Intention Recognition

Generated by Doxygen 1.7.4

Tue Sep 17 2013 10:46:24

Contents

Chapter 1

Intention Recognition

This tool is capable of reading a plan file to ’recognize’ which kit is being built.

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Namespace Index

3.1 Packages

Here are the packages with brief descriptions (if available):

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Module Documentation

6.1 Graphical User Interface

Classes

- class gui.Chart

  Chart display for metrics and likelihoods.
- class gui.CommonGUIComponents

  Common GUI components used across different files of the project.

6.2 Intention Structure

Classes

- class intention.Intention

  Representation of intentions from their definition in the ontology.
- class intention.Metric

  Definition of additive and multiplicative metrics.

Chapter 7

Namespace Documentation

7.1 Package gui

Classes

- class Chart

  Chart display for metrics and likelihoods.
- class CommonGUIComponents

  Common GUI components used across different files of the project.
- class DemoPanel
- class DrawStringPanel
- class MainFrame
- class OntologyChooser
- class OptionFrame
- class PDFChartTransferable
- class ProgressBar

7.2 Package intention

Classes

- class Intention

  Representation of intentions from their definition in the ontology.
- class Metric

  Definition of additive and multiplicative metrics.

7.3 Package main

Classes

- class Launcher

  Main class of the tool.

7.4 Package ontology

Classes

- class Ontology

7.5 Package orderingconstruct

Formal mechanism to allow an ordering of state relationships to represent an intention.

Classes

- class AnyOrder

  A set of state relationships that must all occur in any order.
- class Count

  A state relationship that must be present multiple times.
- class Exist

  A state relationship that must exist.
- class OrderedList

  A set of state relationships that must occur in a specific order.

7.5.1 Detailed Description

Formal mechanism to allow an ordering of state relationships to represent an intention. In this work, an ordering of state relationships represents an intention. As such, we need we need a formal mechanism to allow for this ordering. To do this, we borrow some concepts that are described in OWL-S (Web Ontology Language – Services) (Martin, 2004). OWL-S is described on the Website (http://www.w3.org/Submission/OWL-S/) as an ontology of services enabling a user and software agents to discover, invoke, compose and monitor Web resources offering particular services and having particular properties.

Though intended for web-based services, many of the same ordering constructs are equally applicable to the representation of the sequencing of states. OWL-S defines eight control constructs. Only four of them are used in this project.

- Perform: execution of an action
- Sequence: a list of control constructs to be done in order

- Any-Order: process components (specified as a bag) to be executed in some unspecified order but not concurrently. All components must be executed.
- Iterate: makes no assumption about how many iterations are made or when to initiate, terminate, or resume. The initiation, termination or maintenance condition could be specified with a whileCondition or an untilCondition.

We adapt some of the these control constructs to represent the ordering of states by changing their name and definition as shown in Table 7.1

Table 7.1: Initial State Representation Ordering Constructs.

OWL-S Control Construct	State Representation Ordering Construct	State Representation Definition
Perform	Exist	A state relationship must exist
Sequence	OrderedList	A set of state relationships that must occur in a specific order
Any-Order	Any-Order	A set of state relationships that must all occur in any order
Iterate	Count	A state relationship that must be present multiple times. This often involves multiple instances of a specific object that holds a predefined spatial relationship with one or more instances of another object.

7.6 Package predicate

Definition of a structure for predicates as represented in the ontology.

Classes

- class Predicate

  These are domain-specific states that are of interest to the current intention (or set of intentions) being evaluated.

7.6.1 Detailed Description

Definition of a structure for predicates as represented in the ontology. A predicate is used to specify a binary property of a single object, or a relationship between two objects. For example, the predicate (robot-empty ?robot) is true if the robot ?robot is not holding anything. The predicate (part-location-robot ?part ?robot) is true only if the reference parameter ?part is being held by the target parameter ?robot. A predicate has a unique name of type string, a reference parameter and a target parameter. The reference parameter is the first parameter in the predicate's parameter list and the target parameter is the second parameter in the predicate's parameter list. An ActionPredicate cannot have more than two parameters due to the inherent definition of predicates.

In the case where an ActionPredicate has only one parameter, it is assigned to the reference parameter.

7.7 Package tools

Classes

- class Configuration
- class FileOperator
- class IntFilter

7.7.1 Detailed Description

This package provides all the necessary tools needed by the project.

7.8 Package treecheckbox

Classes

- class CheckTreeCellRenderer
- class CheckTreeManager
- class CheckTreeSelectionModel
- class TreeExample
- class TristateCheckBox

7.8.1 Detailed Description

This package provides all the functions that allow trees rendering and manipulation.

The particularity of these trees is that the root and leaves are selectable via Java check boxes (CheckBox).

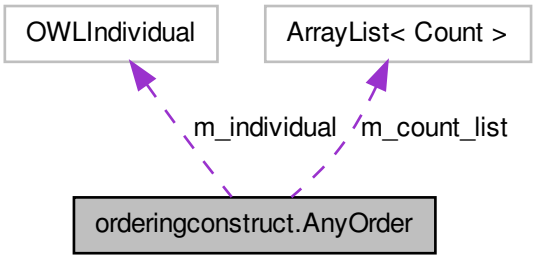
Chapter 8

Class Documentation

8.1 orderingconstruct.AnyOrder Class Reference

A set of state relationships that must all occur in any order.

Collaboration diagram for orderingconstruct.AnyOrder:



Public Member Functions

- AnyOrder ()

  Class constructor.
- void addCountToList (Count count_)

  Add an element of type Count to a list.
- ArrayList< Count > getCountList ()

  Return the list that contains elements of type Count.
- OWLIndividual getIndividual ()

Return an element of AnyOrder that is an OWLIndividual.

- int getPosition ()

  Return the position of an element of type AnyOrder within the structure of an intention.

- int getTotalNumber ()

  Return the total number of elements within an AnyOrder element.

- void setIndividual (OWLIndividual individual_)

  Set an AnyOrder element as an OWLIndividual.

- void setPosition (int position_)

  Set the position of an AnyOrder element within the structure of an intention.

- void setTotalNumber (int total_number_)

  Set the total number of elements within an AnyOrder element.

Private Attributes

- ArrayList< Count > m_count_list

  A List of "Count" elements.

- OWLIndividual m_individual

  OWLIndividual is used to retrieve range individuals from the ontology given the data and object properties.

- int m_position

  Position of an element in an AnyOrder list.

- int m_total_number

  Total number of elements in an AnyOrder list.

8.1.1 Detailed Description

A set of state relationships that must all occur in any order.

Author

Zeid Kootbally *zeid.kootbally@nist.gov*

8.1.2 Constructor & Destructor Documentation

8.1.2.1 orderingconstruct.AnyOrder.AnyOrder ( )

Class constructor.

Initialize the list AnyOrder.m_count_list

8.1.3 Member Function Documentation

8.1.3.1 void orderingconstruct.AnyOrder.addCountToList (Count *count_*)

Add an element of type Count to a list.

Parameters

*count_*	The Count element to add to the list AnyOrder.m_count_list

8.1.3.2 ArrayList<Count> orderingconstruct.AnyOrder.getCountList ( )

Return the list that contains elements of type Count.

Returns

AnyOrder.m_count_list

8.1.3.3 OWLIndividual orderingconstruct.AnyOrder.getIndividual ( )

Return an element of AnyOrder that is an OWLIndividual.

Returns

AnyOrder.m_individual

8.1.3.4 int orderingconstruct.AnyOrder.getPosition ( )

Return the position of an element of type AnyOrder within the structure of an intention.

Returns

AnyOrder.m_position

8.1.3.5 int orderingconstruct.AnyOrder.getTotalNumber ( )

Return the total number of elements within an AnyOrder element.

AnyOrder usually consists of multiple Count elements

Returns

AnyOrder.m_total_number

8.1.3.6 void orderingconstruct.AnyOrder.setIndividual (OWLIndividual *individual_*)

Set an AnyOrder element as an OWLIndividual.

Parameters

*individual_*	OWLIndividual to set to AnyOrder.m_individual

8.1.3.7 void orderingconstruct.AnyOrder.setPosition (int *position_*)

Set the position of an AnyOrder element within the structure of an intention.

Parameters

*position_*	Position to set to AnyOrder.m_position

8.1.3.8 void orderingconstruct.AnyOrder.setTotalNumber (int *total_number_*)

Set the total number of elements within an AnyOrder element.

Parameters

*total_number_*	Total number to set to AnyOrder.m_total_number

8.1.4 Member Data Documentation

8.1.4.1 ArrayList<Count> orderingconstruct.AnyOrder.m_count_list [private]

A List of "Count" elements.

8.1.4.2 OWLIndividual orderingconstruct.AnyOrder.m_individual [private]

OWLIndividual is used to retrieve range individuals from the ontology given the data and object properties.

8.1.4.3 int orderingconstruct.AnyOrder.m_position [private]

Position of an element in an AnyOrder list.

8.1.4.4 int orderingconstruct.AnyOrder.m_total_number [private]

Total number of elements in an AnyOrder list.

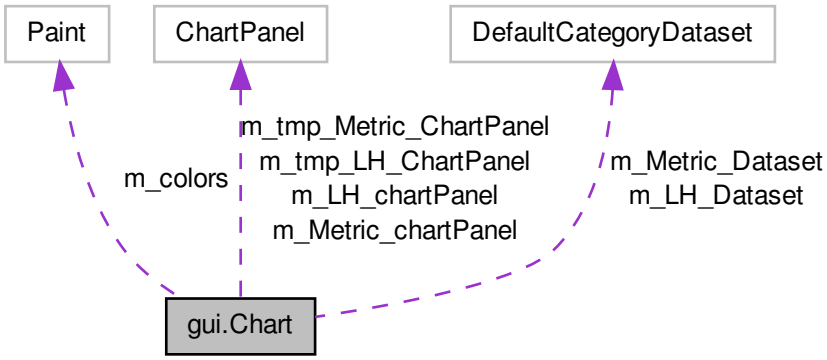
The documentation for this class was generated from the following file:

- src/orderingconstruct/AnyOrder.java

8.2 gui.Chart Class Reference

Chart display for metrics and likelihoods.

Collaboration diagram for gui.Chart:



Classes

- class CustomRenderer
- class CustomRendererLine

Public Member Functions

- Chart ()

  Class constructor.
- LegendItemCollection getLegendItems (CombinedDomainCategoryPlot combined-domaincategoryplot)

  Returns a collection of legend items for one of the subplot of a CombinedDomainCategoryPlot.
- ChartPanel updateChart_metrics (DefaultCategoryDataset metricsDataset)

Static Public Member Functions

- static void setChartTitle (String title_)

  Set the title for the chart displaying intention likelihoods.
- static String getChartTitle ()

  Get the title for the chart displaying intention likelihoods.
- static void setMetricsChartTitle (String title_)

  Set the title for the chart displaying intention metrics.
- static String getMetricsChartTitle ()

  Get the title for the chart displaying intention metrics.
- static ChartPanel createChart_metrics ()
- static void createLikelihoodDataset (double _likelihood, Intention _intention, int _state)
- static void createMetricsDataset (Intention _intention, int _state)

  Create a data set for metrics.

Public Attributes

- ChartPanel m_LH_chartPanel = createChart_likelihood()
- ChartPanel m_Metric_chartPanel = createChart_metrics()
- ChartPanel m_tmp_Metric_ChartPanel
- ChartPanel m_tmp_LH_ChartPanel

Static Public Attributes

- static DefaultCategoryDataset m_LH_Dataset = new DefaultCategoryDataset()

  Data set for likelihoods.
- static DefaultCategoryDataset m_Metric_Dataset = new DefaultCategoryDataset()

  Data set for metrics.

Package Functions

- ChartPanel updateChart_likelihood (DefaultCategoryDataset dataset)

Static Package Attributes

- static Paint[] m_colors

  Array of colors.

Private Member Functions

- ChartPanel createChart_likelihood ()

Static Private Attributes

- static String m_LH_Chart_Title

  Title of the chart window.
- static String m_Metric_Chart_Title

  Title of the metrics chart.

8.2.1 Detailed Description

Chart display for metrics and likelihoods.

This class consists of components that allow the display and selection of metrics and likelihoods

Author

Zeid Kootbally *zeid.kootbally@nist.gov*

Date

September 2013

8.2.2 Constructor & Destructor Documentation

8.2.2.1 gui.Chart.Chart ( )

Class constructor.

- Allow the use of the mouse wheel to zoom in and out on the charts
- Allow the use of autoscroll on the charts

8.2.3 Member Function Documentation

8.2.3.1 ChartPanel gui.Chart.createChart_likelihood ( ) [private]

8.2.3.2 static ChartPanel gui.Chart.createChart_metrics ( ) [static]

8.2.3.3 static void gui.Chart.createLikelihoodDataset (double *_likelihood*, Intention *_intention*, int *_state*) [static]

8.2.3.4 static void gui.Chart.createMetricsDataset (Intention *_intention*, int *_state*) [static]

Create a data set for metrics.

Parameters

_intention	The intention from which we will retrieve the metrics
_state	The current state

8.2.3.5 static String gui.Chart.getChartTitle ( ) [static]

Get the title for the chart displaying intention likelihoods.

8.2.3.6 LegendItemCollection gui.Chart.getLegendItems (CombinedDomainCategoryPlot *combineddomaincategoryplot*)

Returns a collection of legend items for one of the subplot of a CombinedDomainCategoryPlot.

Parameters

combined-domaincate-goryplot	Instance of CombinedDomainCategoryPlot

Returns

The legend items.

8.2.3.7 static String gui.Chart.getMetricsChartTitle ( ) [static]

Get the title for the chart displaying intention metrics.

8.2.3.8 static void gui.Chart.setChartTitle (String *title_*) [static]

Set the title for the chart displaying intention likelihoods.

Parameters

*title_*	Title of the Likelihood chart

8.2.3.9 static void gui.Chart.setMetricsChartTitle (String *title_*) [static]

Set the title for the chart displaying intention metrics.

Parameters

*title_*	Title of the Metrics chart

8.2.3.10 ChartPanel gui.Chart.updateChart_likelihood (DefaultCategoryDataset *dataset*) [package]

8.2.3.11 ChartPanel gui.Chart.updateChart_metrics (DefaultCategoryDataset *metricsDataset*)

8.2.4 Member Data Documentation

8.2.4.1 Paint [] gui.Chart.m_colors [static, package]

Initial value:

```
new Paint[] { CommonGUIComponents.intentionColor9,
               CommonGUIComponents.intentionColor2, CommonGUIComponents.intentio
     nColor3, CommonGUIComponents.intentionColor10,
               CommonGUIComponents.intentionColor1, CommonGUIComponents.intentio
     nColor6, CommonGUIComponents.intentionColor7,
               CommonGUIComponents.intentionColor8 }
```

Array of colors.

8.2.4.2 String gui.Chart.m_LH_Chart_Title [static, private]

Title of the chart window.

8.2.4.3 ChartPanel gui.Chart.m_LH_chartPanel = createChart_likelihood()

8.2.4.4 DefaultCategoryDataset gui.Chart.m_LH_Dataset = new DefaultCategoryDataset() [static]

Data set for likelihoods.

8.2.4.5 String gui.Chart.m_Metric_Chart_Title [static, private]

Title of the metrics chart.

8.2.4.6 ChartPanel gui.Chart.m_Metric_chartPanel = createChart_metrics()

8.2.4.7 DefaultCategoryDataset gui.Chart.m_Metric_Dataset = new DefaultCategoryDataset() [static]

Data set for metrics.

8.2.4.8 ChartPanel gui.Chart.m_tmp_LH_ChartPanel

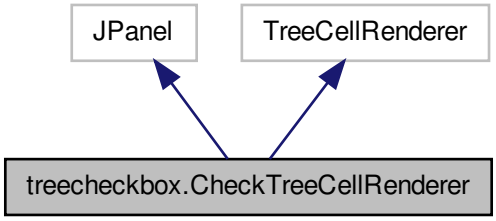
8.2.4.9 ChartPanel gui.Chart.m_tmp_Metric_ChartPanel

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

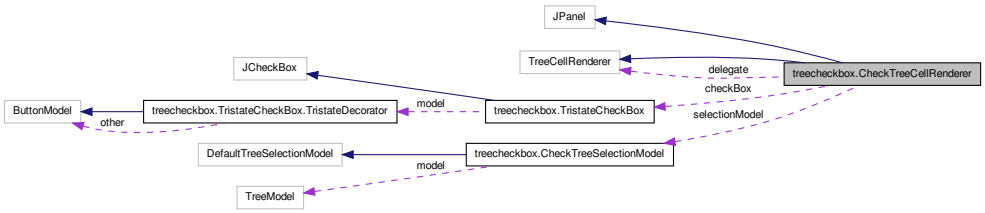
- src/gui/Chart.java

8.3 treecheckbox.CheckTreeCellRenderer Class Reference

Inheritance diagram for treecheckbox.CheckTreeCellRenderer:



Collaboration diagram for treecheckbox.CheckTreeCellRenderer:



Public Member Functions

- CheckTreeCellRenderer (TreeCellRenderer delegate, CheckTreeSelectionModel selectionModel)
- Component getTreeCellRendererComponent (JTree tree, Object value, boolean selected, boolean expanded, boolean leaf, int row, boolean hasFocus)

Private Attributes

- CheckTreeSelectionModel selectionModel
- TreeCellRenderer delegate
- TristateCheckBox checkBox = new TristateCheckBox()

8.3.1 Constructor & Destructor Documentation

8.3.1.1 treecheckbox.CheckTreeCellRenderer.CheckTreeCellRenderer (TreeCellRenderer *delegate,* CheckTreeSelectionModel *selectionModel*)

8.3.2 Member Function Documentation

8.3.2.1 Component treecheckbox.CheckTreeCellRenderer.getTreeCellRendererComponent (JTree *tree,* Object *value,* boolean *selected,* boolean *expanded,* boolean *leaf,* int *row,* boolean *hasFocus*)

8.3.3 Member Data Documentation

8.3.3.1 TristateCheckBox treecheckbox.CheckTreeCellRenderer.checkBox = new TristateCheckBox() [private]

8.3.3.2 TreeCellRenderer treecheckbox.CheckTreeCellRenderer.delegate [private]

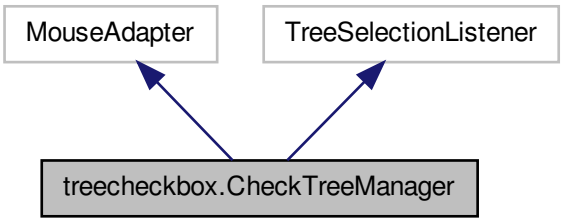
8.3.3.3 CheckTreeSelectionModel treecheckbox.CheckTreeCellRenderer.selectionModel [private]

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

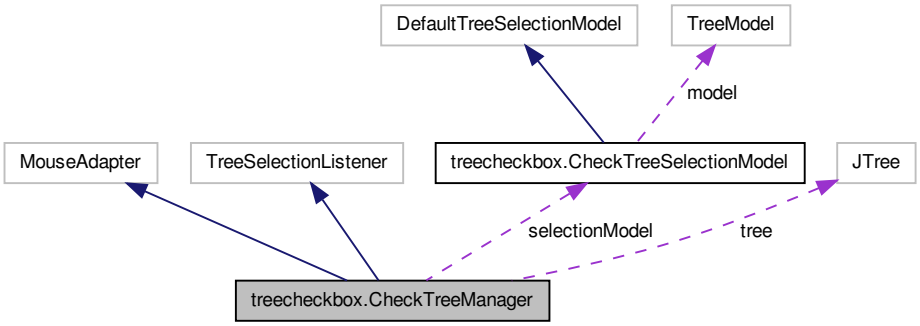
- src/treecheckbox/CheckTreeCellRenderer.java

8.4 treecheckbox.CheckTreeManager Class Reference

Inheritance diagram for treecheckbox.CheckTreeManager:



Collaboration diagram for treecheckbox.CheckTreeManager:



Public Member Functions

- CheckTreeManager (JTree tree)
- void addChildPaths (TreePath path, TreeModel model, List result)
- ArrayList getDescendants (TreePath paths[], TreeModel model)
- ArrayList< Object > getAllCheckedPaths (CheckTreeManager manager, JTree tree)
- void mouseClicked (MouseEvent me)
- CheckTreeSelectionModel getSelectionModel ()
- void valueChanged (TreeSelectionEvent e)

Package Attributes

- int hotspot = new JCheckBox().getPreferredSize().width

Private Attributes

- CheckTreeSelectionModel selectionModel
- JTree tree = new JTree()

8.4.1 Constructor & Destructor Documentation

8.4.1.1 treecheckbox.CheckTreeManager.CheckTreeManager (JTree *tree*)

8.4.2 Member Function Documentation

8.4.2.1 void treecheckbox.CheckTreeManager.addChildPaths (TreePath *path,* TreeModel *model,* List *result*)

8.4.2.2 ArrayList<Object> treecheckbox.CheckTreeManager.getAllCheckedPaths (CheckTreeManager *manager,* JTree *tree*)

8.4.2.3 ArrayList treecheckbox.CheckTreeManager.getDescendants (TreePath *paths[],* TreeModel *model*)

8.4.2.4 CheckTreeSelectionModel treecheckbox.CheckTreeManager.getSelectionModel ( )

8.4.2.5 void treecheckbox.CheckTreeManager.mouseClicked (MouseEvent *me*)

8.4.2.6 void treecheckbox.CheckTreeManager.valueChanged (TreeSelectionEvent *e*)

8.4.3 Member Data Documentation

8.4.3.1 int treecheckbox.CheckTreeManager.hotspot = new JCheckBox().getPreferredSize().width [package]

8.4.3.2 CheckTreeSelectionModel treecheckbox.CheckTreeManager.selectionModel [private]

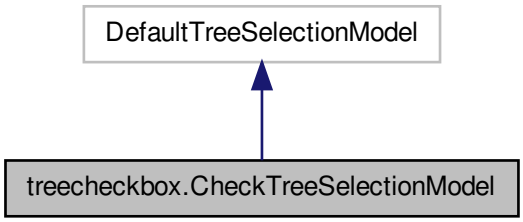
8.4.3.3 JTree treecheckbox.CheckTreeManager.tree = new JTree() [private]

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

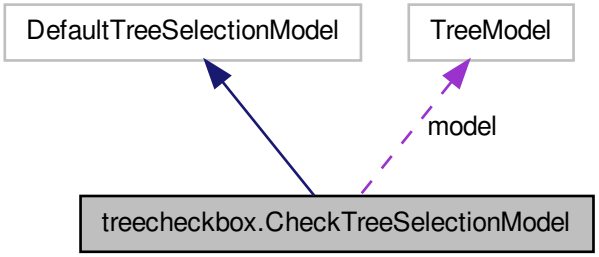
- src/treecheckbox/CheckTreeManager.java

8.5 treecheckbox.CheckTreeSelectionModel Class Reference

Inheritance diagram for treecheckbox.CheckTreeSelectionModel:



Collaboration diagram for treecheckbox.CheckTreeSelectionModel:



Public Member Functions

- CheckTreeSelectionModel (TreeModel model)
- boolean isPartiallySelected (TreePath path)
- boolean isPathSelected (TreePath path, boolean dig)
- void setSelectionPaths (TreePath[] pPaths)
- void addSelectionPaths (TreePath[] paths)
- void removeSelectionPaths (TreePath[] paths)

Private Member Functions

- boolean isDescendant (TreePath path1, TreePath path2)
- boolean areSiblingsSelected (TreePath path)
- void toggleRemoveSelection (TreePath path)

Private Attributes

- TreeModel model

8.5.1 Constructor & Destructor Documentation

8.5.1.1 treecheckbox.CheckTreeSelectionModel.CheckTreeSelectionModel (TreeModel *model*)

8.5.2 Member Function Documentation

8.5.2.1 void treecheckbox.CheckTreeSelectionModel.addSelectionPaths (TreePath[] *paths*)

8.5.2.2 boolean treecheckbox.CheckTreeSelectionModel.areSiblingsSelected (TreePath *path*) [private]

8.5.2.3 boolean treecheckbox.CheckTreeSelectionModel.isDescendant (TreePath *path1,* TreePath *path2*) [private]

8.5.2.4 boolean treecheckbox.CheckTreeSelectionModel.isPartiallySelected (TreePath *path*)

8.5.2.5 boolean treecheckbox.CheckTreeSelectionModel.isPathSelected (TreePath *path,* boolean *dig*)

8.5.2.6 void treecheckbox.CheckTreeSelectionModel.removeSelectionPaths (TreePath[] *paths*)

8.5.2.7 void treecheckbox.CheckTreeSelectionModel.setSelectionPaths (TreePath[] *pPaths*)

8.5.2.8 void treecheckbox.CheckTreeSelectionModel.toggleRemoveSelection (TreePath *path*) [private]

8.5.3 Member Data Documentation

8.5.3.1 TreeModel treecheckbox.CheckTreeSelectionModel.model [private]

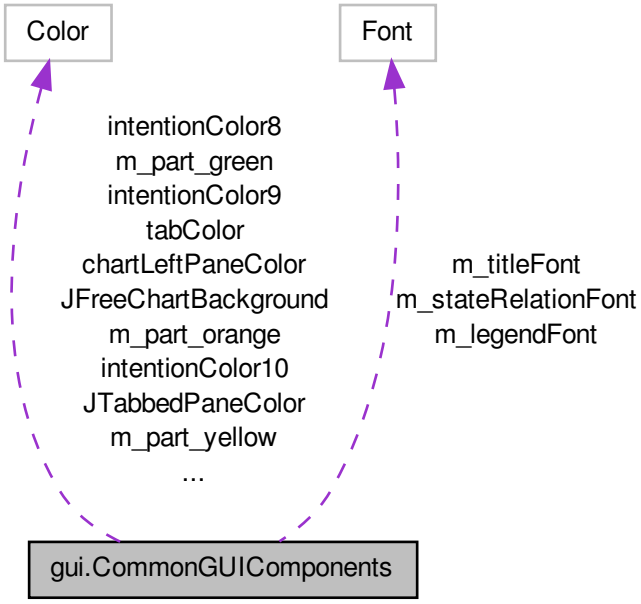
The documentation for this class was generated from the following file:

- src/treecheckbox/CheckTreeSelectionModel.java

8.6 gui.CommonGUIComponents Class Reference

Common GUI components used across different files of the project.

Collaboration diagram for gui.CommonGUIComponents:



Static Public Attributes

- static Color chartPanelColor = new Color(160, 188, 136)
- static Color menuBarColor = new Color(75, 148, 48)
- static Color tabColor = new Color(255, 255, 255)
- static Color chartLeftPaneColor = new Color(255, 255, 255)
- static Color intentionColor1 = new Color(255, 0, 200)
- static Color intentionColor2 = new Color(255, 145, 0)
- static Color intentionColor3 = new Color(118, 175, 60)
- static Color intentionColor6 = new Color(255, 255, 255)
- static Color intentionColor7 = new Color(255, 111, 0)
- static Color intentionColor8 = new Color(0, 255, 34)
- static Color intentionColor9 = new Color(0, 137, 255)
- static Color intentionColor10 = new Color(64, 73, 81)
- static Color state_color = new Color(255, 0, 0)
- static Color JTabbedPaneColor = new Color(255, 145, 0)
- static Color JFreeChartBackground = new Color(214,224,219)
- static Color m_part_orange = new Color(233, 148, 0)

- static Color m_part_green = new Color(71, 144, 30)
- static Color m_part_yellow = new Color(236, 236, 43)
- static Color m_state = new Color(129, 129, 120)
- static Font m_titleFont = new Font("Times",Font.PLAIN, 18)
- static Font m_legendFont = new Font("Times",Font.PLAIN, 13)
- static Font m_stateRelationFont = new Font("Times",Font.PLAIN, 14)

Static Private Attributes

- static Color intentionColor4 = new Color(50, 75, 156)
- static Color intentionColor5 = new Color(242, 9, 9)

8.6.1 Detailed Description

Common GUI components used across different files of the project.

This class defines colors and fonts that are used in multiple source files.

Author

Zeid Kootbally *zeid.kootbally@nist.gov*

Date

September 2013

8.6.2 Member Data Documentation

8.6.2.1 Color gui.CommonGUIComponents.chartLeftPaneColor = new Color(255, 255, 255) [static]

8.6.2.2 Color gui.CommonGUIComponents.chartPanelColor = new Color(160, 188, 136) [static]

8.6.2.3 Color gui.CommonGUIComponents.intentionColor1 = new Color(255, 0, 200) [static]

8.6.2.4 Color gui.CommonGUIComponents.intentionColor10 = new Color(64, 73, 81) [static]

8.6.2.5 Color gui.CommonGUIComponents.intentionColor2 = new Color(255, 145, 0) [static]

8.6.2.6 Color gui.CommonGUIComponents.intentionColor3 = new Color(118, 175, 60) [static]

8.6.2.7 Color gui.CommonGUIComponents.intentionColor4 = new Color(50, 75, 156) [static, private]

8.6.2.8 Color gui.CommonGUIComponents.intentionColor5 = new Color(242, 9, 9) [static, private]

8.6.2.9 Color gui.CommonGUIComponents.intentionColor6 = new Color(255, 255, 255) [static]

8.6.2.10 Color gui.CommonGUIComponents.intentionColor7 = new Color(255, 111, 0) [static]

8.6.2.11 Color gui.CommonGUIComponents.intentionColor8 = new Color(0, 255, 34) [static]

8.6.2.12 Color gui.CommonGUIComponents.intentionColor9 = new Color(0, 137, 255) [static]

8.6.2.13 Color gui.CommonGUIComponents.JFreeChartBackground = new Color(214,224,219) [static]

8.6.2.14 Color gui.CommonGUIComponents.JTabbedPaneColor = new Color(255, 145, 0) [static]

8.6.2.15 Font gui.CommonGUIComponents.m_legendFont = new Font("Times",Font.PLAIN, 13) [static]

8.6.2.16 Color gui.CommonGUIComponents.m_part_green = new Color(71, 144, 30) [static]

8.6.2.17 Color gui.CommonGUIComponents.m_part_orange = new Color(233, 148, 0) [static]

8.6.2.18 Color gui.CommonGUIComponents.m_part_yellow = new Color(236, 236, 43) [static]

8.6.2.19 Color gui.CommonGUIComponents.m_state = new Color(129, 129, 120) [static]

8.6.2.20 Font gui.CommonGUIComponents.m_stateRelationFont = new Font("Times",Font.PLAIN, 14) [static]

8.6.2.21 Font gui.CommonGUIComponents.m_titleFont = new Font("Times",Font.PLAIN, 18) [static]

8.6.2.22 Color gui.CommonGUIComponents.menuBarColor = new Color(75, 148, 48) [static]

8.6.2.23 Color gui.CommonGUIComponents.state_color = new Color(255, 0, 0) [static]

8.6.2.24 Color gui.CommonGUIComponents.tabColor = new Color(255, 255, 255) [static]

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

- src/gui/CommonGUIComponents.java

8.7 tools.Configuration Class Reference

Static Public Member Functions

- static boolean isWindows ()
- static boolean isMac ()
- static boolean isUnix ()

Static Public Attributes

- static String m_OS = System.getProperty("os.name").toLowerCase()

8.7.1 Detailed Description

Author

zeid

8.7.2 Member Function Documentation

8.7.2.1 static boolean tools.Configuration.isMac ( ) [static]

8.7.2.2 static boolean tools.Configuration.isUnix ( ) [static]

8.7.2.3 static boolean tools.Configuration.isWindows ( ) [static]

8.7.3 Member Data Documentation

8.7.3.1 String tools.Configuration.m_OS = System.getProperty("os.name").toLowerCase() [static]

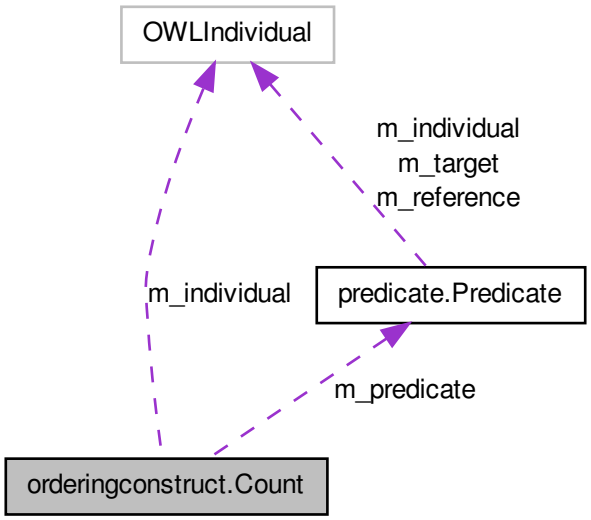
The documentation for this class was generated from the following file:

- src/tools/Configuration.java

8.8 orderingconstruct.Count Class Reference

A state relationship that must be present multiple times.

Collaboration diagram for orderingconstruct.Count:



Public Member Functions

- Count ()

  Class constructor.
- Predicate getPredicate ()

  Return the predicate for the Count element.
- OWLIndividual getIndividual ()

  Return the Count element of type OWLIndividual.
- Integer getOccurence ()

  Return the occurrence associated to a Count element.
- void setPredicate (Predicate predicate_)

  Set a predicate to a Count element.
- void setIndividual (OWLIndividual individual_)

  Set a Count element as an OWLIndividual.
- void setOccurrence (Integer occurrence_)

  Set an occurrence to a Count element.

Private Attributes

- Predicate m_predicate

  Predicate element that constitutes the Count element.
- Integer m_occurrence

  Occurence associated to a Count element.
- OWLIndividual m_individual

  OWLIndividual is used to retrieve range individuals from the ontology given the data and object properties.

8.8.1 Detailed Description

A state relationship that must be present multiple times.

This often involves multiple instances of a specific object that holds a predefined spatial relationship with one or more instances of another object.

Author

Zeid Kootbally *zeid.kootbally@nist.gov*

8.8.2 Constructor & Destructor Documentation

8.8.2.1 orderingconstruct.Count.Count ( )

Class constructor.

8.8.3 Member Function Documentation

8.8.3.1 OWLIndividual orderingconstruct.Count.getIndividual ( )

Return the Count element of type OWLIndividual.

Returns

Count.m_individual

8.8.3.2 Integer orderingconstruct.Count.getOccurence ( )

Return the occurrence associated to a Count element.

Returns

Count.m_occurrence

8.8.3.3 Predicate orderingconstruct.Count.getPredicate ( )

Return the predicate for the Count element.

In the ontology, a Count OWL individual has a predicate.

This is defined by the OWL object property *hasOrderingConstruct_Predicate* where the domain is Count and the range is Predicate.

Returns

Count.m_predicate

8.8.3.4 void orderingconstruct.Count.setIndividual (OWLIndividual *individual_*)

Set a Count element as an OWLIndividual.

Parameters

*individual_*	OWLIndividual to set to Count.m_individual

8.8.3.5 void orderingconstruct.Count.setOccurrence (Integer *occurrence_*)

Set an occurrence to a Count element.

Parameters

*occurrence_* -	Occurrence to set to a Count element

8.8.3.6 void orderingconstruct.Count.setPredicate (Predicate *predicate_*)

Set a predicate to a Count element.

Parameters

*predicate_*	predicate.Predicate element to set to Count.m_predicate

8.8.4 Member Data Documentation

8.8.4.1 OWLIndividual orderingconstruct.Count.m_individual [private]

OWLIndividual is used to retrieve range individuals from the ontology given the data and object properties.

8.8.4.2 Integer orderingconstruct.Count.m_occurrence [private]

Occurence associated to a Count element.

In the ontology, the occurrence specifies the number of parts of a certain type that constitute a kit.

For instance, for the kit $kit_{a2b3c3d1e1}$, the occurrence of parts of type "a" is 2.

This is represented in the ontology with the OWL data property *hasCount_Occurrence*, where the domain is Count and the range is integer.

8.8.4.3 Predicate orderingconstruct.Count.m_predicate [private]

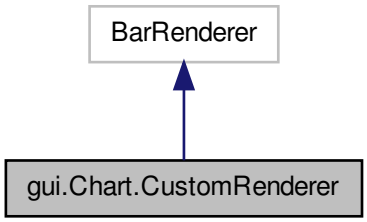
Predicate element that constitutes the Count element.

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

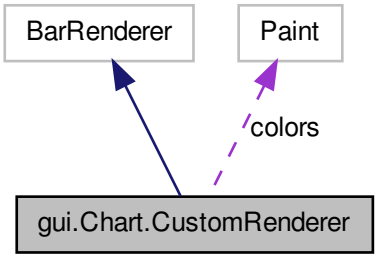
- src/orderingconstruct/Count.java

8.9 gui.Chart.CustomRenderer Class Reference

Inheritance diagram for gui.Chart.CustomRenderer:



Collaboration diagram for gui.Chart.CustomRenderer:



Public Member Functions

- CustomRenderer ()

  Read an array of colors and select each color in order they are in the array.
- Paint getItemPaint (final int row, final int column)

Private Attributes

- Paint[] colors

Static Private Attributes

- static final long serialVersionUID = 1L

8.9.1 Constructor & Destructor Documentation

8.9.1.1 gui.Chart.CustomRenderer.CustomRenderer ( )

Read an array of colors and select each color in order they are in the array.

8.9.2 Member Function Documentation

8.9.2.1 Paint gui.Chart.CustomRenderer.getItemPaint (final int *row*, final int *column*)

8.9.3 Member Data Documentation

8.9.3.1 Paint [] gui.Chart.CustomRenderer.colors [private]

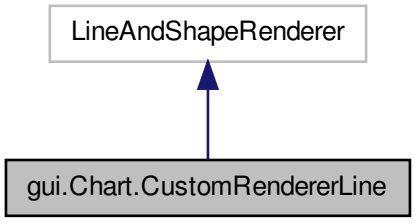
8.9.3.2 final long gui.Chart.CustomRenderer.serialVersionUID = 1L [static, private]

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

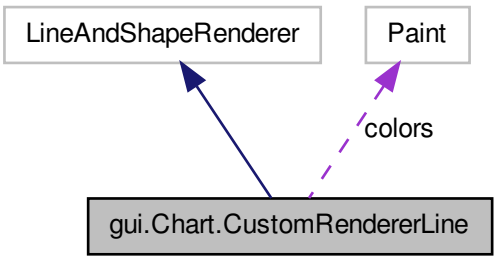
- src/gui/Chart.java

8.10 gui.Chart.CustomRendererLine Class Reference

Inheritance diagram for gui.Chart.CustomRendererLine:



Collaboration diagram for gui.Chart.CustomRendererLine:



Public Member Functions

- CustomRendererLine ()
- Paint getItemPaint (final int row, final int column)

Private Attributes

- Paint[] colors

Static Private Attributes

- static final long serialVersionUID = 1L

8.10.1 Constructor & Destructor Documentation

8.10.1.1 gui.Chart.CustomRendererLine.CustomRendererLine ( )

8.10.2 Member Function Documentation

8.10.2.1 Paint gui.Chart.CustomRendererLine.getItemPaint (final int *row*, final int *column*)

8.10.3 Member Data Documentation

8.10.3.1 Paint [] gui.Chart.CustomRendererLine.colors [private]

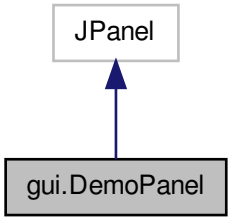
8.10.3.2 final long gui.Chart.CustomRendererLine.serialVersionUID = 1L [static, private]

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

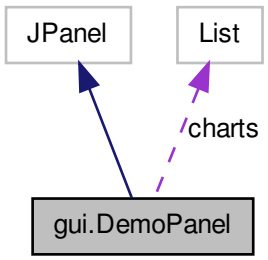
- src/gui/Chart.java

8.11 gui.DemoPanel Class Reference

Inheritance diagram for gui.DemoPanel:



Collaboration diagram for gui.DemoPanel:



Public Member Functions

- DemoPanel (java.awt.LayoutManager layoutmanager)
- void addChart (JFreeChart jfreechart)
- JFreeChart[] getCharts ()

Package Attributes

- List charts

8.11.1 Constructor & Destructor Documentation

8.11.1.1 gui.DemoPanel.DemoPanel (java.awt.LayoutManager *layoutmanager*)

8.11.2 Member Function Documentation

8.11.2.1 void gui.DemoPanel.addChart (JFreeChart *jfreechart*)

8.11.2.2 JFreeChart [ ] gui.DemoPanel.getCharts ( )

8.11.3 Member Data Documentation

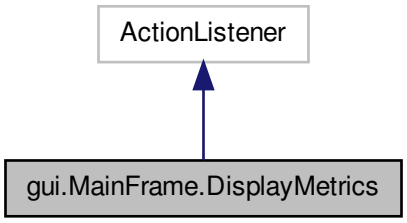
8.11.3.1 List gui.DemoPanel.charts [package]

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

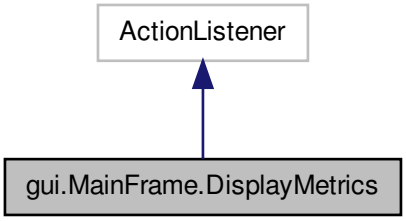
- src/gui/DemoPanel.java

8.12 gui.MainFrame.DisplayMetrics Class Reference

Inheritance diagram for gui.MainFrame.DisplayMetrics:



Collaboration diagram for gui.MainFrame.DisplayMetrics:



Public Member Functions

- void actionPerformed (ActionEvent e)

8.12.1 Detailed Description

Listener for the metrics selection in the JTree of checkboxes

8.12.2 Member Function Documentation

8.12.2.1 void gui.MainFrame.DisplayMetrics.actionPerformed (ActionEvent *e*)

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

- src/gui/MainFrame.java

8.13 DocumentFilter Class Reference

Force a GUI component to contain a certain type of data (text, Integer, Double, etc)

8.13.1 Detailed Description

Force a GUI component to contain a certain type of data (text, Integer, Double, etc)

The user will not be able to enter any type of data other than the one specified. For instance, if the GUI component can contain only Integer and the user wants to type 3.14, the "." will not be enabled.

Author

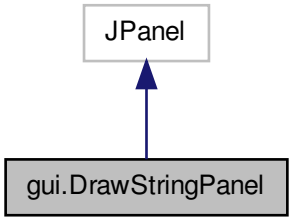
Zeid Kootbally *zeid.kootbally@nist.gov*

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

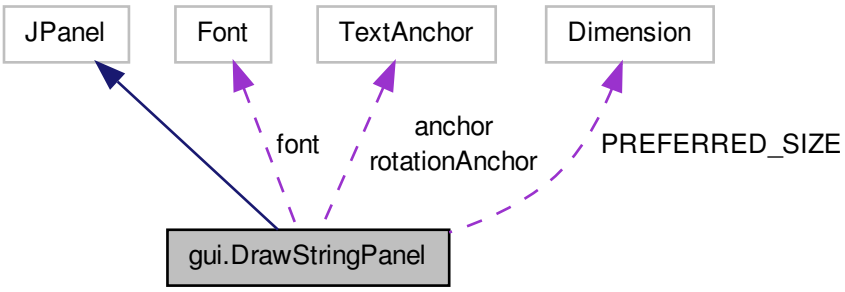
- src/tools/IntFilter.java

8.14 gui.DrawStringPanel Class Reference

Inheritance diagram for gui.DrawStringPanel:



Collaboration diagram for gui.DrawStringPanel:



Public Member Functions

- DrawStringPanel (String s, boolean flag)
- Dimension getPreferredSize ()
- void setAnchor (TextAnchor textanchor)
- void setRotationAnchor (TextAnchor textanchor)
- void setAngle (double d)
- Font getFont ()
- void setFont (Font font1)
- void paintComponent (Graphics g)

Private Attributes

- boolean rotate
- String text
- TextAnchor anchor
- TextAnchor rotationAnchor
- Font font
- double angle

Static Private Attributes

- static final Dimension PREFERRED_SIZE = new Dimension(500, 300)

8.14.1 Constructor & Destructor Documentation

8.14.1.1 gui.DrawStringPanel.DrawStringPanel (String *s*, boolean *flag*)

8.14.2 Member Function Documentation

8.14.2.1 Font gui.DrawStringPanel.getFont ( )

8.14.2.2 Dimension gui.DrawStringPanel.getPreferredSize ( )

8.14.2.3 void gui.DrawStringPanel.paintComponent (Graphics *g*)

8.14.2.4 void gui.DrawStringPanel.setAnchor (TextAnchor *textanchor*)

8.14.2.5 void gui.DrawStringPanel.setAngle (double *d*)

8.14.2.6 void gui.DrawStringPanel.setFont (Font *font1*)

8.14.2.7 void gui.DrawStringPanel.setRotationAnchor (TextAnchor *textanchor*)

8.14.3 Member Data Documentation

8.14.3.1 TextAnchor gui.DrawStringPanel.anchor [private]

8.14.3.2 double gui.DrawStringPanel.angle [private]

8.14.3.3 Font gui.DrawStringPanel.font [private]

8.14.3.4 final Dimension gui.DrawStringPanel.PREFERRED_SIZE = new Dimension(500, 300) [static, private]

8.14.3.5 boolean gui.DrawStringPanel.rotate [private]

8.14.3.6 TextAnchor gui.DrawStringPanel.rotationAnchor [private]

8.14.3.7 String gui.DrawStringPanel.text [private]

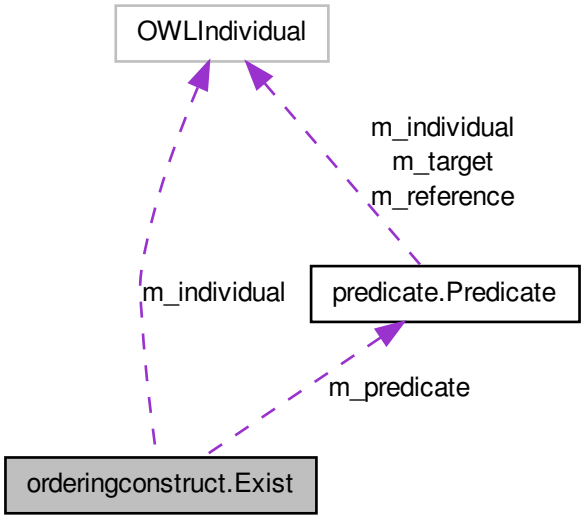
The documentation for this class was generated from the following file:

- src/gui/DrawStringPanel.java

8.15 orderingconstruct.Exist Class Reference

A state relationship that must exist.

Collaboration diagram for orderingconstruct.Exist:



Public Member Functions

- Exist ()
- int getTotalNumber ()
- void setTotalNumber (int total_number_)

  Set the total number Exist elements to an intention.
- OWLIndividual getIndividual ()

  Return the Exist element that is an OWLIndividual.
- void setIndividual (OWLIndividual individual_)

  Set an Exist element as an OWLIndividual.
- int getOccurrence ()

  Return the occurrence that an Exist element appears in an intention.
- void setOccurrence (int occurrence_)

  Set the occurrence an Exist element appears in an intention.
- int getPosition ()

  Return the position of an Exist element in the definition of an intention.
- void setPosition (int position_)

  Set the position of an Exist element within an intention.
- Predicate getPredicate ()

  Return the predicate for an Exist element.
- void setPredicate (Predicate predicate_)

  Set a predicate for an Exist element.

Private Attributes

- int m_occurrence

  Occurrence of an Exist element in an intention.
- OWLIndividual m_individual

  OWLIndividual is used to retrieve range individuals from the ontology given the data and object properties.
- int m_position

  Position of an Exist element within an OrderedList element.
- int m_total_number

  Total number of Exist element within an intention.
- Predicate m_predicate

  Predicate element for an Exist element.

8.15.1 Detailed Description

A state relationship that must exist.

Author

Zeid Kootbally *zeid.kootbally@nist.gov*

8.15.2 Constructor & Destructor Documentation

8.15.2.1 orderingconstruct.Exist.Exist ( )

Class constructor

8.15.3 Member Function Documentation

8.15.3.1 OWLIndividual orderingconstruct.Exist.getIndividual ( )

Return the Exist element that is an OWLIndividual.

Returns

Exist.m_individual

8.15.3.2 int orderingconstruct.Exist.getOccurrence ( )

Return the occurrence that an Exist element appears in an intention.

Returns

Exist.m_occurrence

8.15.3.3 int orderingconstruct.Exist.getPosition ( )

Return the position of an Exist element in the definition of an intention.

Returns

Exist.m_position

8.15.3.4 Predicate orderingconstruct.Exist.getPredicate ( )

Return the predicate for an Exist element.

Returns

Exist.m_predicate

8.15.3.5 int orderingconstruct.Exist.getTotalNumber ( )

8.15.3.6 void orderingconstruct.Exist.setIndividual (OWLIndividual *individual_*)

Set an Exist element as an OWLIndividual.

Parameters

*individual_*	OWLIndividual to set to Exist.m_individual

8.15.3.7 void orderingconstruct.Exist.setOccurrence (int *occurrence_*)

Set the occurrence an Exist element appears in an intention.

Parameters

*occurrence_*	Occurrence to set to Exist.m_occurrence

8.15.3.8 void orderingconstruct.Exist.setPosition (int *position_*)

Set the position of an Exist element within an intention.

Parameters

*position_*	Position to set to Exist.m_position_

8.15.3.9 void orderingconstruct.Exist.setPredicate (Predicate *predicate_*)

Set a predicate for an Exist element.

Parameters

*predicate_*	Predicate to set to Exist.m_predicate

8.15.3.10 void orderingconstruct.Exist.setTotalNumber (int *total_number_*)

Set the total number Exist elements to an intention.

Parameters

*total_number_*	Number of Exist elements to set to Exist.m_total_number

8.15.4 Member Data Documentation

8.15.4.1 OWLIndividual orderingconstruct.Exist.m_individual [private]

OWLIndividual is used to retrieve range individuals from the ontology given the data and object properties.

8.15.4.2 int orderingconstruct.Exist.m_occurrence [private]

Occurrence of an Exist element in an intention.

8.15.4.3 int orderingconstruct.Exist.m_position [private]

Position of an Exist element within an OrderedList element.

8.15.4.4 Predicate orderingconstruct.Exist.m_predicate [private]

Predicate element for an Exist element.

8.15.4.5 int orderingconstruct.Exist.m_total_number [private]

Total number of Exist element within an intention.

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

- src/orderingconstruct/Exist.java

8.16 tools.FileOperator Class Reference

Public Member Functions

- FileOperator ()
- String[] openFile (String path) throws IOException
- int readLines (String path) throws IOException
- ArrayList< ArrayList< ArrayList< String > > > translatePlanToStateRelation (String[] plan)

Static Public Member Functions

- static void saveAllKitsData (String file_path_, FileWriter writer, String kit_, String plan_) throws IOException

8.16.1 Detailed Description

Author

zeid

8.16.2 Constructor & Destructor Documentation

8.16.2.1 tools.FileOperator.FileOperator ( )

8.16.3 Member Function Documentation

8.16.3.1 String [] tools.FileOperator.openFile (String *path*) throws IOException

8.16.3.2 int tools.FileOperator.readLines (String *path*) throws IOException

8.16.3.3 static void tools.FileOperator.saveAllKitsData (String *file_path_*, FileWriter *writer*, String *kit_*, String *plan_*) throws IOException [static]

8.16.3.4 ArrayList<ArrayList<ArrayList<String> > > tools.FileOperator.translatePlanToStateRelation (String[] *plan*)

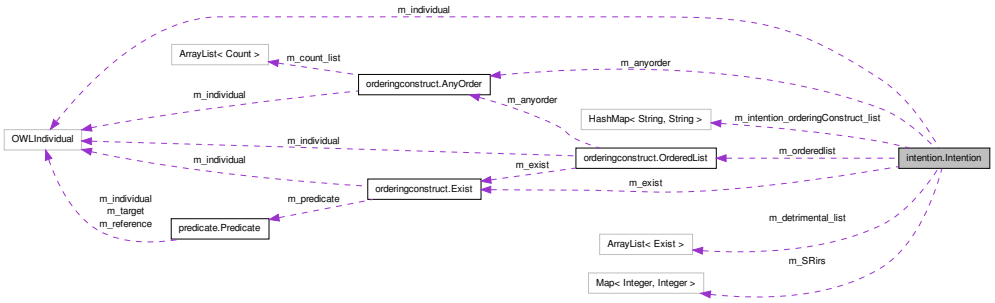
The documentation for this class was generated from the following file:

- src/tools/FileOperator.java

8.17 intention.Intention Class Reference

Representation of intentions from their definition in the ontology.

Collaboration diagram for intention.Intention:



Public Member Functions

- Intention ()

  Class constructor.
- void builDetrimentalList (Exist _exist)

  Add an ordering construct of type Exist to the list of detrimental state relations.
- ArrayList< Exist > getDetrimentalList ()

Return the list of detrimental state relations for an intention.

- OWLIndividual getIndividual ()

  Return an intention of type OWLIndividual.
- String getIntentionName ()

  Return the name of the intention.
- int getNumberStateRelation ()

  Return the number of state relations that constitute an intention.
- OrderedList getOrderedList ()

  Return the ordering construct OrderedList for an intention.
- double getM_am1 ()

  Return the value of AM_1.
- double getM_am2 ()

  Return the value of AM_2.
- double getM_am3 ()

  Return the value of AM_3.
- double getM_am4 ()

  Return the value of AM_4.
- double getM_am5 ()

  Return the value of AM_5.
- double getM_mm1 ()

  Return the value of MM_1.
- double getM_percentComplete_i_s ()

  Return the percentage of completion.
- double getM_percentProductive_i_s ()

  Return the percentage of productive states.
- int getM_SR_i_s ()

  Return the number of matched state relations (SR) in an intention (i) as of the current state (s)
- int getM_SR_Total ()

  Return the number of state relations (SR) (whether matched or not) in an intention (i)
- Map< Integer, Integer > getM_map_SRirs ()

  Return the HashMap that contains $SR_{i,r,s}$.
- int getM_Si ()

  Return the number of states (S) that have occurred since (and including) the first matched state relation in an intention (i)
- int getM_SR_i_r_s ()

  Return the value of $SR_{i,r,s}$ *for the current intention.*
- int getM_found_detrimental_SR ()

  Return the number of detrimental state relations found for an intention.
- int getM_intention_number ()

  Return the ID of an intention.
- double getM_likelihood_observation ()

  Return the likelihood of observation under a kit.
- double getM_probability_kit_observation ()

Return the probability of a kit given observations.

- void setExist (Exist exist_)

  Set the instance of the Java Class Exist to an intention.
- void setIntentionName (String intention_name_)

  Set the name of the intention.
- void setDetrimentalList (ArrayList< Exist > detrimental_list_)

  Set the list of detrimental state relations.
- void setIndividual (OWLIndividual individual_)

  Set the intention as an OWL Individual.
- void setNumberStateRelation (int number_state_relation_)

  Set the number of state relations to an intention.
- void setOrderedList (OrderedList orderedlist_)

  Set the instance of Ordered List to the intention.
- void setM_found_detrimental_SR (int found_detrimental_SR_)

  Set the number of detrimental state relations.
- void setM_am1 (double am1_)

  Set the value for AM_1.
- void setM_am2 (double am2_)

  Set the value for AM_2.
- void setM_am3 (double am3_)

  Set the value for AM_3.
- void setM_am4 (double am4_)

  Set the value for AM_4.
- void setM_am5 (double am5_)

  Set the value for AM_5.
- void setM_mm1 (double mm1_)

  Set the value for MM_1.
- void setM_percentComplete_i_s (double percentComplete_i_s_)

  Set the percentage of completion.
- void setM_percentProductive_i_s (double percentProductive_i_s_)

  Set the percentage of productive states.
- void setM_SR_i_s (int SR_i_s_)

  Set the number of matched state relations (SR*) in an intention (* i*) as of the current state (* s*)*
- void setM_SR_Total (int SR_Total_)

  Set the number of state relations (SR*) (whether matched or not) in an intention (* i*)*
- void setM_Si (int Si_)

  Set the number of states (S*) that have occurred since (and including) the first matched state relation in an intention (* i*)*
- void setM_map_SRirs (Map< Integer, Integer > SRirs_)

  Set a HashMap to m_SRirs.
- void setM_SR_i_r_s (int SR_i_r_s_)

  Set the value of $SR_{i,r,s}$ *for the current intention.*
- void setM_intention_number (int intention_number_)

Set an ID to an intention.

- void setM_likelihood_observation (double likelihood_observation_)

  Set the likelihood of observation under a kit.
- void setM_probability_kit_observation (double probability_kit_observation_)

  Set the probability of a kit given observations.

Static Public Member Functions

- static String getM_built_kit ()

  Return the kit chosen by the user in gui.OptionFrame.
- static String getM_selected_plan ()

  Get the plan chosen by the user in gui.OptionFrame.
- static void setM_built_kit (String built_kit_)

  Set the kit the user chose to build in gui.OptionFrame.
- static void setM_selected_plan (String selected_plan_)

  Set the value of m_selected_plan from the plan selected by the user.

Public Attributes

- HashMap< String, String > m_intention_orderingConstruct_list

Private Attributes

- ArrayList< Exist > m_detrimental_list

  List that contains detrimental states.
- Exist m_exist

  Instance of the ordering construct Exist.
- String m_intention_name

  Name of the intention.
- AnyOrder m_anyorder

  Instance of the ordering construct AnyOrder.
- OrderedList m_orderedlist

  Instance of the ordering construct OrderedList.
- OWLIndividual m_individual

  OWLIndividual (instance of the class Intention)
- int m_number_state_relation

  Number of state relations that consist an intention.
- int m_SR_Total

  Number of state relations (SR) (whether matched or not) in an intention (i)
- int m_Si

  Number of states (S) that have occurred since (and including) the first matched state relation in an intention (i)
- int m_found_detrimental_SR

Number of detrimental state relations ($detrimentalSR$) that have occurred in an intention (i) as of the current state (s).

- double m_am1

  AM_1:Number of observed state relations that are true in an intention (compared to other intentions).

- double m_am2

  AM_2:Percentage of an intention that is complete.

- double m_am3

  AM_3:Number of productive states that have occurred since the first productive state relation in an intention.

- double m_am4

  AM_4:Number of productive states that have occurred (recently) in the past r states.

- double m_am5

  AM_5:Probability of an intention (i) being recognized based on an observation (j).

- double m_mm1

  MM_1: The number of detrimental states.

- double m_percentComplete_i_s

  Percent of productive states for intention i as of state s.

- double m_percentProductive_i_s

  The number of matched state relations (SR) in an intention (i) as of the current state (s)

- int m_SR_i_s

  Number of matched state relations (SR) in the past r states in an intention (i) as of the current state (s).
  In other words, in the most recent (r) states, how many true state relations for an intention exist?

- int m_SR_i_r_s
- int m_intention_number

  An ID given to an intention, for instance, $a_4b_4c_2$ is associated to the ID #2. IDs are given to intentions in ontology.Ontology.sortIntentionList()

- double m_likelihood_observation

  Likelihood for computing AM_5.

- double m_probability_kit_observation

  Probability for computing AM_5.

- Map< Integer, Integer > m_SRirs = new HashMap<Integer, Integer>()

  A HashMap where the key is the state and the value is the number of state relations found in the current state.

Static Private Attributes

- static String m_built_kit

  The kit (intention) that the user chose to build.

- static String m_selected_plan

  The plan the user chose for a given kit (intention)

8.17.1 Detailed Description

Representation of intentions from their definition in the ontology.

This class consists of functions to represent intentions from their definition in the ontology. The ontology is read using OWL API tools and the description of each intention is stored in a list.

Author

zeidk

Date

2013/01/01

Contact: zeid.kootbally@nist.gov

8.17.2 Constructor & Destructor Documentation

8.17.2.1 intention.Intention.Intention ( )

Class constructor.

Instantiate the list of ordering constructs for an intention

Instantiate a list of forbidden ordering constructs (list of detrimental states)

8.17.3 Member Function Documentation

8.17.3.1 void intention.Intention.builDetrimentalList (Exist *_exist*)

Add an ordering construct of type Exist to the list of detrimental state relations.

Parameters

_exist	An ordering construct of type Exist

8.17.3.2 ArrayList<Exist> intention.Intention.getDetrimentalList ( )

Return the list of detrimental state relations for an intention.

Returns

Intention.m_detrimental_list

8.17.3.3 OWLIndividual intention.Intention.getIndividual ( )

Return an intention of type OWLIndividual.

Returns

Intention.m_individual

8.17.3.4 String intention.Intention.getIntentionName ( )

Return the name of the intention.

Returns

Intention.m_intention_name

8.17.3.5 double intention.Intention.getM_am1 ( )

Return the value of AM_1.

Returns

Intention.m_am1

8.17.3.6 double intention.Intention.getM_am2 ( )

Return the value of AM_2.

Returns

Intention.m_am2

8.17.3.7 double intention.Intention.getM_am3 ( )

Return the value of AM_3.

Returns

Intention.m_am3

8.17.3.8 double intention.Intention.getM_am4 ( )

Return the value of AM_4.

Returns

Intention.m_am4

8.17.3.9 double intention.Intention.getM_am5 ( )

Return the value of AM_5.

Returns

Intention.m_am5

8.17.3.10 static String intention.Intention.getM_built_kit ( ) [static]

Return the kit chosen by the user in gui.OptionFrame.

Returns

Intention.m_built_kit

8.17.3.11 int intention.Intention.getM_found_detrimental_SR ( )

Return the number of detrimental state relations found for an intention.

Returns

Intention.m_found_detrimental_SR

8.17.3.12 int intention.Intention.getM_intention_number ( )

Return the ID of an intention.

Returns

Intention.m_intention_number

8.17.3.13 double intention.Intention.getM_likelihood_observation ( )

Return the likelihood of observation under a kit.

Returns

Intention.m_likelihood_observation

8.17.3.14 Map<Integer, Integer> intention.Intention.getM_map_SRirs ( )

Return the HashMap that contains $SR_{i,r,s}$.

Returns

Intention.m_SRirs

8.17.3.15 double intention.Intention.getM_mm1 ( )

Return the value of MM_1.

Returns

Intention.m_mm1

8.17.3.16 double intention.Intention.getM_percentComplete_i_s ( )

Return the percentage of completion.

Returns

Intention.m_percentComplete_i_s

8.17.3.17 double intention.Intention.getM_percentProductive_i_s ( )

Return the percentage of productive states.

Returns

Intention.m_percentProductive_i_s

8.17.3.18 double intention.Intention.getM_probability_kit_observation ( )

Return the probability of a kit given observations.

Returns

Intention.m_probability_kit_observation

8.17.3.19 static String intention.Intention.getM_selected_plan ( ) [static]

Get the plan chosen by the user in gui.OptionFrame.

Returns

Intention.m_selected_plan

8.17.3.20 int intention.Intention.getM_Si ( )

Return the number of states (S) that have occurred since (and including) the first matched state relation in an intention (i)

Returns

Intention.m_Si

8.17.3.21 int intention.Intention.getM_SR_i_r_s ( )

Return the value of $SR_{i,r,s}$ for the current intention.

Returns

Intention.m_SR_i_r_s

8.17.3.22 int intention.Intention.getM_SR_i_s ( )

Return the number of matched state relations (SR) in an intention (i) as of the current state (s)

Returns

Intention.m_SR_i_s

8.17.3.23 int intention.Intention.getM_SR_Total ( )

Return the number of state relations (SR) (whether matched or not) in an intention (i)

Returns

Intention.m_SR_Total

8.17.3.24 int intention.Intention.getNumberStateRelation ( )

Return the number of state relations that constitute an intention.

Returns

Intention.m_number_state_relation

8.17.3.25 OrderedList intention.Intention.getOrderedList ( )

Return the ordering construct OrderedList for an intention.

Returns

Intention.m_orderedlist

8.17.3.26 void intention.Intention.setDetrimentalList (ArrayList< Exist > *detrimental_list_*)

Set the list of detrimental state relations.

Parameters

*detrimental_list_*	Value to set to Intention.m_detrimental_list

8.17.3.27 void intention.Intention.setExist (Exist *exist_*)

Set the instance of the Java Class Exist to an intention.

Parameters

*exist_*	Value to set to Intention.m_exist

8.17.3.28 void intention.Intention.setIndividual (OWLIndividual *individual_*)

Set the intention as an OWL Individual.

Parameters

*individual_*	Value to set to Intention.m_individual

8.17.3.29 void intention.Intention.setIntentionName (String *intention_name_*)

Set the name of the intention.

Parameters

*intention_name_*	Value to set to Intention.m_intention_name

8.17.3.30 void intention.Intention.setM_am1 (double *am1_*)

Set the value for AM_1.

Parameters

*am1_*	Value to set to Intention.m_am1

8.17.3.31 void intention.Intention.setM_am2 (double *am2_*)

Set the value for AM_2.

Parameters

*am2_*	Value to set to Intention.m_am2

8.17.3.32 void intention.Intention.setM_am3 (double *am3_*)

Set the value for AM_3.

Parameters

*am3_*	Value to set to AM_3

8.17.3.33 void intention.Intention.setM_am4 (double *am4_*)

Set the value for AM_4.

Parameters

*am4_*	Value to set to AM_4

8.17.3.34 void intention.Intention.setM_am5 (double *am5_*)

Set the value for AM_5.

Parameters

*am5_*	Value to set to AM_5

8.17.3.35 static void intention.Intention.setM_built_kit (String *built_kit_*) [static]

Set the kit the user chose to build in gui.OptionFrame.

Parameters

*built_kit_*	Value to set to Intention.m_built_kit

8.17.3.36 void intention.Intention.setM_found_detrimental_SR (int *found_detrimental_SR_*)

Set the number of detrimental state relations.

Parameters

*found_detrimental_SR_*	Value to set to Intention.m_found_detrimental_SR

8.17.3.37 void intention.Intention.setM_intention_number (int *intention_number_*)

Set an ID to an intention.

Parameters

*intention_number_*	Value to set to Intention.m_intention_number

8.17.3.38 void intention.Intention.setM_likelihood_observation (double *likelihood_observation_*)

Set the likelihood of observation under a kit.

Parameters

*likelihood_observation_*	Value set to Intention.m_likelihood_observation

8.17.3.39 void intention.Intention.setM_map_SRirs (Map< Integer, Integer > *SRirs_*)

Set a HashMap to m_SRirs.

Parameters

*SRirs_*	Value to set to Intention.m_SRirs

8.17.3.40 void intention.Intention.setM_mm1 (double *mm1_*)

Set the value for MM_1.

Parameters

*mm1_*	Value to set to MM_1

8.17.3.41 void intention.Intention.setM_percentComplete_i_s (double *percentComplete_i_s_*)

Set the percentage of completion.

Parameters

*percentComplete_i_s_*	Value to set to Intention.m_percentComplete_i_s

8.17.3.42 void intention.Intention.setM_percentProductive_i_s (double *percentProductive_i_s_*)

Set the percentage of productive states.

Parameters

*percentProduc i_s_*	Value to set to Intention.m_percentProductive_i_s

8.17.3.43 void intention.Intention.setM_probability_kit_observation (double *probability_kit_observation_*)

Set the probability of a kit given observations.

Parameters

probability_- kit_- observation_-	Value to set to Intention.m_probability_kit_observation

8.17.3.44 static void intention.Intention.setM_selected_plan (String *selected_plan_*) [static]

Set the value of m_selected_plan from the plan selected by the user.

Parameters

*selected_- plan_*	Value to set to Intention.m_selected_plan

8.17.3.45 void intention.Intention.setM_Si (int *Si_*)

Set the number of states (S) that have occurred since (and including) the first matched state relation in an intention (i)

Parameters

*Si_*	The value to set to Intention.m_Si

8.17.3.46 void intention.Intention.setM_SR_i_r_s (int *SR_i_r_s_*)

Set the value of $SR_{i,r,s}$ for the current intention.

Parameters

*SR_i_r_s_*	Value to set to Intention.m_SR_i_r_s

8.17.3.47 void intention.Intention.setM_SR_i_s (int *SR_i_s_*)

Set the number of matched state relations (SR) in an intention (i) as of the current state (s)

Parameters

*SR_i_s_*	The value for Intention.m_SR_i_s

8.17.3.48 void intention.Intention.setM_SR_Total (int *SR_Total_*)

Set the number of state relations (SR) (whether matched or not) in an intention (i)

Parameters

*SR_Total_*	The value to set to Intention.m_SR_Total

8.17.3.49 void intention.Intention.setNumberStateRelation (int *number_state_relation_*)

Set the number of state relations to an intention.

Parameters

*number_state_relation_*	Value to set to Intention.m_number_state_relation

8.17.3.50 void intention.Intention.setOrderedList (OrderedList *orderedlist_*)

Set the instance of Ordered List to the intention.

Parameters

*orderedlist_*	Value to set to Intention.m_orderedlist

8.17.4 Member Data Documentation

8.17.4.1 double intention.Intention.m_am1 [private]

AM_1:*Number of observed state relations that are true in an intention (compared to other intentions).*

The formula for this additive metric for intention i in state s is:

$$AM_{1,i,s} = \frac{SR_{i,s}}{SR_{all,s}} = \frac{SR_{i,s}}{\sum_{i=1}^{p} SR_{i,s}}$$

- $SR_{i,s}$: The number of matched state relations (SR) in an intention (i) as of the current state (s).
- $SR_{all,s}$: The number of matched state relations (SR) in all possible intentions as of the current state (s).

This formula represents the ratio of true states that are in intention i to the sum of all of the true states in all of intentions of interest.

The variable p represents the number of intentions of interest. It is evaluated for every intention of interest at every state.

8.17.4.2 double intention.Intention.m_am2 [private]

AM_2:*Percentage of an intention that is complete.*

The formula for the percentage complete for intention i in state s is:

$$PercentComplete_{i,s} = \frac{SR_{i,s}}{SR_{i,total}}$$

- $SR_{i,s}$: The number of matched state relations (SR) in an intention (i) as of the current state (s).
- $SR_{i,total}$: The number of state relations (SR) (whether matched or not) in an intention (i).

We then normalize this for all intentions of interest to find the additive metric 2 for intention *i* in state *s*.

$$AM_{2,i,s} = \frac{PercentComplete_{i,s}}{\sum_{i=1}^{p} PercentComplete_{i,s}}$$

8.17.4.3 double intention.Intention.m_am3 [private]

AM_3:*Number of productive states that have occurred since the first productive state relation in an intention.*

The formula for the percentage complete for intention i in state s is:

$$PercentComplete_{i,s} = \frac{SR_{i,s}}{S_i}$$

- $SR_{i,s}$: The number of matched state relations (SR) in an intention (i) as of the current state (s).

- S_i: The number of states (S) that have occurred since (and including) the first matched state relation in an intention (i).

We then normalize this for all intentions by determining additive metric 3 for intention i in state s.

$$AM_{3,i,s} = \frac{PercentProductive_{i,s}}{\sum_{i=1}^{p} PercentProductive_i}$$

8.17.4.4 double intention.Intention.m_am4 [private]

AM_4:*Number of productive states that have occurred (recently) in the past r states.*

The formula for AM_4 is:

$$AM_{4,i,s} = \frac{SR_{i,r,s}}{\sum_{i=1}^{p} SR_{i,r,s}}$$

- $SR_{i,r,s}$: The number of matched state relations (SR) in the past r states in an intention (i) as of the current state (s).

  In other words, in the most recent (r)states, how many true state relations for an intention exist?

8.17.4.5 double intention.Intention.m_am5 [private]

AM_5:*Probability of an intention (i) being recognized based on an observation (j).*

- Suppose a kit is described by the number of parts it contains for each type.

  That is, $kit_i = (n_{i_A}, n_{i_B}, \ldots, n_{i_Q})$ has n_{i_A} parts of type "A", n_{i_B} parts of type "B", ..., n_{i_Q} parts of type "Q".

- Suppose an observation is described by the number of parts seen for each type.

  That is, $observation_j = (x_{j_A}, x_{j_B}, \ldots, x_{j_Q})$ has seen x_{j_A} parts of type "A", x_{j_B} parts of type "B",..., x_{j_Q} parts of type "Q".

- The likelihood L of observation j under kit i is given by the multivariate hypergeometric distribution:

$$L(observation_j | kit_i) = \frac{\prod_{p=1}^{q} n_{i_p} choose\ x_{j_p}}{\sum_{p=1}^{q} n_{i_p} choose \sum_{p=1}^{q} x_{j_p}}$$

The additive metric AM_5 for kit_i is the probability of kit_i given $observation_j$:

$$Probability(kit_i | observation_j) = \frac{L(observation_j | kit_i)}{\sum_{n=1}^{N} L(observation_j | kit_n)}$$

where n is the total number of kits that are likely to be built.

8.17.4.6 AnyOrder intention.Intention.m_anyorder [private]

Instance of the ordering construct AnyOrder.

8.17.4.7 String intention.Intention.m_built_kit [static, private]

The kit (intention) that the user chose to build.

8.17.4.8 ArrayList<Exist> intention.Intention.m_detrimental_list [private]

List that contains detrimental states.

8.17.4.9 Exist intention.Intention.m_exist [private]

Instance of the ordering construct Exist.

8.17.4.10 int intention.Intention.m_found_detrimental_SR [private]

Number of detrimental state relations ($detrimentalSR$) that have occurred in an intention (i) as of the current state (s).

Detrimental state relations are state relations that are explicitly prohibited in an intention.

8.17.4.11 OWLIndividual intention.Intention.m_individual [private]

OWLIndividual (instance of the class Intention)

8.17.4.12 String intention.Intention.m_intention_name [private]

Name of the intention.

8.17.4.13 int intention.Intention.m_intention_number [private]

An ID given to an intention, for instance, $a_4b_4c_2$ is associated to the ID #2. IDs are given to intentions in ontology.Ontology.sortIntentionList()

8.17.4.14 HashMap<String, String> intention.Intention.m_intention_orderingConstruct_list

8.17.4.15 double intention.Intention.m_likelihood_observation [private]

Likelihood for computing AM_5.

8.17.4.16 double intention.Intention.m_mm1 [private]

MM_1: *The number of detrimental states*.

It was chosen to be a multiplicative metric because the presence of detrimental states should play a larger role in the overall likelihood of an intention as compared to the additive metrics above.

The formula for MM_1 is:

$$MM_{1,i,s} = \frac{SR_{i,s} - detrimentalSR_{i,s}}{SR_{i,s}}$$

- $SR_{i,s}$: The number of matched state relations (SR) in an intention (i) as of the current state (s).
- $detrimentalSR_{i,s}$: The number of detrimental states relations that have occurred in intention i as of the current state s.

Percent complete for intention i in state s

8.17.4.17 int intention.Intention.m_number_state_relation [private]

Number of state relations that consist an intention.

8.17.4.18 OrderedList intention.Intention.m_orderedlist [private]

Instance of the ordering construct OrderedList.

8.17.4.19 double intention.Intention.m_percentComplete_i_s [private]

Percent of productive states for intention i as of state s.

8.17.4.20 double intention.Intention.m_percentProductive_i_s [private]

The number of matched state relations (SR) in an intention (i) as of the current state (s)

8.17.4.21 double intention.Intention.m_probability_kit_observation [private]

Probability for computing AM_5.

8.17.4.22 String intention.Intention.m_selected_plan [static, private]

The plan the user chose for a given kit (intention)

8.17.4.23 int intention.Intention.m_Si [private]

Number of states (S) that have occurred since (and including) the first matched state relation in an intention (i)

8.17.4.24 int intention.Intention.m_SR_i_r_s [private]

8.17.4.25 int intention.Intention.m_SR_i_s [private]

Number of matched state relations (SR) in the past r states in an intention (i) as of the current state (s).

In other words, in the most recent (r) states, how many true state relations for an intention exist?

8.17.4.26 int intention.Intention.m_SR_Total [private]

Number of state relations (SR) (whether matched or not) in an intention (i)

8.17.4.27 Map<Integer, Integer> intention.Intention.m_SRirs = new HashMap<Integer, Integer>() [private]

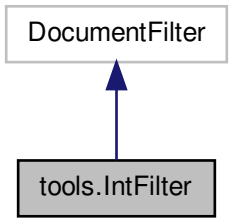
A HashMap where the key is the state and the value is the number of state relations found in the current state.

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

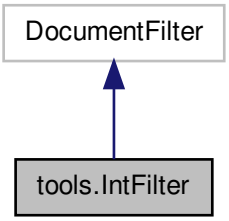
- src/intention/Intention.java

8.18 tools.IntFilter Class Reference

Inheritance diagram for tools.IntFilter:



Collaboration diagram for tools.IntFilter:



Public Member Functions

- void insertString (FilterBypass fb, int offset, String string, AttributeSet attr) throws BadLocationException
- void replace (FilterBypass fb, int offset, int length, String text, AttributeSet attrs) throws BadLocationException
- void remove (FilterBypass fb, int offset, int length) throws BadLocationException

Private Member Functions

- boolean test (String text)

8.18.1 Member Function Documentation

8.18.1.1 void tools.IntFilter.insertString (FilterBypass *fb,* int *offset,* String *string,* AttributeSet *attr*) throws BadLocationException

8.18.1.2 void tools.IntFilter.remove (FilterBypass *fb,* int *offset,* int *length*) throws BadLocationException

8.18.1.3 void tools.IntFilter.replace (FilterBypass *fb,* int *offset,* int *length,* String *text,* AttributeSet *attrs*) throws BadLocationException

8.18.1.4 boolean tools.IntFilter.test (String *text*) [private]

Return true if the text entered contains an Integer, false otherwise

Parameters

text	The text that is checked

Returns

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

- src/tools/IntFilter.java

8.19 main.Launcher Class Reference

Main class of the tool.

Public Member Functions

- Launcher ()

  Constructor.

Static Public Member Functions

- static void main (String[] args) throws OWLException, InterruptedException, InvocationTargetException, ClassNotFoundException, InstantiationException, IllegalAccessException, IOException

  Main of the project. The main file allows:
- static void enabler ()

  Enable some gui components from the class gui.Mainframe.

8.19.1 Detailed Description

Main class of the tool.

Author

Zeid Kootbally *zeid.kootbally@nist.gov*

Precondition

Make sure the kits directory is present in the same directory as this tool
Make sure kittingClasses.owl, kittingInstances_ir.owl, and soap.owl are in the same directory as this tool

8.19.2 Constructor & Destructor Documentation

8.19.2.1 main.Launcher.Launcher ( )

Constructor.

8.19.3 Member Function Documentation

8.19.3.1 static void main.Launcher.enabler ( ) [static]

Enable some gui components from the class gui.Mainframe.

8.19.3.2 static void main.Launcher.main (String[] *args*) throws OWLException, InterruptedException, InvocationTargetException, ClassNotFoundException, InstantiationException, IllegalAccessException, IOException [static]

Main of the project. The main file allows:

- The creation of an object for the class Ontology
- Set the OWLAPI manager
- Initialize all the array lists that are used in the project
- Load the ontology
- Set the OWLAPI reasoner
- Set the OWLAPI data factory
- Parse the ontology to retrieve information on each intention

Parameters

args	

Exceptions

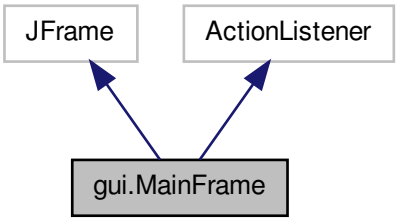
OWLException	
InterruptedException	
InvocationTargetException	
UnsupportedLookAndFeelException	
IllegalAccessException	
InstantiationException	
ClassNotFoundException	
IOException	

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

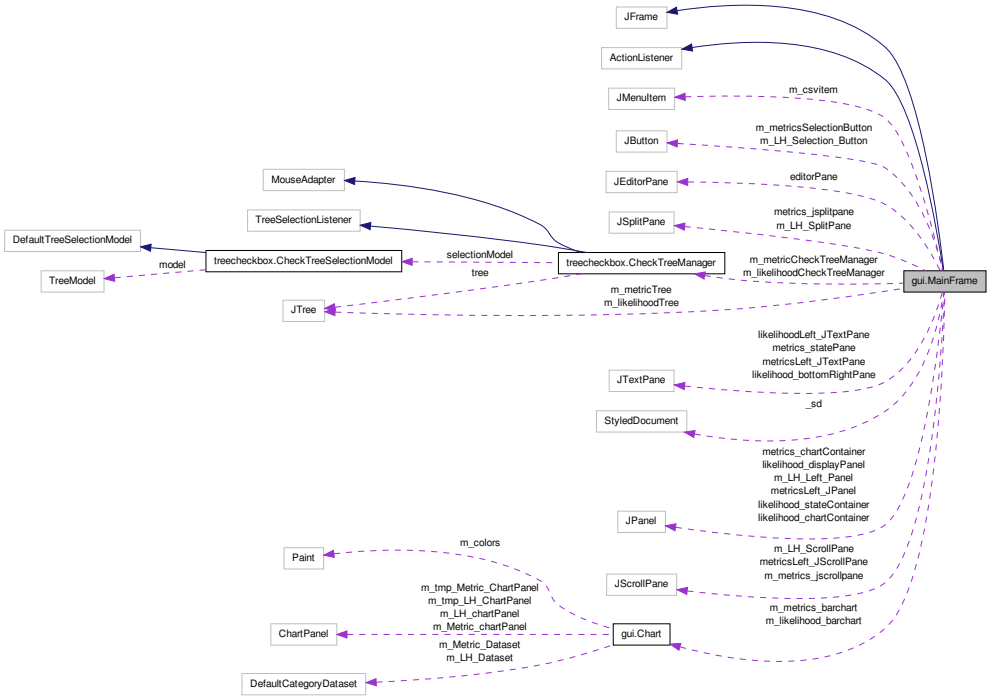
- src/main/Launcher.java

8.20 gui.MainFrame Class Reference

Inheritance diagram for gui.MainFrame:



Collaboration diagram for gui.MainFrame:



Classes

- class DisplayMetrics

Public Member Functions

- MainFrame (String s, JTree metricTree, JTree likelihoodTree)
- void actionPerformed (ActionEvent actionevent)
- void saveToCSV (String path)

  Create the file file and write intention dataset.
- void showErrorMessage (String title, String message)

Static Public Member Functions

- static void updateStateRelationPanel (String s, Color c, JTextPane textpane)

Static Public Attributes

- static final String EXIT_COMMAND = "EXIT"
- static JTextPane likelihood_bottomRightPane = new JTextPane()
- static JTextPane metrics_statePane = new JTextPane()
- static JTextPane likelihoodLeft_JTextPane = new JTextPane()
- static JTextPane metricsLeft_JTextPane = new JTextPane()
- static JScrollPane metricsLeft_JScrollPane = new JScrollPane()
- static JPanel metricsLeft_JPanel = new JPanel()
- static JPanel m_LH_Left_Panel = new JPanel()
- static JPanel metrics_chartContainer
- static JButton m_metricsSelectionButton
- static JButton m_LH_Selection_Button
- static JTree m_metricTree
- static JTree m_likelihoodTree
- static JMenuItem m_csvitem = new JMenuItem("Export to CSV...", 112)

Package Attributes

- CheckTreeManager m_metricCheckTreeManager
- CheckTreeManager m_likelihoodCheckTreeManager

Private Member Functions

- JPanel buildMetricPanel (JTree metricTree)
- JPanel buildLikelihoodPanel (JTree likelihoodTree)

  Build the main panel that contains components for the likelihoods.
- JComponent createContent (JTree metricTree, JTree likelihoodTree)

- void lh_Button_ActionPerformed (java.awt.event.ActionEvent evt)
- JMenuBar createMenuBar ()
- void copyToClipboard ()
- void applyThemeToChart ()
- void exportToPDF ()
- void exportToCSV ()

  Export likelihoods result to a csv file.
- void attemptExit ()
- JPanel createLikelihoodChartPanel ()

  Create the chart panel that will contain the likelihood chart.
- JPanel createMetricsChartPanel ()

  Build the JPanel that will contain the chart displaying metrics values.

Private Attributes

- JPanel likelihood_displayPanel
- JPanel likelihood_chartContainer
- JPanel likelihood_stateContainer
- JEditorPane editorPane
- Chart m_likelihood_barchart = new Chart()
- Chart m_metrics_barchart = new Chart()

Static Private Attributes

- static final long serialVersionUID = 1384873058265918162L
- static StyledDocument _sd
- static JSplitPane metrics_jsplitpane
- static JSplitPane m_LH_SplitPane
- static JScrollPane m_metrics_jscrollpane = new JScrollPane()
- static JScrollPane m_LH_ScrollPane = new JScrollPane()

8.20.1 Constructor & Destructor Documentation

8.20.1.1 gui.MainFrame.MainFrame (String *s*, JTree *metricTree*, JTree *likelihoodTree*)

8.20.2 Member Function Documentation

8.20.2.1 void gui.MainFrame.actionPerformed (ActionEvent *actionevent*)

8.20.2.2 void gui.MainFrame.applyThemeToChart ( ) [private]

8.20.2.3 void gui.MainFrame.attemptExit ( ) [private]

8.20.2.4 JPanel gui.MainFrame.buildLikelihoodPanel (JTree *likelihoodTree*) [private]

Build the main panel that contains components for the likelihoods.

Parameters

likelihoodTree	The JTree that displays intentions

Returns

8.20.2.5 JPanel gui.MainFrame.buildMetricPanel (JTree *metricTree*) [private]

8.20.2.6 void gui.MainFrame.copyToClipboard ( ) [private]

8.20.2.7 JComponent gui.MainFrame.createContent (JTree *metricTree,* JTree *likelihoodTree*) [private]

8.20.2.8 JPanel gui.MainFrame.createLikelihoodChartPanel ( ) [private]

Create the chart panel that will contain the likelihood chart.

Returns

The likelihood panel.

8.20.2.9 JMenuBar gui.MainFrame.createMenuBar ( ) [private]

8.20.2.10 JPanel gui.MainFrame.createMetricsChartPanel ( ) [private]

Build the JPanel that will contain the chart displaying metrics values.

Returns

The JPanel that will contain the chart displaying metrics values.

8.20.2.11 void gui.MainFrame.exportToCSV ( ) [private]

Export likelihoods result to a csv file.

8.20.2.12 void gui.MainFrame.exportToPDF ( ) [private]

8.20.2.13 void gui.MainFrame.lh_Button_ActionPerformed (java.awt.event.ActionEvent *evt*) [private]

8.20.2.14 void gui.MainFrame.saveToCSV (String *path*)

Create the file *file* and write intention dataset.

Parameters

file	

8.20.2.15 void gui.MainFrame.showErrorMessage (String *title*, String *message*)

8.20.2.16 static void gui.MainFrame.updateStateRelationPanel (String *s*, Color *c*, JTextPane *textpane*) [static]

8.20.3 Member Data Documentation

8.20.3.1 StyledDocument gui.MainFrame._sd [static, private]

8.20.3.2 JEditorPane gui.MainFrame.editorPane [private]

8.20.3.3 final String gui.MainFrame.EXIT_COMMAND = "EXIT" [static]

8.20.3.4 JTextPane gui.MainFrame.likelihood_bottomRightPane = new JTextPane() [static]

8.20.3.5 JPanel gui.MainFrame.likelihood_chartContainer [private]

8.20.3.6 JPanel gui.MainFrame.likelihood_displayPanel [private]

8.20.3.7 JPanel gui.MainFrame.likelihood_stateContainer [private]

8.20.3.8 JTextPane gui.MainFrame.likelihoodLeft_JTextPane = new JTextPane() [static]

8.20.3.9 JMenuItem gui.MainFrame.m_csvitem = new JMenuItem("Export to CSV...", 112) [static]

8.20.3.10 JPanel gui.MainFrame.m_LH_Left_Panel = new JPanel() [static]

8.20.3.11 JScrollPane gui.MainFrame.m_LH_ScrollPane = new JScrollPane() [static, private]

8.20.3.12 JButton gui.MainFrame.m_LH_Selection_Button [static]

8.20.3.13 JSplitPane gui.MainFrame.m_LH_SplitPane [static, private]

8.20.3.14 Chart gui.MainFrame.m_likelihood_barchart = new Chart() [private]

8.20.3.15 CheckTreeManager gui.MainFrame.m_likelihoodCheckTreeManager [package]

8.20.3.16 JTree gui.MainFrame.m_likelihoodTree [static]

8.20.3.17 CheckTreeManager gui.MainFrame.m_metricCheckTreeManager [package]

8.20.3.18 Chart gui.MainFrame.m_metrics_barchart = new Chart() [private]

8.20.3.19 JScrollPane gui.MainFrame.m_metrics_jscrollpane = new JScrollPane() [static, private]

8.20.3.20 JButton gui.MainFrame.m_metricsSelectionButton [static]

8.20.3.21 JTree gui.MainFrame.m_metricTree [static]

8.20.3.22 JPanel gui.MainFrame.metrics_chartContainer [static]

8.20.3.23 JSplitPane gui.MainFrame.metrics_jsplitpane [static, private]

8.20.3.24 JTextPane gui.MainFrame.metrics_statePane = new JTextPane() [static]

8.20.3.25 JPanel gui.MainFrame.metricsLeft_JPanel = new JPanel() [static]

8.20.3.26 JScrollPane gui.MainFrame.metricsLeft_JScrollPane = new JScrollPane() [static]

8.20.3.27 JTextPane gui.MainFrame.metricsLeft_JTextPane = new JTextPane() [static]

8.20.3.28 final long gui.MainFrame.serialVersionUID = 1384873058265918162L [static, private]

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

- src/gui/MainFrame.java

8.21 intention.Metric Class Reference

Definition of additive and multiplicative metrics.

Static Public Member Functions

- static int get_AM1_Weight ()
- static int get_AM2_Weight ()
- static int get_AM3_Weight ()
- static int get_AM4_Weight ()
- static int get_AM5_Weight ()
- static void set_AM1_Weight (int weight)
- static void set_AM2_Weight (int weight)

- static void set_AM3_Weight (int weight)
- static void set_AM4_Weight (int weight)
- static void set_AM5_Weight (int weight)

Private Member Functions

- int get_MM1_Weight ()
- void set_MM1_Weight (int weight)

Static Private Attributes

- static int m_AM1_weight
- static int m_AM2_weight
- static int m_AM3_weight
- static int m_AM4_weight
- static int m_AM5_weight
- static int m_MM1_weight

8.21.1 Detailed Description

Definition of additive and multiplicative metrics.

This class consists of additive and multiplicative metrics definitions

Author

zeidk

Date

2013/01/01

Contact: zeid.kootbally@nist.gov

8.21.2 Member Function Documentation

8.21.2.1 static int intention.Metric.get_AM1_Weight ( ) [static]

Get the weight for AM1.

Returns

The weight for AM1.

8.21.2.2 static int intention.Metric.get_AM2_Weight ( ) [static]

Get the weight for AM2.

Returns

The weight for AM2.

8.21.2.3 static int intention.Metric.get_AM3_Weight ( ) [static]

Get the weight for AM3.

Returns

The weight for AM3.

8.21.2.4 static int intention.Metric.get_AM4_Weight ( ) [static]

Get the weight for AM4.

Returns

The weight for AM4.

8.21.2.5 static int intention.Metric.get_AM5_Weight ( ) [static]

Get the weight for AM5.

Returns

The weight for AM5.

8.21.2.6 int intention.Metric.get_MM1_Weight ( ) [private]

Get the weight for MM1.

Returns

The weight for MM1.

8.21.2.7 static void intention.Metric.set_AM1_Weight (int *weight*) [static]

Set the weight for AM1.

Parameters

weight	Value set to the weight for AM1.

8.21.2.8 static void intention.Metric.set_AM2_Weight (int *weight*) [static]

Set the weight for AM2.

Parameters

weight	Value set to the weight for AM2.

8.21.2.9 static void intention.Metric.set_AM3_Weight (int *weight*) [static]

Set the weight for AM3.

Parameters

weight	Value set to the weight for AM3.

8.21.2.10 static void intention.Metric.set_AM4_Weight (int *weight*) [static]

Set the weight for AM4.

Parameters

weight	Value set to the weight for AM4.

8.21.2.11 static void intention.Metric.set_AM5_Weight (int *weight*) [static]

Set the weight for AM5.

Parameters

weight	Value set to the weight for AM5.

8.21.2.12 void intention.Metric.set_MM1_Weight (int *weight*) [private]

Set the weight for MM1.

Parameters

weight	Value set to the weight for MM1.

8.21.3 Member Data Documentation

8.21.3.1 int intention.Metric.m_AM1_weight [static, private]

8.21.3.2 int intention.Metric.m_AM2_weight [static, private]

8.21.3.3 int intention.Metric.m_AM3_weight [static, private]

8.21.3.4 int intention.Metric.m_AM4_weight [static, private]

8.21.3.5 int intention.Metric.m_AM5_weight [static, private]

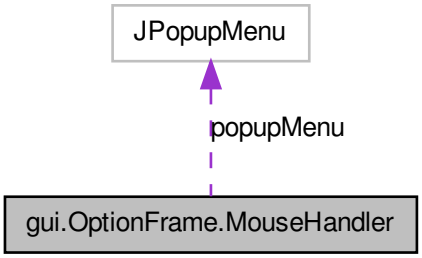
8.21.3.6 int intention.Metric.m_MM1_weight [static, private]

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

- src/intention/Metric.java

8.22 gui.OptionFrame.MouseHandler Class Reference

Collaboration diagram for gui.OptionFrame.MouseHandler:



Public Member Functions

- MouseHandler (javax.swing.JPopupMenu popup)
- void mousePressed (java.awt.event.MouseEvent e)
- void mouseReleased (java.awt.event.MouseEvent e)

Private Attributes

- javax.swing.JPopupMenu popupMenu

8.22.1 Constructor & Destructor Documentation

8.22.1.1 gui.OptionFrame.MouseHandler.MouseHandler (javax.swing.JPopupMenu *popup*)

8.22.2 Member Function Documentation

8.22.2.1 void gui.OptionFrame.MouseHandler.mousePressed (java.awt.event.MouseEvent *e*)

8.22.2.2 void gui.OptionFrame.MouseHandler.mouseReleased (java.awt.event.MouseEvent *e*)

8.22.3 Member Data Documentation

8.22.3.1 javax.swing.JPopupMenu gui.OptionFrame.MouseHandler.popupMenu [private]

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

- src/gui/OptionFrame.java

8.23 Ontology Class Reference

Class for the ontology.

8.23.1 Detailed Description

Class for the ontology.

This class is used to manipulate the ontology and extract data from it.

Author

zeidk

Date

2013/01/01

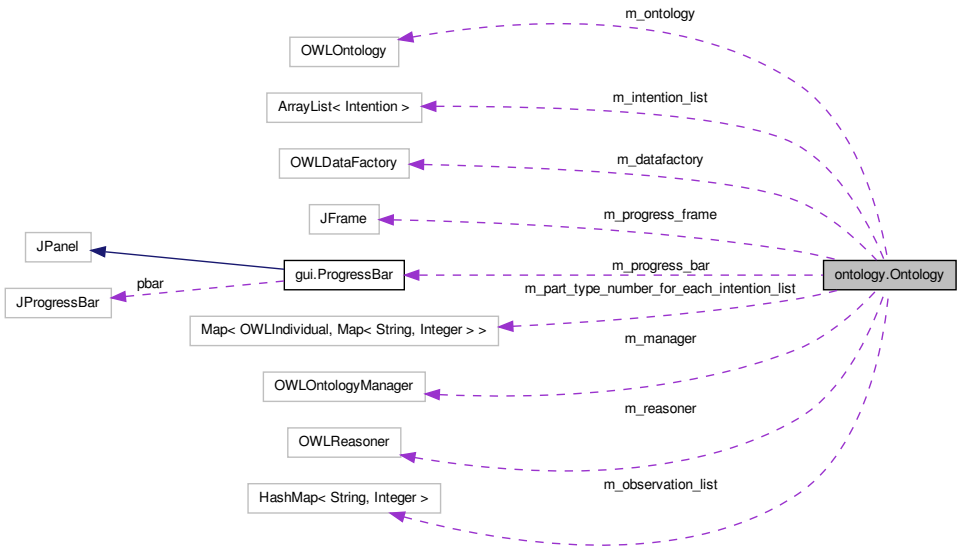
Contact: zeid.kootbally@nist.gov

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

- src/ontology/Ontology.java

8.24 ontology.Ontology Class Reference

Collaboration diagram for ontology.Ontology:



Public Member Functions

- Ontology ()

  Constructor of the Ontology class.
- String getOntologyPath ()

  Get the path of the ontology.
- String getRootClass ()

  Get the class root from the ontology.
- void setRootClass (String rootClass_)

  Set the class root from the ontology.
- Set< OWLClassExpression > getIndividualClass (OWLNamedIndividual individual)
- OWLOntologyManager getManager ()

  Simple getter.
- OWLOntology getOntology ()

  Simple getter.
- String getPath ()

  Simple getter.
- NodeSet< OWLClass > getSubclasses (String myClassName)
- void initializeList ()

- void loadFromFile ()

  Load the ontology from a file.
- void loadOntologyFromPath (String myPath) throws MalformedURLException, OWLException
- void sortIntentionList ()

  Read m_intention_list and re-arrange the list using a number for each intention.
- void parseIntention (NodeSet< OWLClass > myClass) throws InterruptedException, InvocationTargetException, IOException
- void showDialogBox ()
- JTree buildIntentionTree ()
- JTree buildIntentionMetricsTree ()
- void setDataFactory ()
- void setManager ()
- void setManager (OWLOntologyManager manager)

  Simple setter.
- void setOntology (OWLOntology ontology)

  Simple setter.
- void setPath (String path)

  Simple setter.
- void setReasoner (OWLOntology myOntology)

Static Public Member Functions

- static void setInstanceFilePath (String path_)

  Set the path to the OWL instance file The path is retrieved from the field OptionFrame.m_instance_txt_field.
- static String cleanIRI (Object entity)

  Return the name of the entity without the IRI.
- static double computeMetricAM1 (double _sr_i_s, double _sr_all)

  Compute the additive metric AM_1.
- static double computeMetricAM2 (double _percentComplete_i_s, double _percent_complete_total)

  Compute the additive metric AM_2.
- static double computeMetricAM3 (double percent_productive_i_s, double percent_productive_i)

  Compute the additive metric AM_3.
- static double computeMetricAM4 (double sr_i_r_s, double sum_sr_i_r_s)

  Compute the additive metric AM_4.
- static double computeMetricAM5 (Intention intention)

  Compute the additive metric AM_5.
- static double computeMetricMM1 (int sr_i_s, int detrimental)
- static void buildObservationList ()

  Build a list of the parts observed during kitting.
- static void updateObservationList (String part_type)

Update the observation list.

- static void readObservationList ()

  Read the observation list m_observation_list.
- static void chooseTest ()
- static void computeObservationLikelihood (Intention intention_)

  Compute the likelihood L of observation j under kit i.
- static void computeProbabilityFromObservation (Intention intention_)
- static int compute_sum_observation ()
- static int compute_sum_part_type (Intention intention_)
- static ArrayList< String > removeDuplicates (ArrayList< String > list_)

  Remove duplicates in an ArrayList of String.
- static double computePercentComplete (double sr_i_s, double sr_i_total)

  PercentComplete is the percentage complete for an intention in a state.
- static double computePercentProductive (double sr_i_s, double si)

  The formula for the percent productive is used by AM_3 for an intention (i) as of state (s) is:
- static double computeLikelihood (Intention _intention, int _state)

  The overall equation that is used to determine the likelihood of intentions is:

$$L_i = \left[\prod_{1 \le j \le m} MM_j \right] \times \left[\frac{\sum_{k=1}^{n} (AM_k \times W_{AM_k})}{\sum_{k=1}^{n} W_{AM_k}} \right] \times 100$$

  Where:
- static void computeMetricsInformation (ArrayList< ArrayList< ArrayList< String >>> _states) throws InterruptedException, BadLocationException

  Retrieve and compute the information required for metrics computation.
- static String getIndividualClassString (OWLNamedIndividual individual)
- static String getIndividualClassString (OWLIndividual individual)
- static char getSeparator ()

  Simple getter.
- static void matchDetrimentalStateRelationToIntention (ArrayList _state_relation)

  Check for each intention if _state_relation is a detrimental state relation.
- static void updateMainFrame (ArrayList _state_relation, int num)

  Display the current state relation in MainFrame.
- static void matchStateRelationToIntention (ArrayList _state_relation, int num) throws BadLocationException

  Check for each intention if the state relation _state_relation matches any of the intention state relations.
- static void computeProceduresForAM5 ()

Public Attributes

- String m_s_rootClass = "Intention"

Static Public Attributes

- static ArrayList< Intention > m_intention_list
- static OWLOntology m_ontology
- static String m_kitToBuild
- static String m_planToBuild
- static String m_s_subClass = "Kitting"

Static Package Functions

- static int compute_SR_i_r_s (Map< Integer, Integer > _map, Integer _r, Integer _id_current_state)

  Compute the number of true state relations (SR) in the past r states in intention i as of the current state s.
- static BigInteger choose (final int N, final int K)

  Returns a double representation of the Binomial Coefficient, "N choose K", the number of K-element subsets that can be selected from an N-element set.
- static double roundTwoDecimals (double d)

Private Member Functions

- void readIntentionList ()

  Read the intentions previously stored.
- void searchList (String individual)

Static Private Member Functions

- static void buildIntentionList (NodeSet< OWLClass > setOfSubclasses_)

  Parse the ontology and retrieve all the elements associated to each intention.
- static ArrayList< ArrayList< ArrayList< String > > > buildStates (String plan_-path)
- static int cleanDataPropertyInteger (String s)
- static void readForEachIntentionTheNumberOfPartsForEachType ()

  Read the map Map <OWLIndividual,Map<String, Integer> >
- static void getForEachIntentionTheNumberOfPartsForEachType ()

  Retrieve the number of parts each intention contains for each type.
- static void updateForEachIntentionTheNumberOfPartsForEachType ()

  Update m_part_type_number_for_each_intention_list with missing part types.
- static int compute_SR_all_s ()
- static String getReferenceObjectClass (OWLNamedIndividual individual)
- static String getStateRelation (OWLNamedIndividual individual)
- static String getTargetObjectClass (OWLNamedIndividual individual)
- static boolean hasProperty (OWLOntologyManager man, OWLReasoner reasoner, OWLClass cls, OWLObjectPropertyExpression prop)

- static ArrayList< String > input (String state_relation, String target_object, String target_class, String reference_object, String reference_class)
- static Color getPartColor (String _string)
- static String getPartType (String _string)
- static String getStringHead (String _string)
- static String getStringTail (String _string)
- static void printProperties (OWLOntologyManager man, OWLOntology ont, OWLReasoner reasoner, OWLClass cls)

  Print the properties that an instance has to have.

Private Attributes

- OWLOntologyManager m_manager
- JFrame m_progress_frame
- ProgressBar m_progress_bar
- String m_path

Static Private Attributes

- static OWLDataFactory m_datafactory
- static OWLReasoner m_reasoner
- static String m_hasIntention_OrderingConstruct = "#hasIntention_OrderingConstruct"
- static String m_hasOrderingConstruct_Predicate = "#hasOrderingConstruct_Predicate"
- static String m_hasOrderingConstruct_OrderingConstruct = "#hasOrderingConstruct_OrderingConstruct"
- static String m_hasOrderingConstruct_Position = "#hasOrderingConstruct_Position"
- static String m_hasPredicate_TargetObject = "#hasPredicate_TargetObject"
- static String m_hasIntention_Name = "#hasIntention_Name"
- static String m_hasPredicate_ReferenceObject = "#hasPredicate_ReferenceObject"
- static String m_hasCount_Occurrence = "#hasCount_Occurrence"
- static HashMap< String, Integer > m_observation_list = new HashMap<String, Integer>()
- static Map< OWLIndividual, Map< String, Integer > > m_part_type_number_for_each_intention_list = new HashMap<OWLIndividual,Map<String, Integer> >()
- static final String m_ontology_IRI = "http://www.semanticweb.org/ontologies/2013/0/soap.owl"
- static String m_s_ontopath
- static final char m_SEPARATOR = '#'

8.24.1 Constructor & Destructor Documentation

8.24.1.1 ontology.Ontology.Ontology ( )

Constructor of the Ontology class.

8.24.2 Member Function Documentation

8.24.2.1 static void ontology.Ontology.buildIntentionList (NodeSet< OWLClass > *setOfSubclasses_*) [static, private]

Parse the ontology and retrieve all the elements associated to each intention.

Each element found is stored in an attribute of their corresponding Java class. For instance, when the ordering construct "Exist" is found in the definition of an intention, an instance of the Java class Exist is created and information on Exist for the current intention is stored in the instance.

The steps to read an intention are as follows:

- Parse the set of subclasses *setOfSubclasses_* that consists of different domains (Kitting, Assembly, ...).
- In *setOfSubclasses_*, search for the *subclass* defined by the member variable Ontology.m_s_subClass.
- Get each OWL individual of *subclass* with the OWL API function getInstances(OWLClassExpression ce, boolean direct).
- For each individual:
  - Create a Java instance of the class Intention.
  - Set this instance's individual.
    See also

    Intention.setIndividual(OWLIndividual);
  - Set the percentage for this instance to 0.
    See also

    Intention.setPercentage(double)
  - Set the number of state relations for this instance to 0.
    See also

    Intention.setNumberStateRelation(int)

Some of the OWL API functions used are:

- getFlattened(): A convenience method that gets all of the entities contained in the Nodes in this NodeSet.
- getInstances(OWLClassExpression ce, boolean direct): Gets the individuals which are instances of the specified class expression. The individuals are returned as a NodeSet.
- getObjectPropertyValues(OWLNamedIndividual ind, OWLObjectPropertyExpression pe): Gets the object property values for the specified individual and object property expression. The individuals are returned as a NodeSet.

Parameters

setOfSubclass	A set of subclasses built from the root class.

8.24.2.2 JTree ontology.Ontology.buildIntentionMetricsTree ( )

8.24.2.3 JTree ontology.Ontology.buildIntentionTree ( )

8.24.2.4 static void ontology.Ontology.buildObservationList ( ) [static]

Build a list of the parts observed during kitting.

The list is an ArrayList that contains HashMaps. Each HashMap represents the type of part that exists for kitting.

Returns

8.24.2.5 static ArrayList<ArrayList<ArrayList<String> > > ontology.Ontology.buildStates (String *plan_path*) [static, private]

This function builds all the states for a given intention.

Parameters

kit	The kit to build

Returns

An ArrayList that contains all the states for the given kit

8.24.2.6 static BigInteger ontology.Ontology.choose (final int *N*, final int *K*) [static, package]

Returns a double representation of the Binomial Coefficient, "N choose K", the number of K-element subsets that can be selected from an N-element set.

Parameters

N	N-element set
K	K-element subset

Returns

Double representation of the Binomial Coefficient

8.24.2.7 static void ontology.Ontology.chooseTest ( ) [static]

8.24.2.8 static int ontology.Ontology.cleanDataPropertyInteger (String *s*) [static, private]

8.24.2.9 static String ontology.Ontology.cleanIRI (Object *entity*) [static]

Return the name of the entity without the IRI.

For example, if *entity* = [<http://www.semanticweb.org/ontologies/2013/0/soap.owl#Kitt this function returns *Kitting*. This function operates as follows: - Identify the index of the separator *SEPARATOR* - Keep only what is after the SEPARATOR - Remove characters that are not alphanumeric

Parameters

entity	Entity to be trimmed

Returns

The name of the entity without the IRI

8.24.2.10 static int ontology.Ontology.compute_SR_all_s ( ) [static, private]

8.24.2.11 static int ontology.Ontology.compute_SR_i_r_s (Map< Integer, Integer > *_map*, Integer *_r*, Integer *_id_current_state*) [static, package]

Compute the number of true state relations (SR) in the past r states in intention i as of the current state s.

Parameters

_map	Map<Integer, Integer> for an intention i where: • Key: The id of a state (0 for the first state, 1 for the second state, ...). • Value: The number of state relations found for the intention i in the Key state.
_r	The last r states
_id_current_state	The current state

Returns

The value of SR_i_r_s

8.24.2.12 static int ontology.Ontology.compute_sum_observation ( ) [static]

8.24.2.13 static int ontology.Ontology.compute_sum_part_type (Intention *intention_*) [static]

8.24.2.14 static double ontology.Ontology.computeLikelihood (Intention *_intention,* int *_state*) [static]

The overall equation that is used to determine the likelihood of intentions is:

$$L_i = \left[\prod_{1 \leq j \leq m} MM_j \right] \times \left[\frac{\sum_{k=1}^{n} (AM_k \times W_{AM_k})}{\sum_{k=1}^{n} W_{AM_k}} \right] \times 100$$

Where:

- L_i is the likelihood of an intention i
- MM_j is the multiplication metric j
- AM_k is the additive metric k
- W_{AM_k} is the weight of the additive metric k
- m is the total number of multiplicative metrics
- n is the total number of additive metrics

All metrics (whether multiplicative or additive) must contain a value between 0 and 1, where 0 is the lowest value and 1 is the highest value.

Additive metrics (AM_k) along with their associated weights, are added together and then divided by the sum of all their weights.

Weights are associated with the additive metrics to show the relative importance of one metric over another.These weights can contain any value greater than 0.

Multiplicative metrics are significant enough in importance that their value is multiplied in the likelihood equation to carry a heavier effect on the overall likelihood.

Parameters

_intention	The intention for which the likelihood will be computed
_state	State

8.24.2.15 static double ontology.Ontology.computeMetricAM1 (double *_sr_i_s,* double *_sr_all*) [static]

Compute the additive metric AM_1.

AM_1: *Number of observed state relations that are true in an intention (compared to other intentions).*

The formula for this additive metric for intention i in state s is:

$$AM_{1,i,s} = \frac{SR_{i,s}}{SR_{all,s}} = \frac{SR_{i,s}}{\sum_{i=1}^{p} SR_{i,s}}$$

- $SR_{i,s}$: The number of matched state relations (SR) in an intention (i) as of the current state (s).
- $SR_{all,s}$: The number of matched state relations (SR) in all possible intentions as of the current state (s).

This formula represents the ratio of true states that are in intention i to the sum of all of the true states in all of intentions of interest.

The variable p represents the number of intentions of interest. It is evaluated for every intention of interest at every state.

Parameters

_sr_i_s	The number of matched state relations (SR) in an intention (i) as of the current state (s).
_sr_all	The number of matched state relations (SR) in all possible intentions as of the current state (s).

8.24.2.16 static double ontology.Ontology.computeMetricAM2 (double *_percentComplete_i_s*, double *_percent_complete_total*) [static]

Compute the additive metric AM_2.

AM_2: *Percentage of an intention that is complete.*

The formula for the percentage complete for intention i in state s is:

$$PercentComplete_{i,s} = \frac{SR_{i,s}}{SR_{i,total}}$$

We then normalize this for all intentions of interest to find the additive metric 2 for intention *i* in state *s*.

$$AM_{2,i,s} = \frac{PercentComplete_{i,s}}{\sum_{i=1}^{p} PercentComplete_{i,s}}$$

Parameters

_percentComp i_s	Percent complete for intention (i) in state (s).
_percent_complete_total	Sum of percent complete for each intention in state (s).

8.24.2.17 static double ontology.Ontology.computeMetricAM3 (double *percent_productive_i_s*, double *percent_productive_i*) [static]

Compute the additive metric AM_3.

AM_3: *Number of productive states that have occurred since the first productive state relation in an intention.*

The formula for the percentage complete for intention i in state s is:

$$PercentComplete_{i,s} = \frac{SR_{i,s}}{S_i}$$

We then normalize this for all intentions by determining additive metric 3 for intention i in state s.

$$AM_{3,i,s} = \frac{PercentProductive_{i,s}}{\sum_{i=1}^{p} PercentProductive_i}$$

8.24.2.18 static double ontology.Ontology.computeMetricAM4 (double *sr_i_r_s*, double *sum_sr_i_r_s*) [static]

Compute the additive metric AM_4.

AM_4: *Number of productive states that have occurred (recently) in the past r states.*

The formula for AM_4 is:

$$AM_{4,i,s} = \frac{SR_{i,r,s}}{\sum_{i=1}^{p} SR_{i,r,s}}$$

Parameters

sr_i_r_s	The number of matched state relations (SR) in the past r states in an intention (i) as of the current state (s).
sum_sr_i_r_s	The sum of all *sr_i_r_s* for each intention.

Returns

8.24.2.19 static double ontology.Ontology.computeMetricAM5 (Intention *intention*) [static]

Compute the additive metric AM_5.

- Suppose a kit is described by the number of parts it contains for each type.

  That is, $kit_i = (n_{i_A}, n_{i_B}, \ldots, n_{i_Q})$ has n_{i_A} parts of type "A", n_{i_B} parts of type "B", ..., n_{i_Q} parts of type "Q".

- Suppose an observation is described by the number of parts seen for each type.

That is, $observation_j = (x_{j_A}, x_{j_B}, \ldots, x_{j_Q})$ has seen x_{j_A} parts of type "A", x_{j_B} parts of type "B",..., x_{j_Q} parts of type "Q".

- The likelihood L of observation j under kit i is given by the multivariate hypergeometric distribution:

$$L(observation_j|kit_i) = \frac{\prod_{p=1}^{q} n_{i_p} choose\ x_{j_p}}{\sum_{p=1}^{q} n_{i_p} choose \sum_{p=1}^{q} x_{j_p}}$$

The additive metric AM_5 for kit_i is the probability of kit_i given $observation_j$:

$$Probability(kit_i|observation_j) = \frac{L(observation_j|kit_i)}{\sum_{n=1}^{N} L(observation_j|kit_n)}$$

where n is the total number of kits that are likely to be built.

8.24.2.20 static double ontology.Ontology.computeMetricMM1 (int *sr_i_s*, int *detrimental*) [static]

8.24.2.21 static void ontology.Ontology.computeMetricsInformation (ArrayList< ArrayList< ArrayList< String >>> *_states*) throws InterruptedException, BadLocationException [static]

Retrieve and compute the information required for metrics computation.

The following pieces of information are used to determine individual metrics:

- $SR_{i,s}$: The number of matched state relations (SR) in an intention (i) as of the current state (s).
- $SR_{i,total}$: The number of state relations (SR), whether matched or not, in an intention (i).
- $SR_{all,s}$: The number of matched state relations (SR) in all possible intentions as of the current state (s).
- S_{total}: The number of states (S) that have occurred since observation began.
- S_i: The number of states (S) that have occurred since (and including) the first matched state relation in an intention (i).
- $detrimentalSR_{i,s}$: The number of detrimental state relations ($detrimentalSR$) that have occurred in an intention (i) as of the current state (s).
- $SR_{i,r,s}$: The number of matched state relations (SR) in the past r states in an intention (i) as of the current state (s). In other words, in the most recent n states, how many true state relations for an intention exist?

Parameters

_states	A list of states.

Exceptions

InterruptedException	
BadLocationException	

8.24.2.22 static void ontology.Ontology.computeObservationLikelihood (Intention *intention_*) [static]

Compute the likelihood L of observation j under kit i.

The likelihood L of observation j under kit i is given by the multivariate hypergeometric distribution:

$$L(observation_j|kit_i) = \frac{\prod_{p=1}^{q} n_{i_p} choose\ x_{j_p}}{\sum_{p=1}^{q} n_{i_p} choose \sum_{p=1}^{q} x_{j_p}}$$

Returns

8.24.2.23 static double ontology.Ontology.computePercentComplete (double *sr_i_s*, double *sr_i_total*) [static]

PercentComplete is the percentage complete for an intention in a state.

This formula is used by AM_2 and is computed as follows:

$$PercentComplete_{i,s} = \frac{SR_{i,s}}{SR_{i,total}}$$

Parameters

sr_i_s	The Number of matched state relations (SR) in an intention (i) as of the current state (s).
sr_i_total	The number of states (S) that have occurred since observation began.

Returns

8.24.2.24 static double ontology.Ontology.computePercentProductive (double *sr_i_s*, double *si*) [static]

The formula for the percent productive is used by AM_3 for an intention (i) as of state (s) is:

$$PercentProductive_{i,s} = \frac{SR_{i,s}}{S_i}$$

Parameters

sr_i_s	The Number of matched state relations (SR) in an intention (i) as of the current state (s).
si	The number of states (S) that have occurred since (and including) the first matched state relation in an intention (i).

Returns

8.24.2.25 static void ontology.Ontology.computeProbabilityFromObservation (Intention *intention_*) [static]

8.24.2.26 static void ontology.Ontology.computeProceduresForAM5 ( ) [static]

8.24.2.27 static void ontology.Ontology.getForEachIntentionTheNumberOfPartsForEachType ( ) [static, private]

Retrieve the number of parts each intention contains for each type.

Returns

Map <OWLIndividual,Map<String, Integer> > A HashMap that has the intention as the key and a HashMap as the value. The nested HashMap has the Target object (type of part) of the predicate as the key and the number of parts of the given type as value.

8.24.2.28 Set<OWLClassExpression> ontology.Ontology.getIndividualClass (OWLNamedIndividual *individual*)

8.24.2.29 static String ontology.Ontology.getIndividualClassString (OWLNamedIndividual *individual*) [static]

8.24.2.30 static String ontology.Ontology.getIndividualClassString (OWLIndividual *individual*) [static]

8.24.2.31 OWLOntologyManager ontology.Ontology.getManager ( )

Simple getter.

Returns

manager

8.24.2.32 OWLOntology ontology.Ontology.getOntology ( )

Simple getter.

Returns

ontology

8.24.2.33 String ontology.Ontology.getOntologyPath ( )

Get the path of the ontology.

Returns

8.24.2.34 static Color ontology.Ontology.getPartColor (String *_string*) [static, private]

8.24.2.35 static String ontology.Ontology.getPartType (String *_string*) [static, private]

8.24.2.36 String ontology.Ontology.getPath ( )

Simple getter.

Returns

path

8.24.2.37 static String ontology.Ontology.getReferenceObjectClass (OWLNamedIndividual *individual*) [static, private]

Return The class of the reference object for a given instance of predicate

Parameters

individual	An instance of predicate

Returns

The class of the reference object

8.24.2.38 String ontology.Ontology.getRootClass ( )

Get the class root from the ontology.

Returns

The class root

8.24.2.39 static char ontology.Ontology.getSeparator ( ) [static]

Simple getter.

Returns

SEPARATOR

8.24.2.40 static String ontology.Ontology.getStateRelation (OWLNamedIndividual *individual*) [static, private]

Return The state relation for a given instance of predicate

Parameters

individual	An instance of predicate

Returns

The state relation

8.24.2.41 static String ontology.Ontology.getStringHead (String *_string*) [static, private]

8.24.2.42 static String ontology.Ontology.getStringTail (String *_string*) [static, private]

8.24.2.43 NodeSet<OWLClass> ontology.Ontology.getSubclasses (String *myClassName*)

8.24.2.44 static String ontology.Ontology.getTargetObjectClass (OWLNamedIndividual *individual*) [static, private]

Return The class of the target object for a given instance of predicate

Parameters

individual	An instance of predicate

Returns

The class of the target object

8.24.2.45 static boolean ontology.Ontology.hasProperty (OWLOntologyManager *man,* OWLReasoner *reasoner,* OWLClass *cls,* OWLObjectPropertyExpression *prop*) [static, private]

8.24.2.46 void ontology.Ontology.initializeList ( )

8.24.2.47 static ArrayList<String> ontology.Ontology.input (String *state_relation,* String *target_object,* String *target_class,* String *reference_object,* String *reference_class*) [static, private]

8.24.2.48 void ontology.Ontology.loadFromFile ( )

Load the ontology from a file.

8.24.2.49 void ontology.Ontology.loadOntologyFromPath (String *myPath*) throws MalformedURLException, OWLException

8.24.2.50 static void ontology.Ontology.matchDetrimentalStateRelationToIntention (ArrayList *_state_relation*) [static]

Check for each intention if *_state_relation* is a detrimental state relation.

Parameters

_state_relation	state relation of the form "State Relation Name","Target Object" ,"OWL CLass of the target object" ,"Reference object","OWL CLass of the reference object"

8.24.2.51 static void ontology.Ontology.matchStateRelationToIntention (ArrayList *_state_relation,* int *num*) throws BadLocationException [static]

Check for each intention if the state relation *_state_relation* matches any of the intention state relations.

Parameters

_state_relation	A state relation of the form "State Relation Name","Target Object" ,"OWL CLass of the target object" ,"Reference object","OWL CLass of the reference object"

Exceptions

BadLocationException	

8.24.2.52 void ontology.Ontology.parseIntention (NodeSet< OWLClass > *myClass*) throws InterruptedException, InvocationTargetException, IOException

Parse the class *myClass* and store data in arrays (HashMap and ArrayList).

Parameters

myClass	Class to be parsed

Exceptions

InterruptedException	
InvocationTargetException	
IOException	

8.24.2.53 static void ontology.Ontology.printProperties (OWLOntologyManager *man,* OWLOntology *ont,* OWLReasoner *reasoner,* OWLClass *cls*) [static, private]

Print the properties that an instance has to have.

Parameters

man	The manager
ont	The ontology
reasoner	The reasoner
cls	The class for which we need to check the properties

8.24.2.54 static void ontology.Ontology.readForEachIntentionTheNumberOfPartsForEachType ( ) [static, private]

Read the map Map <OWLIndividual,Map<String, Integer> >

8.24.2.55 void ontology.Ontology.readIntentionList ( ) [private]

Read the intentions previously stored.

Display intention individuals.

8.24.2.56 static void ontology.Ontology.readObservationList ( ) [static]

Read the observation list m_observation_list.

8.24.2.57 static ArrayList<String> ontology.Ontology.removeDuplicates (ArrayList< String > *list_*) [static]

Remove duplicates in an ArrayList of String.

The easiest way to remove duplicates is to add the contents to a Set (which will not allow duplicates) and then add the Set back to the ArrayList.

Parameters

*list_*	Original list that contains duplicates

Returns

list_ with duplicates removed

8.24.2.58 static double ontology.Ontology.roundTwoDecimals (double *d*) [static, package]

8.24.2.59 void ontology.Ontology.searchList (String *individual*) [private]

8.24.2.60 void ontology.Ontology.setDataFactory ( )

8.24.2.61 static void ontology.Ontology.setInstanceFilePath (String *path_*) [static]

Set the path to the OWL instance file The path is retrieved from the field OptionFrame.m_-instance_txt_field.

8.24.2.62 void ontology.Ontology.setManager ( )

8.24.2.63 void ontology.Ontology.setManager (OWLOntologyManager *manager*)

Simple setter.

Parameters

manager	

8.24.2.64 void ontology.Ontology.setOntology (OWLOntology *ontology*)

Simple setter.

Parameters

ontology	

8.24.2.65 void ontology.Ontology.setPath (String *path*)

Simple setter.

Parameters

path	

8.24.2.66 void ontology.Ontology.setReasoner (OWLOntology *myOntology*)

8.24.2.67 void ontology.Ontology.setRootClass (String *rootClass_*)

Set the class root from the ontology.

Parameters

rootClass	

8.24.2.68 void ontology.Ontology.showDialogBox ( )

8.24.2.69 void ontology.Ontology.sortIntentionList ( )

Read m_intention_list and re-arrange the list using a number for each intention.

The bubble sort algorithm is used to sort m_intention_list.

Among the 5 intentions in the ontology, numbers are associated to each intention in the following way:

- a4b3c3 = 1
- a4b4c2 = 2
- a2b3c5 = 3
- a4b2c2d1 = 4
- a2b3c3d1e1 = 5

8.24.2.70 static void ontology.Ontology.updateForEachIntentionTheNumberOfPartsForEachType ( ) [static, private]

Update m_part_type_number_for_each_intention_list with missing part types.

m_part_type_number_for_each_intention_list contains the number of parts of each part type that each intention contains. For instance, the kit kit_{a4b4c2} has 4 for part A, 4 for part B, and 2 for part C. However, Parts D and E also need to be included in m_part_type_number_for_each_intention_list with 0 for part D and 0 for part E. The missing part types are retrieved from m_observation_list.

8.24.2.71 static void ontology.Ontology.updateMainFrame (ArrayList *_state_relation,* int *num*) [static]

Display the current state relation in MainFrame.

Parameters

_state_relation	The state relation to display

8.24.2.72 static void ontology.Ontology.updateObservationList (String *part_type*) [static]

Update the observation list.

This function searches for part_type in m_observation_list and updates its value (occurrence for this part type).

Parameters

part_type	The part type to search in m_observation_list

8.24.3 Member Data Documentation

8.24.3.1 OWLDataFactory ontology.Ontology.m_datafactory [static, private]

8.24.3.2 String ontology.Ontology.m_hasCount_Occurrence = "#hasCount_Occurrence" [static, private]

8.24.3.3 String ontology.Ontology.m_hasIntention_Name = "#hasIntention_Name" [static, private]

8.24.3.4 String ontology.Ontology.m_hasIntention_OrderingConstruct = "#hasIntention_OrderingConstruct" [static, private]

8.24.3.5 String ontology.Ontology.m_hasOrderingConstruct_OrderingConstruct = "#hasOrderingConstruct_OrderingConstruct" [static, private]

8.24.3.6 String ontology.Ontology.m_hasOrderingConstruct_Position = "#hasOrderingConstruct_Position" [static, private]

8.24.3.7 String ontology.Ontology.m_hasOrderingConstruct_Predicate = "#hasOrderingConstruct_Predicate" [static, private]

8.24.3.8 String ontology.Ontology.m_hasPredicate_ReferenceObject = "#hasPredicate_ReferenceObject" [static, private]

8.24.3.9 String ontology.Ontology.m_hasPredicate_TargetObject = "#hasPredicate_TargetObject" [static, private]

8.24.3.10 ArrayList<Intention> ontology.Ontology.m_intention_list [static]

8.24.3.11 String ontology.Ontology.m_kitToBuild [static]

8.24.3.12 OWLOntologyManager ontology.Ontology.m_manager [private]

8.24.3.13 HashMap<String,Integer> ontology.Ontology.m_observation_list = new HashMap<String, Integer>() [static, private]

8.24.3.14 OWLOntology ontology.Ontology.m_ontology [static]

8.24.3.15 final String ontology.Ontology.m_ontology_IRI = "http://www.semanticweb.org/ontologies/2013/0/soap.owl" [static, private]

8.24.3.16 Map<OWLIndividual,Map<String, Integer> > ontology.Ontology.m_part_type_number_for_each_intention_list = new HashMap<OWLIndividual,Map<String, Integer> >() [static, private]

8.24.3.17 String ontology.Ontology.m_path [private]

8.24.3.18 String ontology.Ontology.m_planToBuild [static]

8.24.3.19 ProgressBar ontology.Ontology.m_progress_bar [private]

8.24.3.20 JFrame ontology.Ontology.m_progress_frame [private]

8.24.3.21 OWLReasoner ontology.Ontology.m_reasoner [static, private]

8.24.3.22 String ontology.Ontology.m_s_ontopath [static, private]

8.24.3.23 String ontology.Ontology.m_s_rootClass = "Intention"

8.24.3.24 String ontology.Ontology.m_s_subClass = "Kitting" [static]

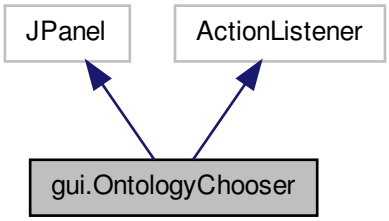
8.24.3.25 final char ontology.Ontology.m_SEPARATOR = '#' [static, private]

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

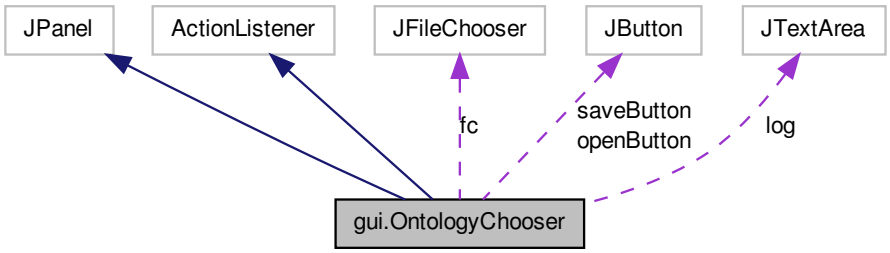
- src/ontology/Ontology.java

8.25 gui.OntologyChooser Class Reference

Inheritance diagram for gui.OntologyChooser:



Collaboration diagram for gui.OntologyChooser:



Public Member Functions

- OntologyChooser ()
- void actionPerformed (ActionEvent e)

Static Public Member Functions

- static void createAndShowGUI ()

Static Protected Member Functions

- static ImageIcon createImageIcon (String path)

Package Attributes

- JButton openButton
- JButton saveButton
- JTextArea log
- JFileChooser fc

Static Private Attributes

- static final String newline = "\n"

8.25.1 Detailed Description

Author

zeid This class consists of methods that allow the user to select the kittingInstances.owl file

8.25.2 Constructor & Destructor Documentation

8.25.2.1 gui.OntologyChooser.OntologyChooser ( )

8.25.3 Member Function Documentation

8.25.3.1 void gui.OntologyChooser.actionPerformed (ActionEvent *e*)

8.25.3.2 static void gui.OntologyChooser.createAndShowGUI ( ) [static]

Create the GUI and show it. For thread safety, this method should be invoked from the event dispatch thread.

8.25.3.3 static ImageIcon gui.OntologyChooser.createImageIcon (String *path*) [static, protected]

Returns an ImageIcon, or null if the path was invalid.

8.25.4 Member Data Documentation

8.25.4.1 JFileChooser gui.OntologyChooser.fc [package]

8.25.4.2 JTextArea gui.OntologyChooser.log [package]

8.25.4.3 final String gui.OntologyChooser.newline = "\n" [static, private]

8.25.4.4 JButton gui.OntologyChooser.openButton [package]

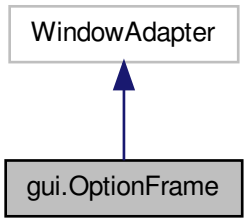
8.25.4.5 JButton gui.OntologyChooser.saveButton [package]

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

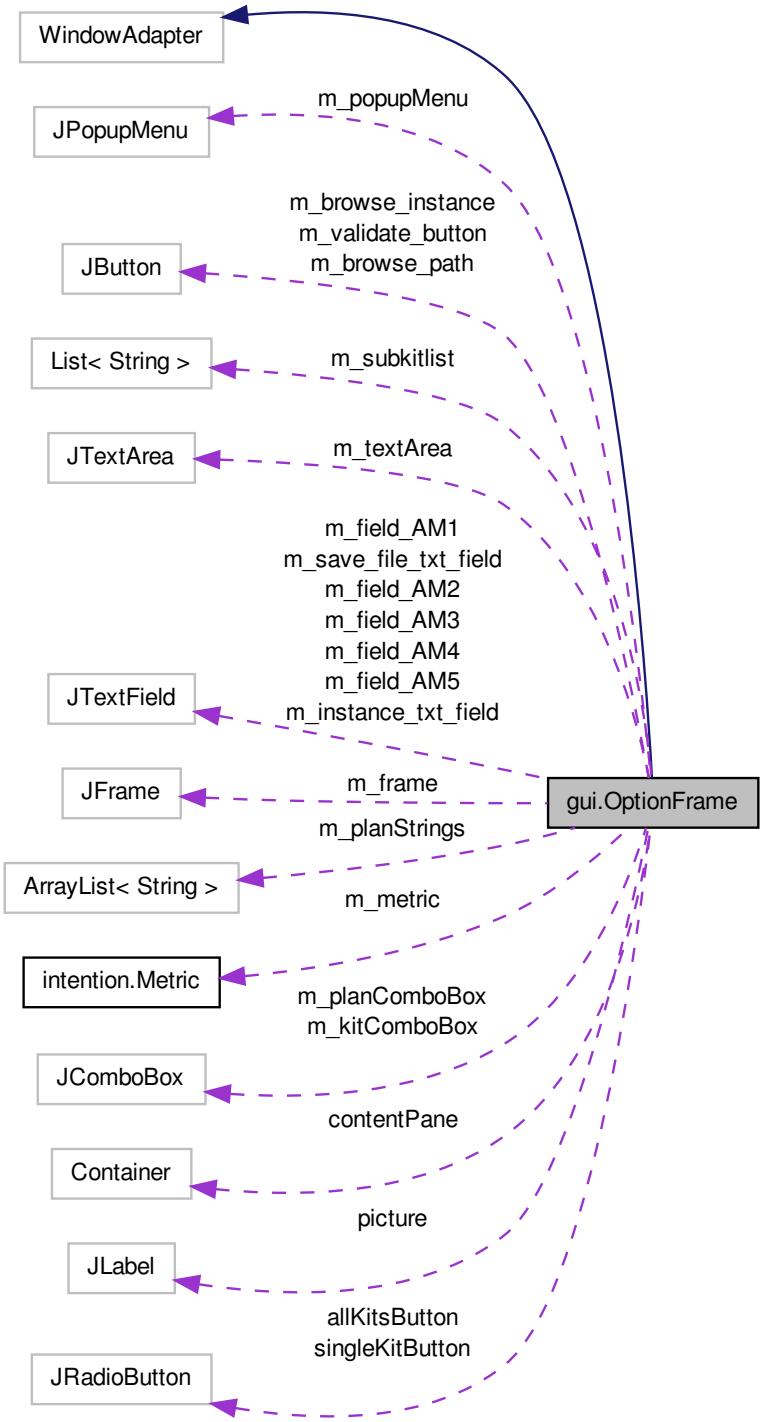
- src/gui/OntologyChooser.java

8.26 gui.OptionFrame Class Reference

Inheritance diagram for gui.OptionFrame:



Collaboration diagram for gui.OptionFrame:



Classes

- class MouseHandler
- class PlanComboBoxListener

Public Member Functions

- OptionFrame ()
- void addComponentsToPane (Container pane, final Ontology onto)
- void updateConfigFile (String text) throws IOException
- List< String > readConfigFile (String configFile)
- void createAndUpdateConfigFile (String instance) throws FileNotFoundException, UnsupportedEncodingException
- void createPopupMenu ()

  Create a popup menu when right-click on a JTextField.
- void createAndShowGUI (Ontology onto)
- void setMetric (Metric metric)
- Metric getMetric ()

Static Public Member Functions

- static ArrayList< String > getKitList ()
- static JButton createButtonFromTemplate (JButton _button)
- static void findFilesinDirectory (String plan_path)

Static Public Attributes

- static JTextField m_instance_txt_field
- static JButton m_browse_instance
- static JFrame m_frame
- static Boolean m_validate
- static JTextArea m_textArea
- static Container contentPane
- static List< String > m_subkitlist
- static JLabel picture
- static JRadioButton singleKitButton

Static Protected Member Functions

- static void updateLabel (String name)

Static Protected Attributes

- static final String NO_DECORATIONS = "no_dec"
- static final String LF_DECORATIONS = "laf_dec"
- static final String WS_DECORATIONS = "ws_dec"
- static final String CREATE_WINDOW = "new_win"
- static final String DEFAULT_ICON = "def_icon"
- static final String FILE_ICON = "file_icon"
- static final String PAINT_ICON = "paint_icon"

Package Attributes

- static JTextField m_save_file_txt_field
- static JTextField m_field_AM1
- static JTextField m_field_AM2
- static JTextField m_field_AM3
- static JTextField m_field_AM4
- static JTextField m_field_AM5
- static JButton m_validate_button
- static JButton m_browse_path
- static JComboBox m_planComboBox
- static JRadioButton allKitsButton

Static Package Attributes

- static final boolean shouldFill = false
- static final boolean shouldWeightX = true
- static final boolean RIGHT_TO_LEFT = false
- static boolean m_bool_allKits = false

Static Private Member Functions

- static void add (Component c, GridBagLayout gbl, GridBagConstraints gbc, int x, int y, int w, int h)

Private Attributes

- JPopupMenu m_popupMenu

Static Private Attributes

- static Metric m_metric
- static JComboBox m_kitComboBox
- static ArrayList< String > m_planStrings = new ArrayList<String>()

8.26.1 Constructor & Destructor Documentation

8.26.1.1 gui.OptionFrame.OptionFrame ( )

8.26.2 Member Function Documentation

8.26.2.1 static void gui.OptionFrame.add (Component *c,* GridBagLayout *gbl,* GridBagConstraints *gbc,* int *x,* int *y,* int *w,* int *h*) [static, private]

8.26.2.2 void gui.OptionFrame.addComponentsToPane (Container *pane,* final Ontology *onto*)

8.26.2.3 void gui.OptionFrame.createAndShowGUI (Ontology *onto*)

Create the GUI and show it. For thread safety, this method should be invoked from the event-dispatching thread.

8.26.2.4 void gui.OptionFrame.createAndUpdateConfigFile (String *instance*) throws FileNotFoundException, UnsupportedEncodingException

8.26.2.5 static JButton gui.OptionFrame.createButtonFromTemplate (JButton *_button*) [static]

8.26.2.6 void gui.OptionFrame.createPopupMenu ( )

Create a popup menu when right-click on a JTextField.

8.26.2.7 static void gui.OptionFrame.findFilesinDirectory (String *plan_path*) [static]

8.26.2.8 static ArrayList<String> gui.OptionFrame.getKitList ( ) [static]

8.26.2.9 Metric gui.OptionFrame.getMetric ( )

8.26.2.10 List<String> gui.OptionFrame.readConfigFile (String *configFile*)

8.26.2.11 void gui.OptionFrame.setMetric (Metric *metric*)

8.26.2.12 void gui.OptionFrame.updateConfigFile (String *text*) throws IOException

8.26.2.13 static void gui.OptionFrame.updateLabel (String *name*) [static, protected]

8.26.3 Member Data Documentation

8.26.3.1 JRadioButton gui.OptionFrame.allKitsButton [package]

8.26.3.2 Container **gui.OptionFrame.contentPane** [static]

8.26.3.3 final String **gui.OptionFrame.CREATE_WINDOW** = "new_win" [static, protected]

8.26.3.4 final String **gui.OptionFrame.DEFAULT_ICON** = "def_icon" [static, protected]

8.26.3.5 final String **gui.OptionFrame.FILE_ICON** = "file_icon" [static, protected]

8.26.3.6 final String **gui.OptionFrame.LF_DECORATIONS** = "laf_dec" [static, protected]

8.26.3.7 boolean **gui.OptionFrame.m_bool_allKits** = false [static, package]

8.26.3.8 JButton **gui.OptionFrame.m_browse_instance** [static]

8.26.3.9 JButton **gui.OptionFrame.m_browse_path** [package]

8.26.3.10 JTextField **gui.OptionFrame.m_field_AM1** [package]

8.26.3.11 JTextField **gui.OptionFrame.m_field_AM2** [package]

8.26.3.12 JTextField **gui.OptionFrame.m_field_AM3** [package]

8.26.3.13 JTextField **gui.OptionFrame.m_field_AM4** [package]

8.26.3.14 JTextField **gui.OptionFrame.m_field_AM5** [package]

8.26.3.15 JFrame **gui.OptionFrame.m_frame** [static]

8.26.3.16 JTextField **gui.OptionFrame.m_instance_txt_field** [static]

8.26.3.17 JComboBox **gui.OptionFrame.m_kitComboBox** [static, private]

8.26.3.18 Metric **gui.OptionFrame.m_metric** [static, private]

8.26.3.19 JComboBox **gui.OptionFrame.m_planComboBox** [package]

8.26.3.20 ArrayList<String> **gui.OptionFrame.m_planStrings** = new ArrayList<String>() [static, private]

8.26.3.21 JPopupMenu **gui.OptionFrame.m_popupMenu** [private]

8.26.3.22 JTextField **gui.OptionFrame.m_save_file_txt_field** [package]

8.26.3.23 List<String> gui.OptionFrame.m_subkitlist [static]

8.26.3.24 JTextArea gui.OptionFrame.m_textArea [static]

8.26.3.25 Boolean gui.OptionFrame.m_validate [static]

8.26.3.26 JButton gui.OptionFrame.m_validate_button [package]

8.26.3.27 final String gui.OptionFrame.NO_DECORATIONS = "no_dec" [static, protected]

8.26.3.28 final String gui.OptionFrame.PAINT_ICON = "paint_icon" [static, protected]

8.26.3.29 JLabel gui.OptionFrame.picture [static]

8.26.3.30 final boolean gui.OptionFrame.RIGHT_TO_LEFT = false [static, package]

8.26.3.31 final boolean gui.OptionFrame.shouldFill = false [static, package]

8.26.3.32 final boolean gui.OptionFrame.shouldWeightX = true [static, package]

8.26.3.33 JRadioButton gui.OptionFrame.singleKitButton [static]

8.26.3.34 final String gui.OptionFrame.WS_DECORATIONS = "ws_dec" [static, protected]

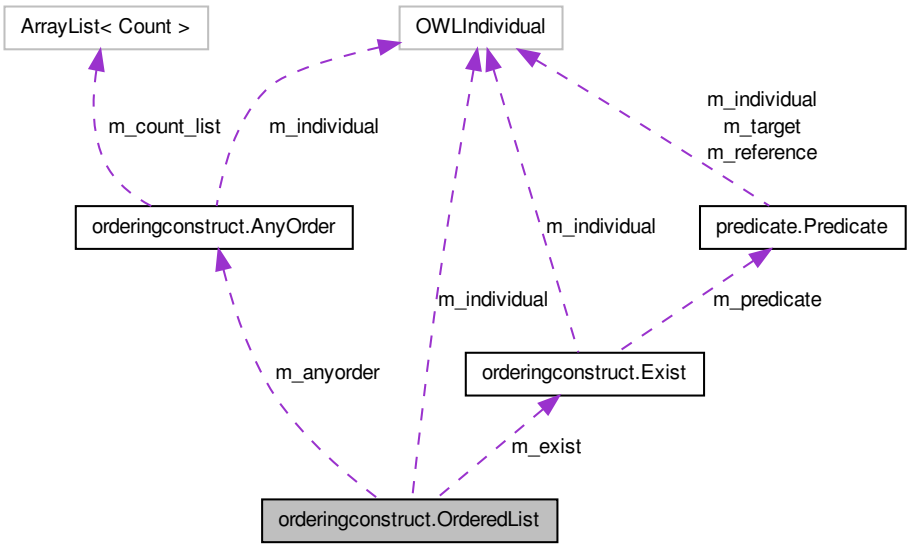
The documentation for this class was generated from the following file:

- src/gui/OptionFrame.java

8.27 orderingconstruct.OrderedList Class Reference

A set of state relationships that must occur in a specific order.

Collaboration diagram for orderingconstruct.OrderedList:



Public Member Functions

- OrderedList ()

  Class constructor.
- Exist getExist ()

  Return the Exist element in the OrderedList element.
- void setExist (Exist exist_)

  Set the Exist element for an OrderedList element.
- AnyOrder getAnyOrder ()

  Return the AnyOrder element in the OrderedList element.
- void setAnyOrder (AnyOrder anyorder_)

  Set the AnyOrder element for an OrderedList element.
- OWLIndividual getIndividual ()

  Return the OWLIndividual for OrderedList.
- void setIndividual (OWLIndividual individual_)

  Set an OrderedList element as an OWLIndividual.

Private Attributes

- Exist m_exist

An element of type Exist that is within an OrderedList element.

- AnyOrder m_anyorder

  An element of type AnyOrder that is within an OrderedList element.

- OWLIndividual m_individual

  OWLIndividual is used to retrieve range individuals from the ontology given the data and object properties.

8.27.1 Detailed Description

A set of state relationships that must occur in a specific order.

Author

Zeid Kootbally *zeid.kootbally@nist.gov*

8.27.2 Constructor & Destructor Documentation

8.27.2.1 orderingconstruct.OrderedList.OrderedList ( )

Class constructor.

8.27.3 Member Function Documentation

8.27.3.1 AnyOrder orderingconstruct.OrderedList.getAnyOrder ( )

Return the AnyOrder element in the OrderedList element.

Returns

OrderedList.m_anyorder

8.27.3.2 Exist orderingconstruct.OrderedList.getExist ( )

Return the Exist element in the OrderedList element.

Returns

OrderedList.m_exist

8.27.3.3 OWLIndividual orderingconstruct.OrderedList.getIndividual ( )

Return the OWLIndividual for OrderedList.

Returns

OrderedList.m_individual

8.27.3.4 void orderingconstruct.OrderedList.setAnyOrder (AnyOrder *anyorder_*)

Set the AnyOrder element for an OrderedList element.

Parameters

*anyorder_*	AnyOrder element to set to OrderedList.m_anyorder

8.27.3.5 void orderingconstruct.OrderedList.setExist (Exist *exist_*)

Set the Exist element for an OrderedList element.

Parameters

*exist_*	Exist element to set to OrderedList.m_exist

8.27.3.6 void orderingconstruct.OrderedList.setIndividual (OWLIndividual *individual_*)

Set an OrderedList element as an OWLIndividual.

Parameters

*individual_*	OWLIndividual to set to OrderedList.m_individual

8.27.4 Member Data Documentation

8.27.4.1 AnyOrder orderingconstruct.OrderedList.m_anyorder [private]

An element of type AnyOrder that is within an OrderedList element.

An OrderedList element can have an AnyOrder element via the object property *hasOrderingConstruct_-OrderingConstruct* where the domain is OrderedList and the range is AnyOrder.

8.27.4.2 Exist orderingconstruct.OrderedList.m_exist [private]

An element of type Exist that is within an OrderedList element.

An OrderedList element can have an Exist element via the object property *hasOrderingConstruct_-OrderingConstruct* where the domain is OrderedList and the range is Exist.

8.27.4.3 OWLIndividual orderingconstruct.OrderedList.m_individual [private]

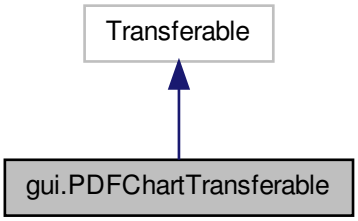
OWLIndividual is used to retrieve range individuals from the ontology given the data and object properties.

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

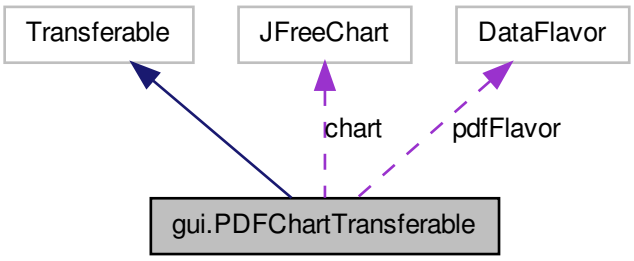
- src/orderingconstruct/OrderedList.java

8.28 gui.PDFChartTransferable Class Reference

Inheritance diagram for gui.PDFChartTransferable:



Collaboration diagram for gui.PDFChartTransferable:



Public Member Functions

- PDFChartTransferable (JFreeChart jfreechart, int i, int j)
- PDFChartTransferable (JFreeChart jfreechart, int i, int j, boolean flag)
- DataFlavor[] getTransferDataFlavors ()
- boolean isDataFlavorSupported (DataFlavor dataflavor)

- Object getTransferData (DataFlavor dataflavor) throws UnsupportedFlavorException, IOException

Static Public Member Functions

- static void writeChartAsPDF (ByteArrayOutputStream bytearrayoutputstream, JFreeChart jfreechart, int i, int j, FontMapper fontmapper) throws IOException

Package Attributes

- final DataFlavor pdfFlavor

Private Attributes

- JFreeChart chart
- int width
- int height

8.28.1 Constructor & Destructor Documentation

8.28.1.1 gui.PDFChartTransferable.PDFChartTransferable (JFreeChart *jfreechart,* int *i,* int *j*)

8.28.1.2 gui.PDFChartTransferable.PDFChartTransferable (JFreeChart *jfreechart,* int *i,* int *j,* boolean *flag*)

8.28.2 Member Function Documentation

8.28.2.1 Object gui.PDFChartTransferable.getTransferData (DataFlavor *dataflavor*) throws UnsupportedFlavorException, IOException

8.28.2.2 DataFlavor [ ] gui.PDFChartTransferable.getTransferDataFlavors ( )

8.28.2.3 boolean gui.PDFChartTransferable.isDataFlavorSupported (DataFlavor *dataflavor*)

8.28.2.4 static void gui.PDFChartTransferable.writeChartAsPDF (ByteArrayOutputStream *bytearrayoutputstream,* JFreeChart *jfreechart,* int *i,* int *j,* FontMapper *fontmapper*) throws IOException `[static]`

8.28.3 Member Data Documentation

8.28.3.1 JFreeChart gui.PDFChartTransferable.chart `[private]`

8.28.3.2 int gui.PDFChartTransferable.height `[private]`

8.28.3.3 final DataFlavor gui.PDFChartTransferable.pdfFlavor [package]

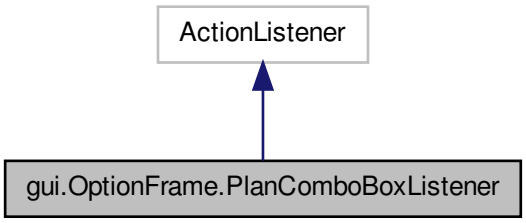
8.28.3.4 int gui.PDFChartTransferable.width [private]

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

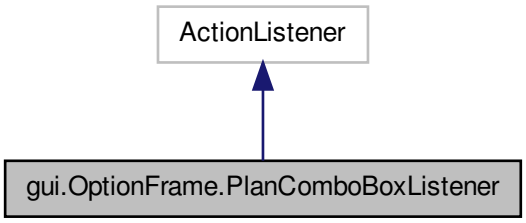
- src/gui/PDFChartTransferable.java

8.29 gui.OptionFrame.PlanComboBoxListener Class Reference

Inheritance diagram for gui.OptionFrame.PlanComboBoxListener:



Collaboration diagram for gui.OptionFrame.PlanComboBoxListener:



Public Member Functions

- void actionPerformed (ActionEvent e)

8.29.1 Member Function Documentation

8.29.1.1 void gui.OptionFrame.PlanComboBoxListener.actionPerformed (ActionEvent *e*)

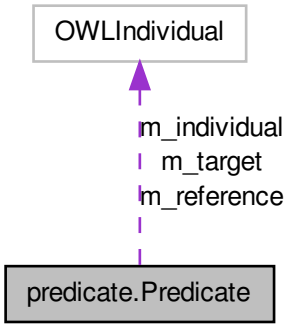
The documentation for this class was generated from the following file:

- src/gui/OptionFrame.java

8.30 predicate.Predicate Class Reference

These are domain-specific states that are of interest to the current intention (or set of intentions) being evaluated.

Collaboration diagram for predicate.Predicate:



Public Member Functions

- Predicate ()

  Class constructor.
- OWLIndividual getIndividual ()

  Return the OWLIndividual Predicate element.
- OWLIndividual getReference ()

  Return the OWLIndividual Reference object.

- String getReferenceObjectClass ()

  Return the OWL class of the Reference object.
- String getStateRelation ()

  Return the state relation associated to a Predicate.
- OWLIndividual getTarget ()

  Return the OWLIndividual Target object.
- String getTargetObjectClass ()

  Return the OWL class of the Target object.
- void setIndividual (OWLIndividual individual_)

  Set a Predicate element as an OWLIndividual.
- void setReference (OWLIndividual individual_)

  Set a Reference object as an OWLIndividual.
- void setReferenceObjectClass (String reference_class_)

  Set the class of a Reference object.
- void setStateRelation (String staterelation_)

  Set the state relation for a Predicate.
- void setTarget (OWLIndividual individual_)

  Set a Target object as an OWLIndividual.
- void setTargetObjectClass (String s)

  Set the class of a Target object.

Private Attributes

- OWLIndividual m_individual

  Predicate from the ontology declared as an OWLIndividual.
- OWLIndividual m_reference

  The reference parameter is the first parameter in the predicate's parameter list.
- String m_reference_class

  The OWL class of the reference parameter.
- String m_staterelation

  The state relation for a predicate.
- OWLIndividual m_target

  The target parameter is the second parameter in the predicate's parameter list.
- String m_target_class

  The OWL class of the target parameter.

8.30.1 Detailed Description

These are domain-specific states that are of interest to the current intention (or set of intentions) being evaluated.

For example, in the manufacturing example to be discussed later in the paper, one state of interest is that the worktable is empty. This is true if the worktable is not under and in contact with any object. The truth-value of predicates can be determined through the

logical combination of state relations. As with state relations, this is captured using the equivalent classes in the ontology.

Author

Zeid Kootbally *zeid.kootbally@nist.gov*

8.30.2 Constructor & Destructor Documentation

8.30.2.1 predicate.Predicate.Predicate ( )

Class constructor.

8.30.3 Member Function Documentation

8.30.3.1 OWLIndividual predicate.Predicate.getIndividual ( )

Return the OWLIndividual Predicate element.

Returns

Predicate.m_individual

8.30.3.2 OWLIndividual predicate.Predicate.getReference ( )

Return the OWLIndividual Reference object.

Returns

Predicate.m_reference

8.30.3.3 String predicate.Predicate.getReferenceObjectClass ( )

Return the OWL class of the Reference object.

Returns

Predicate.m_reference_class

8.30.3.4 String predicate.Predicate.getStateRelation ( )

Return the state relation associated to a Predicate.

Returns

Predicate.m_staterelation

8.30.3.5 OWLIndividual predicate.Predicate.getTarget ( )

Return the OWLIndividual Target object.

Returns

Predicate.m_target

8.30.3.6 String predicate.Predicate.getTargetObjectClass ( )

Return the OWL class of the Target object.

Returns

Predicate.m_target

8.30.3.7 void predicate.Predicate.setIndividual (OWLIndividual *individual_*)

Set a Predicate element as an OWLIndividual.

Parameters

*individual_*	OWLIndividual to set to Predicate.m_individual

8.30.3.8 void predicate.Predicate.setReference (OWLIndividual *individual_*)

Set a Reference object as an OWLIndividual.

Parameters

*individual_*	OWLIndividual to set to Predicate.m_reference

8.30.3.9 void predicate.Predicate.setReferenceObjectClass (String *reference_class_*)

Set the class of a Reference object.

Parameters

*reference_class_*	Class to set to Predicate.m_reference_class

8.30.3.10 void predicate.Predicate.setStateRelation (String *staterelation_*)

Set the state relation for a Predicate.

Parameters

*staterelation_*	State relation to set to Predicate.m_staterelation

8.30.3.11 void predicate.Predicate.setTarget (OWLIndividual *individual_*)

Set a Target object as an OWLIndividual.

Parameters

*individual_*	OWLIndividual to set to Predicate.m_target

8.30.3.12 void predicate.Predicate.setTargetObjectClass (String *s*)

Set the class of a Target object.

Parameters

*target_class_*	Class to set to Predicate.m_target_class

8.30.4 Member Data Documentation

8.30.4.1 OWLIndividual predicate.Predicate.m_individual [private]

Predicate from the ontology declared as an OWLIndividual.

OWLIndividual is used to retrieve range individuals from the ontology given the data and object properties.

8.30.4.2 OWLIndividual predicate.Predicate.m_reference [private]

The reference parameter is the first parameter in the predicate's parameter list.

The reference parameter can be retrieved from the ontology via the object property *hasPredicate_ReferenceObject*

8.30.4.3 String predicate.Predicate.m_reference_class [private]

The OWL class of the reference parameter.

8.30.4.4 String predicate.Predicate.m_staterelation [private]

The state relation for a predicate.

The state relation for a Predicate can be retrieved from the ontology via the object property *hasPredicate_StateRelation*

8.30.4.5 OWLIndividual predicate.Predicate.m_target [private]

The target parameter is the second parameter in the predicate's parameter list.

The target parameter can be retrieved from the ontology via the object property *hasPredicate_TargetObject*

8.30.4.6 String predicate.Predicate.m_target_class [private]

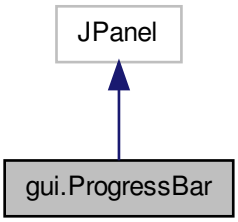
The OWL class of the target parameter.

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

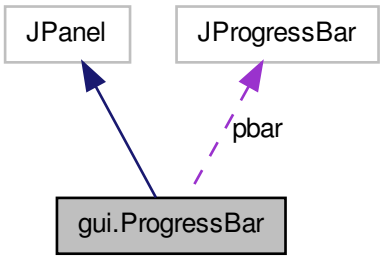
- src/predicate/Predicate.java

8.31 gui.ProgressBar Class Reference

Inheritance diagram for gui.ProgressBar:



Collaboration diagram for gui.ProgressBar:



Public Member Functions

- ProgressBar ()
- void updateBar (int newValue)

Package Attributes

- JProgressBar pbar

Static Package Attributes

- static final int MY_MINIMUM = 0
- static final int MY_MAXIMUM = 100

8.31.1 Constructor & Destructor Documentation

8.31.1.1 gui.ProgressBar.ProgressBar ( )

8.31.2 Member Function Documentation

8.31.2.1 void gui.ProgressBar.updateBar (int *newValue*)

8.31.3 Member Data Documentation

8.31.3.1 final int gui.ProgressBar.MY_MAXIMUM = 100 [static, package]

8.31.3.2 final int gui.ProgressBar.MY_MINIMUM = 0 [static, package]

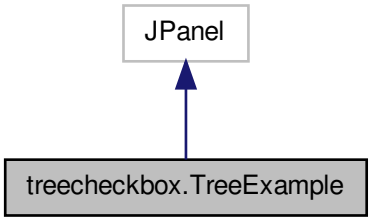
8.31.3.3 JProgressBar gui.ProgressBar.pbar [package]

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

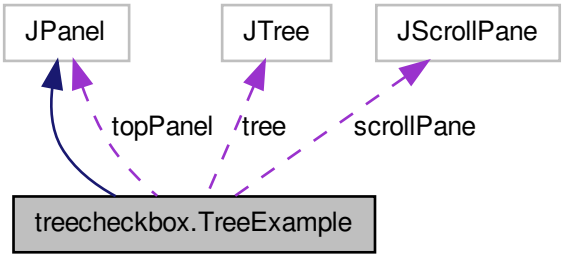
- src/gui/ProgressBar.java

8.32 treecheckbox.TreeExample Class Reference

Inheritance diagram for treecheckbox.TreeExample:



Collaboration diagram for treecheckbox.TreeExample:



Public Member Functions

- TreeExample ()

Private Attributes

- JPanel topPanel
- JTree tree
- JScrollPane scrollPane

Static Private Attributes

- static final long serialVersionUID = 1L

8.32.1 Constructor & Destructor Documentation

8.32.1.1 treecheckbox.TreeExample.TreeExample ( )

8.32.2 Member Data Documentation

8.32.2.1 JScrollPane treecheckbox.TreeExample.scrollPane [private]

8.32.2.2 final long treecheckbox.TreeExample.serialVersionUID = 1L [static, private]

8.32.2.3 JPanel treecheckbox.TreeExample.topPanel [private]

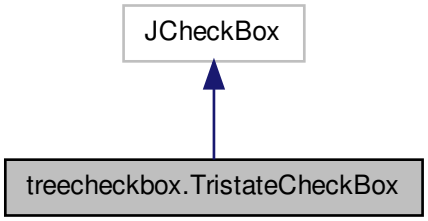
8.32.2.4 JTree treecheckbox.TreeExample.tree [private]

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

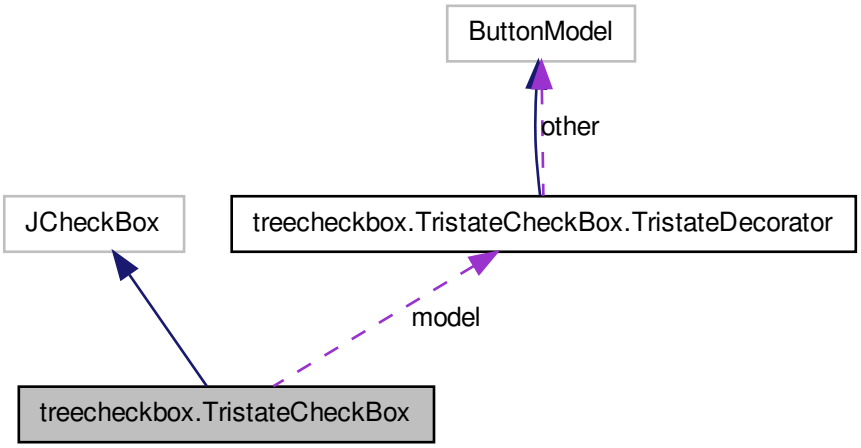
- src/treecheckbox/TreeExample.java

8.33 treecheckbox.TristateCheckBox Class Reference

Inheritance diagram for treecheckbox.TristateCheckBox:



Collaboration diagram for treecheckbox.TristateCheckBox:



Classes

- class TristateDecorator

Public Member Functions

- TristateCheckBox (String text, Icon icon, Boolean initial)
- TristateCheckBox (String text, Boolean initial)
- TristateCheckBox (String text)
- TristateCheckBox ()
- void addMouseListener (MouseListener l)
- void setState (Boolean state)
- Boolean getState ()

Private Attributes

- final TristateDecorator model

8.33.1 Detailed Description

Maintenance tip - There were some tricks to getting this code working:

1. You have to overwite addMouseListener() to do nothing 2. You have to add a mouse event on mousePressed by calling super.addMouseListener() 3. You have to replace the UIActionMap for the keyboard event "pressed" with your own one. 4. You have to remove the UIActionMap for the keyboard event "released". 5. You have to grab focus when the next state is entered, otherwise clicking on the component won't get the focus. 6. You have to make a TristateDecorator as a button model that wraps the original button model and does state management.

8.33.2 Constructor & Destructor Documentation

8.33.2.1 treecheckbox.TristateCheckBox.TristateCheckBox (String *text,* Icon *icon,* Boolean *initial*)

8.33.2.2 treecheckbox.TristateCheckBox.TristateCheckBox (String *text,* Boolean *initial*)

8.33.2.3 treecheckbox.TristateCheckBox.TristateCheckBox (String *text*)

8.33.2.4 treecheckbox.TristateCheckBox.TristateCheckBox ( )

8.33.3 Member Function Documentation

8.33.3.1 void treecheckbox.TristateCheckBox.addMouseListener (MouseListener *l*)

No one may add mouse listeners, not even Swing!

8.33.3.2 Boolean treecheckbox.TristateCheckBox.getState ( )

Return the current state, which is determined by the selection status of the model.

8.33.3.3 void treecheckbox.TristateCheckBox.setState (Boolean *state*)

Set the new state to either SELECTED, NOT_SELECTED or DONT_CARE. If state == null, it is treated as DONT_CARE.

8.33.4 Member Data Documentation

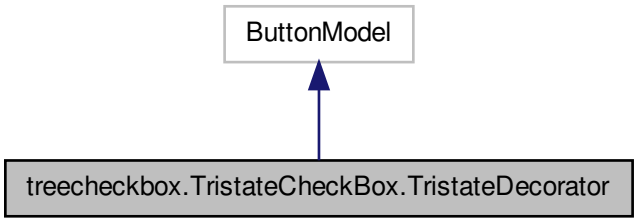
8.33.4.1 final TristateDecorator treecheckbox.TristateCheckBox.model [private]

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

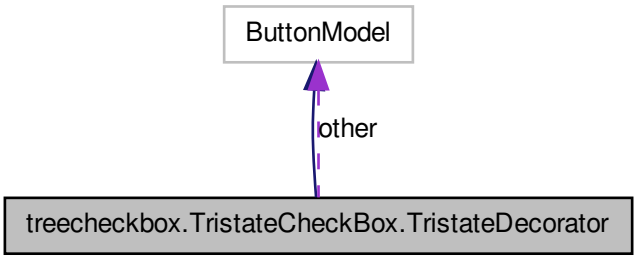
- src/treecheckbox/TristateCheckBox.java

8.34 treecheckbox.TristateCheckBox.TristateDecorator Class Reference

Inheritance diagram for treecheckbox.TristateCheckBox.TristateDecorator:



Collaboration diagram for treecheckbox.TristateCheckBox.TristateDecorator:



Public Member Functions

- void setArmed (boolean b)
- boolean isFocusTraversable ()
- void setEnabled (boolean b)
- boolean isArmed ()
- boolean isSelected ()
- boolean isEnabled ()
- boolean isPressed ()
- boolean isRollover ()
- void setSelected (boolean b)
- void setPressed (boolean b)
- void setRollover (boolean b)
- void setMnemonic (int key)
- int getMnemonic ()
- void setActionCommand (String s)
- String getActionCommand ()
- void setGroup (ButtonGroup group)
- void addActionListener (ActionListener l)
- void removeActionListener (ActionListener l)
- void addItemListener (ItemListener l)
- void removeItemListener (ItemListener l)
- void addChangeListener (ChangeListener l)
- void removeChangeListener (ChangeListener l)
- Object[] getSelectedObjects ()

Private Member Functions

- TristateDecorator (ButtonModel other)
- void setState (Boolean state)
- Boolean getState ()
- void nextState ()

Private Attributes

- final ButtonModel other

8.34.1 Detailed Description

Exactly which Design Pattern is this? Is it an Adapter, a Proxy or a Decorator? In this case, my vote lies with the Decorator, because we are extending functionality and "decorating" the original model with a more powerful model.

8.34.2 Constructor & Destructor Documentation

8.34.2.1 treecheckbox.TristateCheckBox.TristateDecorator.TristateDecorator (ButtonModel *other*) [private]

8.34.3 Member Function Documentation

8.34.3.1 void treecheckbox.TristateCheckBox.TristateDecorator.addActionListener (ActionListener *l*)

8.34.3.2 void treecheckbox.TristateCheckBox.TristateDecorator.addChangeListener (ChangeListener *l*)

8.34.3.3 void treecheckbox.TristateCheckBox.TristateDecorator.addItemListener (ItemListener *l*)

8.34.3.4 String treecheckbox.TristateCheckBox.TristateDecorator.getActionCommand ( )

8.34.3.5 int treecheckbox.TristateCheckBox.TristateDecorator.getMnemonic ( )

8.34.3.6 Object [] treecheckbox.TristateCheckBox.TristateDecorator.getSelectedObjects ( )

8.34.3.7 Boolean treecheckbox.TristateCheckBox.TristateDecorator.getState ( ) [private]

The current state is embedded in the selection / armed state of the model.

We return the SELECTED state when the checkbox is selected but not armed, DONT_CARE state when the checkbox is selected and armed (grey) and NOT_SELECTED when the checkbox is deselected.

8.34.3.8 boolean treecheckbox.TristateCheckBox.TristateDecorator.isArmed ( )

All these methods simply delegate to the "other" model that is being decorated.

8.34.3.9 boolean treecheckbox.TristateCheckBox.TristateDecorator.isEnabled ( )

8.34.3.10 boolean treecheckbox.TristateCheckBox.TristateDecorator.isFocusTraversable ( )

8.34.3.11 boolean treecheckbox.TristateCheckBox.TristateDecorator.isPressed ( )

8.34.3.12 boolean treecheckbox.TristateCheckBox.TristateDecorator.isRollover ( )

8.34.3.13 boolean treecheckbox.TristateCheckBox.TristateDecorator.isSelected ( )

8.34.3.14 void treecheckbox.TristateCheckBox.TristateDecorator.nextState ( ) [private]

We rotate between NOT_SELECTED, SELECTED and DONT_CARE.

8.34.3.15 void treecheckbox.TristateCheckBox.TristateDecorator.removeActionListener (ActionListener *l*)

8.34.3.16 void treecheckbox.TristateCheckBox.TristateDecorator.removeChangeListener (ChangeListener *l*)

8.34.3.17 void treecheckbox.TristateCheckBox.TristateDecorator.removeItemListener (ItemListener *l*)

8.34.3.18 void treecheckbox.TristateCheckBox.TristateDecorator.setActionCommand (String *s*)

8.34.3.19 void treecheckbox.TristateCheckBox.TristateDecorator.setArmed (boolean *b*)

Filter: No one may change the armed status except us.

8.34.3.20 void treecheckbox.TristateCheckBox.TristateDecorator.setEnabled (boolean *b*)

We disable focusing on the component when it is not enabled.

8.34.3.21 void treecheckbox.TristateCheckBox.TristateDecorator.setGroup (ButtonGroup *group*)

8.34.3.22 void treecheckbox.TristateCheckBox.TristateDecorator.setMnemonic (int *key*)

8.34.3.23 void treecheckbox.TristateCheckBox.TristateDecorator.setPressed (boolean *b*)

8.34.3.24 void treecheckbox.TristateCheckBox.TristateDecorator.setRollover (boolean *b*)

8.34.3.25 void treecheckbox.TristateCheckBox.TristateDecorator.setSelected (boolean *b*)

8.34.3.26 void treecheckbox.TristateCheckBox.TristateDecorator.setState (Boolean *state*) [private]

8.34.4 Member Data Documentation

8.34.4.1 final ButtonModel treecheckbox.TristateCheckBox.TristateDecorator.other [private]

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

- src/treecheckbox/TristateCheckBox.java

Chapter 9

File Documentation

9.1 src/gui/Chart.java File Reference

Classes

- class gui.Chart

  Chart display for metrics and likelihoods.
- class gui.Chart.CustomRenderer
- class gui.Chart.CustomRendererLine

Packages

- package gui

9.2 src/gui/CommonGUIComponents.java File Reference

Classes

- class gui.CommonGUIComponents

  Common GUI components used across different files of the project.

Packages

- package gui

9.3 src/gui/DemoPanel.java File Reference

Classes

- class gui.DemoPanel

Packages

- package gui

9.4 src/gui/DrawStringPanel.java File Reference

Classes

- class gui.DrawStringPanel

Packages

- package gui

9.5 src/gui/MainFrame.java File Reference

Classes

- class gui.MainFrame
- class gui.MainFrame.DisplayMetrics

Packages

- package gui

9.6 src/gui/OntologyChooser.java File Reference

Classes

- class gui.OntologyChooser

Packages

- package gui

9.7 src/gui/OptionFrame.java File Reference

Classes

- class gui.OptionFrame
- class gui.OptionFrame.PlanComboBoxListener
- class gui.OptionFrame.MouseHandler

Packages

- package gui

9.8 src/gui/PDFChartTransferable.java File Reference

Classes

- class gui.PDFChartTransferable

Packages

- package gui

9.9 src/gui/ProgressBar.java File Reference

Classes

- class gui.ProgressBar

Packages

- package gui

9.10 src/intention/Intention.java File Reference

Classes

- class intention.Intention

  Representation of intentions from their definition in the ontology.

Packages

- package intention

9.11 src/intention/Metric.java File Reference

Classes

- class intention.Metric

  Definition of additive and multiplicative metrics.

Packages

- package intention

9.12 src/main/Launcher.java File Reference

Contains the main of the program.

Classes

- class main.Launcher

  Main class of the tool.

Packages

- package main

9.12.1 Detailed Description

Contains the main of the program.

Author

Zeid Kootbally *zeid.kootbally@nist.gov*

Version

1.0

Date

September 2013

Precondition

Make sure the kits directory is present in the same directory as this tool
Make sure kittingClasses.owl, kittingInstances_ir.owl, and soap.owl are in the same directory as this tool

9.13 src/ontology/Ontology.java File Reference

Classes

- class ontology.Ontology

Packages

- package ontology

9.14 src/orderingconstruct/AnyOrder.java File Reference

Classes

- class orderingconstruct.AnyOrder

  A set of state relationships that must all occur in any order.

Packages

- package orderingconstruct

  Formal mechanism to allow an ordering of state relationships to represent an intention.

9.15 src/orderingconstruct/Count.java File Reference

Classes

- class orderingconstruct.Count

  A state relationship that must be present multiple times.

Packages

- package orderingconstruct

  Formal mechanism to allow an ordering of state relationships to represent an intention.

9.16 src/orderingconstruct/Exist.java File Reference

Classes

- class orderingconstruct.Exist

  A state relationship that must exist.

Packages

- package orderingconstruct

  Formal mechanism to allow an ordering of state relationships to represent an intention.

9.17 src/orderingconstruct/OrderedList.java File Reference

Classes

- class orderingconstruct.OrderedList

  A set of state relationships that must occur in a specific order.

Packages

- package orderingconstruct

  Formal mechanism to allow an ordering of state relationships to represent an intention.

9.18 src/orderingconstruct/package-info.java File Reference

Packages

- package orderingconstruct

  Formal mechanism to allow an ordering of state relationships to represent an intention.

9.19 src/predicate/package-info.java File Reference

Packages

- package predicate

  Definition of a structure for predicates as represented in the ontology.

9.20 src/tools/package-info.java File Reference

Packages

- package tools

9.21 src/treecheckbox/package-info.java File Reference

Packages

- package treecheckbox

9.22 src/predicate/Predicate.java File Reference

Classes

- class predicate.Predicate

  These are domain-specific states that are of interest to the current intention (or set of intentions) being evaluated.

Packages

- package predicate

  Definition of a structure for predicates as represented in the ontology.

9.23 src/tools/Configuration.java File Reference

Classes

- class tools.Configuration

Packages

- package tools

9.24 src/tools/FileOperator.java File Reference

Classes

- class tools.FileOperator

Packages

- package tools

9.25 src/tools/IntFilter.java File Reference

Classes

- class tools.IntFilter

Packages

- package tools

9.26 src/treecheckbox/CheckTreeCellRenderer.java File Reference

Classes

- class treecheckbox.CheckTreeCellRenderer

Packages

- package treecheckbox

9.27 src/treecheckbox/CheckTreeManager.java File Reference

Classes

- class treecheckbox.CheckTreeManager

Packages

- package treecheckbox

9.28 src/treecheckbox/CheckTreeSelectionModel.java File Reference

Classes

- class treecheckbox.CheckTreeSelectionModel

Packages

- package treecheckbox

9.29 src/treecheckbox/TreeExample.java File Reference

Classes

- class treecheckbox.TreeExample

Packages

- package treecheckbox

9.30 src/treecheckbox/TristateCheckBox.java File Reference

Classes

- class treecheckbox.TristateCheckBox
- class treecheckbox.TristateCheckBox.TristateDecorator

Packages

- package treecheckbox

Index