

## RodSystemEstimator

Generated by Doxygen 1.9.3



<b>1 Hierarchical Index</b>	<b>1</b>
1.1 Class Hierarchy	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Class Documentation</b>	<b>9</b>
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::Viewers::AbstractGraphData Class Reference	11
4.2.1 Member Function Documentation	12
4.2.1.1 getDataset()	12
4.2.1.2 sliceByIndex()	13
4.3 RSE::Core::Array< T > Class Template Reference	13
4.3.1 Detailed Description	14
4.4 RSE::Core::Cable Struct Reference	14
4.4.1 Detailed Description	15
4.5 RSE::Viewers::ConvergenceViewer Class Reference	15
4.5.1 Detailed Description	16
4.6 RSE::Core::Damper Class Reference	16
4.6.1 Detailed Description	17
4.7 RSE::Core::DataBaseCables Class Reference	17
4.7.1 Detailed Description	18
4.8 RSE::Models::DoubleSpinBoxItemDelegate Class Reference	18
4.8.1 Detailed Description	18
4.9 KLP::EnergyFrame Struct Reference	19
4.9.1 Detailed Description	19
4.10 KLP::FrameCollection Struct Reference	19
4.10.1 Detailed Description	20
4.11 KLP::FrameObject< T > Class Template Reference	20
4.12 KLP::FrameObjectIterator< T > Class Template Reference	21
4.12.1 Detailed Description	21
4.13 RSE::Viewers::Graph Class Reference	22
4.14 RSE::Viewers::GraphDataSlicer Class Reference	23
4.15 RSE::Models::GraphListModel Class Reference	24

4.16 KLP::Index Struct Reference . . . . .	25
4.16.1 Detailed Description . . . . .	25
4.17 KLP::IndexData Struct Reference . . . . .	25
4.17.1 Detailed Description . . . . .	26
4.18 RSE::Core::IO Class Reference . . . . .	26
4.18.1 Detailed Description . . . . .	26
4.19 RSE::Viewers::KinematicsGraphData Class Reference . . . . .	27
4.19.1 Detailed Description . . . . .	27
4.19.2 Member Function Documentation . . . . .	27
4.19.2.1 getDataset() . . . . .	28
4.19.2.2 type() . . . . .	28
4.20 RSE::Viewers::KLPGraphViewer Class Reference . . . . .	28
4.20.1 Detailed Description . . . . .	29
4.21 RSE::App::MainWindow Class Reference . . . . .	30
4.21.1 Detailed Description . . . . .	32
4.22 RSE::Core::Project Class Reference . . . . .	33
4.23 RSE::Viewers::PropertyTreeWidget Class Reference . . . . .	34
4.24 KLP::Result Class Reference . . . . .	36
4.24.1 Detailed Description . . . . .	37
4.25 KLP::ResultInfo Struct Reference . . . . .	37
4.26 RSE::Models::ResultListModel Class Reference . . . . .	37
4.27 RSE::Core::RodSystem Class Reference . . . . .	38
4.28 RSE::Core::RodSystemParameters Struct Reference . . . . .	39
4.28.1 Detailed Description . . . . .	39
4.29 RSE::Models::RodSystemTableModel Class Reference . . . . .	39
4.29.1 Detailed Description . . . . .	40
4.30 RSE::Core::Array< T >::Row< U > Class Template Reference . . . . .	40
4.30.1 Detailed Description . . . . .	41
4.31 RSE::Core::ScalarDataObject Class Reference . . . . .	41
4.31.1 Detailed Description . . . . .	42
4.31.2 Member Function Documentation . . . . .	42
4.31.2.1 addItem() . . . . .	42
4.31.2.2 clone() . . . . .	42
4.31.2.3 import() . . . . .	42
4.32 RSE::Solution::SolutionManager Class Reference . . . . .	43
4.32.1 Detailed Description . . . . .	44
4.33 RSE::Solution::SolutionOptions Class Reference . . . . .	44
4.34 RSE::Viewers::SpaceTimeGraphData Class Reference . . . . .	45
4.34.1 Detailed Description . . . . .	45
4.34.2 Member Function Documentation . . . . .	45
4.34.2.1 getDataset() . . . . .	46
4.34.2.2 type() . . . . .	46

4.35 RSE::Core::Spans Struct Reference . . . . .	46
4.35.1 Detailed Description . . . . .	47
4.36 KLP::StateFrame Struct Reference . . . . .	47
4.36.1 Detailed Description . . . . .	47
4.37 RSE::Core::Support Class Reference . . . . .	47
4.37.1 Detailed Description . . . . .	48
4.38 RSE::Core::VectorDataObject Class Reference . . . . .	48
4.38.1 Detailed Description . . . . .	49
4.38.2 Member Function Documentation . . . . .	49
4.38.2.1 addItem() . . . . .	49
4.38.2.2 clone() . . . . .	49
4.38.2.3 import() . . . . .	49
<b>5 File Documentation</b> . . . . .	<b>51</b>
5.1 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/doublespinboxitemdelegate.cpp File Reference . . . . .	51
5.1.1 Detailed Description . . . . .	51
5.2 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/doublespinboxitemdelegate.h File Reference . . . . .	51
5.2.1 Detailed Description . . . . .	52
5.3 doublespinboxitemdelegate.h . . . . .	52
5.4 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/mainwindow.cpp File Refer- ence . . . . .	52
5.4.1 Detailed Description . . . . .	53
5.5 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/mainwindow.h File Reference	53
5.5.1 Detailed Description . . . . .	54
5.6 mainwindow.h . . . . .	54
5.7 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/rodsystemtablemodel.cpp File Reference . . . . .	56
5.7.1 Detailed Description . . . . .	56
5.8 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/rodsystemtablemodel.h File Reference . . . . .	56
5.8.1 Detailed Description . . . . .	57
5.9 rodsystemtablemodel.h . . . . .	57
5.10 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/uiconstants.h File Reference	57
5.10.1 Detailed Description . . . . .	58
5.11 uiconstants.h . . . . .	58
5.12 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/abstractdataobject.cpp File Reference . . . . .	58
5.12.1 Detailed Description . . . . .	59
5.13 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/abstractdataobject.h File Ref- erence . . . . .	59
5.13.1 Detailed Description . . . . .	59
5.14 abstractdataobject.h . . . . .	60

5.15 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/aliasdata.h File Reference . .	60
5.15.1 Detailed Description . . . . .	61
5.16 aliasdata.h . . . . .	61
5.17 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/array.cpp File Reference . .	61
5.17.1 Detailed Description . . . . .	62
5.18 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/array.h File Reference . . . .	62
5.18.1 Detailed Description . . . . .	63
5.19 array.h . . . . .	63
5.20 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/constants.h File Reference .	64
5.20.1 Detailed Description . . . . .	65
5.21 constants.h . . . . .	65
5.22 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/damper.cpp File Reference .	65
5.22.1 Detailed Description . . . . .	65
5.23 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/damper.h File Reference . .	66
5.23.1 Detailed Description . . . . .	66
5.24 damper.h . . . . .	66
5.25 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.cpp File Reference . . . . .	67
5.25.1 Detailed Description . . . . .	67
5.26 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.h File Reference . . . . .	67
5.26.1 Detailed Description . . . . .	67
5.27 databasecables.h . . . . .	68
5.28 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/fileutilities.cpp File Reference .	68
5.28.1 Detailed Description . . . . .	68
5.29 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/fileutilities.h File Reference .	69
5.29.1 Detailed Description . . . . .	69
5.30 fileutilities.h . . . . .	69
5.31 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.cpp File Reference . . . .	70
5.31.1 Detailed Description . . . . .	70
5.32 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.h File Reference . . . . .	70
5.32.1 Detailed Description . . . . .	71
5.33 io.h . . . . .	71
5.34 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/numericalutilities.h File Reference . . . . .	71
5.34.1 Detailed Description . . . . .	72
5.35 numericalutilities.h . . . . .	72
5.36 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/project.cpp File Reference .	72
5.36.1 Detailed Description . . . . .	73
5.37 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/project.h File Reference . . .	73
5.37.1 Detailed Description . . . . .	74
5.38 project.h . . . . .	74
5.39 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.cpp File Reference	75

5.39.1 Detailed Description	75
5.40 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.h File Reference	76
5.40.1 Detailed Description	76
5.41 rodsystem.h	76
5.42 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.cpp File Reference	77
5.42.1 Detailed Description	77
5.43 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.h File Reference	77
5.43.1 Detailed Description	78
5.44 scalardataobject.h	78
5.45 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionmanager.cpp File Reference	78
5.45.1 Detailed Description	79
5.46 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionmanager.h File Reference	79
5.46.1 Detailed Description	79
5.47 solutionmanager.h	80
5.48 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionoptions.cpp File Reference	80
5.48.1 Detailed Description	80
5.49 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionoptions.h File Reference	81
5.49.1 Detailed Description	81
5.50 solutionoptions.h	81
5.51 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.cpp File Reference	82
5.51.1 Detailed Description	82
5.52 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.h File Reference	82
5.52.1 Detailed Description	83
5.53 support.h	83
5.54 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.cpp File Reference	83
5.54.1 Detailed Description	84
5.55 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.h File Reference	84
5.55.1 Detailed Description	84
5.56 vectordataobject.h	84
5.57 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/aliasklp.h File Reference	85
5.57.1 Detailed Description	85
5.58 aliasklp.h	85
5.59 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/framecollection.h File Reference	85
5.59.1 Detailed Description	86
5.60 framecollection.h	86
5.61 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobject.cpp File Reference	87

5.61.1 Detailed Description	87
5.62 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobject.h File Reference	87
5.62.1 Detailed Description	88
5.63 frameobject.h	88
5.64 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.cpp File Reference	89
5.64.1 Detailed Description	89
5.65 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.h File Reference	89
5.65.1 Detailed Description	89
5.66 frameobjectiterator.h	90
5.67 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/index.h File Reference	90
5.67.1 Detailed Description	91
5.68 index.h	91
5.69 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/result.cpp File Reference	91
5.69.1 Detailed Description	92
5.70 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/result.h File Reference	92
5.70.1 Detailed Description	92
5.71 result.h	93
5.72 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/types.h File Reference	93
5.72.1 Detailed Description	94
5.73 types.h	94
5.74 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/main/main.cpp File Reference	95
5.74.1 Detailed Description	95
5.75 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/abstractgraphdata.cpp File Reference	95
5.75.1 Detailed Description	96
5.76 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/abstractgraphdata.h File Reference	96
5.76.1 Detailed Description	96
5.77 abstractgraphdata.h	97
5.78 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/aliasviewers.h File Reference	97
5.78.1 Detailed Description	98
5.79 aliasviewers.h	98
5.80 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/apputilities.cpp File Reference	98
5.80.1 Detailed Description	99
5.81 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/apputilities.h File Reference	99
5.81.1 Detailed Description	99
5.82 apputilities.h	100
5.83 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.cpp File Reference	100
5.83.1 Detailed Description	100



5.84	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.h File Reference	100
5.84.1	Detailed Description	101
5.85	convergenceviewer.h	101
5.86	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graph.cpp File Reference	101
5.86.1	Detailed Description	102
5.87	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graph.h File Reference	102
5.87.1	Detailed Description	102
5.88	graph.h	103
5.89	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphdataslicer.cpp File Reference	104
5.89.1	Detailed Description	104
5.90	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphdataslicer.h File Reference	104
5.90.1	Detailed Description	104
5.91	graphdataslicer.h	105
5.92	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphlistmodel.cpp File Reference	105
5.92.1	Detailed Description	105
5.93	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphlistmodel.h File Reference	106
5.93.1	Detailed Description	106
5.94	graphlistmodel.h	106
5.95	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/kinematicsgraphdata.cpp File Reference	107
5.95.1	Detailed Description	107
5.96	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/kinematicsgraphdata.h File Reference	107
5.96.1	Detailed Description	108
5.97	kinematicsgraphdata.h	108
5.98	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.cpp File Reference	108
5.98.1	Detailed Description	109
5.99	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.h File Reference	109
5.99.1	Detailed Description	109
5.100	klpgraphviewer.h	110
5.101	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/propertytreewidget.cpp File Reference	111
5.101.1	Detailed Description	111
5.102	/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/propertytreewidget.h File Reference	111
5.102.1	Detailed Description	112
5.103	propertytreewidget.h	112

5.104	<a href="#">/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/resultlistmodel.cpp</a>	<a href="#">File Reference</a>	113
5.104.1	<a href="#">Detailed Description</a>		113
5.105	<a href="#">/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/resultlistmodel.h</a>	<a href="#">File Reference</a>	114
5.105.1	<a href="#">Detailed Description</a>		114
5.106	<a href="#">resultlistmodel.h</a>		114
5.107	<a href="#">/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/spacetimegraphdata.cpp</a>	<a href="#">File Reference</a>	115
5.107.1	<a href="#">Detailed Description</a>		115
5.108	<a href="#">/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/spacetimegraphdata.h</a>	<a href="#">File Reference</a>	115
5.108.1	<a href="#">Detailed Description</a>		115
5.109	<a href="#">spacetimegraphdata.h</a>		116

# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

RSE::Core::Array< T > . . . . .	13
RSE::Core::Array< double > . . . . .	13
RSE::Core::Cable . . . . .	14
RSE::Core::Damper . . . . .	16
RSE::Core::DataBaseCables . . . . .	17
KLP::EnergyFrame . . . . .	19
KLP::FrameCollection . . . . .	19
KLP::FrameObject< T > . . . . .	20
KLP::FrameObject< float > . . . . .	20
KLP::FrameObjectIterator< T > . . . . .	21
RSE::Viewers::Graph . . . . .	22
KLP::Index . . . . .	25
KLP::IndexData . . . . .	25
RSE::Core::IO . . . . .	26
RSE::Core::Project . . . . .	33
QMainWindow	
RSE::App::MainWindow . . . . .	30
QObject	
RSE::Core::AbstractDataObject . . . . .	9
RSE::Core::ScalarDataObject . . . . .	41
RSE::Core::VectorDataObject . . . . .	48
RSE::Solution::SolutionManager . . . . .	43
RSE::Viewers::AbstractGraphData . . . . .	11
RSE::Viewers::EnergyGraphData . . . . .	??
RSE::Viewers::KinematicsGraphData . . . . .	27
RSE::Viewers::SpaceTimeGraphData . . . . .	45
RSE::Viewers::GraphDataSlicer . . . . .	23
QStandardItemModel	
RSE::Models::GraphListModel . . . . .	24
RSE::Models::ResultListModel . . . . .	37
RSE::Models::RodSystemTableModel . . . . .	39
QStyledItemDelegate	
RSE::Models::DoubleSpinBoxItemDelegate . . . . .	18
QTreeWidgetItem	
RSE::Viewers::PropertyTreeWidget . . . . .	34

QWidget	
RSE::Viewers::ConvergenceViewer . . . . .	15
RSE::Viewers::KLPGraphViewer . . . . .	28
KLP::Result . . . . .	36
KLP::ResultInfo . . . . .	37
RSE::Core::RodSystem . . . . .	38
RSE::Core::RodSystemParameters . . . . .	39
RSE::Core::Array< T >::Row< U > . . . . .	40
RSE::Solution::SolutionOptions . . . . .	44
RSE::Core::Spans . . . . .	46
KLP::StateFrame . . . . .	47
RSE::Core::Support . . . . .	47

## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">RSE::Core::AbstractDataObject</a>	
Data object which is designed in the way to be represented in a table easily . . . . .	9
<a href="#">RSE::Viewers::AbstractGraphData</a> . . . . .	11
<a href="#">RSE::Core::Array&lt; T &gt;</a>	
Numerical array class . . . . .	13
<a href="#">RSE::Core::Cable</a>	
Mechanical properties of a cable . . . . .	14
<a href="#">RSE::Viewers::ConvergenceViewer</a>	
Class to represent convergence of viscosities . . . . .	15
<a href="#">RSE::Core::Damper</a>	
Class to compute and collect properties of a damper . . . . .	16
<a href="#">RSE::Core::DataBaseCables</a>	
Aggregate data of cables . . . . .	17
<a href="#">RSE::Models::DoubleSpinBoxItemDelegate</a>	
Class to specify how table values can be edited . . . . .	18
<a href="#">KLP::EnergyFrame</a>	
Energy quantities associated with a frame . . . . .	19
<a href="#">RSE::Viewers::EnergyGraphData</a>	
Class to deal with energy entities of KLP-data . . . . .	??
<a href="#">KLP::FrameCollection</a>	
Set of all quantities belonged to a frame . . . . .	19
<a href="#">KLP::FrameObject&lt; T &gt;</a> . . . . .	20
<a href="#">KLP::FrameObjectIterator&lt; T &gt;</a>	
Class to iterate through data of a record . . . . .	21
<a href="#">RSE::Viewers::Graph</a> . . . . .	22
<a href="#">RSE::Viewers::GraphDataSlicer</a> . . . . .	23
<a href="#">RSE::Models::GraphListModel</a> . . . . .	24
<a href="#">KLP::Index</a>	
Structure to navigate through records . . . . .	25
<a href="#">KLP::IndexData</a>	
Data of each record . . . . .	25
<a href="#">RSE::Core::IO</a>	
Class to save the project and solution data . . . . .	26
<a href="#">RSE::Viewers::KinematicsGraphData</a>	
Class to deal with kinematics of KLP-data . . . . .	27

<a href="#">RSE::Viewers::KLPGraphViewer</a>	
Class to graphically represent content of KLP output files	28
<a href="#">RSE::App::MainWindow</a>	
Central window of the program	30
<a href="#">RSE::Core::Project</a>	33
<a href="#">RSE::Viewers::PropertyTreeWidget</a>	34
<a href="#">KLP::Result</a>	
Class to aggregate all the records	36
<a href="#">KLP::ResultInfo</a>	37
<a href="#">RSE::Models::ResultListModel</a>	37
<a href="#">RSE::Core::RodSystem</a>	38
<a href="#">RSE::Core::RodSystemParameters</a>	
Parameters of a rod system	39
<a href="#">RSE::Models::RodSystemTableModel</a>	
Table model to set and represent data of a rod system	39
<a href="#">RSE::Core::Array&lt; T &gt;::Row&lt; U &gt;</a>	
Proxy class to acquire a row by index	40
<a href="#">RSE::Core::ScalarDataObject</a>	
Scalar data object	41
<a href="#">RSE::Solution::SolutionManager</a>	
Class to control the solution process	43
<a href="#">RSE::Solution::SolutionOptions</a>	44
<a href="#">RSE::Viewers::SpaceTimeGraphData</a>	
Class to deal with spacetime KLP-data	45
<a href="#">RSE::Core::Spans</a>	
Computed parameters of spans	46
<a href="#">KLP::StateFrame</a>	
Kinematic and dynamic quantities associated with a frame	47
<a href="#">RSE::Core::Support</a>	
Class to aggregate data of supports	47
<a href="#">RSE::Core::VectorDataObject</a>	
Vector data object	48

## 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/ <a href="#">doublespinboxitemdelegate.cpp</a>	
DoubleSpinBoxItemDelegate . . . . .	51
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ <a href="#">doublespinboxitemdelegate.h</a>	
DoubleSpinBoxItemDelegate . . . . .	51
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ <a href="#">mainwindow.cpp</a>	
Definition of the MainWindow class . . . . .	52
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ <a href="#">mainwindow.h</a>	
Declaration of the MainWindow class . . . . .	53
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ <a href="#">rodsystemtablemodel.cpp</a>	
Definition of the RodSystemTableModel class . . . . .	56
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ <a href="#">rodsystemtablemodel.h</a>	
Declaration of the RodSystemTableModel class . . . . .	56
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/central/ <a href="#">uiconstants.h</a>	
Graphical constants shared between several widgets . . . . .	57
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">abstractdataobject.cpp</a>	
Implementation of the AbstractDataObject class . . . . .	58
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">abstractdataobject.h</a>	
Declaration of the AbstractDataObject class . . . . .	59
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">aliasdata.h</a>	
Specification of data types used in a project . . . . .	60
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">array.cpp</a>	
Implementation of the Array class . . . . .	61
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">array.h</a>	
Declaration of the Array class . . . . .	62
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">constants.h</a>	
Computational constants . . . . .	64
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">damper.cpp</a>	
Definition of the Damper class . . . . .	65
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">damper.h</a>	
Declaration the Damper class . . . . .	66
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">databasecables.cpp</a>	
Definition of the DataBaseCables class . . . . .	67
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">databasecables.h</a>	
Declaration of the DataBaseCables class . . . . .	67
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/ <a href="#">fileutilities.cpp</a>	
Definition of utilites targeted to working with files . . . . .	68

/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/fileutilities.h	
Declaration of utilities targeted to working with files	69
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.cpp	
Definition of the IO class	70
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.h	
Declaration of the IO class	70
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/numericalutilities.h	
Declaration of utilities targeted to numerical operations	71
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/project.cpp	
Definition of the Project class	72
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/project.h	
Declaration of the Project class	73
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.cpp	
Definition of the RodSystem class	75
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.h	
Declaration of the RodSystem class	76
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.cpp	
Implementation of the ScalarDataObject class	77
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.h	
Declaration of the ScalarDataObject class	77
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionmanager.cpp	
Definition of the SolutionManager class	78
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionmanager.h	
Declaration of the SolutionManager class	79
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionoptions.cpp	
Definition of the SolutionOptions class	80
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionoptions.h	
Declaration of the SolutionOptions class	81
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.cpp	
Definition of the Support class	82
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.h	
Declaration of the Support class	82
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.cpp	
Implementation of the VectorDataObject class	83
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/vectordataobject.h	
Declaration of the VectorDataObject class	84
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/aliasklp.h	
Declaration of aliases used in KLP results	85
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/framecollection.h	
Collection of the data associated with the specified frame	85
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobject.cpp	
Definition of the FrameObject class	87
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobject.h	
Declaration of the FrameObject class	87
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.cpp	
Definition of the FrameObjectIterator class	89
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobjectiterator.h	
Declaration of the FrameObjectIterator class	89
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/index.h	
Specification of a structure to index records	90
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/result.cpp	
Definition of the Result class	91
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/result.h	
Declaration of the Result class	92
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/types.h	
Specification of data types in a KLP file	93
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/main/main.cpp	
Startup	95



/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/abstractgraphdata.cpp	
Definition of the AbstractGraphData class	95
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/abstractgraphdata.h	
Declaration of the AbstractGraphData class	96
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/aliasviewers.h	
Declaration of aliases used in viewers	97
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/apputilities.cpp	
Definition of utilites targeted to working with application data	98
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/apputilities.h	
Declaration of utilities targeted to working with application data	99
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.cpp	
Definition of the ConvergenceViewer class	100
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/convergenceviewer.h	
Declaration of the ConvergenceViewer class	100
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/energygraphdata.cpp	
Definition of the EnergyGraphData class	??
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/energygraphdata.h	
Declaration of the EnergyGraphData class	??
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graph.cpp	
Definition of the Graph class	101
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graph.h	
Declaration of the Graph class	102
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphdataslicer.cpp	
Definition of the GraphDataSlicer class	104
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphdataslicer.h	
Declaration of the GraphDataSlicer class	104
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphlistmodel.cpp	
Definition of the GraphListModel class	105
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphlistmodel.h	
Declaration of the GraphListModel class	106
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/kinematicsgraphdata.cpp	
Definition of the KinematisGraphData class	107
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/kinematicsgraphdata.h	
Declaration of the KinematisGraphData class	107
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.cpp	
Definition of the KLPGraphViewer class	108
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.h	
Declaration of the KLPGraphViewer class	109
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/propertytreewidget.cpp	
Definition of the PropertyTreeWidget class	111
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/propertytreewidget.h	
Declaration of the PropertyTreeWidget class	111
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/resultlistmodel.cpp	
Definition of the KLPResultListModel class	113
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/resultlistmodel.h	
Declaration of the KLPResultListModel class	114
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/spacetimegraphdata.cpp	
Definition of the SpaceTimeGraphData class	115
/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/spacetimegraphdata.h	
Declaration of the SpaceTimeGraphData class	115



## Chapter 4

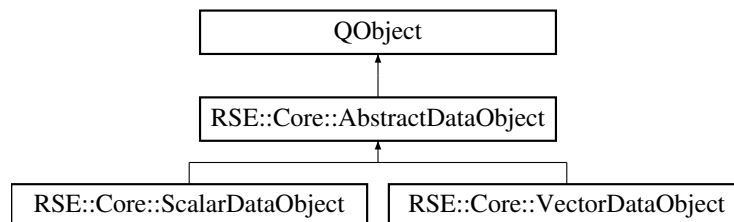
# Class Documentation

### 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]
```

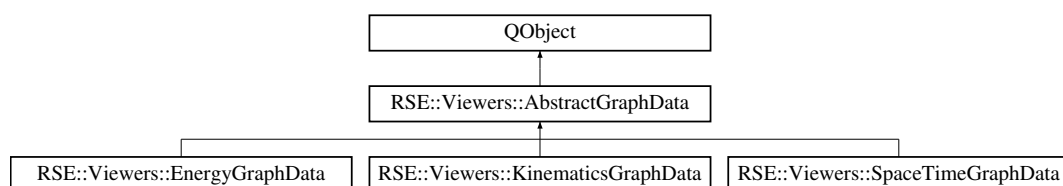
Implemented in [RSE::Core::ScalarDataObject](#), and [RSE::Core::VectorDataObject](#).

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::Viewers::AbstractGraphData Class Reference

Inheritance diagram for RSE::Viewers::AbstractGraphData:



## Public Types

- enum **Category** {  
  **cSpaceTime** , **cKinematics** , **cForce** , **cEnergy** ,  
  **cModal** , **cEstimation** }
- enum **Direction** { **dFirst** , **dSecond** , **dThird** , **dFull** }

## Public Member Functions

- **AbstractGraphData** (Category category, Direction direction)
- virtual GraphDataset **getDataset** ([KLP::FrameCollection](#) const &collection, qint64 sliceIndex=-1) const =0
- virtual int **type** () const =0
- Category **category** () const
- Direction **direction** () const

## Protected Member Functions

- GraphDataset **getAbsoluteData** ([KLP::FloatFrameObject](#) const components[], qint64 iStart, qint64 iEnd) const  
  *Compute the module of specified components.*
- GraphDataset **sliceByIndex** ([KLP::FloatFrameObject](#) const &component, qint64 index) const
- GraphDataset **sliceByDirectionAndIndex** ([KLP::FloatFrameObject](#) const components[], Direction direction, qint64 index) const  
  *Slice data through the specified direction and index.*

## Protected Attributes

- Category **mCategory**
- Direction **mDirection**

## 4.2.1 Member Function Documentation

### 4.2.1.1 getDataset()

```
virtual GraphDataset RSE::Viewers::AbstractGraphData::getDataset (
    KLP::FrameCollection const & collection,
    qint64 sliceIndex = -1 ) const [pure virtual]
```

Implemented in [RSE::Viewers::EnergyGraphData](#), [RSE::Viewers::KinematicsGraphData](#), and [RSE::Viewers::SpaceTimeGraphData](#).

#### 4.2.1.2 sliceByIndex()

```
GraphDataset AbstractGraphData::sliceByIndex (
    KLP::FloatFrameObject const & component,
    quint64 index ) const [protected]
```

Slice data by index

##### Returns

if index is positive, return the value located at the specified index. Otherwise, return the full set of values

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

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

## 4.3 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 >  
QDataStream & **operator**<< (QDataStream &stream, [Array](#)< K > const &array)  
*Write an array to a text stream.*

### 4.3.1 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](#)

## 4.4 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,  $m^2$ .*

### 4.4.1 Detailed Description

Mechanical properties of a cable.

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

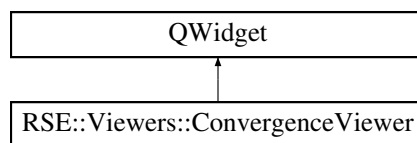
- `/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.h`

## 4.5 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**

### 4.5.1 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](#)

## 4.6 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.*

### 4.6.1 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](#)

## 4.7 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**

### 4.7.1 Detailed Description

Aggregate data of cables.

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

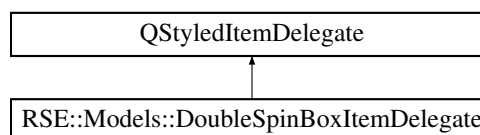
- </home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.h>
- </home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/databasecables.cpp>

## 4.8 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.*

### 4.8.1 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>

## 4.9 KLP::EnergyFrame Struct Reference

Energy quantities associated with a frame.

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

### Public Attributes

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

### 4.9.1 Detailed Description

Energy quantities associated with a frame.

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

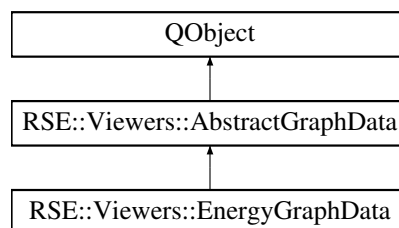
- </home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/framecollection.h>

## 4.10 RSE::Viewers::EnergyGraphData Class Reference

Class to deal with energy entities of KLP-data.

```
#include <energygraphdata.h>
```

Inheritance diagram for RSE::Viewers::EnergyGraphData:



### Public Types

- enum **EnergyType** { **eKinetic** , **ePotential** , **eFull** }

### Public Member Functions

- **EnergyGraphData** (EnergyType type, Direction direction=Direction::dFull)
- GraphDataset [getDataset](#) ([KLP::FrameCollection](#) const &collection, quint64 sliceIndex) const override  
*Retrieve the data of the specified type and direction from a given frame.*
- int [type](#) () const override

## Private Attributes

- EnergyType **mType**

## Additional Inherited Members

### 4.10.1 Detailed Description

Class to deal with energy entities of KLP-data.

### 4.10.2 Member Function Documentation

#### 4.10.2.1 `getDataset()`

```
GraphDataset EnergyGraphData::getDataset (
    KLP::FrameCollection const & collection,
    qint64 sliceIndex ) const [override], [virtual]
```

Retrieve the data of the specified type and direction from a given frame.

Implements [RSE::Viewers::AbstractGraphData](#).

#### 4.10.2.2 `type()`

```
int RSE::Viewers::EnergyGraphData::type ( ) const [inline], [override], [virtual]
```

Implements [RSE::Viewers::AbstractGraphData](#).

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

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

## 4.11 KLP::FrameCollection Struct Reference

Set of all quantities belonged to a frame.

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

## Public Attributes

- `int numRods`  
*Number of rods.*
- `float time`  
*Time.*
- `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.*

### 4.11.1 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`

## 4.12 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** () const
- iterator **end** () const
- iterator **operator[]** (int index) const

## Private Attributes

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

## Friends

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

## 4.13 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**
- qint64 const **mStep**

## Friends

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

### 4.13.1 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>

## 4.14 RSE::Viewers::Graph Class Reference

### Public Member Functions

- **Graph** (QString const &name)
- QString const & **name** () const
- GraphIDType **id** () const
- [AbstractGraphData](#) \*\* **data** ()
- bool **isData** (int iData) const
- bool **isDataSlicer** () const
- [GraphDataSlicer](#) & **dataSlicer** () const
- QVector< int > **indicesUniqueData** () const  
*Retrieve indices of assigned graph data.*
- int **indexTimeData** () const  
*Retrieve the index of the time data.*
- bool **isTimeData** () const
- QCPGraph::LineStyle **lineStyle** () const
- uint **lineWidth** () const
- QColor **color** () const
- QCPScatterStyle::ScatterShape **scatterShape** () const
- double **scatterSize** () const
- QStringList const & **axesLabels** () const
- void **setName** (QString const &name)
- void **setData** ([AbstractGraphData](#) \*pData, int iData)  
*Replace directional data with a new one.*

- void **setData** ([AbstractGraphData](#) \*pXData=nullptr, [AbstractGraphData](#) \*pYData=nullptr, [AbstractGraphData](#) \*pZData=nullptr)  
*Specify data for all axes.*
- void **eraseData** (int iData)
- bool **createDataSlicer** (GraphDataSlicer::SliceType type, PointerResult pResult)  
*Create an instance of object to slice one of datasets.*
- void **removeDataSlicer** ()  
*Remove data slicer if created.*
- void **setLineStyle** (QCPGraph::LineStyle const &lineStyle)
- void **setLineWidth** (uint lineWidth)
- void **setColor** (QColor const &color)
- void **setScatterShape** (QCPScatterStyle::ScatterShape const &scatterShape)
- void **setScatterSize** (double scatterSize)
- void **setAxisLabel** (QString const &label, int iData)
- void **setAxesLabels** (QStringList const &labels)

## Static Public Member Functions

- static GraphIDType **maxGraphID** ()

## Private Attributes

- QString **mName**
- GraphIDType **mID**
- [AbstractGraphData](#) \* **mpData** [KLP::kNumDirections] = {nullptr, nullptr, nullptr}
- [GraphDataSlicer](#) \* **mpDataSlicer** = nullptr
- QCPGraph::LineStyle **mLineStyle** = QCPGraph::lsLine
- uint **mLineWidth** = 1
- QColor **mColor** = Qt::blue
- QCPScatterStyle::ScatterShape **mScatterShape** = QCPScatterStyle::ssNone
- double **mScatterSize** = 5
- QStringList **mAxesLabels** = QStringList({" X", " Y", " Z"})

## Static Private Attributes

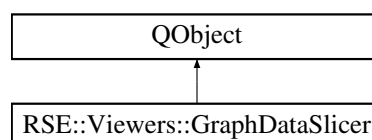
- static GraphIDType **smMaxGraphID** = 0

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

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

## 4.15 RSE::Viewers::GraphDataSlicer Class Reference

Inheritance diagram for RSE::Viewers::GraphDataSlicer:



## Public Types

- enum **SliceType** { **sdX** , **sdY** , **sdZ** }

## Public Member Functions

- **GraphDataSlicer** (SliceType type, [AbstractGraphData](#) const \*pData, KLP::PointerResult pResult)
- SliceType **type** () const
- qint64 **index** () const
- float **value** () const
- std::pair< qint64, qint64 > **limitsIndices** () const
- std::pair< float, float > **limitsValues** () const
- bool **isTime** () const
- void **setResult** (KLP::PointerResult pResult)  
*Specify a set of results.*
- void **setIndex** (qint64 index)  
*Determine the boundaries of the dataset.*
- float **setClosestValue** (float searchValue)  
*Find the closest value to the required one in the dataset.*

## Private Member Functions

- void **setLimits** ()  
*Specify limits of values and indices for slicing.*

## Private Attributes

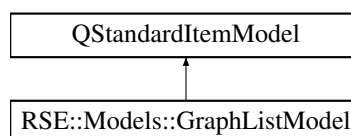
- SliceType const **mType**
- [AbstractGraphData](#) const \* **mpData**
- qint64 **mIndex**
- GraphDataset **mDataset**
- std::pair< qint64, qint64 > **mLimitsIndices**
- std::pair< float, float > **mLimitsValues**
- bool **mIsTime**

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

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

## 4.16 RSE::Models::GraphListModel Class Reference

Inheritance diagram for RSE::Models::GraphListModel:



## Public Member Functions

- **GraphListModel** (Viewers::MapGraphs &graphs, QObject \*pParent=nullptr)
- void **create** ()  
*Create a new default graph.*
- void **updateContent** ()  
*Create items linked to graphs.*
- void **removeSelected** ()  
*Remove selected graphs.*
- void **selectItem** (int iSelect=-1)  
*Select an item by index.*

## Private Member Functions

- void **clearContent** ()  
*Remove all the items created.*
- void **renameItem** (QStandardItem \*pItem)  
*Rename a graph after editing.*

## Private Attributes

- Viewers::MapGraphs & **mGraphs**

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

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

## 4.17 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.*

### 4.17.1 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`

## 4.18 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.*

### 4.18.1 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`

## 4.19 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**

### 4.19.1 Detailed Description

Class to save the project and solution data.

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

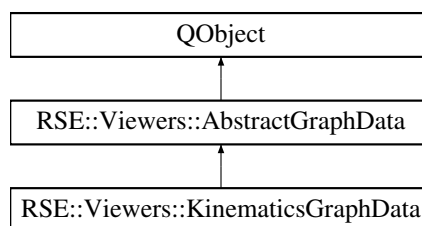
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/io.cpp](#)

## 4.20 RSE::Viewers::KinematicsGraphData Class Reference

Class to deal with kinematics of KLP-data.

```
#include <kinematicsgraphdata.h>
```

Inheritance diagram for RSE::Viewers::KinematicsGraphData:



## Public Types

- enum **KinematicsType** {  
**kStrain** , **kDisplacement** , **kRotation** , **kSpeed** ,  
**kAngularSpeed** , **kAcceleration** , **kAngularAcceleration** }

## Public Member Functions

- **KinematicsGraphData** (KinematicsType type, Direction direction=Direction::dFull)
- GraphDataset **getDataset** ([KLP::FrameCollection](#) const &collection, quint64 sliceIndex) const override  
*Retrieve the data of the specified type and direction from a given frame.*
- int **type** () const override

## Private Attributes

- KinematicsType **mType**

## Additional Inherited Members

### 4.20.1 Detailed Description

Class to deal with kinematics of KLP-data.

### 4.20.2 Member Function Documentation

#### 4.20.2.1 getDataset()

```
GraphDataset KinematicsGraphData::getDataset (
    KLP::FrameCollection const & collection,
    qint64 sliceIndex ) const [override], [virtual]
```

Retrieve the data of the specified type and direction from a given frame.

Implements [RSE::Viewers::AbstractGraphData](#).

#### 4.20.2.2 type()

```
int RSE::Viewers::KinematicsGraphData::type ( ) const [inline], [override], [virtual]
```

Implements [RSE::Viewers::AbstractGraphData](#).

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

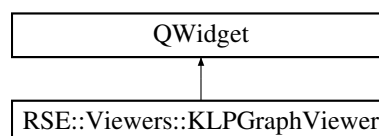
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/kinematicsgraphdata.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/kinematicsgraphdata.cpp](#)

## 4.21 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.*
- void **setGraphs** (MapGraphs &&graphs)  
*Replace the current set of graphs with the new one.*
- void **plot** ()  
*Plot the resulting set of graphs.*

## 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 **processSelectedGraphs** ()  
*Process selected graphs.*
- 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.*
- void **plotSurface** (PointerGraph const pGraph, PointerResult const pResult)  
*Represent plottable data as a surface.*
- void **plotCurve** (PointerGraph const pGraph, PointerResult const pResult)  
*Represent plottable data as a curve.*



## Private Attributes

- QString **mLastPath**
- QSettings & **mSettings**
- ads::CDockManager \* **mpDockManager** = nullptr
- QCustomPlot \* **mpFigure**
- QListView \* **mpListResults**
- QTextEdit \* **mpTextInfo**
- QListView \* **mpListGraphs**
- [RSE::Viewers::PropertyTreeWidget](#) \* **mpPropertyTreeWidget**
- [RSE::Models::ResultListModel](#) \* **mpResultListModel**
- [RSE::Models::GraphListModel](#) \* **mpGraphListModel**
- KLP::Results **mResults**
- MapGraphs **mGraphs**

### 4.21.1 Detailed Description

Class to graphically represent content of KLP output files.

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

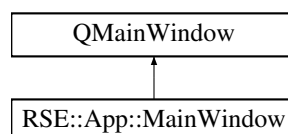
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.h](#)
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.cpp](#)

## 4.22 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**
- std::shared\_ptr< [Viewers::KLPGraphViewer](#) > **mpGraphViewer**
- 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**

### 4.22.1 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](#)

## 4.23 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

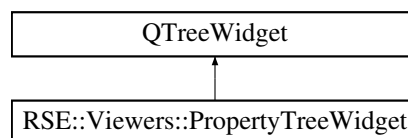
- static const QString **skProjectExtension**  
*[Project](#) extension.*

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](#)

## 4.24 RSE::Viewers::PropertyTreeWidget Class Reference

Inheritance diagram for RSE::Viewers::PropertyTreeWidget:



## Signals

- void **graphChanged** ()

## Public Member Functions

- **PropertyTreeWidget** (QWidget \*pParent=nullptr)
- void **setSelectedGraph** (PointerGraph pGraph)  
*Specify the single graph which properties need to be edited.*
- void **setSelectedResult** (PointerResult pResult)  
*Specify the single result to control slicing of data.*

## Private Member Functions

- void **initialize** ()  
*Intialize the widget and specify service data.*
- void **updateValues** ()  
*Represent values of properties.*
- void **createHierarchy** ()  
*Create a hierarchy of properties: keys and empty values.*
- QTreeWidgetItem \* **createDirectionalDataItem** (QString const &name)  
*Create a nested hierarchy of directional items.*
- void **createDataSlicerItem** ()  
*Create a nested hierarchy of items to slice data.*
- void **createAxesLabelsItem** ()  
*Create an item to specify labels for axes.*
- void **specifyConnections** ()  
*Enable widgets to communicate.*
- void **setBlockedSignals** (bool flag)  
*Set the blocked state of all widgets.*
- int **currentDataIndex** (int iData, int iChild)  
*Get current data index.*
- QString **currentDataText** (int iData, int iChild)  
*Get current data text.*
- void **setTypeWidget** (int iData)  
*Represent the type of the given graph data.*
- void **setColorItem** (QColor const &color)  
*Set the color of the graph.*
- void **resetSlicerWidgetsData** ()  
*Clear data of widgets associated with the data slicer.*
- void **updateSlicerWidgetsData** ()  
*Specify the data of the widget associated with the slicer.*
- void **setBlockedSlicerWidgetsSignals** (bool flag)  
*Set the enabled state of items to slice data.*
- void **assignGraphData** (int iData)  
*Assign new graph data.*
- void **assignVisualProperties** ()  
*Assign visual properties of current graph.*
- void **assignSlicer** ()  
*Make a new instance of the data slicer or delete the current one.*
- void **setSlicerIndex** (qint64 index)  
*Specify the leading index for slicing.*
- void **setSlicerValue** (float value)  
*Specifiy the leading value for slicing.*
- EnumData **getEnumData** (QMetaObject const &metaObject, std::string const &nameEnumerator)  
*Retrieve translated keys and icons from a meta object.*
- void **makeTranslationMap** ()  
*Specify translations for enum options.*

## Private Attributes

- PointerGraph **mpGraph** = nullptr
- PointerResult **mpResult** = nullptr
- QList< QTreeWidgetItem \* > **mDataItems**
- QTreeWidgetItem \* **mpDataSlicerItem**
- QMap< QString, QString > **mEnumTranslator**
- QComboBox \* **mpLineStyleWidget**
- QSpinBox \* **mpLineWidthWidget**
- QTreeWidgetItem \* **mpColorItem**
- QComboBox \* **mpScatterShapeWidget**
- QDoubleSpinBox \* **mpScatterSizeWidget**
- QTreeWidgetItem \* **mpAxesLabelsItem**

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

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

## 4.25 KLP::Result Class Reference

Class to aggregate all the records.

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

## Public Member Functions

- **Result** (QString const &pathFile)
- bool **isEmpty** () const
- QVector< double > const & **time** () const
- QString const & **pathFile** () const
- int **numRods** (qint64 iFrame) const  
*Get the number of rods associated with the requested frame.*
- qint64 **numTotalRecords** () const
- qint64 **numTimeRecords** () const
- [ResultInfo](#) **info** () const  
*Retrieve general information about a result file.*
- [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.*
- [FloatFrameObject](#) **getFrameObject** (qint64 iFrame, [RecordType](#) type, float normFactor=1.0f, qint64 shift=0) const  
*Get the object associated with the requested frame.*

## 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 **mNumTotalRecords**  
*Number of records.*
- QVector< double > **mTime**  
*Time array.*
- char **mNumBytesRod**  
*Number of bytes per rod.*

### 4.25.1 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>

## 4.26 KLP::ResultInfo Struct Reference

### Public Attributes

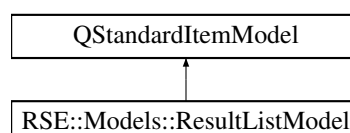
- QDateTime **creationDateTime**
- qint64 **numTotalRecords** = 0
- qint64 **numTimeRecords** = 0
- uint **fileSize** = 0
- uint **ID** = -1

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

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

## 4.27 RSE::Models::ResultListModel Class Reference

Inheritance diagram for RSE::Models::ResultListModel:





## Public Member Functions

- **ResultListModel** (KLP::Results &results, QObject \*pParent=nullptr)
- void **updateData** ()  
*Update results from files.*
- void **updateContent** ()  
*Create items linked to results.*
- void **removeSelected** ()  
*Remove selected results.*
- void **selectItem** (int iSelect=-1)  
*Select an item by index.*

## Private Member Functions

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

## Private Attributes

- KLP::Results & **mResults**

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

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

## 4.28 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](#)

## 4.29 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, m<sup>2</sup>.*
- `double` **force**  
*Stretching force, N.*
- `int` **numRods** = 0  
*Number of rods.*

### 4.29.1 Detailed Description

Parameters of a rod system.

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

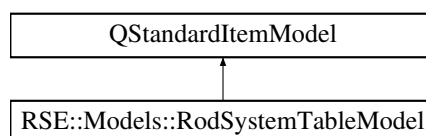
- [/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/rodsystem.h](#)

## 4.30 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

### 4.30.1 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](#)

## 4.31 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`

### 4.31.1 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:

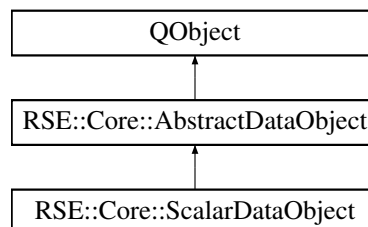
- `/home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/array.h`

## 4.32 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](#).*
- 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

### 4.32.1 Detailed Description

Scalar data object.

### 4.32.2 Member Function Documentation

#### 4.32.2.1 addItem()

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

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

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

#### 4.32.2.2 clone()

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

Clone a scalar data object.

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

#### 4.32.2.3 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:

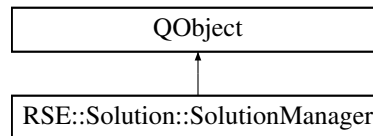
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[scalardataobject.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[scalardataobject.cpp](#)

## 4.33 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

### 4.33.1 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](#)

## 4.34 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:

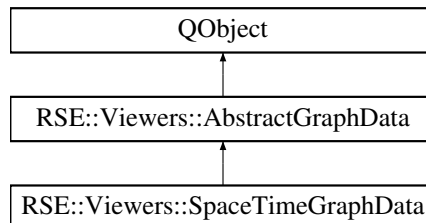
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[solutionoptions.h](#)
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/[solutionoptions.cpp](#)

## 4.35 RSE::Viewers::SpaceTimeGraphData Class Reference

Class to deal with spacetime KLP-data.

```
#include <spacetimegraphdata.h>
```

Inheritance diagram for RSE::Viewers::SpaceTimeGraphData:



### Public Types

- enum **SpaceTimeType** {  
**stTime** , **stParameter** , **stNaturalLength** , **stAccumulatedNaturalLength** ,  
**stCoordinate** }

### Public Member Functions

- **SpaceTimeGraphData** (SpaceTimeType type, Direction direction=Direction::dFull)
- GraphDataset **getDataset** ([KLP::FrameCollection](#) const &collection, quint64 sliceIndex) const override  
*Retrieve the data of the specified type and direction from a given frame.*
- int **type** () const override

### Private Attributes

- SpaceTimeType **mType**

### Additional Inherited Members

#### 4.35.1 Detailed Description

Class to deal with spacetime KLP-data.

#### 4.35.2 Member Function Documentation



4.35.2.1 `getDataset()`

```
GraphDataset SpaceTimeGraphData::getDataset (
    KLP::FrameCollection const & collection,
    qint64 sliceIndex ) const [override], [virtual]
```

Retrieve the data of the specified type and direction from a given frame.

Implements [RSE::Viewers::AbstractGraphData](#).

4.35.2.2 `type()`

```
int RSE::Viewers::SpaceTimeGraphData::type ( ) const [inline], [override], [virtual]
```

Implements [RSE::Viewers::AbstractGraphData](#).

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

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

## 4.36 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.*

### 4.36.1 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>

## 4.37 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]

### 4.37.1 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>

## 4.38 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.*

### 4.38.1 Detailed Description

Class to aggregate data of supports.

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

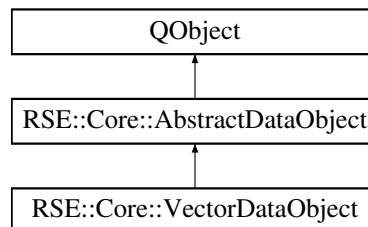
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.h
- /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/support.cpp

## 4.39 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](#).*
- 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

### 4.39.1 Detailed Description

Vector data object.

### 4.39.2 Member Function Documentation

#### 4.39.2.1 addItem()

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

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

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

#### 4.39.2.2 clone()

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

Clone a vector data object.

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

#### 4.39.2.3 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 Viewers
57 {
58     class KLPGraphViewer;
59 }
60
61 namespace App
62 {
63     class MainWindow : public QMainWindow
64     {
65     public:
66         Q_OBJECT
67     public:
```



```

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

```

```

155     RSE::Solution::SolutionOptions* mpSolutionOptions;
156     RSE::Core::IO* mpIO;
157     // Settings
158     QSharedPointer<QSettings> mpSettings;
159 };
160
161 }
162
163 }
164
165 #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
25 class RodSystemTableModel : public QStandardItemModel
26 {
27     Q_OBJECT
28
29 public:
30     RodSystemTableModel(QObject* pParent = nullptr);
31     ~RodSystemTableModel() = default;
32     void setRodSystem(Core::RodSystem* pRodSystem);
33     void updateContent();
34     void insertAfterSelected();
35     void removeSelected();
36
37 signals:
38     void modified();
39
40 private:
41     void clearContent();
42     void setChangedData(QStandardItem* pItem);
43
44 private:
45     Core::RodSystem* mpRodSystem = nullptr;
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
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/s<sup>2</sup>.*
- 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

August 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::Utilities::File
13 {
14     QPair<Core::AbstractDataObject::ObjectType, QSharedPointer<QFile>> getDataObjectFile(QString const& path,
15         QString const& fileName);
16     QString loadFileContent(QString const& path);
17 }
18
19 #endif // FILEUTILITIES_H
```

## 5.31 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/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/RodSystem↵ Estimator/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/numericalutilities.h File Reference

Declaration of utilities targeted to numerical operations.

```
#include <QtMath>
```

## Functions

- bool **RSE::Utilities::Numerical::fuzzyCompare** (double p1, double p2, double eps)  
*Compare two floating-point numbers.*

### 5.34.1 Detailed Description

Declaration of utilities targeted to numerical operations.

#### Author

Pavel Lakiza

#### Date

August 2022

## 5.35 numericalutilities.h

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

```

1
2
3
4
5
6
7
8 #ifndef NUMERICALUTILITIES_H
9 #define NUMERICALUTILITIES_H
10
11 #include <QtMath>
12
13 namespace RSE::Utilities::Numerical
14 {
15
16
17 inline bool fuzzyCompare(double p1, double p2, double eps)
18 {
19     return (qAbs(p1 - p2) <= eps * qMin(qAbs(p1), qAbs(p2)));
20 }
21
22 }
23
24 #endif // NUMERICALUTILITIES_H

```

## 5.36 /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 "scalarsdataobject.h"
#include "vectorsdataobject.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.*

### 5.36.1 Detailed Description

Definition of the Project class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.37 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/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 \* >

### 5.37.1 Detailed Description

Declaration of the Project class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.38 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

```

## 5.39 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/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.*

### 5.39.1 Detailed Description

Definition of the RodSystem class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.40 /home/qinterfly/Library/Projects/Current/RodSystem↔ Estimator/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](#)

### 5.40.1 Detailed Description

Declaration of the RodSystem class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.41 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     struct Cable;
13
14     struct Spans
15     {
16         Spans(int numRods) : u0(numRods), uL(numRods), L(numRods) { }
17
18         std::vector<double> u0;
19         std::vector<double> uL;
20         std::vector<double> L;
21         double projectedForce;
22     };
23
24     struct RodSystemParameters
25     {
```

```

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

```

## 5.42 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.cpp File Reference

Implementation of the ScalarDataObject class.

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

### 5.42.1 Detailed Description

Implementation of the ScalarDataObject class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.43 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/scalardataobject.h File Reference

Declaration of the ScalarDataObject class.

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

## Classes

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

### 5.43.1 Detailed Description

Declaration of the ScalarDataObject class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.44 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     DataItemType& addItem(DataValueType key) override;
24     static quint32 numberInstances() { return smNumInstances; }
25     void import(QTextStream& stream) override;
26
27 private:
28     static quint32 smNumInstances;
29 };
30
31 }
32
33 #endif // SCALARDATAOBJECT_H

```

## 5.45 [/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"

```



### 5.45.1 Detailed Description

Definition of the SolutionManager class.

Author

Pavel Lakiza

Date

July 2022

## 5.46 /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.*

### 5.46.1 Detailed Description

Declaration of the SolutionManager class.

Author

Pavel Lakiza

Date

July 2022

## 5.47 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

```

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

Definition of the SolutionOptions class.

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

### 5.48.1 Detailed Description

Definition of the SolutionOptions class.

**Author**

Pavel Lakiza

**Date**

July 2022

## 5.49 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/core/solutionoptions.h File Reference

Declaration of the SolutionOptions class.

### Classes

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

#### 5.49.1 Detailed Description

Declaration of the SolutionOptions class.

**Author**

Pavel Lakiza

**Date**

July 2022

## 5.50 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

```

## 5.51 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/core/support.cpp File Reference

Definition of the Support class.

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

### 5.51.1 Detailed Description

Definition of the Support class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.52 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/core/support.h File Reference

Declaration of the Support class.

### Classes

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

*Class to aggregate data of supports.*

### 5.52.1 Detailed Description

Declaration of the Support class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.53 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

```

## 5.54 /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

### 5.54.1 Detailed Description

Implementation of the VectorDataObject class.

Author

Pavel Lakiza

Date

July 2022

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

Declaration of the VectorDataObject class.

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

### Classes

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

### 5.55.1 Detailed Description

Declaration of the VectorDataObject class.

Author

Pavel Lakiza

Date

July 2022

## 5.56 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 16 class VectorDataObject : public AbstractDataObject
12 17 {
13 18 public:
14 19     VectorDataObject(QString const& name);
15 20     ~VectorDataObject();
16 21     AbstractDataObject* clone() const override;
17 22     DataItemType& addItem(DataValueType key) override;
18 23     static quint32 numberInstances() { return smNumInstances; }
19 24     void import(QTextStream& stream) override;
20 25
21 26 private:
22 27     static quint32 smNumInstances;
23 28 };
24 29
25 30
26 31 }
27 32
28 33 #endif // VECTORDATAOBJECT_H
```

## 5.57 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/aliasklp.h File Reference

Declaration of aliases used in KLP results.

```
#include <vector>
#include <memory>
```

### Typedefs

- using **KLP::PointerResult** = std::shared\_ptr< Result >
- using **KLP::Results** = std::vector< PointerResult >

### 5.57.1 Detailed Description

Declaration of aliases used in KLP results.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.58 aliasklp.h

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

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

## 5.59 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/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

### 5.59.1 Detailed Description

Collection of the data associated with the specified frame.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.60 framecollection.h

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

```

1
2
3
4
5
6
7
8 #ifndef FRAMECOLLECTION_H
9 #define FRAMECOLLECTION_H
10
11 #include "frameobject.h"
12
13 namespace KLP
14 {
15
16     const int kNumDirections = 3;
17
18     using FloatFrameObject = FrameObject<float>;
19
20     struct EnergyFrame
21     {
22     {
23         FloatFrameObject kinetic;
24         FloatFrameObject potential;
25         FloatFrameObject full;
26     };
27
28     struct StateFrame
29     {
30     {
31         FloatFrameObject displacements[kNumDirections];
32         FloatFrameObject rotations[kNumDirections];
33         FloatFrameObject forces[kNumDirections];
34         FloatFrameObject moments[kNumDirections];
35     };

```



```

36
37 struct FrameCollection
38 {
39     int numRods;
40     float time;
41     FloatFrameObject parameter;
42     FloatFrameObject naturalLength;
43     FloatFrameObject accumulatedNaturalLength;
44     FloatFrameObject coordinates[kNumDirections];
45     StateFrame state;
46     StateFrame projectedState;
47     StateFrame firstDerivativeState;
48     StateFrame secondDerivativeState;
49     StateFrame errorState;
50     FloatFrameObject strain;
51     std::vector<StateFrame> modalStates;
52     FloatFrameObject frequencies;
53     EnergyFrame energy;
54 };
55
56 }
57
58 #endif // FRAMECOLLECTION_H

```

## 5.61 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/klp/frameobject.cpp File Reference

Definition of the FrameObject class.

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

### 5.61.1 Detailed Description

Definition of the FrameObject class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.62 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/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 > const &frameObject)`

### 5.62.1 Detailed Description

Declaration of the FrameObject class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.63 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() const { return iterator(mpData, mNormFactor, mStep); }
29     iterator end() const { return begin() + mSize; }
30     iterator operator[](int index) const { return begin() + index; }
31     template<typename K> friend QDebug operator<<(QDebug stream, FrameObject<K> const& 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> const& 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

```

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

Definition of the FrameObjectIterator class.

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

### 5.64.1 Detailed Description

Definition of the FrameObjectIterator class.

Author

Pavel Lakiza

Date

July 2022

## 5.65 /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.*

### 5.65.1 Detailed Description

Declaration of the FrameObjectIterator class.

Author

Pavel Lakiza

Date

July 2022

## 5.66 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, quint64 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     quint64 const mStep;
45 };
46
47 }
48
49 #endif // FRAMEOBJECTITERATOR_H

```

## 5.67 /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.*

### 5.67.1 Detailed Description

Specification of a structure to index records.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.68 index.h

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

```

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

```

## 5.69 /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"

```

### 5.69.1 Detailed Description

Definition of the Result class.

Author

Pavel Lakiza

Date

July 2022

## 5.70 /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.*

### 5.70.1 Detailed Description

Declaration of the Result class.

Author

Pavel Lakiza

Date

July 2022

## 5.71 result.h

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

```

1
2 #ifndef RESULT_H
3 #define RESULT_H
4
5 #include <QString>
6 #include <QDateTime>
7 #include "index.h"
8 #include "framecollection.h"
9
10 namespace KLP
11 {
12
13 struct ResultInfo
14 {
15     QDateTime creationDateTime;
16     quint64 numTotalRecords = 0;
17     quint64 numTimeRecords = 0;
18     uint fileSize = 0;
19     uint ID = -1;
20 };
21
22 class Result
23 {
24 public:
25     explicit Result(QString const& pathFile);
26     ~Result() = default;
27     bool isEmpty() const { return mContent.isEmpty(); }
28     QVector<double> const& time() const { return mTime; }
29     QString const& pathFile() const { return mkPathFile; }
30     int numRods(quint64 iFrame) const;
31     quint64 numTotalRecords() const { return mNumTotalRecords; }
32     quint64 numTimeRecords() const { return mTime.size(); }
33     ResultInfo info() const;
34     FrameCollection getFrameCollection(quint64 iFrame) const;
35     void update();
36
37 private:
38     bool read();
39     void buildIndex();
40     void setStateFrameData(StateFrame& state, RecordType type, quint64 iFrame, quint64 iStartData,
41         std::vector<float> const& normFactors) const;
42     FloatFrameObject getFrameObject(quint64 iFrame, RecordType type, float normFactor = 1.0f, quint64 shift
43         = 0) const;
44
45 private:
46     QString const mkPathFile;
47     QByteArray mContent;
48     std::vector<Index> mIndex;
49     quint64 mNumTotalRecords;
50     QVector<double> mTime;
51     char mNumBytesRod;
52 };
53
54 #endif // RESULT_H

```

## 5.72 /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.*

### 5.72.1 Detailed Description

Specification of data types in a KLP file.

**Author**

Pavel Lakiza

**Date**

July 2022

## 5.73 types.h

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

```

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

```



```

46     FI    = 32, // Finite element image: set of coordinates (X, Y, Z) to plot lines
47     FM2   = 33, // ?
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

```

## 5.74 /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.*

#### 5.74.1 Detailed Description

Startup.

##### Author

Pavel Lakiza

##### Date

July 2022

## 5.75 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/abstractgraphdata.cpp File Reference

Definition of the AbstractGraphData class.

```

#include "abstractgraphdata.h"

```

### 5.75.1 Detailed Description

Definition of the AbstractGraphData class.

Author

Pavel Lakiza

Date

July 2022

## 5.76 [/home/qinterfly/Library/Projects/Current/RodSystem←](#) Estimator/src/viewers/abstractgraphdata.h File Reference

Declaration of the AbstractGraphData class.

```
#include <QObject>
#include "aliasviewers.h"
#include "framecollection.h"
```

### Classes

- class [RSE::Viewers::AbstractGraphData](#)

### 5.76.1 Detailed Description

Declaration of the AbstractGraphData class.

Author

Pavel Lakiza

Date

July 2022

## 5.77 abstractgraphdata.h

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

```

1
2
3
4
5
6
7
8 #ifndef ABSTRACTGRAPHDATA_H
9 #define ABSTRACTGRAPHDATA_H
10
11 #include <QObject>
12 #include "aliasviewers.h"
13 #include "framecollection.h"
14
15 namespace RSE::Viewers
16 {
17
18 class AbstractGraphData : public QObject
19 {
20     Q_OBJECT
21
22 public:
23     enum Category
24     {
25         cSpaceTime,
26         cKinematics,
27         cForce,
28         cEnergy,
29         cModal,
30         cEstimation
31     };
32     enum Direction
33     {
34         dFirst,
35         dSecond,
36         dThird,
37         dFull
38     };
39     Q_ENUM(Category)
40     Q_ENUM(Direction)
41     AbstractGraphData(Category category, Direction direction);
42     virtual ~AbstractGraphData() = 0;
43     virtual GraphDataset getDataset(KLP::FrameCollection const& collection, qint64 sliceIndex = -1) const
44     = 0;
45     virtual int type() const = 0;
46     Category category() const { return mCategory; }
47     Direction direction() const { return mDirection; }
48
49 protected:
50     GraphDataset getAbsoluteData(KLP::FloatFrameObject const components[], qint64 iStart, qint64 iEnd)
51     const;
52     GraphDataset sliceByIndex(KLP::FloatFrameObject const& component, qint64 index) const;
53     GraphDataset sliceByDirectionAndIndex(KLP::FloatFrameObject const components[], Direction direction,
54     qint64 index) const;
55
56 protected:
57     Category mCategory;
58     Direction mDirection;
59 };
60 #endif // ABSTRACTGRAPHDATA_H

```

## 5.78 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/aliasviewers.h File Reference

Declaration of aliases used in viewers.

```

#include <QtGlobal>
#include <QVector>
#include <map>

```

## Typedefs

- using **RSE::Viewers::GraphIDType** = qint64
- using **RSE::Viewers::GraphValueType** = double
- using **RSE::Viewers::GraphDataset** = QVector< GraphValueType >
- using **RSE::Viewers::PointerGraph** = std::shared\_ptr< Graph >
- using **RSE::Viewers::MapGraphs** = std::map< GraphIDType, PointerGraph >

### 5.78.1 Detailed Description

Declaration of aliases used in viewers.

#### Author

Pavel Lakiza

#### Date

August 2022

## 5.79 aliasviewers.h

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

```

1
2
3
4
5
6
7
8 #ifndef ALIASVIEWERS_H
9 #define ALIASVIEWERS_H
10
11 #include <QtGlobal>
12 #include <QVector>
13 #include <map>
14
15 namespace RSE::Viewers
16 {
17
18 class Graph;
19 using GraphIDType = qint64;
20 using GraphValueType = double;
21 using GraphDataset = QVector<GraphValueType>;
22 using PointerGraph = std::shared_ptr<Graph>;
23 using MapGraphs = std::map<GraphIDType, PointerGraph>;
24
25 }
26
27 #endif // ALIASVIEWERS_H

```

## 5.80 /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"

```

### 5.80.1 Detailed Description

Definition of utilites targeted to working with application data.

**Author**

Pavel Lakiza

**Date**

July 2022

## 5.81 /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.*

### 5.81.1 Detailed Description

Declaration of utilities targeted to working with application data.

**Author**

Pavel Lakiza

**Date**

July 2022

## 5.82 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

```

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

Definition of the ConvergenceViewer class.

```

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

```

### 5.83.1 Detailed Description

Definition of the ConvergenceViewer class.

**Author**

Pavel Lakiza

**Date**

July 2022

## 5.84 /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.*

### 5.84.1 Detailed Description

Declaration of the ConvergenceViewer class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.85 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

```

## 5.86 /home/qinterfly/Library/Projects/Current/RodSystem↔ Estimator/src/viewers/energygraphdata.cpp File Reference

Definition of the EnergyGraphData class.

```

#include "energygraphdata.h"
#include "klp/framecollection.h"

```

### 5.86.1 Detailed Description

Definition of the EnergyGraphData class.

Author

Pavel Lakiza

Date

August 2022

## 5.87 [/home/qinterfly/Library/Projects/Current/RodSystem←](#) Estimator/src/viewers/energygraphdata.h File Reference

Declaration of the EnergyGraphData class.

```
#include "abstractgraphdata.h"
```

### Classes

- class [RSE::Viewers::EnergyGraphData](#)  
*Class to deal with energy entities of KLP-data.*

### 5.87.1 Detailed Description

Declaration of the EnergyGraphData class.

Author

Pavel Lakiza

Date

August 2022



## 5.88 energygraphdata.h

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

```

1
2
3
4
5
6
7
8 #ifndef ENERGYGRAPHDATA_H
9 #define ENERGYGRAPHDATA_H
10
11 #include "abstractgraphdata.h"
12
13 namespace RSE::Viewers
14 {
15
16
17 class EnergyGraphData : public AbstractGraphData
18 {
19     Q_OBJECT
20
21 public:
22     enum EnergyType
23     {
24         eKinetic,
25         ePotential,
26         eFull
27     };
28     Q_ENUM(EnergyType)
29     EnergyGraphData(EnergyType type, Direction direction = Direction::dFull);
30     ~EnergyGraphData() = default;
31     GraphDataset getDataset(KLP::FrameCollection const& collection, qint64 sliceIndex) const override;
32     int type() const override { return mType; }
33
34 private:
35     EnergyType mType;
36 };
37
38 }
39
40 #endif // ENERGYGRAPHDATA_H

```

## 5.89 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/graph.cpp File Reference

Definition of the Graph class.

```

#include "graph.h"
#include "spacetimegraphdata.h"

```

### Typedefs

- using **DataSign** = QPair< AbstractGraphData::Category, int >

### 5.89.1 Detailed Description

Definition of the Graph class.

#### Author

Pavel Lakiza

#### Date

August 2022

## 5.90 /home/qinterfly/Library/Projects/Current/RodSystem↔ Estimator/src/viewers/graph.h File Reference

Declaration of the Graph class.

```
#include <QString>
#include <QColor>
#include "qcustomplot.h"
#include "abstractgraphdata.h"
#include "aliasviewers.h"
#include "aliasklp.h"
#include "graphdataslicer.h"
```

### Classes

- class [RSE::Viewers::Graph](#)

### 5.90.1 Detailed Description

Declaration of the Graph class.

#### Author

Pavel Lakiza

#### Date

August 2022

## 5.91 graph.h

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

```
1
2
3
4
5
6
7
8 #ifndef GRAPH_H
9 #define GRAPH_H
10
11 #include <QString>
12 #include <QColor>
13 #include "qcustomplot.h"
14 #include "abstractgraphdata.h"
15 #include "aliasviewers.h"
16 #include "aliasklp.h"
17 #include "graphdataslicer.h"
18
19 namespace RSE::Viewers
20 {
21
22 using KLP::PointerResult;
23
24 class Graph
25 {
26 public:
27     Graph(QString const& name);
28     ~Graph();
29     // Getters of data properties
30     static GraphIDType maxGraphID() { return smMaxGraphID; }
31     QString const& name() const { return mName; }
32     GraphIDType id() const { return mID; }
```

```

33     AbstractGraphData** data() { return mpData; }
34     bool isData(int iData) const { return mpData[iData]; }
35     bool isDataSlicer() const { return mpDataSlicer; }
36     GraphDataSlicer& dataSlicer() const { return *mpDataSlicer; }
37     QVector<int> indicesUniqueData() const;
38     int indexTimeData() const;
39     bool isTimeData() const { return indexTimeData() >= 0; }
40     // Getters of visual properties
41     QCPGraph::LineStyle lineStyle() const { return mLineStyle; }
42     uint lineWidth() const { return mLineWidth; }
43     QColor color() const { return mColor; }
44     QCPScatterStyle::ScatterShape scatterShape() const { return mScatterShape; }
45     double scatterSize() const { return mScatterSize; }
46     QStringList const& axesLabels() const { return mAxesLabels; }
47     // Setters of data properties
48     void setName(QString const& name) { mName = name; }
49     void setData(AbstractGraphData* pData, int iData);
50     void setData(AbstractGraphData* pData = nullptr, AbstractGraphData* pYData = nullptr,
AbstractGraphData* pZData = nullptr);
51     void eraseData(int iData) { setData(nullptr, iData); }
52     bool createDataSlicer(GraphDataSlicer::SliceType type, PointerResult pResult);
53     void removeDataSlicer();
54     // Setters of visual properties
55     void setLineStyle(QCPGraph::LineStyle const& lineStyle) { mLineStyle = lineStyle; }
56     void setLineWidth(uint lineWidth) { mLineWidth = lineWidth; }
57     void setColor(QColor const& color) { mColor = color; }
58     void setScatterShape(QCPScatterStyle::ScatterShape const& scatterShape) { mScatterShape =
scatterShape; }
59     void setScatterSize(double scatterSize) { mScatterSize = scatterSize; }
60     void setAxisLabel(QString const& label, int iData) { mAxesLabels[iData] = label; }
61     void setAxesLabels(QStringList const& labels) { mAxesLabels = labels; }
62 private:
63     QString mName;
64     GraphIDType mID;
65     // Data
66     AbstractGraphData* mpData[KLP::kNumDirections] = {nullptr, nullptr, nullptr};
67     GraphDataSlicer* mpDataSlicer = nullptr;
68     // Line options
69     QCPGraph::LineStyle mLineStyle = QCPGraph::lsLine;
70     uint mLineWidth = 1;
71     QColor mColor = Qt::blue;
72     // Scatter options
73     QCPScatterStyle::ScatterShape mScatterShape = QCPScatterStyle::ssNone;
74     double mScatterSize = 5;
75     // Axes labels
76     QStringList mAxesLabels = QStringList({" X", " Y", " Z"});
77
78 private:
79     static GraphIDType smMaxGraphID;
80 };
81
82
83
84
85 #endif // GRAPH_H

```

## 5.92 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphdataslicer.cpp File Reference

Definition of the GraphDataSlicer class.

```

#include "graphdataslicer.h"
#include "spacetimegraphdata.h"
#include "klp/result.h"

```

### 5.92.1 Detailed Description

Definition of the GraphDataSlicer class.

**Author**

Pavel Lakiza

**Date**

August 2022

## 5.93 [/home/qinterfly/Library/Projects/Current/RodSystem↵](#) Estimator/src/viewers/graphdataslicer.h File Reference

Declaration of the GraphDataSlicer class.

```
#include <QObject>
#include "aliasklp.h"
#include "aliasviewers.h"
```

### Classes

- class [RSE::Viewers::GraphDataSlicer](#)

#### 5.93.1 Detailed Description

Declaration of the GraphDataSlicer class.

**Author**

Pavel Lakiza

**Date**

August 2022

## 5.94 graphdataslicer.h

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

```
1
2
3
4
5
6
7
8 #ifndef GRAPHDATASLICER_H
9 #define GRAPHDATASLICER_H
10
11 #include <QObject>
12 #include "aliasklp.h"
13 #include "aliasviewers.h"
14
15 namespace RSE::Viewers
16 {
17
18     class AbstractGraphData;
19
20     class GraphDataSlicer : public QObject
21     {
22     public:
23         Q_OBJECT
24     public:
```

```

25     enum SliceType
26     {
27         sdX,
28         sdY,
29         sdZ
30     };
31     Q_ENUM(SliceType)
32     GraphDataSlicer(SliceType type, AbstractGraphData const* pData, KLP::PointerResult pResult);
33     // Getters
34     SliceType type() const { return mType; }
35     quint64 index() const { return mIndex; }
36     float value() const { return mDataset[mIndex]; }
37     std::pair<quint64, quint64> limitsIndices() const { return mLimitsIndices; }
38     std::pair<float, float> limitsValues() const { return mLimitsValues; }
39     bool isTime() const { return mIsTime; }
40     // Setters
41     void setResult(KLP::PointerResult pResult);
42     void setIndex(quint64 index);
43     float setClosestValue(float searchValue);
44
45 private:
46     void setLimits();
47
48 private:
49     SliceType const mType;
50     AbstractGraphData const* mpData;
51     quint64 mIndex;
52     GraphDataset mDataset;
53     std::pair<quint64, quint64> mLimitsIndices;
54     std::pair<float, float> mLimitsValues;
55     bool mIsTime;
56 };
57
58 }
59
60 #endif // GRAPHDATASLICER_H

```

## 5.95 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphlistmodel.cpp File Reference

Definition of the GraphListModel class.

```

#include <QListView>
#include "graphlistmodel.h"
#include "graph.h"

```

### 5.95.1 Detailed Description

Definition of the GraphListModel class.

Author

Pavel Lakiza

Date

July 2022

## 5.96 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/graphlistmodel.h File Reference

Declaration of the GraphListModel class.

```

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

```

## Classes

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

### 5.96.1 Detailed Description

Declaration of the GraphListModel class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.97 graphlistmodel.h

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

```

1
2
3
4
5
6
7
8 #ifndef GRAPHLISTMODEL_H
9 #define GRAPHLISTMODEL_H
10
11 #include <QStandardItemModel>
12 #include "aliasviewers.h"
13
14 namespace RSE
15 {
16
17 namespace Models
18 {
19
20 class GraphListModel : public QStandardItemModel
21 {
22     Q_OBJECT
23
24 public:
25     GraphListModel(Viewers::MapGraphs& graphs, QObject* pParent = nullptr);
26     void create();
27     void updateContent();
28     void removeSelected();
29     void selectItem(int iSelect = -1);
30
31 private:
32     void clearContent();
33     void renameItem(QStandardItem* pItem);
34
35 private:
36     Viewers::MapGraphs& mGraphs;
37 };
38
39 }
40
41 }
42
43 #endif // GRAPHLISTMODEL_H

```

## 5.98 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/kinematicsgraphdata.cpp File Reference

Definition of the KinematisGraphData class.

```

#include "kinematicsgraphdata.h"
#include "klp/framecollection.h"

```

### 5.98.1 Detailed Description

Definition of the KinematisGraphData class.

Author

Pavel Lakiza

Date

August 2022

## 5.99 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/kinematicsgraphdata.h File Reference

Declaration of the KinematisGraphData class.

```
#include "abstractgraphdata.h"
```

### Classes

- class [RSE::Viewers::KinematicsGraphData](#)  
*Class to deal with kinematics of KLP-data.*

### 5.99.1 Detailed Description

Declaration of the KinematisGraphData class.

Author

Pavel Lakiza

Date

August 2022

## 5.100 kinematicsgraphdata.h

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

```

1
8 #ifndef KINEMATICSGRAPHDATA_H
9 #define KINEMATICSGRAPHDATA_H
10
11 #include "abstractgraphdata.h"
12
13 namespace RSE::Viewers
14 {
15
17 class KinematicsGraphData : public AbstractGraphData
18 {
19     Q_OBJECT
20
21 public:
22     enum KinematicsType
23     {
24         kStrain,
25         kDisplacement,
26         kRotation,
27         kSpeed,
28         kAngularSpeed,
29         kAcceleration,
30         kAngularAcceleration
31     };
32     Q_ENUM(KinematicsType)
33     KinematicsGraphData(KinematicsType type, Direction direction = Direction::dFull);
34     ~KinematicsGraphData() = default;
35     GraphDataset getDataset(KLP::FrameCollection const& collection, qint64 sliceIndex) const override;
36     int type() const override { return mType; }
37
38 private:
39     KinematicsType mType;
40 };
41
42 }
43
44 #endif // KINEMATICSGRAPHDATA_H

```

## 5.101 /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 "resultlistmodel.h"
#include "graphlistmodel.h"
#include "graph.h"

```

### Typedefs

- using **CurveData** = QPair< QVector< GraphDataset >, QVector< int > >



## Functions

- CurveData **getCurveData** (PointerGraph const pGraph, PointerResult const pResult, QVector< int > const &indicesData)

*Helper function to retrieve data associated with a 2D-curve.*

### 5.101.1 Detailed Description

Definition of the KLPGraphViewer class.

#### Author

Pavel Lakiza

#### Date

August 2022

## 5.102 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/klpgraphviewer.h File Reference

Declaration of the KLPGraphViewer class.

```
#include "qcustomplot.h"
#include "klp/aliasklp.h"
#include "aliasviewers.h"
#include "propertytreewidget.h"
```

## Classes

- class [RSE::Viewers::KLPGraphViewer](#)

*Class to graphically represent content of KLP output files.*

### 5.102.1 Detailed Description

Declaration of the KLPGraphViewer class.

#### Author

Pavel Lakiza

#### Date

August 2022

## 5.103 klpgraphviewer.h

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

```

1
2
3
4
5
6
7
8 #ifndef KLPGRAPHVIEWER_H
9 #define KLPGRAPHVIEWER_H
10
11 #include "qcustomplot.h"
12 #include "klp/aliasklp.h"
13 #include "aliasviewers.h"
14 #include "propertytreewidget.h"
15
16 QT_BEGIN_NAMESPACE
17 class QSettings;
18 QT_END_NAMESPACE
19
20 namespace ads
21 {
22 class CDockManager;
23 class CDockWidget;
24 }
25
26 namespace KLP
27 {
28 class ResultInfo;
29 }
30
31 namespace RSE
32 {
33
34 namespace Models
35 {
36 class ResultListModel;
37 class GraphListModel;
38 }
39
40 namespace Viewers
41 {
42
43 class PropertyTreeWidget;
44
45 class KLPGraphViewer : public QWidget
46 {
47     Q_OBJECT
48
49 public:
50     KLPGraphViewer(QString const& lastPath, QSettings& settings, QWidget* pParent = nullptr);
51     ~KLPGraphViewer();
52     // Results
53     void openResultsDialog();
54     void openResults(QStringList const& locationFiles);
55     // Graphs
56     void setGraphs(MapGraphs&& graphs);
57     void plot();
58
59 private:
60     // Content
61     void initialize();
62     void createContent();
63     ads::CDockWidget* createResultWidget();
64     ads::CDockWidget* createFigureWidget();
65     ads::CDockWidget* createConstructorWidget();
66     ads::CDockWidget* createPropertyWidget();
67     // Results
68     void processSelectedResults();
69     void showResultInfo(KLP::ResultInfo const& info);
70     // Graphs
71     void processSelectedGraphs();
72     // Settings
73     void saveSettings();
74     void restoreSettings();
75     void closeEvent(QCloseEvent* pEvent) override;
76     // Plotting
77     void plotSurface(PointerGraph const pGraph, PointerResult const pResult);
78     void plotCurve(PointerGraph const pGraph, PointerResult const pResult);
79
80 private:
81     QString mLastPath;
82     QSettings& mSettings;
83     // GUI
84     ads::CDockManager* mpDockManager = nullptr;
85     QCustomPlot* mpFigure;
86     QListView* mpListResults;
87     QTextEdit* mpTextInfo;
88     QListView* mpListGraphs;

```

```

90     RSE::Viewers::PropertyTreeWidget* mpPropertyTreeWidget;
91     // Models
92     RSE::Models::ResultListModel* mpResultListModel;
93     RSE::Models::GraphListModel* mpGraphListModel;
94     // Data
95     KLP::Results mResults;
96     MapGraphs mGraphs;
97 };
98
99 }
100
101 }
102
103 #endif // KLPGRAPHVIEWER_H

```

## 5.104 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/propertytreewidget.cpp File Reference ↩

Definition of the PropertyTreeWidget class.

```

#include <QMetaEnum>
#include <QComboBox>
#include <QSpinBox>
#include <QSlider>
#include <QHeaderView>
#include "propertytreewidget.h"
#include "graph.h"
#include "abstractgraphdata.h"
#include "spacetimegraphdata.h"
#include "kinematicsgraphdata.h"
#include "energygraphdata.h"
#include "klp/result.h"

```

### 5.104.1 Detailed Description

Definition of the PropertyTreeWidget class.

#### Author

Pavel Lakiza

#### Date

August 2022

## 5.105 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/propertytreewidget.h File Reference ↩

Declaration of the PropertyTreeWidget class.

```

#include <QTreeWidget>
#include "aliasklp.h"
#include "aliasviewers.h"

```

## Classes

- class [RSE::Viewers::PropertyTreeWidget](#)

## Typedefs

- using **RSE::Viewers::EnumData** = QPair< QStringList, QList< QIcon > >

### 5.105.1 Detailed Description

Declaration of the PropertyTreeWidget class.

#### Author

Pavel Lakiza

#### Date

August 2022

## 5.106 propertytreewidget.h

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

```

1
2
3
4
5
6
7
8 #ifndef PROPERTYTREETEWIDGET_H
9 #define PROPERTYTREETEWIDGET_H
10
11 #include <QTreeWidget>
12 #include "aliasklp.h"
13 #include "aliasviewers.h"
14
15 QT_BEGIN_NAMESPACE
16 class QComboBox;
17 class QSpinBox;
18 class QDoubleSpinBox;
19 QT_END_NAMESPACE
20
21 namespace KLP
22 {
23     class Result;
24 }
25
26 namespace RSE
27 {
28
29     namespace Viewers
30     {
31
32         using EnumData = QPair<QStringList, QList<QIcon>>;
33         using KLP::PointerResult;
34
35         class PropertyTreeWidget : public QTreeWidget
36         {
37             Q_OBJECT
38
39         public:
40             PropertyTreeWidget(QWidget* pParent = nullptr);
41             void setSelectedGraph(PointerGraph pGraph);
42             void setSelectedResult(PointerResult pResult);
43
44         signals:
45             void graphChanged();
46
47         private:
48             void initialize();
49             void updateValues();

```

```

50     // Construct widgets
51     void createHierarchy();
52     QTreeWidgetItem* createDirectionalDataItem(QString const& name);
53     void createDataSlicerItem();
54     void createAxesLabelsItem();
55     void specifyConnections();
56     void setBlockedSignals(bool flag);
57     // Data and visual widgets
58     int currentDataIndex(int iData, int iChild);
59     QString currentDataText(int iData, int iChild);
60     void setTypeWidget(int iData);
61     void setColorItem(QColor const& color);
62     // Slicer widgets
63     void resetSlicerWidgetsData();
64     void updateSlicerWidgetsData();
65     void setBlockedSlicerWidgetsSignals(bool flag);
66     // Assign new properties to the current graph
67     void assignGraphData(int iData);
68     void assignVisualProperties();
69     void assignSlicer();
70     void setSlicerIndex(qint64 index);
71     void setSlicerValue(float value);
72     // Translation of enum keys
73     EnumData getEnumData(QMetaObject const& metaObject, std::string const& nameEnumerator);
74     void makeTranslationMap();
75
76 private:
77     // Data items
78     PointerGraph mpGraph = nullptr;
79     PointerResult mpResult = nullptr;
80     QList<QTreeWidgetItem*> mDataItems;
81     QTreeWidgetItem* mpDataSlicerItem;
82     // Visual items
83     QMap<QString, QString> mEnumTranslator;
84     QComboBox* mpLineStyleWidget;
85     QSpinBox* mpLineWidthWidget;
86     QTreeWidgetItem* mpColorItem;
87     QComboBox* mpScatterShapeWidget;
88     QDoubleSpinBox* mpScatterSizeWidget;
89     QTreeWidgetItem* mpAxesLabelsItem;
90 };
91
92 }
93
94 }
95
96 #endif // PROPERTYTREETWIDGET_H

```

## 5.107 /home/qinterfly/Library/Projects/Current/RodSystemEstimator/src/viewers/resultlistmodel.cpp File Reference

Definition of the KLPRResultListModel class.

```

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

```

### 5.107.1 Detailed Description

Definition of the KLPRResultListModel class.

Author

Pavel Lakiza

Date

July 2022

## 5.108 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/resultlistmodel.h File Reference

Declaration of the KLPResultListModel class.

```
#include <QStandardItemModel>
#include "klp/aliasklp.h"
```

### Classes

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

### 5.108.1 Detailed Description

Declaration of the KLPResultListModel class.

#### Author

Pavel Lakiza

#### Date

July 2022

## 5.109 resultlistmodel.h

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

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

## 5.110 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/spacetimegraphdata.cpp File Reference

Definition of the SpaceTimeGraphData class.

```
#include "spacetimegraphdata.h"  
#include "klp/framecollection.h"
```

### 5.110.1 Detailed Description

Definition of the SpaceTimeGraphData class.

Author

Pavel Lakiza

Date

August 2022

## 5.111 /home/qinterfly/Library/Projects/Current/RodSystem↵ Estimator/src/viewers/spacetimegraphdata.h File Reference

Declaration of the SpaceTimeGraphData class.

```
#include "abstractgraphdata.h"
```

### Classes

- class [RSE::Viewers::SpaceTimeGraphData](#)  
*Class to deal with spacetime KLP-data.*

### 5.111.1 Detailed Description

Declaration of the SpaceTimeGraphData class.

Author

Pavel Lakiza

Date

August 2022

## 5.112 spacetimegraphdata.h

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

```
1
2
3
4
5
6
7
8 #ifndef SPACETIMEGRAPHDATA_H
9 #define SPACETIMEGRAPHDATA_H
10
11 #include "abstractgraphdata.h"
12
13 namespace RSE::Viewers
14 {
15
16
17 class SpaceTimeGraphData : public AbstractGraphData
18 {
19     Q_OBJECT
20
21 public:
22     enum SpaceTimeType
23     {
24         stTime,
25         stParameter,
26         stNaturalLength,
27         stAccumulatedNaturalLength,
28         stCoordinate
29     };
30     Q_ENUM(SpaceTimeType)
31     SpaceTimeGraphData(SpaceTimeType type, Direction direction = Direction::dFull);
32     ~SpaceTimeGraphData() = default;
33     GraphDataset getDataset(KLP::FrameCollection const& collection, qint64 sliceIndex) const override;
34     int type() const override { return mType; }
35
36 private:
37     SpaceTimeType mType;
38 };
39
40 }
41
42 #endif // SPACETIMEGRAPHDATA_H
```