parent, const QStyleOptionViewItem &option, const QModelIndex &index) const override
Create a double value editor.
- void **setEditorData** (QWidget *pEditor, const QModelIndex &index) const override
Specify data to show.
- void **setModelData** (QWidget *pEditor, QAbstractItemModel *pModel, const QModelIndex &index) const override
Set data to a model.
- void **updateEditorGeometry** (QWidget *pEditor, const QStyleOptionViewItem &option, const QModelIndex &index) const override
Set a geometry to render.

4.7.1 Detailed Description

Class to specify how table values can be edited.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/[doublespinboxitemdelegate.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/[doublespinboxitemdelegate.cpp](#)

4.8 KLP::EnergyFrame Struct Reference

Energy quantities associated with a frame.

```
#include <framecollection.h>
```

Public Attributes

- [FloatFrameObject](#) **kinetic**
- [FloatFrameObject](#) **potential**
- [FloatFrameObject](#) **full**

4.8.1 Detailed Description

Energy quantities associated with a frame.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/[framecollection.h](#)

4.9 KLP::FrameCollection Struct Reference

Set of all quantities belonged to a frame.

```
#include <framecollection.h>
```

Public Attributes

- int **numRods**
Number of rods.
- [FloatFrameObject](#) **parameter**
Parameter.
- [FloatFrameObject](#) **naturalLength**
Natural length.
- [FloatFrameObject](#) **accumulatedNaturalLength**
- [FloatFrameObject](#) **coordinates** [kNumDirections]
Coordinates.
- [StateFrame](#) **state**
Regular state.
- [StateFrame](#) **projectedState**
Projected regular state.
- [StateFrame](#) **firstDerivativeState**
First-order derivate of the state with respect to time.
- [StateFrame](#) **secondDerivativeState**
Second-order derivate of the state with respect to time.
- [StateFrame](#) **errorState**
State error.
- [FloatFrameObject](#) **strain**
Strain.
- std::vector< [StateFrame](#) > **modalStates**
Set of modal states.
- [FloatFrameObject](#) **frequencies**
Frequencies.
- [EnergyFrame](#) **energy**
Energy.

4.9.1 Detailed Description

Set of all quantities belonged to a frame.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/[framecollection.h](#)

4.10 KLP::FrameObject< T > Class Template Reference

Public Types

- using **iterator** = [FrameObjectIterator](#)< T >

Public Member Functions

- **FrameObject** (T const *pData=nullptr, T normFactor=1.0, qint64 size=0, qint64 step=1)
- bool **isEmpty** () const
- qint64 **size** () const
- **iterator begin** ()
- **iterator end** ()
- **iterator operator[]** (int index)

Private Attributes

- T const * **mpData**
- T **mNormFactor**
- qint64 **mSize**
- qint64 **mStep**

Friends

- template<typename K >
QDebug **operator<<** (QDebug stream, [FrameObject< K >](#) &frameObject)

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/[frameobject.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/[frameobject.cpp](#)

4.11 KLP::FrameObjectIterator< T > Class Template Reference

Class to iterate through data of a record.

```
#include <frameobjectiterator.h>
```

Public Types

- using **self_type** = [FrameObjectIterator< T >](#)
- using **iterator_category** = std::random_access_iterator_tag
- using **difference_type** = std::ptrdiff_t
- using **value_type** = T
- using **pointer** = T const *
- using **reference** = T const &

Public Member Functions

- **FrameObjectIterator** (pointer pData, T normFactor, qint64 step)
- value_type **operator*** ()
- **self_type & operator++** ()
- **self_type operator++** (int)
- **self_type operator+** (const difference_type &movement)
- difference_type **operator-** (const [FrameObjectIterator](#) &another) const

Private Attributes

- pointer **mpData**
- T **mNormFactor**
- quint64 const **mStep**

Friends

- bool **operator==** ([self_type](#) const &first, [self_type](#) const &second)
- bool **operator!=** ([self_type](#) const &first, [self_type](#) const &second)

4.11.1 Detailed Description

```
template<typename T>
class KLP::FrameObjectIterator< T >
```

Class to iterate through data of a record.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/[frameobjectiterator.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/[frameobjectiterator.cpp](#)

4.12 KLP::Index Struct Reference

Structure to navigate through records.

```
#include <index.h>
```

Public Member Functions

- **Index** ()
Base constructor.

Public Attributes

- std::vector< [IndexData](#) > **data**
Data.
- quint64 **recordShift** = 0
Shift of the main record.
- quint64 **relativeDataShift** = 0
Relative shift of data.

4.12.1 Detailed Description

Structure to navigate through records.

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

- `/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/index.h`

4.13 KLP::IndexData Struct Reference

Data of each record.

```
#include <index.h>
```

Public Attributes

- `qint64 position = 0`
Position of a record in the buffer.
- `qint64 size = 0`
Size of a record.
- `qint64 step = 1`
Step for iterating inside a record.
- `qint64 partSize = 0`
Partial length of a quantity inside a record.

4.13.1 Detailed Description

Data of each record.

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

- `/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/index.h`

4.14 RSE::Core::IO Class Reference

Class to save the project and solution data.

```
#include <io.h>
```

Public Member Functions

- `IO (QString const &lastPath)`
- `QString const &lastPath () const`
- `QString const &extension () const`
- `void saveAs (QString const &pathFile, Project &project, Solution::SolutionOptions &options)`
Save the project and solution data to a file.
- `IOPair open (QString const &pathFile, DataBaseCables const &dataBaseCables)`
Read the computational data from a file.

Private Attributes

- const QString **mkProjectExtension** = ".rse"
- QString **mLastPath**

4.14.1 Detailed Description

Class to save the project and solution data.

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

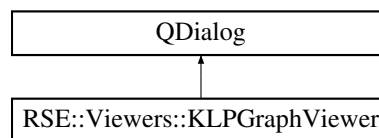
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.cpp](#)

4.15 RSE::Viewers::KLPGraphViewer Class Reference

Class to graphically represent content of KLP output files.

```
#include <klpgraphviewer.h>
```

Inheritance diagram for RSE::Viewers::KLPGraphViewer:



Public Member Functions

- **KLPGraphViewer** (QString const &lastPath, QSettings &settings, QWidget *pParent=nullptr)
- void **openResultsDialog** ()
Open results using a file system dialog.
- void **openResults** (QStringList const &locationFiles)
Open a set of results using their locations.

Private Member Functions

- void **initialize** ()
Intialize default graphical objects.
- void **createContent** ()
Construct graphical interface.
- ads::CDockWidget * **createResultWidget** ()
Create a widget to open and deal with KLP results.
- ads::CDockWidget * **createFigureWidget** ()
Create a widget to plot graphs.
- ads::CDockWidget * **createConstructorWidget** ()
Create a widget to construct graphs.
- ads::CDockWidget * **createPropertyWidget** ()
Create a widget to modify properties of graphs.
- void **processSelectedResults** ()
Process selected results.
- void **showResultInfo** (KLP::ResultInfo const &info)
Show information about the selected result.
- void **saveSettings** ()
Save settings to a file.
- void **restoreSettings** ()
Restore settings from a file.
- void **closeEvent** (QCloseEvent *pEvent) override
Save settings and delete handling widgets before closing the window.

Private Attributes

- QString **mLastPath**
- QSettings & **mSettings**
- ads::CDockManager * **mpDockManager** = nullptr
- QCustomPlot * **mpFigure**
- QListView * **mpListResults**
- QTextEdit * **mpTextInfo**
- RSE::Models::KLPResultListModel * **mpResultListModel**
- Results **mResults**

4.15.1 Detailed Description

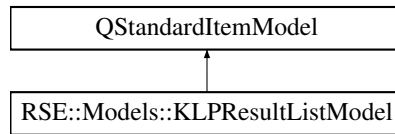
Class to graphically represent content of KLP output files.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/[klpgraphviewer.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/[klpgraphviewer.cpp](#)

4.16 RSE::Models::KLPResultListModel Class Reference

Inheritance diagram for RSE::Models::KLPResultListModel:



Public Member Functions

- **KLPResultListModel** (Results &results, QObject *pParent=nullptr)
- void **updateContent** ()
Create items linked to results.
- void **removeSelected** ()
Remove selected results.

Private Member Functions

- void **clearContent** ()
Remove all the items created.

Private Attributes

- Results & **mResults**

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

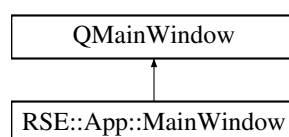
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/[klpresultlistmodel.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/[klpresultlistmodel.cpp](#)

4.17 RSE::App::MainWindow Class Reference

Central window of the program.

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

Inheritance diagram for RSE::App::MainWindow:



Public Member Functions

- **MainWindow** (QWidget *parent=nullptr)
- void **setMassCable** (double value)
Set mass of a cable.
- void **setMassLoadedCable** (double value)
Set mass of a cable with ice on it.
- void **setWorkingLength** (double value)
Specify working length.
- void **setBouncerLength** (double value)
Set length of a bouncer.
- void **setSpringLength** (double value)
Assign length of a spring.
- void **setSpringStiffness** (double value)
Set spring stiffness of a damper.
- void **setCable** (QString const &name)
Assign cables to a rod system.
- void **setForce** (double value)
Specify stretching force.
- void **setLongitudinalStiffness** (double value)
Specify longitudinal stiffness of all supports.
- void **setVerticalStiffness** (double value)
Specify vertical stiffness of all supports.
- void **createProject** ()
Open a new project.
- void **openProjectDialog** ()
Open a project by means of a dialog window.
- void **openProject** (QString const &pathFile)
Open a project using a path specified.
- void **saveAsProject** ()
Save the project using a dialog window.
- void **saveProject** ()
Save the current project.

Private Slots

- void **saveSettings** ()
Save current graphical settings of floating widgets.
- void **restoreSettings** ()
Read graphical settings from a file.
- void **computeSpring** ()
Compute parameters of a spring.
- void **computeSpans** ()
Compute length of all cables.
- void **runRodSystemSolution** ()
Solve the rod system.
- void **runOptimizationSolution** ()
Optimize viscosities of dampers.
- void **appendOutputData** (QByteArray const &data)
Process the message from the solution process.

- void **showConvergence** ()
Represent the convergence of the optimization process.
- void **showResults** ()
Represent the results obtained via KLPALGSYS.
- void **setProjectTitle** ()
Set the name of a project.
- void **setProjectData** ()
Set project data.
- void **setSolutionOptions** ()
Set the data to be used as the solution parameters.
- void **setCurrentCable** ()
Select a current cable.
- void **setBlockedSignals** (bool)
(Un)Block all the signals from widget
- void **aboutProgram** ()
Show the information about the program.

Private Member Functions

- void **initialize** ()
Set the state and geometry of the central window.
- void **createContent** ()
Create all the widgets and links between them.
- void **createDefaultProject** ()
Create a default project.
- void **createDefaultSolutionOptions** ()
Create default solution options.
- void **closeEvent** (QCloseEvent *pEvent) override
Save settings and parameters of project while closing the central window.
- ads::CDockWidget * **createDamperWidget** ()
Create a widget to specify data of a damper.
- ads::CDockWidget * **createRodSystemWidget** ()
Create a widget to set and control data of a rod system.
- ads::CDockWidget * **createSupportWidget** ()
Create a widget to specify data of supports.
- ads::CDockWidget * **createCalculationWidget** ()
Create a widget to control the solution process.
- ads::CDockWidget * **createConsole** ()
Construct a widget to view solution information.
- void **specifyMenuConnections** ()
Specify menu interactions.

Private Attributes

- Ui::MainWindow * **mpUi**
- ads::CDockManager * **mpDockManager**
- [Models::RodSystemTableModel](#) * **mpRodSystemTableModel**
- [Models::DoubleSpinBoxItemDelegate](#) * **mpDoubleSpinBoxItemDelegate**
- QDoubleSpinBox * **mpMassCable**
- QDoubleSpinBox * **mpMassLoadedCable**
- QDoubleSpinBox * **mpWorkingLength**
- QDoubleSpinBox * **mpBouncerLength**
- QDoubleSpinBox * **mpSpringLength**
- QDoubleSpinBox * **mpSpringStiffness**
- QComboBox * **mpNameCable**
- QDoubleSpinBox * **mpForce**
- QDoubleSpinBox * **mpLongitudinalStiffness**
- QDoubleSpinBox * **mpVerticalStiffness**
- QSpinBox * **mpNumCalcModes**
- QSpinBox * **mpNumDampModes**
- QSpinBox * **mpStepModes**
- QDoubleSpinBox * **mpTolTrunc**
- QTextEdit * **mpConsole**
- [RSE::Core::Project](#) * **mpProject**
- [RSE::Solution::SolutionManager](#) * **mpSolutionManager**
- [RSE::Solution::SolutionOptions](#) * **mpSolutionOptions**
- [RSE::Core::IO](#) * **mpIO**
- QSharedPointer< QSettings > **mpSettings**

4.17.1 Detailed Description

Central window of the program.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/[mainwindow.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/[mainwindow.cpp](#)

4.18 RSE::Core::Project Class Reference

Public Member Functions

- **Project** (QString const &name, [DataBaseCables](#) dataBaseCables, [Damper](#) damper, [RodSystem](#) rodSystem, [Support](#) support)
- QString const & **name** () const
- void **setName** (QString const &name)
- [Damper](#) & **damper** ()
- [RodSystem](#) & **rodSystem** ()
- [Support](#) & **support** ()
- [DataBaseCables](#) const & **dataBaseCables** () const
- void **readTemplateData** (QString const &path)
Read template data.
- void **writeCalcData** (QString const &path, [Solution::SolutionOptions](#) const &options)
Write the computational data.

Private Member Functions

- [AbstractDataObject](#) * **addDataObject** (AbstractDataObject::ObjectType type)
Create a data object with the specified type.
- void **importDataObjects** (QString const &path, QString const &fileName)
Import several data objects from a file.
- void **readProjectID** (QString const &path)
Read the identifier of a project.
- void **modifyScalarDataObjects** ()
Modify scalar data objects.
- void **modifyVectorDataObjects** ([Spans](#) const &spans)
Modify vector data objects.
- void **writeDataObjects** (DataObjects const &dataObjects, QString const &path, QString const &fileName)
Write data objects to a file.
- void **writeRods** (QString const &path, QString const &fileName)
Write data of rods.
- void **writeProgram** (QString const &path, QString const &fileName, int numRods, int numCalcModes)
Write data of a program.

Private Attributes

- QString **mName**
Name of a project.
- [Damper](#) **mDamper**
Parameters of a damper.
- [RodSystem](#) **mRodSystem**
Parameters of a rod system.
- [Support](#) **mSupport**
Parameters of supports.
- [DataBaseCables](#) **mDataBaseCables**
Database of cables.
- DataObjects **mScalarDataObjects**
Data objects.
- DataObjects **mVectorDataObjects**
- int **mProjectID**
[Project](#) identifier.
- QStringList **mRods**
Content of the file named RODS.
- QStringList **mProgram**
Content of the file name PROG.

Static Private Attributes

- static const QString **skProjectExtension**
[Project](#) extension.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[project.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[project.cpp](#)

4.19 KLP::Result Class Reference

Class to aggregate all the records.

```
#include <result.h>
```

Public Member Functions

- **Result** (QString const &pathFile)
- bool **isEmpty** () const
- std::vector< float > const & **time** () const
- QString const & **pathFile** () const
- int **numRods** (qint64 iFrame) const
Get the number of rods associated with the requested frame.
- **ResultInfo** **info** () const
Retrieve general information about a result file.
- **FloatFrameObject** **getFrameObject** (qint64 iFrame, **RecordType** type, float normFactor=1.0f, qint64 shift=0) const
Get the object associated with the requested frame.
- **FrameCollection** **getFrameCollection** (qint64 iFrame) const
Retrieve the collection of the frame objects.
- void **update** ()
Retrieve the updated content from the file.

Private Member Functions

- bool **read** ()
Read all the content of the file.
- void **buildIndex** ()
Construct an object to navigate through records.
- void **setStateFrameData** (**StateFrame** &state, **RecordType** type, qint64 iFrame, qint64 iStartData, std::vector< float > const &normFactors) const
Specify state data for each direction.

Private Attributes

- QString const **mkPathFile**
Path to the KLP file.
- QByteArray **mContent**
Content of the file.
- std::vector< **Index** > **mIndex**
Index of the data buffer.
- qint64 **mNumRecords**
Number of records.
- std::vector< float > **mTime**
Time array.
- char **mNumBytesRod**
Number of bytes per rod.

4.19.1 Detailed Description

Class to aggregate all the records.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/[result.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/[result.cpp](#)

4.20 KLP::ResultInfo Struct Reference

Public Attributes

- QDateTime **creationDateTime**
Creation date and time.
- quint64 **numTotalRecords**
Total number of records.
- quint64 **numTimeRecords**
Number of time records.
- uint **fileSize**
File size, Kb.
- uint **identifier**
Unique identifier.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/[result.h](#)

4.21 RSE::Core::RodSystem Class Reference

Public Member Functions

- **RodSystem** (std::vector< double > distances, [Cable](#) const &cable, double force)
- std::vector< double > const & **distances** () const
- std::string const & **nameCable** () const
- double **force** () const
- int **numRods** () const
- double **massPerLength** () const
- void **setDistances** (std::vector< double > const &distances)
Specify distances between supports.
- void **setCable** ([Cable](#) const &cable)
Modify the cable used in the rod system.
- void **setForce** (double force)
- [Spans](#) **computeSpans** ()
Compute characteristics of spans.

Private Attributes

- [RodSystemParameters](#) **mParameters**
- `std::string` **mNameCable**

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

- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.cpp](#)

4.22 RSE::Core::RodSystemParameters Struct Reference

Parameters of a rod system.

```
#include <rodsystem.h>
```

Public Attributes

- `std::vector< double >` **distances**
Distance between supports, m.
- `double` **massPerLength**
Mass per length, kg.
- `double` **youngsModulus**
Youngs modulus, Pa.
- `double` **area**
Area of a cross-section, m².
- `double` **force**
Stretching force, N.
- `int` **numRods** = 0
Number of rods.

4.22.1 Detailed Description

Parameters of a rod system.

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

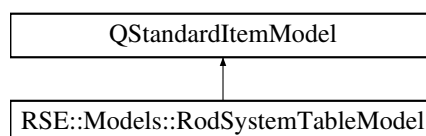
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.h](#)

4.23 RSE::Models::RodSystemTableModel Class Reference

Table model to set and represent data of a rod system.

```
#include <rodsystemtablemodel.h>
```

Inheritance diagram for RSE::Models::RodSystemTableModel:



Signals

- void **modified** ()

Public Member Functions

- **RodSystemTableModel** (QObject *pParent=nullptr)
- void **setRodSystem** ([Core::RodSystem](#) *pRodSystem)
Acquire the pointer to a rod system.
- void **updateContent** ()
Represent all data of a rod system.
- void **insertAfterSelected** ()
Insert fresh rows after selected ones.
- void **removeSelected** ()
Remove the selected rows.

Private Member Functions

- void **clearContent** ()
Remove all the objects created.
- void **setChangedData** (QStandardItem *pltem)
Set the changed distances between supports.

Private Attributes

- [Core::RodSystem](#) * **mpRodSystem** = nullptr

4.23.1 Detailed Description

Table model to set and represent data of a rod system.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/[rodsystemtablemodel.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/[rodsystemtablemodel.cpp](#)

4.24 RSE::Core::Array< T >::Row< U > Class Template Reference

Proxy class to acquire a row by index.

Public Member Functions

- **Row** (T *pData)
- T & **operator[]** (IndexType iCol)
- T const & **operator[]** (IndexType iCol) const
- T * **data** ()

Private Attributes

- `T * mpRow`

4.24.1 Detailed Description

```
template<typename T>
template<typename U>
class RSE::Core::Array< T >::Row< U >
```

Proxy class to acquire a row by index.

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

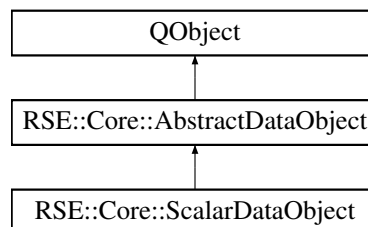
- `/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/array.h`

4.25 RSE::Core::ScalarDataObject Class Reference

Scalar data object.

```
#include <scalardataobject.h>
```

Inheritance diagram for RSE::Core::ScalarDataObject:



Public Member Functions

- **ScalarDataObject** (QString const &name)
Construct a scalar data object.
- **~ScalarDataObject** ()
Decrease a number of instances while being destroyed.
- **AbstractDataObject * clone** () const override
Clone a scalar data object.
- **DataItemType & addItem** (DataValueType key) override
Insert a new item into [ScalarDataObject](#).
- virtual void **import** (QTextStream &stream) override
Import a scalar data object from a file.

Static Public Member Functions

- static quint32 **numberInstances** ()

Static Private Attributes

- static quint32 **smNumInstances** = 0

Additional Inherited Members

4.25.1 Detailed Description

Scalar data object.

4.25.2 Member Function Documentation

4.25.2.1 addItem()

```
DataItemType & ScalarDataObject::addItem (
    DataValueType key ) [override], [virtual]
```

Insert a new item into [ScalarDataObject](#).

Implements [RSE::Core::AbstractDataObject](#).

4.25.2.2 clone()

```
AbstractDataObject * ScalarDataObject::clone ( ) const [override], [virtual]
```

Clone a scalar data object.

Implements [RSE::Core::AbstractDataObject](#).

4.25.2.3 import()

```
void ScalarDataObject::import (
    QTextStream & stream ) [override], [virtual]
```

Import a scalar data object from a file.

Implements [RSE::Core::AbstractDataObject](#).

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

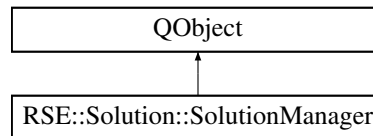
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.cpp](#)

4.26 RSE::Solution::SolutionManager Class Reference

Class to control the solution process.

```
#include <solutionmanager.h>
```

Inheritance diagram for RSE::Solution::SolutionManager:



Public Slots

- void **stopSolution** ()
Stop the solution process.

Signals

- void **outputSent** (QByteArray)
- void **rodSystemSolved** ()
- void **optimizationSolved** ()
- void **optimizationStepPerformed** ()

Public Member Functions

- **SolutionManager** (QString const &rootPath, QString const &relativeInputPath, QString const &relativeOutputPath)
- void **solveRodSystem** (Core::Project &project, [SolutionOptions](#) const &options)
Solve a rod system.
- void **solveOptimization** (Core::Project &project, [SolutionOptions](#) const &options)
Optimize viscosities of dampers as to damp selected set of modes.
- void **runVisualizer** ()
Run the visualizer of a rod system.

Private Member Functions

- void **processRodSystemStream** ()
Process the output of the rod system solver.
- void **processOptimizationStream** ()
Process the optimization output.
- void **runParserProcess** ()
Prepare data for the optimization process.
- void **writeOptimizationInput** (QString const &pathFile, int numDampers, [SolutionOptions](#) const &options)
Write the input data for optimization of viscosities.
- int **getRodSystemStatus** ()
Check if the solution process if finished.

Private Attributes

- QString **mRootPath**
- QString **mInputPath**
- QString **mOutputPath**
- QProcess * **mpRodSystemSolver** = nullptr
- QProcess * **mpOptimizationSolver** = nullptr

4.26.1 Detailed Description

Class to control the solution process.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[solutionmanager.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[solutionmanager.cpp](#)

4.27 RSE::Solution::SolutionOptions Class Reference

Public Member Functions

- **SolutionOptions** (int numCalcModes, int numDampModes, int stepModes, double tolTrunc)
- int **numCalcModes** () const
- int **numDampModes** () const
- int **stepModes** () const
- double **tolTrunc** () const
- void **setNumCalcModes** (int numCalcModes)
- void **setNumDampModes** (int numDampModes)
- void **setStepModes** (int stepModes)
- void **setTolTrunc** (double tolTrunc)

Private Attributes

- int **mNumCalcModes**
Number of computational modes.
- int **mNumDampModes**
Number of modes to be damped.
- int **mStepModes**
Step through computational modes.
- double **mTolTrunc**
Limit to truncate computational modes.

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

- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[solutionoptions.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[solutionoptions.cpp](#)

4.28 RSE::Core::Spans Struct Reference

Computed parameters of spans.

```
#include <rodsystem.h>
```

Public Member Functions

- **Spans** (int numRods)

Public Attributes

- `std::vector< double > u0`
Constant at the left end.
- `std::vector< double > uL`
Constant at the right end.
- `std::vector< double > L`
Length of a rod, m .
- `double projectedForce`
Projected stretching force, N .

4.28.1 Detailed Description

Computed parameters of spans.

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

- `/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.h`

4.29 KLP::StateFrame Struct Reference

Kinematic and dynamic quantities associated with a frame.

```
#include <framecollection.h>
```

Public Attributes

- `FloatFrameObject displacements [kNumDirections]`
- `FloatFrameObject rotations [kNumDirections]`
- `FloatFrameObject forces [kNumDirections]`
- `FloatFrameObject moments [kNumDirections]`

4.29.1 Detailed Description

Kinematic and dynamic quantities associated with a frame.

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

- </home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/framecollection.h>

4.30 RSE::Core::Support Class Reference

Class to aggregate data of supports.

```
#include <support.h>
```

Public Member Functions

- **Support** (double longitudinalStiffness, double verticalStiffness)
- double **longitudinalStiffness** () const
- double **verticalStiffness** () const
- void **setLongitudinalStiffness** (double longitudinalStiffness)
- void **setVerticalStiffness** (double verticalStiffness)

Private Attributes

- double **mLongitudinalStiffness**
Longitudinal stiffness (1), N/m.
- double **mVerticalStiffness**
Vertical stiffness (2), N/m.

4.30.1 Detailed Description

Class to aggregate data of supports.

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

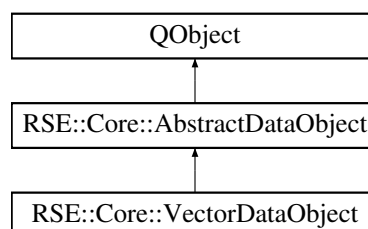
- </home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.h>
- </home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.cpp>

4.31 RSE::Core::VectorDataObject Class Reference

Vector data object.

```
#include <vectordataobject.h>
```

Inheritance diagram for RSE::Core::VectorDataObject:



Public Member Functions

- **VectorDataObject** (QString const &name)
Construct a vector data object.
- **~VectorDataObject** ()
Decrease a number of instances while being destroyed.
- **AbstractDataObject** * **clone** () const override
Clone a vector data object.
- **DataItemType** & **addItem** (DataValueType key) override
Insert a new item into [VectorDataObject](#).
- virtual void **import** (QTextStream &stream) override
Import a vector data object from a file.

Static Public Member Functions

- static quint32 **numberInstances** ()

Static Private Attributes

- static quint32 **smNumInstances** = 0

Additional Inherited Members

4.31.1 Detailed Description

Vector data object.

4.31.2 Member Function Documentation

4.31.2.1 addItem()

```
DataItemType & VectorDataObject::addItem (
    DataValueType key ) [override], [virtual]
```

Insert a new item into [VectorDataObject](#).

Implements [RSE::Core::AbstractDataObject](#).

4.31.2.2 clone()

```
AbstractDataObject * VectorDataObject::clone ( ) const [override], [virtual]
```

Clone a vector data object.

Implements [RSE::Core::AbstractDataObject](#).

4.31.2.3 import()

```
void VectorDataObject::import (
    QTextStream & stream ) [override], [virtual]
```

Import a vector data object from a file.

Implements [RSE::Core::AbstractDataObject](#).

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

- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.cpp](#)

Chapter 5

File Documentation

5.1 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/central/doublespinboxitemdelegate.cpp File Reference

DoubleSpinBoxItemDelegate.

```
#include <QDoubleSpinBox>
#include "doublespinboxitemdelegate.h"
```

5.1.1 Detailed Description

DoubleSpinBoxItemDelegate.

Author

Date

2022

5.2 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/central/doublespinboxitemdelegate.h File Reference

DoubleSpinBoxItemDelegate.

```
#include <QStyledItemDelegate>
```

Classes

- class [RSE::Models::DoubleSpinBoxItemDelegate](#)
Class to specify how table values can be edited.

5.2.1 Detailed Description

DoubleSpinBoxItemDelegate.

Author

Date

2022

5.3 doublespinboxitemdelegate.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef DOUBLESPINBOXITEMDELEGATE_H
9 #define DOUBLESPINBOXITEMDELEGATE_H
10
11 #include <QStyledItemDelegate>
12
13 namespace RSE::Models
14 {
15
16
17 class DoubleSpinBoxItemDelegate : public QStyledItemDelegate
18 {
19 public:
20     DoubleSpinBoxItemDelegate(QObject* parent = nullptr);
21     QWidget* createEditor(QWidget* parent, const QStyleOptionViewItem& option, const QModelIndex& index)
22         const override;
23     void setEditorData(QWidget* pEditor, const QModelIndex& index) const override;
24     void setModelData(QWidget* pEditor, QAbstractItemModel* pModel, const QModelIndex& index) const
25         override;
26     void updateEditorGeometry(QWidget* pEditor, const QStyleOptionViewItem& option, const QModelIndex&
27         index) const override;
28 };
29 #endif // DOUBLESPINBOXITEMDELEGATE_H

```

5.4 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/central/mainwindow.cpp File Reference

Definition of the MainWindow class.

```

#include <QVBoxLayout>
#include <QGridLayout>
#include <QLabel>
#include <QTextEdit>
#include <QDoubleSpinBox>
#include <QSpinBox>
#include <QSpacerItem>
#include <QSettings>
#include <QTableView>
#include <QHeaderView>
#include <QToolBar>
#include <QComboBox>
#include <QFileDialog>
#include <QMessageBox>

```

```
#include "DockManager.h"
#include "DockWidget.h"
#include "DockAreaWidget.h"
#include "ads_globals.h"
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "uiconstants.h"
#include "rodsystemtablemodel.h"
#include "doublespinboxitemdelegate.h"
#include "core/project.h"
#include "core/solutionoptions.h"
#include "core/solutionmanager.h"
#include "core/io.h"
#include "viewers/convergenceviewer.h"
#include "viewers/klpgraphviewer.h"
```

Functions

- QDoubleSpinBox * **createDoubleField** (double value, double maxValue=1e3, int numDecimals=3)
Create a field to input a floating-point number.
- QSpinBox * **createIntegerField** (int value, int maxValue=1000)
Create a field to input an integer.

5.4.1 Detailed Description

Definition of the MainWindow class.

Author

Pavel Lakiza

Date

July 2022

5.5 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/mainwindow.h File Reference

Declaration of the MainWindow class.

```
#include <QMainWindow>
```

Classes

- class [RSE::App::MainWindow](#)
Central window of the program.

5.5.1 Detailed Description

Declaration of the MainWindow class.

Author

Pavel Lakiza

Date

July 2022

5.6 mainwindow.h

[Go to the documentation of this file.](#)

```
1
2
3
4
5
6
7
8 #ifndef MAINWINDOW_H
9 #define MAINWINDOW_H
10
11 #include <QMainWindow>
12
13 QT_BEGIN_NAMESPACE
14 namespace Ui
15 {
16     class MainWindow;
17 }
18
19 class QSettings;
20 class QDoubleSpinBox;
21 class QSpinBox;
22 class QTableView;
23 class QTextEdit;
24 class QProcess;
25 class QComboBox;
26 QT_END_NAMESPACE
27
28 namespace ads
29 {
30     class CDockManager;
31     class CDockWidget;
32 }
33
34 namespace RSE
35 {
36 }
37
38 namespace Core
39 {
40     class Project;
41     class IO;
42 }
43
44 namespace Solution
45 {
46     class SolutionManager;
47     class SolutionOptions;
48 }
49
50 namespace Models
51 {
52     class RodSystemTableModel;
53     class DoubleSpinBoxItemDelegate;
54 }
55
56 namespace App
57 {
58 }
59
60 class MainWindow : public QMainWindow
61 {
62     Q_OBJECT
63
64 public:
65     MainWindow(QWidget* parent = nullptr);
66     ~MainWindow();
67     // Set parameters of a damper
68     void setMassCable(double value);
69     void setMassLoadedCable(double value);
```

```

68     void setWorkingLength(double value);
69     void setBouncerLength(double value);
70     void setSpringLength(double value);
71     void setSpringStiffness(double value);
72     // Set parameters of a rod system
73     void setCable(QString const& name);
74     void setForce(double value);
75     // Set parameters of supports
76     void setLongitudinalStiffness(double value);
77     void setVerticalStiffness(double value);
78     // Deal with projects
79     void createProject();
80     void openProjectDialog();
81     void openProject(QString const& pathFile);
82     void saveAsProject();
83     void saveProject();
84
85 private:
86     // Content
87     void initialize();
88     void createContent();
89     void createDefaultProject();
90     void createDefaultSolutionOptions();
91     void closeEvent(QCloseEvent* pEvent) override;
92     ads::CDockWidget* createDamperWidget();
93     ads::CDockWidget* createRodSystemWidget();
94     ads::CDockWidget* createSupportWidget();
95     ads::CDockWidget* createCalculationWidget();
96     ads::CDockWidget* createConsole();
97     // Signals & Slots
98     void specifyMenuConnections();
99
100 private slots:
101     // Settings
102     void saveSettings();
103     void restoreSettings();
104     // Recompute
105     void computeSpring();
106     void computeSpans();
107     // Controlling the solution process
108     void runRodSystemSolution();
109     void runOptimizationSolution();
110     void appendOutputData(QByteArray const& data);
111     void showConvergence();
112     void showResults();
113     // Set project data
114     void setProjectTitle();
115     void setProjectData();
116     void setSolutionOptions();
117     void setCurrentCable();
118     void setBlockedSignals(bool);
119     void aboutProgram();
120
121 private:
122     // GUI
123     Ui::MainWindow* mpUi;
124     ads::CDockManager* mpDockManager;
125     Models::RodSystemTableModel* mpRodSystemTableModel;
126     Models::DoubleSpinBoxItemDelegate* mpDoubleSpinBoxItemDelegate;
127     // Parameters of a damper
128     QDoubleSpinBox* mpMassCable;
129     QDoubleSpinBox* mpMassLoadedCable;
130     QDoubleSpinBox* mpWorkingLength;
131     QDoubleSpinBox* mpBouncerLength;
132     QDoubleSpinBox* mpSpringLength;
133     QDoubleSpinBox* mpSpringStiffness;
134     // Parameters of a rod system
135     QComboBox* mpNameCable;
136     QDoubleSpinBox* mpForce;
137     // Parameters of supports
138     QDoubleSpinBox* mpLongitudinalStiffness;
139     QDoubleSpinBox* mpVerticalStiffness;
140     // Options of computational process
141     QSpinBox* mpNumCalcModes;
142     QSpinBox* mpNumDampModes;
143     QSpinBox* mpStepModes;
144     QDoubleSpinBox* mpTolTrunc;
145     QTextEdit* mpConsole;
146     // Project
147     RSE::Core::Project* mpProject;
148     RSE::Solution::SolutionManager* mpSolutionManager;
149     RSE::Solution::SolutionOptions* mpSolutionOptions;
150     RSE::Core::IO* mpIO;
151     // Settings
152     QSharedPointer<QSettings> mpSettings;
153 };
154

```

```
155 }
156
157 }
158
159 #endif // MAINWINDOW_H
```

5.7 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/central/rodsystemtablemodel.cpp File Reference

Definition of the RodSystemTableModel class.

```
#include <QTableView>
#include "rodsystemtablemodel.h"
#include "core/rodsystem.h"
```

5.7.1 Detailed Description

Definition of the RodSystemTableModel class.

Author

Pavel Lakiza

Date

July 2022

5.8 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/central/rodsystemtablemodel.h File Reference

Declaration of the RodSystemTableModel class.

```
#include <QStandardItemModel>
```

Classes

- class [RSE::Models::RodSystemTableModel](#)
Table model to set and represent data of a rod system.

5.8.1 Detailed Description

Declaration of the RodSystemTableModel class.

Author

Pavel Lakiza

Date

July 2022

5.9 rodsystemtablemodel.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef RODSYSTEMTABLEMODEL_H
9 #define RODSYSTEMTABLEMODEL_H
10
11 #include <QStandardItemModel>
12
13 namespace RSE
14 {
15
16 namespace Core
17 {
18 class RodSystem;
19 }
20
21 namespace Models
22 {
23
24 class RodSystemTableModel : public QStandardItemModel
25 {
26     Q_OBJECT
27
28 public:
29     RodSystemTableModel(QObject* pParent = nullptr);
30     ~RodSystemTableModel() = default;
31     void setRodSystem(Core::RodSystem* pRodSystem);
32     void updateContent();
33     void insertAfterSelected();
34     void removeSelected();
35
36 signals:
37     void modified();
38
39 private:
40     void clearContent();
41     void setChangedData(QStandardItem* pItem);
42
43 private:
44     Core::RodSystem* mpRodSystem = nullptr;
45 };
46
47 }
48
49 }
50
51
52
53
54 #endif // RODSYSTEMTABLEMODEL_H

```

5.10 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/central/uiconstants.h File Reference

Graphical constants shared between several widgets.

```
#include <QString>
```

Variables

- const QString **RSE::UiConstants::Settings::skGeometry** = "geometry"
- const QString **RSE::UiConstants::Settings::skState** = "state"
- const QString **RSE::UiConstants::Settings::skDockingState** = "dockingState"

5.10.1 Detailed Description

Graphical constants shared between several widgets.

Author

Pavel Lakiza

Date

July 2022

5.11 uiconstants.h

[Go to the documentation of this file.](#)

```
1
2
3
4
5
6
7
8 #ifndef UICONSTANTS_H
9 #define UICONSTANTS_H
10
11 #include <QString>
12
13 namespace RSE::UiConstants
14 {
15
16 namespace Settings
17 {
18     const QString skGeometry      = "geometry";
19     const QString skState         = "state";
20     const QString skDockingState = "dockingState";
21 }
22
23 }
24
25 #endif // UICONSTANTS_H
```

5.12 /home/qinterfly/Library/Projects/Current/RodSystem↔ Estimator/src/core/abstractdataobject.cpp File Reference

Implementation of the AbstractDataObject class.

```
#include "abstractdataobject.h"
#include "constants.h"
```

5.12.1 Detailed Description

Implementation of the AbstractDataObject class.

Author

Pavel Lakiza

Date

July 2022

5.13 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/abstractdataobject.h File Reference

Declaration of the AbstractDataObject class.

```
#include <QObject>
#include <QString>
#include <QDataStream>
#include <map>
#include "array.h"
#include "aliasdata.h"
```

Classes

- class [RSE::Core::AbstractDataObject](#)

Data object which is designed in the way to be represented in a table easily.

Typedefs

- using **RSE::Core::DataItemType** = Array< DataValueType >
- using **RSE::Core::DataHolder** = std::multimap< DataKeyType, DataItemType >

Functions

- QDataStream & **RSE::Core::operator<<** (QDataStream &stream, AbstractDataObject const &obj)
Print a data object to a binary stream.

5.13.1 Detailed Description

Declaration of the AbstractDataObject class.

Author

Pavel Lakiza

Date

July 2022

5.14 abstractdataobject.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef ABSTRACTDATAOBJECT_H
9 #define ABSTRACTDATAOBJECT_H
10
11 #include <QObject>
12 #include <QString>
13 #include <QDataStream>
14 #include <map>
15 #include "array.h"
16 #include "aliasdata.h"
17
18 namespace RSE::Core
19 {
20
21 using DataItemType = Array<DataValueType>;
22 using DataHolder = std::multimap<DataKeyType, DataItemType>;
23
24 class AbstractDataObject : public QObject
25 {
26 public:
27     enum ObjectType
28     {
29         kScalar,
30         kVector,
31         kMatrix,
32         kSurface
33     };
34     AbstractDataObject(ObjectType type, QString const& name);
35     virtual ~AbstractDataObject() = 0;
36     virtual AbstractDataObject* clone() const = 0;
37     virtual DataItemType& addItem(DataKeyType key) = 0;
38     void removeItem(DataValueType key);
39     bool changeItemKey(DataKeyType oldKey, DataKeyType newKey, DataHolder* items = nullptr);
40     bool setArrayValue(DataKeyType key, DataValueType newValue, IndexType iRow = 0, IndexType iColumn = 0);
41     DataValueType arrayValue(DataKeyType key, IndexType iRow = 0, IndexType iColumn = 0);
42     std::vector<DataKeyType> keys() const;
43     quint32 numberItems() const { return mItems.size(); }
44     DataHolder const& getItems() { return mItems; }
45     DataIDType id() const { return mID; }
46     ObjectType type() const { return mkType; }
47     QString const& name() const { return mName; }
48     void setName(QString const& name) { mName = name; }
49     static DataIDType maxObjectID() { return smMaxObjectID; }
50     static void setMaxObjectID(DataIDType iMaxObjectID) { smMaxObjectID = iMaxObjectID; }
51     virtual void serialize(QDataStream& stream) const;
52     virtual void deserialize(QDataStream& stream);
53     friend QDataStream& operator<<(QDataStream& stream, AbstractDataObject const& obj);
54     virtual void import(QTextStream& stream) = 0;
55     void write(QTextStream& stream) const;
56
57 protected:
58     const ObjectType mkType;
59     QString mName;
60     DataIDType mID;
61     DataHolder mItems;
62
63 private:
64     static DataIDType smMaxObjectID;
65 };
66
67 inline QDataStream& operator<<(QDataStream& stream, AbstractDataObject const& obj)
68 {
69     obj.serialize(stream);
70     return stream;
71 }
72
73
74
75
76
77 #endif // ABSTRACTDATAOBJECT_H

```

5.15 /home/qinterfly/Library/Projects/Current/RodSystem← Estimator/src/core/aliasdata.h File Reference

Specification of data types used in a project.

```
#include <QtGlobal>
```

Typedefs

- using **RSE::Core::DataValueType** = double
- using **RSE::Core::DataKeyType** = double
- using **RSE::Core::DataIDType** = qint64

5.15.1 Detailed Description

Specification of data types used in a project.

Author

Pavel Lakiza

Date

May 2021

5.16 aliasdata.h

[Go to the documentation of this file.](#)

```
1
2
3 #ifndef ALIASDATA_H
4 #define ALIASDATA_H
5
6 #include <QtGlobal>
7
8 namespace RSE::Core
9 {
10
11     using DataValueType = double;
12     using DataKeyType = double;
13     using DataIDType = qint64;
14
15 }
16
17 #endif // ALIASDATA_H
```

5.17 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/core/array.cpp File Reference

Implementation of the Array class.

```
#include "array.h"
```

5.17.1 Detailed Description

Implementation of the Array class.

Author

Pavel Lakiza

Date

March 2021

5.18 /home/qinterfly/Library/Projects/Current/RodSystem← Estimator/src/core/array.h File Reference

Declaration of the Array class.

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

Classes

- class [RSE::Core::Array< T >](#)
Numerical array class.
- class [RSE::Core::Array< T >::Row< U >](#)
Proxy class to acquire a row by index.

Typedefs

- using **RSE::Core::IndexType** = quint32

Functions

- template<typename K >
QDebug **RSE::Core::operator**<< (QDebug stream, Array< K > &array)
Print all array values using the matrix format.
- template<typename K >
QDataStream & **RSE::Core::operator**<< (QDataStream &stream, Array< K > const &array)
Write an array to a binary stream.
- template<typename K >
QDataStream & **RSE::Core::operator**>> (QDataStream &stream, Array< K > &array)
Read an array from a stream.
- template<typename K >
QTextStream & **RSE::Core::operator**<< (QTextStream &stream, Array< K > const &array)
Write an array to a text stream.

5.18.1 Detailed Description

Declaration of the Array class.

Author

Pavel Lakiza

Date

July 2022

5.19 array.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef ARRAY_H
9 #define ARRAY_H
10
11 #include <QDebug>
12 #include "constants.h"
13
14 namespace RSE::Core
15 {
16
17 using IndexType = quint32;
18
19 template<typename T>
20 class Array
21 {
22 private:
23     template <typename U> class Row;
24
25 public:
26     Array(IndexType numRows = 0, IndexType numCols = 0);
27     Array(Array<T> const& another);
28     Array(Array<T>&& another);
29     ~Array();
30     T* data() { return mpData; }
31     void resize(IndexType numRows, IndexType numCols);
32     void removeColumn(IndexType iRemoveColumn);
33     void swapColumns(IndexType iFirstColumn, IndexType iSecondColumn);
34     void clear();
35     IndexType rows() const { return mNumRows; };
36     IndexType cols() const { return mNumCols; };
37     IndexType size() const { return mNumRows * mNumCols; }
38     Row<T> operator[](IndexType iRow) { return Row<T>(&mpData[mNumCols * iRow]); };
39     Row<T> operator[] (IndexType iRow) const { return Row<T>(&mpData[mNumCols * iRow]); };
40     Array& operator=(Array<T> const& another);
41     template<typename K> friend QDebug operator<<(QDebug stream, Array<K>& array);
42     template<typename K> friend QDataStream& operator<<(QDataStream& stream, Array<K> const& array);
43     template<typename K> friend QDataStream& operator>>(QDataStream& stream, Array<K>& array);
44     template<typename K> friend QTextStream& operator<<(QTextStream& stream, Array<K> const& array);
45     template<typename K> friend QTextStream& operator>>(QTextStream& stream, Array<K>& array);
46
47 private:
48     IndexType mNumRows;
49     IndexType mNumCols;
50     T* mpData = nullptr;
51     template <typename U>
52     class Row
53     {
54     public:
55         Row() = delete;
56         Row(T* pData) : mpRow(pData) { };
57         ~Row() { }
58         T& operator[] (IndexType iCol) { return mpRow[iCol]; }
59         T const& operator[] (IndexType iCol) const { return mpRow[iCol]; }
60         T* data() { return mpRow; }
61     private:
62         T* mpRow;
63     };
64 };
65
66 template<typename K>

```

```

72 inline QDebug operator<<(QDebug stream, Array<K>& array)
73 {
74     IndexType const& numRows = array.mNumRows;
75     IndexType const& numCols = array.mNumCols;
76     stream = stream.noquote();
77     stream << QString("Array size: %1 x %2").arg(QString::number(numRows), QString::number(numCols));
78     stream << Qt::endl;
79     for (IndexType iRow = 0; iRow != numRows; ++iRow)
80     {
81         for (IndexType jCol = 0; jCol != numCols; ++jCol)
82             stream << QString::number(array[iRow][jCol]);
83         stream << Qt::endl;
84     }
85     return stream;
86 }
87
89 template<typename K>
90 inline QDataStream& operator<<(QDataStream& stream, Array<K> const& array)
91 {
92     stream << array.mNumRows << array.mNumCols;
93     IndexType const& size = array.size();
94     for (IndexType i = 0; i != size; ++i)
95         stream << array.mpData[i];
96     return stream;
97 }
98
100 template<typename K>
101 inline QDataStream& operator>>(QDataStream& stream, Array<K>& array)
102 {
103     delete[] array.mpData;
104     stream >> array.mNumRows >> array.mNumCols;
105     IndexType const& size = array.size();
106     array.mpData = new K[size];
107     for (IndexType i = 0; i != size; ++i)
108         stream >> array.mpData[i];
109     return stream;
110 }
111
113 template<typename K>
114 inline QTextStream& operator<<(QTextStream& stream, Array<K> const& array)
115 {
116     IndexType const& numRows = array.mNumRows;
117     IndexType const& numCols = array.mNumCols;
118     for (IndexType iRow = 0; iRow != numRows; ++iRow)
119     {
120         for (IndexType jCol = 0; jCol != numCols; ++jCol)
121             stream << QString::number(array[iRow][jCol], 'g', RSE::Constants::kWritingPrecision);
122         stream << Qt::endl;
123     }
124     return stream;
125 }
126
127 }
128
129 #endif // ARRAY_H

```

5.20 /home/qinterfly/Library/Projects/Current/RodSystem← Estimator/src/core/constants.h File Reference

Computational constants.

Variables

- const double **RSE::Constants::kGravitationalAcceleration** = 9.8067
Gravitational acceleration, m/s².
- const int **RSE::Constants::kWritingPrecision** = 15
Number of digits to be written to a file.

5.20.1 Detailed Description

Computational constants.

Author

Pavel Lakiza

Date

July 2022

5.21 constants.h

[Go to the documentation of this file.](#)

```
1
2
3 #ifndef CONSTANTS_H
4 #define CONSTANTS_H
5
6 namespace RSE::Constants
7 {
8
9     const double kGravitationalAcceleration = 9.8067;
10
11     const int kWritingPrecision = 15;
12
13 }
14
15 #endif // CONSTANTS_H
```

5.22 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/core/damper.cpp File Reference

Definition of the Damper class.

```
#include "damper.h"
#include "constants.h"
```

5.22.1 Detailed Description

Definition of the Damper class.

Author

Pavel Lakiza

Date

July 2022

5.23 /home/qinterfly/Library/Projects/Current/RodSystem↔ Estimator/src/core/damper.h File Reference

Declaration the Damper class.

```
#include <QPair>
```

Classes

- class [RSE::Core::Damper](#)

Class to compute and collect properties of a damper.

5.23.1 Detailed Description

Declaration the Damper class.

Author

Pavel Lakiza

Date

July 2022

5.24 damper.h

[Go to the documentation of this file.](#)

```
1
2
3
4
5
6
7
8 #ifndef DAMPER_H
9 #define DAMPER_H
10
11 #include <QPair>
12
13 namespace RSE::Core
14 {
15
16
17 class Damper
18 {
19 public:
20     Damper(double massCable, double massLoadedCable, double workingLength, double bouncerLength,
21           double springLength = 0, double springStiffness = 0);
22     ~Damper() = default;
23     // Get parameteres of damper
24     double massCable() const { return mMassCable; }
25     double massLoadedCable() const { return mMassLoadedCable; }
26     double workingLength() const { return mWorkingLength; }
27     double bouncerLength() const { return mBouncerLength; }
28     double springLength() const { return mSpringLength; }
29     double springStiffness() const { return mSpringStiffness; }
30     // Set parameters of a damper
31     void setMassCable(double massCable) { mMassCable = massCable; }
32     void setMassLoadedCable(double massLoadedCable) { mMassLoadedCable = massLoadedCable; }
33     void setWorkingLength(double workingLength) { mWorkingLength = workingLength; }
34     void setBouncerLength(double bouncerLength) { mBouncerLength = bouncerLength; }
35     void setSpringLength(double springLength) { mSpringLength = springLength; }
36     void setSpringStiffness(double springStiffness) { mSpringStiffness = springStiffness; }
37     // Compute characteristics of a damper
38     void computeSpring();
39
40 private:
41     double mMassCable;
42     double mMassLoadedCable;
43     double mWorkingLength;
44     double mBouncerLength;
45     double mSpringLength = 0.0;
46     double mSpringStiffness = 0.0;
47 };
48
49
50
51
52
53
54
55
56
57 #endif // DAMPER_H
```


5.25 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.cpp File Reference

Definition of the DataBaseCables class.

```
#include <QFile>
#include <QTextStream>
#include "databasecables.h"
```

5.25.1 Detailed Description

Definition of the DataBaseCables class.

Author

Pavel Lakiza

Date

July 2022

5.26 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.h File Reference

Declaration of the DataBaseCables class.

```
#include <QString>
#include <unordered_map>
```

Classes

- struct [RSE::Core::Cable](#)
Mechanical properties of a cable.
- class [RSE::Core::DataBaseCables](#)
Aggregate data of cables.

5.26.1 Detailed Description

Declaration of the DataBaseCables class.

Author

Pavel Lakiza

Date

July 2022

5.27 databasecables.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef DATACABLES_H
9 #define DATACABLES_H
10
11 #include <QString>
12 #include <unordered_map>
13
14 namespace RSE::Core
15 {
16
17
18 struct Cable
19 {
20     std::string name;
21     double bendingStiffness;
22     double torsionalStiffness;
23     double massPerLength;
24     double youngsModulus;
25     double area;
26 };
27
28
29 class DataBaseCables
30 {
31 public:
32     DataBaseCables(QString const& directory, QString const& fileName);
33     ~DataBaseCables() = default;
34     std::vector<std::string> names() const;
35     Cable const& getItem(std::string const& name) const { return mData.at(name); }
36
37 private:
38     bool readDataBase(QString const& pathFile);
39
40 private:
41     std::unordered_map<std::string, Cable> mData;
42 };
43
44 #endif // DATACABLES_H

```

5.28 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/core/fileutilities.cpp File Reference

Definition of utilites targeted to working with files.

```

#include <QDebug>
#include <QString>
#include <QFile>
#include <QDir>
#include <QPair>
#include "fileutilities.h"

```

5.28.1 Detailed Description

Definition of utilites targeted to working with files.

Author

Pavel Lakiza

Date

July 2022

5.29 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/fileutilities.h File Reference

Declaration of utilities targeted to working with files.

```
#include <QSharedPointer>
#include "abstractdataobject.h"
```

Functions

- `QPair< Core::AbstractDataObject::ObjectType, QSharedPointer< QFile > > RSE::Utilities::File::getDataObjectFile` (QString const &path, QString const &fileName)
Retrieve a pair consisted of a data object file and its type.
- `QString RSE::Utilities::File::loadFileContent` (QString const &path)
Load all the content of a file.

5.29.1 Detailed Description

Declaration of utilities targeted to working with files.

Author

Pavel Lakiza

Date

July 2022

5.30 fileutilities.h

[Go to the documentation of this file.](#)

```
1
2
3 #ifndef FILEUTILITIES_H
4 #define FILEUTILITIES_H
5
6 #include <QSharedPointer>
7 #include "abstractdataobject.h"
8
9 class QFile;
10 class QString;
11
12 namespace RSE
13 {
14     namespace Utilities
15     {
16         namespace File
17         {
18             QPair<Core::AbstractDataObject::ObjectType, QSharedPointer<QFile>> getDataObjectFile(QString const& path,
19                 QString const& fileName);
20             QString loadFileContent(QString const& path);
21         }
22     }
23 }
24
25 #endif // FILEUTILITIES_H
```

5.31 [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.cpp](#) File Reference

Definition of the IO class.

```
#include <QFile>
#include <QFileInfo>
#include <QDir>
#include "io.h"
#include "project.h"
```

5.31.1 Detailed Description

Definition of the IO class.

Author

Pavel Lakiza

Date

July 2022

5.32 [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.h](#) File Reference

Declaration of the IO class.

```
#include <QString>
#include <QPair>
#include "solutionoptions.h"
```

Classes

- class [RSE::Core::IO](#)
Class to save the project and solution data.

Typedefs

- using [RSE::Core::IOPair](#) = QPair< Project *, [RSE::Solution::SolutionOptions](#) * >

5.32.1 Detailed Description

Declaration of the IO class.

Author

Pavel Lakiza

Date

July 2022

5.33 io.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef IO_H
9 #define IO_H
10
11 #include <QString>
12 #include <QPair>
13 #include "solutionoptions.h"
14
15 namespace RSE
16 {
17
18     namespace Core
19     {
20
21         class Project;
22         class DataBaseCables;
23
24         using IOPair = QPair<Project*, RSE::Solution::SolutionOptions*>;
25
26         class IO
27         {
28         public:
29             IO(QString const& lastPath);
30             ~IO() = default;
31             QString const& lastPath() const { return mLastPath; }
32             QString const& extension() const { return mkProjectExtension; }
33             void saveAs(QString const& pathFile, Project& project, Solution::SolutionOptions& options);
34             IOPair open(QString const& pathFile, DataBaseCables const& dataBaseCables);
35
36         private:
37             const QString mkProjectExtension = ".rse";
38             QString mLastPath;
39         };
40     };
41
42 }
43
44 }
45
46 #endif // IO_H

```

5.34 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/core/project.cpp File Reference

Definition of the Project class.

```

#include <QFile>
#include <QLocale>
#include "project.h"
#include "scalardataobject.h"
#include "vectordataobject.h"
#include "fileutilities.h"
#include "solutionoptions.h"

```

Functions

- void **clearDataObjects** (DataObjects &dataObjects)
Helper function to clear a container consisted of pointers to data objects.
- QStringList **readAllLines** (QString const &path, QString const &fileName)
Read all the lines from a file.
- void **replaceStringEntry** (QString &string, int numSkipEntries, QString subString)
Replace a substring after specified number of skips.
- void **writeAllLines** (QStringList const &lines, QString const &path, QString const &fileName)
Write all the lines to a file.

5.34.1 Detailed Description

Definition of the Project class.

Author

Pavel Lakiza

Date

July 2022

5.35 /home/qinterfly/Library/Projects/Current/RodSystem↔ Estimator/src/core/project.h File Reference

Declaration of the Project class.

```
#include <QString>
#include "abstractdataobject.h"
#include "damper.h"
#include "rodsystem.h"
#include "support.h"
#include "databasecables.h"
```

Classes

- class [RSE::Core::Project](#)

Typedefs

- using **RSE::Core::DataObjects** = std::vector< AbstractDataObject * >

5.35.1 Detailed Description

Declaration of the Project class.

Author

Pavel Lakiza

Date

July 2022

5.36 project.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef PROJECT_H
9 #define PROJECT_H
10
11 #include <QString>
12 #include "abstractdataobject.h"
13 #include "damper.h"
14 #include "rodsystem.h"
15 #include "support.h"
16 #include "databasecables.h"
17
18 namespace RSE
19 {
20
21     namespace Solution
22     {
23         class SolutionOptions;
24     }
25
26     namespace Core
27     {
28         class ScalarDataObject;
29         class VectorDataObject;
30     }
31
32     using DataObjects = std::vector<AbstractDataObject*>;
33
34     class Project
35     {
36     public:
37         Project(QString const& name, DataBaseCables dataBaseCables, Damper damper, RodSystem rodSystem,
38             Support support);
39         ~Project();
40         QString const& name() const { return mName; }
41         void setName(QString const& name) { mName = name; }
42         Damper& damper() { return mDamper; }
43         RodSystem& rodSystem() { return mRodSystem; }
44         Support& support() { return mSupport; }
45         DataBaseCables const& dataBaseCables() const { return mDataBaseCables; }
46         // IO
47         void readTemplateData(QString const& path);
48         void writeCalcData(QString const& path, Solution::SolutionOptions const& options);
49
50     private:
51         AbstractDataObject* addDataObject(AbstractDataObject::ObjectType type);
52         void importDataObjects(QString const& path, QString const& fileName);
53         void readProjectID(QString const& path);
54         // Modify data objects
55         void modifyScalarDataObjects();
56         void modifyVectorDataObjects(Spans const& spans);
57         // IO
58         void writeDataObjects(DataObjects const& dataObjects, QString const& path, QString const& fileName);
59         void writeRods(QString const& path, QString const& fileName);
60         void writeProgram(QString const& path, QString const& fileName, int numRods, int numCalcModes);
61
62     private:
63         QString mName;
64         Damper mDamper;
65         RodSystem mRodSystem;
66         Support mSupport;
67     };
68

```

```

70     DataBaseCables mDataBaseCables;
72     DataObjects mScalarDataObjects;
73     DataObjects mVectorDataObjects;
75     int mProjectID;
77     QStringList mRods;
79     QStringList mProgram;
81     static const QString skProjectExtension;
82 };
83
84 }
85
86 }
87
88 #endif // PROJECT_H

```

5.37 /home/qinterfly/Library/Projects/Current/RodSystem← Estimator/src/core/rodsystem.cpp File Reference

Definition of the RodSystem class.

```

#include <functional>
#include <stdlib.h>
#include <stdio.h>
#include <gsl/gsl_multiroots.h>
#include "rodsystem.h"
#include "constants.h"
#include "databasecables.h"

```

Typedefs

- using **IntegralFun** = std::function< double(double)>

Functions

- double **integrate** (IntegralFun const &f, double const &a, double const &b, int const &n)
Compute integral using the MidPoint rule.
- double **x1** (double u, double u0, double uL)
- double **x2** (double u, double u0, double uL)
- double **Q1** (double u0, double uL)
- double **Q2** (double u, double u0, double uL)
- double **Nf** (double u, double u0, double uL)
- double **LL** (double L, double u0, double uL, [RodSystemParameters](#) const *pParameters)
- double **projForce** (double u0, double uL, double L, [RodSystemParameters](#) const *pParameters)
- int **equations** (const gsl_vector *pState, void *pVoidParameters, gsl_vector *pFun)
System of equations.

5.37.1 Detailed Description

Definition of the RodSystem class.

Author

Pavel Lakiza

Date

July 2022

5.38 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.h File Reference

Declaration of the RodSystem class.

```
#include <QString>
#include <vector>
#include <gsl/gsl_vector.h>
```

Classes

- struct [RSE::Core::Spans](#)
Computed parameters of spans.
- struct [RSE::Core::RodSystemParameters](#)
Parameters of a rod system.
- class [RSE::Core::RodSystem](#)

5.38.1 Detailed Description

Declaration of the RodSystem class.

Author

Pavel Lakiza

Date

July 2022

5.39 rodsystem.h

[Go to the documentation of this file.](#)

```
1
2
3 #ifndef RODSYSTEM_H
4 #define RODSYSTEM_H
5
6 #include <QString>
7 #include <vector>
8 #include <gsl/gsl_vector.h>
9
10 namespace RSE::Core
11 {
12
13     struct Cable;
14
15     struct Spans
16     {
17         Spans(int numRods) : u0(numRods), uL(numRods), L(numRods) { }
18
19         std::vector<double> u0;
20         std::vector<double> uL;
21         std::vector<double> L;
22         double projectedForce;
23     };
24
25     struct RodSystemParameters
26     {
```

```

39     std::vector<double> distances;
40     double massPerLength;
41     double youngsModulus;
42     double area;
43     double force;
44     int numRods = 0;
45 };
46
47 class RodSystem
48 {
49 public:
50     RodSystem(std::vector<double> distances, Cable const& cable, double force);
51     // Get parameters of a system
52     std::vector<double> const& distances() const { return mParameters.distances; }
53     std::string const& nameCable() const { return mNameCable; }
54     double force() const { return mParameters.force; }
55     int numRods() const { return mParameters.numRods; }
56     double massPerLength() const { return mParameters.massPerLength; }
57     // Set parameters of a system
58     void setDistances(std::vector<double> const& distances);
59     void setCable(Cable const& cable);
60     void setForce(double force) { mParameters.force = force; };
61     // Compute parameters of spans
62     Spans computeSpans();
63
64 private:
65     RodSystemParameters mParameters;
66     std::string mNameCable;
67 };
68
69 #endif // RODSYSTEM_H

```

5.40 /home/qinterfly/Library/Projects/Current/RodSystem← Estimator/src/core/scalardataobject.cpp File Reference

Implementation of the ScalarDataObject class.

```
#include "scalardataobject.h"
```

5.40.1 Detailed Description

Implementation of the ScalarDataObject class.

Author

Pavel Lakiza

Date

July 2022

5.41 /home/qinterfly/Library/Projects/Current/RodSystem← Estimator/src/core/scalardataobject.h File Reference

Declaration of the ScalarDataObject class.

```
#include "abstractdataobject.h"
```

Classes

- class [RSE::Core::ScalarDataObject](#)
Scalar data object.

5.41.1 Detailed Description

Declaration of the ScalarDataObject class.

Author

Pavel Lakiza

Date

July 2022

5.42 scalardataobject.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef SCALARDATAOBJECT_H
9 #define SCALARDATAOBJECT_H
10
11 #include "abstractdataobject.h"
12
13 namespace RSE::Core
14 {
15
16
17 class ScalarDataObject : public AbstractDataObject
18 {
19 public:
20     ScalarDataObject(QString const& name);
21     ~ScalarDataObject();
22     AbstractDataObject* clone() const override;
23     DataItem* addItem(DataValueType key) override;
24     static quint32 numberInstances() { return smNumInstances; }
25     virtual void import(QTextStream& stream) override;
26 private:
27     static quint32 smNumInstances;
28 };
29
30 }
31
32 #endif // SCALARDATAOBJECT_H

```

5.43 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/core/solutionmanager.cpp File Reference

Definition of the SolutionManager class.

```

#include <QFileInfo>
#include <QDir>
#include "solutionoptions.h"
#include "solutionmanager.h"

```

5.43.1 Detailed Description

Definition of the SolutionManager class.

Author

Pavel Lakiza

Date

July 2022

5.44 [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionmanager.h](#) File Reference

Declaration of the SolutionManager class.

```
#include <QString>
#include <QProcess>
#include <QObject>
#include <QTextStream>
#include "project.h"
```

Classes

- class [RSE::Solution::SolutionManager](#)
Class to control the solution process.

5.44.1 Detailed Description

Declaration of the SolutionManager class.

Author

Pavel Lakiza

Date

July 2022

5.45 solutionmanager.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef SOLUTIONMANAGER_H
9 #define SOLUTIONMANAGER_H
10
11 #include <QString>
12 #include <QProcess>
13 #include <QObject>
14 #include <QTextStream>
15 #include "project.h"
16
17 namespace RSE::Solution
18 {
19
20 class SolutionOptions;
21
22 class SolutionManager : public QObject
23 {
24     Q_OBJECT
25
26 public:
27     SolutionManager(QString const& rootPath, QString const& relativeInputPath, QString const&
28         relativeOutputPath);
29     ~SolutionManager();
30     void solveRodSystem(Core::Project& project, SolutionOptions const& options);
31     void solveOptimization(Core::Project& project, SolutionOptions const& options);
32     void runVisualizer();
33
34 signals:
35     void outputSent(QByteArray);
36     void rodSystemSolved();
37     void optimizationSolved();
38     void optimizationStepPerformed();
39
40 public slots:
41     void stopSolution();
42
43 private:
44     void processRodSystemStream();
45     void processOptimizationStream();
46     void runParserProcess();
47     void writeOptimizationInput(QString const& pathFile, int numDampers, SolutionOptions const& options);
48     int getRodSystemStatus();
49
50 private:
51     QString mRootPath;
52     QString mInputPath;
53     QString mOutputPath;
54     QProcess* mpRodSystemSolver = nullptr;
55     QProcess* mpOptimizationSolver = nullptr;
56 };
57
58 }
59
60
61
62 #endif // SOLUTIONMANAGER_H

```

5.46 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/core/solutionoptions.cpp File Reference

Definition of the SolutionOptions class.

```
#include "solutionoptions.h"
```

5.46.1 Detailed Description

Definition of the SolutionOptions class.

Author

Pavel Lakiza

Date

July 2022

5.47 [/home/qinterfly/Library/Projects/Current/RodSystem←](#) Estimator/src/core/solutionoptions.h File Reference

Declaration of the SolutionOptions class.

Classes

- class [RSE::Solution::SolutionOptions](#)

5.47.1 Detailed Description

Declaration of the SolutionOptions class.

Author

Pavel Lakiza

Date

July 2022

5.48 solutionoptions.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef SOLUTIONOPTIONS_H
9 #define SOLUTIONOPTIONS_H
10
11 namespace RSE
12 {
13
14 namespace Solution
15 {
16
17 class SolutionOptions
18 {
19 public:
20     SolutionOptions() = default;
21     SolutionOptions(int numCalcModes, int numDampModes, int stepModes, double tolTrunc);
22     ~SolutionOptions() = default;
23     // Get parameters
24     int numCalcModes() const { return mNumCalcModes; }
25     int numDampModes() const { return mNumDampModes; }
26     int stepModes() const { return mStepModes; }
27     double tolTrunc() const { return mTolTrunc; }
28     // Set parameters
29     void setNumCalcModes(int numCalcModes) { mNumCalcModes = numCalcModes; }
30     void setNumDampModes(int numDampModes) { mNumDampModes = numDampModes; }
31     void setStepModes(int stepModes) { mStepModes = stepModes; }

```

```
32     void setTolTrunc(double tolTrunc) { mTolTrunc = tolTrunc; }
33
34 private:
35     int mNumCalcModes;
36     int mNumDampModes;
37     int mStepModes;
38     double mTolTrunc;
39 };
40
41
42
43
44
45
46
47
48
49
50 #endif // SOLUTIONOPTIONS_H
```

5.49 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.cpp File Reference ↩

Definition of the Support class.

```
#include "support.h"
```

5.49.1 Detailed Description

Definition of the Support class.

Author

Pavel Lakiza

Date

July 2022

5.50 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.h File Reference ↩

Declaration of the Support class.

Classes

- class [RSE::Core::Support](#)
Class to aggregate data of supports.

5.50.1 Detailed Description

Declaration of the Support class.

Author

Pavel Lakiza

Date

July 2022

5.51 support.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef SUPPORT_H
9 #define SUPPORT_H
10
11 namespace RSE::Core
12 {
13
14
15 class Support
16 {
17 public:
18     Support(double longitudinalStiffness, double verticalStiffness);
19     ~Support() = default;
20     // Get characteristics
21     double longitudinalStiffness() const { return mLongitudinalStiffness; }
22     double verticalStiffness() const { return mVerticalStiffness; }
23     // Set characteristics
24     void setLongitudinalStiffness(double longitudinalStiffness) { mLongitudinalStiffness =
        longitudinalStiffness; }
25     void setVerticalStiffness(double verticalStiffness) { mVerticalStiffness = verticalStiffness; }
26
27 private:
28     double mLongitudinalStiffness;
29     double mVerticalStiffness;
30 };
31
32
33
34 }
35
36 #endif // SUPPORT_H

```

5.52 /home/qinterfly/Library/Projects/Current/RodSystem← Estimator/src/core/vectordataobject.cpp File Reference

Implementation of the VectorDataObject class.

```
#include "vectordataobject.h"
```

Variables

- const IndexType **skNumElements** = 3

5.52.1 Detailed Description

Implementation of the VectorDataObject class.

Author

Pavel Lakiza

Date

July 2022

5.53 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.h File Reference

Declaration of the VectorDataObject class.

```
#include "abstractdataobject.h"
```

Classes

- class [RSE::Core::VectorDataObject](#)
Vector data object.

5.53.1 Detailed Description

Declaration of the VectorDataObject class.

Author

Pavel Lakiza

Date

July 2022

5.54 vectordataobject.h

[Go to the documentation of this file.](#)

```
1
2
3 8 #ifndef VECTORDATAOBJECT_H
4 9 #define VECTORDATAOBJECT_H
5 10
6 11 #include "abstractdataobject.h"
7 12
8 13 namespace RSE::Core
9 14 {
10 15
11 17 class VectorDataObject : public AbstractDataObject
12 18 {
13 19 public:
14 20     VectorDataObject(QString const& name);
15 21     ~VectorDataObject();
16 22     AbstractDataObject* clone() const override;
17 23     DataItemType& addItem(DataValueType key) override;
18 24     static quint32 numberInstances() { return smNumInstances; }
19 25     virtual void import(QTextStream& stream) override;
20 26
21 27 private:
22 28     static quint32 smNumInstances;
23 29 };
24 30
25 31 }
26 32
27 33 #endif // VECTORDATAOBJECT_H
```

5.55 /home/qinterfly/Library/Projects/Current/RodSystem↔ Estimator/src/klp/framecollection.h File Reference

Collection of the data associated with the specified frame.

```
#include "frameobject.h"
```

Classes

- struct [KLP::EnergyFrame](#)
Energy quantities associated with a frame.
- struct [KLP::StateFrame](#)
Kinematic and dynamic quantities associated with a frame.
- struct [KLP::FrameCollection](#)
Set of all quantities belonged to a frame.

Typedefs

- using **KLP::FloatFrameObject** = FrameObject< float >

Variables

- const int **KLP::kNumDirections** = 3

5.55.1 Detailed Description

Collection of the data associated with the specified frame.

Author

Pavel Lakiza

Date

July 2022

5.56 framecollection.h

[Go to the documentation of this file.](#)

```

1
9 #ifndef FRAMECOLLECTION_H
10 #define FRAMECOLLECTION_H
11
12 #include "frameobject.h"
13
14 namespace KLP
15 {
16
17     const int kNumDirections = 3;
18
19     using FloatFrameObject = FrameObject<float>;
20
21     struct EnergyFrame
22     {
23     {
24         FloatFrameObject kinetic;
25         FloatFrameObject potential;
26         FloatFrameObject full;
27     };
28
29     struct StateFrame
30     {
31     {
32         FloatFrameObject displacements[kNumDirections];
33         FloatFrameObject rotations[kNumDirections];
34         FloatFrameObject forces[kNumDirections];
35         FloatFrameObject moments[kNumDirections];
36     };
37
38     struct FrameCollection
39     {
40     {
41         int numRods;
42         FloatFrameObject parameter;
43         FloatFrameObject naturalLength;
44         FloatFrameObject accumulatedNaturalLength;
45         FloatFrameObject coordinates[kNumDirections];
46         StateFrame state;
47         StateFrame projectedState;
48         StateFrame firstDerivativeState;
49         StateFrame secondDerivativeState;
50         StateFrame errorState;
51         FloatFrameObject strain;
52         std::vector<StateFrame> modalStates;
53         FloatFrameObject frequencies;
54         EnergyFrame energy;
55     };
56
57     };
58
59     }
60
61     #endif // FRAMECOLLECTION_H

```

5.57 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/klp/frameobject.cpp File Reference

Definition of the FrameObject class.

```
#include "frameobject.h"
```

5.57.1 Detailed Description

Definition of the FrameObject class.

Author

Pavel Lakiza

Date

July 2022

5.58 /home/qinterfly/Library/Projects/Current/RodSystem↔ Estimator/src/klp/frameobject.h File Reference

Declaration of the FrameObject class.

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

Classes

- class [KLP::FrameObject< T >](#)

Functions

- `template<typename K >`
`QDebug KLP::operator<< (QDebug stream, FrameObject< K > &frameObject)`

5.58.1 Detailed Description

Declaration of the FrameObject class.

Author

Pavel Lakiza

Date

July 2022

5.59 frameobject.h

[Go to the documentation of this file.](#)

```
1
2
3
4
5
6
7
8 #ifndef FRAMEOBJECT_H
9 #define FRAMEOBJECT_H
10
11 #include <QDebug>
12 #include "frameobjectiterator.h"
13
14 namespace KLP
15 {
16
17 template <typename T>
18 class FrameObject
19 {
20 public:
21     using iterator = FrameObjectIterator<T>;
22
23 public:
24     FrameObject(T const* pData = nullptr, T normFactor = 1.0, qint64 size = 0, qint64 step = 1);
25     ~FrameObject() = default;
26     bool isEmpty() const { return !mpData; }
27     qint64 size() const { return mSize; }
28     iterator begin() { return iterator(&mpData[0], mNormFactor, mStep); }
29     iterator end() { return iterator(&mpData[mSize], mNormFactor, mStep); }
30     iterator operator[](int index) { return begin() + index; }
```

```

31     template<typename K> friend QDebug operator<<(QDebug stream, FrameObject<K>& frameObject);
32
33 private:
34     T const* mpData;
35     T mNormFactor;
36     qint64 mSize;
37     qint64 mStep;
38 };
39
40 template<typename K>
41 inline QDebug operator<<(QDebug stream, FrameObject<K>& frameObject)
42 {
43     stream = stream.noquote();
44     for (auto it = frameObject.begin(); it != frameObject.end(); ++it)
45         stream << QString::number(*it) << Qt::endl;
46     return stream;
47 }
48
49 }
50
51 #endif // FRAMEOBJECT_H

```

5.60 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.cpp File Reference

Definition of the FrameObjectIterator class.

```
#include "frameobjectiterator.h"
```

5.60.1 Detailed Description

Definition of the FrameObjectIterator class.

Author

Pavel Lakiza

Date

July 2022

5.61 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.h File Reference

Declaration of the FrameObjectIterator class.

```
#include <QtGlobal>
```

Classes

- class [KLP::FrameObjectIterator< T >](#)
Class to iterate through data of a record.

5.61.1 Detailed Description

Declaration of the FrameObjectIterator class.

Author

Pavel Lakiza

Date

July 2022

5.62 frameobjectiterator.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef FRAMEOBJECTITERATOR_H
9 #define FRAMEOBJECTITERATOR_H
10
11 #include <QtGlobal>
12
13 namespace KLP
14 {
15
16
17 template <typename T>
18 class FrameObjectIterator
19 {
20 public:
21     using self_type          = FrameObjectIterator<T>;
22     using iterator_category  = std::random_access_iterator_tag;
23     using difference_type    = std::ptrdiff_t;
24     using value_type         = T;
25     using pointer            = T const*;
26     using reference          = T const&;
27
28 public:
29     FrameObjectIterator(pointer pData, T normFactor, qint64 step);
30     // Access
31     value_type operator*() { return *mpData * mNormFactor; }
32     // Operators
33     self_type& operator++() { mpData += mStep; return *this; }
34     self_type operator++(int) { self_type temp = *this; ++(*this); return temp; }
35     self_type operator+(const difference_type& movement) { auto pOldData = mpData; mpData += movement *
mStep; self_type temp = *this; mpData = pOldData; return temp; }
36     difference_type operator-(const FrameObjectIterator& another) const { return mpData - another.mpData;
}
37     // Comparison
38     friend bool operator==(self_type const& first, self_type const& second) { return first.mpData ==
second.mpData; };
39     friend bool operator!=(self_type const& first, self_type const& second) { return !(first == second);
};
40
41 private:
42     pointer mpData;
43     T mNormFactor;
44     qint64 const mStep;
45 };
46
47 }
48
49 #endif // FRAMEOBJECTITERATOR_H

```

5.63 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/klp/index.h File Reference

Specification of a structure to index records.

```

#include <QtGlobal>
#include "types.h"
#include <vector>

```

Classes

- struct [KLP::IndexData](#)
Data of each record.
- struct [KLP::Index](#)
Structure to navigate through records.

5.63.1 Detailed Description

Specification of a structure to index records.

Author

Pavel Lakiza

Date

July 2022

5.64 index.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef INDEX_H
9 #define INDEX_H
10
11 #include <QtGlobal>
12 #include "types.h"
13 #include <vector>
14
15 namespace KLP
16 {
17
18     struct IndexData
19     {
20     public:
21         quint64 position = 0;
22         quint64 size = 0;
23         quint64 step = 1;
24         quint64 partSize = 0;
25     };
26
27     struct Index
28     {
29     public:
30         Index() { data.resize(RecordType::MAX_RECORD); }
31         std::vector<IndexData> data;
32         quint64 recordShift = 0;
33         quint64 relativeDataShift = 0;
34     };
35 }
36
37 #endif // INDEX_H

```

5.65 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/klp/result.cpp File Reference

Definition of the Result class.

```

#include <QFile>
#include <QDateTime>
#include "result.h"

```

5.65.1 Detailed Description

Definition of the Result class.

Author

Pavel Lakiza

Date

July 2022

5.66 /home/qinterfly/Library/Projects/Current/RodSystem← Estimator/src/klp/result.h File Reference

Declaration of the Result class.

```
#include <QString>
#include <QDateTime>
#include "index.h"
#include "framecollection.h"
```

Classes

- struct [KLP::ResultInfo](#)
- class [KLP::Result](#)
Class to aggregate all the records.

5.66.1 Detailed Description

Declaration of the Result class.

Author

Pavel Lakiza

Date

July 2022

5.67 result.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef RESULT_H
9 #define RESULT_H
10
11 #include <QString>
12 #include <QDateTime>
13 #include "index.h"
14 #include "framecollection.h"
15
16 namespace KLP
17 {
18
19 struct ResultInfo
20 {
21     QDateTime creationDateTime;
22     qint64 numTotalRecords;
23     qint64 numTimeRecords;
24     uint fileSize;
25     uint identifier;
26 };
27
28 class Result
29 {
30 public:
31     explicit Result(QString const& pathFile);
32     ~Result() = default;
33     bool isEmpty() const { return mContent.isEmpty(); }
34     std::vector<float> const& time() const { return mTime; }
35     QString const& pathFile() const { return mkPathFile; }
36     int numRods(qint64 iFrame) const;
37     ResultInfo info() const;
38     FloatFrameObject getFrameObject(qint64 iFrame, RecordType type, float normFactor = 1.0f, qint64 shift
= 0) const;
39     FrameCollection getFrameCollection(qint64 iFrame) const;
40     void update();
41
42 private:
43     bool read();
44     void buildIndex();
45     void setStateFrameData(StateFrame& state, RecordType type, qint64 iFrame, qint64 iStartData,
std::vector<float> const& normFactors) const;
46
47 private:
48     QString const mkPathFile;
49     QByteArray mContent;
50     std::vector<Index> mIndex;
51     qint64 mNumRecords;
52     std::vector<float> mTime;
53     char mNumBytesRod;
54 };
55
56 }
57
58 #endif // RESULT_H

```

5.68 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/klp/types.h File Reference

Specification of data types in a KLP file.

Enumerations

- enum [KLP::RecordType](#) {
R = 2 , **Xi** = 3 , **S** = 4 , **SS** = 5 ,
X1 = 6 , **X2** = 7 , **X3** = 8 , **U** = 9 ,
Ut = 10 , **Utt** = 11 , **EPS** = 12 , **UI** = 13 ,
BETA = 15 , **Qm** = 16 , **qm** = 17 , **AE** = 18 ,
MF = 19 , **MV** = 20 , **ND** = 21 , **FM** = 22 ,

```

ERR = 23 , MASS = 24 , RMASS = 25 , IP = 26 ,
CSM = 27 , CS = 28 , CSP = 29 , CSE = 30 ,
CSG = 31 , FI = 32 , FM2 = 33 , EM = 34 ,
EN = 35 , MAX_RECORD }

```

Types of records.

- enum `KLP::NondimensionalType` {
Time = 0 , **Displacement** = 1 , **Force** = 2 , **Moment** = 3 ,
DistributedForce = 7 , **DistributedMoment** = 8 , **Speed** = 9 , **Acceleration** = 10 ,
MAX_NONDIM }

Types of nondimensional coefficients.

5.68.1 Detailed Description

Specification of data types in a KLP file.

Author

Pavel Lakiza

Date

July 2022

5.69 types.h

[Go to the documentation of this file.](#)

```

1
2 #ifndef TYPES_H
3 #define TYPES_H
4
5 namespace KLP
6 {
7
8     enum RecordType
9     {
10         R      = 2, // Rods
11         Xi     = 3, // Parameter
12         S      = 4, // Natural length
13         SS     = 5, // Accumulated natural length
14         X1     = 6, // Coordinate X1
15         X2     = 7, // Coordinate X2
16         X3     = 8, // Coordinate X3
17         U      = 9, // State vector: [U1, U2, U3, w1, w2, w3, Q1, Q2, Q3, M1, M2, M3]
18         Utt    = 10, // First-order derivative of the state vector with respect to time
19         Ut     = 11, // Second-order derivative of the state vector with respect to time
20         EPS    = 12, // Tensile-compressive strain
21         U1     = 13, // Projected state vector: [U1L, U2L, U3L, w1, w2, w3, Q1L, Q2L, Q3L, M1L, M2L, M3L]
22         BETA   = 15, // Rotation matrix
23         Qm     = 16, // Loads
24         qm     = 17, // Distributed loads
25         AE     = 18, // Aerodynamic
26         MF     = 19, // Eigenfrequencies
27         MV     = 20, // Eigenvectors
28         ND     = 21, // Nondimensional coefficients [use NondimensionalType to navigate]
29         FM     = 22, // Finite element model
30         ERR    = 23, // Computational errors of the state vector
31         MASS   = 24, // Total mass and the center of gravity
32         RMASS  = 25, // Masses of rods
33         IP     = 26, // Cross sections
34         CSM    = 27, // ?
35         CS     = 28, // ?
36         CSP    = 29, // ?
37         CSE    = 30, // ?
38         CSG    = 31, // ?
39         FI     = 32, // Finite element image: set of coordinates (X, Y, Z) to plot lines
40         FM2    = 33, // ?
41
42     };
43
44 }
45
46 #endif
47

```

```

48     EM    = 34, // Effective masses
49     EN    = 35, // Energy
50     MAX_RECORD
51 };
52
54 enum NondimensionalType
55 {
56     Time           = 0,
57     Displacement   = 1,
58     Force          = 2,
59     Moment         = 3,
60     DistributedForce = 7,
61     DistributedMoment = 8,
62     Speed          = 9,
63     Acceleration   = 10,
64     MAX_NONDIM
65 };
66
67 }
68
69 #endif // TYPES_H

```

5.70 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/main/main.cpp File Reference ↩

Startup.

```

#include <QFile>
#include <QApplication>
#include "central/mainwindow.h"
#include "viewers/apputilities.h"

```

Functions

- `int main (int argc, char *argv[])`
Startup point.

5.70.1 Detailed Description

Startup.

Author

Pavel Lakiza

Date

July 2022

5.71 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/appaliases.h File Reference ↩

Declaration of aliases used in the application.

```

#include <vector>
#include <memory>

```

Typedefs

- using **RSE::Results** = std::vector< std::shared_ptr< [KLP::Result](#) > >

5.71.1 Detailed Description

Declaration of aliases used in the application.

Author

Pavel Lakiza

Date

July 2022

5.72 appaliases.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef APPALIASES_H
9 #define APPALIASES_H
10
11 #include <vector>
12 #include <memory>
13
14 namespace KLP
15 {
16     class Result;
17 }
18
19 namespace RSE
20 {
21
22     using Results = std::vector<std::shared_ptr<KLP::Result>;
23
24 }
25
26 #endif // APPALIASES_H

```

5.73 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/apputilities.cpp File Reference

Definition of utilites targeted to working with application data.

```

#include <QApplication>
#include <QFontDatabase>
#include <QScreen>
#include <QWidget>
#include "apputilities.h"
#include "fileutilities.h"

```

5.73.1 Detailed Description

Definition of utilites targeted to working with application data.

Author

Pavel Lakiza

Date

July 2022

5.74 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/apputilities.h File Reference

Declaration of utilities targeted to working with application data.

```
#include <QWidget>
```

Functions

- void **RSE::Utilities::App::setStyle** ()
Assign style features to the application.
- void **RSE::Utilities::App::moveToCenter** (QWidget *pChildWidget, QWidget *pLeadingWidget=nullptr)
Align the child widget to the center of the leading widget or, if not specified, to the screen center.

5.74.1 Detailed Description

Declaration of utilities targeted to working with application data.

Author

Pavel Lakiza

Date

July 2022

5.75 apputilities.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef APPUTILITIES_H
9 #define APPUTILITIES_H
10
11 #include <QWidget>
12
13 namespace RSE
14 {
15
16 namespace Utilities
17 {
18
19 namespace App
20 {
21
22 void setStyle();
23 void moveToCenter(QWidget* pChildWidget, QWidget* pLeadingWidget = nullptr);
24
25 }
26
27 }
28
29 }
30
31 #endif // APPUTILITIES_H

```

5.76 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/convergenceviewer.cpp File Reference

Definition of the ConvergenceViewer class.

```

#include <QVBoxLayout>
#include "convergenceviewer.h"

```

5.76.1 Detailed Description

Definition of the ConvergenceViewer class.

Author

Pavel Lakiza

Date

July 2022

5.77 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/convergenceviewer.h File Reference

Declaration of the ConvergenceViewer class.

```

#include <QWidget>
#include "array.h"
#include "qcustomplot.h"

```

Classes

- class [RSE::Viewers::ConvergenceViewer](#)
Class to represent convergence of viscosities.

5.77.1 Detailed Description

Declaration of the ConvergenceViewer class.

Author

Pavel Lakiza

Date

July 2022

5.78 convergenceviewer.h

[Go to the documentation of this file.](#)

```

1
2
3
4
5
6
7
8 #ifndef CONVERGENCEVIEWER_H
9 #define CONVERGENCEVIEWER_H
10
11 #include <QWidget>
12 #include "array.h"
13 #include "qcustomplot.h"
14
15 namespace RSE
16 {
17
18     namespace Viewers
19     {
20
21
22         class ConvergenceViewer : public QWidget
23         {
24         public:
25             ConvergenceViewer(QString const& pathFile, QWidget* pParent = nullptr);
26             ~ConvergenceViewer();
27             void plot();
28
29         private:
30             void initialize();
31             bool read();
32
33         private:
34             QString const mkPathFile;
35             QCustomPlot* mpFigure;
36             QStringList mAvailableColors;
37             QVector<QCPScatterStyle::ScatterShape> mAvailableShapes;
38             QVector<int> mCalcModes;
39             Core::Array<double> mDampingValues;
40         };
41
42     }
43
44 }
45
46 #endif // CONVERGENCEVIEWER_H

```

5.79 [/home/qinterfly/Library/Projects/Current/RodSystem](#) Estimator/src/viewers/klpgraphviewer.cpp File Reference

Definition of the KLPGraphViewer class.

```
#include <QSettings>
#include <QTextEdit>
#include "qcustomplot.h"
#include "DockManager.h"
#include "DockWidget.h"
#include "DockAreaWidget.h"
#include "ads_globals.h"
#include "central/uiconstants.h"
#include "klp/result.h"
#include "apputilities.h"
#include "klpgraphviewer.h"
#include "klpresultlistmodel.h"
```

5.79.1 Detailed Description

Definition of the KLPGraphViewer class.

Author

Pavel Lakiza

Date

July 2022

5.80 [/home/qinterfly/Library/Projects/Current/RodSystem](#) Estimator/src/viewers/klpgraphviewer.h File Reference

Declaration of the KLPGraphViewer class.

```
#include <QDialog>
#include "qcustomplot.h"
#include "appaliases.h"
```

Classes

- class [RSE::Viewers::KLPGraphViewer](#)
Class to graphically represent content of KLP output files.

5.80.1 Detailed Description

Declaration of the KLPGraphViewer class.

Author

Pavel Lakiza

Date

July 2022

5.81 klpgraphviewer.h

[Go to the documentation of this file.](#)

```

1
2  #ifndef KLPGRAPHVIEWER_H
3  #define KLPGRAPHVIEWER_H
4
5  #include <QDialog>
6  #include "qcustomplot.h"
7  #include "appaliases.h"
8
9  QT_BEGIN_NAMESPACE
10 class QSettings;
11 QT_END_NAMESPACE
12
13 namespace ads
14 {
15     class CDockManager;
16     class CDockWidget;
17 }
18
19 namespace KLP
20 {
21     class Result;
22     class ResultInfo;
23 }
24
25 namespace RSE
26 {
27 }
28
29 namespace Models
30 {
31     class KLPResultListModel;
32 }
33
34 namespace Viewers
35 {
36     class KLPGraphViewer : public QDialog
37     {
38     Q_OBJECT
39
40     public:
41         KLPGraphViewer(QString const& lastPath, QSettings& settings, QWidget* pParent = nullptr);
42         ~KLPGraphViewer();
43         // Deal with results
44         void openResultsDialog();
45         void openResults(QStringList const& locationFiles);
46
47     private:
48         // Content
49         void initialize();
50         void createContent();
51         ads::CDockWidget* createResultWidget();
52         ads::CDockWidget* createFigureWidget();
53         ads::CDockWidget* createConstructorWidget();
54         ads::CDockWidget* createPropertyWidget();
55         // Results
56         void processSelectedResults();
57         void showResultInfo(KLP::ResultInfo const& info);
58         // Settings
59         void saveSettings();
60     };
61 }

```

```

67     void restoreSettings();
68     void closeEvent(QCloseEvent* pEvent) override;
69
70 private:
71     QString mLastPath;
72     QSettings& mSettings;
73     // GUI
74     ads::CDockManager* mpDockManager = nullptr;
75     QCustomPlot* mpFigure;
76     QListView* mpListResults;
77     QTextEdit* mpTextInfo;
78     // Models
79     RSE::Models::KLPResultListModel* mpResultListModel;
80     // Data
81     Results mResults;
82 };
83
84 }
85
86 }
87
88 #endif // KLPGRAPHVIEWER_H

```

5.82 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/klpresultlistmodel.cpp File Reference

Definition of the KLPResultListModel class.

```

#include <QFileInfo>
#include <QListView>
#include "klpresultlistmodel.h"
#include "klp/result.h"

```

5.82.1 Detailed Description

Definition of the KLPResultListModel class.

Author

Pavel Lakiza

Date

July 2022

5.83 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/klpresultlistmodel.h File Reference

Declaration of the KLPResultListModel class.

```

#include <QStandardItemModel>
#include "appaliases.h"

```

Classes

- class [RSE::Models::KLPResultListModel](#)

5.83.1 Detailed Description

Declaration of the KLPResultListModel class.

Author

Pavel Lakiza

Date

July 2022

5.84 klpresultlistmodel.h

[Go to the documentation of this file.](#)

```
1
2
3
4
5
6
7
8 #ifndef KLPRESULTLISTMODEL_H
9 #define KLPRESULTLISTMODEL_H
10
11 #include <QStandardItemModel>
12 #include "appaliases.h"
13
14 namespace KLP
15 {
16     class Result;
17 }
18
19 namespace RSE
20 {
21
22     namespace Models
23     {
24
25         class KLPResultListModel : public QStandardItemModel
26         {
27             Q_OBJECT
28
29         public:
30             KLPResultListModel(Results& results, QObject* pParent = nullptr);
31             void updateContent();
32             void removeSelected();
33
34         private:
35             void clearContent();
36
37         private:
38             Results& mResults;
39     };
40
41 }
42
43 }
44
45 #endif // KLPRESULTLISTMODEL_H
```

