U.S.NRC

United States Nuclear Regulatory Commission

Protecting People and the Environment

NUREG/CR-6974, Vol. 2
Appendix A, Part A

Symbolic Nuclear Analysis Package (SNAP)

Common Application Framework for Engineering Analysis (CAFEAN) Preprocessor Plug-in Application Programming Interface

Office of Nuclear Regulatory Research

AVAILABILITY OF REFERENCE MATERIALS IN NRC PUBLICATIONS

NRC Reference Material

As of November 1999, you may electronically access NUREG-series publications and other NRC records at NRC's Public Electronic Reading Room at http://www.nrc.gov/reading-rm.html. Publicly released records include, to name a few, NUREG-series publications; *Federal Register* notices; applicant, licensee, and vendor documents and correspondence; NRC correspondence and internal memoranda; bulletins and information notices; inspection and investigative reports; licensee event reports; and Commission papers and their attachments.

NRC publications in the NUREG series, NRC regulations, and *Title 10, Energy*, in the Code of *Federal Regulations* may also be purchased from one of these two sources.

1. The Superintendent of Documents
   U.S. Government Printing Office
   Mail Stop SSOP
   Washington, DC 20402–0001
   Internet: bookstore.gpo.gov
   Telephone: 202-512-1800
   Fax: 202-512-2250
2. The National Technical Information Service
   Springfield, VA 22161–0002
   www.ntis.gov
   1–800–553–6847 or, locally, 703–605–6000

A single copy of each NRC draft report for comment is available free, to the extent of supply, upon written request as follows:

Address: Office of Administration
Reproduction and Mail Services Branch
U.S. Nuclear Regulatory Commission
Washington, DC 20555-0001
E-mail: DISTRIBUTION@nrc.gov
Facsimile: 301–415–2289

Some publications in the NUREG series that are posted at NRC's Web site address http://www.nrc.gov/reading-rm/doc-collections/nureqs are updated periodically and may differ from the last printed version. Although references to material found on a Web site bear the date the material was accessed, the material available on the date cited may subsequently be removed from the site.

Non-NRC Reference Material

Documents available from public and special technical libraries include all open literature items, such as books, journal articles, and transactions, *Federal Register* notices, Federal and State legislation, and congressional reports. Such documents as theses, dissertations, foreign reports and translations, and non-NRC conference proceedings may be purchased from their sponsoring organization.

Copies of industry codes and standards used in a substantive manner in the NRC regulatory process are maintained at—

The NRC Technical Library
Two White Flint North
11545 Rockville Pike
Rockville, MD 20852–2738

These standards are available in the library for reference use by the public. Codes and standards are usually copyrighted and may be purchased from the originating organization or, if they are American National Standards, from—

American National Standards Institute
11 West 42nd Street
New York, NY 10036–8002
www.ansi.org
212–642–4900

Legally binding regulatory requirements are stated only in laws; NRC regulations; licenses, including technical specifications; or orders, not in NUREG-series publications. The views expressed in contractor-prepared publications in this series are not necessarily those of the NRC.

The NUREG series comprises (1) technical and administrative reports and books prepared by the staff (NUREG–XXXX) or agency contractors (NUREG/CR–XXXX), (2) proceedings of conferences (NUREG/CP–XXXX), (3) reports resulting from international agreements (NUREG/IA–XXXX), (4) brochures (NUREG/BR–XXXX), and (5) compilations of legal decisions and orders of the Commission and Atomic and Safety Licensing Boards and of Directors' decisions under Section 2.206 of NRC's regulations (NUREG–0750).

U.S.NRC
United States Nuclear Regulatory Commission
Protecting People and the Environment

NUREG/CR-6974, Vol. 2
Appendix A, Part A

Symbolic Nuclear Analysis Package (SNAP)

Common Application Framework for Engineering Analysis (CAFEAN) Preprocessor Plug-in Application Programming Interface

Manuscript Completed: June 2008
Date Published: June 2009

Prepared by
K. Jones, J. Rothe, W. Dunsford

Applied Programming Technology, Inc.
Bloomsburg, PA 17815

C. Gingrich, NRC Project Manager

NRC Job Code Y6851

Office of Nuclear Regulatory Research

ABSTRACT

Many of the analytical codes developed by the Office of Nuclear Regulatory Research (RES) rely on a text based input file to specify model parameters and computational options. The formats of the text based input files are often quite complex and usually require careful study before a user can create an input model that functions correctly. The Symbolic Nuclear Analysis Package (SNAP) is primarily a graphical user interface that was developed to simplify the analyst's task of creating input files for the analytic codes as well as helping to visualize code results. SNAP is a Java based computer application that runs on the most popular computer platforms including Windows XP and Vista, LINUX based systems, and Mac OS X. The code architecture used in SNAP is "plug-in" based and very flexible. Third party developers can implement their own user interfaces under SNAP without breaking the interfaces developed by other developers. The application programming interface (API) that is described in this document provides a short tutorial and some guidelines for developing a custom plug-in that works in the SNAP framework. This document also includes the actual API method and data-structure definitions needed to create such a custom interface.

FOREWORD

This document is intended for code developers interested in creating applications or "plug-ins" that work with the Symbolic Nuclear Analysis Package (SNAP).

The code architecture used in the Symbolic Nuclear Analysis Package (SNAP) is plug-in based and very flexible. Third party developers can implement their own user interfaces under SNAP by carefully following the application programming interface (API) that is described in this document. The "Main Report" of this document provides a simple tutorial and some guidelines for developing a custom plug-in that works with SNAP. The SNAP application and its plug-ins are based on the Common Application Framework for Engineering Analysis (CAFEAN) Java-based API. Appendix A of this document is automatically generated directly from the CAFEAN Java code by the JavaDoc application.

The CAFEAN API is still in development and, therefore, the information in this document is likely to change relatively frequently, perhaps as often as once per year. Code developers should make sure that they have the most up-to-date version of this document when using the CAFEAN API.

Table of Contents

Table of Contents

PREFACE

This document is divided into several volumes. The first volume contains the "Main Report"; a general overview of the application programming interface (API) used by the Symbolic Nuclear Analysis Package (SNAP) and providing simple guidance on how to use the SNAP API. The remaining volumes of this document contain "Appendix A," the actual API specification and documentation. Except for its front matter, Appendix A is the direct output from the standard JAVA documentation program known as "JavaDoc". Appendix A is split into multiple volumes due to its large size.

The Main Report as well as Appendix A was written primarily by Ken Jones, John Rothe, and William Dunsford, of Applied Programming Technology, Inc.. (APT, Inc). APT, Inc, is the primary developer of the Symbolic Nuclear Analysis Package (SNAP) and associated Common Application Framework for Engineering Analysis (CAFEAN).

Symbolic Nuclear Analysis Package (SNAP)

Common Application Framework for Engineering Analysis (CAFEAN) Preprocessor Plug-in Application Programming Interface

JavaDoc for the CAFEAN API

Package
com.cafean.client.analysis

Provides the foundation classes for the ModelEditor.

Base classes are provided for representing `models`, `components`, `lists` of components and `connections` between components.

com.cafean.client.analysis
Class AbstractBeanComponent

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.AbstractComponent
         |
         +-com.cafean.client.analysis.AbstractBeanComponent
```

All Implemented Interfaces:
IdentHolder, StateEditable, Cloneable, Checkable, ComponentElement, Cloneable

---

public abstract class **AbstractBeanComponent**
extends AbstractComponent

The base class for ModelEditor `Components` that are full fledged beans.

Fields inherited from class `com.cafean.client.analysis.AbstractComponent`
`leftDiffComponent`, `leftDiffName`, `leftShortDiffName`, `rightDiffComponent`, `rightDiffName`, `rightShortDiffName`

Fields inherited from class `com.cafean.client.analysis.GenericObject`
`DATA_COMPLETE`, `DATA_ERROR`, `DATA_INCOMPLETE`, `DATA_WARNING`

Fields inherited from interface `javax.swing.undo.StateEditable`
`RCSID`

Constructor Summary

public	`AbstractBeanComponent()` Create an AbstractBeanComponent object that does not belong to any model and has a display number of 0
public	`AbstractBeanComponent(AbstractModel model, int componentNumber)` Creates an AbstractBeanComponent object, adds it to `model` and sets the given component number on it.

Method Summary

`AbstractComponent`	`copy(AbstractModel sm)` **Deprecated.** *June 2004*
void	`copyFrom(GenericObject obj)`

boolean	dumpBlockParams(java.io.PrintWriter dumpFile) A stub method implemented here to allow AbstractBeanComponent objects to be PibBlocks, stored and loaded directly to a PIB generated file format.
static void	popupBeanDataDialog(AbstractComponent bean, java.awt.Window parent, boolean modal) Creates a PropertySetDialog for the given object in the given model or toFront()'s an existing one.
static void	popupBeanDataDialog(Object bean, AbstractModel model, AbstractComponent targetComponent, java.awt.Window parent, boolean modal) Creates a PropertySetDialog for the given object in the given model or toFront()'s an existing one.
void	popupDataDialog(java.awt.Window parent, boolean modal) Creates a new bean editing dialog for this object or resets and refreshes this object's current editing dialog.
boolean	readBlockParams(com.apt.xdr.PibFile pibFile, int[] blockparm) A stub method implemented here to allow AbstractBeanComponent objects to be PibBlocks, stored and loaded directly to a PIB generated file format.
void	setComponentNumberConstrained(int val_) Sets this bean-based component's component number to the given number.
boolean	writeBlockParams(com.apt.xdr.PibFile pibFile) A stub method implemented here to allow AbstractBeanComponent objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

Methods inherited from class com.cafean.client.analysis.AbstractComponent

addALDocRef, addComponentListener, addConnection, addMessage, addMessage, addMessage, addToModel, addToModel, canConnectTo, clearConnections, clone, complete, connectTo, connectTo, copy, createDrawnComponent, createSourceData, createTargetData, DBtypeCode, disconnect, disconnectFrom, fireComponentChanged, fireComponentChanged, fireComponentConnected, fireComponentDeleted, fireComponentDisconnected, getALDocDisplayName, getALDocRefs, getALDocRefs, getALDocShortNames, getALDocShortNames, getCatCCComparator, getCategory, getCCNumberComparator, getComponent, getComponentDependencies, getConnectionCount, getConnectionName, getConnections, getConnectionTypes, getCustomPopupActions, getCustomPopupItems, getGroupedConnections, getModel, getName, getNewCompIdent, getOrder, getOrderComparator, getOwner, getRealSize, getSharedComponents, getUniqueID, hasALDocRefs, includeInLoopcheck, isOkayForExport, isOkayForExport, label, popupDataDialog, rebuildConnections, reconnectIdentReferences, reconnectImage, removeALDocRef, removeComponentListener, removeFromModel, removeVerify, restoreALRefState, restoreState, setALDocRefs, setComponentNumber, setDeleted, setModel, setOrder, storeALRefState, toShortString, toString, updateVersion, userDelete, writeName

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder

reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ComponentElement

getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement

getModel

Methods inherited from interface com.cafean.client.analysis.Checkable

isOkayForExport, isOkayForExport

Constructors

AbstractBeanComponent

public **AbstractBeanComponent**()

Create an AbstractBeanComponent object that does not belong to any model and has a display number of 0

(continued on next page)

AbstractBeanComponent

```
public AbstractBeanComponent(AbstractModel model,
                             int componentNumber)
```

Creates an AbstractBeanComponent object, adds it to `model` and sets the given component number on it.

Parameters:

`model` - the AbstractModel to add this component to.

`componentNumber` - the component number to use; if 0 a new number will be given to the component.

Methods

copyFrom

```
public void copyFrom(GenericObject obj)
```

Copy the attributes from a source object to this instance.

This is used for copying data from an edited working copy into the original object. Most notably, it is used to enable undo/redo for the legacy beanless ModelEditor architecture as well as for importing new altered data from restart decks for some plugins.

This should call copyFrom on children that support it.

Note: Never call copyFrom from clone or copy!

---

dumpBlockParams

```
public boolean dumpBlockParams(java.io.PrintWriter dumpFile)
```

A stub method implemented here to allow AbstractBeanComponent objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

---

writeBlockParams

```
public boolean writeBlockParams(com.apt.xdr.PibFile pibFile)
```

A stub method implemented here to allow AbstractBeanComponent objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

---

readBlockParams

```
public boolean readBlockParams(com.apt.xdr.PibFile pibFile,
        int[] blockparm)
```

A stub method implemented here to allow AbstractBeanComponent objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

(continued on next page)

popupBeanDataDialog

```
public static void popupBeanDataDialog(AbstractComponent bean,
        java.awt.Window parent,
        boolean modal)
```

Creates a PropertySetDialog for the given object in the given model or toFront()'s an existing one.

Parameters:

`bean` - the AbstractComponent to show properties for
`parent` - the Window to set as the parent of the newly created dialog
`modal` - if true, the resulting dialog will be modal and block this Thread.

---

popupBeanDataDialog

```
public static void popupBeanDataDialog(Object bean,
        AbstractModel model,
        AbstractComponent targetComponent,
        java.awt.Window parent,
        boolean modal)
```

Creates a PropertySetDialog for the given object in the given model or toFront()'s an existing one.

Parameters:

`bean` - the Object to show properties for
`model` - the AbstractModel containing the given bean and it's target component
`targetComponent` - the AbstractComponent owner of the given bean
`parent` - the Window to set as the parent of the newly created dialog
`modal` - if true, the resulting dialog will be modal and block this Thread.

---

popupDataDialog

```
public void popupDataDialog(java.awt.Window parent,
        boolean modal)
```

Creates a new bean editing dialog for this object or resets and refreshes this object's current editing dialog.

---

setComponentNumberConstrained

```
public void setComponentNumberConstrained(int val_)
```

Sets this bean-based component's component number to the given number. Conflicting numbers are reported to the user via `MainFrame.addMessage(String)`.

Parameters:

`val_` - the component number to use for this component

---

copy

```
public AbstractComponent copy(AbstractModel sm)
```

Deprecated. *June 2004*

Produces a deep clone of a component so that it can be put in a copy model clipboard.

NOTE: This method is no longer used by the CAFEAN core.

com.cafean.client.analysis
Class AbstractComponent

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.AbstractComponent
```

All Implemented Interfaces:
Checkable, ComponentElement, Cloneable, IdentHolder, StateEditable, Cloneable

Direct Known Subclasses:
UserDefinedValue, UserDefinedFunction, Connection, ViewComponent, AbstractBeanComponent

---

public abstract class **AbstractComponent**
extends GenericObject
implements Cloneable, StateEditable, IdentHolder, Cloneable, ComponentElement, Checkable

The base class for ModelEditor Components.
See Also:
ComponentList, ComponentListener

Field Summary

public static	leftDiffComponent
public static	leftDiffName
public static	leftShortDiffName
public static	rightDiffComponent
public static	rightDiffName
public static	rightShortDiffName

Fields inherited from class com.cafean.client.analysis.GenericObject

DATA_COMPLETE, DATA_ERROR, DATA_INCOMPLETE, DATA_WARNING

Fields inherited from interface javax.swing.undo.StateEditable

RCSID

Constructor Summary

public	AbstractComponent() Create a AbstractComponent object that does not belong to any model and has a display number of 0
public	AbstractComponent(AbstractModel model, int componentNumber) Creates an AbstractComponent object, adds it to model and sets the given component number on it.

Method Summary

void	addALDocRef(com.cafean.client.mdocs.ALDocRef ref) Adds the given reference to this component's set of ALDocRef objects.
void	addComponentListener(ComponentListener listener) Adds a new component listener to this abstract component.
void	addConnection(Connection con) The highest level of addConnection must be called after the connection has been added to the component.
void	addMessage(String text, ComponentElement obj, int severityCode)
void	addMessage(String text, int severityCode) Adds the given message as type severityCode
void	addMessage(String summary, String body, ComponentElement obj, int severityCode) Adds the given message as type severityCode
void	addToModel(AbstractModel model) This method adds this AbstractComponent to model.
void	addToModel(AbstractModel model, boolean adjustIdent) This method adds this AbstractComponent to model.
boolean	canConnectTo(AbstractComponent target) This method checks to see if it is allowable for this AbstractComponent to connect to the given target.
void	clearConnections()
Object	clone() Creates and returns a copy of this object.
void	complete() This completes this object's initialization in response to it's creation from a UI event.
Connection	connectTo(AbstractComponent target, ConnectionData tData, ConnectionData sData) This method connects this component to the given target component in the way described by the two ConnectionData objects.

Connection	connectTo(AbstractComponent target, ConnectionData tData, ConnectionData sData, int conID) This method connects this component to the given target component in the way described by the two ConnectionData objects.
AbstractComponent	copy(AbstractModel sm) **Deprecated.** *June 2004*
DrawnComponent	createDrawnComponent() Returns the renderer for this abstract component.
ConnectionData	createSourceData(ConnectionData data) Ensures that the given ConnectionData object is suitable for a connection from this component.
ConnectionData	createTargetData(ConnectionData data) Ensures that the given ConnectionData object is suitable for a connection to this component.
int	DBtypeCode() Returns an int describing the component type.
boolean	disconnect(Connection con) **This should only be used by** Connection.disconnect(). Called by a Connection to remove itself from this component.
void	disconnectFrom(AbstractComponent target) **Deprecated.** *February 2nd 2005*
void	fireComponentChanged() Notifies each ComponentListener registered with this component.
void	fireComponentChanged(ComponentChangedEvent event) Notifies each ComponentListener registered with this component of the given changed event.
void	fireComponentConnected(Connection con) This calls the component connected function on all of the listeners currently listening to this abstract component.
void	fireComponentDeleted() This calls the component deleted function on all of the listeners currently listening to this abstract component
void	fireComponentDisconnected(Connection con) This calls the component disconnected function on all of the listeners currently listening to this abstract component.
String	getALDocDisplayName(String shortName) This component's internally defined set of attribute "display names" that should be used to display the attribute to users.
com.cafean.client.mdocs.ALDocRef[]	getALDocRefs() Retrieves this component's set of ALDocRef objects containing its attribute level documentation references to model documentation objects.
com.cafean.client.mdocs.ALDocRef[]	getALDocRefs(String shortName) Retrieves this component's set of ALDocRef objects containing its attribute level documentation references from the given attribute short name.

String[]	getALDocShortNames() This component's internally defined set of attribute "short names" that can be used as keys to connect attribute level documentation.
String[]	getALDocShortNames(String propName) Returns the set of attribute "short names" that correspond to the given property name and can be used as keys to connect that property to attribute level documentation.
static Comparator	getCatCCComparator() Returns a comparator object for comparing the Category and component number of two AbstractComponent objects; for use in sorting or searching.
abstract Category	getCategory() Retrieves the most narrow category that this component is a member of.
static Comparator	getCCNumberComparator() Returns a comparator object for comparing the component number of two AbstractComponent objects; for use in sorting or searching.
AbstractComponent	getComponent()
AbstractComponent[]	getComponentDependencies(List included) Returns an array of the components that this component depends on and should carry with it during copy/paste operations.
int	getConnectionCount() Returns the number of connections contained in this component.
String	getConnectionName(Connection con) This returns the name of this connection as perceived by this component.
Connection[]	getConnections() This is the default "getConnections" function for abstract components.
Connection[]	getConnectionTypes() Returns an empty instance of every Connection derivative that can be connected to this component type.
Action[]	getCustomPopupActions() Creates Action objects for each of the actions that can be performed on this component.
Vector	getCustomPopupItems() Creates Custom Menu Items for any popup dialog involving this component.
Connection[][]	getGroupedConnections() Returns this component's Connections grouped by type.
AbstractModel	getModel() Return the AbstractModel to which this AbstractComponent belongs.
String	getName() This is overwriting creates a default name for a component in case it is never given one on import.
int	getNewCompIdent(int dbid, boolean preserveUnresolved, boolean useDbId) Retrieves the ident of the component with the given DB_ID in this component's model.

int	getOrder() Gets this component's relative *order* value.
static Comparator	getOrderComparator() Returns a comparator object for comparing the component order of two AbstractComponent objects; for use in sorting or searching.
ComponentElement	getOwner()
boolean	getRealSize(Real length, Real avgDiam, Real avgAngle) This method is mearly a placeholder for the HydroComponent derivative.
SharedComponent[]	getSharedComponents() Returns an array of the SharedComponents that this component depends on and should carry with it during copy/paste operations.
String	getUniqueID() Provides a unique component id for components who may not contain a component number.
boolean	hasALDocRefs(String propName) Returns true if this component has attribute level references to the given property name.
boolean	includeInLoopcheck() Determines if this component should be included in Loopcheck calculations.
boolean	isOkayForExport()
boolean	isOkayForExport(boolean prompt)
String	label() Returns a String suitable for describing the component type on a dialog.
void	popupDataDialog(java.awt.Window parent, boolean modal) Creates a new bean editing dialog for this object or resets and refreshes this object's current editing dialog.
void	rebuildConnections(Vector sourceComps, Vector destComps) Rebuilds this component's related Connection objects using the given component Vectors as a guide.
void	reconnectIdentReferences(boolean preserveUnresolved, boolean useDbId)
void	reconnectImage(Vector sourceComps, Vector destComps) Reconnects the internal linkage of this component not handled by reconnectIdentReferences and rebuildConnections during a copy and/or paste operation.
void	removeALDocRef(com.cafean.client.mdocs.ALDocRef ref) Removes the given reference from this component's set of ALDocRef objects.
void	removeComponentListener(ComponentListener listener) Removes a component listener from this abstract component.
void	removeFromModel(AbstractModel model) This method removes this AbstractComponent from model.

boolean	`removeVerify()` Verifies that this component can be removed from it's model without unforeseen side effects and returns the truth of this assumption.
void	`restoreALRefState(Hashtable state)` Restore the state of the ALDocRef array data block from an earlier edit.
void	`restoreState(Hashtable state)`
void	`setALDocRefs(com.cafean.client.mdocs.ALDocRef[] newAlRefs)` Set this component's set of ALDocRef objects containing its attribute level documentation references to model documentation objects.
void	`setComponentNumber(int num)` Setter for Componet number (user defines) also know as display number.
void	`setDeleted(boolean del)` If call with a parmater of true, the AbstractComponent object flagged as deleted.
void	`setModel(AbstractModel model)` Set the AbstractModel.
void	`setOrder(int order)` Sets this component's relative *order* value.
void	`storeALRefState(Hashtable state)` Store the state of the ALDocRef array data block to permit undo.
String	`toShortString()` Returns an abbreviated string representation of this component.
String	`toString()`
void	`updateVersion()` Updates this object's current version to be greater than or equal to it's model's version.
UndoableEdit	`userDelete(List context)` Performs all **additional** operations required in order to properly delete this component from its model.
String	`writeName()` Returns the string to be used when writing a component out to ASCII file.

Methods inherited from class `com.cafean.client.analysis.GenericObject`

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder

reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ComponentElement

getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement

getModel

Methods inherited from interface com.cafean.client.analysis.Checkable

isOkayForExport, isOkayForExport

Fields

leftDiffComponent

public static com.cafean.client.io.Writeable **leftDiffComponent**

rightDiffComponent

```
public static com.cafean.client.io.Writeable rightDiffComponent
```

---

leftDiffName

```
public static java.lang.String leftDiffName
```

---

leftShortDiffName

```
public static java.lang.String leftShortDiffName
```

---

rightDiffName

```
public static java.lang.String rightDiffName
```

---

rightShortDiffName

```
public static java.lang.String rightShortDiffName
```

Constructors

AbstractComponent

```
public AbstractComponent()
```

Create a AbstractComponent object that does not belong to any model and has a display number of 0

AbstractComponent

```
public AbstractComponent(AbstractModel model,
                         int componentNumber)
```

Creates an AbstractComponent object, adds it to `model` and sets the given component number on it.

Parameters:

`model` - the AbstractModel to add this component to.

`componentNumber` - the component number to use; if 0 a new number will be given to the component.

Methods

getCategory

```
public abstract Category getCategory()
```

Retrieves the most narrow category that this component is a member of.

---

getConnectionTypes

```
public Connection[] getConnectionTypes()
```

Returns an empty instance of every Connection derivative that can be connected to this component type.

Returns:

a Connection[] containing empty instances of each Connection type.

---

addMessage

```
public void addMessage(String text,
        int severityCode)
```

Adds the given message as type `severityCode`

Parameters:

`text` - a String containing the message to be displayed.

---

addMessage

```
public void addMessage(String text,
        ComponentElement obj,
        int severityCode)
```

---

addMessage

```
public void addMessage(String summary,
        String body,
        ComponentElement obj,
        int severityCode)
```

Adds the given message as type `severityCode`

Parameters:

`text` - a String containing the message to be displayed.

---

getSharedComponents

```
public SharedComponent[] getSharedComponents()
```

Returns an array of the SharedComponents that this component depends on and should carry with it during copy/paste operations.

Returns:

a SharedComponent[] containing references to all shared components that this component depends on. Duplicate references will not be copied multiple times.

getComponentDependencies

```
public AbstractComponent[] getComponentDependencies(List included)
```

Returns an array of the components that this component depends on and should carry with it during copy/paste operations.

Parameters:

`included` - a List containing the components included in the current copy operation.

Returns:

an AbstractComponent[] containing references to all the components that this component depends on. Duplicate references will not be copied multiple times.

---

clone

```
public Object clone()
```

Creates and returns a copy of this object.

See Also:

`Object.clone()`

---

copy

```
public AbstractComponent copy(AbstractModel sm)
```

Deprecated. *June 2004*

Produces a deep clone of a component so that it can be put in a copy model clipboard.

NOTE: This method is no longer used by the CAFEAN core.

---

getNewCompIdent

```
public int getNewCompIdent(int dbid,
        boolean preserveUnresolved,
        boolean useDbId)
```

Retrieves the ident of the component with the given DB_ID in this component's model. If dbid is 0, a 0 is returned.

Parameters:

`preserveUnresolved` - if true, and no component is found `dbid` will be returned. if false, 0 will be returned.

`useDbId` - if true, ident references will be reconnected via find_x_ByDB_ID; if false, ident references will be reconnected via find_x_ByIdent

---

getOwner

```
public ComponentElement getOwner()
```

---

getComponent

```
public AbstractComponent getComponent()
```

getModel

`public AbstractModel **getModel**()`

Return the AbstractModel to which this AbstractComponent belongs. If not a member of a model, it will return null.

setModel

`public void **setModel**(AbstractModel model)`

Set the AbstractModel. Note this call does not add the AbstractComponent object to the AbstractModel CBvector.

addToModel

`public void **addToModel**(AbstractModel model)`

This method adds this AbstractComponent to `model`. If the AbstractComponent already belonged to a model, it will be removed from the old model first. A new ident will be assisgned to this component when it is added. This call also calls setModel() for this AbstractComponent.

Parameters:

`model` - the AbstractModel to add this component to

See Also:

`AbstractModel.objectAdded(GenericObject, boolean)`
`AbstractModel.addComponent(AbstractComponent, boolean)`

addToModel

```
public void addToModel(AbstractModel model,
        boolean adjustIdent)
```

This method adds this AbstractComponent to `model`. If the AbstractComponent already belonged to a model, it will be removed from the old model first. This call also calls setModel() for this AbstractComponent.

Parameters:

`model` - the AbstractModel to add this component to
`adjustIdent` - if true, a new ident will be assigned to this component.

See Also:

`AbstractModel.objectAdded(GenericObject, boolean)`
`AbstractModel.addComponent(AbstractComponent, boolean)`

userDelete

`public UndoableEdit **userDelete**(List context)`

Performs all **additional** operations required in order to properly delete this component from its model. This will not be called for `Connection` type components as they can be handled in `userDisconnect`

This empty base implementation is sufficient for nearly all components and simply returns null.

In some plug-in specific cases other components may require notification that this component is about to be deleted by the user. In these cases an UndoableEdit for the changes to these components should be returned by this method.

NOTE: This method does not handle the creation of the UndoableDelete for this component and serves only as a point of entry for additional undoable operations to be performed.

Parameters:

`context` - a List initially containing the components being deleted. Additional objects can be added to this list for use by subsequent components inside their userDelete methods.

Returns:

an UndoableEdit capable of undoing the additional required operations. This returned edit will be added to the deletion's CompoundEdit just before the UndoableDelete that includes this component.

See Also:

`removeVerify()`
`UndoableDelete`

---

removeFromModel

```
public void removeFromModel(AbstractModel model)
```

This method removes this AbstractComponent from `model`.

Parameters:

`model` - the AbstractModel to remove this component from

---

removeVerify

```
public boolean removeVerify()
```

Verifies that this component can be removed from it's model without unforeseen side effects and returns the truth of this assumption.

This method may request user verification from the user.

---

reconnectImage

```
public void reconnectImage(Vector sourceComps,
        Vector destComps)
```

Reconnects the internal linkage of this component not handled by reconnectIdentReferences and rebuildConnections during a copy and/or paste operation.

This method is called on the **source** component and is intended to rebuild appropriate structures for the destination component. (likely found by the destination's DB_ID matching the source's ident)/.

Parameters:

`sourceComps` - a Vector containing the components in the source model being reconnected
`destComps` - a Vector containing the components in the destination model being reconnected

---

rebuildConnections

```
public void rebuildConnections(Vector sourceComps,
        Vector destComps)
```

Rebuilds this component's related Connection objects using the given component Vectors as a guide.

For direct references, the source list is used to determine which connections to preserve and the image list to make the new connections. For ident references `reconnectIdentReferences` is used.

NOTE: This base implementation is sufficient in all known cases.

---

label

```
public String label()
```

Returns a String suitable for describing the component type on a dialog.

This default implementation returns the class name.

---

getName

```
public String getName()
```

This is overwriting creates a default name for a component in case it is never given one on import.

Returns:
a String containing the name of a component, user-defined or generated.

DBtypeCode

```
public int DBtypeCode()
```

Returns an int describing the component type. This is a plugin-specific value not actually used in the CAFEAN base code.

---

setDeleted

```
public void setDeleted(boolean del)
```

If call with a parmater of true, the AbstractComponent object flagged as deleted.

This is a plugin-specific value not actually used in the CAFEAN base code.

---

popupDataDialog

```
public void popupDataDialog(java.awt.Window parent,
        boolean modal)
```

Creates a new bean editing dialog for this object or resets and refreshes this object's current editing dialog.

---

getCustomPopupActions

```
public Action[] getCustomPopupActions()
```

Creates Action objects for each of the actions that can be performed on this component.

null values in the returned array will be interpreted as separators.

---

getCustomPopupItems

`public Vector` **`getCustomPopupItems`**`()`

Creates Custom Menu Items for any popup dialog involving this component. The resulting Vector should contain on JMenu, JMenuItem and JSeparator instances for this component.

Returns:
a Vector containing JMenu's, JMenuItems, and JSeparators.

---

setComponentNumber

`public void` **`setComponentNumber`**`(int num)`

Setter for Componet number (user defines) also know as display number. Also updates the label on the component drawing.

Parameters:
`num` - the number to which this value will be set

---

canConnectTo

`public boolean` **`canConnectTo`**`(AbstractComponent target)`

This method checks to see if it is allowable for this AbstractComponent to connect to the given target.

Parameters:
`target` - the AbstractComponent object to which a connection has been requested.

Returns:
true if allowed, false if not allowed.

---

disconnect

`public boolean` **`disconnect`**`(Connection con)`

This should only be used by `Connection.disconnect()`**.** Called by a `Connection` to remove itself from this component.

Parameters:
`connect` - the connection to be disconnected.

Returns:
true if this component was connected through that connection.

---

disconnectFrom

`public void` **`disconnectFrom`**`(AbstractComponent target)`

Deprecated. *February 2nd 2005*

Disconnects all the links between this component and the target. Use this if the target is being deleted otherwise use `Connection.disconnect()`.

complete

```
public void complete()
```

This completes this object's initialization in response to it's creation from a UI event.

This method should be overriden by subclasses that require user input or default values for a newly created component.

getRealSize

```
public boolean getRealSize(Real length,
        Real avgDiam,
        Real avgAngle)
```

This method is mearly a placeholder for the HydroComponent derivative.

isOkayForExport

```
public boolean isOkayForExport()
```

isOkayForExport

```
public boolean isOkayForExport(boolean prompt)
```

getConnectionCount

```
public int getConnectionCount()
```

Returns the number of connections contained in this component.

getConnections

```
public Connection[] getConnections()
```

This is the default "getConnections" function for abstract components. It is expected that components that contain connections should override this function.

Returns:

An array of Connection objects for this component.

addConnection

```
public void addConnection(Connection con)
```

The highest level of addConnection must be called after the connection has been added to the component. That means, it must be called last in any child versions of add connection.

clearConnections

```
public void clearConnections()
```

updateVersion

```
public void updateVersion()
```

Updates this object's current version to be greater than or equal to it's model's version. If the major versions are the same, this object's minor version is incremented;

toString

```
public String toString()
```

toShortString

```
public String toShortString()
```

Returns an abbreviated string representation of this component. This default implementation returns the component name if it is not "unnamed", the CC number (getCCnumber) if it is not 0-length and finally the normal toString().

getCCNumberComparator

```
public static Comparator getCCNumberComparator()
```

Returns a comparator object for comparing the component number of two AbstractComponent objects; for use in sorting or searching.

getOrderComparator

```
public static Comparator getOrderComparator()
```

Returns a comparator object for comparing the component order of two AbstractComponent objects; for use in sorting or searching.

getCatCCComparator

```
public static Comparator getCatCCComparator()
```

Returns a comparator object for comparing the Category and component number of two AbstractComponent objects; for use in sorting or searching.

connectTo

```
public final Connection connectTo(AbstractComponent target,
        ConnectionData tData,
        ConnectionData sData)
```

This method connects this component to the given target component in the way described by the two ConnectionData objects. This method should not be overriden. The version with the conID parameter should be overriden instead.

Parameters:

`target` - the AbstractComponent target of the connection operation.
`tData` - the ConnectionData on the target where the tool was released.

sData - the ConnectionData on the source where the connection was initiated.

---

connectTo

```
public Connection connectTo(AbstractComponent target,
        ConnectionData tData,
        ConnectionData sData,
        int conID)
```

This method connects this component to the given target component in the way described by the two ConnectionData objects.

Parameters:

target - the AbstractComponent target of the connection operation.
tData - the ConnectionData on the target where the tool was released.
sData - the ConnectionData on the source where the connection was initiated.
conID - the connection number or junction CC number.

---

createSourceData

```
public ConnectionData createSourceData(ConnectionData data)
```

Ensures that the given ConnectionData object is suitable for a connection from this component.

This is normally used to create custom ConnectionData objects from SpecialConnectionData objects when connecting via the Connection tool.

Parameters:

data - the ConnectionData generated from the UI.

Returns:

the ConnectionData to use for connecting from this component.

---

createTargetData

```
public ConnectionData createTargetData(ConnectionData data)
```

Ensures that the given ConnectionData object is suitable for a connection to this component.

This is normally used to create custom ConnectionData objects from SpecialConnectionData objects when connecting via the Connection tool.

Parameters:

data - the ConnectionData generated from the UI.

Returns:

the ConnectionData to use for connecting to this component.

---

createDrawnComponent

```
public DrawnComponent createDrawnComponent()
```

Returns the renderer for this abstract component. This will be extended by any abstract component that needs a renderer.

Returns:

the DrawnComponent extending renderer for this abstract component

addComponentListener

`public void addComponentListener(ComponentListener listener)`

Adds a new component listener to this abstract component. If there are no listeners then this will create a new vector for the listeners to be in.

removeComponentListener

`public void removeComponentListener(ComponentListener listener)`

Removes a component listener from this abstract component. This should be called when that listener is destroyed

Parameters:

`listener` - the ComponentListener that is being removed from this component.

fireComponentChanged

`public void fireComponentChanged()`

Notifies each ComponentListener registered with this component. Creates a new ComponentChangedEvent to pass to `fireComponentChanged(ComponentChangedEvent)`

fireComponentChanged

`public void fireComponentChanged(ComponentChangedEvent event)`

Notifies each ComponentListener registered with this component of the given changed event.

fireComponentDeleted

`public void fireComponentDeleted()`

This calls the component deleted function on all of the listeners currently listening to this abstract component

fireComponentConnected

`public void fireComponentConnected(Connection con)`

This calls the component connected function on all of the listeners currently listening to this abstract component.

Parameters:

`con` - the Connection that has just been established.

fireComponentDisconnected

`public void fireComponentDisconnected(Connection con)`

This calls the component disconnected function on all of the listeners currently listening to this abstract component.

Parameters:

`con` - the Connection that has just been disconected.

getConnectionName

`public String **getConnectionName**(Connection con)`

This returns the name of this connection as perceived by this component.

Parameters:
con - the Connection in question.

Returns:
a label for the connection describing it from the point of view of this component.

getGroupedConnections

`public Connection[][] **getGroupedConnections**()`

Returns this component's Connections grouped by type.

The default implementation returns an empty array.

Returns:
a Connection[][] with each primary dimension being a type.

restoreState

`public void **restoreState**(Hashtable state)`

Restore the state of the bean from an earlier edit by using copyFrom on the previously stored clone.

includeInLoopcheck

`public boolean **includeInLoopcheck**()`

Determines if this component should be included in Loopcheck calculations.

This is a plugin-specific value not actually used in the CAFEAN core.

getOrder

`public int **getOrder**()`

Gets this component's relative *order* value.

The returned order is used as an additional sorting criteria for the `java.util.Comparator` returned by `getOrderComparator()`.

setOrder

`public void **setOrder**(int order)`

Sets this component's relative *order* value.

The given order is used as an additional sorting criteria for the `java.util.Comparator` returned by `getOrderComparator()`.

writeName

`public String writeName()`

Returns the string to be used when writing a component out to ASCII file. If the name is "unnamed", an empty String is returned.

getUniqueID

`public String getUniqueID()`

Provides a unique component id for components who may not contain a component number.

Returns:
the String unique component identifier.

setALDocRefs

`public void setALDocRefs(com.cafean.client.mdocs.ALDocRef[] newAlRefs)`

Set this component's set of ALDocRef objects containing its attribute level documentation references to model documentation objects.

Parameters:
`newAlRefs` - an ALDocRef[] containing this component's references

getALDocRefs

`public com.cafean.client.mdocs.ALDocRef[] getALDocRefs()`

Retrieves this component's set of ALDocRef objects containing its attribute level documentation references to model documentation objects.

Returns:
an ALDocRef[] containing this component's references

getALDocRefs

`public com.cafean.client.mdocs.ALDocRef[] getALDocRefs(String shortName)`

Retrieves this component's set of ALDocRef objects containing its attribute level documentation references from the given attribute short name.

Parameters:
`shortName` - a String containing the internal "short name"

Returns:
an ALDocRef[] containing this component's references

getALDocShortNames

`public String[] getALDocShortNames()`

This component's internally defined set of attribute "short names" that can be used as keys to connect attribute level documentation.

Returns:
a String[] containing this component's attribute short names

---

getALDocShortNames

`public String[] getALDocShortNames(String propName)`

Returns the set of attribute "short names" that correspond to the given property name and can be used as keys to connect that property to attribute level documentation.

Returns:
a String[] containing the given property's attribute short names

---

hasALDocRefs

`public boolean hasALDocRefs(String propName)`

Returns true if this component has attribute level references to the given property name.

Parameters:
`propName` - a String containing the property name to check for al refs; "" (an empty string) for component level references; null for references from any attribute or none.

Returns:
true if any al refs are present for the given property

---

getALDocDisplayName

`public String getALDocDisplayName(String shortName)`

This component's internally defined set of attribute "display names" that should be used to display the attribute to users.

Returns:
a String[] containing this component's attribute display names

---

addALDocRef

`public void addALDocRef(com.cafean.client.mdocs.ALDocRef ref)`

Adds the given reference to this component's set of ALDocRef objects. Note that this method uses the given reference directly without duplication or conversion. References that are to be added to multiple components should be duplicated for each component.

Parameters:
`ref` - the ALDocRef to add to this component

---

removeALDocRef

`public void removeALDocRef(com.cafean.client.mdocs.ALDocRef ref)`

Removes the given reference from this component's set of ALDocRef objects. Note that exactly one occurrance of the given reference will be removed.

Parameters:

`ref` - the ALDocRef to add to this component

---

storeALRefState

```
public void storeALRefState(Hashtable state)
```

Store the state of the ALDocRef array data block to permit undo.

Parameters:

`state` - A hash table containing modified parameters.
`prefix` - Prefix for hash entries.

---

restoreALRefState

```
public void restoreALRefState(Hashtable state)
```

Restore the state of the ALDocRef array data block from an earlier edit.

Parameters:

`state` - A hash table containing modified parameters.
`prefix` - Prefix for hash entries.

---

reconnectIdentReferences

```
public void reconnectIdentReferences(boolean preserveUnresolved,
        boolean useDbId)
```

Resets this component's internal ident references to refer to appropriate components in the current AbstractModel. Intended for use after adding a component to a model and thus it's DB_ID will have been set to it's ident in the previous model. For ident references use find__ByDbId to find the new component, then store it's ident.

Note: If this component's DB_ID is 0 this method does nothing.

com.cafean.client.analysis
Class AbstractModel

```
java.lang.Object
   |
   +--com.cafean.client.analysis.GenericObject
         |
         +--com.cafean.client.analysis.AbstractModel
```

All Implemented Interfaces:
ModelElement, IdentHolder, StateEditable, Cloneable

---

public abstract class **AbstractModel**
extends GenericObject
implements Cloneable, StateEditable, IdentHolder, ModelElement

An abstract representation of a logical model, it's contained components and elements, and facilities to manage them.

This is the central class for a ModelEditor analysis code plugin. All creating, storing, searching and loading goes through the model.

AbstractModel provides the following functionality to derivative models:

- Unique ident number allocation via `objectAdded`
- Categorized component access by component `number`, `ident`, `DB_ID` and `Iterator`
- A list and `Category` for `Connection` derivatives.
- A list of `ViewComponent` instances.
- A `list` of Elements, or *(`GenericObject`s)* accessable by `ident` and `DB_ID`

For more information on the creation of the Category hierarchy, see the `Category` documentation.

Field Summary

public static final	CAT_CONNECTION A Category for Connection objects.
public static final	CAT_CONSTANT The Category inside this model that contains the user defined `constants`.
public static final	CAT_DATA_SOURCE A Category for Data Sources stored in this model
public static final	CAT_FUNCTION The Category inside this model that contains the user defined `functions`.
public static final	CAT_NUMERICS The Category inside this model that contains the user defined numerics.
public static final	CAT_RANGE A Category for Data Sources stored in this model
public static final	CAT_SOURCE_ROOT A convenience Category for selecting all types of Data Sources

public static final	CAT_VALUES The Category inside this model that contains all user defined values.
public static final	CAT_VARIABLE The Category inside this model that contains the user user defined variables .
public static final	CAT_VIEW A Catagory for ViewComponent objects.
public static final	THEORYMAN Value: **Theory Manual**
public static final	USERSMAN Value: **User's Manual**

Fields inherited from class com.cafean.client.analysis.GenericObject
DATA_COMPLETE, DATA_ERROR, DATA_INCOMPLETE, DATA_WARNING

Fields inherited from interface javax.swing.undo.StateEditable
RCSID

Constructor Summary

public	AbstractModel() Creates a new AbstractModel with a new model ident, a default name and a default owner.

Method Summary

void	addComment(String comment) Appends the given string as to the list of comments in this object Proxied to program options.
void	addComponent(AbstractComponent component) Adds the given component to this model and gives it a new ident.
void	addComponent(AbstractComponent component, boolean adjustIdent) Adds the given component to this model.
void	addComponentLookup(AbstractComponent comp, int dbid) Adds the given component to the component lookup table for the given DB_ID.
void	addElement(GenericObject element) Adds an element to the model, adjusting it's DB_ID and ident in the process.
void	addElement(GenericObject element, boolean adjustIdent) Adds a GenericObject to the model.
void	addMultipleComments(Vector com) Add the given Vector of Comments to this object. Proxied to program options.

boolean	addPopupMenuEntries(JPopupMenu menu) Appends this model's custom popup menu entries to the given popup menu.
void	addToView(Category category, ViewComponent view, boolean select) Adds this navigator node to the View Component
ModelErrorReport	buildErrorReport()
boolean	canCreateComponent(Category category) Returns true if a component can be created for the given leaf node category or false if the given category is not a leaf.
boolean	canRemoveComponent(AbstractComponent component) Returns true if the given component can be removed.
boolean	ccNumberCheck(Category category, String emptyMsg, String listName, String context, boolean warnEmpty) Checks that all CC numbers in the given component list are valid and unique.
boolean	checkModel() Performs checks for accuracy on the current AbstractModel
void	cleanUpDeleted() Removes all deleted components from the lists maintained by this model.
void	clearComponentLookup() Clears the pasted component lookup table.
void	clearDbIds() Clears the DB_ID's of this model's directly contained objects by setting DB_ID to 0.
int	compareVersion(GenericObject obj) Compares the version of the given object with that of this model to dermine if the object's version is less than, equal to or greater than the version of this model.
boolean	containsRestartChanges(Iterator iterator) Returns true if any of the given objects is either deleted or has a major version number greater than or equal to that of this model.
void	createCategoryView(Category category, boolean layout, boolean waitVisible) Opens a new view for the current model, initializes the name, and adds all of the components with the selected category.
AbstractComponent	createComponent(Category category) Creates an appropriate AbstractComponent for the given leaf node category.
void	createModelPackage(java.io.File pkg, java.io.File ascii, java.io.File[] additionalFiles) This routine will bundle up a given model into a compressed zip file.
String[][]	createNumericsMapping() This routine provides the ability to map user numerics which are too long to export to a ascii file, to a temporary token placed in the ascii export.

void	deleteAllComments() Removes all Comments from this object. Proxied to program options.
void	deleteComment(int num) Removes the given Comment from this object's comment list. Proxied to program options.
boolean	executeModelValidations(boolean printErrors) This executes all of the tests that are found in this model.
void	executeUserDefinedFunctions() Executes all user defined functions in this model.
boolean	exportModelMetrics(java.io.File dFile) Export a file containing the state of all testable metrics for the given model.
boolean	exportModelMetricsSpec(java.io.File dFile) Export a file containing the metrics specification for the model.
AbstractComponent	findComponentByCC(int cc, Category category) Retrieves the component in this model that has the given CC or component number and is in a subset of the given category.
AbstractComponent	findComponentByCC(int cc, String categoryName) Retrieves the component in this model that has the given CC or component number and is in a subset of the given category.
AbstractComponent	findComponentByDB_ID(int dbid) Retrieves the component in this model that has the given dbid by searching through each category.
AbstractComponent	findComponentByDB_ID(int dbid, Category category) Retrieves the component in this model that has the given dbid and is in a subset of the given category.
AbstractComponent	findComponentByIdent(int ident) Retrieves the component in this model that has the given ident by searching through each category.
AbstractComponent	findComponentByIdent(int ident, Category category) Retrieves the component in this model that has the given ident and is in a subset of the given category.
AbstractComponent	findComponentByUniqueID(String id, String categoryName) Retrieves the component in this model that has the given Unique Identifier and is in a subset of the given category.
ComponentNumberGroup	findComponentGroup(Category category) Retrieves the appropriate ComponentNumberGroup instance for the given Category.
GenericObject	findElementByDB_ID(int dbid) Retrieves the GenericObject in this model with the given DB_ID.
GenericObject	findElementByIdent(int ident) Retrieves the element in this model with the given ident.

SharedComponent	findEquivalentSharedComponent(SharedComponent shared) Retrieves an equivalent SharedComponent for the given component.
com.cafean.client.mdocs.AbstractDocument[]	findModelDocuments(AbstractComponent comp) Retrieves the model documents (notes, etc.) that correspond to the given component.
Real	findReal(String siUnitsName) Finds the Real derivative who's SI units are equal to the given String or an empty Dimless if none is found.
Category[]	getCategories() Retrieves the root Categories for this model type.
AbstractComponent	getCategoryComponent(Category category) Retrieves the optional component associated with the given cateogry.
Object	getCategoryObject(Category category) Retrieves the optional object associated with the given category.
int	getCCNumberIncrement() Retrieves the increment number for retrieved CC numbers.
Classification	getClassification() Returns the minimum classification level required for this model.
String	getComment(int num) Retrieves the comment at the given index. Proxied to program options.
Comparator	getComponentComparator(Category category) Returns a Comparator object suitable for sorting components in the given category.
int	getComponentCount() Retrieves a count of the components in this model by calling getComponentCount on each Category returned by getCategories().
int	getComponentCount(Category category) Retrieves a count of the components in this model that are in a subset of the given category.
ComponentNumberGroup[]	getComponentGroups() Retrieves an array of the ComponentNumberGroup instances used by this model to ensure that each component has a valid component number.
Iterator	getComponentIterator(Category category) Creates an iterator suitable for traversing all the components that are in subsets of the given category.
AbstractComponent[]	getComponents() Retrieves all components in this model.
AbstractComponent[]	getComponents(Category category) Retrieves all components in this model that are in a subset of the given category.
String	getCreateDate() Gets this model's creation date as a String.

DeckWriter	getDeckWriter(String filename, boolean restart) Gets the DeckWriter implementation for this model if one is available.
String	getDescription() Retrieves the description of this object. Proxied to program options.
Real	getDimensionless() Returns a new dimensionless value based on the units for this model's plugin.
Object	getDocLinks(String key) Provides a hashtable containing the reference material links.
DocumentLink[]	getDocumentLinks(Object obj, String docType) Return the array of document links for a given object.
GenericObject	getElementAt(int i) Retrieves the element at the given index.
int	getElementCount() Retrieves a count of the elements contained in this model's element list.
Iterator	getElementIterator() Retrieves a new Iterator for use in traversing this model's Element list.
String	getExportNote() Getter for property exportNote.
int	getExportUnits() This gets the units for this model's ASCII export.
String	getHtmlUnits(Class unitsClass) Retrieve the current units string in HTML format for an engineering units class.
int	getInitialCCNumber(Category category) Retrieves the first CC number available to components of the given component class.
int	getLastCCNumber(Category category) Retrieves the largest CC number in use that is available to components of the given component Category.
double	getLenScaleFactor() Gets the scale factor used to adjust cell length for drawn components rendering components of this model.
com.cafean.client.organize.AbstractLoopCheck	getLoopCheck(Vector components) Retrieves a new loop checker appropriate for checking hydraulic loops in this model.
int	getMaxCCNumber(Category category) Retrieves the last CC number available to components of the given component Category.
AbstractModel	getModel()
int	getModelDB_ID() Retrieves this model's ID on the database server.

int	`getModelMajorVersion()` Returns the major version number of this model.
int	`getModelMinorVersion()` Retrieves this model's minor version number
Object	`getModelOptions()` Returns the java bean object that contains this model's "model options" if this model uses such a thing.
JPanel	`getModelPanel()` Returns a `javax.swing.JPanel` to be displayed when the model is selected in the Navigator.
String	`getName()` Returns this model's name.
NavigatorComponent	`getNavigatorComponent()` Returns a `com.cafean.client.ui.navigator.NavigatorComponent` implementation used to display the contents of the model in the Navigator.
int	`getNewComponentIdent(int dbid, Category category, boolean preserveUnresolved, boolean useDbId)` Retrieves the ident of the Component with the given DB_ID in this model.
static int	`getNewElementIdent(AbstractModel model, int dbid, boolean preserveUnresolved, boolean useDbId)` Retrieves the ident of the element with the given DB_ID or ident in the given model.
int	`getNextCCNumber(Category category)` Retrieves a unique and available CC number for the given component Category.
int	`getNextCCNumber(Category category, int requestedNumber)` Retrieves a new and available CC number for the given component Category.
int	`getNumComments()` Retrieves the count of Comment objects stored in this object. Proxied to program options.
String	`getOwner()` Retrieves the username of the creator of this file.
OwnershipManager	`getOwnershipManager()` Returns the ownership manager for this model.
int	`getParentDB_ID()` Retrieves the ID of this model's parent model in the database.
MECodePlugin	`getPlugin()` Returns a reference to the plugin that created this model.
static String[][]	`getPluginDocFileData(String plugin_id)` Provides a listing of Plugin File data by media name/filename
abstract String	`getPluginId()` Retrieves the name of this model's code family.

ProgramOptions	getProgramOptions() **Deprecated.**
int	getProjectDB_ID() Retrieves the ID of this model's project in the database.
Real	getRealByIndex(int index) Returns the Real at the given index.
AbstractComponent[]	getRootComponents() Returns the set of AbstractComponents that should appear in the Navigator as peer nodes to ModelOptions.
Comparator	getRootNodeComparator() Returns a Comparator object suitable for sorting the root nodes of the model node that represents this model in the Navigator.
java.io.File	getSaveFile() Retrieves the file this model was last saved to or opened from.
String	getSaveFileName() Retrieves the name of the file this model was last saved to or opened from.
SnapUndoManager	getUndoManager() Retrieves the undo manager for the undo/redo stack for this model.
int	getUnitIndex(Real unit) Finds the index of the given unit or -1 if the unit is not found inside this model.
int	getUnitIndex(String siUnitsName) Finds the index of the unit who's SI units are equal to the given String or -1.
int	getUnits() Returns the current units for this model.
String[]	getUnitsDisplay() Creates an array of strings for displaying model units for selection.
ValidationOptions	getValidationOptions() This gets the ValidationOptions that is stored inside a given model.
ValidationTest[]	getValidationTests() This gets the array of ValidationTests from the current model.
double	getWidthScaleFactor() Gets the scale factor used to adjust cell width for drawn components rendering components of this model.
boolean	hasModelMetrics() Return true if this model supports output of model metrics.
void	incrementMajorVersion() Increments this model's major version number and sets it's minor version to 0.
void	incrementMinorVersion()

boolean	`isDirty()` if true, this model has been modified since it was saved.
boolean	`isEditingRestart()` Returns true if this model is subsequent edits to this model will be considered restart edits and as such, should be graphically represented.
boolean	`isRestartableModel()` Returns true if this model can be used to export a restart run.
boolean	`isValidCCNumber(int ccNumber, Category category)` Returns true if the given cc number is theoretically valid.
boolean	`isViewLock()`
boolean	`joinPipe(Vector selected)` This function joins two pipes into one.
abstract void	`layoutComponents(Vector drawnComponents, ProgressMon progress)` Lays out the given drawn components using a plugin-specific organization method.
DrawnComponent	`loadDrawnComponent(com.apt.xdr.PibBlock block)` Creates a custom DrawnComponent derivative from the given PibBlock.
abstract void	`loadRestartData(RestartData data)` Loads the given timeslice of RestartData into this model's components.
AbstractComponent	`lookupComponent(int dbid)` Finds the component for the given dbid in the component lookup table.
void	`objectAdded(GenericObject object, boolean assignIdent)` Updates the model's top ident and the given object's ident and DB_ID upon addition to the model.
boolean	`performLoopCheck(Vector components)` Performs a loop check on the given vector of components.
void	`postEdit(UndoableEdit edit)` Adds the given undoable edit to this model's undo/redo stack.
void	`reconnectIdentReferences(boolean preserveUnresolved, boolean useDbId)` Reconnects the ident references of this models directly contained objects.
boolean	`removeComponent(AbstractComponent component)` Removes the given component from the model view.
boolean	`removeElement(GenericObject element)` Removes the specified GenericObject from the model.
boolean	`renodalizePipeCells(AbstractComponent component)` This function renodalizes a pipe cells.
void	`renumberComponents(AbstractComponent[] components, int offset)` Ensures that the given components have appropriately unique CC numbers within their model.

void	reportModelCheck(boolean status) **Deprecated.** *0.27.0 - December 2007*
abstract void	saveModel() This method saves this AbstractModel to a SAM file using it's current save file or requesting a file if none is set.
abstract void	saveModel(boolean showProgress) This method saves this AbstractModel to a SAM file using it's current save file or requesting a file if none is set.
void	setCCNumberIncrement(int increment) Sets the increment number for retrieved CC numbers.
void	setCreateDate(String cDate) Sets this model's creation date to the date in the given String.
void	setDescription(String desc) Sets this object's description to the given desc. Proxied to program options.
void	setDirty(boolean dirty) sets this model dirty; a dirty model has been modified since it was saved.
void	setExportNote(String exportNote) Setter for property exportNote.
void	setIdent(AbstractModel model) Sets the ident of this model to that of the given model for use in copy/paste.
void	setIdent(int uid)
void	setLenScaleFactor(double fact) Gets the scale factor used to adjust cell length for drawn components rendering components of this model.
void	setModelDB_ID(int dbid) Sets this model's DB_ID to the given dbid.
void	setModelMajorVersion(int version) Sets this model's major version number.
void	setModelMinorVersion(int version) Sets this model's minor version number.
void	setName(String name) Override GenericObject SetName to make sure view title is updated when this is called.
void	setOwner(String name) Sets the username of the creator of this file.
void	setParentDB_ID(int pdbid) Sets this model's parent model's database ID.
void	setProgramOptions(ProgramOptions options) Sets the program options object that will be used by this model.

void	setProjectDB_ID(int pdbid) Sets this model's database project ID.
void	setSaveFile(java.io.File file) Sets the name of the file this model was last saved to or opened from.
void	setUnits(int units) Sets the current units for this model.
void	setUnitsConstrained(int units) Sets the units constrained by the available unit types.
void	setViewLock(boolean locked, boolean updateViews)
void	setWidthScaleFactor(double fact) Gets the scale factor used to adjust cell width for drawn components rendering components of this model.
String	showComment(int num) Retrieves the text in the comment at the given index. Proxied to program options.
boolean	splitPipe(Vector selected) This function splits a pipe into two pipes.
com.apt.xdr.PibBlock	storeDrawnComponent(java.awt.Component component) Creates a custom PibBlock derivative from the given Component.
String	toString()
static void	updateDocLinks(String plugin_id) Provides the ability to update the existing reference documentation at runtime by removing the old reference documentation for the current plugin (if it exists) and importing the new documentation.
void	updateNumerics(String[][] numerics) This routine is fired after the numerics mapping has been created from the existing model numerics and can be used to modify the existing numerics appropriately.
static void	updatePluginDocFileData(String plugin_id, String[] filenames) Updates the plugin documentation media filenames.
void	validateAllComponents() Validates each component in every category in this model.

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder

reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ModelElement

getModel

Fields

CAT_CONNECTION

public static final com.cafean.client.analysis.Category **CAT_CONNECTION**

A Category for Connection objects.

CAT_DATA_SOURCE

public static final com.cafean.client.analysis.Category **CAT_DATA_SOURCE**

A Category for Data Sources stored in this model

CAT_SOURCE_ROOT

public static final com.cafean.client.analysis.Category **CAT_SOURCE_ROOT**

A convenience Category for selecting all types of Data Sources

CAT_RANGE

`public static final com.cafean.client.analysis.Category` **`CAT_RANGE`**

A Category for Data Sources stored in this model

CAT_VIEW

`public static final com.cafean.client.analysis.Category` **`CAT_VIEW`**

A Catagory for ViewComponent objects.

CAT_CONSTANT

`public static final com.cafean.client.analysis.Category` **`CAT_CONSTANT`**

The Category inside this model that contains the user defined `constants` .

CAT_VARIABLE

`public static final com.cafean.client.analysis.Category` **`CAT_VARIABLE`**

The Category inside this model that contains the user user defined `variables` .

CAT_FUNCTION

`public static final com.cafean.client.analysis.Category` **`CAT_FUNCTION`**

The Category inside this model that contains the user defined `functions` .

CAT_VALUES

`public static final com.cafean.client.analysis.Category` **`CAT_VALUES`**

The Category inside this model that contains all user defined `values`.

CAT_NUMERICS

`public static final com.cafean.client.analysis.Category` **`CAT_NUMERICS`**

The Category inside this model that contains the user defined numerics.

USERSMAN

`public static final java.lang.String` **`USERSMAN`**

Constant value: **`User's Manual`**

THEORYMAN

`public static final java.lang.String` **`THEORYMAN`**

Constant value: **Theory Manual**

Constructors

AbstractModel

```
public AbstractModel()
```

Creates a new AbstractModel with a new model ident, a default name and a default owner.

Methods

isViewLock

```
public boolean isViewLock()
```

---

setViewLock

```
public void setViewLock(boolean locked,
        boolean updateViews)
```

---

getPlugin

```
public MECodePlugin getPlugin()
```

Returns a reference to the plugin that created this model. This implementation finds the plugin with a plugin id that matches this model's code family name.

Returns:

the MECodePlugin that create this model.

---

getCategories

```
public Category[] getCategories()
```

Retrieves the root Categories for this model type.

The root categories are the top level parent categories that among them contain all the components in this model. This is not determines on an instance by instance basis. All instances of a particular model should have the same categories.

Derivatives should take care to include AbstractModel's categories in their getCategories method.

---

getComponents

```
public AbstractComponent[] getComponents(Category category)
```

Retrieves all components in this model that are in a subset of the given category.

Parameters:

`category` - the Category that the desired components are a member

Returns:
an AbstractComponent[] containing the desired components.

---

getComponents

`public AbstractComponent[] getComponents()`

Retrieves all components in this model. This base implementation retrieves all of the components in each Category returned by `getCategories()` by calling `getComponentIterator(Category)`.

NOTE: This method is intended for use in situations such as saving and copy/paste operations where the complete contents of the model is necessary regardless of the enabled/disabled of any properties. It is not used by methods that are intended to work with *available* components (such as `getComponentCount()`, `findComponentByIdent(int, Category)`.

WARNING: This method could be very expensive if called repeatedly.

Model extensions that enable or disable categories should override this method to return the entire contents of the model regardless of what categories are included in `getCategories()`.

Returns:
an AbstractComponent[] containing the desired components.

---

findComponentByCC

```
public AbstractComponent findComponentByCC(int cc,
        Category category)
```

Retrieves the component in this model that has the given CC or component number and is in a subset of the given category.

Parameters:
`cc` - the component number of the desired component.
`category` - the Category that the desired component is a subset of.

Returns:
the AbstractComponent desired or null if not found.

---

findComponentByCC

```
public AbstractComponent findComponentByCC(int cc,
        String categoryName)
```

Retrieves the component in this model that has the given CC or component number and is in a subset of the given category.

Parameters:
`cc` - the component number of the desired component.
`category` - a String containing the name of the category

Returns:
the AbstractComponent desired or null if not found.

---

findComponentByIdent

`public AbstractComponent findComponentByIdent(int ident)`

Retrieves the component in this model that has the given ident by searching through each category.

Parameters:

`ident` - the unique id number of the desired component.

Returns:

the AbstractComponent desired or null if not found.

---

findComponentByIdent

```
public AbstractComponent findComponentByIdent(int ident,
        Category category)
```

Retrieves the component in this model that has the given ident and is in a subset of the given category.

Parameters:

`ident` - the unique id number of the desired component.
`category` - the Category that the desired component is a subset of.

Returns:

the AbstractComponent desired or null if not found.

---

findComponentByDB_ID

```
public AbstractComponent findComponentByDB_ID(int dbid,
        Category category)
```

Retrieves the component in this model that has the given dbid and is in a subset of the given category.

Parameters:

`dbid` - the DB_ID of the desired component.
`category` - the Category that the desired component is a subset of.

Returns:

the AbstractComponent desired or null if not found.

---

findComponentByDB_ID

```
public AbstractComponent findComponentByDB_ID(int dbid)
```

Retrieves the component in this model that has the given dbid by searching through each category.

Note: This is potentially a very expensive operation. Use `findComponentByDB_ID(int, Category)` wherever possible.

Parameters:

`dbid` - the DB_ID of the desired component.

Returns:

the AbstractComponent desired or null if not found.

---

addComponent

```
public void addComponent(AbstractComponent component)
```

Adds the given component to this model and gives it a new ident. The component's Category will be used to determine which internal list it is added to.

Parameters:

`component` - the AbstractComponent to add.

---

addComponent

```
public void addComponent(AbstractComponent component,
        boolean adjustIdent)
```

Adds the given component to this model. The component's Category will be used to determine which internal list it is added to.

If adjustIdent is true, a new ident will be assigned to the given component; if false, an ident will be assigned if the component's current ident is 0. If the component has a valid ident and adjustIdent is false, the top ident counter will be updated to include the component's current ident.

Extending classes should ensure that `objectAdded(GenericObject, boolean)` is called on each component that **not** passed to this method.

Parameters:

`component` - the AbstractComponent to add.

`adjustIdent` - if true, the given component will be treated as a new component to this model and given a new ident;

Throws:

`IllegalArgumentException` - if the given component is in a Category that is not handled by this model.

---

createComponent

```
public AbstractComponent createComponent(Category category)
```

Creates an appropriate AbstractComponent for the given leaf node category.

Derivatives of `createComponent` should call this only if the given `Category` cannot be handled by the derivative model.

Parameters:

`category` - the Category of the desired new component.

Returns:

an AbstractComponent of the desired category

Throws:

`java.lang.IllegalArgumentException` - if there is no such category or the given category is a parent category.

---

canCreateComponent

```
public boolean canCreateComponent(Category category)
```

Returns true if a component can be created for the given leaf node category or false if the given category is not a leaf. This method does not guarantee that a subsequent call to createComponent will create a component but should be used as an indicator as to whether this functionality is normally available.

Overriding methods could be used for such things as disallowing the explicit creation of connections or other mapping type components.

This base method returns true if there is no OwnershipManager or the return from otherwise.

Parameters:

`category` - the Category of the desired new component.

Returns:
true if the component can be created

---

canRemoveComponent

`public boolean` **`canRemoveComponent`**`(AbstractComponent component)`

Returns true if the given component can be removed. This method does not guarantee that the subsequent removal would succeed but should be used as an indicator as to whether this functionality is normally available.

Overriding methods could be used for such things as disallowing the explicit deletion of connections or other mapping type components.

This base method returns true if there is no OwnershipManager or the return from otherwise.

Parameters:
`category` - the Category of the desired new component.

Returns:
true if the component can be created

---

removeComponent

`public boolean` **`removeComponent`**`(AbstractComponent component)`

Removes the given component from the model view.

Subclasses should override this method. Derivative methods should call this before removing a component.

Parameters:
`component` - the component to be removed from the current model

Returns:
true if the component removal was handled and a component deleted event was fired.

---

getComponentIterator

`public Iterator` **`getComponentIterator`**`(Category category)`

Creates an iterator suitable for traversing all the components that are in subsets of the given category.

Parameters:
`category` - the Category to create an iterator for.

Returns:
an Iterator that will traverse only the subset of components given.

See Also:
`ComponentList.iterator(Category)`

---

getComponentCount

`public int` **`getComponentCount`**`(Category category)`

Retrieves a count of the components in this model that are in a subset of the given category.

Note that this operation is `O(1)` for root categories and `O(n)` for subsets where `n` is the total number of components in this model.

Parameters:

`category` - the Category to retrieve a count for.

---

getCategoryObject

`public Object **getCategoryObject**(Category category)`

Retrieves the optional object associated with the given category.

The returned object is assumed to be a JavaBean with appropriate BeanInfo and editor set. This object's properties will be used whenever the Category's properties are desired (such as selection in the Navigator).

Parameters:

`category` - the Category for the desired properties delegate object

Returns:

the Object to use as the properties of the given Category or null if the given Cagetory has no properties delegate

---

getCategoryComponent

`public AbstractComponent **getCategoryComponent**(Category category)`

Retrieves the optional component associated with the given cateogry. This component, if not null, will be displayed as the first node in the navigator under the Category, but will not contribute to the overall component count.

NOTE: This method may be called **many** times, so caching newly created objects may be prudent. Furthermore, the component must be correctly configured before it is returned (its model must be set, etc.).

Parameters:

`category` - the Category of the component

Returns:

the component associated with the Category

---

validateAllComponents

`public void **validateAllComponents**()`

Validates each component in every category in this model.

See Also:

`GenericObject.validate()`

---

getComponentCount

`public int **getComponentCount**()`

Retrieves a count of the components in this model by calling `getComponentCount` on each Category returned by `getCategories()`.

---

getNewComponentIdent

```
public int getNewComponentIdent(int dbid,
        Category category,
        boolean preserveUnresolved,
        boolean useDbId)
```

Retrieves the ident of the Component with the given DB_ID in this model. If dbid is 0, a 0 is returned.

Parameters:

`preserveUnresolved` - if true, and no Component is found `dbid` will be returned. if false, 0 will be returned.
`useDbId` - if true, ident references will be reconnected via findComponentByDB_ID; if false, ident references will be reconnected via findComponentByIdent

---

getPluginId

```
public abstract String getPluginId()
```

Retrieves the name of this model's code family.

See Also:

`MEPluginData.getPluginId()`

---

saveModel

```
public abstract void saveModel()
```

This method saves this AbstractModel to a SAM file using it's current save file or requesting a file if none is set.

See Also:

`getSaveFile()`

---

saveModel

```
public abstract void saveModel(boolean showProgress)
```

This method saves this AbstractModel to a SAM file using it's current save file or requesting a file if none is set.

See Also:

`getSaveFile()`

---

loadRestartData

```
public abstract void loadRestartData(RestartData data)
```

Loads the given timeslice of RestartData into this model's components.

Parameters:

`data` - the RestartData object containing the timeslice and related data to load.

---

getDeckWriter

```
public DeckWriter getDeckWriter(String filename,
        boolean restart)
```

Gets the DeckWriter implementation for this model if one is available.

Parameters:

`filename` - a String containing the full path of the file to be written.
`restart` - if true, this DeckWriter should write a restart deck if possible

Returns:

a DeckWriter configured to write an input deck to the given file.

---

addPopupMenuEntries

```
public boolean addPopupMenuEntries(JPopupMenu menu)
```

Appends this model's custom popup menu entries to the given popup menu.

For example: A model's "Properties" item.

Parameters:

`menu` - the JPopupMenu to append this model's menu entries to.

Returns:

true if any entries were added.

---

checkModel

```
public boolean checkModel()
```

Performs checks for accuracy on the current AbstractModel

Returns:

the status of the model after checks

---

buildErrorReport

```
public ModelErrorReport buildErrorReport()
```

---

getLoopCheck

```
public com.cafean.client.organize.AbstractLoopCheck getLoopCheck(Vector components)
```

Retrieves a new loop checker appropriate for checking hydraulic loops in this model.

Parameters:

`components` - the Vector of HydroComponent instances to check

Returns:

a LookCheck instance appropriate for this model

---

performLoopCheck

```
public boolean performLoopCheck(Vector components)
```

Performs a loop check on the given vector of components.

See Also:

getLoopCheck(Vector)

---

getComponentGroups

`public ComponentNumberGroup[] getComponentGroups()`

Retrieves an array of the ComponentNumberGroup instances used by this model to ensure that each component has a valid component number.

Returns:

a ComponentNumberGroup[] containing this model's component number groups; or null if ComponentNumberGroups are not supported by this model.

---

findComponentGroup

`public ComponentNumberGroup findComponentGroup(Category category)`

Retrieves the appropriate ComponentNumberGroup instance for the given Category.

Parameters:

`category` - the Category to retrieve a ComponentNumberGroup for.

Returns:

the ComponentNumberGroup used to ensure valid component numbers for the given Category of components.

---

getInitialCCNumber

`public int getInitialCCNumber(Category category)`

Retrieves the first CC number available to components of the given component class. This is simply the beginning of the pool of possible CC numbers for the given cc group.

NOTE: The returned CC number may or may not be in use.

Returns:

the lowest possible CC number for the given Category

See Also:

getNextCCNumber(Category, int)

---

getMaxCCNumber

`public int getMaxCCNumber(Category category)`

Retrieves the last CC number available to components of the given component Category. This is simply the end of the pool of possible CC numbers for the given cc group.

NOTE: The returned CC number may or may not be in use.

Returns:

the highest possible CC number for the given Category

See Also:

getNextCCNumber(Category, int)

setCCNumberIncrement

`public void setCCNumberIncrement(int increment)`

Sets the increment number for retrieved CC numbers.

Parameters:

`increment` - the new increment used for allocating new cc numbers.

See Also:

`getNextCCNumber(Category, int)`

---

getCCNumberIncrement

`public int getCCNumberIncrement()`

Retrieves the increment number for retrieved CC numbers.

Returns:

the increment used for allocating new cc numbers

See Also:

`getNextCCNumber(Category, int)`

---

getNextCCNumber

`public int getNextCCNumber(Category category)`

Retrieves a unique and available CC number for the given component Category.

Parameters:

`category` - the Category of the component requiring a new CC number

See Also:

`getNextCCNumber(Category, int)`

---

getNextCCNumber

```
public int getNextCCNumber(Category category,
        int requestedNumber)
```

Retrieves a new and available CC number for the given component Category. If the requested cc number of available it will be returned, otherwise the next available CC number will be returned.

Available is defined as: greater than or equal to the initial cc number, lower than or equal to the maximum cc number and not currently used by another component in the given cc group.

Parameters:

`category` - the Category of the component requiring a new CC number

`requestedNumber` - The desired CC number. If it is available, it will be returned.

Returns:

the requested cc number if available or a newly allocate cc number.

getLastCCNumber

```
public int getLastCCNumber(Category category)
```

Retrieves the largest CC number in use that is available to components of the given component Category.

Parameters:

`category` - the Category of components to find the last cc for

Returns:

the largest cc number in use by the cc group for the given category

isValidCCNumber

```
public boolean isValidCCNumber(int ccNumber,
        Category category)
```

Returns true if the given cc number is theoretically valid.

ccNumberCheck

```
public boolean ccNumberCheck(Category category,
        String emptyMsg,
        String listName,
        String context,
        boolean warnEmpty)
```

Checks that all CC numbers in the given component list are valid and unique. Errors are printed for invalid and duplicate cc numbers.

Parameters:

`category` - the Category of components to check CC numbers for.
`emptyMsg` - a String containing the empty list message such as There are no
`listName` - the name of the given list of components
`context` - a String containing a description of the given context such as in this model.
`warnEmpty` - if true a warning will be added to the MessageWindow if `components` is empty.

layoutComponents

```
public abstract void layoutComponents(Vector drawnComponents,
        ProgressMon progress)
```

Lays out the given drawn components using a plugin-specific organization method.

Parameters:

`drawnComponents` - a Vector of DrawnComponent instances to be organized.
`progress` - a ProgressMon to notify of progress.

objectAdded

```
public void objectAdded(GenericObject object,
        boolean assignIdent)
```

Updates the model's top ident and the given object's ident and DB_ID upon addition to the model.

If assignIdent is true, or `object`'s ident is 0, a new ident will be assigned and `object`'s `DB_ID` will be set to it's previous ident.

Parameters:

`object` - the GenericObject being added to the model.
`assignIdent` - if true, a new ident will be assigned, if false the top ident will be updated.

---

reportModelCheck

```
public void reportModelCheck(boolean status)
```

Deprecated. *0.27.0 - December 2007*

Displays a report of the model check to the message window and a OptionPane.

Parameters:

`status` - the result of `checkModel()`

---

removeElement

```
public boolean removeElement(GenericObject element)
```

Removes the specified GenericObject from the model.

Parameters:

`element` - the GenericObject to remove from the model

Returns:

true if removal successful, FALSE otherwise.

---

addElement

```
public void addElement(GenericObject element)
```

Adds an element to the model, adjusting it's DB_ID and ident in the process.

Parameters:

`element` - the GenericObject to add to the model.

---

addElement

```
public void addElement(GenericObject element,
        boolean adjustIdent)
```

Adds a GenericObject to the model.

Parameters:

`element` - the GenericObject to add to the model.
`adjustIdent` - if true, the GenericObject's DB_ID will be set to it's ident, and it's ident will be assigned to a new model-unique ident.

---

getElementIterator

```
public Iterator getElementIterator()
```

Retrieves a new Iterator for use in traversing this model's Element list.

Returns:
an Iterator for traversing the element list

---

getElementCount

```
public int getElementCount()
```

Retrieves a count of the elements contained in this model's element list.

Returns:
the number of elements in the element list.

---

getElementAt

```
public GenericObject getElementAt(int i)
```

Retrieves the element at the given index.

Returns:
the GenericObject at the given index in the element list

---

findElementByIdent

```
public GenericObject findElementByIdent(int ident)
```

Retrieves the element in this model with the given ident.

Parameters:
`ident` - the ident of the desired element.

Returns:
the GenericObject in the element list with the given ident

---

findElementByDB_ID

```
public GenericObject findElementByDB_ID(int dbid)
```

Retrieves the GenericObject in this model with the given DB_ID.

Parameters:
`dbid` - the DB_ID of the desired element

---

getNewElementIdent

```
public static int getNewElementIdent(AbstractModel model,
        int dbid,
        boolean preserveUnresolved,
        boolean useDbId)
```

Retrieves the ident of the element with the given DB_ID or ident in the given model. If dbid is 0, a 0 is returned.

Parameters:

`preserveUnresolved` - if true, and no element is found `dbid` will be returned. if false, 0 will be returned.
`useDbId` - if true, ident references will be reconnected via findElementByDB_ID; if false, ident references will be reconnected via findElementByIdent

---

setModelDB_ID

`public void` **`setModelDB_ID`**`(int dbid)`

Sets this model's DB_ID to the given dbid.

Parameters:
`dbid` - the new DB_ID value to use

See Also:
`getModelDB_ID()`

---

setIdent

`public void` **`setIdent`**`(AbstractModel model)`

Sets the ident of this model to that of the given model for use in copy/paste.

The model's ident is used to determine if a paste operation should treat the target model as the same model. When pasting to the same model some operations (such as copying shared components) are not performed.

---

setIdent

`public void` **`setIdent`**`(int uid)`

Sets the unique ident of this object. This should only be used by the AbstractModel or related structures when adding this object to a model.

---

getModelDB_ID

`public int` **`getModelDB_ID`**`()`

Retrieves this model's ID on the database server.

Unlike that of AbstractComponents, this DB_ID is actually a database ID. This is only used for submitting and restarting runs when a database server is being used.

Returns:
this model's database ID.

---

setProjectDB_ID

`public void` **`setProjectDB_ID`**`(int pdbid)`

Sets this model's database project ID.

Parameters:
`pdbid` - the new database project ID.

See Also:
`getProjectDB_ID()`

getProjectDB_ID

`public int` **`getProjectDB_ID`**`()`

Retrieves the ID of this model's project in the database.

This is only used for submitting and restarting runs when a database server is being used.

Returns:
the ID of this model's project in the database.

reconnectIdentReferences

`public void` **`reconnectIdentReferences`**`(boolean preserveUnresolved, boolean useDbId)`

Reconnects the ident references of this models directly contained objects. These objects include `component lists`, program options, model options, etc.

Models that include root components are required to handle the reconnection of those components.

Parameters:
`preserveUnresolved` - if true, ident references that aren't resolvable will be left dangling, if false, they will be set to 0.
`useDbId` - if true, ident references will be reconnected via find_x_ByDB_ID; if false, ident references will be reconnected via find_x_ByIdent

See Also:
`GenericObject.reconnectIdentReferences(boolean, boolean)`
`getNewComponentIdent(int, Category, boolean, boolean)`

clearComponentLookup

`public void` **`clearComponentLookup`**`()`

Clears the pasted component lookup table.

See Also:
`lookupComponent(int)`

lookupComponent

`public AbstractComponent` **`lookupComponent`**`(int dbid)`

Finds the component for the given dbid in the component lookup table. The component lookup table is used during copy/paste operations by `getNewComponentIdent(int, Category, boolean, boolean)` to speed the lookup of particular components.

This table is an internal optimization and should not be required by plug-ins.

Parameters:
`dbid` - the DB_ID of the component to find.

Returns:
the AbstractComponent stored with the given DB_ID.

addComponentLookup

```
public void addComponentLookup(AbstractComponent comp,
        int dbid)
```

Adds the given component to the component lookup table for the given DB_ID. A new lookup table will be created if one is not available.

Parameters:

`comp` - the AbstractComponent to store in the lookup table with the given key DB_ID.
`dbid` - the DB_ID key to use to allow lookups of the given component.

See Also:

`lookupComponent(int)`

clearDbIds

```
public void clearDbIds()
```

Clears the DB_ID's of this model's directly contained objects by setting DB_ID to 0. These objects include `component lists`, program options, model options, etc.

Models that include root components are required to handle the DB_ID clearing of those components.

setParentDB_ID

```
public void setParentDB_ID(int pdbid)
```

Sets this model's parent model's database ID.

Parameters:

`pdbid` - the new parent model ID.

See Also:

`getParentDB_ID()`

getParentDB_ID

```
public int getParentDB_ID()
```

Retrieves the ID of this model's parent model in the database. The parent model is the model that this model is a restart of.

This is only used for submitting and restarting runs when a database server is being used.

Returns:

the ID of this model's parent model in the database.

getSaveFileName

```
public String getSaveFileName()
```

Retrieves the name of the file this model was last saved to or opened from.

Returns:

a String containing the path to the file this model was last saved to or loaded from or null if neither has happened.

---

getSaveFile

```
public java.io.File getSaveFile()
```

Retrieves the file this model was last saved to or opened from.

Returns:

the file this model was last saved to or opened from or null if neither has happened.

---

setSaveFile

```
public void setSaveFile(java.io.File file)
```

Sets the name of the file this model was last saved to or opened from.

Parameters:

`name` - a String containing the path to the file this model was last saved to or loaded from.

---

compareVersion

```
public int compareVersion(GenericObject obj)
```

Compares the version of the given object with that of this model to dermine if the object's version is less than, equal to or greater than the version of this model.

This is method is primarily used to determine if a model has changes that justify a restart, and if so, which components should be part of that restart.

Restartable components are those that have a version greater than or equal to that of their model.

Parameters:

`obj` - the GenericObject to compare major and minor versions with.

Returns:

-1 if obj's major is less than model's major or objs minor is less than model's minor; 0 if model and obj's versions are the same; 1 if obj's major and minor are greater than model's

See Also:

`containsRestartChanges(Iterator)`

---

getModelMajorVersion

```
public int getModelMajorVersion()
```

Returns the major version number of this model.

Returns:

the current major version number

See Also:

`compareVersion(GenericObject)`

---

setModelMajorVersion

`public void setModelMajorVersion(int version)`

Sets this model's major version number.

Parameters:
`version` - the new version number

See Also:
`compareVersion(GenericObject)`

---

getModelMinorVersion

`public int getModelMinorVersion()`

Retrieves this model's minor version number

Returns:
the current minor version number.

See Also:
`compareVersion(GenericObject)`

---

setModelMinorVersion

`public void setModelMinorVersion(int version)`

Sets this model's minor version number.

Parameters:
`version` - the new version number

See Also:
`compareVersion(GenericObject)`

---

incrementMajorVersion

`public void incrementMajorVersion()`

Increments this model's major version number and sets it's minor version to 0.

See Also:
`compareVersion(GenericObject)`

---

incrementMinorVersion

`public void incrementMinorVersion()`

See Also:
`compareVersion(GenericObject)`

cleanUpDeleted

`public void cleanUpDeleted()`

Removes all deleted components from the lists maintained by this model.

isEditingRestart

`public boolean isEditingRestart()`

Returns true if this model is subsequent edits to this model will be considered restart edits and as such, should be graphically represented.

Note: The base implementation returns false;

isRestartableModel

`public boolean isRestartableModel()`

Returns true if this model can be used to export a restart run.

A restartable run is defined as one that has a parent dbid and has some components that have a major version number greater or equal to the major version of this model.

NOTE: Derived classes should note that a model cannot be restartable if it's parent dbid is <= 0;

containsRestartChanges

`public boolean containsRestartChanges(Iterator iterator)`

Returns true if any of the given objects is either deleted or has a major version number greater than or equal to that of this model.

Parameters:

`iterator` - an Iterator through a container of `GenericObject` references to check deleted or compare major version numbers.

setCreateDate

`public void setCreateDate(String cDate)`

Sets this model's creation date to the date in the given String. The given date is parsed with `DateFormat.getDateTimeInstance(DateFormat.MEDIUM,DateFormat.MEDIUM);`

Parameters:

`cDate` - a String containing the new creation date.

See Also:

`DateFormat.getDateTimeInstance()`

getCreateDate

`public String getCreateDate()`

Gets this model's creation date as a String. return a String containing this model's creation date.

See Also:
`setCreateDate(String)`

---

getOwner

`public String **getOwner**()`

Retrieves the username of the creator of this file.

Returns:
a String containing the username of the creator of this file or the current user if no owner is set.

---

setOwner

`public void **setOwner**(String name)`

Sets the username of the creator of this file.

Parameters:
`name` - a String containing the username of the creator of this file.

---

splitPipe

`public boolean **splitPipe**(Vector selected)`

This function splits a pipe into two pipes.

This method is part of the ModelEditor Renodalization system and should be overriden by models that wish to support the Split Pipe feature.

This is the empty error version of this function to indicate which models don't support this feature

Parameters:
`selected` - the Vector of selected drawn components

---

joinPipe

`public boolean **joinPipe**(Vector selected)`

This function joins two pipes into one.

This method is part of the ModelEditor Renodalization system and should be overriden by models that wish to support the Join Pipe feature.

This is the empty error version of this function to indicate which models don't support this feature.

Parameters:
`selected` - the Vector of selected drawn components

---

renodalizePipeCells

`public boolean **renodalizePipeCells**(AbstractComponent component)`

This function renodalizes a pipe cells.

This method is part of the ModelEditor Renodalization system and should be overriden by models that wish to support the Renodalize Cells feature.

This is the empty error version of this function to indicate which models don't support this feature..

Parameters:

`component` - an AbstractComponent reference to the pipe to renodalize

---

getName

```
public String getName()
```

Returns this model's name. The default implementation currently uses this model's ProgramOptions object.

---

setName

```
public void setName(String name)
```

Override GenericObject SetName to make sure view title is updated when this is called. The default implementation currently uses this model's ProgramOptions object.

Parameters:

`name` - a String containing the new name.

---

setDescription

```
public void setDescription(String desc)
```

Sets this object's description to the given desc. Proxied to program options.

See Also:

`GenericObject.setDescription(String)`

---

getDescription

```
public String getDescription()
```

Retrieves the description of this object. Proxied to program options.

See Also:

`GenericObject.getDescription()`

---

addComment

```
public void addComment(String comment)
```

Appends the given string as to the list of comments in this object Proxied to program options.

See Also:

`GenericObject.addComment(String)`

---

addMultipleComments

`public void addMultipleComments(Vector com)`

Add the given Vector of Comments to this object. Proxied to program options.

See Also:
`GenericObject.addMultipleComments(Vector)`

---

getNumComments

`public int getNumComments()`

Retrieves the count of Comment objects stored in this object. Proxied to program options.

See Also:
`GenericObject.getNumComments()`

---

deleteComment

`public void deleteComment(int num)`

Removes the given Comment from this object's comment list. Proxied to program options.

See Also:
`GenericObject.deleteComment(int)`

---

deleteAllComments

`public void deleteAllComments()`

Removes all Comments from this object. Proxied to program options.

See Also:
`GenericObject.deleteAllComments()`

---

showComment

`public String showComment(int num)`

Retrieves the text in the comment at the given index. Proxied to program options.

See Also:
`GenericObject.showComment(int)`

---

getComment

`public String getComment(int num)`

Retrieves the comment at the given index. Proxied to program options.

See Also:
`GenericObject.getComment(int)`

getModelOptions

`public Object getModelOptions()`

Returns the java bean object that contains this model's "model options" if this model uses such a thing.

getRootComponents

`public AbstractComponent[] getRootComponents()`

Returns the set of AbstractComponents that should appear in the Navigator as peer nodes to ModelOptions.

getProgramOptions

`public ProgramOptions getProgramOptions()`

Deprecated.

Retrieves this model's ProgramOptions object, or null if this model type does not use ProgramOptions.

Returns:

the ProgramOptions object for this model or null if none is used.

setProgramOptions

`public void setProgramOptions(ProgramOptions options)`

Sets the program options object that will be used by this model.

Parameters:

`options` - a ProgramOptions object to be used for this model's options.

renumberComponents

```
public void renumberComponents(AbstractComponent[] components,
          int offset)
```

Ensures that the given components have appropriately unique CC numbers within their model. Sets new unique component numbers on all components with duplicate or invalid numbers.

Parameters:

`components` - the AbstractComponent[] containing the components to be renumbered

`offset` - the amount to offset each cc number by; if 0, renumber using the first available numbers for each component

getLenScaleFactor

`public double getLenScaleFactor()`

Gets the scale factor used to adjust cell length for drawn components rendering components of this model.

getWidthScaleFactor

`public double getWidthScaleFactor()`

Gets the scale factor used to adjust cell width for drawn components rendering components of this model.

setLenScaleFactor

```
public void setLenScaleFactor(double fact)
```

Gets the scale factor used to adjust cell length for drawn components rendering components of this model.

setWidthScaleFactor

```
public void setWidthScaleFactor(double fact)
```

Gets the scale factor used to adjust cell width for drawn components rendering components of this model.

getDocumentLinks

```
public DocumentLink[] getDocumentLinks(Object obj,
        String docType)
```

Return the array of document links for a given object.

loadDrawnComponent

```
public DrawnComponent loadDrawnComponent(com.apt.xdr.PibBlock block)
```

Creates a custom DrawnComponent derivative from the given PibBlock.

Returns:
null of no custom components exist for the given block

storeDrawnComponent

```
public com.apt.xdr.PibBlock storeDrawnComponent(java.awt.Component component)
```

Creates a custom PibBlock derivative from the given Component.

Returns:
null of no custom block exist for the given component

executeUserDefinedFunctions

```
public void executeUserDefinedFunctions()
```

Executes all user defined functions in this model.

findEquivalentSharedComponent

```
public SharedComponent findEquivalentSharedComponent(SharedComponent shared)
```

Retrieves an equivalent SharedComponent for the given component. Shared components are carried over when copy/pasting between models. This allows for such things as shared geometry references.

See Also:
SharedComponent.isEquivalent(SharedComponent)

getExportNote

`public String getExportNote()`

Getter for property exportNote.

Returns:
Value of property exportNote.

setDirty

`public void setDirty(boolean dirty)`

sets this model dirty; a dirty model has been modified since it was saved.

isDirty

`public boolean isDirty()`

if true, this model has been modified since it was saved.

setExportNote

`public void setExportNote(String exportNote)`

Setter for property exportNote.

Parameters:
`exportNote` - New value of property exportNote.

toString

`public String toString()`

getUnitsDisplay

`public String[] getUnitsDisplay()`

Creates an array of strings for displaying model units for selection.

findReal

`public Real findReal(String siUnitsName)`

Finds the Real derivative who's SI units are equal to the given String or an empty Dimless if none is found.

getRealByIndex

`public Real getRealByIndex(int index)`

Returns the Real at the given index. This method allows an editor to select from the array of units by index.

getUnitIndex

`public int **getUnitIndex**(Real unit)`

Finds the index of the given unit or -1 if the unit is not found inside this model.

getUnitIndex

`public int **getUnitIndex**(String siUnitsName)`

Finds the index of the unit who's SI units are equal to the given String or -1.

getDimensionless

`public Real **getDimensionless**()`

Returns a new dimensionless value based on the units for this model's plugin.

getUnits

`public int **getUnits**()`

Returns the current units for this model.

setUnits

`public void **setUnits**(int units)`

Sets the current units for this model.

setUnitsConstrained

`public void **setUnitsConstrained**(int units)`

Sets the units constrained by the available unit types.

getExportUnits

`public int **getExportUnits**()`

This gets the units for this model's ASCII export. The default is the same units that are used everywhere else. Some codes however are written to accept only files using certain units. This method is called before any ASCII is written out, including the AsciiViewer.

getValidationTests

`public ValidationTest[] **getValidationTests**()`

This gets the array of ValidationTests from the current model. This defaults to returning an empty array.

getValidationOptions

`public ValidationOptions **getValidationOptions**()`

This gets the ValidationOptions that is stored inside a given model. This defaults to return null.

---

executeModelValidations

`public boolean` **`executeModelValidations`**`(boolean printErrors)`

This executes all of the `tests` that are found in `this` model.

Parameters:

`printErrors` - the boolean flag that determines if errors should be written to the MainFrame.

Returns:

true if all the tests passed.

---

hasModelMetrics

`public boolean` **`hasModelMetrics`**`()`

Return true if this model supports output of model metrics.

---

exportModelMetricsSpec

`public boolean` **`exportModelMetricsSpec`**`(java.io.File dFile)`

Export a file containing the metrics specification for the model.

Parameters:

`dFile` - the output file.

---

exportModelMetrics

`public boolean` **`exportModelMetrics`**`(java.io.File dFile)`

Export a file containing the state of all testable metrics for the given model.

Parameters:

`dFile` - the output file.

---

getModel

`public AbstractModel` **`getModel`**`()`

---

getRootNodeComparator

`public Comparator` **`getRootNodeComparator`**`()`

Returns a Comparator object suitable for sorting the root nodes of the model node that represents this model in the Navigator. A model's "root nodes" consist of the Model Options node, the nodes for each of the Categories returned by `getCategories()`, and the components returned by `getRootComponents()`.

Returns:

a java.util.Comparator object suitable for sorting root nodes or null if no sorting is desired.

See Also:
`Collections.sort(java.util.List)`

---

getComponentComparator

```
public Comparator getComponentComparator(Category category)
```

Returns a Comparator object suitable for sorting components in the given category.

This base implementation returns `AbstractComponent.getCCNumberComparator()` for all categories.

Returns:
a java.util.Comparator object suitable for sorting components in the given Category. Must not be null.

See Also:
`Collections.sort(java.util.List)`

---

createCategoryView

```
public void createCategoryView(Category category,
        boolean layout,
        boolean waitVisible)
```

Opens a new view for the current model, initializes the name, and adds all of the components with the selected category. This may be overriden in plug-ins for specific create view methods.

Parameters:
`category` - the Category from the navigator.

---

addToView

```
public void addToView(Category category,
        ViewComponent view,
        boolean select)
```

Adds this navigator node to the View Component

---

getClassification

```
public Classification getClassification()
```

Returns the minimum classification level required for this model.

See Also:
`Classification`

---

getOwnershipManager

```
public OwnershipManager getOwnershipManager()
```

Returns the ownership manager for this model. This base implementation returns null as most ModelEditor plug-ins do not support object ownership.

---

findModelDocuments

```
public com.cafean.client.mdocs.AbstractDocument[] findModelDocuments(AbstractComponent
comp)
```

Retrieves the model documents (notes, etc.) that correspond to the given component.

Parameters:

`comp` - the AbstractComponent for which to retrieve model documents

Returns:

an AbstractDocument[] containing the corresponding documents

---

getDocLinks

```
public Object getDocLinks(String key)
```

Provides a hashtable containing the reference material links.

Parameters:

`model` - the AbstractModel to which to retrieve document links.
`key` - the String of reference material requested.

Returns:

an object containing the requested document reference data.

---

updateDocLinks

```
public static void updateDocLinks(String plugin_id)
```

Provides the ability to update the existing reference documentation at runtime by removing the old reference documentation for the current plugin (if it exists) and importing the new documentation.

---

getPluginDocFileData

```
public static String[][] getPluginDocFileData(String plugin_id)
```

Provides a listing of Plugin File data by media name/filename

Parameters:

`plugin_id` - the String text plug-in id for doc info.

Returns:

the String[][] of media name/file name pairs.

---

updatePluginDocFileData

```
public static void updatePluginDocFileData(String plugin_id,
        String[] filenames)
```

Updates the plugin documentation media filenames.

Parameters:

`plugin_id` - the String plug-in id to update media filename data.
`media` - the String[] of media names to update.

`filenames` - the String[] of filenames to update.

---

createNumericsMapping

```
public String[][] createNumericsMapping()
```

This routine provides the ability to map user numerics which are too long to export to a ascii file, to a temporary token placed in the ascii export.

Returns:

a String[][] where the first dimension is name of the numeric which will appear in the export, the second dimension is the original name provided for reconnection purposes.

---

updateNumerics

```
public void updateNumerics(String[][] numerics)
```

This routine is fired after the numerics mapping has been created from the existing model numerics and can be used to modify the existing numerics appropriately.

Parameters:

`String[][]` - the user numerics mapping created.

---

createModelPackage

```
public void createModelPackage(java.io.File pkg,
        java.io.File ascii,
        java.io.File[] additionalFiles)
```

This routine will bundle up a given model into a compressed zip file. Resources included are as follows: - Model Save File - Model ASCII Export - Model View Templates

Parameters:

`pkg` - the File bundle

`ascii` - the File which contains the ASCII export.

---

findComponentByUniqueID

```
public AbstractComponent findComponentByUniqueID(String id,
        String categoryName)
```

Retrieves the component in this model that has the given Unique Identifier and is in a subset of the given category.

Parameters:

`cc` - the unique component identifier of the desired component.

`category` - a String containing the name of the category

Returns:

the AbstractComponent desired or null if not found.

---

getHtmlUnits

```
public String getHtmlUnits(Class unitsClass)
```

Retrieve the current units string in HTML format for an engineering units class.

Parameters:

`unitsClass` - the class name of the the unit type.

---

getNavigatorComponent

`public NavigatorComponent` **`getNavigatorComponent`**`()`

Returns a `com.cafean.client.ui.navigator.NavigatorComponent` implementation used to display the contents of the model in the Navigator. The default implementation returns a `com.cafean.client.ui.navigator.DefaultNavigatorModelComponent` that displays model components in a tree.

---

getModelPanel

`public JPanel` **`getModelPanel`**`()`

Returns a `javax.swing.JPanel` to be displayed when the model is selected in the Navigator.

---

postEdit

`public void` **`postEdit`**`(UndoableEdit edit)`

Adds the given undoable edit to this model's undo/redo stack.

Parameters:

`edit` - the UndoableEdit that has occurred.

---

getUndoManager

`public SnapUndoManager` **`getUndoManager`**`()`

Retrieves the undo manager for the undo/redo stack for this model.

Returns:

the SnapUndoManager handling undo/redo for this model.

com.cafean.client.analysis
Class Category

```
java.lang.Object
   |
   +-com.cafean.client.analysis.Category
```

All Implemented Interfaces:
Cloneable, Comparable

---

public class **Category**
extends Object
implements Comparable, Cloneable

A node in a hierarchy of component types within a `model`.

Categories are used to represent each specific `component` type. Each of these categories can also have parent and child Categories representing subsets and supersets of `component` types.

Any given model should have a Category for each of its component types and a `root Category` for each component or subset of components that are to be manipulated by the ModelEditor user interface.

WARNING: Do not check Category equivalence with ==. Use `equals(Object)` instead as derivative plugins may need to replace a defined Category.

An abbreviated example of creating a hierarchy of Category objects:

```
      public static final Category CAT_PIPE = new Category("Pipes", null, ICON_PIPE,
ImageMgr.class.getResource("pipe32.gif"));
      public static final Category CAT_PUMP = new Category("Pumps", null, ICON_PUMP,
ImageMgr.class.getResource("pump32.gif"));
      private static final Category[] HYDROS = { CAT_PIPE, CAT_PUMP };
      public static final Category CAT_HYDRAULIC = new Category("Hydraulic Components", null,
                                                                ICON_HYDRAULIC, null,
                                                                HYDRAULIC_COMPONENTS );
      private static final Category[] COMPONENTS = { CAT_HYDRAULIC, CAT_CONTROL_SYS };
```

With the above example, `getCategories` would then be overriden like this:

```
public Category[] getCategories()
{
    Category[] supers = super.getCategories();
    Category[] categories = new Category[COMPONENTS.length+supers.length];
    System.arraycopy(COMPONENTS, 0, categories, 0, COMPONENTS.length );
    System.arraycopy(supers, 0, categories, COMPONENTS.length, supers.length );
    return categories;
}
```

Constructor Summary

public	`Category(String name, String tooltip, ImageIcon treeIcon, java.net.URL imageURL)` Creates a new Category with the given name, tooltip, tree icon and image URL that is considered a `visual` Category.
public	`Category(String name, String tooltip, ImageIcon treeIcon, java.net.URL imageURL, boolean visual)` Creates a new Category with the given name, tooltip, tree icon, image URL, and `visual` status.
public	`Category(String name, String tooltip, ImageIcon treeIcon, java.net.URL imageURL, Category[] children)` Creates a new Category with the given name and child nodes.

Method Summary

void	`addChild(Category category)` Adds the given category as a child to this Category.
Object	`clone()` Produces a shallow clone of this Category, including it's fields and child array but not the children themselves.
int	`compareTo(Object object)`
Category	`deepClone()` Produces a deep clone of this Category, including it's fields and clones of it's child nodes.
boolean	`equals(Object o)` Checks equivalence based on reference equivalence or name equivalence.
Category[]	`getChildren()` Retrieves this `Category`'s direct children.
ImageIcon	`getCollapsedIcon()` Return the collapsed tree node icon for use in the Navigator component.
ImageIcon	`getExpandedIcon()` Return the expanded tree node icon for use in the Navigator component.

java.net.URL	getImageURL() Returns the URL of this Category's Tool Box image as retrieved from Class.getResource.
Category	getParent() Retrieves this Category's direct parent.
Category	getRootParent() Retrieves this Category's root parent.
String	getToolTipText() Gets the tooltip text to use for this Category when name is not appropriate.
boolean	isSubset(Category category) Returns true if this category is a subset of the given category.
boolean	isSuperset(Category category) Returns true if this category is a superset of the given category.
boolean	isVisual() Returns true if this category is of Visual Components.
String	toString()

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.lang.Comparable

compareTo

Constructors

Category

```
public Category(String name,
                String tooltip,
                ImageIcon treeIcon,
                java.net.URL imageURL)
```

Creates a new Category with the given name, tooltip, tree icon and image URL that is considered a visual Category.

Parameters:

name - a String containing the desired name of this category.
tooltip - a String containing the tooltip text to use for this Category
treeIcon - the ImageIcon to use for this Category in the Navigator.
imageURL - the URL of this Category's Tool Box image as retrieved from Class.getResource.

Category

```
public Category(String name,
                String tooltip,
                ImageIcon treeIcon,
                java.net.URL imageURL,
                boolean visual)
```

Creates a new Category with the given name, tooltip, tree icon, image URL, and `visual` status.

Parameters:

name - a String containing the desired name of this category.
`tooltip` - a String containing the tooltip text to use for this Category
`treeIcon` - the ImageIcon to use for this Category in the Navigator.
`imageURL` - the URL of this Category's Tool Box image as retrieved from `Class.getResource`.
`visual` - if true, this Category is to be considered `visual`.

Category

```
public Category(String name,
                String tooltip,
                ImageIcon treeIcon,
                java.net.URL imageURL,
                Category[] children)
```

Creates a new Category with the given name and child nodes. This Category will not be `visual` as it will have children.

Parameters:

`name` - a String containing the desired name of this category.
`tooltip` - a String containing the tooltip text to use for this Category
`treeIcon` - the ImageIcon to use for this Category in the Navigator.
`imageURL` - the URL of this Category's Tool Box image as retrieved from `Class.getResource`.
`children` - a Category[] of child nodes.

Methods

addChild

```
public void addChild(Category category)
```

Adds the given category as a child to this Category.

---

getChildren

```
public Category[] getChildren()
```

Retrieves this `Category`'s direct children.

Returns:

a Category[] containing the direct child Catagories

---

getParent

```
public Category getParent()
```

Retrieves this `Category`'s direct parent.

Returns:
the Category that is this Category's direct parent.

---

getRootParent

`public Category getRootParent()`

Retrieves this `Category`'s root parent. The root parent is the Category that is this Category's parent and has no parent of it's own.

Returns:
the Category that is this Category's direct parent.

---

isSuperset

`public boolean isSuperset(Category category)`

Returns true if this category is a superset of the given category.

Parameters:
`category` - the `Category` that this may or may not be a superset of

Returns:
true if this is a superset of the given `Category`

---

isSubset

`public boolean isSubset(Category category)`

Returns true if this category is a subset of the given category.

Parameters:
`category` - the `Category` that this may or may not be a subset of

Returns:
true if this is a subset of the given `Category`

---

compareTo

`public int compareTo(Object object)`

---

toString

`public String toString()`

---

getToolTipText

`public String getToolTipText()`

Gets the tooltip text to use for this Category when name is not appropriate.

Returns:
a String containing the tooltip for this Category

---

isVisual

```
public boolean isVisual()
```

Returns true if this category is of Visual Components.

Visual components have no children and can appear in views and the Tool Box. They should also have `DrawnComponents` defined.

---

getExpandedIcon

```
public ImageIcon getExpandedIcon()
```

Return the expanded tree node icon for use in the Navigator component.

Returns:
the tree icon for this Category.

---

getCollapsedIcon

```
public ImageIcon getCollapsedIcon()
```

Return the collapsed tree node icon for use in the Navigator component.

Returns:
the tree icon for this Category.

---

getImageURL

```
public java.net.URL getImageURL()
```

Returns the URL of this Category's Tool Box image as retrieved from `Class.getResource`.

---

equals

```
public boolean equals(Object o)
```

Checks equivalence based on reference equivalence or name equivalence.

See Also:
`Object.equals(java.lang.Object)`

---

clone

```
public Object clone()
```

Produces a shallow clone of this Category; including it's fields and child array but not the children themselves.

See Also:
`java.lang.Cloneable`

deepClone

`public Category deepClone()`

Produces a deep clone of this Category, including it's fields and clones of it's child nodes.

See Also:

`clone()`
`java.lang.Cloneable`

com.cafean.client.analysis
Interface ComponentCreator

public interface **ComponentCreator**
extends

This interface allows for custom creation code to be passed around.

Method Summary

void	createComponent(AbstractModel model) Creates a new component, and adds it to the given model

Methods

createComponent

`public void createComponent(AbstractModel model)`

Creates a new component, and adds it to the given model

com.cafean.client.analysis
Interface ComponentElement

All Superinterfaces:
ModelElement

All Known Implementing Classes:
SnapPreferences, DummyCompElement, AbstractComponent

---

public interface **ComponentElement**
extends ModelElement

An interface describing an object that is contained by an `AbstractComponent` either directly or by being a part of an object that is contained by an `AbstractComponent`.

Method Summary

`AbstractComponent`	`getComponent()` Retrieves the `AbstractComponent` that contains this ComponentElement.
`ComponentElement`	`getOwner()` Retrieves the owner of this ComponentElement.

Methods inherited from interface `com.cafean.client.analysis.ModelElement`

`getModel`

Methods

getComponent

`public AbstractComponent` **`getComponent`**`()`

Retrieves the `AbstractComponent` that contains this ComponentElement.

This method may call a parent's getComponent() and so on rather than a direct reference.

Returns:
the AbstractModel that contains this ModelElement

---

getOwner

`public ComponentElement` **`getOwner`**`()`

Retrieves the owner of this ComponentElement.

com.cafean.client.analysis
Class ComponentList

```
java.lang.Object
   |
   +-com.cafean.client.analysis.ElementList
      |
      +-com.cafean.client.analysis.ComponentList
```

All Implemented Interfaces:
Cloneable, Cloneable

---

public class **ComponentList**
extends ElementList
implements Cloneable, Cloneable

A storage class for AbstractComponent instances.

ComponentList requires that all it's contained components have unique ident numbers.

Components are optionally stored and retrieved in sorted order via binary searches for efficiency.

Constructor Summary

public	`ComponentList(Category category, boolean sorted)` Creates a new instance of ComponentList with the given category name and the given initial sorted state.

Method Summary

`AbstractComponent`	`findByCC(int cc, Category category)` Retrieves the AbstractComponent with the given component number.
`boolean`	`isContained(Category category)` Returns true if the given Category is contained in this list.
`Iterator`	`iterator(Category category)` Retutrns an Iterator object for iterating through this list.
`int`	`size(Category category)` Returns the number of components in this list that are part of the given category.
`AbstractComponent[]`	`toArray(Category category)` Returns an array containing all of the components in this list that are in the given Category.
`String`	`toString()` Returns a String representation of this ComponentList.

Methods inherited from class `com.cafean.client.analysis.ElementList`

add, clear, clearDbIds, clone, contains, findByDB_ID, findByIdent, findByIdent, get, getCategory, indexOf, isSorted, iterator, reconnectIdentReferences, remove, removeDeleted, setCategory, size, sort, toArray, toArray, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

ComponentList

```
public ComponentList(Category category,
                     boolean sorted)
```

Creates a new instance of ComponentList with the given category name and the given initial sorted state.

Parameters:

category - the Category that is the superset of all components that will be stored in this ComponentList.
sorted - if true all additions will be inserted in thier sorted positions.

See Also:

ElementList.sort()

Methods

findByCC

```
public AbstractComponent findByCC(int cc,
        Category category)
```

Retrieves the AbstractComponent with the given component number. if the component number is 0 null is returned.

Parameters:

cc - the cc number of the desired component.
category - the Category of the desired component

Returns:

an AbstractComponent corresponding to the search parameters

size

```
public int size(Category category)
```

Returns the number of components in this list that are part of the given category. If the category is this list's category, ElementList.size() is returned.

Parameters:

category - the Category to return the number of components in.

toArray

`public AbstractComponent[] toArray(Category category)`

Returns an array containing all of the components in this list that are in the given Category. This will return an empty array if there are no components inside the list with the given category.

Parameters:

`category` - the Category to components for

Returns:

an AbstractComponent[] containing in the given Category.

---

toString

`public String toString()`

Returns a String representation of this ComponentList.

---

iterator

`public Iterator iterator(Category category)`

Retutrns an Iterator object for iterating through this list.

Returns:

an Iterator suitable for traversing the components in this ComponentList that are in the given Category.

---

isContained

`public boolean isContained(Category category)`

Returns true if the given Category is contained in this list.

Parameters:

`category` - the Category to check

com.cafean.client.analysis
Interface ComponentListener

All Known Implementing Classes:
DrawnComponent, AsciiViewer

---

public interface **ComponentListener**
extends

A component listener listens to an abstract component for commands to update their rendering of the component.

Method Summary

void	componentChanged(ComponentChangedEvent evt) A component changed event tells the listener that the internal data of the component has changed.
void	componentConnected(Connection con) A component connected event tells the listener when a component completes a connection to a different component.
void	componentDeleted() A component deleted event tells the listener that the component has been deleted by the user.
void	componentDisconnected(Connection con) A component disconnected event tells the listener when a component disconnects from a different compnoent.

Methods

componentChanged

```
public void componentChanged(ComponentChangedEvent evt)
```

A component changed event tells the listener that the internal data of the component has changed. This causes DrawnComponents to re-initialize their data.

---

componentDeleted

```
public void componentDeleted()
```

A component deleted event tells the listener that the component has been deleted by the user. The listener should stop rendering that object and remove itself from the model as well.

---

componentConnected

```
public void componentConnected(Connection con)
```

A component connected event tells the listener when a component completes a connection to a different component.

Parameters:

con - The Connection that has just been created.

---

componentDisconnected

```
public void componentDisconnected(Connection con)
```

A component disconnected event tells the listener when a component disconnects from a different compnoent.

com.cafean.client.analysis
Class Connection

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.AbstractComponent
         |
         +-com.cafean.client.analysis.Connection
```

All Implemented Interfaces:
IdentHolder, StateEditable, Cloneable, Checkable, ComponentElement, Cloneable

Direct Known Subclasses:
ConnectionBean

---

public abstract class **Connection**
extends AbstractComponent

A class representing a connection between two components in an AbstractModel. The two sides are referred to as `left` and `right` and have their `idents` stored in the Connection.

`ConnectionData` objects are used to describe each side and can be used, for instance, to determine which `ConnectingPt` in a `DrawnComponent` corresponds to a given Connection, or passed to `AbstractComponent.connectTo(AbstractComponent, ConnectionData, ConnectionData)` to describe the desired Connection.

The convenience methods `getThisSideData(AbstractComponent)`, `getOtherSide(AbstractComponent)` and `getOtherSideData(AbstractComponent)` have been provided to simplify Connection related code.

See Also:
`DrawnComponent.createConnectionPt(double, double, double, int, int, int, int, ConnectionData)`

Fields inherited from class `com.cafean.client.analysis.AbstractComponent`
`leftDiffComponent`, `leftDiffName`, `leftShortDiffName`, `rightDiffComponent`, `rightDiffName`, `rightShortDiffName`

Fields inherited from class `com.cafean.client.analysis.GenericObject`
`DATA_COMPLETE`, `DATA_ERROR`, `DATA_INCOMPLETE`, `DATA_WARNING`

Fields inherited from interface `javax.swing.undo.StateEditable`
`RCSID`

Constructor Summary

public	`Connection()` Creates a new instance of Connection with no model and no left or right component.
public	`Connection(AbstractModel model, AbstractComponent leftComp)` Creates a new instance of Connection with the given model and the given left component.

public	Connection(AbstractModel model, AbstractComponent leftComp, int cc) Creates a new instance of Connection with the given model and the given left component.
public	Connection(AbstractModel model, int leftComp, int rightComp, int cc) Creates a new instance of Connection with the given model and the given left component.
public	Connection(AbstractModel model, AbstractComponent leftComp, AbstractComponent rightComp, int cc) Creates a new instance of Connection with the given model and the given left component.

Method Summary

Object	clone()
DrawnComponent	createDrawnComponent() Creates a DrawnConnection to render this Connection.
void	disconnect() Disconnects this Connection between two AbstractComponent objects.
Category	getCategory() Retrieves the most narrow category that this component is a member of. NOTE: This is the Category of the Connection class and has no relevance to the components it connects.
java.awt.Color	getConnectionColor() Returns the Color used to paint this Connection in a View.
java.awt.Stroke	getConnectionStroke() Returns the Stroke used to paint this connection in a View.
Vector	getCustomPopupItems() Creates Custom Menu Items for any popup dialog involving this component
String	getDocDescription(AbstractComponent side, boolean includeComponent) Returns an HTML formatted documentation description from of the given side of this connection.
AbstractComponent	getLeftComponent() Retrieves the left component of this connection via ident.
int	getLeftComponentID() Retrieves the ident of the left component of this connection.
abstract ConnectionData	getLeftConnectData() Retrieves a ConnectionData object describing the connection to the component on the left of this connection.
AbstractComponent	getOtherSide(AbstractComponent component) Retrieves the component that is on the other side of this Connection.
ConnectionData	getOtherSideData(AbstractComponent component) Retrieves the ConnectionData for the other side of this Connection.
AbstractComponent	getRightComponent() Retrieves the right component of this connection via ident.

int	getRightComponentID() Retrieves the ident of the right component of this connection.
abstract ConnectionData	getRightConnectData() Retrieves a ConnectionData object describing the connection to the component on the right of this connection.
ConnectionData	getThisSideData(AbstractComponent component) Retrieves the ConnectionData for this side of this Connection.
boolean	isEqualTo(Connection con) This determines if the Connection passed in is the equivelant of this Connection.
boolean	isIndependentComponent() Returns true if this Connection can exists without it's left and right components.
boolean	isVisual() Returns true if this connection should be visually represented.
String	label()
void	reconnectIdentReferences(boolean preserveUnresolved, boolean useDbId)
void	setLeftComponent(AbstractComponent component) Sets the left component of this connection.
void	setLeftComponentID(int ident) Sets the left component of this Connection by ident.
void	setRightComponent(AbstractComponent component) Sets the right component of this connection.
void	setRightComponentID(int ident) Sets the right component of this Connection by ident.
String	toString()
void	userDisconnect() Disconnects the connection from the Navigator.

Methods inherited from class com.cafean.client.analysis.AbstractComponent

addALDocRef, addComponentListener, addConnection, addMessage, addMessage, addMessage, addToModel, addToModel, canConnectTo, clearConnections, clone, complete, connectTo, connectTo, copy, createDrawnComponent, createSourceData, createTargetData, DBtypeCode, disconnect, disconnectFrom, fireComponentChanged, fireComponentChanged, fireComponentConnected, fireComponentDeleted, fireComponentDisconnected, getALDocDisplayName, getALDocRefs, getALDocRefs, getALDocShortNames, getALDocShortNames, getCatCCComparator, getCategory, getCCNumberComparator, getComponent, getComponentDependencies, getConnectionCount, getConnectionName, getConnections, getConnectionTypes, getCustomPopupActions, getCustomPopupItems, getGroupedConnections, getModel, getName, getNewCompIdent, getOrder, getOrderComparator, getOwner, getRealSize, getSharedComponents, getUniqueID, hasALDocRefs, includeInLoopcheck, isOkayForExport, isOkayForExport, label, popupDataDialog, rebuildConnections, reconnectIdentReferences, reconnectImage, removeALDocRef, removeComponentListener, removeFromModel, removeVerify, restoreALRefState, restoreState, setALDocRefs, setComponentNumber, setDeleted, setModel, setOrder, storeALRefState, toShortString, toString, updateVersion, userDelete, writeName

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder

reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ComponentElement

getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement

getModel

Methods inherited from interface com.cafean.client.analysis.Checkable

isOkayForExport, isOkayForExport

Constructors

Connection

```
public Connection()
```

Creates a new instance of Connection with no model and no left or right component.

Connection

```
public Connection(AbstractModel model,
                  AbstractComponent leftComp)
```

Creates a new instance of Connection with the given model and the given left component.

Parameters:

`model` - the AbstractModel that this Connection is to be part of.
`leftComp` - the AbstractComponent that is the left component of this connection.

Connection

```
public Connection(AbstractModel model,
                  AbstractComponent leftComp,
                  int cc)
```

Creates a new instance of Connection with the given model and the given left component.

Parameters:

`model` - the AbstractModel that this Connection is to be part of.
`leftComp` - the AbstractComponent that is the left component of this Connection.
`cc` - the component number of this Connection

Connection

```
public Connection(AbstractModel model,
                  int leftComp,
                  int rightComp,
                  int cc)
```

Creates a new instance of Connection with the given model and the given left component.

Parameters:

`model` - the AbstractModel that this Connection is to be part of.
`leftComp` - the ident of the AbstractComponent that is the left component of this Connection.
`rightComp` - the ident of the AbstractComponent that is the right component of this Connection.
`cc` - the component number of this Connection

Connection

```
public Connection(AbstractModel model,
                  AbstractComponent leftComp,
                  AbstractComponent rightComp,
                  int cc)
```

Creates a new instance of Connection with the given model and the given left component.

Parameters:

`model` - the AbstractModel that this Connection is to be part of.
`leftComp` - the AbstractComponent that is the left component of this Connection.
`rightComp` - the AbstractComponent that is the right component of this Connection.
`cc` - the component number of this Connection

Methods

getLeftConnectData

```
public abstract ConnectionData getLeftConnectData()
```

Retrieves a ConnectionData object describing the connection to the component on the left of this connection.

---

getRightConnectData

```
public abstract ConnectionData getRightConnectData()
```

Retrieves a ConnectionData object describing the connection to the component on the right of this connection.

---

isIndependentComponent

```
public boolean isIndependentComponent()
```

Returns true if this Connection can exists without it's left and right components.

Non-Independent connections are deleted if and when their surrounding components are no longer available.

---

getOtherSide

```
public AbstractComponent getOtherSide(AbstractComponent component)
```

Retrieves the component that is on the other side of this Connection. If the given component is the left, this returns the right and vice versa.

Returns:

component the AbstractComponent that is the other side of this Connection or null if there is no other side.

Throws:

`IllegalArgumentException` - if the given AbstractComponent is neither side of this Connection.

---

getOtherSideData

```
public ConnectionData getOtherSideData(AbstractComponent component)
```

Retrieves the ConnectionData for the other side of this Connection. If the given component is the left, this returns the right and vice versa.

Parameters:

`component` - the AbstractComponent that is the other side of this Connection.

Throws:

`IllegalArgumentException` - if the given AbstractComponent is neither side of this Connection.

---

getThisSideData

`public ConnectionData **getThisSideData**(AbstractComponent component)`

Retrieves the ConnectionData for this side of this Connection.

Parameters:

`component` - the AbstractComponent that is the desired side of this Connection.

Throws:

`IllegalArgumentException` - if the given AbstractComponent is neither side of this Connection.

---

getLeftComponent

`public AbstractComponent **getLeftComponent**()`

Retrieves the left component of this connection via ident. NOTE: Derivatives of Component should override this where possible to use an appropriate category for findComponentByIdent.

---

getLeftComponentID

`public int **getLeftComponentID**()`

Retrieves the ident of the left component of this connection.

---

setLeftComponentID

`public void **setLeftComponentID**(int ident)`

Sets the left component of this Connection by ident.

---

setLeftComponent

`public void **setLeftComponent**(AbstractComponent component)`

Sets the left component of this connection.

Parameters:

`component` - the AbstractComponent to use as this connection's left component

---

getRightComponent

`public AbstractComponent **getRightComponent**()`

Retrieves the right component of this connection via ident. NOTE: Derivatives of Component should override this where possible to use an appropriate category for findComponentByIdent.

getRightComponentID

```
public int getRightComponentID()
```

Retrieves the ident of the right component of this connection.

setRightComponentID

```
public void setRightComponentID(int ident)
```

Sets the right component of this Connection by ident.

setRightComponent

```
public void setRightComponent(AbstractComponent component)
```

Sets the right component of this connection.

Parameters:

`component` - the AbstractComponent to use as this connection's right component

getCategory

```
public Category getCategory()
```

Retrieves the most narrow category that this component is a member of. NOTE: This is the Category of the Connection class and has no relevance to the components it connects.

toString

```
public String toString()
```

getDocDescription

```
public String getDocDescription(AbstractComponent side,
        boolean includeComponent)
```

Returns an HTML formatted documentation description from of the given side of this connection.

Parameters:

`side` - the AbstractComponent on the side to describe from

Returns:

a String contianing the documentation description

clone

```
public Object clone()
```

Creates and returns a copy of this object.

disconnect

```
public void disconnect()
```

Disconnects this Connection between two AbstractComponent objects.

Removes this connection from both components and from the model. Fires all appropriate events to notify the Navigator, views and other listeners.

Handlers for user interactive disconnection should use userDisconnect() instead.

userDisconnect

```
public void userDisconnect()
```

Disconnects the connection from the Navigator. This is the method called when a connection is disconnected from the Navigator or from one of this Connection's DrawnConnect renderers.

reconnectIdentReferences

```
public void reconnectIdentReferences(boolean preserveUnresolved,
        boolean useDbId)
```

Resets this component's internal ident references to refer to appropriate components in the current AbstractModel. Intended for use after adding a component to a model and thus it's DB_ID will have been set to it's ident in the previous model. For ident references use find__ByDbId to find the new component, then store it's ident.

Note: If this component's DB_ID is 0 this method does nothing.

label

```
public String label()
```

Returns a String suitable for describing the component type on a dialog.

This default implementation returns the class name.

isVisual

```
public boolean isVisual()
```

Returns true if this connection should be visually represented. Visually represented means that this Connection will have DrawnConnection objects created for it if both sides of the Connection are present in a view.

isEqualTo

```
public boolean isEqualTo(Connection con)
```

This determines if the Connection passed in is the equivelant of this Connection. Used instead of GenericObject.equals(Object) to allow multiple equivalent connections between two components.

Returns:

true if both sides of each connection are the same.

createDrawnComponent

`public DrawnComponent createDrawnComponent()`

Creates a `DrawnConnection` to render this Connection.

Returns:
a DrawnConnection for this Connection.

---

getConnectionColor

`public java.awt.Color getConnectionColor()`

Returns the Color used to paint this Connection in a View.

---

getConnectionStroke

`public java.awt.Stroke getConnectionStroke()`

Returns the Stroke used to paint this connection in a View.

---

getCustomPopupItems

`public Vector getCustomPopupItems()`

Creates Custom Menu Items for any popup dialog involving this component

Returns:
a Vector of Action objects for this Connection.

com.cafean.client.analysis
Class ConnectionBean

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.AbstractComponent
         |
         +-com.cafean.client.analysis.Connection
            |
            +-com.cafean.client.analysis.ConnectionBean
```

All Implemented Interfaces:
IdentHolder, StateEditable, Cloneable, Checkable, ComponentElement, Cloneable

---

public abstract class **ConnectionBean**
extends Connection

A class representing a connection between two bean components in an AbstractModel.
See Also:
Connection

Fields inherited from class com.cafean.client.analysis.AbstractComponent
leftDiffComponent, leftDiffName, leftShortDiffName, rightDiffComponent, rightDiffName, rightShortDiffName

Fields inherited from class com.cafean.client.analysis.GenericObject
DATA_COMPLETE, DATA_ERROR, DATA_INCOMPLETE, DATA_WARNING

Fields inherited from interface javax.swing.undo.StateEditable
RCSID

Constructor Summary

public	ConnectionBean()

Method Summary

boolean	dumpBlockParams(java.io.PrintWriter dumpFile) A stub method implemented here to allow ConnectionBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.
Vector	getCustomPopupItems() Creates Custom Menu Items for any popup dialog involving this component.
void	popupDataDialog(java.awt.Window parent, boolean modal)

boolean	readBlockParams(com.apt.xdr.PibFile pibFile, int[] blockparm) A stub method implemented here to allow ConnectionBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.
boolean	writeBlockParams(com.apt.xdr.PibFile pibFile) A stub method implemented here to allow ConnectionBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

Methods inherited from class com.cafean.client.analysis.Connection

clone, createDrawnComponent, disconnect, getCategory, getConnectionColor, getConnectionStroke, getCustomPopupItems, getDocDescription, getLeftComponent, getLeftComponentID, getLeftConnectData, getOtherSide, getOtherSideData, getRightComponent, getRightComponentID, getRightConnectData, getThisSideData, isEqualTo, isIndependentComponent, isVisual, label, reconnectIdentReferences, setLeftComponent, setLeftComponentID, setRightComponent, setRightComponentID, toString, userDisconnect

Methods inherited from class com.cafean.client.analysis.AbstractComponent

addALDocRef, addComponentListener, addConnection, addMessage, addMessage, addMessage, addToModel, addToModel, canConnectTo, clearConnections, clone, complete, connectTo, connectTo, copy, createDrawnComponent, createSourceData, createTargetData, DBtypeCode, disconnect, disconnectFrom, fireComponentChanged, fireComponentChanged, fireComponentConnected, fireComponentDeleted, fireComponentDisconnected, getALDocDisplayName, getALDocRefs, getALDocRefs, getALDocShortNames, getALDocShortNames, getCatCCComparator, getCategory, getCCNumberComparator, getComponent, getComponentDependencies, getConnectionCount, getConnectionName, getConnections, getConnectionTypes, getCustomPopupActions, getCustomPopupItems, getGroupedConnections, getModel, getName, getNewCompIdent, getOrder, getOrderComparator, getOwner, getRealSize, getSharedComponents, getUniqueID, hasALDocRefs, includeInLoopcheck, isOkayForExport, isOkayForExport, label, popupDataDialog, rebuildConnections, reconnectIdentReferences, reconnectImage, removeALDocRef, removeComponentListener, removeFromModel, removeVerify, restoreALRefState, restoreState, setALDocRefs, setComponentNumber, setDeleted, setModel, setOrder, storeALRefState, toShortString, toString, updateVersion, userDelete, writeName

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable
restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder
reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ComponentElement
getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement
getModel

Methods inherited from interface com.cafean.client.analysis.Checkable
isOkayForExport, isOkayForExport

Constructors

ConnectionBean

```
public ConnectionBean()
```

Methods

dumpBlockParams

```
public boolean dumpBlockParams(java.io.PrintWriter dumpFile)
```

A stub method implemented here to allow ConnectionBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

writeBlockParams

```
public boolean writeBlockParams(com.apt.xdr.PibFile pibFile)
```

A stub method implemented here to allow ConnectionBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

readBlockParams

```
public boolean readBlockParams(com.apt.xdr.PibFile pibFile,
        int[] blockparm)
```

A stub method implemented here to allow ConnectionBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

---

getCustomPopupItems

```
public Vector getCustomPopupItems()
```

Creates Custom Menu Items for any popup dialog involving this component. The resulting Vector should contain on JMenu, JMenuItem and JSeparator instances for this component.

Returns:

a Vector containing JMenu's, JMenuItems, and JSeparators.

---

popupDataDialog

```
public void popupDataDialog(java.awt.Window parent,
        boolean modal)
```

Creates a new bean editing dialog for this object or resets and refreshes this object's current editing dialog.

com.cafean.client.analysis
Class ConnectionData

```
java.lang.Object
   |
   +-com.cafean.client.analysis.ConnectionData
```

All Implemented Interfaces:
Cloneable

Direct Known Subclasses:
SpecialConnectionData

---

public abstract class **ConnectionData**
extends Object
implements Cloneable

A simple representation of one side of a Connection.

This object is used by `DrawnComponent` and `DrawnConnection` objects to find an appropriate `ConnectingPt` object based on one side of a Connection. Instances of derivatives of this class represent each side.

The `equals(Object)` method is used by `DrawnComponent` and `DrawnConnection` objects to determine the appropriate `ConnectingPt` and must be implemented completely by each derivative class.

NOTE: All ConnectionData derivatives must be proper JavaBeans to be included as embedded connection points in `DrawnViewComponents`.

See Also:
`DrawnComponent.createConnectionPt(double, double, double, int, int, int, int, ConnectionData)`, `ConnectingPt.getConnectionData()`

Constructor Summary

public	`ConnectionData()` Creates a new instance of ConnectionData
public	`ConnectionData(int index)` Creates a new instance of Connection data with a specific connection point index specified.

Method Summary

Object	`clone()`
boolean	`equals(Object obj)` Determines the equivalence of this ConnectionData with the given object.
int	`getConnectionIndex()` Retrieves this ConnectionData's ConnectingPt index.
void	`setConnectionIndex(int connectionIndex)` Sets this ConnectionData's ConnectingPt index.

String	toString()

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

ConnectionData

public **ConnectionData**()

Creates a new instance of ConnectionData

ConnectionData

public **ConnectionData**(int index)

Creates a new instance of Connection data with a specific connection point index specified.

Parameters:

index - the connection point index of this connection.

Methods

equals

public boolean **equals**(Object obj)

Determines the equivalence of this ConnectionData with the given object.

Derivative classes must override this method and must include a comparison of connectionIndex as the default implementation returns reference equality.

See Also:

Object.equals(java.lang.Object)

clone

public Object **clone**()

getConnectionIndex

public int **getConnectionIndex**()

Retrieves this ConnectionData's ConnectingPt index.

WARNING: This value has a a Connection type specific meaning.

setConnectionIndex

```
public void setConnectionIndex(int connectionIndex)
```

Sets this ConnectionData's ConnectingPt index.

WARNING: This value has a a Connection type specific meaning.

---

toString

```
public String toString()
```

com.cafean.client.analysis
Class ConnectionList

```
java.lang.Object
   |
   +-com.cafean.client.analysis.ConnectionList
```

All Implemented Interfaces:
Cloneable

---

public class **ConnectionList**
extends Object
implements Cloneable

A list of connections to be used inside an AbstractComponent.

Constructor Summary

public	`ConnectionList(AbstractComponent component)` Creates a new instance of ConnectionList
public	`ConnectionList(AbstractComponent component, ConnectionList list)` Creates a new instance of connectionList, copying an existing list.

Method Summary

void	`addConnection(Connection connection)` Adds the given component to this ConnectionList.
void	`clear()` Removes all entries in this list.
Object	`clone()`
boolean	`contains(Connection con)` Returns true if the given component is referred to by this subsystem.
int	`getConnectionCount()` Returns the count of the component references in this Subsystem.
Connection[]	`getConnections()` Retrieves an array of the components referred to by this subsystem.
Iterator	`getIterator()` Retrieves an Iterator suitable for traversing this entire list.
void	`reconnectIdentReferences(AbstractModel model, boolean preserveUnresolved, boolean useDbId)` Resets this component's internal ident references to refer to appropriate components in the current AbstractModel.

void	removeConnection(Connection con) Removes the given component from this subsystem.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

ConnectionList

```
public ConnectionList(AbstractComponent component)
```

Creates a new instance of ConnectionList

ConnectionList

```
public ConnectionList(AbstractComponent component,
                      ConnectionList list)
```

Creates a new instance of connectionList, copying an existing list.

Methods

addConnection

```
public void addConnection(Connection connection)
```

Adds the given component to this ConnectionList.

Parameters:

connection - the Connection to add to this ConnectionList.

clear

```
public void clear()
```

Removes all entries in this list.

clone

```
public Object clone()
```

getConnections

```
public Connection[] getConnections()
```

Retrieves an array of the components referred to by this subsystem.

Returns:

an AbstractComponent[] containing the components referred to

---

reconnectIdentReferences

```
public void reconnectIdentReferences(AbstractModel model,
        boolean preserveUnresolved,
        boolean useDbId)
```

Resets this component's internal ident references to refer to appropriate components in the current AbstractModel. Intended for use after adding a component to a model and thus it's DB_ID will have been set to it's ident in the previous model. For ident references use find__ByDbId to find the new component, then store it's ident.

Note: If this component's DB_ID is 0 this method does nothing.

Parameters:

`model` - the AbstractModel to reconnectIdents inside of.

`preserveUnresolved` - if true, ident references that aren't resolvable will be left dangling, if false, they will be set to 0.

`useDbId` - if true, ident references will be reconnected via find_x_ByDB_ID; if false, ident references will be reconnected via find_x_ByIdent

---

contains

```
public boolean contains(Connection con)
```

Returns true if the given component is referred to by this subsystem.

---

removeConnection

```
public void removeConnection(Connection con)
```

Removes the given component from this subsystem.

---

getConnectionCount

```
public int getConnectionCount()
```

Returns the count of the component references in this Subsystem.

---

getIterator

```
public Iterator getIterator()
```

Retrieves an Iterator suitable for traversing this entire list.

Returns:

an Iterator suitable for iterating over the entirety of this list

com.cafean.client.analysis
Class DummyCompElement

```
java.lang.Object
   |
   +-com.cafean.client.analysis.DummyCompElement
```

All Implemented Interfaces:
ComponentElement

---

public class **DummyCompElement**
extends Object
implements ComponentElement

An empty component element for use in stub methods or in specialized editors for an entire component. This element should be used when an editor does not actually change the value directly and does not fire an event immediately.

Constructor Summary

public	DummyCompElement(ComponentElement owner) Creates a new instance of DummyCompElement

Method Summary

boolean	equals(Object obj)
AbstractComponent	getComponent()
AbstractModel	getModel()
ComponentElement	getOwner()

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface com.cafean.client.analysis.ComponentElement

getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement

getModel

Constructors

DummyCompElement

public **DummyCompElement**(ComponentElement owner)

Creates a new instance of DummyCompElement

Methods

equals

public boolean **equals**(Object obj)

---

getOwner

public ComponentElement **getOwner**()

---

getComponent

public AbstractComponent **getComponent**()

---

getModel

public AbstractModel **getModel**()

com.cafean.client.analysis
Class ElementBean

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.ElementBean
```

All Implemented Interfaces:
IdentHolder, StateEditable, Cloneable

---

public abstract class **ElementBean**
extends GenericObject

The base class for ModelEditor `Elements` that are full fledged beans.
See Also:
`AbstractModel.addElement(GenericObject)`

Fields inherited from class `com.cafean.client.analysis.GenericObject`
DATA_COMPLETE, DATA_ERROR, DATA_INCOMPLETE, DATA_WARNING

Fields inherited from interface `javax.swing.undo.StateEditable`
RCSID

Constructor Summary

public	`ElementBean()` Creates a new ElementBean with a DATA_INCOMPLETE data state and named unnamed
public	`ElementBean(int componentNumber)` Creates a new ElementBean with a DATA_INCOMPLETE data state and named unnamed

Method Summary

boolean	`dumpBlockParams(java.io.PrintWriter dumpFile)` A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.
boolean	`readBlock(com.apt.xdr.PibFile pibFile)` A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.
boolean	`readBlockParams(com.apt.xdr.PibFile pibFile)` A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.
boolean	`readBlockParams(com.apt.xdr.PibFile pibFile, int[] blockparm)` A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

void	restoreState(String prefix, Hashtable state) Restore the state of the bean from an earlier edit.
void	storeState(String prefix, Hashtable state) Store the state of the bean to permit undo.
boolean	writeBlockParams(com.apt.xdr.PibFile pibFile) A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder

reconnectIdentReferences

Constructors

ElementBean

public **ElementBean**()

Creates a new ElementBean with a DATA_INCOMPLETE data state and named unnamed

ElementBean

public **ElementBean**(int componentNumber)

Creates a new ElementBean with a DATA_INCOMPLETE data state and named unnamed

Methods

dumpBlockParams

```
public boolean dumpBlockParams(java.io.PrintWriter dumpFile)
```

A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

---

readBlock

```
public boolean readBlock(com.apt.xdr.PibFile pibFile)
```

A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

---

writeBlockParams

```
public boolean writeBlockParams(com.apt.xdr.PibFile pibFile)
```

A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

---

readBlockParams

```
public boolean readBlockParams(com.apt.xdr.PibFile pibFile)
```

A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

---

readBlockParams

```
public boolean readBlockParams(com.apt.xdr.PibFile pibFile,
        int[] blockparm)
```

A stub method implemented here to allow ElementBean objects to be PibBlocks, stored and loaded directly to a PIB generated file format.

This method simply returns true.

---

storeState

```
public void storeState(String prefix,
        Hashtable state)
```

Store the state of the bean to permit undo.

Parameters:

`state` - a Hashtable containing modified parameters.
`prefix` - a String containing the prefix for hash entries.

---

restoreState

```
public void restoreState(String prefix,
         Hashtable state)
```

Restore the state of the bean from an earlier edit.

Parameters:

`state` - a Hashtable containing modified parameters.
`prefix` - a String containing the prefix for hash entries.

com.cafean.client.analysis
Class ElementList

```
java.lang.Object
   |
   +-com.cafean.client.analysis.ElementList
```

All Implemented Interfaces:
Cloneable

Direct Known Subclasses:
ComponentList

---

public class **ElementList**
extends Object
implements Cloneable

A storage class for GenericObjects.

ElementList requires that all it's contained elements have unique ident numbers.

Elements are optionally stored and retrieved in sorted order via binary searches for efficiency.

Constructor Summary

public	`ElementList()` Creates a new instance of ElementList with a category of `Components` that starts in a sorted state.
public	`ElementList(boolean sorted)` Creates a new instance of ElementList with a category of `Components`.
public	`ElementList(String category, boolean sorted)` Creates a new instance of ElementList with the given category name and the given initial sorted state.

Method Summary

void	`add(GenericObject object)` Adds the given object to this list.
void	`clear()` Removes all elements from this list.
void	`clearDbIds()` Resets the DB_ID's of all contained elements.
Object	`clone()` Note: Only the contained references to GenericObject's are copied.

boolean	contains(GenericObject element) Returns true if this list contains an object with the same ident as the given object.
GenericObject	findByDB_ID(int dbid) Retrieves the GenericObject with the given DB_ID.
GenericObject	findByIdent(int identNumber) Retrieves the GenericObject with the given ident.
GenericObject	findByIdent(int identNumber, boolean includeDeleted) Retrieves the GenericObject with the given ident.
GenericObject	get(int index) Retrieves the element at the given index.
String	getCategory() Retrieves the name of the category of the elements contained by thisElementList
int	indexOf(GenericObject element) Returns the index of the given element in this list or -1.
boolean	isSorted() Returns true if this list has been sorted and will continue to add objects in a sorted order.
Iterator	iterator() Retutrns an Iterator object for iterating through this list.
void	reconnectIdentReferences(boolean preserveUnresolved, boolean useDbId) Reconnects ident references for this list's contained elements.
boolean	remove(GenericObject element) Removes the given object from this list.
void	removeDeleted() Removes all elements contained in the list that are marked deleted as indicated by GenericObject.isDeleted()
void	setCategory(String category) Sets the name of the category of the elements contained by thisElementList
int	size() Retrieves the size of this ElementList.
void	sort() Sorts this list and indicates that any objects added in the future should be added into thier sorted order.
Object[]	toArray() Returns an array containing all of the elements in this list in the correct order.
Object[]	toArray(Object[] a) Returns an array containing all of the elements in this list in the correct order; the runtime type of the returned array is that of the specified array.
String	toString() Returns a String representation of this ElementList.

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

ElementList

`public` **`ElementList`**`()`

Creates a new instance of ElementList with a category of `Components` that starts in a sorted state.

See Also:
`sort()`

ElementList

`public` **`ElementList`**`(boolean sorted)`

Creates a new instance of ElementList with a category of `Components`.

Parameters:
`sorted` - if true all additions will be inserted in thier sorted positions.

See Also:
`sort()`

ElementList

```
public ElementList(String category,
                   boolean sorted)
```

Creates a new instance of ElementList with the given category name and the given initial sorted state.

Parameters:
`category` - a String containing the category name to use for this element list.
`sorted` - if true all additions will be inserted in thier sorted positions.

See Also:
`sort()`

Methods

findByIdent

`public GenericObject` **`findByIdent`**`(int identNumber)`

Retrieves the GenericObject with the given ident. If identNumber is 0, null is returned.

Parameters:
`identNumber` - the ident of the required object.

Returns:
a GenericObject object corresponding to the search parameters

---

findByIdent

```
public GenericObject findByIdent(int identNumber,
        boolean includeDeleted)
```

Retrieves the GenericObject with the given ident. If identNumber is 0, null is returned.

Parameters:
`identNumber` - the ident of the required object.
`includeDeleted` - if true; objects set deleted will still be resturned.

Returns:
a GenericObject object corresponding to the search parameters

---

findByDB_ID

```
public GenericObject findByDB_ID(int dbid)
```

Retrieves the GenericObject with the given DB_ID. if the DB_ID is 0, null is returned.

Parameters:
`dbid` - the DB_ID of the desired object.

Returns:
a GenericObject object corresponding to the search parameters

---

reconnectIdentReferences

```
public void reconnectIdentReferences(boolean preserveUnresolved,
        boolean useDbId)
```

Reconnects ident references for this list's contained elements.

Parameters:
`preserveUnresolved` - if true, ident references that aren't resolvable will be left dangling, if false, they will be set to 0.
`useDbId` - if true, ident references will be reconnected via find_x_ByDB_ID; if false, ident references will be reconnected via find_x_ByIdent

See Also:
`GenericObject.clearDbIds()`

---

clearDbIds

```
public void clearDbIds()
```

Resets the DB_ID's of all contained elements.

See Also:
`GenericObject.clearDbIds()`

---

removeDeleted

`public void removeDeleted()`

Removes all elements contained in the list that are marked deleted as indicated by `GenericObject.isDeleted()`

---

get

`public GenericObject get(int index)`

Retrieves the element at the given index.

Parameters:
`index` - the index of the desired element

Returns:
the AbstractComponent of the desired element

---

size

`public int size()`

Retrieves the size of this ElementList.

---

add

`public void add(GenericObject object)`

Adds the given object to this list. If isSorted returns true the object will be added into sorted order and adding duplicates will cause an IllegalArgumentException to be thrown.

Parameters:
`object` - the GenericObject to add to this list.

Throws:
`IllegalArgumentException` - if the given object is already in this list.

---

remove

`public boolean remove(GenericObject element)`

Removes the given object from this list.

Parameters:
`element` - the GenericObject to remove

Returns:
true if the given element was actually removed.

---

indexOf

`public int indexOf(GenericObject element)`

Returns the index of the given element in this list or -1.

Parameters:

`element` - the GenericObject to find the index of

---

contains

```
public boolean contains(GenericObject element)
```

Returns true if this list contains an object with the same ident as the given object.

---

clear

```
public void clear()
```

Removes all elements from this list.

---

isSorted

```
public boolean isSorted()
```

Returns true if this list has been sorted and will continue to add objects in a sorted order.

---

sort

```
public void sort()
```

Sorts this list and indicates that any objects added in the future should be added into thier sorted order.

---

toArray

```
public Object[] toArray()
```

Returns an array containing all of the elements in this list in the correct order.

Returns:

an array containing all of the elements in this list in the correct order.

---

toArray

```
public Object[] toArray(Object[] a)
```

Returns an array containing all of the elements in this list in the correct order; the runtime type of the returned array is that of the specified array. If the list fits in the specified array, it is returned therein. Otherwise, a new array is allocated with the runtime type of the specified array and the size of this list.

If the list fits in the specified array with room to spare (i.e., the array has more elements than the list), the element in the array immediately following the end of the collection is set to null. This is useful in determining the length of the list *only* if the caller knows that the list does not contain any null elements.

Parameters:

a - the array into which the elements of the list are to be stored, if it is big enough; otherwise, a new array of the same runtime type is allocated for this purpose.

Returns:

an array containing the elements of the list.

Throws:

`ArrayStoreException` - if the runtime type of a is not a supertype of the runtime type of every element in this list.

---

iterator

```
public Iterator iterator()
```

Retutrns an Iterator object for iterating through this list.

---

getCategory

```
public String getCategory()
```

Retrieves the name of the category of the elements contained by thisElementList

Returns:

a String containing the category name of this ElementList

---

setCategory

```
public void setCategory(String category)
```

Sets the name of the category of the elements contained by thisElementList

Parameters:

`category` - a String containing the name of this ElementList

---

toString

```
public String toString()
```

Returns a String representation of this ElementList.

---

clone

```
public Object clone()
```

Note: Only the contained references to GenericObject's are copied. The GenericObject's themselves are not cloned.

com.cafean.client.analysis
Class GenericObject

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
```

All Implemented Interfaces:
IdentHolder, StateEditable, Cloneable

Direct Known Subclasses:
ElementBean, AbstractModel, AbstractComponent

---

public abstract class **GenericObject**
extends Object
implements Cloneable, StateEditable, IdentHolder

A generic model object with a name, description data state etc. This class is the base for all objects are contained in an ElementList.

Field Summary

public static final	DATA_COMPLETE Indication that this object is complete and error free. Value: **2147483647**
public static final	DATA_ERROR Indication that this object has a fatal error that requires attention. Value: **2**
public static final	DATA_INCOMPLETE Indication that this object is missing required data. Value: **0**
public static final	DATA_WARNING Indication that this object has a non-fatal error that requires attention. Value: **1**

Fields inherited from interface javax.swing.undo.StateEditable

RCSID

Constructor Summary

public	GenericObject() Creates a new GenericObject with a DATA_INCOMPLETE data state and named unnamed
public	GenericObject(int componentNumber) Creates a new GenericObject with a DATA_INCOMPLETE data state and named unnamed

Method Summary	
void	addComment(String comment) Appends the given string as to the list of comments in this object
void	addMultipleComments(Vector comments) Add the given Vector of Comments to this object.
boolean	checkRealArrayList(ArrayList table, String desc) Performs a check of the given ArrayList<Real> by iterating through each row and checking for Unknown values.
boolean	checkRealArrayTable(ArrayList table, String desc) Performs a check of the given ArrayList<Real[]> by iterating through each row and checking for Unknown values.
void	clearDbIds() Resets all the DB_ID's associated with this base object to 0
Object	clone() Creates a deep copy of this object.
void	closeAllViews() Closes all open views of this object.
int	compareTo(Object o) Compares this GenericObject with the given object based on ident for a natural ordering.
void	copyFrom(GenericObject o) Copy the attributes from a source object to this instance.
void	createDataPages(GenericObject original) Creates data pages for this object's Component View based on the given original object.
static void	debug(String message)
void	deleteAllComments() Removes all Comments from this object.
void	deleteComment(int num) Removes the given Comment from this object's comment list.
boolean	equals(Object obj) Returns true if the given object is this object, determined by reference equivalence.
static void	fixme(String message) Prints the given message to stderr in FIXME format.
String	getCCnumber() Retrieves this objects's CC number.
String	getComment(int num) Retrieves the comment at the given index.

String[]	getComments() Retrieves all the comments inside this object as Strings
String	getComments(int index) Retrieves the comment at the given index.
static String	getComponentCCNumber(GenericObject component) Returns the CC number of the given component or "0" if null.
int	getComponentNumber() Retrieves this object's component number.
int	getDataState() Retrieves this object's data state.
int	getDB_ID() Retrieves this object's DB_ID
String	getDescription() Retrieves the description of this object.
int	getIdent() Retrieves the ident of this object, NOT the DB_ID, and not the CC/Code Number.
int	getMajorCreationVersion() Retrieves the major revision number that this object was created with.
int	getMajorVersion() Retrieves this objects current major revision number.
int	getMinorCreationVersion() Retrieves ths minor revision number that this object was created with.
int	getMinorVersion() Retrieves this object's current minor revison number.
String	getName() Retrieves this object's name
static int	getNewCompIdent(AbstractModel model, int dbid, boolean preserveUnresolved, boolean useDbId) Retrieves the ident of the component with the given DB_ID in this component's model.
int	getNumComments() Retrieves the count of Comment objects stored in this object.
boolean	isDeleted() Returns true if this object has been set deleted.
void	popupDataDialog() Creates a new ComponentView for this object or resets and refreshes this object's current ComponentView.
void	popupDataDialog(java.awt.Window parent, boolean modal) Creates a new ComponentView for this object or resets and refreshes this object's current ComponentView.

boolean	rangeCheck(double i, double min, double max, String msg) Checks whether the given value i is within the range specified by the min and max.
boolean	rangeCheck(double i, double min, double max, String msg, boolean printMsg) Checks whether the given value i is within the range specified by the min and max.
boolean	rangeCheck(float i, float min, float max, String msg) Checks whether the given value i is within the range specified by the min and max.
boolean	rangeCheck(float i, float min, float max, String msg, boolean printMsg) Checks whether the given value i is within the range specified by the min and max.
boolean	rangeCheck(int i, int min, int max, String msg) Checks whether the given value i is within the range specified by the min and max.
boolean	rangeCheck(Int i, int min, int max, String msg) Checks whether the given value i is within the range specified by the min and max.
boolean	rangeCheck(int i, int min, int max, String msg, boolean printMsg) Checks whether the given value i is within the range specified by the min and max.
boolean	rangeCheck(Int i, int min, int max, String msg, boolean printMsg) Checks whether the given value i is within the range specified by the min and max.
boolean	rangeCheck(Real i, double min, double max, String msg) Checks whether the given value i is within the range specified by the min and max.
boolean	rangeCheck(Real r, double min, double max, String msg, boolean printMsg) Checks whether the given value i is within the range specified by the min and max.
void	reconnectIdentReferences(boolean preserveUnresolved, boolean useDbId) Resets this component's internal ident references to refer to appropriate components in the current AbstractModel.
void	restoreState(Hashtable state) Restore the state of the bean from an earlier edit by using copyFrom on the previously stored clone.
void	restoreState(String prefix, Hashtable state) Restore the state of the bean from an earlier edit.
void	setComments(int index, String value) Sets the comment at the given index.
void	setComments(String[] comments) Sets all of the comments inside this object as Strings
void	setComponentNumber(int i) Setter for the component number.
void	setCreationVersion(int major, int minor) Sets this object's creation version to the given numbers.
void	setDataState(int ds) Sets this object's data state.

void	setDB_ID(int dbid) Sets this object's DB_ID
void	setDeleted(boolean del) Sets this object deleted and thus inoperative.
void	setDescription(String desc) Sets this object's description to the given desc.
void	setIdent(int uid) Sets the unique ident of this object.
void	setMajorCreationVersion(int version) Sets this object's creation version to the given version.
void	setMajorVersion(int major_tag) Sets this objects current major revision number.
void	setMinorCreationVersion(int version) Sets this object's creation version to the given version.
void	setMinorVersion(int minor_tag) Sets this objects current minor revision number.
void	setName(String name) Sets this object's name to the given name after trimming it.
String	showComment(int num) Retrieves the text in the comment at the given index.
void	storeState(Hashtable state) Stores the state of the object to permit undo by cloning itself and storing the clone.
void	storeState(String prefix, Hashtable state) Store the state of the bean to permit undo.
static void	trace(String message) Prints the given message to stderr in TRACE format.
void	updateVersion() Updates this object's current version to be greater than or equal to it's model's version.
void	validate() Determines this object's general data state by examining it's internal data.
static void	writeArrayLoadValue(java.io.PrintWriter out, String header, Dimless[] values) writes out a value in TRACE 'LOAD Format' for each element in the given array.
static void	writeArrayLoadValue(java.io.PrintWriter out, String header, Dimless[] values, int columns) writes out a value in TRACE 'LOAD Format' for each element in the given array.
static void	writeArrayLoadValue(java.io.PrintWriter out, String header, int[] values) writes out a value in TRACE 'LOAD Format' for each element in the given array.

static void	writeArrayLoadValue(java.io.PrintWriter out, String header, int[] values, int columns) writes out a value in TRACE 'LOAD Format' for each element in the given array.
static void	writeArrayLoadValue(java.io.PrintWriter out, String header, Object[] values) writes out a value in TRACE 'LOAD Format' for each element in the given array.
static void	writeArrayLoadValue(java.io.PrintWriter out, String header, Object[] values, int columns) writes out a value in TRACE 'LOAD Format' for each element in the given array.
static void	writeArrayLoadValue(java.io.PrintWriter out, String header, Real[] values) writes out a value in TRACE 'LOAD Format' for each element in the given array.
static void	writeArrayLoadValue(java.io.PrintWriter out, String header, Real[] values, int columns) writes out a value in TRACE 'LOAD Format' for each element in the given array.
static void	writeMuxLoadArray(java.io.PrintWriter out, String header, Object[] xValues, Object[] yValues) writes the given demultiplexed x,y pairs to the given writer in multiplexed TRAC LOAD format.
static void	writeMuxLoadArray(java.io.PrintWriter out, String header, Real[] xValues, Real[] yValues) writes the given demultiplexed x,y pairs to the given writer in multiplexed TRAC LOAD format.
static void	writeSP(java.io.PrintWriter out, int value, int width) Write an integer to a PrintWriter as a fixed width string, padding the left with spaces.
static void	writeSP(java.io.PrintWriter out, String text, int width) Write a fixed width string to a PrintWriter, padding the left with spaces.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder

reconnectIdentReferences

Fields

DATA_INCOMPLETE

public static final int **DATA_INCOMPLETE**

Indication that this object is missing required data.
Constant value: **0**

DATA_WARNING

```
public static final int DATA_WARNING
```

Indication that this object has a non-fatal error that requires attention.
Constant value: **1**

DATA_ERROR

```
public static final int DATA_ERROR
```

Indication that this object has a fatal error that requires attention.
Constant value: **2**

DATA_COMPLETE

```
public static final int DATA_COMPLETE
```

Indication that this object is complete and error free.
Constant value: **2147483647**

Constructors

GenericObject

```
public GenericObject()
```

Creates a new GenericObject with a DATA_INCOMPLETE data state and named unnamed

---

GenericObject

```
public GenericObject(int componentNumber)
```

Creates a new GenericObject with a DATA_INCOMPLETE data state and named unnamed

Methods

restoreState

```
public void restoreState(String prefix,
        Hashtable state)
```

Restore the state of the bean from an earlier edit.

Parameters:

`state` - a Hashtable containing modified parameters.
`prefix` - a String containing the prefix for hash entries.

---

restoreState

```
public void restoreState(Hashtable state)
```

Restore the state of the bean from an earlier edit by using copyFrom on the previously stored clone.

Parameters:

`state` - A hash table containing modified parameters.

---

storeState

```
public void storeState(String prefix,
         Hashtable state)
```

Store the state of the bean to permit undo.

Parameters:

`state` - a Hashtable containing modified parameters.
`prefix` - a String containing the prefix for hash entries.

---

storeState

```
public void storeState(Hashtable state)
```

Stores the state of the object to permit undo by cloning itself and storing the clone. NOTE: If the component storing its state needs a deep copy that it's clone() method does not provide, it must override storeState to find that functionality elsewhere.

Parameters:

`state` - A hash table containing modified parameters.

---

clone

```
public Object clone()
```

Creates a deep copy of this object.

---

copyFrom

```
public void copyFrom(GenericObject o)
```

Copy the attributes from a source object to this instance.

This is used for copying data from an edited working copy into the original object. Most notably, it is used to enable undo/redo for the legacy beanless ModelEditor architecture as well as for importing new altered data from restart decks for some plugins.

This should call copyFrom on children that support it.

Note: Never call copyFrom from clone or copy!

Parameters:

`o` - the GenericObject source object.

---

setName

```
public void setName(String name)
```

Sets this object's name to the given name after trimming it.

Parameters:

`name` - a String containging the desired name.

getName

`public String getName()`

Retrieves this object's name

Returns:
a String containing this GenericObject's name.

addComment

`public void addComment(String comment)`

Appends the given string as to the list of comments in this object

addMultipleComments

`public void addMultipleComments(Vector comments)`

Add the given Vector of Comments to this object.

Parameters:
`comments` - the Vector of Comment objects to add.

setDescription

`public void setDescription(String desc)`

Sets this object's description to the given desc.

Parameters:
`desc` - a String containing this object's new description.

getDescription

`public String getDescription()`

Retrieves the description of this object.

Returns:
a String containing this object's description.

getNumComments

`public int getNumComments()`

Retrieves the count of Comment objects stored in this object.

deleteComment

`public void deleteComment(int num)`

Removes the given Comment from this object's comment list.

deleteAllComments

```
public void deleteAllComments()
```

Removes all Comments from this object.

showComment

```
public String showComment(int num)
```

Retrieves the text in the comment at the given index.

getComment

```
public String getComment(int num)
```

Retrieves the comment at the given index.

getComments

```
public String[] getComments()
```

Retrieves all the comments inside this object as Strings

getComments

```
public String getComments(int index)
```

Retrieves the comment at the given index.

setComments

```
public void setComments(int index,
        String value)
```

Sets the comment at the given index.

setComments

```
public void setComments(String[] comments)
```

Sets all of the comments inside this object as Strings

getIdent

```
public int getIdent()
```

Retrieves the ident of this object, NOT the DB_ID, and not the CC/Code Number.

For internal use only.

Do not show this to users.

Set by the model when added.

Returns:
the unique id of this object.

---

setIdent

```
public void setIdent(int uid)
```

Sets the unique ident of this object. This should only be used by the AbstractModel or related structures when adding this object to a model.

---

isDeleted

```
public boolean isDeleted()
```

Returns true if this object has been set deleted.

See Also:
`setDeleted(boolean)`

---

setDeleted

```
public void setDeleted(boolean del)
```

Sets this object `deleted` and thus inoperative. If this object has any children, it will propigate this deleted status to it's children via this method.

---

reconnectIdentReferences

```
public void reconnectIdentReferences(boolean preserveUnresolved,
        boolean useDbId)
```

Resets this component's internal ident references to refer to appropriate components in the current AbstractModel. Intended for use after adding a component to a model and thus it's DB_ID will have been set to it's ident in the previous model. For ident references use find__ByDbId to find the new component, then store it's ident.

Note: If this component's DB_ID is 0 this method does nothing.

Parameters:
`preserveUnresolved` - if true, ident references that aren't resolvable will be left dangling, if false, they will be set to 0.
`useDbId` - if true, ident references will be reconnected via find_x_ByDB_ID; if false, ident references will be reconnected via find_x_ByIdent

---

clearDbIds

```
public void clearDbIds()
```

Resets all the DB_ID's associated with this base object to 0

---

getDB_ID

```
public int getDB_ID()
```

Retrieves this object's DB_ID

Returns:

the DB_ID

---

setDB_ID

`public void setDB_ID(int dbid)`

Sets this object's DB_ID

Parameters:
`dbid` - the new DB_ID

---

compareTo

`public int compareTo(Object o)`

Compares this GenericObject with the given object based on ident for a natural ordering.

---

validate

`public void validate()`

Determines this object's general data state by examining it's internal data. The data state is used to color code objects so the user can see the current state. The `validate` method will set the state attribute to one of these values.

Note that this state is not inclusive of the checks in `isOkayForExport` and is used only to valdiate data required for the visual representation.

Derivatives should call super.validate() before determining thier own data state. This implementation sets this object's data state to `DATA_COMPLETE` without any analysis.

---

getDataState

`public int getDataState()`

Retrieves this object's data state.

Returns:
a DATA_* enumerated value such as DATA_COMPLETE or DATA_INCOMPLETE.

See Also:
`validate()`

---

setDataState

`public void setDataState(int ds)`

Sets this object's data state.

See Also:
`getDataState()`
`validate()`

---

getCCnumber

```
public String getCCnumber()
```

Retrieves this objects's CC number. The CC number is the object's display number in String form.

---

getComponentNumber

```
public int getComponentNumber()
```

Retrieves this object's component number.

---

getComponentCCNumber

```
public static String getComponentCCNumber(GenericObject component)
```

Returns the CC number of the given component or "0" if null.

---

setComponentNumber

```
public void setComponentNumber(int i)
```

Setter for the component number. Also known as the display number

Parameters:

i - the number to set this component number to

---

closeAllViews

```
public void closeAllViews()
```

Closes all open views of this object.

---

popupDataDialog

```
public void popupDataDialog()
```

Creates a new ComponentView for this object or resets and refreshes this object's current ComponentView.

---

popupDataDialog

```
public void popupDataDialog(java.awt.Window parent,
        boolean modal)
```

Creates a new ComponentView for this object or resets and refreshes this object's current ComponentView.

---

createDataPages

```
public void createDataPages(GenericObject original)
```

Creates data pages for this object's Component View based on the given original object.

---

equals

`public boolean equals(Object obj)`

Returns true if the given object is this object, determined by reference equivalence. Due to the current undo system, this method cannot be overriden to do a more useful comparison.

Parameters:
`obj` - {@inheritDoc}

See Also:
`Object.equals(java.lang.Object)`

---

updateVersion

`public void updateVersion()`

Updates this object's current version to be greater than or equal to it's model's version. If the major versions are the same, this object's minor version is incremented. NOTE: This implementation does nothing. Derivative classes must implement.

---

setCreationVersion

```
public void setCreationVersion(int major,
        int minor)
```

Sets this object's creation version to the given numbers. NOTE: Should only be used on load.

Parameters:
`major` - the new major creation version number
`minor` - the new minor creation version number

---

setMajorCreationVersion

`public void setMajorCreationVersion(int version)`

Sets this object's creation version to the given version. NOTE: Should only be used on load.

Parameters:
`version` - the new creation version number

---

setMinorCreationVersion

`public void setMinorCreationVersion(int version)`

Sets this object's creation version to the given version. NOTE: Should only be used on load.

Parameters:
`version` - the new creation version number

---

setMajorVersion

`public void setMajorVersion(int major_tag)`

Sets this objects current major revision number.

setMinorVersion

```
public void setMinorVersion(int minor_tag)
```

Sets this objects current minor revision number.

getMajorVersion

```
public int getMajorVersion()
```

Retrieves this objects current major revision number.

getMinorVersion

```
public int getMinorVersion()
```

Retrieves this object's current minor revison number.

getMajorCreationVersion

```
public int getMajorCreationVersion()
```

Retrieves the major revision number that this object was created with.

getMinorCreationVersion

```
public int getMinorCreationVersion()
```

Retrieves ths minor revision number that this object was created with.

checkRealArrayTable

```
public boolean checkRealArrayTable(ArrayList table,
          String desc)
```

Performs a check of the given ArrayList<Real[]> by iterating through each row and checking for Unknown values.

Parameters:

`table` - the ArrayList to check
`desc` - a String containing a description usable in the error message window

checkRealArrayList

```
public boolean checkRealArrayList(ArrayList table,
          String desc)
```

Performs a check of the given ArrayList<Real> by iterating through each row and checking for Unknown values.

Parameters:

`table` - the ArrayList to check
`desc` - a String containing a description usable in the error message window

Returns:

false if the ArrayList contains unknown values, false otherwise.

---

fixme

```
public static void fixme(String message)
```

Prints the given message to stderr in FIXME format. The fixme format is specified as: `FIXME['class name'.'method name'<'line number'>]: 'message'`

---

debug

```
public static void debug(String message)
```

---

trace

```
public static void trace(String message)
```

Prints the given message to stderr in TRACE format. The TRACE format is specified as: `TRACE['class name'.'method name'<'line number'>]: 'message'`

---

getNewCompIdent

```
public static int getNewCompIdent(AbstractModel model,
        int dbid,
        boolean preserveUnresolved,
        boolean useDbId)
```

Retrieves the ident of the component with the given DB_ID in this component's model. If dbid is 0, a 0 is returned.

Parameters:

`preserveUnresolved` - if true, and no component is found `dbid` will be returned. if false, 0 will be returned.

`useDbId` - if true, ident references will be reconnected via find_x_ByDB_ID; if false, ident references will be reconnected via find_x_ByIdent

---

rangeCheck

```
public boolean rangeCheck(int i,
        int min,
        int max,
        String msg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the value to be checked

`min` - the minimum allowed value

`max` - the maximum allowed value

`msg` - a String containing the description of the object being checked

Returns:

true if the value is within the range specified

---

rangeCheck

```
public boolean rangeCheck(int i,
          int min,
          int max,
          String msg,
          boolean printMsg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the value to be checked
`min` - the minimum allowed value
`max` - the maximum allowed value
`msg` - a String containing the description of the object being checked
`printMsg` - if true, an error mesage will be printed if the value is out of range

Returns:

true if the value is within the range specified

---

rangeCheck

```
public boolean rangeCheck(double i,
          double min,
          double max,
          String msg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the value to be checked
`min` - the minimum allowed value
`max` - the maximum allowed value
`msg` - a String containing the description of the object being checked
`printMsg` - if true, an error mesage will be printed if the value is out of range

Returns:

true if the value is within the range specified

---

rangeCheck

```
public boolean rangeCheck(double i,
          double min,
          double max,
          String msg,
          boolean printMsg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the value to be checked
`min` - the minimum allowed value
`max` - the maximum allowed value
`msg` - a String containing the description of the object being checked
`printMsg` - if true, an error mesage will be printed if the value is out of range

Returns:

true if the value is within the range specified

---

rangeCheck

```
public boolean rangeCheck(float i,
          float min,
          float max,
          String msg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the value to be checked
`min` - the minimum allowed value
`max` - the maximum allowed value
`msg` - a String containing the description of the object being checked

Returns:

true if the value is within the range specified

---

rangeCheck

```
public boolean rangeCheck(float i,
          float min,
          float max,
          String msg,
          boolean printMsg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the value to be checked
`min` - the minimum allowed value
`max` - the maximum allowed value
`msg` - a String containing the description of the object being checked
`printMsg` - if true, an error mesage will be printed if the value is out of range

Returns:

true if the value is within the range specified

---

rangeCheck

```
public boolean rangeCheck(Int i,
          int min,
          int max,
          String msg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the Int value to be checked
`min` - the minimum allowed value
`max` - the maximum allowed value
`msg` - a String containing the description of the object being checked

Returns:

true if the value is within the range specified

---

rangeCheck

```
public boolean rangeCheck(Int i,
          int min,
          int max,
          String msg,
          boolean printMsg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the Int value to be checked
`min` - the minimum allowed value
`max` - the maximum allowed value
`msg` - a String containing the description of the object being checked
`printMsg` - if true, an error mesage will be printed if the value is out of range

Returns:

true if the value is within the range specified

---

rangeCheck

```
public boolean rangeCheck(Real i,
          double min,
          double max,
          String msg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the Real value to be checked
`min` - the minimum allowed value
`max` - the maximum allowed value
`msg` - a String containing the description of the object being checked

Returns:

true if the value is within the range specified

---

rangeCheck

```
public boolean rangeCheck(Real r,
          double min,
          double max,
          String msg,
          boolean printMsg)
```

Checks whether the given value i is within the range specified by the min and max.

Parameters:

`i` - the value to be checked
`min` - the minimum allowed value
`max` - the maximum allowed value
`msg` - a String containing the description of the object being checked

printMsg - if true, an error mesage will be printed if the value is out of range

Returns:

true if the value is within the range specified

---

writeSP

```
public static void writeSP(java.io.PrintWriter out,
          String text,
          int width)
```

Write a fixed width string to a PrintWriter, padding the left with spaces. If the given text is longer than width, it will be truncated.

Parameters:

out - the java.io.PrintWriter to write the padded String to.

text - the String to pad and write out to out.

width - the desired width to pad text to.

---

writeSP

```
public static void writeSP(java.io.PrintWriter out,
          int value,
          int width)
```

Write an integer to a PrintWriter as a fixed width string, padding the left with spaces. If the given text is longer than width, it will be truncated.

Parameters:

out - the java.io.PrintWriter to write the padded String to.

value - the int to write to the given writer

width - the desired width to pad value to.

---

writeMuxLoadArray

```
public static void writeMuxLoadArray(java.io.PrintWriter out,
          String header,
          Object[] xValues,
          Object[] yValues)
```

writes the given demultiplexed x,y pairs to the given writer in multiplexed TRAC LOAD format.

Parameters:

out - the PrintWriter to write the values to.

header - the comment/header to begin each line with; cell length uses "* dx * ";

xValues - an Object[] containing the X values to be written.

yValues - an Object[] containing the Y values to be written.

---

writeMuxLoadArray

```
public static void writeMuxLoadArray(java.io.PrintWriter out,
          String header,
          Real[] xValues,
          Real[] yValues)
```

writes the given demultiplexed x,y pairs to the given writer in multiplexed TRAC LOAD format.

Parameters:

`out` - the PrintWriter to write the values to.
`header` - the comment/header to begin each line with; cell length uses "* dx * ";
`xValues` - an Object[] containing the X values to be written.
`yValues` - an Object[] containing the Y values to be written.

---

writeArrayLoadValue

```
public static void writeArrayLoadValue(java.io.PrintWriter out,
        String header,
        int[] values)
```

writes out a value in TRACE 'LOAD Format' for each element in the given array.

Parameters:

`out` - the PrintWriter to write the values to.
`header` - the comment/header to begin each line with; cell length uses "* dx * ";
`values` - an int[] containing the values to be written.

---

writeArrayLoadValue

```
public static void writeArrayLoadValue(java.io.PrintWriter out,
        String header,
        int[] values,
        int columns)
```

writes out a value in TRACE 'LOAD Format' for each element in the given array.

Parameters:

`out` - the PrintWriter to write the values to.
`header` - the comment/header to begin each line with; cell length uses "* dx * ";
`values` - an int[] containing the values to be written.
`columns` - the number of values to output per line

---

writeArrayLoadValue

```
public static void writeArrayLoadValue(java.io.PrintWriter out,
        String header,
        Real[] values)
```

writes out a value in TRACE 'LOAD Format' for each element in the given array.

Parameters:

`out` - the PrintWriter to write the values to.
`header` - the comment/header to begin each line with; cell length uses "* dx * ";
`values` - an Object[] containing the values to be written.

---

writeArrayLoadValue

```
public static void writeArrayLoadValue(java.io.PrintWriter out,
        String header,
        Real[] values,
        int columns)
```

writes out a value in TRACE 'LOAD Format' for each element in the given array.

Parameters:

`out` - the PrintWriter to write the values to.
`header` - the comment/header to begin each line with; cell length uses "* dx * ";
`values` - an Object[] containing the values to be written.
`columns` - the number of values to output per line

---

writeArrayLoadValue

```
public static void writeArrayLoadValue(java.io.PrintWriter out,
        String header,
        Dimless[] values)
```

writes out a value in TRACE 'LOAD Format' for each element in the given array.

Parameters:

`out` - the PrintWriter to write the values to.
`header` - the comment/header to begin each line with; cell length uses "* dx * ";
`values` - an Dimless[] containing the values to be written.

---

writeArrayLoadValue

```
public static void writeArrayLoadValue(java.io.PrintWriter out,
        String header,
        Dimless[] values,
        int columns)
```

writes out a value in TRACE 'LOAD Format' for each element in the given array.

Parameters:

`out` - the PrintWriter to write the values to.
`header` - the comment/header to begin each line with; cell length uses "* dx * ";
`values` - an Dimless[] containing the values to be written.
`columns` - the number of values to output per line

---

writeArrayLoadValue

```
public static void writeArrayLoadValue(java.io.PrintWriter out,
        String header,
        Object[] values)
```

writes out a value in TRACE 'LOAD Format' for each element in the given array.

Parameters:

`out` - the PrintWriter to write the values to.
`header` - the comment/header to begin each line with; cell length uses "* dx * ";
`values` - an Object[] containing the values to be written.

---

writeArrayLoadValue

```
public static void writeArrayLoadValue(java.io.PrintWriter out,
        String header,
        Object[] values,
        int columns)
```

writes out a value in TRACE 'LOAD Format' for each element in the given array.

Parameters:

`out` - the PrintWriter to write the values to.
`header` - the comment/header to begin each line with; cell length uses "* dx * ";
`values` - an Object[] containing the values to be written.
`columns` - the number of values to output per line

com.cafean.client.analysis
Interface IdentHolder

All Known Implementing Classes:
GenericObject

---

public interface **IdentHolder**
extends

A placeholder interface indicating that this object contains ident references that must be reconnected during AbstractModel#reconnectIdentReferences

---

Method Summary

void	`reconnectIdentReferences(boolean preserveUnresolved, boolean useDbId)` Resets this component's internal ident references to refer to appropriate components in the current AbstractModel.

Methods

reconnectIdentReferences

```
public void reconnectIdentReferences(boolean preserveUnresolved,
        boolean useDbId)
```

Resets this component's internal ident references to refer to appropriate components in the current AbstractModel. Intended for use after adding a component to a model and thus it's DB_ID will have been set to it's ident in the previous model. For ident references use find__ByDbId to find the new component, then store it's ident.

Note: If this component's DB_ID is 0 this method does nothing.

Parameters:

`preserveUnresolved` - if true, ident references that aren't resolvable will be left dangling, if false, they will be set to 0.

`useDbId` - if true, ident references will be reconnected via find_x_ByDB_ID; if false, ident references will be reconnected via find_x_ByIdent

com.cafean.client.analysis
Interface ModelDependent

All Known Implementing Classes:
NamedIntEditor, NamelistBooleanEditor, ComponentSelectionEditor, NamelistIntEditor, NamelistDoubleEditor, RealArrayEditor, RealBeanEditor, RealTextField, RealEditor

---

public interface **ModelDependent**
extends

An interface describing an object that is dependent on an `AbstractModel` either directly or by being a part of an object that is.

This interface is similar to `ModelElement` in that it includes an accessor for the object's model but in this case the interface is intended for use by PropertyEditors that require a model reference to edit a given value.

Method Summary

`AbstractModel`	`getModel()` Retrieves the model that this object depends on.
`void`	`setModel(AbstractModel model)` Sets the model that this object depends on.

Methods

getModel

```
public AbstractModel getModel()
```

Retrieves the model that this object depends on. This method may call a parent's getModel() and so on rather than a direct reference.

Returns:
the AbstractModel that this object depends on

---

setModel

```
public void setModel(AbstractModel model)
```

Sets the model that this object depends on.

Parameters:
`model` - the AbstractModel that this object will now depend on

com.cafean.client.analysis
Interface ModelElement

All Subinterfaces:
ComponentElement

All Known Implementing Classes:
AbstractModel

---

public interface **ModelElement**
extends

An interface describing an object that is contained by an `AbstractModel` either directly or by being a part of an object that is.

Method Summary

`AbstractModel`	`getModel()` Retrieves the model that contains this ModelElement.

Methods

getModel

`public AbstractModel` **`getModel()`**

Retrieves the model that contains this ModelElement.

This method may call a parent's getModel() and so on rather than a direct reference.

Returns:
the AbstractModel that contains this ModelElement

com.cafean.client.analysis
Interface SharedComponent

public interface **SharedComponent**
extends

An interface describing an AbstractComponent that is shared among other components. Examples include shared geometry or materials objects.

Method Summary

boolean	`isEquivalent(SharedComponent component)` Returns true if this and the given shared component are equivalent.

Methods

isEquivalent

`public boolean` **`isEquivalent`**`(SharedComponent component)`

Returns true if this and the given shared component are equivalent. This differs from `equals` in that the given component is not necessarily required to have identical data to be equivalent.

Parameters:

`component` - the SharedComponent to test for equivalence

com.cafean.client.analysis
Class SpecialConnectionData

```
java.lang.Object
   |
   +-com.cafean.client.analysis.ConnectionData
      |
      +-com.cafean.client.analysis.SpecialConnectionData
```

All Implemented Interfaces:
Cloneable

---

public class **SpecialConnectionData**
extends ConnectionData

SpecialConnectionData is an extention of ConnectionData that tells the system that the user must supply information for the connection to proceed.

See Also:
AbstractComponent.connectTo(AbstractComponent, ConnectionData, ConnectionData),
AbstractComponent.createSourceData(ConnectionData),
AbstractComponent.createTargetData(ConnectionData)

Constructor Summary

public	SpecialConnectionData() Creates a new instance of SpecialConnectData
public	SpecialConnectionData(int index) Creates a new instance of SpecialConnectData with the given index.

Method Summary

Object	clone()
String	toString()

Methods inherited from class com.cafean.client.analysis.ConnectionData

clone, equals, getConnectionIndex, setConnectionIndex, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

---

Constructors

SpecialConnectionData

`public SpecialConnectionData()`

Creates a new instance of SpecialConnectData

SpecialConnectionData

`public SpecialConnectionData(int index)`

Creates a new instance of SpecialConnectData with the given index.

Parameters:

`index` - {@inheritDoc}

Methods

clone

`public Object clone()`

---

toString

`public String toString()`

com.cafean.client.analysis
Class ValidationOptions

```
java.lang.Object
   |
   +-com.cafean.client.analysis.ValidationOptions
```

All Implemented Interfaces:
StateEditable

---

public class **ValidationOptions**
extends Object
implements StateEditable

This class is used to store all of a model's validation options into MED. This should be a property of each model for a plugin. Model specific options are stored in parallel string arrays.

Fields inherited from interface javax.swing.undo.StateEditable
RCSID

Constructor Summary

public	ValidationOptions() Creates a new instance of ValidationOptions

Method Summary

void	addOption(String key, String option) Adds a new option to be stored.
boolean	dumpBlockParams(java.io.PrintWriter dumpFile) Simple dumpBlockParams added so that ValidationOptions can be subblocks.
String[]	getData() This gets the current data array from this options.
String[]	getKeys() This gets the current keys array from this options.
String	getOption(String key, String def) Attempts to find an option with the given key.
boolean	readBlock(com.apt.xdr.PibFile pibFile) Simple readBlock added so that ValidationOptions can be subblocks.
boolean	readBlockParams(com.apt.xdr.PibFile pibFile) Simple readBlockParams added so that ValidationOptions can be subblocks.
boolean	readBlockParams(com.apt.xdr.PibFile pibFile, int[] blockparm) Simple readBlockParams added so that ValidationOptions can be subblocks.

void	restoreState(Hashtable state)
void	restoreState(String prefix, Hashtable state) Loads the current state of this ValidationOptions from a hashtable.
void	setData(String[] data) This sets the current data array for this options.
void	setKeys(String[] keys) This sets the current keys array for this options.
void	setPrefix(String prefix) This method is called by ValidationTest in load and store option.
void	storeState(Hashtable state)
void	storeState(String prefix, Hashtable state) Stores the current state of this ValidationOptions to a hashtable.
boolean	writeBlockParams(com.apt.xdr.PibFile pibFile) Simple writeBlockParams added so that ValidationOptions can be subblocks.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Constructors

ValidationOptions

```
public ValidationOptions()
```

Creates a new instance of ValidationOptions

Methods

addOption

```
public void addOption(String key,
        String option)
```

Adds a new option to be stored. This option will be associated with the given key. If the key already exists, the data in option will overwrite what currently exists.

getOption

```
public String getOption(String key,
        String def)
```

Attempts to find an option with the given key. If no option is found the default is returned. The current prefix will be prepended to the key.

---

setPrefix

```
public void setPrefix(String prefix)
```

This method is called by ValidationTest in load and store option. This sets the current prefix. This prefix is used to prevent data from overwriting between ValidationTests. The prefix used is the name of the current ValidationTest, and is why the ValidationTests for a given plugin must have unique names.

---

getKeys

```
public String[] getKeys()
```

This gets the current keys array from this options. This should be used only for MED file saving code. Any changes to this array will result in lost data.

---

setKeys

```
public void setKeys(String[] keys)
```

This sets the current keys array for this options. This should be used only for MED file loading code. Any changes to this array will result in lost data.

---

getData

```
public String[] getData()
```

This gets the current data array from this options. This should be used only for MED file saving code. Any changes to this array will result in lost data.

---

setData

```
public void setData(String[] data)
```

This sets the current data array for this options. This should be used only for MED file loading code. Any changes to this array will result in lost data.

---

readBlock

```
public boolean readBlock(com.apt.xdr.PibFile pibFile)
```

Simple readBlock added so that ValidationOptions can be subblocks.

---

dumpBlockParams

```
public boolean dumpBlockParams(java.io.PrintWriter dumpFile)
```

Simple dumpBlockParams added so that ValidationOptions can be subblocks.

writeBlockParams

```
public boolean writeBlockParams(com.apt.xdr.PibFile pibFile)
```

Simple writeBlockParams added so that ValidationOptions can be subblocks.

readBlockParams

```
public boolean readBlockParams(com.apt.xdr.PibFile pibFile,
        int[] blockparm)
```

Simple readBlockParams added so that ValidationOptions can be subblocks.

readBlockParams

```
public boolean readBlockParams(com.apt.xdr.PibFile pibFile)
```

Simple readBlockParams added so that ValidationOptions can be subblocks.

storeState

```
public void storeState(String prefix,
        Hashtable state)
```

Stores the current state of this ValidationOptions to a hashtable. This ensures that this object is state editable

restoreState

```
public void restoreState(String prefix,
        Hashtable state)
```

Loads the current state of this ValidationOptions from a hashtable. This ensures that this object is state editable

restoreState

```
public void restoreState(Hashtable state)
```

storeState

```
public void storeState(Hashtable state)
```

com.cafean.client.analysis
Class ValidationTest

```
java.lang.Object
   |
   +-com.cafean.client.analysis.ValidationTest
```

---

public abstract class **ValidationTest**
extends Object

This class represents a model-level test that is executed when the plugin verifies that a model's data is valid.

Field Summary

public static	VERBOSITY_HIGH Verbose output level.
public static	VERBOSITY_LOW Normal output message level.
public static	VERBOSITY_QUIET Few output messages.

Constructor Summary

public	ValidationTest()

Method Summary

boolean	canRepair() Determines if this validation test can be self correcting.
abstract String	getDisplayName() This returns a short String that is a human readable name for this validation test.
abstract String	getName() Gets the name of the specified Test.
abstract String	getShortDescription() This returns a long, detailed description of this validation test, its user options, and background information.
int	getVerbosity() Gets the verbosity level for this test.
boolean	isEnabled() Determines if this validation test is enabled for the current model.
void	loadOptions(ValidationOptions options) Loads the options for a validationTest from the given ValidationOptions object.

void	repair() If this validation test can self-correct, this is where it does so.
abstract boolean	runValidation(boolean printErrors) This method is the actual validation.
void	saveOptions(ValidationOptions options) Stores the options for a validationTest to the given ValidationOptions object.
void	setEnabled(boolean enabled) Sets this validation test enabled.
void	setVerbosity(int verbosity) Sets the verbosity level for this test.
String	toString() Returns a short string representing this test

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Fields

VERBOSITY_QUIET

public static int **VERBOSITY_QUIET**

Few output messages.

VERBOSITY_LOW

public static int **VERBOSITY_LOW**

Normal output message level.

VERBOSITY_HIGH

public static int **VERBOSITY_HIGH**

Verbose output level.

Constructors

ValidationTest

public **ValidationTest**()

Methods

getName

`public abstract String **getName**()`

Gets the name of the specified Test. The test names must be unique for a given plugin.

---

loadOptions

`public void **loadOptions**(ValidationOptions options)`

Loads the options for a validationTest from the given ValidationOptions object. Options are stored as a String with another String given as the "key" to that option. When this class is overwritten, the override for this method should always call super loadOptions. This will be called right before the validationTest executes.

Parameters:

`options` - the ValidationOptions from which data will be loaded.

---

saveOptions

`public void **saveOptions**(ValidationOptions options)`

Stores the options for a validationTest to the given ValidationOptions object. Options are stored as a String with another String given as the "key" to that option. When this class is overwritten, the override for this method should always call super saveOptions. This should be called after the options are edited by the user.

Parameters:

`options` - the ValidationOptions where data will be saved.

---

getDisplayName

`public abstract String **getDisplayName**()`

This returns a short String that is a human readable name for this validation test.

---

getShortDescription

`public abstract String **getShortDescription**()`

This returns a long, detailed description of this validation test, its user options, and background information.

---

runValidation

`public abstract boolean **runValidation**(boolean printErrors)`

This method is the actual validation. If this returns false, the model will be considered to have failed.

Parameters:

`printErrors` - a boolean flag that should enable writing error messages to the MainFrame.

Returns:

true if the validation ran and discovered no errors.

---

isEnabled

`public boolean **isEnabled**()`

Determines if this validation test is enabled for the current model. This should be an option stored in #saveOption and #loadOption.

---

setEnabled

`public void **setEnabled**(boolean enabled)`

Sets this validation test enabled.

---

getVerbosity

`public int **getVerbosity**()`

Gets the verbosity level for this test.

---

setVerbosity

`public void **setVerbosity**(int verbosity)`

Sets the verbosity level for this test. Acceptable values are:

1. VERBOSITY_NONE
2. VERBOSITY_LOW
3. VERBOSITY_HIGH

---

canRepair

`public boolean **canRepair**()`

Determines if this validation test can be self correcting. This is currently not supported. The default is false.

---

repair

`public void **repair**()`

If this validation test can self-correct, this is where it does so. Any user interaction should be specified here. This is currently not supported. The default is no operations.

---

toString

`public String **toString**()`

Returns a short string representing this test

com.cafean.client.analysis
Class ViewComponent

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.AbstractComponent
         |
         +-com.cafean.client.analysis.ViewComponent
```

All Implemented Interfaces:
PropertyController, IdentHolder, StateEditable, Cloneable, Checkable, ComponentElement, Cloneable

---

public class **ViewComponent**
extends AbstractComponent
implements Cloneable, ComponentElement, Checkable, Cloneable, StateEditable, IdentHolder, PropertyController

The ViewComponent is an `AbstractComponent` that represents a `drawing` inside the ModelEditor. This contains all of the information needed to create a new `DrawnView` from data stored in a local file. It also contains the picture that is associated with a given view, and gets displayed when it is `rendered` inside a `DrawnView`.

Field Summary

public static final	MAX_PAGE_HEIGHT Value: **85000**
public static final	MAX_PAGE_WIDTH Value: **110000**
public static final	PIXELS_PER_METER Value: **20**

Fields inherited from class `com.cafean.client.analysis.AbstractComponent`

leftDiffComponent, leftDiffName, leftShortDiffName, rightDiffComponent, rightDiffName, rightShortDiffName

Fields inherited from class `com.cafean.client.analysis.GenericObject`

DATA_COMPLETE, DATA_ERROR, DATA_INCOMPLETE, DATA_WARNING

Fields inherited from interface `javax.swing.undo.StateEditable`

RCSID

Fields inherited from interface `com.cafean.client.ui.beans.PropertyController`

ALL, COLOR_OPTIONAL, DISABLED, NONE, OPTIONAL, REQUIRED

Constructor Summary

public	`ViewComponent()` Creates a new instance of ViewComponent
public	`ViewComponent(AbstractModel model, int num)` Creates a new instance of ViewComponent with a new model and display number

Method Summary

void	`addAnnotation(Annotation comp)` Adds an `Annotation` to this ViewComponent, or to the `DrawnView` if the view has been opened.
void	`addComponent(AbstractComponent comp, boolean select)` Adds a new component to this view, either by adding it's pibblock if the view is not created, or by adding it directly to the view.
void	`addComponents(Iterator itr)` Uses the given iterator to add all of those components to the View, if it exists, or adds them to the PibBlock list if they don't.
void	`addComponents(Iterator itr, boolean select)` Uses the given iterator to add all of those components to the View, if it exists, or adds them to the PibBlock list if they don't.
void	`addPibBlock(com.apt.xdr.PibBlock block)` Adds a new Pibblock to this View Component.
void	`buildView()` This initializes a new DrawnView based on a View Component.
boolean	`canConnectTo(AbstractComponent target)` This method checks to see if it is allowable for this AbstractComponent to connect to the given target.
void	`clearViewSelection()` Clears the current selection on the View if it exists
Object	`clone()` this clone method copies all primitive data types.
void	`complete()` When a ViewComponent is created, it immediately opens.
AbstractComponent	`copy(AbstractModel sm)` the following copy method produces a deep clone of a composite base so that it can be put in a copy model clipboard.
void	`copyFrom(GenericObject o)` Copy the attributes from a clone to this instance.
DrawnComponent	`createDrawnComponent()` Returns the renderer for this abstract component. This will be extended by any abstract component that needs a renderer.

void	dockView() Builds this view and displays it as a docked window.
void	enableArrowKeyMovement() Enables arrow key movement of graphical elements in this view.
void	enableVisualGrouping() Enables visual grouping for this view.
void	fireComponentChanged() This calls the component changed function on all of the listeners currently listening to this abstract component
void	fireComponentDeleted() This calls the component deleted function on all of the listeners currently listening to this abstract component
byte[]	getAnnotationArray() Gets an byte array containing the XML encoded stream of annotations from this DrawnView.
static String	getAttributeGroup(String property) Retrieves the name of the Attribute Group for the given property name.
static String[]	getAttributeGroups() Retrieves the Attribute Group Names for this View Component.
int	getAttributeIndex(String propertyName)
static String[]	getAttributesForGroup(String groupName) Retrieves the Attribute Names for the given group.
java.awt.Color	getBackground() Gets the background color from the DrawnView, or from the local variable if the DrawnView has not been created.
java.awt.Dimension	getCanvasSize() Gets the size of the view's canvas or the locally stored canvas size, if the DrawnView has never been opened.
Category	getCategory()
String	getCCnumber()
int[]	getColorArray() This returns the current background in the form of an array of three integers.
Vector	getCustomPopupItems() Creates Custom Menu Items for any popup dialog involving this component
DrawnView	getDrawnView() A getter for the DrawnView owned by this ViewComponent.

byte[]	getEmbedConnArray() Gets an byte array containing the XML encoded stream of annotations from this DrawnView.
EmbeddedConnectionMap	getEmbeddedCons()
java.awt.Color	getGridLineColor() Getter for property lineColor.
int	getHorizGridSpacing() Getter for property horizGridSpacing.
int	getHorizSnapSpacing() Getter for property horizSnapSpacing.
ImageIcon	getImage() Gets the ImageIcon for rendering this ViewComponent.
byte[]	getImageData() Gets the icon image for the DrawnViewComponent that renders this ViewComponent.
ViewImgDataElem	getImageElement() A stub method used in ViewComponentBeanInfo to edit the image data byte[] contained in this ViewComponent.
java.awt.Point	getLocation() Gets the location of the DrawnView or the locally stored location, if the DrawnView has never been opened.
com.apt.xdr.PibBlock[ ]	getPibBlocks() Returns the array of compressed PibBlocks stored inside this view.
int	getPixelsPerMeter() Getter for property pixelsPerMeter.
boolean	getShowPoints() Returns whether or not the connection points are drawn on AbstractComponents inside this view.
java.awt.Dimension	getSize() Gets the size of the DrawnView or the locally stored size, if the DrawnView has never been opened.
float	getTransparency() Getter for property transparency.
String	getUniqueID()
int	getVertGridSpacing() Getter for property vertGridSpacing.
int	getVertSnapSpacing() Getter for property vertSnapSpacing.

java.awt.Point	getViewPosition() Getter for property viewPosition.
double	getWidthScaleFactor() Gets this view's width scale factor.
double	getZoomScale() Gets the current zoom factor from the view's zoom panel or the locally stored zoom factor, if the DrawnView has never been opened.
boolean	hasBlocks() Determines whether this view has anything inside it's blocks vector.
boolean	isArrowKeyMovementEnabled() Returns true if arrow key movement is enabled for this view.
boolean	isGridAbove() Returns true if this View's grid will be painted on top of its contained elements.
boolean	isLocked() Returns true if this view is locked.
boolean	isPropertyActive(String propertyName)
boolean	isPropertyEnabled(String propertyName)
boolean	isPropertyRequired(String propertyName)
boolean	isPropertyResizable(String propertyName)
boolean	isPropertyRestartEditable(String propertyName)
boolean	isRestartResizable(String propertyName)
boolean	isShowGrid() Getter for property showGrid.
boolean	isSnapToGrid() Getter for property snapToGrid.
boolean	isViewVisible() Returns true if the view is currently visible, or false if the view is not visible, or doesn't exist yet.
boolean	isVisualGroupingEnabled() Returns true if visual grouping is enabled for this view.
String	label()
void	layoutView() Runs the organize function on the view component if it exists.

void	load(ViewCompRec rec) Loads this view's data from the given ViewCompRec previously stored with store(PibFile)
static java.awt.Component	loadComponent(com.apt.xdr.PibBlock block, AbstractModel model)
void	popupDataDialog(java.awt.Window parent, boolean modal)
void	readAnnotationArray(byte[] byteArr, AbstractModel model) Retrieves all of the Annotations from an XML encoded byte array of annotations, and adds them to this ViewComponent.
void	readEmbedConnArray(byte[] byteArr) Retrieves all of the Annotations from an XML encoded byte array of annotations, and adds them to this ViewComponent.
void	removeFromModel(AbstractModel model)
boolean	removeVerify()
void	setBackground(java.awt.Color background) Sets the background color on the DrawnView if it has been created, and stores it inside the local variable.
void	setCanvasSize(java.awt.Dimension canvasSize) Sets the locally stored canvas size.
void	setColorArray(int[] array) Sets the current background from an array of three integers.
void	setEmbeddedCons(EmbeddedConnectionMap cons)
void	setGridAbove(boolean above) If set to true, this View's grid will be painted on top of its contained elements.
void	setGridLineColor(java.awt.Color lineColor) Setter for property lineColor.
void	setHorizGridSpacing(int horizGridSpacing) Setter for property horizGridSpacing.
void	setHorizSnapSpacing(int horizSnapSpacing) Setter for property horizSnapSpacing.
void	setImageData(byte[] imageData) Sets the icon image for the DrawnViewComponent that renders this ViewComponent.
void	setImageElement(ViewImgDataElem element) A stub method used in ViewComponentBeanInfo to edit the image data byte[] contained in this ViewComponent.
void	setLocation(java.awt.Point location) Sets the the locally stored location of the DrawnView.

void	setLocked(boolean locked) sets this view's locked state
void	setLockedConstrained(boolean locked) sets this view's locked state
void	setPibBlocks(com.apt.xdr.PibBlock[] array) Replaces the array of PibBlocks for this view.
void	setPixelsPerMeter(int pixelsPerMeter) Setter for property pixelsPerMeter.
void	setShowGrid(boolean showGrid) Setter for property showGrid.
void	setShowPoints(boolean showPoints) Turns on displaying the connection points on AbstractComponents inside this view.
void	setSize(java.awt.Dimension size) Sets the size of the DrawnView or the locally stored size, if the DrawnView has never been opened.
void	setSnapToGrid(boolean snapToGrid) Setter for property snapToGrid.
void	setStoredBackground(java.awt.Color background) This just sets the background color of the stored local variable.
void	setTransparency(float transparency) Setter for property transparency.
void	setVertGridSpacing(int vertGridSpacing) Setter for property vertGridSpacing.
void	setVertSnapSpacing(int vertSnapSpacing) Setter for property vertSnapSpacing.
void	setViewPosition(java.awt.Point viewPosition) Setter for property viewPosition.
void	setVisible(boolean value) Shows or hides this ViewComponent's DrawnView and it's accompanying dialog or panel.
void	setVisible(boolean value, ViewCompRec record) Shows or hides this ViewComponent's DrawnView and it's accompanying dialog or panel.
void	setWidthScaleFactor(double factor) Sets this view's width scale factor.
void	setZoomScale(double zoomScale) Sets the locally stored zoom scale.
boolean	store(com.apt.xdr.PibFile file) Writes this view and it's contents to the given PibFile.

com.apt.xdr.PibBlock	storeComponent(java.awt.Component c) Stores the given Component into a PibBlock for use in saving into a PIB formatted file.
String	toString() ViewComponents use their name as their to-string, as opposed to including "View CCnumber" before their name.
void	undockView() Builds this view and displays it as an undocked separate window.

Methods inherited from class com.cafean.client.analysis.AbstractComponent

addALDocRef, addComponentListener, addConnection, addMessage, addMessage, addMessage, addToModel, addToModel, canConnectTo, clearConnections, clone, complete, connectTo, connectTo, copy, createDrawnComponent, createSourceData, createTargetData, DBtypeCode, disconnect, disconnectFrom, fireComponentChanged, fireComponentChanged, fireComponentConnected, fireComponentDeleted, fireComponentDisconnected, getALDocDisplayName, getALDocRefs, getALDocRefs, getALDocShortNames, getALDocShortNames, getCatCCComparator, getCategory, getCCNumberComparator, getComponent, getComponentDependencies, getConnectionCount, getConnectionName, getConnections, getConnectionTypes, getCustomPopupActions, getCustomPopupItems, getGroupedConnections, getModel, getName, getNewCompIdent, getOrder, getOrderComparator, getOwner, getRealSize, getSharedComponents, getUniqueID, hasALDocRefs, includeInLoopcheck, isOkayForExport, isOkayForExport, label, popupDataDialog, rebuildConnections, reconnectIdentReferences, reconnectImage, removeALDocRef, removeComponentListener, removeFromModel, removeVerify, restoreALRefState, restoreState, setALDocRefs, setComponentNumber, setDeleted, setModel, setOrder, storeALRefState, toShortString, toString, updateVersion, userDelete, writeName

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder
reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ComponentElement
getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement
getModel

Methods inherited from interface com.cafean.client.analysis.Checkable
isOkayForExport, isOkayForExport

Methods inherited from interface com.cafean.client.ui.beans.PropertyController
getAttributeIndex, isPropertyActive, isPropertyEnabled, isPropertyRequired, isPropertyResizable, isPropertyRestartEditable, isRestartResizable

Fields

PIXELS_PER_METER

public static final int **PIXELS_PER_METER**

Constant value: **20**

MAX_PAGE_WIDTH

public static final int **MAX_PAGE_WIDTH**

Constant value: **110000**

MAX_PAGE_HEIGHT

public static final int **MAX_PAGE_HEIGHT**

Constant value: **85000**

Constructors

ViewComponent

public **ViewComponent**()

Creates a new instance of ViewComponent

ViewComponent

```
public ViewComponent(AbstractModel model,
                     int num)
```

Creates a new instance of ViewComponent with a new model and display number

Methods

enableVisualGrouping

```
public final void enableVisualGrouping()
```

Enables visual grouping for this view. This method must be called on the view component when initially created by `AbstractModel.createComponent(Category)`.

isVisualGroupingEnabled

```
public final boolean isVisualGroupingEnabled()
```

Returns true if visual grouping is enabled for this view.

enableArrowKeyMovement

```
public final void enableArrowKeyMovement()
```

Enables arrow key movement of graphical elements in this view. This method must be called on the view component when initially created by `AbstractModel.createComponent(Category)`.

isArrowKeyMovementEnabled

```
public final boolean isArrowKeyMovementEnabled()
```

Returns true if arrow key movement is enabled for this view.

removeVerify

```
public boolean removeVerify()
```

Verifies that this component can be removed from it's model without unforeseen side effects and returns the truth of this assumption.

This method may request user verification from the user.

removeFromModel

```
public void removeFromModel(AbstractModel model)
```

This method removes this AbstractComponent from `model`.

load

```
public void load(ViewCompRec rec)
```

Loads this view's data from the given ViewCompRec previously stored with `store(PibFile)`

---

clone

`public Object clone()`

this clone method copies all primitive data types. it will only need to be overridden if inherited classes contain objects which need to be copied. This should not be used for copy/paste as it initializes contained DrawnComponents and thus breaks the subclass copy() methods.

---

copy

`public AbstractComponent copy(AbstractModel sm)`

the following copy method produces a deep clone of a composite base so that it can be put in a copy model clipboard. Connections to objects in the Vector which is the argument are preserved. Connections to all other objects are removed. NOTE: COPY/PASTE should use this in place of clone()

---

label

`public String label()`

Returns a String suitable for describing the component type on a dialog.

This default implementation returns the class name.

---

copyFrom

`public void copyFrom(GenericObject o)`

Copy the attributes from a clone to this instance.

Parameters:
`o` - The cloned object.

---

toString

`public String toString()`

ViewComponents use their name as their to-string, as opposed to including "View CCnumber" before their name.

---

canConnectTo

`public boolean canConnectTo(AbstractComponent target)`

This method checks to see if it is allowable for this AbstractComponent to connect to the given target.

Parameters:
`target` - the AbstractComponent object to which a connection has been requested.

Returns:
true if allowed, false if not allowed.

---

getCategory

```
public Category getCategory()
```

Retrieves the most narrow category that this component is a member of.

---

setVisible

```
public void setVisible(boolean value)
```

Shows or hides this ViewComponent's DrawnView and it's accompanying dialog or panel.

---

setVisible

```
public void setVisible(boolean value,
        ViewCompRec record)
```

Shows or hides this ViewComponent's DrawnView and it's accompanying dialog or panel.

Parameters:

`value` - the visibility flag

`record` - an optional `com.cafean.client.io.med.ViewCompRec` used to determine whether the view is docked or undocked, its undocked location, etc. This value may be `null`

---

layoutView

```
public void layoutView()
```

Runs the organize function on the view component if it exists.

---

buildView

```
public void buildView()
```

This initializes a new DrawnView based on a View Component. Since this does not set the view visible, functions can be called on the view before it is displayed. This is used to speed up generating a new view of components.

---

addPibBlock

```
public void addPibBlock(com.apt.xdr.PibBlock block)
```

Adds a new Pibblock to this View Component. While loading a sam file drawing records are added to the ViewComponent without translating them at all.

---

getPibBlocks

```
public com.apt.xdr.PibBlock[] getPibBlocks()
```

Returns the array of compressed PibBlocks stored inside this view.

---

setPibBlocks

```
public void setPibBlocks(com.apt.xdr.PibBlock[] array)
```

Replaces the array of PibBlocks for this view.

Parameters:

`array` - the PibBlock[] for this closed view.

---

addComponent

```
public void addComponent(AbstractComponent comp,
        boolean select)
```

Adds a new component to this view, either by adding it's pibblock if the view is not created, or by adding it directly to the view.

---

addComponents

```
public void addComponents(Iterator itr)
```

Uses the given iterator to add all of those components to the View, if it exists, or adds them to the PibBlock list if they don't.

Parameters:

`itr` - An Iterator on a list of AbstractComponents

---

addComponents

```
public void addComponents(Iterator itr,
        boolean select)
```

Uses the given iterator to add all of those components to the View, if it exists, or adds them to the PibBlock list if they don't.

Parameters:

`itr` - an Iterator on a list of AbstractComponents

`select` - if true and the view is visible the added components will be selected.

---

hasBlocks

```
public boolean hasBlocks()
```

Determines whether this view has anything inside it's blocks vector. Basically whether this view has been opened or is empty.

Returns:

true if this view is new, or has already been opened.

---

store

```
public boolean store(com.apt.xdr.PibFile file)
```

Writes this view and it's contents to the given PibFile. May call `storeDrawnComponent` to create pib blocks for components that are not part of the core ModelEditor distribution.

Parameters:

`file` - the PibFile to write this ViewComponent to

Returns:

true on success; false on failure with messages printed to the message window.

---

storeComponent

`public com.apt.xdr.PibBlock storeComponent(java.awt.Component c)`

Stores the given Component into a PibBlock for use in saving into a PIB formatted file.

Parameters:

`a` - PibBlock containing the stored component

---

getDrawnView

`public DrawnView getDrawnView()`

A getter for the DrawnView owned by this ViewComponent. This is NULL if this view has never been opened.

Returns:

The DrawnView owned by this ViewComponent

---

getLocation

`public java.awt.Point getLocation()`

Gets the location of the `DrawnView` or the locally stored location, if the DrawnView has never been opened.

Returns:

the java.awt.Point location of the DrawnView on the screen.

---

setLocation

`public void setLocation(java.awt.Point location)`

Sets the the locally stored location of the DrawnView. This is used for loading the data from the file, or setting the initial position.

Parameters:

`location` - the java.awt.Point location of the DrawnView on the screen.

---

getSize

`public java.awt.Dimension getSize()`

Gets the size of the `DrawnView` or the locally stored size, if the DrawnView has never been opened.

Returns:

the java.awt.Dimension size of the DrawnView on the screen.

---

setSize

`public void setSize(java.awt.Dimension size)`

Sets the size of the `DrawnView` or the locally stored size, if the DrawnView has never been opened.

Parameters:

`size` - the java.awt.Dimension size of the DrawnView on the screen.

getCanvasSize

```
public java.awt.Dimension getCanvasSize()
```

Gets the size of the `view's` canvas or the locally stored canvas size, if the DrawnView has never been opened.

Returns:
the java.awt.Dimension size of the DrawnView's canvas.

See Also:
`DrawnView.getCanvasSize()`

---

setCanvasSize

```
public void setCanvasSize(java.awt.Dimension canvasSize)
```

Sets the locally stored canvas size.

Parameters:
`canvasSize` - the java.awt.Dimension size of the DrawnView's canvas.

---

getZoomScale

```
public double getZoomScale()
```

Gets the current zoom factor from the `view's` zoom panel or the locally stored zoom factor, if the DrawnView has never been opened.

Returns:
the double zoom factor from the DrawnView.

See Also:
`DrawnView.getZoomScale()`

---

setZoomScale

```
public void setZoomScale(double zoomScale)
```

Sets the locally stored zoom scale.

Parameters:
`zoomScale` - the double for the zoom scale factor of the DrawnView.

---

setShowPoints

```
public void setShowPoints(boolean showPoints)
```

Turns on displaying the connection points on AbstractComponents inside this view.

---

getShowPoints

```
public boolean getShowPoints()
```

Returns whether or not the connection points are drawn on AbstractComponents inside this view.

Returns:
true if the connection points are drawn.

---

undockView

`public void undockView()`

Builds this view and displays it as an undocked separate window.

---

dockView

`public void dockView()`

Builds this view and displays it as a docked window.

---

getCustomPopupItems

`public Vector getCustomPopupItems()`

Creates Custom Menu Items for any popup dialog involving this component

---

getViewPosition

`public java.awt.Point getViewPosition()`

Getter for property viewPosition.

Returns:
Value of property viewPosition.

---

setViewPosition

`public void setViewPosition(java.awt.Point viewPosition)`

Setter for property viewPosition.

Parameters:
`viewPosition` - New value of property viewPosition.

---

isViewVisible

`public boolean isViewVisible()`

Returns true if the view is currently visible, or false if the view is not visible, or doesn't exist yet.

---

getImage

`public ImageIcon getImage()`

Gets the ImageIcon for rendering this ViewComponent.

Returns:

the ImageIcon for the DrawnViewComponent

---

getImageData

```
public byte[] getImageData()
```

Gets the icon image for the DrawnViewComponent that renders this ViewComponent.

Returns:
the byte[] holding the raw image data directly from the file.

---

setImageData

```
public void setImageData(byte[] imageData)
```

Sets the icon image for the DrawnViewComponent that renders this ViewComponent. This also creates the image from the raw datas.

Parameters:
imageData - the byte[] holding the raw image data directly from the file.

---

createDrawnComponent

```
public DrawnComponent createDrawnComponent()
```

Returns the renderer for this abstract component. This will be extended by any abstract component that needs a renderer.

---

clearViewSelection

```
public void clearViewSelection()
```

Clears the current selection on the View if it exists

---

addAnnotation

```
public void addAnnotation(Annotation comp)
```

Adds an Annotation to this ViewComponent, or to the DrawnView if the view has been opened.

Parameters:
comp - the Annotation.

---

getAnnotationArray

```
public byte[] getAnnotationArray()
```

Gets an byte array containing the XML encoded stream of annotations from this DrawnView. This byte array is used to store the annotations into the PibBlock for this ViewComponent

Returns:
the byte[] containing all of the XML encoded annotations.

---

readAnnotationArray

```
public void readAnnotationArray(byte[] byteArr,
        AbstractModel model)
```

Retrieves all of the Annotations from an XML encoded byte array of annotations, and adds them to this ViewComponent.

Parameters:

byteArr - the byte[] containing all of the XML encoded annotations.
model - the AbstractModel.

See Also:

addAnnotation(Annotation)

---

getEmbedConnArray

```
public byte[] getEmbedConnArray()
```

Gets an byte array containing the XML encoded stream of annotations from this DrawnView. This byte array is used to store the annotations into the PibBlock for this ViewComponent

Returns:

the byte[] containing all of the XML encoded annotations.

---

readEmbedConnArray

```
public void readEmbedConnArray(byte[] byteArr)
```

Retrieves all of the Annotations from an XML encoded byte array of annotations, and adds them to this ViewComponent.

Parameters:

byteArr - the byte[] containing all of the XML encoded annotations.
model - the AbstractModel.

See Also:

addAnnotation(Annotation)

---

complete

```
public void complete()
```

When a ViewComponent is created, it immediately opens.

---

fireComponentChanged

```
public void fireComponentChanged()
```

This calls the component changed function on all of the listeners currently listening to this abstract component

---

fireComponentDeleted

```
public void fireComponentDeleted()
```

This calls the component deleted function on all of the listeners currently listening to this abstract component

getBackground

`public java.awt.Color` **`getBackground`**`()`

Gets the background color from the DrawnView, or from the local variable if the DrawnView has not been created.

Returns:
Value of property background.

setBackground

`public void` **`setBackground`**`(java.awt.Color background)`

Sets the background color on the DrawnView if it has been created, and stores it inside the local variable.

Parameters:
`background` - the Color of the background.

setStoredBackground

`public void` **`setStoredBackground`**`(java.awt.Color background)`

This just sets the background color of the stored local variable. This is used inside the GUI editor, to allow the background to be changed and still be cancellable.

Parameters:
`background` - the Color of the background.

getColorArray

`public int[]` **`getColorArray`**`()`

This returns the current background in the form of an array of three integers.

Returns:
the int[] containing the RGB values for the background color.

See Also:
`getBackground()`

setColorArray

`public void` **`setColorArray`**`(int[] array)`

Sets the current background from an array of three integers.

Parameters:
`array` - the int[] containing the RGB values for the background color.

See Also:
`getBackground()`

loadComponent

```
public static java.awt.Component loadComponent(com.apt.xdr.PibBlock block,
        AbstractModel model)
```

---

getPixelsPerMeter

```
public int getPixelsPerMeter()
```

Getter for property pixelsPerMeter.

Returns:
Value of property pixelsPerMeter.

---

setPixelsPerMeter

```
public void setPixelsPerMeter(int pixelsPerMeter)
```

Setter for property pixelsPerMeter.

Parameters:
`pixelsPerMeter` - New value of property pixelsPerMeter.

---

getEmbeddedCons

```
public EmbeddedConnectionMap getEmbeddedCons()
```

---

setEmbeddedCons

```
public void setEmbeddedCons(EmbeddedConnectionMap cons)
```

---

setImageElement

```
public void setImageElement(ViewImgDataElem element)
```

A stub method used in ViewComponentBeanInfo to edit the image data byte[] contained in this ViewComponent.

---

getImageElement

```
public ViewImgDataElem getImageElement()
```

A stub method used in ViewComponentBeanInfo to edit the image data byte[] contained in this ViewComponent.

---

popupDataDialog

```
public void popupDataDialog(java.awt.Window parent,
        boolean modal)
```

Creates a new bean editing dialog for this object or resets and refreshes this object's current editing dialog.

---

isShowGrid

```
public boolean isShowGrid()
```

Getter for property showGrid.

Returns:
Value of property showGrid.

---

setShowGrid

```
public void setShowGrid(boolean showGrid)
```

Setter for property showGrid.

Parameters:
`showGrid` - New value of property showGrid.

---

isSnapToGrid

```
public boolean isSnapToGrid()
```

Getter for property snapToGrid.

Returns:
Value of property snapToGrid.

---

setSnapToGrid

```
public void setSnapToGrid(boolean snapToGrid)
```

Setter for property snapToGrid.

Parameters:
`snapToGrid` - New value of property snapToGrid.

---

getGridLineColor

```
public java.awt.Color getGridLineColor()
```

Getter for property lineColor.

Returns:
Value of property lineColor.

---

setGridLineColor

```
public void setGridLineColor(java.awt.Color lineColor)
```

Setter for property lineColor.

Parameters:

`lineColor` - New value of property lineColor.

---

getTransparency

`public float` **`getTransparency`**`()`

Getter for property transparency.

Returns:
Value of property transparency.

---

setTransparency

`public void` **`setTransparency`**`(float transparency)`

Setter for property transparency.

Parameters:
`transparency` - New value of property transparency.

---

getHorizGridSpacing

`public int` **`getHorizGridSpacing`**`()`

Getter for property horizGridSpacing.

Returns:
Value of property horizGridSpacing.

---

setHorizGridSpacing

`public void` **`setHorizGridSpacing`**`(int horizGridSpacing)`

Setter for property horizGridSpacing.

Parameters:
`horizGridSpacing` - New value of property horizGridSpacing.

---

getVertGridSpacing

`public int` **`getVertGridSpacing`**`()`

Getter for property vertGridSpacing.

Returns:
Value of property vertGridSpacing.

---

setVertGridSpacing

`public void` **`setVertGridSpacing`**`(int vertGridSpacing)`

Setter for property vertGridSpacing.

Parameters:

vertGridSpacing - New value of property vertGridSpacing.

---

getVertSnapSpacing

```
public int getVertSnapSpacing()
```

Getter for property vertSnapSpacing.

Returns:
Value of property vertSnapSpacing.

---

setVertSnapSpacing

```
public void setVertSnapSpacing(int vertSnapSpacing)
```

Setter for property vertSnapSpacing.

Parameters:
vertSnapSpacing - New value of property vertSnapSpacing.

---

getHorizSnapSpacing

```
public int getHorizSnapSpacing()
```

Getter for property horizSnapSpacing.

Returns:
Value of property horizSnapSpacing.

---

setHorizSnapSpacing

```
public void setHorizSnapSpacing(int horizSnapSpacing)
```

Setter for property horizSnapSpacing.

Parameters:
horizSnapSpacing - New value of property horizSnapSpacing.

---

isLocked

```
public boolean isLocked()
```

Returns true if this view is locked.

---

setLocked

```
public void setLocked(boolean locked)
```

sets this view's locked state

---

setLockedConstrained

```
public void setLockedConstrained(boolean locked)
```

sets this view's locked state

---

isGridAbove

```
public boolean isGridAbove()
```

Returns true if this View's grid will be painted on top of its contained elements.

---

setGridAbove

```
public void setGridAbove(boolean above)
```

If set to true, this View's grid will be painted on top of its contained elements.

---

isPropertyEnabled

```
public boolean isPropertyEnabled(String propertyName)
```

---

isPropertyRequired

```
public boolean isPropertyRequired(String propertyName)
```

---

isPropertyRestartEditable

```
public boolean isPropertyRestartEditable(String propertyName)
```

---

getAttributeIndex

```
public int getAttributeIndex(String propertyName)
```

---

isPropertyResizable

```
public boolean isPropertyResizable(String propertyName)
```

---

isRestartResizable

```
public boolean isRestartResizable(String propertyName)
```

---

isPropertyActive

```
public boolean isPropertyActive(String propertyName)
```

getAttributeGroups

`public static String[] getAttributeGroups()`

Retrieves the Attribute Group Names for this View Component.

Returns:
a String[] in which each entry is an attribute group name

getAttributeGroup

`public static String getAttributeGroup(String property)`

Retrieves the name of the Attribute Group for the given property name.

Parameters:
`property` - a String containing the name of the property

Returns:
a String containg the group name for the given property or null

getAttributesForGroup

`public static String[] getAttributesForGroup(String groupName)`

Retrieves the Attribute Names for the given group.

Returns:
a String[] containing the attribute names for the given group name or a 0 length String[] if none are found.

getCCnumber

`public String getCCnumber()`

Retrieves this objects's CC number. The CC number is the object's display number in String form.

getWidthScaleFactor

`public double getWidthScaleFactor()`

Gets this view's width scale factor. This factor is intended for use in scaling the diameter or width of components that may have one dimension much larger than the other; such as a 10 meter long, 0.1 meter wide pipe.

setWidthScaleFactor

`public void setWidthScaleFactor(double factor)`

Sets this view's width scale factor. This factor is intended for use in scaling the diameter or width of components that may have one dimension much larger than the other; such as a 10 meter long, 0.1 meter wide pipe.

getUniqueID

```
public String getUniqueID()
```

Provides a unique component id for components who may not contain a component number.

Package

com.cafean.client.analysis.numerics

com.cafean.client.analysis.numerics
Class FunctionSource

```
java.lang.Object
   |
   +-com.cafean.client.analysis.numerics.FunctionSource
```

---

public class **FunctionSource**
extends Object

Holds the source code for the user defined function along with a reference to the UDF.

Constructor Summary

public	`FunctionSource()` Creates a new instance of FunctionSource
public	`FunctionSource(ComponentElement owner)` Construct an empty instance.
public	`FunctionSource(String sourceCode, ComponentElement owner)` Construct an empty instance.

Method Summary

`ComponentElement`	`getOwner()` Getter for property owner.
`String`	`getSourceCode()` Getter for property sourceCode.
`void`	`setOwner(ComponentElement owner)` Setter for property owner.
`void`	`setSourceCode(String sourceCode)` Setter for property sourceCode.

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

FunctionSource

`public FunctionSource()`

Creates a new instance of FunctionSource

FunctionSource

```
public FunctionSource(ComponentElement owner)
```

Construct an empty instance.

Parameters:
owner - the owner of this subelement.

FunctionSource

```
public FunctionSource(String sourceCode,
                      ComponentElement owner)
```

Construct an empty instance.

Parameters:
owner - the owner of this subelement.

Methods

getSourceCode

```
public String getSourceCode()
```

Getter for property sourceCode.

Returns:
Value of property sourceCode.

setSourceCode

```
public void setSourceCode(String sourceCode)
```

Setter for property sourceCode.

Parameters:
sourceCode - New value of property sourceCode.

getOwner

```
public ComponentElement getOwner()
```

Getter for property owner.

Returns:
Value of property owner.

setOwner

```
public void setOwner(ComponentElement owner)
```

Setter for property owner.

Parameters:

`owner` - New value of property owner.

com.cafean.client.analysis.numerics
Class ParametricIteration

```
java.lang.Object
   |
   +-com.cafean.client.analysis.numerics.ParametricIteration
```

---

public class **ParametricIteration**
extends Object

This will perform the iteration used for exporting a series of ASCII runs.

Field Summary

public static final	NAME_NUMBER Indicates that the file name will include the iteration point for each constant used. Value: **0**
public static final	NAME_VALUE Indicates that the file name will include the name and value for each constant used. Value: **1**

Constructor Summary

public	ParametricIteration() Creates a new instance of ParametricIteration
public	ParametricIteration(UserDefinedConstant[] constants, String baseFile, int convention)

Method Summary

boolean	hasNext() Determines if there's more iteration to be done.
String	next() Iterates through the Constants and gets the next file name.
void	resetIteration() Resets this iterator
void	setBaseFile(String baseFile) Sets the basefile name.
void	setConstants(UserDefinedConstant[] constants) Sets the constants.
void	setModel(AbstractModel model) Sets the model on this iteration.

void	setNameConvention(int convention) Sets the naming convention for determining the name of each file.
String	writeConstantTable(UserDefinedConstant constant) Writes out the value table of a tabular parametric constant.
String	writeDescription() Writes the export description of this model.
String	writeExportNote(int count) Builds an export note for the current state of the parametric export.
String	writeName(String name)
String	writeTableHeader() Writes the export file table header.
String	writeTableLine(int count, String filename, String filePath) Returns a single line of the export file table in HTML format.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Fields

NAME_NUMBER

public static final int **NAME_NUMBER**

Indicates that the file name will include the iteration point for each constant used.
Constant value: **0**

NAME_VALUE

public static final int **NAME_VALUE**

Indicates that the file name will include the name and value for each constant used. This may produce large file names.
Constant value: **1**

Constructors

ParametricIteration

public **ParametricIteration**()

Creates a new instance of ParametricIteration

ParametricIteration

```
public ParametricIteration(UserDefinedConstant[] constants,
                           String baseFile,
                           int convention)
```

Methods

setModel

```
public void setModel(AbstractModel model)
```

Sets the model on this iteration.

---

setBaseFile

```
public void setBaseFile(String baseFile)
```

Sets the basefile name.

---

setConstants

```
public void setConstants(UserDefinedConstant[] constants)
```

Sets the constants.

---

setNameConvention

```
public void setNameConvention(int convention)
```

Sets the naming convention for determining the name of each file.

---

resetIteration

```
public void resetIteration()
```

Resets this iterator

---

hasNext

```
public boolean hasNext()
```

Determines if there's more iteration to be done.

---

next

```
public String next()
```

Iterates through the Constants and gets the next file name.

---

writeExportNote

```
public String writeExportNote(int count)
```

Builds an export note for the current state of the parametric export. This note is set inside the model before each export.

---

writeName

```
public String writeName(String name)
```

---

writeDescription

```
public String writeDescription()
```

Writes the export description of this model.

---

writeTableHeader

```
public String writeTableHeader()
```

Writes the export file table header. This returns a String in HTML format for defining the table header.

---

writeTableLine

```
public String writeTableLine(int count,
        String filename,
        String filePath)
```

Returns a single line of the export file table in HTML format.

---

writeConstantTable

```
public String writeConstantTable(UserDefinedConstant constant)
```

Writes out the value table of a tabular parametric constant. If the passed constant is incremental this will return an empty string.

com.cafean.client.analysis.numerics
Class UserDefinedConstant

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.AbstractComponent
         |
         +-com.cafean.client.analysis.numerics.UserDefinedValue
            |
            +-com.cafean.client.analysis.numerics.UserDefinedConstant
```

All Implemented Interfaces:
PropertyController, IdentHolder, StateEditable, Cloneable, Checkable, ComponentElement, Cloneable

---

public class **UserDefinedConstant**
extends UserDefinedValue
implements Cloneable, ComponentElement, Checkable, Cloneable, StateEditable, IdentHolder, PropertyController

The UserDefinedConstant serves as a UserDefinedValue that may be parametric. This may be used as an input variable for a `UserDefinedFunction` and it may be refered to directly by a `Real`. A parametric constant has an array of values, and a default index. When the constant is not the parameter for a parametric ASCII export or a parametric job submission, the current value is always the value at the default index.

Fields inherited from class `com.cafean.client.analysis.AbstractComponent`
`leftDiffComponent`, `leftDiffName`, `leftShortDiffName`, `rightDiffComponent`, `rightDiffName`, `rightShortDiffName`

Fields inherited from class `com.cafean.client.analysis.GenericObject`
`DATA_COMPLETE`, `DATA_ERROR`, `DATA_INCOMPLETE`, `DATA_WARNING`

Fields inherited from interface `javax.swing.undo.StateEditable`
`RCSID`

Fields inherited from interface `com.cafean.client.ui.beans.PropertyController`
`ALL`, `COLOR_OPTIONAL`, `DISABLED`, `NONE`, `OPTIONAL`, `REQUIRED`

Constructor Summary

public	`UserDefinedConstant()` Creates a new UserDefinedConstant
public	`UserDefinedConstant(AbstractModel model, int componentNumber)` Creates a new instance of UserDefinedConstant inside a model with a specified component number.

Method Summary

void	addParameter(Real parameter) Appends a new parameter to the end of the parameter list.
Object	clone() Creates and returns a copy of this object.
void	complete()
void	copyFrom(GenericObject object) Copy the attributes from a source object to this instance. This is used for copying data from an edited working copy into the original object. Most notably, it is used to enable undo/redo for the legacy beanless ModelEditor architecture as well as for importing new altered data from restart decks for some plugins. This should call copyFrom on children that support it. **Note: Never call copyFrom from clone or copy!**
int	getAttributeIndex(String property) Returns a relative index that can be used to order property lists.
Category	getCategory() The category for a UserDefinedConstant is always CAT_CONSTANT
int	getCurrentIndex() This gets the transient value used for determining where inside a parametric iteration this constant is.
boolean	getCurrentParametric() Gets whether this constant is the current parameter or not
Real	getCurrentValue() If this constant is the current parameter of a parametric run or job submission, this returns the value at the current index.
int	getDefaultIndex() Gets the default index inside this constant.
Real	getDefaultValue() Gets the default value of this constant.
Real	getEndValue() Gets the end value of this constant.
Real	getIncrement() Gets the increment of this constant.
Real	getParameterAt(int index) Returns the parameter at a given index
Real[]	getParameters()
int	getParametersCount() Gets the number of parameters in this constant.

String	getRunValues() Returns a short string that indicates this constants current values for a parametric export.
Real	getStartValue() Gets the start value of this constant.
Real	getUnits()
boolean	hasNext() Determines if the parameter has processed all of its parameters during parametric export or job submission.
boolean	isIncremental() Gets the flag that indicates if this parametric constant is incremental.
boolean	isParametric() Determines if this constant is defined as a parametric constant that has multiple values or not.
boolean	isPropertyActive(String propertyName) Returns false if this object has a property with the given name that is considered inactive; otherwise true.
boolean	isPropertyEnabled(String propertyName) Returns false if this object has a property with the given name that has dependency code that fails;true otherwise
boolean	isPropertyRequired(String propertyName) Returns false if this object has a property with the given name that has requirement code that fails; true otherwise.
boolean	isPropertyResizable(String propertyName) Returns false if this object has an array property with the given name that should not normally be resizable.
boolean	isPropertyRestartEditable(String propertyName) Returns true if this object has a property with the given name that should be editable during a restart edit; false otherwise.
boolean	isRestartResizable(String propertyName) Returns false if this object has an array property with the given name that should not be resizable while editing a restart.
void	iterate() Iterates the current index inside a parameter during parametric export or job submission.
String	label() Returns a String suitable for describing the component type on a dialog. This default implementation returns the class name.
void	removeParameter(int index) Removes the parameter at the given index.
void	resetIteration() Resets the current index of a parameter during parametric export or job submission.

void	restoreState(Hashtable state)
void	setCurrentParametric(boolean currentParametric) Sets this constant as the current parameter during a parametric export or job submission.
void	setDefaultIndex(int defaultIndex) Sets the default index inside this constant.
void	setDefaultValue(Real val_) Sets the default value of this constant.
void	setEndValue(Real val_) Sets the end value of this constant.
void	setIncrement(Real val_) Sets the increment of this constant.
void	setIncremental(boolean val_) Sets the flag that indicates if this parametric constant is incremental.
void	setParameter(Real parameter) Setter for the value in a non-parametric constant.
void	setParameterAt(int index, Real parameter) Sets a parametric value at the given index.
void	setParameters(Real[] parameters)
void	setParametric(boolean parametric) Sets this constant to be parametric with multiple values or constant with only one.
void	setStartValue(Real val_) Sets the start value of this constant.
void	setUnits(Real units) This sets all of the values inside this constant to have the same units as the Real passed in.
void	storeState(Hashtable state)
String	toString()

Methods inherited from class com.cafean.client.analysis.numerics.UserDefinedValue

createDrawnComponent, fireComponentDeleted, getCurrentValue, getUniqueID, getUnits, popupDataDialog, replaceNumericNames, restoreState, setName, setUnits, setUnits

Methods inherited from class com.cafean.client.analysis.AbstractComponent

addALDocRef, addComponentListener, addConnection, addMessage, addMessage, addMessage, addToModel, addToModel, canConnectTo, clearConnections, clone, complete, connectTo, connectTo, copy, createDrawnComponent, createSourceData, createTargetData, DBtypeCode, disconnect, disconnectFrom, fireComponentChanged, fireComponentChanged, fireComponentConnected, fireComponentDeleted, fireComponentDisconnected, getALDocDisplayName, getALDocRefs, getALDocRefs, getALDocShortNames, getALDocShortNames, getCatCCComparator, getCategory, getCCNumberComparator, getComponent, getComponentDependencies, getConnectionCount, getConnectionName, getConnections, getConnectionTypes, getCustomPopupActions, getCustomPopupItems, getGroupedConnections, getModel, getName, getNewCompIdent, getOrder, getOrderComparator, getOwner, getRealSize, getSharedComponents, getUniqueID, hasALDocRefs, includeInLoopcheck, isOkayForExport, isOkayForExport, label, popupDataDialog, rebuildConnections, reconnectIdentReferences, reconnectImage, removeALDocRef, removeComponentListener, removeFromModel, removeVerify, restoreALRefState, restoreState, setALDocRefs, setComponentNumber, setDeleted, setModel, setOrder, storeALRefState, toShortString, toString, updateVersion, userDelete, writeName

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder

reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ComponentElement

getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement

getModel

Methods inherited from interface com.cafean.client.analysis.Checkable
isOkayForExport, isOkayForExport

Methods inherited from interface com.cafean.client.ui.beans.PropertyController
getAttributeIndex, isPropertyActive, isPropertyEnabled, isPropertyRequired, isPropertyResizable, isPropertyRestartEditable, isRestartResizable

Constructors

UserDefinedConstant

```
public UserDefinedConstant()
```

Creates a new UserDefinedConstant

UserDefinedConstant

```
public UserDefinedConstant(AbstractModel model,
                           int componentNumber)
```

Creates a new instance of UserDefinedConstant inside a model with a specified component number.

Parameters:

`model` - the AbstractModel.
`number` - the component number for this function.

Methods

copyFrom

```
public void copyFrom(GenericObject object)
```

Copy the attributes from a source object to this instance.

This is used for copying data from an edited working copy into the original object. Most notably, it is used to enable undo/redo for the legacy beanless ModelEditor architecture as well as for importing new altered data from restart decks for some plugins.

This should call copyFrom on children that support it.

Note: Never call copyFrom from clone or copy!

clone

```
public Object clone()
```

Creates and returns a copy of this object.

getCurrentValue

```
public Real getCurrentValue()
```

If this constant is the current parameter of a parametric run or job submission, this returns the value at the current index. Otherwise, this returns the value at the default index.

Returns:
Real the current value.

---

getCurrentIndex

`public int` **`getCurrentIndex`**`()`

This gets the transient value used for determining where inside a parametric iteration this constant is. This is used when exporting a parametric run.

---

getCategory

`public Category` **`getCategory`**`()`

The category for a UserDefinedConstant is always `CAT_CONSTANT`

Returns:
the Category for this UserDefinedConstant.

---

toString

`public String` **`toString`**`()`

---

label

`public String` **`label`**`()`

Returns a String suitable for describing the component type on a dialog.

This default implementation returns the class name.

---

getParametersCount

`public int` **`getParametersCount`**`()`

Gets the number of parameters in this constant.

Returns:
The number of values in the parameter array.

---

getParameterAt

`public Real` **`getParameterAt`**`(int index)`

Returns the parameter at a given index

Parameters:
`index` - the index of the parameter.

Returns:
the Real value at the given index.

---

setParameters

```
public void setParameters(Real[] parameters)
```

---

getParameters

```
public Real[] getParameters()
```

---

setParameterAt

```
public void setParameterAt(int index,
        Real parameter)
```

Sets a parametric value at the given index.

Parameters:
`index` - the index.
`parameter` - the Real parameter to be set.

---

addParameter

```
public void addParameter(Real parameter)
```

Appends a new parameter to the end of the parameter list.

Parameters:
`parameter` - the Real that will be appended.

---

removeParameter

```
public void removeParameter(int index)
```

Removes the parameter at the given index. This will not delete the last parameter.

Parameters:
`index` - the index of the parameter to be removed.

---

iterate

```
public void iterate()
```

Iterates the current index inside a parameter during parametric export or job submission.

---

resetIteration

```
public void resetIteration()
```

Resets the current index of a parameter during parametric export or job submission.

---

hasNext

`public boolean hasNext()`

Determines if the parameter has processed all of its parameters during parametric export or job submission.

Returns:
true if the current index is less then the number of parameters.

---

getDefaultIndex

`public int getDefaultIndex()`

Gets the default index inside this constant.

Returns:
The index used to determine the value outside of a parametric export.

---

setDefaultIndex

`public void setDefaultIndex(int defaultIndex)`

Sets the default index inside this constant.

Parameters:
`defaultIndex` - The index used to determine the value outside of a parametric export.

---

setCurrentParametric

`public void setCurrentParametric(boolean currentParametric)`

Sets this constant as the current parameter during a parametric export or job submission.

Parameters:
`currentParametric` - whether or not this constant is the current parameter.

---

getCurrentParametric

`public boolean getCurrentParametric()`

Gets whether this constant is the current parameter or not

---

isParametric

`public boolean isParametric()`

Determines if this constant is defined as a parametric constant that has multiple values or not.

Returns:
true if this constant is set parametric.

---

setParametric

`public void setParametric(boolean parametric)`

Sets this constant to be parametric with multiple values or constant with only one.

Parameters:

`parametric` - whether or not this constant has multiple values.

---

isIncremental

`public boolean isIncremental()`

Gets the flag that indicates if this paramétric constant is incremental.

---

setIncremental

`public void setIncremental(boolean val_)`

Sets the flag that indicates if this parametric constant is incremental.

---

getDefaultValue

`public Real getDefaultValue()`

Gets the default value of this constant.

---

setDefaultValue

`public void setDefaultValue(Real val_)`

Sets the default value of this constant.

---

getStartValue

`public Real getStartValue()`

Gets the start value of this constant.

---

setStartValue

`public void setStartValue(Real val_)`

Sets the start value of this constant.

---

getEndValue

`public Real getEndValue()`

Gets the end value of this constant.

---

setEndValue

`public void setEndValue(Real val_)`

Sets the end value of this constant.

---

getIncrement

`public Real getIncrement()`

Gets the increment of this constant.

---

setIncrement

`public void setIncrement(Real val_)`

Sets the increment of this constant.

---

getUnits

`public Real getUnits()`

Returns a clone of the current value.

---

setUnits

`public void setUnits(Real units)`

This sets all of the values inside this constant to have the same units as the Real passed in.

Parameters:

`units` - the Real containing the units to be set.

---

isPropertyEnabled

`public boolean isPropertyEnabled(String propertyName)`

Returns false if this object has a property with the given name that has dependency code that fails;true otherwise

Parameters:

`propertyName` - a String containing the property name to check

---

isPropertyRequired

`public boolean isPropertyRequired(String propertyName)`

Returns false if this object has a property with the given name that has requirement code that fails; true otherwise.

Parameters:

`propertyName` - a String containing the property name to check

---

isPropertyRestartEditable

`public boolean isPropertyRestartEditable(String propertyName)`

Returns true if this object has a property with the given name that should be editable during a restart edit; false otherwise.

Parameters:

propertyName - a String containing the property name to check

---

getAttributeIndex

```
public int getAttributeIndex(String property)
```

Returns a relative index that can be used to order property lists.

Parameters:
propertyName - a String containing the property name to check

---

isPropertyResizable

```
public boolean isPropertyResizable(String propertyName)
```

Returns false if this object has an array property with the given name that should not normally be resizable.

Parameters:
propertyName - a String containing the property name to check

---

isRestartResizable

```
public boolean isRestartResizable(String propertyName)
```

Returns false if this object has an array property with the given name that should not be resizable while editing a restart.

Parameters:
propertyName - a String containing the property name to check

---

isPropertyActive

```
public boolean isPropertyActive(String propertyName)
```

Returns false if this object has a property with the given name that is considered inactive; otherwise true.

Parameters:
propertyName - a String containing the name of the property to check

Returns:
false if the property is inactive, true otherwise

---

setParameter

```
public void setParameter(Real parameter)
```

Setter for the value in a non-parametric constant.

---

storeState

```
public void storeState(Hashtable state)
```

Stores the state of the object to permit undo by cloning itself and storing the clone. NOTE: If the component storing its state needs a deep copy that it's clone() method does not provide, it must override storeState to find that functionality elsewhere.

restoreState

`public void restoreState(Hashtable state)`

Restore the state of the bean from an earlier edit by using copyFrom on the previously stored clone.

complete

`public void complete()`

This completes this object's initialization in response to it's creation from a UI event.

This method should be overriden by subclasses that require user input or default values for a newly created component.

getRunValues

`public String getRunValues()`

Returns a short string that indicates this constants current values for a parametric export.

com.cafean.client.analysis.numerics
Class UserDefinedFunction

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.AbstractComponent
         |
         +-com.cafean.client.analysis.numerics.UserDefinedFunction
```

All Implemented Interfaces:
IdentHolder, StateEditable, Cloneable, Checkable, ComponentElement, Cloneable

---

public class **UserDefinedFunction**
extends AbstractComponent

This is the class that allows the user to write a block of python code that will be executed by the python interpreter. The function can have multiple inputs, that are either UserDefinedConstants or UserDefinedVariables, and multiple outputs that are UserDefinedVariables.

Fields inherited from class com.cafean.client.analysis.AbstractComponent
leftDiffComponent, leftDiffName, leftShortDiffName, rightDiffComponent, rightDiffName, rightShortDiffName

Fields inherited from class com.cafean.client.analysis.GenericObject
DATA_COMPLETE, DATA_ERROR, DATA_INCOMPLETE, DATA_WARNING

Fields inherited from interface javax.swing.undo.StateEditable
RCSID

Constructor Summary

public	UserDefinedFunction() Creates a new instance of UserDefinedFunctions
public	UserDefinedFunction(AbstractModel model, int number) Creates a new instance of UserDefinedFunction inside a model with a specified component number.

Method Summary

void	addInputId(int id) Adds the UserDefinedValue with the given ident into the input array, if the value is not already in the input array.
void	addOutputId(int id) Adds an output variable's ident to the output array if it doesn't already exist.

void	clearInputs() Removes all the entries in the input array
void	clearOutputs() This removes all of the variables from the output array.
Object	clone()
void	complete()
void	copyFrom(GenericObject object)
DrawnComponent	createDrawnComponent() UserDefinedFunctions are not visual components.
boolean	execute() Executes the python code in this function.
static UserDefinedConstant	findConstant(AbstractModel model, String name) Finds the UserDefinedConstant in the given model with the given name.
static UserDefinedFunction	findFunction(AbstractModel model, int ident) Finds the UserDefinedFunction in the given model with the given name.
static UserDefinedVariable	findVariable(AbstractModel model, String name) Finds the UserDefinedVariable in the given model with the given name.
Category	getCategory() The category for a UserDefinedFunction is always CAT_FUNCTION
Vector	getCustomPopupItems()
FunctionSource	getFunction() Returns the current python code for this function.
UserDefinedValue	getInputAt(int index) Returns the UserDefinedValue at the given index in the input array.
int	getInputCount() Gets the number of input variables.
int	getInputIdAt(int index) Gets the ident of the input varible at the given index.
double	getInputValue(int index) Returns the current value of the input at a given index.
double	getInputValue(String name) Returns the current value of the input variable with the given name.
UserDefinedVariable	getOutputAt(int index) Gets the UserDefinedVariable from the output array at a given index.

int	getOutputCount() Returns the number of variables this function outputs to.
int	getOutputIdAt(int index) Returns the ident stored at a given index in the output array.
String	getPythonHeader() Returns the header that is prepended to the python code before execution.
static String	getPythonInit() Gets the python initialization string.
String	label()
void	popupDataDialog(java.awt.Window parent, boolean modal)
void	reconnectIdentReferences(boolean preserveUnresolved, boolean useDbId)
void	removeInputAt(int index) Removes the input at the given index.
void	removeOutputAt(int index) Removes a the variable at the given index from the output array.
void	setFunction(FunctionSource function) Sets the python code for this function.
void	setInputIdAt(int index, int id) Sets the ident of the UserDefinedValue to be stored inside the input array at a given index.
void	setOutputIdAt(int index, int id) Sets the ident of an output at the given index.
void	setOutputValue(int index, double value) Sets the SI value given on the output variable at the given index.
void	setOutputValue(String name, double value) Sets the SI value on on the output with the given name.
String	toString()
void	validate()

Methods inherited from class com.cafean.client.analysis.AbstractComponent

addALDocRef, addComponentListener, addConnection, addMessage, addMessage, addMessage, addToModel, addToModel, canConnectTo, clearConnections, clone, complete, connectTo, connectTo, copy, createDrawnComponent, createSourceData, createTargetData, DBtypeCode, disconnect, disconnectFrom, fireComponentChanged, fireComponentChanged, fireComponentConnected, fireComponentDeleted, fireComponentDisconnected, getALDocDisplayName, getALDocRefs, getALDocRefs, getALDocShortNames, getALDocShortNames, getCatCCComparator, getCategory, getCCNumberComparator, getComponent, getComponentDependencies, getConnectionCount, getConnectionName, getConnections, getConnectionTypes, getCustomPopupActions, getCustomPopupItems, getGroupedConnections, getModel, getName, getNewCompIdent, getOrder, getOrderComparator, getOwner, getRealSize, getSharedComponents, getUniqueID, hasALDocRefs, includeInLoopcheck, isOkayForExport, isOkayForExport, label, popupDataDialog, rebuildConnections, reconnectIdentReferences, reconnectImage, removeALDocRef, removeComponentListener, removeFromModel, removeVerify, restoreALRefState, restoreState, setALDocRefs, setComponentNumber, setDeleted, setModel, setOrder, storeALRefState, toShortString, toString, updateVersion, userDelete, writeName

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder

reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ComponentElement

getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement

getModel

Methods inherited from interface com.cafean.client.analysis.Checkable

isOkayForExport, isOkayForExport

Constructors

UserDefinedFunction

```
public UserDefinedFunction()
```

Creates a new instance of UserDefinedFunctions

UserDefinedFunction

```
public UserDefinedFunction(AbstractModel model,
                           int number)
```

Creates a new instance of UserDefinedFunction inside a model with a specified component number.

Parameters:

model - the AbstractModel.

number - the component number for this function.

Methods

popupDataDialog

```
public void popupDataDialog(java.awt.Window parent,
        boolean modal)
```

Creates a new bean editing dialog for this object or resets and refreshes this object's current editing dialog.

clone

```
public Object clone()
```

Creates and returns a copy of this object.

copyFrom

```
public void copyFrom(GenericObject object)
```

Copy the attributes from a source object to this instance.

This is used for copying data from an edited working copy into the original object. Most notably, it is used to enable undo/redo for the legacy beanless ModelEditor architecture as well as for importing new altered data from restart decks for some plugins.

This should call copyFrom on children that support it.

Note: Never call copyFrom from clone or copy!

reconnectIdentReferences

```
public void reconnectIdentReferences(boolean preserveUnresolved,
        boolean useDbId)
```

Resets this component's internal ident references to refer to appropriate components in the current AbstractModel. Intended for use after adding a component to a model and thus it's DB_ID will have been set to it's ident in the previous model. For ident references use find__ByDbId to find the new component, then store it's ident.

Note: If this component's DB_ID is 0 this method does nothing.

---

getCustomPopupItems

```
public Vector getCustomPopupItems()
```

Creates Custom Menu Items for any popup dialog involving this component. The resulting Vector should contain on JMenu, JMenuItem and JSeparator instances for this component.

---

getCategory

```
public Category getCategory()
```

The category for a UserDefinedFunction is always CAT_FUNCTION

Returns:
the Category for this UserDefinedFunction.

---

createDrawnComponent

```
public DrawnComponent createDrawnComponent()
```

UserDefinedFunctions are not visual components. Therefore they always return null for when creating a DrawnComponent.

Returns:
null

---

getOutputCount

```
public int getOutputCount()
```

Returns the number of variables this function outputs to.

Returns:
the integer size of the output array.

---

getOutputIdAt

```
public int getOutputIdAt(int index)
```

Returns the ident stored at a given index in the output array.

Parameters:
index - the index into the output array.

Returns:

the ident stored at the given index.

---

setOutputIdAt

```
public void setOutputIdAt(int index,
          int id)
```

Sets the ident of an output at the given index.

Parameters:

`index` - the index into the output array.
`id` - the ident of the output variable.

---

addOutputId

```
public void addOutputId(int id)
```

Adds an output variable's ident to the output array if it doesn't already exist.

Parameters:

`id` - the ident of the output variable to be added.

---

clearOutputs

```
public void clearOutputs()
```

This removes all of the variables from the output array.

---

removeOutputAt

```
public void removeOutputAt(int index)
```

Removes a the variable at the given index from the output array.

Parameters:

`index` - the index into the output array.

---

setOutputValue

```
public void setOutputValue(int index,
          double value)
```

Sets the SI value given on the output variable at the given index.

Parameters:

`index` - the integer index into the output array.
`value` - the SI value being set on the output variable.

---

getOutputAt

```
public UserDefinedVariable getOutputAt(int index)
```

Gets the UserDefinedVariable from the output array at a given index.

Parameters:

index - the integer index into the output array.
the - UserDefinedVariable stored at the given index.

---

setOutputValue

```
public void setOutputValue(String name,
          double value)
```

Sets the SI value on on the output with the given name.

Parameters:

name - the name of the output variable.
value - the SI value to be set.

---

getInputCount

```
public int getInputCount()
```

Gets the number of input variables.

Returns:

the size of the input array.

---

getInputIdAt

```
public int getInputIdAt(int index)
```

Gets the ident of the input varible at the given index.

Parameters:

index - the integer index into the input array.
the - ident of the UserDefinedValue in the input array.

---

setInputIdAt

```
public void setInputIdAt(int index,
          int id)
```

Sets the ident of the UserDefinedValue to be stored inside the input array at a given index.

Parameters:

index - the integer index into the input array.
id - of the UserDefinedVariable to be set in the array.

---

addInputId

```
public void addInputId(int id)
```

Adds the UserDefinedValue with the given ident into the input array, if the value is not already in the input array.

Parameters:

id - the ident of the UserDefinedValue to be added.

---

clearInputs

`public void clearInputs()`

Removes all the entries in the input array

---

removeInputAt

`public void removeInputAt(int index)`

Removes the input at the given index.

Parameters:
`index` - the index into the input array.

---

getInputValue

`public double getInputValue(int index)`

Returns the current value of the input at a given index.

Parameters:
`index` - the integer index into the input array.

---

getInputAt

`public UserDefinedValue getInputAt(int index)`

Returns the UserDefinedValue at the given index in the input array.

Parameters:
`index` - the integer index into the input array.

Returns:
the UserDefinedValue at the index.

---

getInputValue

`public double getInputValue(String name)`

Returns the current value of the input variable with the given name. If no input variable with that name is found, this returns `Real.Unknown`

Parameters:
`name` - the String containing the name of the input variable.

---

getFunction

`public FunctionSource getFunction()`

Returns the current python code for this function.

Returns:
the String containing the python code.

setFunction

```
public void setFunction(FunctionSource function)
```

Sets the python code for this function.

Parameters:

`function` - the String containing the python code.

toString

```
public String toString()
```

label

```
public String label()
```

Returns a String suitable for describing the component type on a dialog.

This default implementation returns the class name.

execute

```
public boolean execute()
```

Executes the python code in this function.

Returns:

the success or failure of this execution.

findConstant

```
public static UserDefinedConstant findConstant(AbstractModel model,
        String name)
```

Finds the UserDefinedConstant in the given model with the given name.

Parameters:

`model` - the AbstractModel that contains the constant.
`the` - String name of the constant.

findVariable

```
public static UserDefinedVariable findVariable(AbstractModel model,
        String name)
```

Finds the UserDefinedVariable in the given model with the given name.

Parameters:

`model` - the AbstractModel that contains the variable.
`the` - String name of the variable.

findFunction

```
public static UserDefinedFunction findFunction(AbstractModel model,
        int ident)
```

Finds the UserDefinedFunction in the given model with the given name.

Parameters:

`model` - the AbstractModel that contains the function.
`the` - String name of the function.

getPythonInit

```
public static String getPythonInit()
```

Gets the python initialization string.

Returns:

the String containing the results of the python initialziation.

getPythonHeader

```
public String getPythonHeader()
```

Returns the header that is prepended to the python code before execution.

Returns:

the Sttring that preceeds the user code

validate

```
public void validate()
```

Determines this object's general data state by examining it's internal data. The data state is used to color code objects so the user can see the current state. The `validate` method will set the state attribute to one of these values.

Note that this state is not inclusive of the checks in `isOkayForExport` and is used only to valdiate data required for the visual representation.

Derivatives should call super.validate() before determining thier own data state. This implementation sets this object's data state to `DATA_COMPLETE` without any analysis.

complete

```
public void complete()
```

This completes this object's initialization in response to it's creation from a UI event.

This method should be overriden by subclasses that require user input or default values for a newly created component.

com.cafean.client.analysis.numerics
Class UserDefinedValue

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.AbstractComponent
         |
         +-com.cafean.client.analysis.numerics.UserDefinedValue
```

All Implemented Interfaces:
IdentHolder, StateEditable, Cloneable, Checkable, ComponentElement, Cloneable

Direct Known Subclasses:
UserDefinedConstant, UserDefinedVariable

---

public abstract class **UserDefinedValue**
extends AbstractComponent

A UserDefinedValue is a numeric component that may be referenced by a Real value inside an AbstractModel. UserDefinedValues are rendered by DrawnUserValues, and have to contain functions for getting the current value, and setting the units.

Fields inherited from class com.cafean.client.analysis.AbstractComponent
leftDiffComponent, leftDiffName, leftShortDiffName, rightDiffComponent, rightDiffName, rightShortDiffName

Fields inherited from class com.cafean.client.analysis.GenericObject
DATA_COMPLETE, DATA_ERROR, DATA_INCOMPLETE, DATA_WARNING

Fields inherited from interface javax.swing.undo.StateEditable
RCSID

Constructor Summary

public	UserDefinedValue() Creates a new instance of UserDefinedValue
public	UserDefinedValue(AbstractModel model, int number) Creates a new instance of a UserDefinedValue inside a model with a specified component number.

Method Summary

DrawnComponent	createDrawnComponent() This creates a new DrawnUserValue to render this UserDefinedValue inside a view.
void	fireComponentDeleted()

abstract Real	getCurrentValue() This gets the current value for this user defined value.
String	getUniqueID()
Real	getUnits() Returns a clone of the current value.
void	popupDataDialog(java.awt.Window parent, boolean modal)
static String	replaceNumericNames(String token, AbstractModel model) Replaces named references to user defined constants in the given token with the values of those constants.
void	restoreState(Hashtable table)
void	setName(String val_)
abstract void	setUnits(Real units) Sets the value inside this numeric to use the same units as the given Real.
void	setUnits(String units) Sets the units on this numeric based on the SI units given

Methods inherited from class com.cafean.client.analysis.AbstractComponent

addALDocRef, addComponentListener, addConnection, addMessage, addMessage, addMessage, addToModel, addToModel, canConnectTo, clearConnections, clone, complete, connectTo, connectTo, copy, createDrawnComponent, createSourceData, createTargetData, DBtypeCode, disconnect, disconnectFrom, fireComponentChanged, fireComponentChanged, fireComponentConnected, fireComponentDeleted, fireComponentDisconnected, getALDocDisplayName, getALDocRefs, getALDocRefs, getALDocShortNames, getALDocShortNames, getCatCCComparator, getCategory, getCCNumberComparator, getComponent, getComponentDependencies, getConnectionCount, getConnectionName, getConnections, getConnectionTypes, getCustomPopupActions, getCustomPopupItems, getGroupedConnections, getModel, getName, getNewCompIdent, getOrder, getOrderComparator, getOwner, getRealSize, getSharedComponents, getUniqueID, hasALDocRefs, includeInLoopcheck, isOkayForExport, isOkayForExport, label, popupDataDialog, rebuildConnections, reconnectIdentReferences, reconnectImage, removeALDocRef, removeComponentListener, removeFromModel, removeVerify, restoreALRefState, restoreState, setALDocRefs, setComponentNumber, setDeleted, setModel, setOrder, storeALRefState, toShortString, toString, updateVersion, userDelete, writeName

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder

reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ComponentElement

getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement

getModel

Methods inherited from interface com.cafean.client.analysis.Checkable

isOkayForExport, isOkayForExport

Constructors

UserDefinedValue

public **UserDefinedValue**()

Creates a new instance of UserDefinedValue

UserDefinedValue

```
public UserDefinedValue(AbstractModel model,
                        int number)
```

Creates a new instance of a UserDefinedValue inside a model with a specified component number.

Parameters:

`model` - the AbstractModel.
`number` - the component number for this function.

Methods

getCurrentValue

```
public abstract Real getCurrentValue()
```

This gets the current value for this user defined value.

Returns:

the Real with the current value.

---

createDrawnComponent

```
public DrawnComponent createDrawnComponent()
```

This creates a new DrawnUserValue to render this UserDefinedValue inside a view.

Returns:

the DrawnUserValue for this numeric.

---

setUnits

```
public abstract void setUnits(Real units)
```

Sets the value inside this numeric to use the same units as the given Real.

Parameters:

`the` - Real containing the new units.

---

setUnits

```
public void setUnits(String units)
```

Sets the units on this numeric based on the SI units given

Parameters:

`units` - the SI units to be set.

---

getUnits

```
public Real getUnits()
```

Returns a clone of the current value.

popupDataDialog

```
public void popupDataDialog(java.awt.Window parent,
        boolean modal)
```

Creates a new bean editing dialog for this object or resets and refreshes this object's current editing dialog.

fireComponentDeleted

```
public void fireComponentDeleted()
```

This calls the component deleted function on all of the listeners currently listening to this abstract component

restoreState

```
public void restoreState(Hashtable table)
```

Restore the state of the bean from an earlier edit by using copyFrom on the previously stored clone.

setName

```
public void setName(String val_)
```

Sets this object's name to the given name after trimming it.

replaceNumericNames

```
public static String replaceNumericNames(String token,
        AbstractModel model)
```

Replaces named references to user defined constants in the given token with the values of those constants.

Parameters:

`token` - a String containing a single line to have numerics replaced in
`model` - the AbstractModel containing the numerics to check

getUniqueID

```
public String getUniqueID()
```

Provides a unique component id for components who may not contain a component number.

com.cafean.client.analysis.numerics
Class UserDefinedVariable

```
java.lang.Object
   |
   +-com.cafean.client.analysis.GenericObject
      |
      +-com.cafean.client.analysis.AbstractComponent
         |
         +-com.cafean.client.analysis.numerics.UserDefinedValue
            |
            +-com.cafean.client.analysis.numerics.UserDefinedVariable
```

All Implemented Interfaces:
PropertyController, IdentHolder, StateEditable, Cloneable, Checkable, ComponentElement, Cloneable

---

public class **UserDefinedVariable**
extends UserDefinedValue
implements Cloneable, ComponentElement, Checkable, Cloneable, StateEditable, IdentHolder, PropertyController

The UserDefinedVariable is a UserDefinedValue that can have it's current value set as the output of a `UserDefinedFunction`. As a UserDefinedValue it can be used as input for `functions`, or as the value of a `Real`

Fields inherited from class com.cafean.client.analysis.AbstractComponent
leftDiffComponent, leftDiffName, leftShortDiffName, rightDiffComponent, rightDiffName, rightShortDiffName

Fields inherited from class com.cafean.client.analysis.GenericObject
DATA_COMPLETE, DATA_ERROR, DATA_INCOMPLETE, DATA_WARNING

Fields inherited from interface javax.swing.undo.StateEditable
RCSID

Fields inherited from interface com.cafean.client.ui.beans.PropertyController
ALL, COLOR_OPTIONAL, DISABLED, NONE, OPTIONAL, REQUIRED

Constructor Summary

public	UserDefinedVariable() Creates a new instance of UserDefinedVariable.
public	UserDefinedVariable(AbstractModel model, int componentNumber) Creates a new instance of UserDefinedVariable inside a model with a specified component number.

Method Summary

Object	clone()
void	complete()
void	copyFrom(GenericObject object)
int	getAttributeIndex(String property) Returns a relative index that can be used to order property lists.
Category	getCategory() The category for a UserDefinedVariable is always CAT_VARIABLE
Real	getCurrentValue()
Real	getUnits()
boolean	isPropertyActive(String propertyName) Returns false if this object has a property with the given name that is considered inactive; otherwise true.
boolean	isPropertyEnabled(String propertyName) Returns false if this object has a property with the given name that has dependency code that fails;true otherwise
boolean	isPropertyRequired(String propertyName) Returns false if this object has a property with the given name that has requirement code that fails; true otherwise.
boolean	isPropertyResizable(String propertyName) Returns false if this object has an array property with the given name that should not normally be resizable.
boolean	isPropertyRestartEditable(String propertyName) Returns true if this object has a property with the given name that should be editable during a restart edit; false otherwise.
boolean	isRestartResizable(String propertyName) Returns false if this object has an array property with the given name that should not be resizable while editing a restart.
String	label()
void	restoreState(Hashtable state)
void	setCurrentValue(Real value) Sets the current value inside this variable.
void	setCurrentValueDouble(double siValue) Sets the current value inside this variable by setting its SI value.
void	setUnits(Real units)

void	storeState(Hashtable state)
String	toString()

Methods inherited from class com.cafean.client.analysis.numerics.UserDefinedValue

createDrawnComponent, fireComponentDeleted, getCurrentValue, getUniqueID, getUnits, popupDataDialog, replaceNumericNames, restoreState, setName, setUnits, setUnits

Methods inherited from class com.cafean.client.analysis.AbstractComponent

addALDocRef, addComponentListener, addConnection, addMessage, addMessage, addMessage, addToModel, addToModel, canConnectTo, clearConnections, clone, complete, connectTo, connectTo, copy, createDrawnComponent, createSourceData, createTargetData, DBtypeCode, disconnect, disconnectFrom, fireComponentChanged, fireComponentChanged, fireComponentConnected, fireComponentDeleted, fireComponentDisconnected, getALDocDisplayName, getALDocRefs, getALDocRefs, getALDocShortNames, getALDocShortNames, getCatCCComparator, getCategory, getCCNumberComparator, getComponent, getComponentDependencies, getConnectionCount, getConnectionName, getConnections, getConnectionTypes, getCustomPopupActions, getCustomPopupItems, getGroupedConnections, getModel, getName, getNewCompIdent, getOrder, getOrderComparator, getOwner, getRealSize, getSharedComponents, getUniqueID, hasALDocRefs, includeInLoopcheck, isOkayForExport, isOkayForExport, label, popupDataDialog, rebuildConnections, reconnectIdentReferences, reconnectImage, removeALDocRef, removeComponentListener, removeFromModel, removeVerify, restoreALRefState, restoreState, setALDocRefs, setComponentNumber, setDeleted, setModel, setOrder, storeALRefState, toShortString, toString, updateVersion, userDelete, writeName

Methods inherited from class com.cafean.client.analysis.GenericObject

addComment, addMultipleComments, checkRealArrayList, checkRealArrayTable, clearDbIds, clone, closeAllViews, compareTo, copyFrom, createDataPages, debug, deleteAllComments, deleteComment, equals, fixme, getCCnumber, getComment, getComments, getComments, getComponentCCNumber, getComponentNumber, getDataState, getDB_ID, getDescription, getIdent, getMajorCreationVersion, getMajorVersion, getMinorCreationVersion, getMinorVersion, getName, getNewCompIdent, getNumComments, isDeleted, popupDataDialog, popupDataDialog, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, rangeCheck, reconnectIdentReferences, restoreState, restoreState, setComments, setComments, setComponentNumber, setCreationVersion, setDataState, setDB_ID, setDeleted, setDescription, setIdent, setMajorCreationVersion, setMajorVersion, setMinorCreationVersion, setMinorVersion, setName, showComment, storeState, storeState, trace, updateVersion, validate, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeArrayLoadValue, writeMuxLoadArray, writeMuxLoadArray, writeSP, writeSP

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface javax.swing.undo.StateEditable

restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.IdentHolder
reconnectIdentReferences

Methods inherited from interface com.cafean.client.analysis.ComponentElement
getComponent, getOwner

Methods inherited from interface com.cafean.client.analysis.ModelElement
getModel

Methods inherited from interface com.cafean.client.analysis.Checkable
isOkayForExport, isOkayForExport

Methods inherited from interface com.cafean.client.ui.beans.PropertyController
getAttributeIndex, isPropertyActive, isPropertyEnabled, isPropertyRequired, isPropertyResizable, isPropertyRestartEditable, isRestartResizable

Constructors

UserDefinedVariable

```
public UserDefinedVariable()
```

Creates a new instance of UserDefinedVariable.

UserDefinedVariable

```
public UserDefinedVariable(AbstractModel model,
                           int componentNumber)
```

Creates a new instance of UserDefinedVariable inside a model with a specified component number.

Parameters:

`model` - the AbstractModel.
`number` - the component number for this function.

Methods

copyFrom

```
public void copyFrom(GenericObject object)
```

Copy the attributes from a source object to this instance.

This is used for copying data from an edited working copy into the original object. Most notably, it is used to enable undo/redo for the legacy beanless ModelEditor architecture as well as for importing new altered data from restart decks for some plugins.

This should call copyFrom on children that support it.

Note: Never call copyFrom from clone or copy!

---

clone

```
public Object clone()
```

Creates and returns a copy of this object.

---

getCurrentValue

```
public Real getCurrentValue()
```

This gets the current value for this user defined value.

---

setCurrentValue

```
public void setCurrentValue(Real value)
```

Sets the current value inside this variable.

Parameters:

`value` - The Real containing the new value for this variable.

---

setCurrentValueDouble

```
public void setCurrentValueDouble(double siValue)
```

Sets the current value inside this variable by setting its SI value.

Parameters:

`siValue` - the double SI value.

---

getCategory

```
public Category getCategory()
```

The category for a UserDefinedVariable is always `CAT_VARIABLE`

Returns:

the Category for this UserDefinedVariable.

---

toString

```
public String toString()
```

---

label

`public String label()`

Returns a String suitable for describing the component type on a dialog.

This default implementation returns the class name.

---

setUnits

`public void setUnits(Real units)`

Sets the value inside this numeric to use the same units as the given Real.

---

getUnits

`public Real getUnits()`

Returns a clone of the current value.

---

isPropertyEnabled

`public boolean isPropertyEnabled(String propertyName)`

Returns false if this object has a property with the given name that has dependency code that fails;true otherwise

Parameters:

`propertyName` - a String containing the property name to check

---

isPropertyRequired

`public boolean isPropertyRequired(String propertyName)`

Returns false if this object has a property with the given name that has requirement code that fails; true otherwise.

Parameters:

`propertyName` - a String containing the property name to check

---

isPropertyRestartEditable

`public boolean isPropertyRestartEditable(String propertyName)`

Returns true if this object has a property with the given name that should be editable during a restart edit; false otherwise.

Parameters:

`propertyName` - a String containing the property name to check

---

isPropertyActive

`public boolean isPropertyActive(String propertyName)`

Returns false if this object has a property with the given name that is considered inactive; otherwise true.

Parameters:

`propertyName` - a String containing the name of the property to check

Returns:
false if the property is inactive, true otherwise

---

getAttributeIndex

`public int getAttributeIndex(String property)`

Returns a relative index that can be used to order property lists.

Parameters:
`propertyName` - a String containing the property name to check

---

isPropertyResizable

`public boolean isPropertyResizable(String propertyName)`

Returns false if this object has an array property with the given name that should not normally be resizable.

Parameters:
`propertyName` - a String containing the property name to check

---

isRestartResizable

`public boolean isRestartResizable(String propertyName)`

Returns false if this object has an array property with the given name that should not be resizable while editing a restart.

Parameters:
`propertyName` - a String containing the property name to check

---

storeState

`public void storeState(Hashtable state)`

Stores the state of the object to permit undo by cloning itself and storing the clone. NOTE: If the component storing its state needs a deep copy that it's clone() method does not provide, it must override storeState to find that functionality elsewhere.

---

restoreState

`public void restoreState(Hashtable state)`

Restore the state of the bean from an earlier edit by using copyFrom on the previously stored clone.

---

complete

`public void complete()`

This completes this object's initialization in response to it's creation from a UI event.

This method should be overriden by subclasses that require user input or default values for a newly created component.

Package

com.cafean.client.classify

com.cafean.client.classify
Class Classification

```
java.lang.Object
   |
   +-com.cafean.client.classify.Classification
```

All Implemented Interfaces:
Comparable

---

public class **Classification**
extends Object
implements Comparable

An encapsulization of a ModelEditor classification level. The ModelEditor displays an indicator of the the highest classification level currently in use.
See Also:
ClassificationListener

Field Summary

public static final	NOFORN no foreign
public static final	NONE No classification required.
public static final	RESTRICTED confidential restricted data
public static final	UNCLASSIFIED specifically unclassified

Method Summary

int	compareTo(Object o) Compares this object with the specified object for order.
boolean	equals(Object o)
java.awt.Color	getBackground() Returns the color that should be used for the background of any classification indicators displaying this classification.
java.awt.Color	getForeground() Returns the color that should be used for the foreground of any classification indicators displaying this classification.
String	toString() returns the name of this classification level

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `java.lang.Comparable`

`compareTo`

Fields

NONE

`public static final com.cafean.client.classify.Classification` **`NONE`**

No classification required. Use this classification level if no classification policy (or display) is required.

UNCLASSIFIED

`public static final com.cafean.client.classify.Classification` **`UNCLASSIFIED`**

specifically unclassified

RESTRICTED

`public static final com.cafean.client.classify.Classification` **`RESTRICTED`**

confidential restricted data

NOFORN

`public static final com.cafean.client.classify.Classification` **`NOFORN`**

no foreign

Methods

equals

`public boolean` **`equals`**`(Object o)`

toString

`public String` **`toString`**`()`

returns the name of this classification level

getForeground

`public java.awt.Color` **`getForeground`**`()`

Returns the color that should be used for the foreground of any classification indicators displaying this classification.

---

getBackground

`public java.awt.Color` **`getBackground`**`()`

Returns the color that should be used for the background of any classification indicators displaying this classification.

---

compareTo

`public int` **`compareTo`**`(Object o)`

Compares this object with the specified object for order. Returns a negative integer, zero, or a positive integer as this object is less than, equal to, or greater than the specified object.

Parameters:

`o` - the object to be compared.

Returns:

a negative integer, zero, or a positive integer as this object is less than, equal to, or greater than the specified object.

Throws:

`ClassCastException` - if the specified object's type prevents it from being compared to this object.

com.cafean.client.classify
Interface ClassificationListener

All Known Implementing Classes:
ClassificationPanel

---

public interface **ClassificationListener**
extends

An interface describing a listener for changes in the ModelEditor's global classification level.
See Also:
`Classification`

Method Summary

void	`classificationChanged(Classification oldLevel, Classification newLevel)` Notifies this listener that the current application classification level has changed.

Methods

classificationChanged

```
public void classificationChanged(Classification oldLevel,
        Classification newLevel)
```

Notifies this listener that the current application classification level has changed.

See Also:
`addClassificationListener`
`removeClassificationListener`
`updateClassification`

com.cafean.client.classify
Class ClassificationPanel

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-javax.swing.JComponent
            |
            +-javax.swing.JPanel
               |
               +-com.cafean.client.classify.ClassificationPanel
```

All Implemented Interfaces:
ClassificationListener, java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, HasGetTransferHandler, java.io.Serializable, javax.accessibility.Accessible

---

public class **ClassificationPanel**
extends JPanel
implements javax.accessibility.Accessible, java.io.Serializable, HasGetTransferHandler, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, ClassificationListener

A simple ModelEditor classification level display panel.

Fields inherited from class javax.swing.JComponent
TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component
BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver
ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Constructor Summary

public	ClassificationPanel() Creates new form ClassificationPanel

Method Summary

void	addNotify()
void	classificationChanged(Classification oldLevel, Classification newLevel) Notifies this listener that the current application classification level has changed.

void	removeNotify()

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUI, getUIClassID, setUI, updateUI

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, disable, enable, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getActionForKeyStroke, getActionMap, getAlignmentX, getAlignmentY, getAncestorListeners, getAutoscrolls, getBaseline, getBaselineResizeBehavior, getBorder, getBounds, getClientProperty, getComponentPopupMenu, getConditionForKeyStroke, getDebugGraphicsOptions, getDefaultLocale, getFontMetrics, getGraphics, getHeight, getInheritsPopupMenu, getInputMap, getInputMap, getInputVerifier, getInsets, getInsets, getListeners, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getPopupLocation, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getTransferHandler, getUIClassID, getVerifyInputWhenFocusTarget, getVetoableChangeListeners, getVisibleRect, getWidth, getX, getY, grabFocus, isDoubleBuffered, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingForPrint, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, print, printAll, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, requestFocus, requestFocusInWindow, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setActionMap, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setComponentPopupMenu, setDebugGraphicsOptions, setDefaultLocale, setDoubleBuffered, setEnabled, setFocusTraversalKeys, setFont, setForeground, setInheritsPopupMenu, setInputMap, setInputVerifier, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setTransferHandler, setVerifyInputWhenFocusTarget, setVisible, unregisterKeyboardAction, update, updateUI

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class java.awt.Component

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Methods inherited from interface javax.swing.TransferHandler.HasGetTransferHandler

getTransferHandler

Methods inherited from interface javax.accessibility.Accessible
getAccessibleContext

Methods inherited from interface com.cafean.client.classify.ClassificationListener
classificationChanged

Constructors

ClassificationPanel

```
public ClassificationPanel()
```

Creates new form ClassificationPanel

Methods

addNotify

```
public void addNotify()
```

removeNotify

```
public void removeNotify()
```

classificationChanged

```
public void classificationChanged(Classification oldLevel,
        Classification newLevel)
```

Notifies this listener that the current application classification level has changed.

Package
com.cafean.client.event

This package contains a set of event objects used to notify listeners of various AbstractComponent related events.

com.cafean.client.event
Class ArrayChangedEvent

```
java.lang.Object
   |
   +-java.util.EventObject
      |
      +-com.cafean.client.event.ComponentChangedEvent
         |
         +-com.cafean.client.event.ArrayChangedEvent
```

All Implemented Interfaces:
java.io.Serializable

Direct Known Subclasses:
ElementsRemovedEvent, ElementsAddedEvent

---

public class **ArrayChangedEvent**
extends ComponentChangedEvent

An event object describing a change to an array contained in a component.

Constructor Summary

public	`ArrayChangedEvent(AbstractComponent component, Object[] oldArr, Object[] newArr)` Creates a new array changed event for the given component with the given old and new array references.

Method Summary

`Object[]`	`getNewArray()` Retrieves the new array reference.
`Object[]`	`getOldArray()` Retrieves the old array reference.

Methods inherited from class `com.cafean.client.event.ComponentChangedEvent`

`getComponent`

Methods inherited from class `java.util.EventObject`

`getSource, toString`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

ArrayChangedEvent

```
public ArrayChangedEvent(AbstractComponent component,
                         Object[] oldArr,
                         Object[] newArr)
```

Creates a new array changed event for the given component with the given old and new array references.

Parameters:

`component` - the AbstractComponent that has changed
`oldArr` - an Object[] reference to the original array
`newArr` - an Object[] reference to the newly changed array

Methods

getNewArray

```
public Object[] getNewArray()
```

Retrieves the new array reference. Note: The old and new array references may be the same array and may be the same or different sizes.

---

getOldArray

```
public Object[] getOldArray()
```

Retrieves the old array reference. Note: The old and new array references may be the same array and may be the same or different sizes.

com.cafean.client.event
Class ComponentChangedEvent

```
java.lang.Object
   |
   +-java.util.EventObject
      |
      +-com.cafean.client.event.ComponentChangedEvent
```

All Implemented Interfaces:
java.io.Serializable

Direct Known Subclasses:
ArrayChangedEvent

---

public class **ComponentChangedEvent**
extends EventObject

An event object describing a general change to an AbstractComponent. Derivative classes can and should describe a more specific change.

Constructor Summary

public	`ComponentChangedEvent(AbstractComponent component)` Creates a new instance of ComponentChangedEvent

Method Summary

`AbstractComponent`	`getComponent()` retrieves the component whos change is described by this event

Methods inherited from class `java.util.EventObject`

`getSource, toString`

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructors

ComponentChangedEvent

`public` **`ComponentChangedEvent`**`(AbstractComponent component)`

Creates a new instance of ComponentChangedEvent

Methods

getComponent

`public AbstractComponent` **`getComponent`**`()`

retrieves the component whos change is described by this event

com.cafean.client.event
Class ElementsAddedEvent

```
java.lang.Object
   |
   +-java.util.EventObject
      |
      +-com.cafean.client.event.ComponentChangedEvent
         |
         +-com.cafean.client.event.ArrayChangedEvent
            |
            +-com.cafean.client.event.ElementsAddedEvent
```

All Implemented Interfaces:
java.io.Serializable

---

public class **ElementsAddedEvent**
extends ArrayChangedEvent

An event object describing a change to an array contained in a component.

Constructor Summary

public	ElementsAddedEvent(AbstractComponent component, Object[] oldArr, Object[] newArr, int[] indexes) Creates a new event describing the addition of elements to an array in a particular component.

Method Summary

int[]	getIndexes() Retrieves the indexes (in the old array), of the elements that were removed.

Methods inherited from class com.cafean.client.event.ArrayChangedEvent

getNewArray, getOldArray

Methods inherited from class com.cafean.client.event.ComponentChangedEvent

getComponent

Methods inherited from class java.util.EventObject

getSource, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

---

Constructors

ElementsAddedEvent

```
public ElementsAddedEvent(AbstractComponent component,
                          Object[] oldArr,
                          Object[] newArr,
                          int[] indexes)
```

Creates a new event describing the addition of elements to an array in a particular component.

Parameters:

`component` - the AbstractComponent that has changed
`oldArr` - an Object[] reference to the original array
`newArr` - an Object[] reference to the newly changed array

Methods

getIndexes

```
public int[] getIndexes()
```

Retrieves the indexes (in the old array), of the elements that were removed.

com.cafean.client.event
Class ElementsRemovedEvent

```
java.lang.Object
   |
   +-java.util.EventObject
       |
       +-com.cafean.client.event.ComponentChangedEvent
           |
           +-com.cafean.client.event.ArrayChangedEvent
               |
               +-com.cafean.client.event.ElementsRemovedEvent
```

All Implemented Interfaces:
java.io.Serializable

---

public class **ElementsRemovedEvent**
extends ArrayChangedEvent

An event object describing the removal of elements from an array in a given component.

Constructor Summary

public	ElementsRemovedEvent(AbstractComponent component, Object[] oldArr, Object[] newArr, int[] indexes) Creates a new event describing the removal of elements from an array in a particular component.

Method Summary

int[]	getIndexes() Retrieves the indexes (in the old array), of the elements that were removed.

Methods inherited from class com.cafean.client.event.ArrayChangedEvent

getNewArray, getOldArray

Methods inherited from class com.cafean.client.event.ComponentChangedEvent

getComponent

Methods inherited from class java.util.EventObject

getSource, toString

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

---

Constructors

ElementsRemovedEvent

```
public ElementsRemovedEvent(AbstractComponent component,
                            Object[] oldArr,
                            Object[] newArr,
                            int[] indexes)
```

Creates a new event describing the removal of elements from an array in a particular component.

Parameters:

`component` - the AbstractComponent that has changed
`oldArr` - an Object[] reference to the original array
`newArr` - an Object[] reference to the newly changed array

Methods

getIndexes

```
public int[] getIndexes()
```

Retrieves the indexes (in the old array), of the elements that were removed.

Package
com.cafean.client.io

Provides the several utility classes to assist in loading and storing ModelEditor models.

Important classes to note are:

- `MEDReader`
-

  Contains a set of static methods used directly to store and load PIB blocks for a given model.

- `Writeable`
-

  An interface describing objects that have an ASCII representation that can be *written* to a `java.io.PrintWriter` for display or export.

com.cafean.client.io
Class MEDReader

```
java.lang.Object
   |
   +-com.cafean.client.io.MEDReader
```

---

public class **MEDReader**
extends Object

This file contains the necessary operations for reading in ModelEditor PibBlocks. These blocks are generic across plugins, and can be read and written using static methods in this class.

Constructor Summary

public	MEDReader()

Method Summary

static MedVersionInfoRec	buildInfoRecord(AbstractModel model)
static UserDefinedConstant	loadUserConstant(UserConstantRec rec, AbstractModel model) Loads the given UserConstantRec into a UserDefinedConstant object created in the given model.
static UserDefinedFunction	loadUserFunction(UserFunctionRec rec, AbstractModel model) Loads the given UserFunctionRec into a UserDefinedFunction object created in the given model.
static UserDefinedVariable	loadUserVariable(UserVariableRec rec, AbstractModel model) Loads the given UserVariableRec into a UserDefinedVariable object created in the given model.
static void	loadVisualComponents(Vector drawingBlocks, Vector viewBlocks, AbstractModel model) Loads the ViewComponent, DrawnComponent and DrawnAnnotation records from the given Vectors.
static com.apt.xdr.PibBlock	readDrawingBlock(com.apt.xdr.PibFile file, String blockname, int[] blockparm) Reads a PibBlock of the required type from the given PibFile if blockname specified is one that is handled by the MEDReader.
static UserConstantRec	storeUserConstant(UserDefinedConstant var) Stores the given UserDefinedConstant into a UserConstantRec object for use in saving it into a PIB formatted file.
static UserFunctionRec	storeUserFunction(UserDefinedFunction var) Stores the given UserDefinedFunction into a UserFunctionRec object for use in saving it into a PIB formatted file.

static UserVariableRec	storeUserVariable(UserDefinedVariable var) Stores the given UserDefinedVariable into a UserVariableRec object for use in saving it into a PIB formatted file.
static boolean	verifyVersion(com.apt.xdr.PibFile file)

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructors

MEDReader

public **MEDReader**()

Methods

loadVisualComponents

public static void **loadVisualComponents**(Vector drawingBlocks,
Vector viewBlocks,
AbstractModel model)

Loads the ViewComponent, DrawnComponent and DrawnAnnotation records from the given Vectors. Plugin provided drawing types must be loaded after this method is called and added to the ViewComponent at that time.

Parameters:

drawingBlocks - a Vector containing the PibBlock derivative drawing blocks to load and add to the appropriate view.
viewBlocks - a Vector containing the ViewCompRec's to create views for.
model - the AbstractModel to add the loaded views to.

See Also:

ViewComponent.addPibBlock(PibBlock)

readDrawingBlock

public static com.apt.xdr.PibBlock **readDrawingBlock**(com.apt.xdr.PibFile file,
String blockname,
int[] blockparm)

Reads a PibBlock of the required type from the given PibFile if blockname specified is one that is handled by the MEDReader.

storeUserVariable

public static UserVariableRec **storeUserVariable**(UserDefinedVariable var)

Stores the given UserDefinedVariable into a UserVariableRec object for use in saving it into a PIB formatted file.

loadUserVariable

```
public static UserDefinedVariable loadUserVariable(UserVariableRec rec,
        AbstractModel model)
```

Loads the given UserVariableRec into a UserDefinedVariable object created in the given model. The loaded variable must be added to the model before use.

---

storeUserConstant

```
public static UserConstantRec storeUserConstant(UserDefinedConstant var)
```

Stores the given UserDefinedConstant into a UserConstantRec object for use in saving it into a PIB formatted file.

---

loadUserConstant

```
public static UserDefinedConstant loadUserConstant(UserConstantRec rec,
        AbstractModel model)
```

Loads the given UserConstantRec into a UserDefinedConstant object created in the given model. The loaded variable must be added to the model before use.

---

storeUserFunction

```
public static UserFunctionRec storeUserFunction(UserDefinedFunction var)
```

Stores the given UserDefinedFunction into a UserFunctionRec object for use in saving it into a PIB formatted file.

---

loadUserFunction

```
public static UserDefinedFunction loadUserFunction(UserFunctionRec rec,
        AbstractModel model)
```

Loads the given UserFunctionRec into a UserDefinedFunction object created in the given model. The loaded variable must be added to the model before use.

---

verifyVersion

```
public static boolean verifyVersion(com.apt.xdr.PibFile file)
```

---

buildInfoRecord

```
public static MedVersionInfoRec buildInfoRecord(AbstractModel model)
```

com.cafean.client.io
Interface Writeable

public interface **Writeable**
extends

Interface implemented by components that know how to write themselves.

Method Summary

void	write(java.io.PrintWriter out) Write the ascii output for the component to the given PrintWriter.

Methods

write

```
public void write(java.io.PrintWriter out)
```

Write the ascii output for the component to the given PrintWriter.

Parameters:

`out` - the PrintWriter to write this object to.

Package
com.cafean.client.ui

Provides the basic user interface classes for the ModelEditor.

Classes of note to plugin writers are:

- `ComponentSelector` - A selection dialog for components in a model
- `DrawnComponent` - A renderer for an `AbstractComponent`
- `MainFrame` - The central main class for the ModelEditor.
- `NamedValueSelector` - A selector for values with a related string.
- `RealTextField` - a JTextField that is specialized for working with `Real` values
- `TableSorter` - A sorter utility for JTable's.

com.cafean.client.ui
Class AsciiViewer

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-java.awt.Window
            |
            +-java.awt.Dialog
               |
               +-javax.swing.JDialog
                  |
                  +-com.cafean.client.ui.AsciiViewer
```

All Implemented Interfaces:
RefreshableDialog, java.awt.event.MouseListener, ComponentListener, java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, javax.accessibility.Accessible, HasGetTransferHandler, RootPaneContainer, javax.accessibility.Accessible, WindowConstants

---

public class **AsciiViewer**
extends JDialog
implements WindowConstants, javax.accessibility.Accessible, RootPaneContainer, HasGetTransferHandler, javax.accessibility.Accessible, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, ComponentListener, java.awt.event.MouseListener, RefreshableDialog

A viewer for the ASCII export of a `Writeable` component.
See Also:
`MECodePlugin.getAsciiStyledDocument(Writeable)`

Nested Class Summary

class	`AsciiViewer.Updater` AsciiViewer.Updater

Fields inherited from class `java.awt.Dialog`

DEFAULT_MODALITY_TYPE

Fields inherited from class `java.awt.Component`

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface `java.awt.image.ImageObserver`

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Fields inherited from interface `javax.swing.WindowConstants`

DISPOSE_ON_CLOSE, DO_NOTHING_ON_CLOSE, EXIT_ON_CLOSE, HIDE_ON_CLOSE

Constructor Summary

public	AsciiViewer(java.awt.Frame parent, Writeable comp, MECodePlugin plugin) Creates new viewer to show the given Writeable component with the given Document.
public	AsciiViewer(java.awt.Dialog parent, Writeable comp, MECodePlugin plugin) Creates new viewer to show the given Writeable component with the given Document.
public	AsciiViewer(java.awt.Frame parent, Writeable comp, StyledDocument doc) Creates new viewer to show the given Writeable component with the given Document.
public	AsciiViewer(java.awt.Dialog parent, Writeable comp, StyledDocument doc) Creates new viewer to show the given Writeable component with the given Document.

Method Summary

void	componentChanged(ComponentChangedEvent evt)
void	componentConnected(Connection con)
void	componentDeleted()
void	componentDisconnected(Connection con)
void	createPopupMenu(int x, int y)
Writeable	getWriteable() Retrieves the object that this viewer is viewing.
void	mouseClicked(java.awt.event.MouseEvent e)
void	mouseEntered(java.awt.event.MouseEvent e)
void	mouseExited(java.awt.event.MouseEvent e)
void	mousePressed(java.awt.event.MouseEvent e)
void	mouseReleased(java.awt.event.MouseEvent e)
void	refresh() Updates this AsciiViewer's data
void	setTargetComponent(AbstractComponent comp) Sets the listening component for this viewer.
void	setVisible(boolean visible)
void	unitsChanged()

Methods inherited from class `javax.swing.JDialog`

getAccessibleContext, getContentPane, getDefaultCloseOperation, getGlassPane, getGraphics, getJMenuBar, getLayeredPane, getRootPane, getTransferHandler, isDefaultLookAndFeelDecorated, remove, repaint, setContentPane, setDefaultCloseOperation, setDefaultLookAndFeelDecorated, setGlassPane, setJMenuBar, setLayeredPane, setLayout, setTransferHandler, update

Methods inherited from class `java.awt.Dialog`

addNotify, getAccessibleContext, getModalityType, getTitle, hide, isModal, isResizable, isUndecorated, setModal, setModalityType, setResizable, setTitle, setUndecorated, setVisible, show, toBack

Methods inherited from class `java.awt.Window`

addNotify, addPropertyChangeListener, addPropertyChangeListener, addWindowFocusListener, addWindowListener, addWindowStateListener, applyResourceBundle, applyResourceBundle, createBufferStrategy, createBufferStrategy, dispose, getAccessibleContext, getBufferStrategy, getFocusableWindowState, getFocusCycleRootAncestor, getFocusOwner, getFocusTraversalKeys, getGraphicsConfiguration, getIconImages, getInputContext, getListeners, getLocale, getModalExclusionType, getMostRecentFocusOwner, getOwnedWindows, getOwner, getOwnerlessWindows, getToolkit, getWarningString, getWindowFocusListeners, getWindowListeners, getWindows, getWindowStateListeners, hide, isActive, isAlwaysOnTop, isAlwaysOnTopSupported, isFocusableWindow, isFocusCycleRoot, isFocused, isLocationByPlatform, isShowing, pack, postEvent, removeNotify, removeWindowFocusListener, removeWindowListener, removeWindowStateListener, reshape, setAlwaysOnTop, setBounds, setBounds, setCursor, setFocusableWindowState, setFocusCycleRoot, setIconImage, setIconImages, setLocationByPlatform, setLocationRelativeTo, setMinimumSize, setModalExclusionType, setSize, setSize, setVisible, show, toBack, toFront

Methods inherited from class `java.awt.Container`

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class `java.awt.Component`

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.awt.image.ImageObserver`

imageUpdate

Methods inherited from interface `java.awt.MenuContainer`

getFont, postEvent, remove

Methods inherited from interface `javax.accessibility.Accessible`

getAccessibleContext

Methods inherited from interface javax.accessibility.Accessible
getAccessibleContext

Methods inherited from interface javax.swing.RootPaneContainer
getContentPane, getGlassPane, getLayeredPane, getRootPane, setContentPane, setGlassPane, setLayeredPane

Methods inherited from interface javax.swing.TransferHandler.HasGetTransferHandler
getTransferHandler

Methods inherited from interface com.cafean.client.analysis.ComponentListener
componentChanged, componentConnected, componentDeleted, componentDisconnected

Methods inherited from interface java.awt.event.MouseListener
mouseClicked, mouseEntered, mouseExited, mousePressed, mouseReleased

Methods inherited from interface com.cafean.client.ui.RefreshableDialog
refresh, unitsChanged

Constructors

AsciiViewer

```
public AsciiViewer(java.awt.Frame parent,
                   Writeable comp,
                   MECodePlugin plugin)
```

Creates new viewer to show the given Writeable component with the given Document.

AsciiViewer

```
public AsciiViewer(java.awt.Dialog parent,
                   Writeable comp,
                   MECodePlugin plugin)
```

Creates new viewer to show the given Writeable component with the given Document.

AsciiViewer

```
public AsciiViewer(java.awt.Frame parent,
                   Writeable comp,
                   StyledDocument doc)
```

Creates new viewer to show the given Writeable component with the given Document.

AsciiViewer

```
public AsciiViewer(java.awt.Dialog parent,
                   Writeable comp,
                   StyledDocument doc)
```

Creates new viewer to show the given Writeable component with the given Document.

Methods

setTargetComponent

```
public void setTargetComponent(AbstractComponent comp)
```

Sets the listening component for this viewer. This viewer will be added as a component listener to the given component and will be registered for refresh events.

Parameters:

comp - the AbstractComponent target for this viewer

getWriteable

```
public Writeable getWriteable()
```

Retrieves the object that this viewer is viewing.

setVisible

```
public void setVisible(boolean visible)
```

refresh

```
public void refresh()
```

Updates this AsciiViewer's data

componentChanged

```
public void componentChanged(ComponentChangedEvent evt)
```

componentConnected

```
public void componentConnected(Connection con)
```

componentDeleted

```
public void componentDeleted()
```

componentDisconnected

```
public void componentDisconnected(Connection con)
```

createPopupMenu

```
public void createPopupMenu(int x,
         int y)
```

mouseClicked

```
public void mouseClicked(java.awt.event.MouseEvent e)
```

mouseEntered

```
public void mouseEntered(java.awt.event.MouseEvent e)
```

mouseExited

```
public void mouseExited(java.awt.event.MouseEvent e)
```

mousePressed

```
public void mousePressed(java.awt.event.MouseEvent e)
```

mouseReleased

```
public void mouseReleased(java.awt.event.MouseEvent e)
```

unitsChanged

```
public void unitsChanged()
```

com.cafean.client.ui
Class AsciiViewer.Updater

```
java.lang.Object
   |
   +-java.lang.Thread
      |
      +-com.cafean.client.ui.AsciiViewer.Updater
```

All Implemented Interfaces:
Runnable

---

public class **AsciiViewer.Updater**
extends Thread

A background updater thread for ASCII Views

Fields inherited from class `java.lang.Thread`

`MAX_PRIORITY, MIN_PRIORITY, NORM_PRIORITY`

Constructor Summary

public	`AsciiViewer.Updater()`

Method Summary

void	`disable()` Disables and deactivates this Updater.
void	`run()` Continuously updates and sleeps while enabled to update the AsciiViewer by calling `writeComponent`.
void	`update()` Requests that an update of the AsciiViewer be performed on a background thread.
void	`Updater()` Creates a new Updater as a minimum priority daemon task.

Methods inherited from class `java.lang.Thread`

`activeCount, checkAccess, countStackFrames, currentThread, destroy, dumpStack, enumerate, getAllStackTraces, getContextClassLoader, getDefaultUncaughtExceptionHandler, getId, getName, getPriority, getStackTrace, getState, getThreadGroup, getUncaughtExceptionHandler, holdsLock, interrupt, interrupted, isAlive, isDaemon, isInterrupted, join, join, join, resume, run, setContextClassLoader, setDaemon, setDefaultUncaughtExceptionHandler, setName, setPriority, setUncaughtExceptionHandler, sleep, sleep, start, stop, stop, suspend, toString, yield`

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.lang.Runnable
run

Constructors

AsciiViewer.Updater

public **AsciiViewer.Updater**()

Methods

Updater

public void **Updater**()

Creates a new Updater as a minimum priority daemon task.

---

run

public void **run**()

Continuously updates and sleeps while enabled to update the AsciiViewer by calling writeComponent.

See Also:

disable()
update()

---

disable

public void **disable**()

Disables and deactivates this Updater.

---

update

public void **update**()

Requests that an update of the AsciiViewer be performed on a background thread.

com.cafean.client.ui
Class BeanBox

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-javax.swing.JComponent
            |
            +-javax.swing.JPanel
               |
               +-com.cafean.client.ui.BeanBox
```

All Implemented Interfaces:
StateEditable, java.io.Serializable, java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, HasGetTransferHandler, java.io.Serializable, javax.accessibility.Accessible

---

public class **BeanBox**
extends JPanel
implements javax.accessibility.Accessible, java.io.Serializable, HasGetTransferHandler, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, java.io.Serializable, StateEditable

The BeanBox is the actual canvas that contains all of the components and annotations inside the ZoomablePanel of a DrawnView. All the methods that manipulate the components in a view, including selecting the components and building the popup menu for right-clicking are included here.

Fields inherited from class javax.swing.JComponent
TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component
BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver
ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Fields inherited from interface javax.swing.undo.StateEditable
RCSID

Constructor Summary

public	BeanBox(AbstractModel model) Constructs a new BeanBox.

Method Summary

java.awt.Component	add(java.awt.Component comp) Adds the specified component to this container.
void	addBoxSelectionListener(BoxSelectionListener listener) Adds the given listener to the list that is notified when the BeanBox selection set may have changed.
DrawnConnection	addDrawnConnection(Connection con) Adds a DrawnConnection object to this BeanBox for the given connection if and only if both sides of the connection are present in this view.
void	addNotify() Notifies this component that it now has a parent.
void	addSelectedComponent(Object selected) Add a single object to the selection.
void	addSelectedComponents(Object[] sel) Adds an array of beans to the current selection.
java.awt.Component	addToFront(java.awt.Component comp) Adds the specified component to this container.
void	addVisualGroup(VisualGroup group) Adds the given VisualGroup to this beanbox and assigns a new ident.
void	addVisualGroup(VisualGroup group, boolean adjustIdent) Adds the given VisualGroup to this beanbox and assigns a new ident (if necessary).
void	boundsChanged(DrawnComponent component) This is used to inform the BeanBox that the bounds of a DrawnComponent may have changed because of a component's internal values being edited.
void	calcSelectionBounds() Calculate the bounds of selected beans.
boolean	canAlign() The align operation can only be performed if two or more components are selected.
boolean	canCopy() The Copy or cut command can only be performed if there is at least one object selected.
boolean	canDelete() The delete operation can only be performed if a copy opperation could be performed.
boolean	canGroup() Returns true if the current selection can be grouped.
boolean	canPaste() The Paste command can only be executed if there are contents on the clipboard, and the Data on the clipboard is appropriate for the ModelEditor.
boolean	canUngroup() Returns true if the current selection can be ungrouped.

void	clearSelection() Clears the current selection.
void	connectionPointRemoved(DrawnComponent component) This is used to inform the BeanBox that the connecting points of a DrawnComponent may have been added or removed because of a component's internal values being edited.
boolean	copy() Serializes the current selection to the system clipboard.
JPopupMenu	createPopupMenu(java.awt.event.MouseEvent evt) Creates a popup menu suitable for this GlassPanel and the model it is viewing.
JMenu	createZoomMenu() Creates a Zoom menu suitable for this view.
void	cut() Serialize the current selection to the system clipboard, and then remove it from the BeanBox.
void	delete() Remove the current selection from the model.
void	deselectGroup(int groupID) Deselects the objects in the given group and all parent and child groups.
java.awt.Component[]	findComponentsInside(int x1, int y1, int x2, int y2) Retreive the list of components located within a rectangular region bounded by the points (x1,y1) and (x2,y2).
VisualGroup	findGroupByIdent(int ident) Retrieves the VisualGroup instance in this BeanBox that has the given ident.
VisualGroup	findGroupByOldIdent(int oldIdent) Retrieves the VisualGroup instance in this BeanBox that has the given old ident.
void	fitToWindow() sets this BeanBox's zoom such that its contents will fit in the current window size in the same way as the "Fit To Window" menu item.
Vector	getAnnotations() This gets fills a vector with all of the annotations it contains.
Vector	getComponents(boolean includeConnections) Retrieves a Vector of all the Components in this BeanBox.
DrawnComponent	getDrawnComponent(AbstractComponent comp) Retrieves the DrawnComponent rendering the given component in this BeanBox.
DrawnComponent	getDrawnComponent(int compIdent) Retrieves the DrawnComponent rendering the given component in this BeanBox.
Vector	getDrawnComponents() Retrieves a Vector of all the Components in this BeanBox, including Annotations, DrawnComponents and DrawnConnections.

java.awt.Dimension	getMaximumSize() Return the maximum size of this component.
java.awt.Dimension	getMinimumSize() Return the minimum size of this component.
AbstractModel	getModel() Getter for the AbstractModel that contains the ViewComponent this BeanBox represents.
int	getNumSelected() Returns the number of selected components.
java.awt.Dimension	getPanelSize() Return the panelSize parameter from the ZoomablePanel parent class.
java.awt.Dimension	getPreferredSize() Return the preferred size of this component.
Double	getScale() Return the scale parameter from the ZoomablePanel parent class.
java.awt.Component	getSelected(int i) Returns a selected bean at the given index
java.awt.Component[]	getSelection() Return the array of selected objects.
java.awt.Component[]	getSelection(Vector selComps) Retreive the list of selected components in the drawing order.
java.awt.Rectangle	getSelectionBounds() Return the bounds of selected beans.
ViewComponent	getViewComponent() Retrieves the ViewComponent that corresponds with the DrawnView that contains this BeanBox.
VisualGroup[]	getVisualGroups() Retrieves the list of VisualGroup instances in this BeanBox.
double	getWidthScaleFactor() Gets this view's width scale factor.
void	group() Groups the current selection as if the Group button had been pressed by the user.
boolean	isSelected(java.awt.Component component) Returns true if the given component is part of the current selection.
static java.awt.Component[]	loadComponents(AbstractModel model, com.apt.xdr.PibBlock[] blocks) Loads annotations and drawn components from the given array of PibBlock's.
static java.awt.Component[]	loadComponents(AbstractModel model, com.apt.xdr.PibBlock[] blocks, boolean loadDrawn) Loads annotations and drawn components from the given array of PibBlock's.

static VisualGroup[]	loadVisualGroups(com.apt.xdr.PibBlock[] blocks) Loads visual groups from the given array of PibBlock's.
void	minimizeView() Minimizes the current view to the minimum possible with the current components.
void	organizeSelection() Organize the selected components.
void	organizeView(Vector drawnComponents, boolean relative) Calls AbstractModel.layoutComponents with the given components.
void	organizeView(Vector drawnComponents, boolean relative, boolean includeUndo) Calls AbstractModel.layoutComponents with the given components.
void	paint(java.awt.Graphics g)
void	paintImmediately(int x, int y, int w, int h)
void	paintImmediately(JComponent component) Immediately paints the region of the (scaled) BeanBox that the given component is located.
void	paste() Attempts to paste the current clipboard contents into this view.
void	print(java.awt.Graphics g)
void	reconnectGroupIDs() Reconnects all invalid VisualGroup ID references.
void	redrawSelection() Redraw the selected components, or all components in this view if none are selected
void	refresh(java.awt.Rectangle r) This refreshses a given java.awt.Rectangle that indicates a dirty region.
void	remove(java.awt.Component comp) Removes the specified component from this container.
void	remove(java.awt.Component comp, boolean removeConnections) Removes the specified component from this container.
void	remove(int index) Removes the component, specified by index, from this container.
void	removeBoxSelectionListener(BoxSelectionListener listener) Removes the given listener from the list that is notified when the BeanBox selection set may have changed.
void	removeNotify()
void	removeSelectedComponent(Object sel) This removes the Object from the list of selected components.

void	removeVisualGroup(VisualGroup group, CompoundEdit edit) Removes the given group from this beanbox and adds an appropriate undoable edit to the given CompoundEdit.
void	renumberSelectedComponents() This takes the selected components and has the model renumber their component numbers.
void	repaint(long tm, int x, int y, int width, int height)
void	resetConnections() This resets all the connections that connect to the selected components
void	restoreState(Hashtable state) Restore the state of the bean from an earlier edit.
void	revalidate()
void	selectCategory(Category cat) This clears the current selection, and adds all the components that are of the given categories subset to the "selected" list.
void	selectGroup(int groupID) Selects the objects in the given group and all parent and child groups.
void	setBackground(java.awt.Color bg)
void	setPaintEnabled(boolean b) Enables or disables the repainting of this BeanBox.
void	setScale(Double scale) Set the scale parameter.
void	setSelectedComponent(java.awt.Component focus) Clear the current selection and add a single object to the selection.
void	showAllConnections() This goes through all of the components currently in the view, and tries to render their connections.
void	showAllConnections(boolean undoable) This goes through all of the components currently in the view, and tries to render their connections.
void	storeState(Hashtable state) Store the state of the bean to permit undo.
void	toFront(ConnectionRenderer connection) Moves the given connection renderer to the front of the drawing stack.
String	toString()
boolean	tryLockOrganize() Locks BeanBox's organizeView method so that only one organize occurs for a given view at a time, and so that it can be determined when the organize is complete.

void	ungroup() Ungroups the current selection as if the Ungroup button had been pressed by the user.
void	unlockOrganize() Unlocks orgainzeView() to be used by any thread.
void	updateComponentList(Vector components, boolean organize) Synchronizes the component list in this BeanBox to the given list of components.
void	updateSelection() Updates this BeaBox's current selection and notifies MainFrame and the main property view of the new current model and current selection.

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUI, getUIClassID, setUI, updateUI

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, disable, enable, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getActionForKeyStroke, getActionMap, getAlignmentX, getAlignmentY, getAncestorListeners, getAutoscrolls, getBaseline, getBaselineResizeBehavior, getBorder, getBounds, getClientProperty, getComponentPopupMenu, getConditionForKeyStroke, getDebugGraphicsOptions, getDefaultLocale, getFontMetrics, getGraphics, getHeight, getInheritsPopupMenu, getInputMap, getInputMap, getInputVerifier, getInsets, getInsets, getListeners, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getPopupLocation, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getTransferHandler, getUIClassID, getVerifyInputWhenFocusTarget, getVetoableChangeListeners, getVisibleRect, getWidth, getX, getY, grabFocus, isDoubleBuffered, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingForPrint, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, print, printAll, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, requestFocus, requestFocusInWindow, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setActionMap, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setComponentPopupMenu, setDebugGraphicsOptions, setDefaultLocale, setDoubleBuffered, setEnabled, setFocusTraversalKeys, setFont, setForeground, setInheritsPopupMenu, setInputMap, setInputVerifier, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setTransferHandler, setVerifyInputWhenFocusTarget, setVisible, unregisterKeyboardAction, update, updateUI

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class `java.awt.Component`

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Methods inherited from interface javax.swing.TransferHandler.HasGetTransferHandler

getTransferHandler

Methods inherited from interface javax.accessibility.Accessible
getAccessibleContext

Methods inherited from interface javax.swing.undo.StateEditable
restoreState, storeState

Constructors

BeanBox

public **BeanBox**(AbstractModel model)

Constructs a new BeanBox. Each BeanBox has a reference to the AbstractModel that contains the ViewComponent that constructed the DrawnView.

Parameters:

model - the AbstractModel.

Methods

addNotify

public void **addNotify**()

Notifies this component that it now has a parent. Overridden here to add cursor update key-listeners to the new parent

removeNotify

public void ***removeNotify***()

getModel

public AbstractModel **getModel**()

Getter for the AbstractModel that contains the ViewComponent this BeanBox represents.

Returns:

the AbstractModel.

getAnnotations

public Vector **getAnnotations**()

This gets fills a vector with all of the annotations it contains.

Returns:
the Vector containing annotations.

---

addDrawnConnection

`public DrawnConnection addDrawnConnection(Connection con)`

Adds a DrawnConnection object to this BeanBox for the given connection if and only if both sides of the connection are present in this view.

Parameters:
`con` - the Connection that is being added to the view.

Returns:
the DrawnConnection created from the Connection.

---

getDrawnComponent

`public DrawnComponent getDrawnComponent(int compIdent)`

Retrieves the DrawnComponent rendering the given component in this BeanBox.

Parameters:
`compIdent` - the `GenericObject.getIdent()` of the AbstractComponent to retrieve a DrawnComponent for.

Returns:
the DrawnComponent requested or null.

---

getDrawnComponent

`public DrawnComponent getDrawnComponent(AbstractComponent comp)`

Retrieves the DrawnComponent rendering the given component in this BeanBox.

Parameters:
`comp` - the AbstractComponent to retrieve a DrawnComponent for.

Returns:
the DrawnComponent requested or null.

---

remove

`public void remove(int index)`

Removes the component, specified by `index`, from this container.

Parameters:
`index` - the index of the component to be removed.

See Also:
`add(Component)`

---

remove

```
public void remove(java.awt.Component comp)
```

Removes the specified component from this container.

Parameters:

comp - the component to be removed

See Also:

add(Component)
Container.remove(int)

---

remove

```
public void remove(java.awt.Component comp,
        boolean removeConnections)
```

Removes the specified component from this container.

Parameters:

comp - the component to be removed
removeConnections - if true all related DrawnConnections will be removed.

See Also:

add(Component)

---

boundsChanged

```
public void boundsChanged(DrawnComponent component)
```

This is used to inform the BeanBox that the bounds of a DrawnComponent may have changed because of a component's internal values being edited. This informs all of the DrawnConnections that the component has changed shape.

Parameters:

component - the DrawnComponent whose shape has changed.

---

connectionPointRemoved

```
public void connectionPointRemoved(DrawnComponent component)
```

This is used to inform the BeanBox that the connecting points of a DrawnComponent may have been added or removed because of a component's internal values being edited. This informs all of the DrawnConnections that the component's connection points have changed.

Parameters:

component - the DrawnComponent whose shape has changed.

---

add

```
public java.awt.Component add(java.awt.Component comp)
```

Adds the specified component to this container. If the component is a ConnectionRenderer it will be prepended; otherwise it will be appended.

Components are painted in the reverse order that they exist in the BeanBox, thus ConnectionRenderer objects must be first.

Parameters:
comp - the Component to be added

Returns:
the Component argument

See Also:
Container.add(java.awt.Component)

---

addToFront

```
public java.awt.Component addToFront(java.awt.Component comp)
```

Adds the specified component to this container. If the component is a ConnectionRenderer it will be prepended; otherwise it will be appended.

Components are painted in the reverse order that they exist in the BeanBox, thus ConnectionRenderer objects must be first.

Parameters:
comp - the Component to be added

Returns:
the Component argument

See Also:
add(Component)

---

getMinimumSize

```
public java.awt.Dimension getMinimumSize()
```

Return the minimum size of this component. Calculated from the bounds of all child components.

Returns:
A dimension object indicating this component's minimum size.

---

getMaximumSize

```
public java.awt.Dimension getMaximumSize()
```

Return the maximum size of this component. The panelSize component of the zoomable panel containing this beanbox.

Returns:
A dimension object indicating this component's maximum size.

---

getPreferredSize

```
public java.awt.Dimension getPreferredSize()
```

Return the preferred size of this component. This is the panel size scaled by the current scale factor.

Returns:
A dimension object indicating this component's prefered size.

getPanelSize

```
public java.awt.Dimension getPanelSize()
```

Return the panelSize parameter from the ZoomablePanel parent class.

Returns:
the Dimension size of the ZoomablePanel.

See Also:
`ZoomablePanel.getPanelSize()`

setScale

```
public void setScale(Double scale)
```

Set the scale parameter. This property is contained in the ZoomablePanel parent class.

Parameters:
`scale` - The new value for the scale parameter.

getScale

```
public Double getScale()
```

Return the scale parameter from the ZoomablePanel parent class.

Returns:
the Double scale factor of the ZoomablePanel.

See Also:
`ZoomablePanel.getScale()`

refresh

```
public void refresh(java.awt.Rectangle r)
```

This refreshses a given `java.awt.Rectangle` that indicates a dirty region.

Parameters:
`r` - the Rectangle indicating the dirty region to be refreshed.

See Also:
`GlassPanel.repaint(long, int, int, int, int)`

repaint

```
public void repaint(long tm,
                    int x,
                    int y,
                    int width,
                    int height)
```

print

```
public void print(java.awt.Graphics g)
```

setPaintEnabled

```
public void setPaintEnabled(boolean b)
```

Enables or disables the repainting of this BeanBox.

paint

```
public void paint(java.awt.Graphics g)
```

paintImmediately

```
public void paintImmediately(int x,
        int y,
        int w,
        int h)
```

paintImmediately

```
public void paintImmediately(JComponent component)
```

Immediately paints the region of the (scaled) BeanBox that the given component is located.

See Also:

paintImmediately(int, int, int, int)

delete

```
public void delete()
```

Remove the current selection from the model.

cut

```
public void cut()
```

Serialize the current selection to the system clipboard, and then remove it from the BeanBox.

copy

```
public boolean copy()
```

Serializes the current selection to the system clipboard.

Returns:
true if the copy command is successful.

---

canPaste

```
public boolean canPaste()
```

The Paste command can only be executed if there are contents on the clipboard, and the Data on the clipboard is appropriate for the ModelEditor.

Returns:
true if a paste operation can be performed

---

canCopy

```
public boolean canCopy()
```

The Copy or cut command can only be performed if there is at least one object selected.

Returns:
true if a cut or copy operation can be performed

---

canDelete

```
public boolean canDelete()
```

The delete operation can only be performed if a copy opperation could be performed.

Returns:
true if a delete operation can be performed

See Also:
canCopy()

---

canAlign

```
public boolean canAlign()
```

The align operation can only be performed if two or more components are selected.

Returns:
true if an align operation can be performed

---

paste

```
public void paste()
```

Attempts to paste the current clipboard contents into this view.

---

loadVisualGroups

```
public static VisualGroup[] loadVisualGroups(com.apt.xdr.PibBlock[] blocks)
```

Loads visual groups from the given array of PibBlock's.

Parameters:

`model` - the AbstractModel to load components for
`blocks` - the PibBlock[] to load

---

loadComponents

```
public static java.awt.Component[] loadComponents(AbstractModel model,
        com.apt.xdr.PibBlock[] blocks)
```

Loads annotations and drawn components from the given array of PibBlock's.

Parameters:

`model` - the AbstractModel to load components for
`blocks` - the PibBlock[] to load

See Also:

`loadComponents(AbstractModel, PibBlock[], boolean)`

---

loadComponents

```
public static java.awt.Component[] loadComponents(AbstractModel model,
        com.apt.xdr.PibBlock[] blocks,
        boolean loadDrawn)
```

Loads annotations and drawn components from the given array of PibBlock's.

Parameters:

`model` - the AbstractModel to load components for
`blocks` - the PibBlock[] to load
`loadDrawn` - if true, DrawnComponents will be loaded; if false, only annotations will be loaded.

---

toFront

```
public void toFront(ConnectionRenderer connection)
```

Moves the given connection renderer to the front of the drawing stack.

---

findComponentsInside

```
public java.awt.Component[] findComponentsInside(int x1,
        int y1,
        int x2,
        int y2)
```

Retreive the list of components located within a rectangular region bounded by the points (x1,y1) and (x2,y2).

Parameters:

`x1` - the int x coordinate of the first corner.
`y1` - the int y coordinate of the first corner.
`x2` - the int x coordinate of the second corner.
`y2` - the int y coordinate of the second corner.

Returns:

the Component[] of Components that exist inside the given region.

getSelection

`public java.awt.Component[] getSelection(Vector selComps)`

Retreive the list of selected components in the drawing order. This makes sure that the BeanBox is not one of the selected components.

Parameters:

`selComps` - the Vector containing the selected components.

Returns:

a Component[] containing the selected components in drawing order.

storeState

`public void storeState(Hashtable state)`

Store the state of the bean to permit undo.

Parameters:

`state` - A hash table containing modified parameters.

restoreState

`public void restoreState(Hashtable state)`

Restore the state of the bean from an earlier edit.

Parameters:

`state` - A hash table containing modified parameters.

setSelectedComponent

`public void setSelectedComponent(java.awt.Component focus)`

Clear the current selection and add a single object to the selection.

Parameters:

`focus` - The bean to add to the selection. If null, the current beanbox is used.

removeSelectedComponent

`public void removeSelectedComponent(Object sel)`

This removes the Object from the list of selected components. If the Object is a DrawnComponent, it has it's selected flag set false.

Parameters:

`sel` - the Object that will be removed from the selection.

See Also:

`DrawnComponent.setSelected(boolean)`

addSelectedComponent

`public void addSelectedComponent(Object selected)`

Add a single object to the selection. If the object is already selected it will be removed from the selection, otherwise it will be added.

Parameters:

`selected` - the Object that will be added to the selection.

See Also:

`DrawnComponent.setSelected(boolean)`

addSelectedComponents

`public void addSelectedComponents(Object[] sel)`

Adds an array of beans to the current selection. If the bean is already selected, it will be removed from the current selection.

Parameters:

`sel` - The beans to add to the selection.

getNumSelected

`public int getNumSelected()`

Returns the number of selected components.

clearSelection

`public void clearSelection()`

Clears the current selection.

getSelected

`public java.awt.Component getSelected(int i)`

Returns a selected bean at the given index

Parameters:

`i` - the index of the bean in the selection.

getSelection

`public java.awt.Component[] getSelection()`

Return the array of selected objects. This will contain any DrawnComponents, Display Beans or Annotations but will not include DrawnSubComponent instances and thus the size of the returned array may not match `getNumSelected()`.

isSelected

`public boolean isSelected(java.awt.Component component)`

Returns true if the given component is part of the current selection.

getSelectionBounds

```
public java.awt.Rectangle getSelectionBounds()
```

Return the bounds of selected beans.

calcSelectionBounds

```
public void calcSelectionBounds()
```

Calculate the bounds of selected beans.

createPopupMenu

```
public JPopupMenu createPopupMenu(java.awt.event.MouseEvent evt)
```

Creates a popup menu suitable for this GlassPanel and the model it is viewing.

Returns:
a JPopupMenu containing appropriate items such as cut, copy paste and view properties.

See Also:
`AbstractComponent.getCustomPopupItems()`

toString

```
public String toString()
```

renumberSelectedComponents

```
public void renumberSelectedComponents()
```

This takes the selected `components` and has the model `renumber` their component numbers.

tryLockOrganize

```
public boolean tryLockOrganize()
```

Locks BeanBox's organizeView method so that only one organize occurs for a given view at a time, and so that it can be determined when the organize is complete.

unlockOrganize

```
public void unlockOrganize()
```

Unlocks orgainzeView() to be used by any thread.

organizeView

```
public void organizeView(Vector drawnComponents,
        boolean relative)
```

Calls `AbstractModel.layoutComponents` with the given components. This will produce an undo object for reseting the components to their original position.

Parameters:

`relative` - if true, the minmum x and y coordinates will be preserved.

---

organizeView

```
public void organizeView(Vector drawnComponents,
        boolean relative,
        boolean includeUndo)
```

Calls `AbstractModel.layoutComponents` with the given components.

Parameters:

`drawnComponents` - the Vector of DrawnComponents that will be organized.

`relative` - if true, the minmum x and y coordinates will be preserved.

`includeUndo` - if true, an undo object will be placed on the undo stack for reverting the organization.

---

getComponents

```
public Vector getComponents(boolean includeConnections)
```

Retrieves a Vector of all the Components in this BeanBox.

Parameters:

`includeConnections` - if true, DrawnConnection objects will be included in the list.

Returns:

a Vector containing all the Components in the BeanBox.

---

getDrawnComponents

```
public Vector getDrawnComponents()
```

Retrieves a Vector of all the Components in this BeanBox, including `Annotations`, `DrawnComponents` and `DrawnConnections`.

Returns:

a Vector containing all the Components in the BeanBox.

---

createZoomMenu

```
public JMenu createZoomMenu()
```

Creates a Zoom menu suitable for this view.

Returns:

the JMenu containing all of the possible zoom options for this view.

fitToWindow

`public void fitToWindow()`

sets this BeanBox's zoom such that its contents will fit in the current window size in the same way as the "Fit To Window" menu item.

getViewComponent

`public ViewComponent getViewComponent()`

Retrieves the ViewComponent that corresponds with the DrawnView that contains this BeanBox.

getWidthScaleFactor

`public double getWidthScaleFactor()`

Gets this view's width scale factor. This factor is intended for use in scaling the diameter or width of components that may have one dimension much larger than the other; such as a 10 meter long, 0.1 meter wide pipe.

updateComponentList

```
public void updateComponentList(Vector components,
        boolean organize)
```

Synchronizes the component list in this BeanBox to the given list of components. This is used from the InsertComponentDialog.

Parameters:

`components` - the Vector of `components` that should be rendered in this view.
`organize` - true if all the components in the view should be organized.

showAllConnections

`public void showAllConnections()`

This goes through all of the components currently in the view, and tries to render their connections.

See Also:

`addDrawnConnection(Connection)`

showAllConnections

`public void showAllConnections(boolean undoable)`

This goes through all of the components currently in the view, and tries to render their connections.

Parameters:

`undoable` - if true, an undoable event will be posted for this change

See Also:

`addDrawnConnection(Connection)`

organizeSelection

`public void **organizeSelection**()`

Organize the selected components. At least two components must be selected for the organize routine to make any sense.

---

redrawSelection

`public void **redrawSelection**()`

Redraw the selected components, or all components in this view if none are selected

---

selectCategory

`public void **selectCategory**(Category cat)`

This clears the current selection, and adds all the components that are of the given categories subset to the "selected" list.

---

resetConnections

`public void **resetConnections**()`

This resets all the connections that connect to the selected components

---

minimizeView

`public void **minimizeView**()`

Minimizes the current view to the minimum possible with the current components. This will translate all the components, until the left most is 20 from the left edge, and the upper most is 20 from the top edge, then it will clip the lower right edges to be 20 from the right most component and bottom most component.

---

updateSelection

`public void **updateSelection**()`

Updates this BeaBox's current selection and notifies MainFrame and the main property view of the new current model and current selection.

---

addBoxSelectionListener

`public void **addBoxSelectionListener**(BoxSelectionListener listener)`

Adds the given listener to the list that is notified when the BeanBox selection set may have changed. Note that notification does not ensure that the selection has changed, only that it may have changed.

Parameters:

`listener` - the BoxSelectionListener to add for notification of selection changes

See Also:

`removeBoxSelectionListener(BoxSelectionListener)`

---

removeBoxSelectionListener

```
public void removeBoxSelectionListener(BoxSelectionListener listener)
```

Removes the given listener from the list that is notified when the BeanBox selection set may have changed.

Parameters:

`listener` - the BoxSelectionListener to remove from the list of selection listeners

See Also:

`addBoxSelectionListener(BoxSelectionListener)`

---

reconnectGroupIDs

```
public void reconnectGroupIDs()
```

Reconnects all invalid VisualGroup ID references.

See Also:

`com.cafean.client.ui.util.Groupable`
`com.cafean.client.ui.util.VisualGroup`
`group()`
`ungroup()`

---

canGroup

```
public boolean canGroup()
```

Returns true if the current selection can be grouped.

---

canUngroup

```
public boolean canUngroup()
```

Returns true if the current selection can be ungrouped.

---

group

```
public void group()
```

Groups the current selection as if the Group button had been pressed by the user.

This method handles undo events, selection updates and repainting.

See Also:

`com.cafean.client.ui.util.Groupable`
`com.cafean.client.ui.util.VisualGroup`

---

ungroup

```
public void ungroup()
```

Ungroups the current selection as if the Ungroup button had been pressed by the user.

This method handles undo events, selection updates and repainting.

See Also:

`com.cafean.client.ui.util.Groupable`
`com.cafean.client.ui.util.VisualGroup`

---

selectGroup

`public void` **`selectGroup`**`(int groupID)`

Selects the objects in the given group and all parent and child groups.

Parameters:

`groupID` - the group to select the contents of.

See Also:

`com.cafean.client.ui.util.Groupable`
`com.cafean.client.ui.util.VisualGroup`

---

deselectGroup

`public void` **`deselectGroup`**`(int groupID)`

Deselects the objects in the given group and all parent and child groups.

Parameters:

`groupID` - the group to de-select the contents of.

See Also:

`com.cafean.client.ui.util.Groupable`
`com.cafean.client.ui.util.VisualGroup`

---

getVisualGroups

`public VisualGroup[]` **`getVisualGroups`**`()`

Retrieves the list of VisualGroup instances in this BeanBox.

Returns:

a VisualGroup[] containing references to the current groups.

See Also:

`com.cafean.client.ui.util.Groupable`
`com.cafean.client.ui.util.VisualGroup`

---

addVisualGroup

`public void` **`addVisualGroup`**`(VisualGroup group)`

Adds the given VisualGroup to this beanbox and assigns a new ident.

Parameters:

`group` - the VisualGroup to add to this BeanBox

See Also:

com.cafean.client.ui.util.Groupable
com.cafean.client.ui.util.VisualGroup

---

addVisualGroup

```
public void addVisualGroup(VisualGroup group,
        boolean adjustIdent)
```

Adds the given VisualGroup to this beanbox and assigns a new ident (if necessary).

Parameters:

`group` - the VisualGroup to add to this BeanBox
`adjustIdent` - if true (or the current ident is 0) a new ident will be allocated for the given group.

See Also:

com.cafean.client.ui.util.Groupable
com.cafean.client.ui.util.VisualGroup

---

removeVisualGroup

```
public void removeVisualGroup(VisualGroup group,
        CompoundEdit edit)
```

Removes the given group from this beanbox and adds an appropriate undoable edit to the given CompoundEdit.

Note that this method zeros out the groupIDs of any Groupables that currently refer to the given group.

See Also:

com.cafean.client.ui.util.Groupable
com.cafean.client.ui.util.VisualGroup

---

findGroupByIdent

```
public VisualGroup findGroupByIdent(int ident)
```

Retrieves the VisualGroup instance in this BeanBox that has the given ident.

See Also:

com.cafean.client.ui.util.Groupable
com.cafean.client.ui.util.VisualGroup

---

findGroupByOldIdent

```
public VisualGroup findGroupByOldIdent(int oldIdent)
```

Retrieves the VisualGroup instance in this BeanBox that has the given old ident.

The oldIdent field is used when pasting from the clipboard to allow pasted objects to reconnect to the newly pasted group without interfering with existing groups and group references.

See Also:

com.cafean.client.ui.util.Groupable
com.cafean.client.ui.util.VisualGroup

---

setBackground

public void **setBackground**(java.awt.Color bg)

---

revalidate

public void **revalidate**()

com.cafean.client.ui
Interface BoxSelectionListener

public interface **BoxSelectionListener**
extends

An interface describing a listener that is to be notified when the selection in a particular BeanBox instance may have changed.

Method Summary

void	boxSelectionChanged(BeanBox box) Notifies this listener that the selection in the given BeanBox may have changed.

Methods

boxSelectionChanged

```
public void boxSelectionChanged(BeanBox box)
```

Notifies this listener that the selection in the given BeanBox may have changed.

Parameters:

box - the BeanBox who's selection has changed

com.cafean.client.ui
Class ClientPluginLoader

```
java.lang.Object
   |
   +-com.cafean.client.ui.ClientPluginLoader
```

All Implemented Interfaces:
PluginListener

---

public class **ClientPluginLoader**
extends Object
implements PluginListener

A delegate used to load client plugins into the given Vector

Constructor Summary

public	`ClientPluginLoader(Vector plugins)` Creates a new instance of ClientPluginLoader to load client plugins into the given Vectors.

Method Summary

void	`pluginException(Exception ex)` Called when there is an exception thrown reading in a plugin.
void	`pluginLoaded(Object plugin)` Adds the given plugin to the list of plugin's to be loaded in `loadPlugins`.

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Methods inherited from interface `com.cafean.CodePlugins.CodePluginClassLoader.PluginListener`

`pluginException, pluginLoaded`

Constructors

ClientPluginLoader

`public` **`ClientPluginLoader`**`(Vector plugins)`

Creates a new instance of ClientPluginLoader to load client plugins into the given Vectors.

Methods

pluginLoaded

`public void` **`pluginLoaded`**`(Object plugin)`

Adds the given plugin to the list of plugin's to be loaded in `loadPlugins`.

Parameters:

`plugin` - the ClientCodePlugin object that has been loaded by the `com.cafean.CodePlugins.CodePluginClassLoader`; all other types are ignored.

---

pluginException

`public void` **`pluginException`**`(Exception ex)`

Called when there is an exception thrown reading in a plugin.

Parameters:

`ex` - the Exception thrown while reading in a plugin.

com.cafean.client.ui
Class ComponentSelectionDialog

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-java.awt.Window
            |
            +-java.awt.Dialog
               |
               +-javax.swing.JDialog
                  |
                  +-com.cafean.client.ui.ComponentSelectionDialog
```

All Implemented Interfaces:
ListSelectionListener, java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, javax.accessibility.Accessible, HasGetTransferHandler, RootPaneContainer, javax.accessibility.Accessible, WindowConstants

---

public class **ComponentSelectionDialog**
extends JDialog
implements WindowConstants, javax.accessibility.Accessible, RootPaneContainer, HasGetTransferHandler, javax.accessibility.Accessible, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, ListSelectionListener

This dialog allows for the selection of a component that is stored by ident. A Category may be specified before the dialog is set visible which will limit the selection scope to the components in that Category. Optionally, the dialog may be set to enable the "Clear" button which clears the current selection. Additionally, the ident of a currently selected component can be set.

Fields inherited from class `java.awt.Dialog`
`DEFAULT_MODALITY_TYPE`

Fields inherited from class `java.awt.Component`
`BOTTOM_ALIGNMENT`, `CENTER_ALIGNMENT`, `LEFT_ALIGNMENT`, `RIGHT_ALIGNMENT`, `TOP_ALIGNMENT`

Fields inherited from interface `java.awt.image.ImageObserver`
`ABORT`, `ALLBITS`, `ERROR`, `FRAMEBITS`, `HEIGHT`, `PROPERTIES`, `SOMEBITS`, `WIDTH`

Fields inherited from interface `javax.swing.WindowConstants`
`DISPOSE_ON_CLOSE`, `DO_NOTHING_ON_CLOSE`, `EXIT_ON_CLOSE`, `HIDE_ON_CLOSE`

Constructor Summary

public	`ComponentSelectionDialog(java.awt.Frame parent)` Creates new form ComponentSelectionDialog.

public	ComponentSelectionDialog(java.awt.Dialog parent) Creates new form ComponentSelectionDialog.

Method Summary

AbstractComponent	getSelection() Returns the component selected by the dialog, or NULL if it was cleared
boolean	isCancelled() Returns true unless the dialog was exited using the OK button
void	setAllowEmpty(boolean allowEmpty) Sets whether or not this dialog can allow no selection
void	setAvailableComponents(AbstractComponent[] components) Initializes the array of available components.
void	setCategory(Category category) Sets the Category that will be used to fill this dialog.
void	setComponentComparator(Comparator comparator) Sets the Comparator instance used to sort the components displayed in this component selection dialog.
void	setCreator(ComponentCreator creator) Sets the custom component creator that this dialog will execute if the user presses the create button.
void	setModel(AbstractModel model) Sets the AbstractModel for this component
void	setSelection(AbstractComponent comp) Sets the initially selected component
void	setSelection(int selection) Sets the initial selection ident
void	setShowCategory(boolean show) If set to true, the category number column will be shown.
void	setShowComponentNumber(boolean show) If set to true, the component number column will be shown.
void	setShowIcons(boolean showIcons) Sets whether or not this dialog will show component category icons
void	setTitle(String title) Sets the default title.
void	setVisible(boolean visible) This is where the dialog fills the list of components by Category, or by reading in all the components in the model.
void	valueChanged(ListSelectionEvent e)

Methods inherited from class `javax.swing.JDialog`

getAccessibleContext, getContentPane, getDefaultCloseOperation, getGlassPane, getGraphics, getJMenuBar, getLayeredPane, getRootPane, getTransferHandler, isDefaultLookAndFeelDecorated, remove, repaint, setContentPane, setDefaultCloseOperation, setDefaultLookAndFeelDecorated, setGlassPane, setJMenuBar, setLayeredPane, setLayout, setTransferHandler, update

Methods inherited from class `java.awt.Dialog`

addNotify, getAccessibleContext, getModalityType, getTitle, hide, isModal, isResizable, isUndecorated, setModal, setModalityType, setResizable, setTitle, setUndecorated, setVisible, show, toBack

Methods inherited from class `java.awt.Window`

addNotify, addPropertyChangeListener, addPropertyChangeListener, addWindowFocusListener, addWindowListener, addWindowStateListener, applyResourceBundle, applyResourceBundle, createBufferStrategy, createBufferStrategy, dispose, getAccessibleContext, getBufferStrategy, getFocusableWindowState, getFocusCycleRootAncestor, getFocusOwner, getFocusTraversalKeys, getGraphicsConfiguration, getIconImages, getInputContext, getListeners, getLocale, getModalExclusionType, getMostRecentFocusOwner, getOwnedWindows, getOwner, getOwnerlessWindows, getToolkit, getWarningString, getWindowFocusListeners, getWindowListeners, getWindows, getWindowStateListeners, hide, isActive, isAlwaysOnTop, isAlwaysOnTopSupported, isFocusableWindow, isFocusCycleRoot, isFocused, isLocationByPlatform, isShowing, pack, postEvent, removeNotify, removeWindowFocusListener, removeWindowListener, removeWindowStateListener, reshape, setAlwaysOnTop, setBounds, setBounds, setCursor, setFocusableWindowState, setFocusCycleRoot, setIconImage, setIconImages, setLocationByPlatform, setLocationRelativeTo, setMinimumSize, setModalExclusionType, setSize, setSize, setVisible, show, toBack, toFront

Methods inherited from class `java.awt.Container`

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class `java.awt.Component`

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.awt.image.ImageObserver`

imageUpdate

Methods inherited from interface `java.awt.MenuContainer`

getFont, postEvent, remove

Methods inherited from interface `javax.accessibility.Accessible`

getAccessibleContext

Methods inherited from interface javax.accessibility.Accessible
getAccessibleContext

Methods inherited from interface javax.swing.RootPaneContainer
getContentPane, getGlassPane, getLayeredPane, getRootPane, setContentPane, setGlassPane, setLayeredPane

Methods inherited from interface javax.swing.TransferHandler.HasGetTransferHandler
getTransferHandler

Methods inherited from interface javax.swing.event.ListSelectionListener
valueChanged

Constructors

ComponentSelectionDialog

`public` **`ComponentSelectionDialog`**`(java.awt.Frame parent)`

Creates new form ComponentSelectionDialog. This initializes the table and sets the window location.

ComponentSelectionDialog

`public` **`ComponentSelectionDialog`**`(java.awt.Dialog parent)`

Creates new form ComponentSelectionDialog. This initializes the table and sets the window location.

Methods

setCategory

`public void` **`setCategory`**`(Category category)`

Sets the Category that will be used to fill this dialog.

setAllowEmpty

`public void` **`setAllowEmpty`**`(boolean allowEmpty)`

Sets whether or not this dialog can allow no selection

setShowIcons

`public void` **`setShowIcons`**`(boolean showIcons)`

Sets whether or not this dialog will show component category icons

---

setSelection

```
public void setSelection(int selection)
```

Sets the initial selection ident

---

setSelection

```
public void setSelection(AbstractComponent comp)
```

Sets the initially selected component

---

setModel

```
public void setModel(AbstractModel model)
```

Sets the AbstractModel for this component

---

setCreator

```
public void setCreator(ComponentCreator creator)
```

Sets the custom component creator that this dialog will execute if the user presses the create button. Calling this with a value other then null before the dialog is set visible makes the create button appear.

---

setShowComponentNumber

```
public void setShowComponentNumber(boolean show)
```

If set to true, the component number column will be shown.

---

setShowCategory

```
public void setShowCategory(boolean show)
```

If set to true, the category number column will be shown.

---

setTitle

```
public void setTitle(String title)
```

Sets the default title.

---

setVisible

```
public void setVisible(boolean visible)
```

This is where the dialog fills the list of components by Category, or by reading in all the components in the model.

---

setAvailableComponents

`public void` **`setAvailableComponents`**`(AbstractComponent[] components)`

Initializes the array of available components. This allows for a selection from a group of components defined by the editor.

---

setComponentComparator

`public void` **`setComponentComparator`**`(Comparator comparator)`

Sets the Comparator instance used to sort the components displayed in this component selection dialog.

Parameters:

`comparator` - the Comparator instance to use or null if no sorting should be performed.

---

isCancelled

`public boolean` **`isCancelled`**`()`

Returns true unless the dialog was exited using the OK button

---

getSelection

`public AbstractComponent` **`getSelection`**`()`

Returns the component selected by the dialog, or NULL if it was cleared

---

valueChanged

`public void` **`valueChanged`**`(ListSelectionEvent e)`

com.cafean.client.ui
Class ComponentSelector

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-java.awt.Window
            |
            +-java.awt.Dialog
               |
               +-javax.swing.JDialog
                  |
                  +-com.cafean.client.ui.ComponentSelector
```

All Implemented Interfaces:
java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, javax.accessibility.Accessible, HasGetTransferHandler, RootPaneContainer, javax.accessibility.Accessible, WindowConstants

---

public class **ComponentSelector**
extends JDialog

A dialog for selecting a single component from a list provided upon creation. `Usage: ComponentSelector dlg = new ComponentSelector(...); dlg.setVisible(true); AbstractComponent selection = dlg.getSelection(); dlg.dispose();`

Fields inherited from class `java.awt.Dialog`
DEFAULT_MODALITY_TYPE

Fields inherited from class `java.awt.Component`
BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface `java.awt.image.ImageObserver`
ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Fields inherited from interface `javax.swing.WindowConstants`
DISPOSE_ON_CLOSE, DO_NOTHING_ON_CLOSE, EXIT_ON_CLOSE, HIDE_ON_CLOSE

Constructor Summary

public	`ComponentSelector(java.awt.Frame parent, boolean modal, AbstractComponent[] comps)` Creates new form CompomentSelector
public	`ComponentSelector(JDialog parent, boolean modal, AbstractComponent[] comps)` Creates new form CompomentSelector

public	ComponentSelector(JDialog parent, boolean modal, Vector comps) Creates new form CompomentSelector
public	ComponentSelector(java.awt.Frame parent, boolean modal, Vector comps) Creates new form CompomentSelector

Method Summary

AbstractComponent	getSelection() Return the selected component or null if there no selection has been made.
void	hideComponentNumber() Hides the component number field of this ComponentSelector.
void	hideComponentType() Hides the component type field of this ComponentSelector.
boolean	isCanceled() This returns true if the dialog is closed by any other means then by pressing the OK button.
void	setSelected(AbstractComponent comp)
void	showCancelButton(boolean b) If this operation is not something that should be cancellable, this function can be called to prevent the cancel button from being displayed.

Methods inherited from class javax.swing.JDialog

getAccessibleContext, getContentPane, getDefaultCloseOperation, getGlassPane, getGraphics, getJMenuBar, getLayeredPane, getRootPane, getTransferHandler, isDefaultLookAndFeelDecorated, remove, repaint, setContentPane, setDefaultCloseOperation, setDefaultLookAndFeelDecorated, setGlassPane, setJMenuBar, setLayeredPane, setLayout, setTransferHandler, update

Methods inherited from class java.awt.Dialog

addNotify, getAccessibleContext, getModalityType, getTitle, hide, isModal, isResizable, isUndecorated, setModal, setModalityType, setResizable, setTitle, setUndecorated, setVisible, show, toBack

Methods inherited from class java.awt.Window

addNotify, addPropertyChangeListener, addPropertyChangeListener, addWindowFocusListener, addWindowListener, addWindowStateListener, applyResourceBundle, applyResourceBundle, createBufferStrategy, createBufferStrategy, dispose, getAccessibleContext, getBufferStrategy, getFocusableWindowState, getFocusCycleRootAncestor, getFocusOwner, getFocusTraversalKeys, getGraphicsConfiguration, getIconImages, getInputContext, getListeners, getLocale, getModalExclusionType, getMostRecentFocusOwner, getOwnedWindows, getOwner, getOwnerlessWindows, getToolkit, getWarningString, getWindowFocusListeners, getWindowListeners, getWindows, getWindowStateListeners, hide, isActive, isAlwaysOnTop, isAlwaysOnTopSupported, isFocusableWindow, isFocusCycleRoot, isFocused, isLocationByPlatform, isShowing, pack, postEvent, removeNotify, removeWindowFocusListener, removeWindowListener, removeWindowStateListener, reshape, setAlwaysOnTop, setBounds, setBounds, setCursor, setFocusableWindowState, setFocusCycleRoot, setIconImage, setIconImages, setLocationByPlatform, setLocationRelativeTo, setMinimumSize, setModalExclusionType, setSize, setSize, setVisible, show, toBack, toFront

Methods inherited from class `java.awt.Container`

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class `java.awt.Component`

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.awt.image.ImageObserver`

imageUpdate

Methods inherited from interface `java.awt.MenuContainer`

getFont, postEvent, remove

Methods inherited from interface `javax.accessibility.Accessible`

getAccessibleContext

Methods inherited from interface javax.accessibility.Accessible
getAccessibleContext

Methods inherited from interface javax.swing.RootPaneContainer
getContentPane, getGlassPane, getLayeredPane, getRootPane, setContentPane, setGlassPane, setLayeredPane

Methods inherited from interface javax.swing.TransferHandler.HasGetTransferHandler
getTransferHandler

Constructors

ComponentSelector

```
public ComponentSelector(java.awt.Frame parent,
                         boolean modal,
                         AbstractComponent[] comps)
```

Creates new form CompomentSelector

Parameters:

`parent` - the parent `java.awt.Frame` of this dialog.
`modal` - true if this dialog shoud be modal.
`comps` - the AbstractComponent[] list to choose from.

ComponentSelector

```
public ComponentSelector(JDialog parent,
                         boolean modal,
                         AbstractComponent[] comps)
```

Creates new form CompomentSelector

Parameters:

`parent` - the parent `javax.swing.JDialog` of this dialog.
`modal` - true if this dialog shoud be modal.
`comps` - the AbstractComponent[] list to choose from.

ComponentSelector

```
public ComponentSelector(JDialog parent,
                         boolean modal,
                         Vector comps)
```

Creates new form CompomentSelector

Parameters:

parent - the parent java.awt.Frame of this dialog.
modal - true if this dialog shoud be modal.
comps - the Vector of components to choose from.

ComponentSelector

```
public ComponentSelector(java.awt.Frame parent,
                         boolean modal,
                         Vector comps)
```

Creates new form CompomentSelector

Parameters:

parent - the parent javax.swing.JDialog of this dialog.
modal - true if this dialog shoud be modal.
comps - the Vector of components to choose from.

Methods

hideComponentType

```
public void hideComponentType()
```

Hides the component type field of this ComponentSelector.

---

hideComponentNumber

```
public void hideComponentNumber()
```

Hides the component number field of this ComponentSelector.

---

setSelected

```
public void setSelected(AbstractComponent comp)
```

---

getSelection

```
public AbstractComponent getSelection()
```

Return the selected component or null if there no selection has been made.

Returns:

the AbstractComponent selected from the list by the user.

---

showCancelButton

```
public void showCancelButton(boolean b)
```

If this operation is not something that should be cancellable, this function can be called to prevent the cancel button from being displayed.

Parameters:

b - boolean value determining if the cancel button should be visible.

isCanceled

```
public boolean isCanceled()
```

This returns true if the dialog is closed by any other means then by pressing the OK button.

Returns:

false if the OK button was used to close this dialog.

com.cafean.client.ui
Class ConnectingPt

```
java.lang.Object
   |
   +-com.cafean.client.ui.ConnectingPt
```

All Implemented Interfaces:
Cloneable

---

public class **ConnectingPt**
extends Object
implements Cloneable

The ConnectingPt are the small target and source points on `components` that are used to both connect two components using the `connect` tool, and to render a connection betweeen two components using a `DrawnConnection`

Field Summary

public static final	`CROSSFLOW_DROP_ZONE` A connection target for a crossflow connect Value: **2**
public static final	`DISABLED_CONNECTOR` a connection point that is non-functional Value: **2147483647**
public static final	`INLET_DROP_ZONE` Connection target that points into the component. Value: **0**
public static final	`INLET_MOVEABLE_CONNECTOR` A connection source that points into the component. Value: **3**
public static final	`INVISIBLE_CONNECTOR` A connection point that does not draw on the drawn component Value: **6**
public static final	`OUTLET_DROP_ZONE` A Connection target that points out from the component Value: **1**
public static final	`OUTLET_MOVEABLE_CONNECTOR` A connection source that points out of the component Value: **4**
public static final	`STATIC_CONNECTOR` A connection source that cannot be moved Value: **5**

Constructor Summary

public	`ConnectingPt(int type, int linkNum)` The default constructor for a ConnectingPt.
public	`ConnectingPt(int type, int linkNum, ConnectionData data)` The default constructor for a ConnectingPt.

Method Summary

Object	`clone()`
java.awt.Color[]	`getBadgeColors()` Gets the colors of the Badges for this ConnectingPt
java.awt.Shape[]	`getBadges()` Gets the Badges for this ConnectingPt
ConnectionData	`getConnectionData()` Gets the ConnectionData that defines this ConnectingPt's position.
java.awt.Shape	`getHandle()`
double	`getHandleSize()` Gets the size of the handle.
boolean	`getNeedsTarget()` Returns true if connections started from this ConnectingPt will allow connections to whole components.
Pad	`getPad()` This returns the actual drawn location on the panel where a connection can be initiated or completed with the `connect` tool.
static java.awt.geom.Point2D.Double	`getRotatedLocation(double theta, java.awt.geom.Point2D.Double pt, java.awt.geom.Point2D.Double ctr)` computes the new location of a point after it has been rotated.
static java.awt.Point	`getRotatedLocation(double theta, java.awt.Point pt, java.awt.Point ctr)` computes the new location of a point after it has been rotated.
int	`getType()` Returns the type of this ConnectingPt.
boolean	`isConnected()` Returns true if there is a `Connection` to the `AbstractComponent` that woulc be drawn to this ConnectingP existst, even if there is no `DrawnConnection` for that `Connection` on this `DrawnView`
boolean	`isVisible()` Returns true if this ConnectingPt is a connectinPt that can be visible.
void	`move(double diffx, double diffy)` Moves a ConnectingPt by the distance in each dimension.

void	rotate(double theta, double x, double y) Rotates a ConnectingPt clockwize by the angle theta, around the point specified by x and y.
void	setBadgeColors(java.awt.Color[] badgeColors) Sets the colors of the Badges for this ConnectingPt
void	setBadges(java.awt.Shape[] theBadges) Sets the Badges for this ConnectingPt
void	setConnected() Should be called if there is a Connection to the AbstractComponent that woulc be drawn to this ConnectingPt exists, even if there is no DrawnConnection for that Connection on this DrawnView.
void	setConnectPtType(int type) Sets the type of this ConnectingPt.
void	setDisconnected() Should be called if there is a Connection to the AbstractComponent that woulc be drawn to this ConnectingPt has been disconnected.This specifically gets called by the DrawnComponent when a Connection has been disconnected.
void	setHandle(java.awt.Shape theHandle)
void	setHandleSize(double hsHeight) Sets the size of the handle.
void	setNeedsTarget(boolean val_) Sets whether connections started from this ConnectingPt will allow connections to whole components.
void	setPad(Pad thePad) This sets the actual drawn location on the panel where a connection can be initiated or completed with the connect tool.
String	toString() Constructs an informative label for this ConnectingPt, including type, and pad coordinates
boolean	typeIsConnector() This determines if this ConnectingPt is of a type that can initiate a connection.
boolean	typeIsCrossflow() This determines if this ConnectingPt is the used to represent the outlet of a component.
boolean	typeIsDropZone() This determines if this ConnectingPt is of a type that can complete a connection.
boolean	typeIsInlet() This determines if this ConnectingPt is the used to represent the inlet of a component.
boolean	typeIsOutlet() This determines if this ConnectingPt is the used to represent the outlet of a component.
boolean	typeIsStatic() This determines if this ConnectingPt is of a type that can be moved around.

Methods inherited from class `java.lang.Object`

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Fields

INLET_DROP_ZONE

`public static final int` **`INLET_DROP_ZONE`**

Connection target that points into the component.
Constant value: **0**

OUTLET_DROP_ZONE

`public static final int` **`OUTLET_DROP_ZONE`**

A Connection target that points out from the component
Constant value: **1**

CROSSFLOW_DROP_ZONE

`public static final int` **`CROSSFLOW_DROP_ZONE`**

A connection target for a crossflow connect
Constant value: **2**

INLET_MOVEABLE_CONNECTOR

`public static final int` **`INLET_MOVEABLE_CONNECTOR`**

A connection source that points into the component.
Constant value: **3**

OUTLET_MOVEABLE_CONNECTOR

`public static final int` **`OUTLET_MOVEABLE_CONNECTOR`**

A connection source that points out of the component
Constant value: **4**

STATIC_CONNECTOR

`public static final int` **`STATIC_CONNECTOR`**

A connection source that cannot be moved
Constant value: **5**

INVISIBLE_CONNECTOR

`public static final int` **`INVISIBLE_CONNECTOR`**

A connection point that does not draw on the drawn component
Constant value: **6**

DISABLED_CONNECTOR

```
public static final int DISABLED_CONNECTOR
```

a connection point that is non-functional
Constant value: **2147483647**

Constructors

ConnectingPt

```
public ConnectingPt(int type,
                    int linkNum)
```

The default constructor for a ConnectingPt.

Parameters:
`type` - the enumerated type of connectingPt. Acceptable values are.

- INLET_DROP_ZONE
- OUTLET_DROP_ZONE
- CROSSFLOW_DROP_ZONE
- INLET_MOVEABLE_CONNECTOR
- OUTLET_MOVEABLE_CONNECTOR
- INVISIBLE_CONNECTOR

`linkNum` - the index of this connectionPoint.

ConnectingPt

```
public ConnectingPt(int type,
                    int linkNum,
                    ConnectionData data)
```

The default constructor for a ConnectingPt.

Parameters:
`type` - the enumerated type of connectingPt. Acceptable values are.

- INLET_DROP_ZONE
- OUTLET_DROP_ZONE
- CROSSFLOW_DROP_ZONE
- INLET_MOVEABLE_CONNECTOR
- OUTLET_MOVEABLE_CONNECTOR

`linkNum` - the index of this connectionPoint.
`data` - the ConnectionData that defines this ConnectingPt's position.

Methods

getConnectionData

`public ConnectionData` **`getConnectionData`**`()`

Gets the ConnectionData that defines this ConnectingPt's position. This data is used by the `connect` tool to determine the details of the connection.

Returns:
the ConnectionData.

---

toString

`public String` **`toString`**`()`

Constructs an informative label for this ConnectingPt, including type, and pad coordinates

Returns:
the String for quickly representing this ConnectingPt.

See Also:
`getPad()`

---

typeIsConnector

`public boolean` **`typeIsConnector`**`()`

This determines if this ConnectingPt is of a type that can initiate a connection.

Returns:
true if the type is one of the following:

- `INLET_MOVEABLE_CONNECTOR`
- `OUTLET_MOVEABLE_CONNECTOR`
- `STATIC_CONNECTOR`

---

typeIsDropZone

`public boolean` **`typeIsDropZone`**`()`

This determines if this ConnectingPt is of a type that can complete a connection.

Returns:

true if the type is one of the following:

- INLET_DROP_ZONE
- OUTLET_DROP_ZONE
- CROSSFLOW_DROP_ZONE

---

typeIsStatic

```
public boolean typeIsStatic()
```

This determines if this ConnectingPt is of a type that can be moved around.

Returns:
true if the type is one of the following:

- INLET_DROP_ZONE
- OUTLET_DROP_ZONE
- CROSSFLOW_DROP_ZONE
- STATIC_CONNECTOR

---

typeIsInlet

```
public boolean typeIsInlet()
```

This determines if this ConnectingPt is the used to represent the inlet of a component.

Returns:
true if the type is one of the following:

- INLET_DROP_ZONE
- INLET_MOVEABLE_CONNECTOR

---

clone

```
public Object clone()
```

---

typeIsOutlet

```
public boolean typeIsOutlet()
```

This determines if this ConnectingPt is the used to represent the outlet of a component.

Returns:
true if the type is one of the following:

- OUTLET_DROP_ZONE
- OUTLET_MOVEABLE_CONNECTOR

---

typeIsCrossflow

```
public boolean typeIsCrossflow()
```

This determines if this ConnectingPt is the used to represent the outlet of a component.

Returns:
true if the type is CROSSFLOW_DROP_ZONE.

---

getType

```
public int getType()
```

Returns the type of this ConnectingPt.

Returns:
one of the following:

- INLET_DROP_ZONE
- OUTLET_DROP_ZONE
- CROSSFLOW_DROP_ZONE
- INLET_MOVEABLE_CONNECTOR
- OUTLET_MOVEABLE_CONNECTOR

---

isVisible

```
public boolean isVisible()
```

Returns true if this ConnectingPt is a connectinPt that can be visible.

Returns:
true if the type is INVISIBLE_CONNECTOR.

---

setConnectPtType

```
public void setConnectPtType(int type)
```

Sets the type of this ConnectingPt.

Parameters:

type - the new type of this ConnectingPoint, acceptable values are:

- INLET_DROP_ZONE
- OUTLET_DROP_ZONE
- CROSSFLOW_DROP_ZONE
- INLET_MOVEABLE_CONNECTOR
- OUTLET_MOVEABLE_CONNECTOR

---

isConnected

```
public boolean isConnected()
```

Returns true if there is a Connection to the AbstractComponent that woulc be drawn to this ConnectingP existst, even if there is no DrawnConnection for that Connection on this DrawnView

Returns:
true if this ConnectingPt is currently connected.

---

setConnected

```
public void setConnected()
```

Should be called if there is a Connection to the AbstractComponent that woulc be drawn to this ConnectingPt exists, even if there is no DrawnConnection for that Connection on this DrawnView. This specifically gets called by the connect tool after a connection has been completed.

---

setDisconnected

```
public void setDisconnected()
```

Should be called if there is a Connection to the AbstractComponent that woulc be drawn to this ConnectingPt has been disconnected.This specifically gets called by the DrawnComponent when a Connection has been disconnected.

---

getPad

```
public Pad getPad()
```

This returns the actual drawn location on the panel where a connection can be initiated or completed with the connect tool.

Returns:
the Pad for this ConnectingPt.

---

setPad

```
public void setPad(Pad thePad)
```

This sets the actual drawn location on the panel where a connection can be initiated or completed with the connect tool.

Parameters:
thePad - the Pad for this ConnectingPt.

---

getHandle

```
public java.awt.Shape getHandle()
```

---

setHandle

```
public void setHandle(java.awt.Shape theHandle)
```

---

getBadges

```
public java.awt.Shape[] getBadges()
```

Gets the Badges for this ConnectingPt

Returns:
the Shape[] containing the badges for this ConnectingPt

---

setBadges

```
public void setBadges(java.awt.Shape[] theBadges)
```

Sets the Badges for this ConnectingPt

Parameters:
`theBadges` - the Shape[] containing the badges for this ConnectingPt

---

setBadgeColors

```
public void setBadgeColors(java.awt.Color[] badgeColors)
```

Sets the colors of the Badges for this ConnectingPt

Parameters:
`badgeColors` - the Color[] containing the colors for the badges for this ConnectingPt

---

getBadgeColors

```
public java.awt.Color[] getBadgeColors()
```

Gets the colors of the Badges for this ConnectingPt

Returns:
the Color[] containing the colors for the badges for this ConnectingPt

---

setHandleSize

```
public void setHandleSize(double hsHeight)
```

Sets the size of the handle.

Parameters:

`hsHeight` - the handle size for both width and height.

---

getHandleSize

```
public double getHandleSize()
```

Gets the size of the handle.

Returns:

the handle size for both width and height.

---

move

```
public void move(double diffx,
         double diffy)
```

Moves a ConnectingPt by the distance in each dimension. All badges, the Pad, and the Handle are translated by diffx and diffy ammount.

Parameters:

`diffx` - the distance to translate in the X direction.

`diffy` - the distance to translate in the Y direction.

See Also:

`AffineTransform.translate(double, double)`

---

rotate

```
public void rotate(double theta,
         double x,
         double y)
```

Rotates a ConnectingPt clockwize by the angle theta, around the point specified by x and y. All badges, the Pad, and the Handle are rotated as well.

Parameters:

`theta` - the angular distance to rotate, clockwise in radians.

`x` - the x point to rotate about.

`y` - the y point to rotate about.

See Also:

`AffineTransform.rotate(double)`

---

getRotatedLocation

```
public static java.awt.Point getRotatedLocation(double theta,
         java.awt.Point pt,
         java.awt.Point ctr)
```

computes the new location of a point after it has been rotated.

Parameters:

`theta` - the angle in radians of rotation (remember Java rotates clockwise)

`pt` - the initial location of the Point to be rotated

`ctr` - the Point that is the center of rotation

getRotatedLocation

```
public static java.awt.geom.Point2D.Double getRotatedLocation(double theta,
         java.awt.geom.Point2D.Double pt,
         java.awt.geom.Point2D.Double ctr)
```

computes the new location of a point after it has been rotated.

Parameters:

`theta` - the angle in radians of rotation (remember Java rotates clockwise)

`pt` - the initial location of the Point2D#Double to be rotated

`ctr` - the Point2D#Double that is the center of rotation

getNeedsTarget

```
public boolean getNeedsTarget()
```

Returns true if connections started from this ConnectingPt will allow connections to whole components. If false, connections started from this ConnectingPt require a ConnectingPt as a target.

setNeedsTarget

```
public void setNeedsTarget(boolean val_)
```

Sets whether connections started from this ConnectingPt will allow connections to whole components. If false, connections started from this ConnectingPt require a ConnectingPt as a target.

com.cafean.client.ui
Class ConnectionSetPanel

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-javax.swing.JComponent
            |
            +-javax.swing.JPanel
               |
               +-com.cafean.client.ui.ConnectionSetPanel
```

All Implemented Interfaces:
java.io.Serializable, java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, HasGetTransferHandler, java.io.Serializable, javax.accessibility.Accessible

---

public class **ConnectionSetPanel**
extends JPanel
implements javax.accessibility.Accessible, java.io.Serializable, HasGetTransferHandler, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, java.io.Serializable

A panel for displaying a set of connections to a given component.

Fields inherited from class `javax.swing.JComponent`
TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class `java.awt.Component`
BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface `java.awt.image.ImageObserver`
ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Constructor Summary

public	`ConnectionSetPanel()` Creates new form ConnectionsPanel
public	`ConnectionSetPanel(AbstractComponent component, Connection[] connections)` Creates new form ConnectionsPanel

Method Summary

void	`addConnectionSelectionListener(ConnectionSelectionListener listener)` Adds a selection listener to the list of listeners in this panel

void	clearSelection() Clears the currently selected connections
void	init(AbstractComponent component, Connection[] connections) Initializes this panel for use in displaying the given connections
void	refresh(Connection[] connections) Refreshes this panel with the given Connection list.
void	removeConnectionSelectionListener(ConnectionSelectionListener listener) Removes a selection listener from the list of listeners in this panel

Methods inherited from class javax.swing.JPanel

getAccessibleContext, getUI, getUIClassID, setUI, updateUI

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, disable, enable, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getActionForKeyStroke, getActionMap, getAlignmentX, getAlignmentY, getAncestorListeners, getAutoscrolls, getBaseline, getBaselineResizeBehavior, getBorder, getBounds, getClientProperty, getComponentPopupMenu, getConditionForKeyStroke, getDebugGraphicsOptions, getDefaultLocale, getFontMetrics, getGraphics, getHeight, getInheritsPopupMenu, getInputMap, getInputMap, getInputVerifier, getInsets, getInsets, getListeners, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getPopupLocation, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getTransferHandler, getUIClassID, getVerifyInputWhenFocusTarget, getVetoableChangeListeners, getVisibleRect, getWidth, getX, getY, grabFocus, isDoubleBuffered, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingForPrint, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, print, printAll, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, requestFocus, requestFocusInWindow, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setActionMap, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setComponentPopupMenu, setDebugGraphicsOptions, setDefaultLocale, setDoubleBuffered, setEnabled, setFocusTraversalKeys, setFont, setForeground, setInheritsPopupMenu, setInputMap, setInputVerifier, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setTransferHandler, setVerifyInputWhenFocusTarget, setVisible, unregisterKeyboardAction, update, updateUI

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class `java.awt.Component`

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.awt.image.ImageObserver`

imageUpdate

Methods inherited from interface `java.awt.MenuContainer`

getFont, postEvent, remove

Methods inherited from interface `javax.swing.TransferHandler.HasGetTransferHandler`

getTransferHandler

Methods inherited from interface javax.accessibility.Accessible
getAccessibleContext

Constructors

ConnectionSetPanel

```
public ConnectionSetPanel()
```

Creates new form ConnectionsPanel

ConnectionSetPanel

```
public ConnectionSetPanel(AbstractComponent component,
                          Connection[] connections)
```

Creates new form ConnectionsPanel

Methods

init

```
public void init(AbstractComponent component,
        Connection[] connections)
```

Initializes this panel for use in displaying the given connections

clearSelection

```
public void clearSelection()
```

Clears the currently selected connections

addConnectionSelectionListener

```
public void addConnectionSelectionListener(ConnectionSelectionListener listener)
```

Adds a selection listener to the list of listeners in this panel

removeConnectionSelectionListener

```
public void removeConnectionSelectionListener(ConnectionSelectionListener listener)
```

Removes a selection listener from the list of listeners in this panel

refresh

```
public void refresh(Connection[] connections)
```

Refreshes this panel with the given Connection list.

com.cafean.client.ui
Class CreateViewsDialog

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-java.awt.Window
            |
            +-java.awt.Dialog
               |
               +-javax.swing.JDialog
                  |
                  +-com.cafean.client.ui.CreateViewsDialog
```

All Implemented Interfaces:
java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, javax.accessibility.Accessible, HasGetTransferHandler, RootPaneContainer, javax.accessibility.Accessible, WindowConstants

---

public class **CreateViewsDialog**
extends JDialog

A dialog intended for creating views for a newly imported model. This dialog assumes that the model contains no existing views.

Fields inherited from class java.awt.Dialog
DEFAULT_MODALITY_TYPE

Fields inherited from class java.awt.Component
BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver
ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Fields inherited from interface javax.swing.WindowConstants
DISPOSE_ON_CLOSE, DO_NOTHING_ON_CLOSE, EXIT_ON_CLOSE, HIDE_ON_CLOSE

Constructor Summary

public	CreateViewsDialog(java.awt.Frame parent, AbstractModel model)
public	CreateViewsDialog(java.awt.Frame parent, AbstractModel model, Category[] rootCats) Creates new CreateViewsDialog to allow the user to select which component categories to create views for.

Method Summary

void	saveSelected()
void	setVisible(boolean b)

Methods inherited from class javax.swing.JDialog

getAccessibleContext, getContentPane, getDefaultCloseOperation, getGlassPane, getGraphics, getJMenuBar, getLayeredPane, getRootPane, getTransferHandler, isDefaultLookAndFeelDecorated, remove, repaint, setContentPane, setDefaultCloseOperation, setDefaultLookAndFeelDecorated, setGlassPane, setJMenuBar, setLayeredPane, setLayout, setTransferHandler, update

Methods inherited from class java.awt.Dialog

addNotify, getAccessibleContext, getModalityType, getTitle, hide, isModal, isResizable, isUndecorated, setModal, setModalityType, setResizable, setTitle, setUndecorated, setVisible, show, toBack

Methods inherited from class java.awt.Window

addNotify, addPropertyChangeListener, addPropertyChangeListener, addWindowFocusListener, addWindowListener, addWindowStateListener, applyResourceBundle, applyResourceBundle, createBufferStrategy, createBufferStrategy, dispose, getAccessibleContext, getBufferStrategy, getFocusableWindowState, getFocusCycleRootAncestor, getFocusOwner, getFocusTraversalKeys, getGraphicsConfiguration, getIconImages, getInputContext, getListeners, getLocale, getModalExclusionType, getMostRecentFocusOwner, getOwnedWindows, getOwner, getOwnerlessWindows, getToolkit, getWarningString, getWindowFocusListeners, getWindowListeners, getWindows, getWindowStateListeners, hide, isActive, isAlwaysOnTop, isAlwaysOnTopSupported, isFocusableWindow, isFocusCycleRoot, isFocused, isLocationByPlatform, isShowing, pack, postEvent, removeNotify, removeWindowFocusListener, removeWindowListener, removeWindowStateListener, reshape, setAlwaysOnTop, setBounds, setBounds, setCursor, setFocusableWindowState, setFocusCycleRoot, setIconImage, setIconImages, setLocationByPlatform, setLocationRelativeTo, setMinimumSize, setModalExclusionType, setSize, setSize, setVisible, show, toBack, toFront

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class `java.awt.Component`

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.awt.image.ImageObserver`

imageUpdate

Methods inherited from interface `java.awt.MenuContainer`

getFont, postEvent, remove

Methods inherited from interface `javax.accessibility.Accessible`

getAccessibleContext

Methods inherited from interface javax.accessibility.Accessible
getAccessibleContext

Methods inherited from interface javax.swing.RootPaneContainer
getContentPane, getGlassPane, getLayeredPane, getRootPane, setContentPane, setGlassPane, setLayeredPane

Methods inherited from interface javax.swing.TransferHandler.HasGetTransferHandler
getTransferHandler

Constructors

CreateViewsDialog

```
public CreateViewsDialog(java.awt.Frame parent,
                         AbstractModel model)
```

CreateViewsDialog

```
public CreateViewsDialog(java.awt.Frame parent,
                         AbstractModel model,
                         Category[] rootCats)
```

Creates new CreateViewsDialog to allow the user to select which component categories to create views for.

Methods

setVisible

```
public void setVisible(boolean b)
```

saveSelected

```
public void saveSelected()
```

com.cafean.client.ui
Interface DeckWriter

public interface **DeckWriter**
extends

This interface must be implemented by any class used for exporting ASCII decks. This allows for the ModelEditor to export ascii decks indipendently of the plugin.

Method Summary

boolean	performExport(boolean exportCheck) This performs the export of an ascii deck.

Methods

performExport

```
public boolean performExport(boolean exportCheck)
```

This performs the export of an ascii deck.

Parameters:

`exportCheck` - the boolean flag to turn on error checking.

Returns:

true if the export was successfull

com.cafean.client.ui
Class DrawnComponent

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-javax.swing.JComponent
            |
            +-com.cafean.client.ui.DrawnComponent
```

All Implemented Interfaces:
AnimationBeanGenerator, ComponentListener, StateEditable, java.awt.event.MouseMotionListener, java.awt.event.MouseListener, Cloneable, java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, HasGetTransferHandler, java.io.Serializable

Direct Known Subclasses:
DrawnViewComponent, DrawnConnection, DrawnUserValue

---

public abstract class **DrawnComponent**
extends JComponent
implements java.io.Serializable, HasGetTransferHandler, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, Cloneable, java.awt.event.MouseListener, java.awt.event.MouseMotionListener, StateEditable, ComponentListener, AnimationBeanGenerator

The DrawnComponent is a renderer for an `AbstractComponent`. It contains all of the information necessary to draw a representation of an `AbstractComponent` inside a `DrawnView`, or a `NodeViewPanel`.

The DrawnComponent is a `ComponentListener` on the `AbstractComponent` that it renders. This allows the DrawnComponent to update itself based on changes made to a component.

A DrawnComponent may be comprised of smaller `shapes` stored in an array. This is primarily used by Hydraulic Components but is available for any DrawnComponent to use.

Field Summary

`public static final`	`BOTTOM` Oriented towards the bottom of the screen Value: **2**
`public static final`	`CENTER` Align in the direct center. Value: **5**
`public static final`	`CENTER_H` align in the center horizontally. Value: **7**

public static final	CENTER_V Align in the center vertically. Value: **6**
public static final	CIRCLE Indicates the Pad for the ConnectingPt should be a Circle Value: **0**
public static final	CROSSHATCH Indicates the Pad for the ConnectingPt should be a Crosshatch Value: **4**
public static final	DIAMOND Indicates the Pad for the ConnectingPt should be a Diamond Value: **3**
public static final	DOWN Oriented towards the bottom of the screen Value: **2**
public static final	LEFT Oriented to the left hand of the screen Value: **0**
public static final	max_positions The number of different orientation enumerations. Value: **4**
public static final	NONE No Pad type is needed for the ConnectingPt Value: **-1**
public static final	PIXELS_P_METER This determines how many pixels are needed to display a single meter in length Value: **25.0**
public static final	RIGHT Oriented to the right-hand of the screen Value: **1**
public static final	SEGMENT_BOTH Indicates a segment has both an inlet and outlet ConnectingPt. Value: **3**
public static final	SEGMENT_INLET Indicates a segment has only an inlet ConnectingPt. Value: **1**
public static final	SEGMENT_NONE Indicates a segment has neither an inlet nor an outlet. Value: **0**

public static final	SEGMENT_OUTLET Indicates a segment has only an outlet ConnectingPt Value: **2**
public static final	SEGMENT_SPECIAL Indicates a segment a special meaning. Value: **4**
public static final	SQUARE Indicates the Pad for the ConnectingPt should be a Crosshatch Value: **5**
public static final	TOP Oriented towards the top of the screen Value: **3**
public static final	TRIANGLE Indicates the Pad for the ConnectingPt should be a Triangle Value: **2**
public static final	UP Oriented towards the top of the screen Value: **3**

Fields inherited from class javax.swing.JComponent

TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Fields inherited from interface javax.swing.undo.StateEditable

RCSID

Constructor Summary

public	DrawnComponent(AbstractComponent c) Creates a default DrawnComponent of the provided AbstractComponent centered at (0,0) with an angle of 0 and an orientation of RIGHT

Method Summary

void	addNotify()

boolean	`canBeResized()` All components can be resized as a default.
void	`Clear()` Clears the current DrawnComponent's connections and labels
void	`clearLinks()` **Deprecated.** *DrawnLinks have been converted to DrawnConnections.*
Object	`clone(AbstractComponent cb)` Creates a clone of a DrawnComponent, including its links and connections
void	`componentChanged(ComponentChangedEvent evt)` When a DrawnComponent's component changes, the drawn component needs to call initDrawing on itself
void	`componentConnected(Connection con)` This should make sure that the ZoomablePanel tries to add the drawnLink to the view when a connection is established.
void	`componentDeleted()` When a DrawnComponent's component is deleted, the Drawn component needs to remove itself from the view.
void	`componentDisconnected(Connection con)` This should make sure that the zoomable panel tries to remove the drawnLink from the view when a connection is removed.
void	`connectLinks()` **Deprecated.** *DrawnLinks are now DrawnConnection objects*
boolean	`contains(double x, double y)` Checks whether this component "contains" the specified coordinates where *x* and *y* are defined to be relative to the coordinate system of this component.
boolean	`contains(int x, int y)`
boolean	`contains(java.awt.Point p)`
java.awt.Shape	`createBorderRegion(java.awt.Shape shape, int face, double shift)` Creates a border shape on the specified face of the given shape
java.awt.Shape	`createCenterShape(java.awt.Shape shape, int face, double shift)` Creates a stripe down the center of a given shape
static void	`createConnectionPrototypes()` Creates the various archtypes of Connection Shapes to be used at connection points
ConnectingPt	`createConnectionPt(double x, double y, double angle, int type, int face, int mark, int link_num, ConnectionData data)` Initializes and creates a new connecting point at the specified location and having the specified characteristics

JComponent[]	createDisplayBeans(int pixelsPerMeter, double widthScaleFactor, ClassLoader loader) Creates and configures a set of AbstractDisplayBeans for use in displaying this DrawnComponent in an animatable view.
JPopupMenu	createPopupMenu() Creates a custom popup context menu for this DrawnComponent that includes an optional Actions menu from sub components.
TemplateEntry	createTemplateEntry() Creates a new TemplateEntry for this DrawnComponent that stores all of the location and state data for this DranwComponent.
void	disconnectAllMyLinks() **Deprecated.** *DrawnLinks have been converted to DrawnConnections*
void	draw(java.awt.Graphics2D g, boolean selected) Tints the current DrawnComponent according to its state.
void	drawLabelStrings(java.awt.Shape obj, java.awt.Graphics2D g, int offset) This draws all of the Strings to at the center of the Object.
void	flip() Flips the orientation of a DrawnComponent to the next one in the given rotation.
java.awt.Point	forTransformPoint(java.awt.Point pt) Transforms a point from LocalSpace coordinates (inlet) up to ComponentSpace coordinates (upper left corner).
BeanBox	getBeanBox() Retrieves the BeanBox that contains this DrawnComponent or null if this DrawnComponent exists outside of a BeanBox.
int	getClockwiseFace(int face) Retrieves the face value that is 90 degrees clockwise from the given face.
AbstractComponent	getComponent() Each DrawnComponent is a rendering object for a specific AbstractComponent.
int	getComponentID() Retrieves the ident of this drawn component's component
java.awt.Point	getConnectingLocation(ConnectingPt point) Returns the absolute screen location of the given ConnectingPt.
ConnectingPt	getConnectingPt(ConnectionData data) Retrieves the ConnectingPt for the given ConnectionData or null of none is found.
ConnectingPt	getConnectingPt(int i) Returns a ConnectingPt for a given int.
ConnectingPt	getConnectingPtAt(double x, double y) Retrieves the connector at the given coordinates

int	getConnectSize() Retrieves the number of ConnectingPts the user specifies to be within the Connections vector of the current DrawnComponent.
int	getCounterFace(int face) Retrieves the face value that is 90 degrees counter-clockwise from the given face.
int	getCrossflowIndex(ConnectingPt cp) Gets the number of ConnectingPt components that are crossflow Points that occur before the given ConnectingPt.
Action[]	getCustomPopupActions() Retrieves the curstom popup actions from this drawn component's AbstractComponent.
Vector	getCustomPopupItems() Returns the custom popup items of this DrawnComponent's target component
Vector	getCustomPopupItems(java.awt.event.MouseEvent evt) Returns the custom popup items of this DrawnComponent's target component as returned by getCustomPopupItems() as well as those popup items that are appropriate for the given mouse event.
float	getDefaultDrawLength() The Default Draw Length for a DrawnComponent is 50 pixels.
float	getDefaultDrawWidth() The Default Draw Width for a DrawnComponent is 50 pixels.
double	getDrawAngle() Gets the angle, in radians describing the angle of this DrawnComponent
int	getDrawingFace(java.awt.geom.Point2D.Double pt) Returns the face the DrawnComponent is connected to when the object is a source, ie, LEFT RIGHT TOP BOTTOM.
java.awt.Shape	getDrawingObject() Gets the Shape that should be drawn when this DrawnComponent is painted.
int	getFaceByAngle(double ang, boolean invert) Determines the appropriate orientation for a given angle.
java.awt.Color	getFillColor() Returns the color to be used for filling this DrawnComponent's normal shape.
GlassPanel	getGlassPane() Retrieves the GlassPanel from the ZoomablePanel that contains this DrawnComponent.
int	getGroupID() Retrieves the ID of this object's visual group.
int	getHandleSize() This gets the user defined size of the handles from the user preferences.
static java.awt.Color	getIndicatorColor() Gets the user preference for the Connection Color.

double	getLength() The length of a drawn component is the same as it's height.
double	getLenScaleFactor() This gets the Length Scale Factor from the AbstractModel.
float	getMaxHeight() This gets the maximum height for this DrawnComponent.
float	getMaxWidth() This gets the maximum width for this DrawnComponent.
float	getMinWidth() This gets the minimum width for this DrawnComponent.
java.awt.Shape	getMirrorImageShape(java.awt.Shape shape, int mirrorCode) Creates a mirror image of a given shape s
java.awt.Shape	getNormalObj() Gets the Shape that is used to drawn the component in the DrawnPanel
int	getNumberConnections() Retrieves the number of ConnectingPts the user specifies to be within the Connections vector of the current DrawnComponent.
int	getOppositeFace(int face) Retrieves the face value that is the exact opposite of the given face.
int	getOrientation() Returns the orientation of the DrawnComponent
JMenu	getOrientationMenu() Creates a menu appropriate for selecting the desired orientation for this drawn component.
static String	getOrientationName(int orientation) Retrieves the name of the given orientation.
java.awt.Container	getParent() This either gets the actual parent of the DrawnComponent, if it exists inside a Container.
java.awt.Dimension	getPreferredSize() Provides the preferred size for this component.
ConnectingPt	getSelectedConnector(double x, double y) Retrieves the connector at the given coordinates
ConnectingPt	getSelectedDropZone(double x, double y) Retrieves the drop zone at the given coordinates
DrawnSubComponent	getSubComponentAt(int x, int y) Returns the sub-component at the given coordinates, or null of no sub-component exists at that location.
String	getToolTipText()

String	getToolTipText(int x, int y)
double	getWidthScaleFactor() This gets the width scale factor from the AbstractModel.
double	getX_Pos() Gets the X coordinate of the center point of this DrawnComponent.
double	getXDistBetweenCPs() This distance is used in the layout algorithm to allow for drawn objects which are not square, such as segmented pipes.
double	getXDistBetweenXflowCPs() This distance is used in the layout algorithm to allow for drawn objects which are not square, such as segmented pipes.
double	getY_Pos() Gets the Y coordinate of the center point of this DrawnComponent.
ZoomablePanel	getZoomablePanel() Retrieves the ZoomablePanel that contains this DrawnComponent or null if this DrawnComponent exists outside of a ZoomablePanel.
boolean	hasSubComponents() Returns true if this DrawnComponent has DrawnSubComponents contained within it
void	InitDrawing() Initializes and scales this component to prepare it for painting.
boolean	isAutoScale() Getter for property autoScale that determines whether scaleIt calls scale this Drawn Component.
boolean	isDrawBadges() Getter for property drawBadges.
boolean	isGroupIDValid() Returns true if this object's visual group ID is valid.
boolean	isObjectInsideBounds(java.awt.geom.Rectangle2D.Double rect) Determines if an object is inside the given rectangle
boolean	isPlenumShaped() Plenums are traditionally shaped differently from other components.
boolean	isPosnSet() Determines if the DrawnComponent's position has been initialized.
boolean	isScalable() All components can be scaled as a default.
static boolean	isSegmentSet(int test, int segment) Returns true if the given segment bit set is included in the given test value.

boolean	isSelected() Getter for property selected.
boolean	isValveShaped() Valves are traditionally shaped differently from other components.
void	loadDrawnComponent(com.apt.xdr.PibBlock block) This function loads a DrawnComponent from a DrawnComponentRec.
void	mouseClicked(java.awt.event.MouseEvent e)
void	mouseDragged(java.awt.event.MouseEvent evt)
void	mouseEntered(java.awt.event.MouseEvent e)
void	mouseExited(java.awt.event.MouseEvent e)
void	mouseMoved(java.awt.event.MouseEvent e)
void	mousePressed(java.awt.event.MouseEvent e)
void	mouseReleased(java.awt.event.MouseEvent e)
void	moveRel(double x, double y, boolean last) Move the drawing relative to its previous position
void	moveTo(double x, double y, boolean last) Move the drawing so the center is the specified position
void	paint(java.awt.Graphics g) Paints this component.
void	paintComponent(java.awt.Graphics g) Tints the current DrawnComponent according to its state.
void	print(java.awt.Graphics g) Prints this component.
void	readTemplateEntry(TemplateEntry entry) Sets the data on this DrawnComponent from a TemplateEntry read in from a view template file.
void	removeNotify()
void	repositionLinks() Updates any connection point location related data for this drawing (and potentially its connected neighbors) to allow drawn connections to be redirected to the appropriate locations.
void	repositionLinks(boolean last) **Deprecated.** *DrawnLinks have been converted to DrawConnections.*

void	resetPosition() resetPosition can be used to set the correct positions of components when they become visible, such as when the display layer is changed.
void	restoreState(Hashtable state) Restore the state of the bean from an earlier edit.
java.awt.Point	revTransformPoint(java.awt.Point pt) Transforms a point from ComponentSpace coordinates (upper left corner) down to scaled localSpace coordinates (inlet).
java.awt.Shape	rotateTo(double theta, java.awt.Shape s, java.awt.geom.Point2D.Double pt, Vector ConnectPts) Rotate the specified shape around a given point by the given angle.
java.awt.Shape	rotateTo(double theta, java.awt.Shape s, java.awt.Point pt, Vector ConnectPts) Rotate the specified shape around a given point by the given angle.
boolean	scaleIt() Scale the drawing so that it reflects the true relative size
void	setAutoScale(boolean autoScale) Setter for property autoScale that determines whether scaleIt calls scale this Drawn Component.
void	setBackupComponent(AbstractComponent backup) When a DrawnComponent is rendering a cloneed object for displaying changes, the backup component is the source of external data for that component.
void	setBounds(int x, int y, int width, int height) This is used to set the current scale factors on a component when the bounds are changed by the user.
void	setComponent(AbstractComponent comp) This sets the AbstractComponent that is being rendered by this DrawnComponent.
void	setDrawAngle() sets the drawAngle value depending on the DrawnComponent's current orientation
void	setDrawBadges(boolean drawBadges) Setter for property drawBadges.
void	setDrawHeight(double l) Sets the drawing height of the DrawnComponent.
void	setDrawWidth(double w) Sets the drawing width of the DrawnComponent.
void	setEqualTo(DrawnComponent dc) Sets the position and orientation of the current DrawnComponent to be the same as the DrawnComponent provided as an argument.
void	setGroupID(int groupID) Sets the ID of this object's visual group.

void	setGroupIDValid(boolean valid) Sets the flag that is used to determine if this object's visual group ID must be reconnected after a paste operation.
void	setLabelString(String str, int index) Adds or replaces a String in the labels array.
void	setLenScaleFactor(double factor)
void	setOrientation(int orientation) Sets the orientation of the current DrawnComponent to the specified orientation.
void	setOrientationByAngle(double a) Sets the orientation by an angle measure, in radians
void	setOrientationConstrained(int orientation) Sets the orientation and produces an undoable edit event which is added to the undo stack.
void	setParent(java.awt.Container parent) Setter for the parent value of this DrawnComponent.
void	setPixelsPerMeter(int ppm)
void	setSelected(boolean selected) Setter for property selected.
boolean	setSizeTo(double len, double wid) Resets the drawing size to the given dimension and then reinitializes the DrawnComponent.
void	setWidthScaleFactor(double factor)
void	setX_Pos(double position) Sets the location of this DrawnComponent's center to the given position in the X dimension.
void	setY_Pos(double position) Sets the location of this DrawnComponent's center to the given position in the Y dimension.
boolean	showConnections()
com.apt.xdr.PibBlock	store(int viewNum) This function returns a PibBlock for a drawn component.
void	storeState(Hashtable state) Store the state of the bean to permit undo.
String	toString() The String produced is based on the toString of the component.
java.awt.Point	translateConnectionToScreen(ConnectingPt pt) This converts the center of a passed connecting Point into the coordinates on the zoomable panel.

java.awt.Point	translatePointToScreen(java.awt.Point point) This converts a java.awt.Point from local coordinates into coordinates on the zoomable panel.

Methods inherited from class javax.swing.JComponent

addAncestorListener, addNotify, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, disable, enable, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getActionForKeyStroke, getActionMap, getAlignmentX, getAlignmentY, getAncestorListeners, getAutoscrolls, getBaseline, getBaselineResizeBehavior, getBorder, getBounds, getClientProperty, getComponentPopupMenu, getConditionForKeyStroke, getDebugGraphicsOptions, getDefaultLocale, getFontMetrics, getGraphics, getHeight, getInheritsPopupMenu, getInputMap, getInputMap, getInputVerifier, getInsets, getInsets, getListeners, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getPopupLocation, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getTransferHandler, getUIClassID, getVerifyInputWhenFocusTarget, getVetoableChangeListeners, getVisibleRect, getWidth, getX, getY, grabFocus, isDoubleBuffered, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingForPrint, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, print, printAll, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, requestFocus, requestFocusInWindow, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setActionMap, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setComponentPopupMenu, setDebugGraphicsOptions, setDefaultLocale, setDoubleBuffered, setEnabled, setFocusTraversalKeys, setFont, setForeground, setInheritsPopupMenu, setInputMap, setInputVerifier, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setTransferHandler, setVerifyInputWhenFocusTarget, setVisible, unregisterKeyboardAction, update, updateUI

Methods inherited from class java.awt.Container

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class java.awt.Component

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.awt.image.ImageObserver`

imageUpdate

Methods inherited from interface `java.awt.MenuContainer`

getFont, postEvent, remove

Methods inherited from interface `javax.swing.TransferHandler.HasGetTransferHandler`

getTransferHandler

Methods inherited from interface java.awt.event.MouseListener
mouseClicked, mouseEntered, mouseExited, mousePressed, mouseReleased

Methods inherited from interface java.awt.event.MouseMotionListener
mouseDragged, mouseMoved

Methods inherited from interface javax.swing.undo.StateEditable
restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.ComponentListener
componentChanged, componentConnected, componentDeleted, componentDisconnected

Methods inherited from interface com.cafean.client.ui.AnimationBeanGenerator
createDisplayBeans

Fields

NONE

public static final int **NONE**

No Pad type is needed for the ConnectingPt
Constant value: **-1**

CIRCLE

public static final int **CIRCLE**

Indicates the Pad for the ConnectingPt should be a Circle
Constant value: **0**

TRIANGLE

public static final int **TRIANGLE**

Indicates the Pad for the ConnectingPt should be a Triangle
Constant value: **2**

DIAMOND

public static final int **DIAMOND**

Indicates the Pad for the ConnectingPt should be a Diamond
Constant value: **3**

CROSSHATCH

`public static final int CROSSHATCH`

Indicates the Pad for the ConnectingPt should be a Crosshatch
Constant value: **4**

SQUARE

`public static final int SQUARE`

Indicates the Pad for the ConnectingPt should be a Crosshatch
Constant value: **5**

SEGMENT_NONE

`public static final int SEGMENT_NONE`

Indicates a segment has neither an inlet nor an outlet.
Constant value: **0**

SEGMENT_INLET

`public static final int SEGMENT_INLET`

Indicates a segment has only an inlet ConnectingPt.
Constant value: **1**

SEGMENT_OUTLET

`public static final int SEGMENT_OUTLET`

Indicates a segment has only an outlet ConnectingPt
Constant value: **2**

SEGMENT_BOTH

`public static final int SEGMENT_BOTH`

Indicates a segment has both an inlet and outlet ConnectingPt.
Constant value: **3**

SEGMENT_SPECIAL

`public static final int SEGMENT_SPECIAL`

Indicates a segment a special meaning.
Constant value: **4**

LEFT

`public static final int LEFT`

Oriented to the left hand of the screen

Constant value: **0**

RIGHT

```
public static final int RIGHT
```

Oriented to the right-hand of the screen
Constant value: **1**

DOWN

```
public static final int DOWN
```

Oriented towards the bottom of the screen
Constant value: **2**

BOTTOM

```
public static final int BOTTOM
```

Oriented towards the bottom of the screen
Constant value: **2**

TOP

```
public static final int TOP
```

Oriented towards the top of the screen
Constant value: **3**

UP

```
public static final int UP
```

Oriented towards the top of the screen
Constant value: **3**

---

max_positions

```
public static final int max_positions
```

The number of different orientation enumerations.
Constant value: **4**

CENTER

```
public static final int CENTER
```

Align in the direct center.
Constant value: **5**

CENTER_V

```
public static final int CENTER_V
```

Align in the center vertically.
Constant value: **6**

CENTER_H

```
public static final int CENTER_H
```

align in the center horizontally.
Constant value: **7**

PIXELS_P_METER

```
public static final double PIXELS_P_METER
```

This determines how many pixels are needed to display a single meter in length
Constant value: **25.0**

Constructors

DrawnComponent

```
public DrawnComponent(AbstractComponent c)
```

Creates a default DrawnComponent of the provided AbstractComponent centered at (0,0) with an angle of 0 and an orientation of RIGHT

Parameters:

`c` - the AbstractComponent of the component to be modeled

Methods

isSegmentSet

```
public static boolean isSegmentSet(int test,
        int segment)
```

Returns true if the given segment bit set is included in the given test value.

Parameters:

`test` - the bitset to check the segment mask against
`segment` - the SEGMENT_* mask to check against test

---

setBackupComponent

```
public void setBackupComponent(AbstractComponent backup)
```

When a DrawnComponent is rendering a cloneed object for displaying changes, the backup component is the source of external data for that component. For example, the Backup component is used to display externally defined edge angles.

Parameters:

`backup` - the AbstractComponent that contains the original component data.

---

getOrientationName

```
public static String getOrientationName(int orientation)
```

Retrieves the name of the given orientation.

Parameters:
`orientation` - the enumeration orientation type.

Returns:
the String that best describes that orientation.

InitDrawing

```
public void InitDrawing()
```

Initializes and scales this component to prepare it for painting.

(Not cell length, just length) All the Connections of the Component are examined. If the ConnectionData from the Connection has a ConnectingPt, that ConnectingPt is set selected.

DrawnComponent derivatives should call this as the last step in their InitDrawing method.

---

createDisplayBeans

```
public JComponent[] createDisplayBeans(int pixelsPerMeter,
        double widthScaleFactor,
        ClassLoader loader)
```

Creates and configures a set of `AbstractDisplayBeans` for use in displaying this DrawnComponent in an animatable view.

Either a set of beans or a single bean may be returned. If multiple beans are returned they should be positioned appripropriately in relation to one another.

Note: The base implementation returns null.

Returns:
a javax.swing.JComponent[] array or null of no beans are available for this DrawnComponent type

---

getIndicatorColor

```
public static java.awt.Color getIndicatorColor()
```

Gets the user preference for the Connection Color.

Returns:
the Color selected by the user for the Connections

---

getGroupID

```
public int getGroupID()
```

Retrieves the ID of this object's visual group.

NOTE: For this feature to be used this object must implement `com.cafean.client.ui.util.Groupable`.

---

setGroupID

```
public void setGroupID(int groupID)
```

Sets the ID of this object's visual group.

NOTE: For this feature to be used this object must implement `com.cafean.client.ui.util.Groupable`.

---

isGroupIDValid

`public boolean **isGroupIDValid**()`

Returns true if this object's visual group ID is valid. Valid visual group IDs are not reconnected during a paste operation.

NOTE: For this feature to be used this object must implement `com.cafean.client.ui.util.Groupable`.

---

setGroupIDValid

`public void **setGroupIDValid**(boolean valid)`

Sets the flag that is used to determine if this object's visual group ID must be reconnected after a paste operation.

NOTE: For this feature to be used this object must implement `com.cafean.client.ui.util.Groupable`.

---

getLenScaleFactor

`public double **getLenScaleFactor**()`

This gets the Length Scale Factor from the AbstractModel.

Returns:
the double length scale factor from the AbstractModel.

See Also:
`AbstractModel.getLenScaleFactor()`

---

getWidthScaleFactor

`public double **getWidthScaleFactor**()`

This gets the width scale factor from the AbstractModel.

Returns:
the double width scale factor from the AbstractModel.

See Also:
`AbstractModel.getLenScaleFactor()`

---

setLenScaleFactor

`public void **setLenScaleFactor**(double factor)`

---

setWidthScaleFactor

`public void **setWidthScaleFactor**(double factor)`

setPixelsPerMeter

```
public void setPixelsPerMeter(int ppm)
```

isValveShaped

```
public boolean isValveShaped()
```

Valves are traditionally shaped differently from other components. This flag indicates that this DrawnComponent is drawn like a valve. This is used by the Organizer.

Returns:
false as a default.

isPlenumShaped

```
public boolean isPlenumShaped()
```

Plenums are traditionally shaped differently from other components. This flag indicates that this DrawnComponent is drawn like a plenum. This is used by the Organizer.

Returns:
false as a default.

createConnectionPrototypes

```
public static void createConnectionPrototypes()
```

Creates the various archtypes of Connection Shapes to be used at connection points

setBounds

```
public void setBounds(int x,
        int y,
        int width,
        int height)
```

This is used to set the current scale factors on a component when the bounds are changed by the user. If autoscale is on and the model preference for autoscaling components is on the scale factor is set to 1, otherwise it is calculated based on the minimum bounds of the component.

Parameters:
`x` - the upper left corner X position.
`y` - the upper left corner Y position.
`width` - the width of the bounds.
`height` - the height of the bounds. {@inheritDoc}

getNormalObj

```
public java.awt.Shape getNormalObj()
```

Gets the Shape that is used to drawn the component in the DrawnPanel

Returns:

Shape the Shape rendered by InitDrawning.

---

toString

```
public String toString()
```

The String produced is based on the toString of the component.

Returns:

the String for quickly identifying this Drawing

---

clone

```
public Object clone(AbstractComponent cb)
```

Creates a clone of a DrawnComponent, including its links and connections

Parameters:

`cb` - the AbstractComponent of the DrawnComponent that is being cloned

Returns:

o an Object representing the clone of the DrawnComponent

---

createConnectionPt

```
public ConnectingPt createConnectionPt(double x,
        double y,
        double angle,
        int type,
        int face,
        int mark,
        int link_num,
        ConnectionData data)
```

Initializes and creates a new connecting point at the specified location and having the specified characteristics

Parameters:

`data` - the ConnectionData that defines the new ConnectingPt actual connection information.

`x` - the x-value of the coordinate of the desired connection point

`y` - the y-value of the coordiinate of the desired connection point

`angle` - the desired drawn angle, in radians, of the shape which will be drawn at this connection point

`type` - the type of ConnectingPt this is to represent. Should be one of the following:

- ConnectingPt.INLET_DROP_ZONE
- ConnectingPt.OUTLET_DROP_ZONE
- ConnectingPt.CROSSFLOW_DROP_ZONE
- ConnectingPt.INLET_MOVEABLE_CONNECTOR
- ConnectingPt.OUTLET_MOVEABLE_CONNECTOR
- ConnectingPt.STATIC_CONNECTOR
- ConnectingPt.INVISIBLE_CONNECTOR

`face` - the face on which the point is to be drawn

`mark` - the shape to be drawn at this connecting point. Should be one of the following values:

- DrawnComponent.NONE
- DrawnComponent.CIRCLE
- (usually for INLET_MOVEABLE_CONNECTOR and OUTLET_MOVEABLE_CONNECTOR)
- DrawnComponent.TRIANGLE
- (usually for INLET_DROP_ZONE and OUTLET_DROP_ZONE)
- DrawnComponent.DIAMOND
- DrawnComponent.CROSSHATCH

`link_num` - the number of the link associated with this point; -1 if there is no link associated with this point

Returns:
The ConnectingPt that has been created.

---

setEqualTo

`public void` **`setEqualTo`**`(DrawnComponent dc)`

Sets the position and orientation of the current DrawnComponent to be the same as the DrawnComponent provided as an argument. The connections and the owner composite base are not changed.

Parameters:
`dc` - a DrawnComponent to which the current DrawnComponent's position and orientation are to be set

---

Clear

`public void` **`Clear`**`()`

Clears the current DrawnComponent's connections and labels

---

clearLinks

`public void` **`clearLinks`**`()`

Deprecated. *DrawnLinks have been converted to DrawnConnections.*

Clears all of the DrawnLinks off of this DrawnComponent

---

getDefaultDrawWidth

`public float` **`getDefaultDrawWidth`**`()`

The Default Draw Width for a DrawnComponent is 50 pixels.

Returns:
50

getDefaultDrawLength

`public float getDefaultDrawLength()`

The Default Draw Length for a DrawnComponent is 50 pixels.

Returns:
50

getLength

`public double getLength()`

The length of a drawn component is the same as it's height.

Returns:
the length of this drawn component

See Also:
`JComponent.getHeight()`

getDrawAngle

`public double getDrawAngle()`

Gets the angle, in radians describing the angle of this DrawnComponent

Returns:
the angle of this DrawnComponent.

getDrawingObject

`public java.awt.Shape getDrawingObject()`

Gets the Shape that should be drawn when this DrawnComponent is painted. This defaults to the current normalObject

Returns:
the Shape for drawing.

connectLinks

`public void connectLinks()`

Deprecated. *DrawnLinks are now DrawnConnection objects*

This reconnects the DrawnLinks of this DrawnComponent

setDrawAngle

`public void setDrawAngle()`

sets the drawAngle value depending on the DrawnComponent's current orientation

getToolTipText

```
public String getToolTipText()
```

getToolTipText

```
public String getToolTipText(int x,
          int y)
```

paintComponent

```
public void paintComponent(java.awt.Graphics g)
```

Tints the current DrawnComponent according to its state. Also draws the normalObj of the DrawnComponent, as well as the component's labels, connection points, and links

Parameters:

g - the Graphics2D object which will do the painting

paint

```
public void paint(java.awt.Graphics g)
```

Paints this component. Overridden here to prevent failures in drawn component painting code to cause the UI to become unresponsive.

print

```
public void print(java.awt.Graphics g)
```

Prints this component. Overridden here to prevent failures in drawn component painting code to cause the UI to become unresponsive.

getFillColor

```
public java.awt.Color getFillColor()
```

Returns the color to be used for filling this DrawnComponent's normal shape.

Returns:

a java.awt.Color object appropriate for filling this component.

draw

```
public void draw(java.awt.Graphics2D g,
          boolean selected)
```

Tints the current DrawnComponent according to its state. Also draws the normalObj of the DrawnComponent, as well as the component's labels, connection points, and links

Parameters:

`g` - the Graphics2D object which will do the painting

`selected` - indicates whether the current DrawnComponent is currently selected

---

drawLabelStrings

```
public void drawLabelStrings(java.awt.Shape obj,
          java.awt.Graphics2D g,
          int offset)
```

This draws all of the Strings to at the center of the Object.

Parameters:

`obj` - the Shape that is being drawn.

`g` - the Graphics2D object.

`offset` - the vertical offset for the strings.

---

setSizeTo

```
public boolean setSizeTo(double len,
          double wid)
```

Resets the drawing size to the given dimension and then reinitializes the DrawnComponent.

Parameters:

`len` - the new length in pixels.

`wid` - the new width in pixels

Returns:

true if the size given is valid.

See Also:

`InitDrawing()`

---

getXDistBetweenCPs

```
public double getXDistBetweenCPs()
```

This distance is used in the layout algorithm to allow for drawn objects which are not square, such as segmented pipes. The absolute value of the distance is returned to allow for the first layout pass when the object may or may not have been flipped to its final orientation yet.

Returns:

The distance in pixels between centers of the first and second ConnectingPts.

---

getXDistBetweenXflowCPs

```
public double getXDistBetweenXflowCPs()
```

This distance is used in the layout algorithm to allow for drawn objects which are not square, such as segmented pipes. The absolute value of the distance is returned to allow for the first layout pass when the object may or may not have been flipped to its final orientation yet.

Returns:

The distance in pixels between the third and fourth ConnectingPts.

scaleIt

```
public boolean scaleIt()
```

Scale the drawing so that it reflects the true relative size

Returns:
true if the component is scaled.

canBeResized

```
public boolean canBeResized()
```

All components can be resized as a default. Override this to return false if this comp should not be resizable.

Returns:
true if this DrawnComponent can be scaled.

isScalable

```
public boolean isScalable()
```

All components can be scaled as a default. Override this to return false if this comp should not be scalable.

Returns:
true if the DrawnComponent can be scaled.

moveTo

```
public void moveTo(double x,
                   double y,
                   boolean last)
```

Move the drawing so the center is the specified position

Parameters:
`x` - the new center x position
`y` - the new center y position
`last` - false while moving, true on the final move

moveRel

```
public void moveRel(double x,
                    double y,
                    boolean last)
```

Move the drawing relative to its previous position

Parameters:
`x` - the change in x position in pixels.
`y` - the change in y position in pixels.
`last` - true when this move is the final movement.

resetPosition

```
public void resetPosition()
```

resetPosition can be used to set the correct positions of components when they become visible, such as when the display layer is changed. It should be overloaded by classes that need it. In the base class it does nothing.

---

rotateTo

```
public java.awt.Shape rotateTo(double theta,
        java.awt.Shape s,
        java.awt.Point pt,
        Vector ConnectPts)
```

Rotate the specified shape around a given point by the given angle.

Parameters:

`theta` - the angular distance to rotate in radians.
`s` - the Shape to be rotated
`pt` - the Point around which the shape is rotated
`ConnectPts` - the vector of `ConnectingPt` that must also be moved.

Returns:

the Shape after it has been rotated.

---

rotateTo

```
public java.awt.Shape rotateTo(double theta,
        java.awt.Shape s,
        java.awt.geom.Point2D.Double pt,
        Vector ConnectPts)
```

Rotate the specified shape around a given point by the given angle.

Parameters:

`theta` - the angular distance to rotate in radians.
`s` - the Shape to be rotated
`pt` - the Point2D.Double around which the shape is rotated
`ConnectPts` - the vector of `ConnectingPt` that must also be moved.

Returns:

the Shape after it has been rotated.

---

getMirrorImageShape

```
public java.awt.Shape getMirrorImageShape(java.awt.Shape shape,
        int mirrorCode)
```

Creates a mirror image of a given shape s

Parameters:

`shape` - the Shape to mirror
`mirrorCode` - the direction to do the mirroring; same as orientation flags: 0 or 1 for left/right or 2 or 3 for top/bottom

Returns:

the mirrored Shape

createBorderRegion

```
public java.awt.Shape createBorderRegion(java.awt.Shape shape,
         int face,
         double shift)
```

Creates a border shape on the specified face of the given shape

Parameters:

`shape` - the shape which will have a border added
`face` - the face to add the border to
`shift` - width of border (pixels)

Returns:

the Shape representing the border region

createCenterShape

```
public java.awt.Shape createCenterShape(java.awt.Shape shape,
         int face,
         double shift)
```

Creates a stripe down the center of a given shape

Parameters:

`shape` - the shape which will have the center strip added
`face` - the face to add the border to
`shift` - width of border (pixels)

Returns:

the center Shape

isObjectInsideBounds

```
public boolean isObjectInsideBounds(java.awt.geom.Rectangle2D.Double rect)
```

Determines if an object is inside the given rectangle

Parameters:

`rect` - the bounding rectangle

Returns:

true if the object is inside the rectangle false otherwise.

contains

```
public boolean contains(java.awt.Point p)
```

contains

```
public boolean contains(int x,
         int y)
```

contains

```
public boolean contains(double x,
                        double y)
```

Checks whether this component "contains" the specified coordinates where *x* and *y* are defined to be relative to the coordinate system of this component.

Note that `getBounds()` includes the position of this component within the BeanBox where the given x and y should **not**.

Parameters:

`x` - the x coordinate
`y` - the y coordinate

Returns:

true if the given coordinates fall within the bounds of this DrawnComponent

getConnectingPtAt

```
public ConnectingPt getConnectingPtAt(double x,
                                      double y)
```

Retrieves the connector at the given coordinates

Parameters:

`x` - the x position
`y` - the y position

Returns:

the ConnectingPt at the given coordinates that is a connector

getSelectedConnector

```
public ConnectingPt getSelectedConnector(double x,
                                         double y)
```

Retrieves the connector at the given coordinates

Parameters:

`x` - the x position
`y` - the y position

Returns:

the ConnectingPt at the given coordinates that is a connector

getSelectedDropZone

```
public ConnectingPt getSelectedDropZone(double x,
                                        double y)
```

Retrieves the drop zone at the given coordinates

Parameters:

x - the x position
y - the y position

Returns:
the ConnectingPt at the given coordinates that is a drop zone

---

getCrossflowIndex

```
public int getCrossflowIndex(ConnectingPt cp)
```

Gets the number of ConnectingPt components that are crossflow Points that occur before the given ConnectingPt. Returns -1 if the ConnectingPt is not found.

Parameters:
cp - the ConnectingPoint.

Returns:
the number of crossflow ConnectingPts that were created before the given point.

---

repositionLinks

```
public void repositionLinks()
```

Updates any connection point location related data for this drawing (and potentially its connected neighbors) to allow drawn connections to be redirected to the appropriate locations.

This method is called at the end of each move operation by the select tool and by the @{link #moveTo} method.

Potential uses in derivatives may include calling InitDrawing on the connected neighbors of this drawing.

---

repositionLinks

```
public void repositionLinks(boolean last)
```

Deprecated. *DrawnLinks have been converted to DrawConnections.*

Adjusts the position of all links coming out of an object

Parameters:
last - true if this was the final move.

---

getDrawingFace

```
public int getDrawingFace(java.awt.geom.Point2D.Double pt)
```

Returns the face the DrawnComponent is connected to when the object is a source, ie, LEFT RIGHT TOP BOTTOM. Used for drawing a component on the main canvas.

Parameters:
pt - the argument point

Returns:
an int representing the face

---

flip

`public void flip()`

Flips the orientation of a DrawnComponent to the next one in the given rotation.

- LEFT
- RIGHT
- DOWN
- UP

---

getOrientation

`public int getOrientation()`

Returns the orientation of the DrawnComponent

Returns:

The enumeration of the current orientation of this DrawnComponent. Acceptable values are:

---

setOrientation

`public void setOrientation(int orientation)`

Sets the orientation of the current DrawnComponent to the specified orientation. If the int specified is invalid, the orientation will be set to RIGHT.

Parameters:

`orientation` - the int enumeration for the orientation.

See Also:

`getOrientation()`

---

setOrientationConstrained

`public void setOrientationConstrained(int orientation)`

Sets the orientation and produces an undoable edit event which is added to the undo stack. This allows for undoing setting the orientation of a Drawn component.

---

setOrientationByAngle

`public void setOrientationByAngle(double a)`

Sets the orientation by an angle measure, in radians

Parameters:

`a` - the angle measure, in radians

---

getFaceByAngle

```
public int getFaceByAngle(double ang,
        boolean invert)
```

Determines the appropriate orientation for a given angle. This is used to determine the angle of a Pad facing for a ConnectingPt.

Parameters:

angle - The angle in radians that is in question.
invert - Will return the oposite face if this is true.

Returns:
Calculates which face the angle is closest to representing. Possible values are:

- BOTTOM
- TOP
- RIGHT
- LEFT

---

getOppositeFace

```
public int getOppositeFace(int face)
```

Retrieves the face value that is the exact opposite of the given face.

Parameters:

face - the face enumeration.

Returns:
the oposite face. Possible values are:

- BOTTOM
- TOP
- RIGHT
- LEFT

---

getCounterFace

```
public int getCounterFace(int face)
```

Retrieves the face value that is 90 degrees counter-clockwise from the given face.

Parameters:

face - the face enumeration.

Returns:

the face 90 degrees counter-clockwize from the given face. Possible values are:

- BOTTOM
- TOP
- RIGHT
- LEFT

---

getClockwiseFace

```
public int getClockwiseFace(int face)
```

Retrieves the face value that is 90 degrees clockwise from the given face.

Parameters:
face - the face enumeration.

Returns:
the face 90 degrees clockwise from the given face. Possible values are:

- BOTTOM
- TOP
- RIGHT
- LEFT

---

getMaxWidth

```
public float getMaxWidth()
```

This gets the maximum width for this DrawnComponent.

Returns:
the maximum width of this drawn component.

---

getMaxHeight

```
public float getMaxHeight()
```

This gets the maximum height for this DrawnComponent.

Returns:
the maximum height of this drawn component.

---

getMinWidth

```
public float getMinWidth()
```

This gets the minimum width for this DrawnComponent.

Returns:
the minimum width of this drawn component.

---

getComponent

`public AbstractComponent getComponent()`

Each DrawnComponent is a rendering object for a specific `AbstractComponent`. This function is used to gain access to the component that is being rendered.

Returns:
the AbstractComponent associated with this DrawnComponent.

---

setComponent

`public void setComponent(AbstractComponent comp)`

This sets the `AbstractComponent` that is being rendered by this DrawnComponent. This component must be able to be rendered by this DrawnComponent.

Parameters:
`comp` - the AbstractComponent to be rendered.

---

getComponentID

`public int getComponentID()`

Retrieves the ident of this drawn component's component

Returns:
the primary key of the `component`.

---

getX_Pos

`public double getX_Pos()`

Gets the X coordinate of the center point of this DrawnComponent.

Returns:
the X coordinate of the center of the bounds.

---

getY_Pos

`public double getY_Pos()`

Gets the Y coordinate of the center point of this DrawnComponent.

Returns:
the Y coordinate of the center of the bounds.

---

setX_Pos

`public void setX_Pos(double position)`

Sets the location of this DrawnComponent's center to the given position in the X dimension.

Parameters:

`position` - the new x coordinate for the center of this DrawnComponent.

---

setY_Pos

```
public void setY_Pos(double position)
```

Sets the location of this DrawnComponent's center to the given position in the Y dimension.

Parameters:

`position` - the new x coordinate for the center of this DrawnComponent.

---

setDrawWidth

```
public void setDrawWidth(double w)
```

Sets the drawing width of the DrawnComponent. If the specified width is invalid, the drawing width will be set to the default drawing width.

Parameters:

`w` - the requested drawing width

---

setDrawHeight

```
public void setDrawHeight(double l)
```

Sets the drawing height of the DrawnComponent. If the specified length is invalid, the drawing height will be set to the default drawing height.

Parameters:

`l` - the requested drawing height

---

disconnectAllMyLinks

```
public void disconnectAllMyLinks()
```

Deprecated. *DrawnLinks have been converted to DrawnConnections*

Disconnects all DrawnLinks of the current DrawnComponent

---

getHandleSize

```
public int getHandleSize()
```

This gets the user defined size of the handles from the user preferences.

Returns:

the size of the Resize Handle specified by the user in the Snap Preferences section

---

getConnectSize

```
public int getConnectSize()
```

Retrieves the number of ConnectingPts the user specifies to be within the Connections vector of the current DrawnComponent. Note: Do not confuse with `getNumberConnections()`

Returns:

the number of ConnectingPts the user specifies to be within the Connections vector of the current DrawnComponent. Checks the value specified by the user in the Snap Preferences set.

---

getNumberConnections

```
public int getNumberConnections()
```

Retrieves the number of ConnectingPts the user specifies to be within the Connections vector of the current DrawnComponent.

---

getConnectingPt

```
public ConnectingPt getConnectingPt(ConnectionData data)
```

Retrieves the ConnectingPt for the given ConnectionData or null of none is found. This is used to find which ConnectingPt a `DrawnConnection` should connect to.

Parameters:

`data` - the ConnectionData to find a ConnectionPt for.

Returns:

the ConnectingPt or null

---

getConnectingPt

```
public ConnectingPt getConnectingPt(int i)
```

Returns a ConnectingPt for a given int. If the specified argument is too large, will return the last ConnectingPt in the Connections vector.

Parameters:

`i` - the index of the required ConnectingPt in the Connections vector

Returns:

the ConnectingPt at the specified index

---

getOrientationMenu

```
public JMenu getOrientationMenu()
```

Creates a menu appropriate for selecting the desired orientation for this drawn component.

Returns:

a JMenu with appropriate items or null of no orientations are appropriate

---

setLabelString

```
public void setLabelString(String str,
        int index)
```

Adds or replaces a String in the labels array. If there is already a String at index n, the String is replaced. If the index is greater than the number of items currently in the array the String is added to the end.

Parameters:

`str` - the String to be added
`index` - the int index into the label array.

---

isPosnSet

```
public boolean isPosnSet()
```

Determines if the DrawnComponent's position has been initialized.

Returns:

- false if the center of the DrawnComponent is (0.0, 0.0)
- true if the center of the DrawnComponent is not (0.0, 0.0)

---

createPopupMenu

```
public JPopupMenu createPopupMenu()
```

Creates a custom popup context menu for this DrawnComponent that includes an optional Actions menu from sub components.

Returns:

the JPopupMenu created from the component.

See Also:

`getComponent()`

---

revTransformPoint

```
public java.awt.Point revTransformPoint(java.awt.Point pt)
```

Transforms a point from ComponentSpace coordinates (upper left corner) down to scaled localSpace coordinates (inlet). This takes into account the component scale factor.

---

forTransformPoint

```
public java.awt.Point forTransformPoint(java.awt.Point pt)
```

Transforms a point from LocalSpace coordinates (inlet) up to ComponentSpace coordinates (upper left corner). This takes into account the component scale factor.

---

getCustomPopupActions

```
public Action[] getCustomPopupActions()
```

Retrieves the curstom popup actions from this drawn component's `AbstractComponent`.

Returns:

the Action[] returned from the component.

See Also:

`AbstractComponent.getCustomPopupActions()`

isDrawBadges

`public boolean isDrawBadges()`

Getter for property drawBadges.

Returns:
Value of property drawBadges.

setDrawBadges

`public void setDrawBadges(boolean drawBadges)`

Setter for property drawBadges.

Parameters:
`drawBadges` - New value of property drawBadges.

isAutoScale

`public boolean isAutoScale()`

Getter for property autoScale that determines whether scaleIt calls scale this Drawn Component.

Returns:
Value of property autoScale.

setAutoScale

`public void setAutoScale(boolean autoScale)`

Setter for property autoScale that determines whether scaleIt calls scale this Drawn Component.

Parameters:
`autoScale` - If true, scaleIt calls will resize the visual representation of this drawn component.

getPreferredSize

`public java.awt.Dimension getPreferredSize()`

Provides the preferred size for this component. Overridden here to maintain size during cut&paste operations.

Returns:
the Dimension containing the un-scaled width and height.

mouseEntered

`public void mouseEntered(java.awt.event.MouseEvent e)`

mouseExited

```
public void mouseExited(java.awt.event.MouseEvent e)
```

---

mouseMoved

```
public void mouseMoved(java.awt.event.MouseEvent e)
```

---

mousePressed

```
public void mousePressed(java.awt.event.MouseEvent e)
```

---

mouseClicked

```
public void mouseClicked(java.awt.event.MouseEvent e)
```

---

mouseReleased

```
public void mouseReleased(java.awt.event.MouseEvent e)
```

---

mouseDragged

```
public void mouseDragged(java.awt.event.MouseEvent evt)
```

---

showConnections

```
public boolean showConnections()
```

---

getBeanBox

```
public BeanBox getBeanBox()
```

Retrieves the BeanBox that contains this DrawnComponent or null if this DrawnComponent exists outside of a BeanBox.

Returns:
the BeanBox at the top of the ancestor list.

---

getGlassPane

```
public GlassPanel getGlassPane()
```

Retrieves the GlassPanel from the ZoomablePanel that contains this DrawnComponent.

Returns:
the GlassPanel from the ZoomablePanel.

See Also:
`getZoomablePanel()`

---

getZoomablePanel

`public ZoomablePanel` **`getZoomablePanel`**`()`

Retrieves the ZoomablePanel that contains this DrawnComponent or null if this DrawnComponent exists outside of a ZoomablePanel.

Returns:
the ZoomablePanel at the top of the ancestor list.

---

translateConnectionToScreen

`public java.awt.Point` **`translateConnectionToScreen`**`(ConnectingPt pt)`

This converts the center of a passed connecting Point into the coordinates on the zoomable panel. This assumes that the passed point is contained within this DrawnComponent.

Parameters:
`pt` - The ConnectingPt whose center is to be translated.

Returns:
The coordinates on the zoomable panel for the connecting Point's center.

---

translatePointToScreen

`public java.awt.Point` **`translatePointToScreen`**`(java.awt.Point point)`

This converts a `java.awt.Point` from local coordinates into coordinates on the zoomable panel. Local coordinates are based around the ceter of the DrawnComponent.

Parameters:
`point` - the Point that is getting translated.

Returns:
The coordinates on the zoomable panel for the Point.

---

storeState

`public void` **`storeState`**`(Hashtable state)`

Store the state of the bean to permit undo.

Parameters:
`state` - A hash table containing modified parameters.

---

restoreState

`public void restoreState(Hashtable state)`

Restore the state of the bean from an earlier edit.

Parameters:

`state` - A hash table containing modified parameters.

---

componentChanged

`public void componentChanged(ComponentChangedEvent evt)`

When a DrawnComponent's component changes, the drawn component needs to call initDrawing on itself

---

componentDeleted

`public void componentDeleted()`

When a DrawnComponent's component is deleted, the Drawn component needs to remove itself from the view.

---

componentConnected

`public void componentConnected(Connection con)`

This should make sure that the ZoomablePanel tries to add the drawnLink to the view when a connection is established.

---

removeNotify

`public void removeNotify()`

---

addNotify

`public void addNotify()`

---

componentDisconnected

`public void componentDisconnected(Connection con)`

This should make sure that the zoomable panel tries to remove the drawnLink from the view when a connection is removed.

---

setParent

`public void setParent(java.awt.Container parent)`

Setter for the parent value of this DrawnComponent. This is used for situations where the DrawnComponent needs a parent reference to a panel that it doesn't exist inside. For example: The NodeViewPanel doesn't actually own the DrawnComponent it displays.

Parameters:

parent - the Container that owns this DrawnComponent.

---

getParent

```
public java.awt.Container getParent()
```

This either gets the actual parent of the DrawnComponent, if it exists inside a Container. Otherwise, it gets the locally held parent value.

See Also:
setParent(Container)

---

isSelected

```
public boolean isSelected()
```

Getter for property selected. This is true if this DrawnComponent is currently selected in the BeanBox.

Returns:
Value of property selected.

---

setSelected

```
public void setSelected(boolean selected)
```

Setter for property selected.

Parameters:
selected - New value of property selected.

---

getCustomPopupItems

```
public Vector getCustomPopupItems()
```

Returns the custom popup items of this DrawnComponent's target component

---

getCustomPopupItems

```
public Vector getCustomPopupItems(java.awt.event.MouseEvent evt)
```

Returns the custom popup items of this DrawnComponent's target component as returned by getCustomPopupItems() as well as those popup items that are appropriate for the given mouse event.

Note that the coordinates of the given event are in BeanBox coordinates and must take the DrawnComponent's location into account.

Parameters:
evt - the MouseEvent generated by the user's right-click on the BeanBox.

---

store

```
public com.apt.xdr.PibBlock store(int viewNum)
```

This function returns a PibBlock for a drawn component.

Parameters:

`comp` - the DrawnComponent to be converted to PibBlock format.
`viewid` - the unique identifier of a ViewComponent.

Returns:

the DrawnComponentRec to store the given component.

---

loadDrawnComponent

```
public void loadDrawnComponent(com.apt.xdr.PibBlock block)
```

This function loads a DrawnComponent from a DrawnComponentRec.

Parameters:

`base` - the AbstractComponent whose DrawnComponent is being loaded.
`rec` - the DrawnComponentRec being loaded.

Returns:

the DrawnComponent loaded from base.

---

hasSubComponents

```
public boolean hasSubComponents()
```

Returns true if this DrawnComponent has DrawnSubComponents contained within it

---

getSubComponentAt

```
public DrawnSubComponent getSubComponentAt(int x,
        int y)
```

Returns the sub-component at the given coordinates, or null of no sub-component exists at that location.

---

createTemplateEntry

```
public TemplateEntry createTemplateEntry()
```

Creates a new TemplateEntry for this DrawnComponent that stores all of the location and state data for this DranwComponent. Any DrawnComponent that has unique data associated with it should store that data in an extension of TemplateEntry and overwrite this method.

---

readTemplateEntry

```
public void readTemplateEntry(TemplateEntry entry)
```

Sets the data on this DrawnComponent from a TemplateEntry read in from a view template file. Any DrawnComponent that has unique data associated with it should read that data in, assuming a TemplateEntry.

---

getConnectingLocation

```
public java.awt.Point getConnectingLocation(ConnectingPt point)
```

Returns the absolute screen location of the given ConnectingPt.

com.cafean.client.ui
Class DrawnConnection

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-javax.swing.JComponent
            |
            +-com.cafean.client.ui.DrawnComponent
               |
               +-com.cafean.client.ui.DrawnConnection
```

All Implemented Interfaces:
ConnectionRenderer, Cloneable, java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, HasGetTransferHandler, java.io.Serializable, AnimationBeanGenerator, ComponentListener, StateEditable, java.awt.event.MouseMotionListener, java.awt.event.MouseListener, Cloneable

---

public class **DrawnConnection**
extends DrawnComponent
implements Cloneable, java.awt.event.MouseListener, java.awt.event.MouseMotionListener, StateEditable, ComponentListener, AnimationBeanGenerator, java.io.Serializable, HasGetTransferHandler, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, Cloneable, ConnectionRenderer

A renderer for a `Connection` between two components that draws a segmented line between the `DrawnComponent` renderers of the two sides of the connection.

Junction Knees algorithm adapted from Graphic Gems II This algorithm for determining the best "neat" path between two objects is described in Graphics Gems * gem number IV.2. By Claudio Rosato, A Simple Connection Algorithm for 2D

Fields inherited from class `com.cafean.client.ui.DrawnComponent`

BOTTOM, CENTER, CENTER_H, CENTER_V, CIRCLE, CROSSHATCH, DIAMOND, DOWN, LEFT, max_positions, NONE, PIXELS_P_METER, RIGHT, SEGMENT_BOTH, SEGMENT_INLET, SEGMENT_NONE, SEGMENT_OUTLET, SEGMENT_SPECIAL, SQUARE, TOP, TRIANGLE, UP

Fields inherited from class `javax.swing.JComponent`

TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class `java.awt.Component`

BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface `java.awt.image.ImageObserver`

ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Fields inherited from interface `javax.swing.undo.StateEditable`

RCSID

Constructor Summary

public	DrawnConnection(Connection connection) Creates a renderer for the given Connection object.
public	DrawnConnection(Connection connection, DrawnComponent source, DrawnComponent target) Creates a renderer for the given Connection object and initializes its' source and target DrawnComponent references.

Method Summary

boolean	addPoint(java.awt.Point p) Adds a path point at the given point, if the point falls on an existing line segment.
boolean	canAddPoint(java.awt.Point p) Return true if a point can be added to the path.
boolean	canRemovePoint(java.awt.Point p) Return true if a point can be removed from the path.
void	componentChanged(ComponentChangedEvent evt)
void	componentRemoved(DrawnComponent comp) This should be called by the view when a drawn component has been removed.
void	componentReshaped(DrawnComponent comp)
void	connectionPointRemoved(DrawnComponent comp)
boolean	contains(double x, double y)
boolean	contains(int x, int y)
boolean	contains(java.awt.Point p)
JComponent[]	createDisplayBeans(int pixelsPerMeter, double widthScaleFactor, ClassLoader loader)
static DrawnConnection	createDrawnConnection(Connection connection, DrawnComponent source, DrawnComponent target)
TemplateEntry	createTemplateEntry() Creates a new TemplateEntry for this DrawnComponent that stores all of the location and state data for this DranwComponent.
void	draw(java.awt.Graphics2D g, boolean selected) Draws indicator colored lines between each of this DrawnConnection's plotted points.

java.awt.Point	getMarkedLocation(DrawnComponent destination) Returns the location on the view that a drawn component should use when determining the postion of a remote component for the purpose of selecting a connecting point.
DrawnComponent	getSource()
DrawnComponent	getTarget()
String	getToolTipText(int x, int y)
java.awt.Rectangle	getUsedBounds()
void	InitDrawing()
boolean	isDrawConnSource() returns true if this dc is being rendered as a source(left side) marker.
boolean	isDrawConnTarget() returns true if this dc is being rendered as a target(right side) marker.
boolean	isDrawLine() returns true if this dc is being rendered as a single line (default)
boolean	isObjectInsideBounds(java.awt.geom.Rectangle2D.Double rect)
boolean	isStraightLine() returns true if this DC is configured to draw as a single straight line
boolean	isStraightLine(java.awt.Point location) returns true if this DC is configured to draw as a single straight line
void	loadDrawnComponent(com.apt.xdr.PibBlock block) Loads this DrawnConnection from the given DrawnComponentRec.
void	mouseClicked(java.awt.event.MouseEvent e)
void	mouseDragged(java.awt.event.MouseEvent e) Handle mouseDragged events for segment and point manipulation
void	mouseEntered(java.awt.event.MouseEvent e)
void	mouseExited(java.awt.event.MouseEvent e)
void	mouseMoved(java.awt.event.MouseEvent e)
void	mousePressed(java.awt.event.MouseEvent e) Handle mousePressed events to support segment and point manipulation
void	mouseReleased(java.awt.event.MouseEvent e) Handle mouseReleased events to support segment and point manipulation

void	readTemplateEntry(TemplateEntry entry) Sets the data on this DrawnComponent from a TemplateEntry read in from a view template file.
void	removeClosestPoint(java.awt.Point p) Attempts to remove the closest point in this DrawnConnection's set of points.
void	repaint()
void	restoreState(Hashtable state) Restore the state of the bean from an earlier edit.
void	setGeneratePoints(boolean generatePoints) Setter for property generatePoints.
void	setGeneratePoints(java.awt.Point location, boolean generatePoints) returns true if this DC is configured to draw as a single straight line
void	setLocation(int x, int y)
void	setPath(java.awt.Point[] path) Sets the path points that this DrawnConnection will use for connecting.
void	setSelected(boolean selected)
void	setStraightLine(boolean straightLine) sets this PathLine to draw as a single straight line
void	setStraightLine(java.awt.Point location, boolean straightLine) sets this PathLine to draw as a single straight line
com.apt.xdr.PibBlock	store(int viewNum) returns a PibBlock for this drawn component.
void	storeState(Hashtable state) Store the state of the bean to permit undo.
String	toString()
void	translate(int dx, int dy)
void	validate()

Methods inherited from class com.cafean.client.ui.DrawnComponent

addNotify, canBeResized, Clear, clearLinks, clone, componentChanged, componentConnected, componentDeleted, componentDisconnected, connectLinks, contains, contains, contains, createBorderRegion, createCenterShape, createConnectionPrototypes, createConnectionPt, createDisplayBeans, createPopupMenu, createTemplateEntry, disconnectAllMyLinks, draw, drawLabelStrings, flip, forTransformPoint, getBeanBox, getClockwiseFace, getComponent, getComponentID, getConnectingLocation, getConnectingPt, getConnectingPt, getConnectingPtAt, getConnectSize, getCounterFace, getCrossflowIndex, getCustomPopupActions, getCustomPopupItems, getCustomPopupItems, getDefaultDrawLength, getDefaultDrawWidth, getDrawAngle, getDrawingFace, getDrawingObject, getFaceByAngle, getFillColor, getGlassPane, getGroupID, getHandleSize, getIndicatorColor, getLength, getLenScaleFactor, getMaxHeight, getMaxWidth, getMinWidth, getMirrorImageShape, getNormalObj, getNumberConnections, getOppositeFace, getOrientation, getOrientationMenu, getOrientationName, getParent, getPreferredSize, getSelectedConnector, getSelectedDropZone, getSubComponentAt, getToolTipText, getToolTipText, getWidthScaleFactor, getX_Pos, getXDistBetweenCPs, getXDistBetweenXflowCPs, getY_Pos, getZoomablePanel, hasSubComponents, InitDrawing, isAutoScale, isDrawBadges, isGroupIDValid, isObjectInsideBounds, isPlenumShaped, isPosnSet, isScalable, isSegmentSet, isSelected, isValveShaped, loadDrawnComponent, mouseClicked, mouseDragged, mouseEntered, mouseExited, mouseMoved, mousePressed, mouseReleased, moveRel, moveTo, paint, paintComponent, print, readTemplateEntry, removeNotify, repositionLinks, repositionLinks, resetPosition, restoreState, revTransformPoint, rotateTo, rotateTo, scaleIt, setAutoScale, setBackupComponent, setBounds, setComponent, setDrawAngle, setDrawBadges, setDrawHeight, setDrawWidth, setEqualTo, setGroupID, setGroupIDValid, setLabelString, setLenScaleFactor, setOrientation, setOrientationByAngle, setOrientationConstrained, setParent, setPixelsPerMeter, setSelected, setSizeTo, setWidthScaleFactor, setX_Pos, setY_Pos, showConnections, store, storeState, toString, translateConnectionToScreen, translatePointToScreen

Methods inherited from class `javax.swing.JComponent`

addAncestorListener, addNotify, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, disable, enable, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getActionForKeyStroke, getActionMap, getAlignmentX, getAlignmentY, getAncestorListeners, getAutoscrolls, getBaseline, getBaselineResizeBehavior, getBorder, getBounds, getClientProperty, getComponentPopupMenu, getConditionForKeyStroke, getDebugGraphicsOptions, getDefaultLocale, getFontMetrics, getGraphics, getHeight, getInheritsPopupMenu, getInputMap, getInputMap, getInputVerifier, getInsets, getInsets, getListeners, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getPopupLocation, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getTransferHandler, getUIClassID, getVerifyInputWhenFocusTarget, getVetoableChangeListeners, getVisibleRect, getWidth, getX, getY, grabFocus, isDoubleBuffered, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingForPrint, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, print, printAll, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, requestFocus, requestFocusInWindow, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setActionMap, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setComponentPopupMenu, setDebugGraphicsOptions, setDefaultLocale, setDoubleBuffered, setEnabled, setFocusTraversalKeys, setFont, setForeground, setInheritsPopupMenu, setInputMap, setInputVerifier, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setTransferHandler, setVerifyInputWhenFocusTarget, setVisible, unregisterKeyboardAction, update, updateUI

Methods inherited from class `java.awt.Container`

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class `java.awt.Component`

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class `java.lang.Object`

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface `java.awt.image.ImageObserver`

imageUpdate

Methods inherited from interface `java.awt.MenuContainer`

getFont, postEvent, remove

Methods inherited from interface `javax.swing.TransferHandler.HasGetTransferHandler`

getTransferHandler

Methods inherited from interface java.awt.event.MouseListener
mouseClicked, mouseEntered, mouseExited, mousePressed, mouseReleased

Methods inherited from interface java.awt.event.MouseMotionListener
mouseDragged, mouseMoved

Methods inherited from interface javax.swing.undo.StateEditable
restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.ComponentListener
componentChanged, componentConnected, componentDeleted, componentDisconnected

Methods inherited from interface com.cafean.client.ui.AnimationBeanGenerator
createDisplayBeans

Methods inherited from interface com.cafean.client.ui.ConnectionRenderer
componentRemoved, componentReshaped, connectionPointRemoved, setGeneratePoints

Methods inherited from interface com.cafean.client.ui.FullScreenDrawing
getUsedBounds, translate

Constructors

DrawnConnection

```
public DrawnConnection(Connection connection)
```

Creates a renderer for the given Connection object. The source and target DrawnComponent references will be initialized automatically at draw time.

Parameters:

`connection` - the Connection that this DrawnConnection is rendering.

DrawnConnection

```
public DrawnConnection(Connection connection,
                       DrawnComponent source,
                       DrawnComponent target)
```

Creates a renderer for the given Connection object and initializes its' source and target DrawnComponent references.

Parameters:

`connection` - the Connection that this DrawnConnection is rendering.
`source` - the DrawnComponent that is the left of `connection`
`target` - the DrawnComponent that is the right of `connection`

Methods

isDrawLine

```
public boolean isDrawLine()
```

returns true if this dc is being rendered as a single line (default)

---

isDrawConnSource

```
public boolean isDrawConnSource()
```

returns true if this dc is being rendered as a source(left side) marker. Note that a target(right side) marker may also be used with this marker.

---

isDrawConnTarget

```
public boolean isDrawConnTarget()
```

returns true if this dc is being rendered as a target(right side) marker. Note that a source(left side) marker may also be used with this marker.

InitDrawing

```
public void InitDrawing()
```

Initializes and scales this component to prepare it for painting.

(Not cell length, just length) All the Connections of the Component are examined. If the ConnectionData from the Connection has a ConnectingPt, that ConnectingPt is set selected.

DrawnComponent derivatives should call this as the last step in their InitDrawing method.

---

validate

```
public void validate()
```

---

setLocation

```
public void setLocation(int x,
        int y)
```

---

setPath

```
public void setPath(java.awt.Point[] path)
```

Sets the path points that this DrawnConnection will use for connecting. This method only works for single line connections.

createDisplayBeans

```
public JComponent[] createDisplayBeans(int pixelsPerMeter,
          double widthScaleFactor,
          ClassLoader loader)
```

Creates and configures a set of `AbstractDisplayBeans` for use in displaying this DrawnComponent in an animatable view.

Either a set of beans or a single bean may be returned. If multiple beans are returned they should be positioned appripropriately in relation to one another.

Note: The base implementation returns null.

removeClosestPoint

```
public void removeClosestPoint(java.awt.Point p)
```

Attempts to remove the closest point in this DrawnConnection's set of points.

Parameters:

`p` - the Point to remove the closest path point.

canRemovePoint

```
public boolean canRemovePoint(java.awt.Point p)
```

Return true if a point can be removed from the path.

canAddPoint

```
public boolean canAddPoint(java.awt.Point p)
```

Return true if a point can be added to the path.

addPoint

```
public boolean addPoint(java.awt.Point p)
```

Adds a path point at the given point, if the point falls on an existing line segment.

Parameters:

`p` - the Point at which to add a path point.

Returns:

true if the point is actually added.

mouseReleased

```
public void mouseReleased(java.awt.event.MouseEvent e)
```

Handle mouseReleased events to support segment and point manipulation

See Also:

`mousePressed(MouseEvent)`

mousePressed

```
public void mousePressed(java.awt.event.MouseEvent e)
```

Handle mousePressed events to support segment and point manipulation

See Also:

mouseReleased(MouseEvent)

mouseClicked

```
public void mouseClicked(java.awt.event.MouseEvent e)
```

mouseDragged

```
public void mouseDragged(java.awt.event.MouseEvent e)
```

Handle mouseDragged events for segment and point manipulation

See Also:

mousePressed(MouseEvent)

mouseMoved

```
public void mouseMoved(java.awt.event.MouseEvent e)
```

mouseEntered

```
public void mouseEntered(java.awt.event.MouseEvent e)
```

mouseExited

```
public void mouseExited(java.awt.event.MouseEvent e)
```

getToolTipText

```
public String getToolTipText(int x,
        int y)
```

isObjectInsideBounds

```
public boolean isObjectInsideBounds(java.awt.geom.Rectangle2D.Double rect)
```

Determines if an object is inside the given rectangle

---

contains

```
public boolean contains(double x,
         double y)
```

Checks whether this component "contains" the specified coordinates where *x* and *y* are defined to be relative to the coordinate system of this component.

Note that `getBounds()` includes the position of this component within the BeanBox where the given x and y should **not**.

---

contains

```
public boolean contains(java.awt.Point p)
```

---

contains

```
public boolean contains(int x,
         int y)
```

---

draw

```
public void draw(java.awt.Graphics2D g,
         boolean selected)
```

Draws indicator colored lines between each of this DrawnConnection's plotted points.

Parameters:

`g` - the Graphics2D object which will do the painting

`selected` - indicates whether the current DrawnComponent is currently selected

---

getSource

```
public DrawnComponent getSource()
```

Returns:

the source drawn component

---

getTarget

```
public DrawnComponent getTarget()
```

Returns:

the target drawn component

---

componentRemoved

```
public void componentRemoved(DrawnComponent comp)
```

This should be called by the view when a drawn component has been removed. If the given DrawnComponent is this connection's source or target, it removes itself as well.

---

componentReshaped

```
public void componentReshaped(DrawnComponent comp)
```

---

connectionPointRemoved

```
public void connectionPointRemoved(DrawnComponent comp)
```

---

repaint

```
public void repaint()
```

---

translate

```
public void translate(int dx,
          int dy)
```

---

getUsedBounds

```
public java.awt.Rectangle getUsedBounds()
```

---

componentChanged

```
public void componentChanged(ComponentChangedEvent evt)
```

When a DrawnComponent's component changes, the drawn component needs to call initDrawing on itself

---

toString

```
public String toString()
```

The String produced is based on the toString of the component.

---

setSelected

```
public void setSelected(boolean selected)
```

Setter for property selected.

---

setGeneratePoints

```
public void setGeneratePoints(boolean generatePoints)
```

Setter for property generatePoints.

Parameters:

`generatePoints` - New value of property generatePoints.

---

setGeneratePoints

```
public void setGeneratePoints(java.awt.Point location,
        boolean generatePoints)
```

returns true if this DC is configured to draw as a single straight line

---

isStraightLine

```
public boolean isStraightLine()
```

returns true if this DC is configured to draw as a single straight line

---

isStraightLine

```
public boolean isStraightLine(java.awt.Point location)
```

returns true if this DC is configured to draw as a single straight line

---

setStraightLine

```
public void setStraightLine(java.awt.Point location,
        boolean straightLine)
```

sets this PathLine to draw as a single straight line

---

setStraightLine

```
public void setStraightLine(boolean straightLine)
```

sets this PathLine to draw as a single straight line

---

storeState

```
public void storeState(Hashtable state)
```

Store the state of the bean to permit undo.

Parameters:

`state` - A hash table containing modified parameters.

---

restoreState

```
public void restoreState(Hashtable state)
```

Restore the state of the bean from an earlier edit.

Parameters:

`state` - A hash table containing modified parameters.

---

store

```
public com.apt.xdr.PibBlock store(int viewNum)
```

returns a PibBlock for this drawn component.

Parameters:

`viewNum` - the unique identifier of a ViewComponent.

Returns:

the DrawnComponentRec to store the given component.

---

loadDrawnComponent

```
public void loadDrawnComponent(com.apt.xdr.PibBlock block)
```

Loads this DrawnConnection from the given DrawnComponentRec.

Parameters:

`block` - the DrawnComponentRec being loaded.

---

createDrawnConnection

```
public static DrawnConnection createDrawnConnection(Connection connection,
        DrawnComponent source,
        DrawnComponent target)
```

---

createTemplateEntry

```
public TemplateEntry createTemplateEntry()
```

Creates a new TemplateEntry for this DrawnComponent that stores all of the location and state data for this DranwComponent. Any DrawnComponent that has unique data associated with it should store that data in an extension of TemplateEntry and overwrite this method.

---

readTemplateEntry

```
public void readTemplateEntry(TemplateEntry entry)
```

Sets the data on this DrawnComponent from a TemplateEntry read in from a view template file. Any DrawnComponent that has unique data associated with it should read that data in, assuming a TemplateEntry.

---

getMarkedLocation

```
public java.awt.Point getMarkedLocation(DrawnComponent destination)
```

Returns the location on the view that a drawn component should use when determining the postion of a remote component for the purpose of selecting a connecting point.

Parameters:

`destination` - the DrawnComponent being drawn to.

com.cafean.client.ui
Class DrawnUserValue

```
java.lang.Object
   |
   +-java.awt.Component
      |
      +-java.awt.Container
         |
         +-javax.swing.JComponent
            |
            +-com.cafean.client.ui.DrawnComponent
               |
               +-com.cafean.client.ui.DrawnUserValue
```

All Implemented Interfaces:
Groupable, java.io.Serializable, StateEditable, java.awt.event.ActionListener, java.io.Serializable, java.awt.MenuContainer, java.awt.image.ImageObserver, HasGetTransferHandler, java.io.Serializable, AnimationBeanGenerator, ComponentListener, StateEditable, java.awt.event.MouseMotionListener, java.awt.event.MouseListener, Cloneable

---

public class **DrawnUserValue**
extends DrawnComponent
implements Cloneable, java.awt.event.MouseListener, java.awt.event.MouseMotionListener, StateEditable, ComponentListener, AnimationBeanGenerator, java.io.Serializable, HasGetTransferHandler, java.awt.image.ImageObserver, java.awt.MenuContainer, java.io.Serializable, java.awt.event.ActionListener , StateEditable, java.io.Serializable, Groupable

A Representation of a DrawnUserValue in a DrawnView

Fields inherited from class com.cafean.client.ui.DrawnComponent
BOTTOM, CENTER, CENTER_H, CENTER_V, CIRCLE, CROSSHATCH, DIAMOND, DOWN, LEFT, max_positions, NONE, PIXELS_P_METER, RIGHT, SEGMENT_BOTH, SEGMENT_INLET, SEGMENT_NONE, SEGMENT_OUTLET, SEGMENT_SPECIAL, SQUARE, TOP, TRIANGLE, UP

Fields inherited from class javax.swing.JComponent
TOOL_TIP_TEXT_KEY, UNDEFINED_CONDITION, WHEN_ANCESTOR_OF_FOCUSED_COMPONENT, WHEN_FOCUSED, WHEN_IN_FOCUSED_WINDOW

Fields inherited from class java.awt.Component
BOTTOM_ALIGNMENT, CENTER_ALIGNMENT, LEFT_ALIGNMENT, RIGHT_ALIGNMENT, TOP_ALIGNMENT

Fields inherited from interface java.awt.image.ImageObserver
ABORT, ALLBITS, ERROR, FRAMEBITS, HEIGHT, PROPERTIES, SOMEBITS, WIDTH

Fields inherited from interface javax.swing.undo.StateEditable
RCSID

Fields inherited from interface javax.swing.undo.StateEditable
RCSID

Constructor Summary

public	DrawnUserValue(UserDefinedValue value) Creates a new instance of DrawnUserValue

Method Summary

void	actionPerformed(java.awt.event.ActionEvent evt) Responds to an ActionEvent produced when hitting 'enter' in the text field contained in this drawing.
void	draw(java.awt.Graphics2D g, boolean selected)
JLabel	getLabel() Getter for property label.
JMenu	getOrientationMenu()
JPanel	getPanel() Getter for property panel.
java.awt.Dimension	getPreferredSize()
JLabel	getUnits() Getter for property units.
JLabel	getValueLabel() Getter for property valueLabel.
RealTextField	getValueText() Getter for property valueText.
void	InitDrawing() Updates the values inside this component, and updates it's bounds
boolean	isCustomLabel()
boolean	isEditable() Getter for property editable.
boolean	isShowLabel() Getter for property showLabel.
boolean	isShowUnits() Getter for property showUnits.
void	loadDrawnComponent(com.apt.xdr.PibBlock block)

void	popupEditor() Opens a dialog for editing the properties of this drawing.
void	readByteArray(byte[] byteArr) Reads this DrawnUserValue's data from the given byte[] of XML Encoded information.
void	restoreState(Hashtable state) Restore the state of the bean from an earlier edit.
void	setBackground(java.awt.Color color)
void	setCustomLabel(boolean custom)
void	setEditable(boolean editable) Setter for property editable.
void	setForeground(java.awt.Color color)
void	setLabel(JLabel label) Setter for property label.
void	setPanel(JPanel panel) Setter for property panel.
void	setShowLabel(boolean showLabel) Setter for property showLabel.
void	setShowUnits(boolean showUnits) Setter for property showUnits.
void	setUnits(JLabel units) Setter for property units.
void	setValueLabel(JLabel valueLabel) Setter for property valueLabel.
void	setValueText(RealTextField valueText) Setter for property valueText.
com.apt.xdr.PibBlock	store(int viewNum)
void	storeState(Hashtable state) Store the state of the bean to permit undo.

Methods inherited from class com.cafean.client.ui.DrawnComponent

addNotify, canBeResized, Clear, clearLinks, clone, componentChanged, componentConnected, componentDeleted, componentDisconnected, connectLinks, contains, contains, contains, createBorderRegion, createCenterShape, createConnectionPrototypes, createConnectionPt, createDisplayBeans, createPopupMenu, createTemplateEntry, disconnectAllMyLinks, draw, drawLabelStrings, flip, forTransformPoint, getBeanBox, getClockwiseFace, getComponent, getComponentID, getConnectingLocation, getConnectingPt, getConnectingPt, getConnectingPtAt, getConnectSize, getCounterFace, getCrossflowIndex, getCustomPopupActions, getCustomPopupItems, getCustomPopupItems, getDefaultDrawLength, getDefaultDrawWidth, getDrawAngle, getDrawingFace, getDrawingObject, getFaceByAngle, getFillColor, getGlassPane, getGroupID, getHandleSize, getIndicatorColor, getLength, getLenScaleFactor, getMaxHeight, getMaxWidth, getMinWidth, getMirrorImageShape, getNormalObj, getNumberConnections, getOppositeFace, getOrientation, getOrientationMenu, getOrientationName, getParent, getPreferredSize, getSelectedConnector, getSelectedDropZone, getSubComponentAt, getToolTipText, getToolTipText, getWidthScaleFactor, getX_Pos, getXDistBetweenCPs, getXDistBetweenXflowCPs, getY_Pos, getZoomablePanel, hasSubComponents, InitDrawing, isAutoScale, isDrawBadges, isGroupIDValid, isObjectInsideBounds, isPlenumShaped, isPosnSet, isScalable, isSegmentSet, isSelected, isValveShaped, loadDrawnComponent, mouseClicked, mouseDragged, mouseEntered, mouseExited, mouseMoved, mousePressed, mouseReleased, moveRel, moveTo, paint, paintComponent, print, readTemplateEntry, removeNotify, repositionLinks, repositionLinks, resetPosition, restoreState, revTransformPoint, rotateTo, rotateTo, scaleIt, setAutoScale, setBackupComponent, setBounds, setComponent, setDrawAngle, setDrawBadges, setDrawHeight, setDrawWidth, setEqualTo, setGroupID, setGroupIDValid, setLabelString, setLenScaleFactor, setOrientation, setOrientationByAngle, setOrientationConstrained, setParent, setPixelsPerMeter, setSelected, setSizeTo, setWidthScaleFactor, setX_Pos, setY_Pos, showConnections, store, storeState, toString, translateConnectionToScreen, translatePointToScreen

Methods inherited from class `javax.swing.JComponent`

addAncestorListener, addNotify, addVetoableChangeListener, computeVisibleRect, contains, createToolTip, disable, enable, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getActionForKeyStroke, getActionMap, getAlignmentX, getAlignmentY, getAncestorListeners, getAutoscrolls, getBaseline, getBaselineResizeBehavior, getBorder, getBounds, getClientProperty, getComponentPopupMenu, getConditionForKeyStroke, getDebugGraphicsOptions, getDefaultLocale, getFontMetrics, getGraphics, getHeight, getInheritsPopupMenu, getInputMap, getInputMap, getInputVerifier, getInsets, getInsets, getListeners, getLocation, getMaximumSize, getMinimumSize, getNextFocusableComponent, getPopupLocation, getPreferredSize, getRegisteredKeyStrokes, getRootPane, getSize, getToolTipLocation, getToolTipText, getToolTipText, getTopLevelAncestor, getTransferHandler, getUIClassID, getVerifyInputWhenFocusTarget, getVetoableChangeListeners, getVisibleRect, getWidth, getX, getY, grabFocus, isDoubleBuffered, isLightweightComponent, isManagingFocus, isOpaque, isOptimizedDrawingEnabled, isPaintingForPrint, isPaintingTile, isRequestFocusEnabled, isValidateRoot, paint, paintImmediately, paintImmediately, print, printAll, putClientProperty, registerKeyboardAction, registerKeyboardAction, removeAncestorListener, removeNotify, removeVetoableChangeListener, repaint, repaint, requestDefaultFocus, requestFocus, requestFocus, requestFocusInWindow, resetKeyboardActions, reshape, revalidate, scrollRectToVisible, setActionMap, setAlignmentX, setAlignmentY, setAutoscrolls, setBackground, setBorder, setComponentPopupMenu, setDebugGraphicsOptions, setDefaultLocale, setDoubleBuffered, setEnabled, setFocusTraversalKeys, setFont, setForeground, setInheritsPopupMenu, setInputMap, setInputVerifier, setMaximumSize, setMinimumSize, setNextFocusableComponent, setOpaque, setPreferredSize, setRequestFocusEnabled, setToolTipText, setTransferHandler, setVerifyInputWhenFocusTarget, setVisible, unregisterKeyboardAction, update, updateUI

Methods inherited from class `java.awt.Container`

add, add, add, add, add, addContainerListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, countComponents, deliverEvent, doLayout, findComponentAt, findComponentAt, getAlignmentX, getAlignmentY, getComponent, getComponentAt, getComponentAt, getComponentCount, getComponents, getComponentZOrder, getContainerListeners, getFocusTraversalKeys, getFocusTraversalPolicy, getInsets, getLayout, getListeners, getMaximumSize, getMinimumSize, getMousePosition, getPreferredSize, insets, invalidate, isAncestorOf, isFocusCycleRoot, isFocusCycleRoot, isFocusTraversalPolicyProvider, isFocusTraversalPolicySet, layout, list, list, locate, minimumSize, paint, paintComponents, preferredSize, print, printComponents, remove, remove, removeAll, removeContainerListener, removeNotify, setComponentZOrder, setFocusCycleRoot, setFocusTraversalKeys, setFocusTraversalPolicy, setFocusTraversalPolicyProvider, setFont, setLayout, transferFocusBackward, transferFocusDownCycle, update, validate

Methods inherited from class `java.awt.Component`

action, add, addComponentListener, addFocusListener, addHierarchyBoundsListener, addHierarchyListener, addInputMethodListener, addKeyListener, addMouseListener, addMouseMotionListener, addMouseWheelListener, addNotify, addPropertyChangeListener, addPropertyChangeListener, applyComponentOrientation, areFocusTraversalKeysSet, bounds, checkImage, checkImage, contains, contains, createImage, createImage, createVolatileImage, createVolatileImage, deliverEvent, disable, dispatchEvent, doLayout, enable, enable, enableInputMethods, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, firePropertyChange, getAccessibleContext, getAlignmentX, getAlignmentY, getBackground, getBaseline, getBaselineResizeBehavior, getBounds, getBounds, getColorModel, getComponentAt, getComponentAt, getComponentListeners, getComponentOrientation, getCursor, getDropTarget, getFocusCycleRootAncestor, getFocusListeners, getFocusTraversalKeys, getFocusTraversalKeysEnabled, getFont, getFontMetrics, getForeground, getGraphics, getGraphicsConfiguration, getHeight, getHierarchyBoundsListeners, getHierarchyListeners, getIgnoreRepaint, getInputContext, getInputMethodListeners, getInputMethodRequests, getKeyListeners, getListeners, getLocale, getLocation, getLocation, getLocationOnScreen, getMaximumSize, getMinimumSize, getMouseListeners, getMouseMotionListeners, getMousePosition, getMouseWheelListeners, getName, getParent, getPeer, getPreferredSize, getPropertyChangeListeners, getPropertyChangeListeners, getSize, getSize, getToolkit, getTreeLock, getWidth, getX, getY, gotFocus, handleEvent, hasFocus, hide, imageUpdate, inside, invalidate, isBackgroundSet, isCursorSet, isDisplayable, isDoubleBuffered, isEnabled, isFocusable, isFocusCycleRoot, isFocusOwner, isFocusTraversable, isFontSet, isForegroundSet, isLightweight, isMaximumSizeSet, isMinimumSizeSet, isOpaque, isPreferredSizeSet, isShowing, isValid, isVisible, keyDown, keyUp, layout, list, list, list, list, list, locate, location, lostFocus, minimumSize, mouseDown, mouseDrag, mouseEnter, mouseExit, mouseMove, mouseUp, move, nextFocus, paint, paintAll, postEvent, preferredSize, prepareImage, prepareImage, print, printAll, remove, removeComponentListener, removeFocusListener, removeHierarchyBoundsListener, removeHierarchyListener, removeInputMethodListener, removeKeyListener, removeMouseListener, removeMouseMotionListener, removeMouseWheelListener, removeNotify, removePropertyChangeListener, removePropertyChangeListener, repaint, repaint, repaint, repaint, requestFocus, requestFocusInWindow, reshape, resize, resize, setBackground, setBounds, setBounds, setComponentOrientation, setCursor, setDropTarget, setEnabled, setFocusable, setFocusTraversalKeys, setFocusTraversalKeysEnabled, setFont, setForeground, setIgnoreRepaint, setLocale, setLocation, setLocation, setMaximumSize, setMinimumSize, setName, setPreferredSize, setSize, setSize, setVisible, show, show, size, toString, transferFocus, transferFocusBackward, transferFocusUpCycle, update, validate

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Methods inherited from interface java.awt.image.ImageObserver

imageUpdate

Methods inherited from interface java.awt.MenuContainer

getFont, postEvent, remove

Methods inherited from interface javax.swing.TransferHandler.HasGetTransferHandler

getTransferHandler

Methods inherited from interface java.awt.event.MouseListener
mouseClicked, mouseEntered, mouseExited, mousePressed, mouseReleased

Methods inherited from interface java.awt.event.MouseMotionListener
mouseDragged, mouseMoved

Methods inherited from interface javax.swing.undo.StateEditable
restoreState, storeState

Methods inherited from interface com.cafean.client.analysis.ComponentListener
componentChanged, componentConnected, componentDeleted, componentDisconnected

Methods inherited from interface com.cafean.client.ui.AnimationBeanGenerator
createDisplayBeans

Methods inherited from interface java.awt.event.ActionListener
actionPerformed

Methods inherited from interface javax.swing.undo.StateEditable
restoreState, storeState

Methods inherited from interface com.cafean.client.ui.util.Groupable
getGroupID, isGroupIDValid, setGroupID, setGroupIDValid

Constructors

DrawnUserValue

public **DrawnUserValue**(UserDefinedValue value)

Creates a new instance of DrawnUserValue

Methods

InitDrawing

public void **InitDrawing**()

Updates the values inside this component, and updates it's bounds

actionPerformed

```
public void actionPerformed(java.awt.event.ActionEvent evt)
```

Responds to an ActionEvent produced when hitting 'enter' in the text field contained in this drawing.

draw

```
public void draw(java.awt.Graphics2D g,
         boolean selected)
```

Tints the current DrawnComponent according to its state. Also draws the normalObj of the DrawnComponent, as well as the component's labels, connection points, and links

getPreferredSize

```
public java.awt.Dimension getPreferredSize()
```

Provides the preferred size for this component. Overridden here to maintain size during cut&paste operations.

isEditable

```
public boolean isEditable()
```

Getter for property editable.

Returns:
Value of property editable.

setEditable

```
public void setEditable(boolean editable)
```

Setter for property editable.

Parameters:
`editable` - New value of property editable.

isShowLabel

```
public boolean isShowLabel()
```

Getter for property showLabel.

Returns:
Value of property showLabel.

setShowLabel

```
public void setShowLabel(boolean showLabel)
```

Setter for property showLabel.

Parameters:

showLabel - New value of property showLabel.

---

isShowUnits

```
public boolean isShowUnits()
```

Getter for property showUnits.

Returns:
Value of property showUnits.

---

setShowUnits

```
public void setShowUnits(boolean showUnits)
```

Setter for property showUnits.

Parameters:
showUnits - New value of property showUnits.

---

getLabel

```
public JLabel getLabel()
```

Getter for property label.

Returns:
Value of property label.

---

setLabel

```
public void setLabel(JLabel label)
```

Setter for property label.

Parameters:
label - New value of property label.

---

getUnits

```
public JLabel getUnits()
```

Getter for property units.

Returns:
Value of property units.

---

setUnits

```
public void setUnits(JLabel units)
```

Setter for property units.

Parameters:

units - New value of property units.

---

getPanel

```
public JPanel getPanel()
```

Getter for property panel.

Returns:
Value of property panel.

---

setPanel

```
public void setPanel(JPanel panel)
```

Setter for property panel.

Parameters:
panel - New value of property panel.

---

getValueText

```
public RealTextField getValueText()
```

Getter for property valueText.

Returns:
Value of property valueText.

---

setValueText

```
public void setValueText(RealTextField valueText)
```

Setter for property valueText.

Parameters:
valueText - New value of property valueText.

---

getValueLabel

```
public JLabel getValueLabel()
```

Getter for property valueLabel.

Returns:
Value of property valueLabel.

---

setValueLabel

```
public void setValueLabel(JLabel valueLabel)
```

Setter for property valueLabel.

Parameters:

valueLabel - New value of property valueLabel.

---

popupEditor

```
public void popupEditor()
```

Opens a dialog for editing the properties of this drawing.

---

storeState

```
public void storeState(Hashtable state)
```

Store the state of the bean to permit undo.

Parameters:

state - A hash table containing modified parameters.

---

restoreState

```
public void restoreState(Hashtable state)
```

Restore the state of the bean from an earlier edit.

Parameters:

state - A hash table containing modified parameters.

---

setForeground

```
public void setForeground(java.awt.Color color)
```

---

setBackground

```
public void setBackground(java.awt.Color color)
```

---

readByteArray

```
public void readByteArray(byte[] byteArr)
```

Reads this DrawnUserValue's data from the given byte[] of XML Encoded information.

Parameters:

byteArr - a byte[] containing the result of using storeState to produce a Hashtable of a DrawnUserValue's current state then using java.beans.XMLEncoder to encode that hashtable.

---

store

```
public com.apt.xdr.PibBlock store(int viewNum)
```

This function returns a PibBlock for a drawn component.

loadDrawnComponent

```
public void loadDrawnComponent(com.apt.xdr.PibBlock block)
```

This function loads a DrawnComponent from a DrawnComponentRec.

isCustomLabel

```
public boolean isCustomLabel()
```

setCustomLabel

```
public void setCustomLabel(boolean custom)
```

getOrientationMenu

```
public JMenu getOrientationMenu()
```

Creates a menu appropriate for selecting the desired orientation for this drawn component.

NRC FORM 335 (9-2004) NRCMD 3.7

U.S. NUCLEAR REGULATORY COMMISSION

BIBLIOGRAPHIC DATA SHEET

(See instructions on the reverse)

1. REPORT NUMBER (Assigned by NRC, Add Vol., Supp., Rev., and Addendum Numbers, if any.)

NUREG/CR-6974, Vol. 2

2. TITLE AND SUBTITLE

Symbolic Nuclear Analysis Package (SNAP)
Common Application Framework for Engineering Analysis (CAFEAN) Preprocessor Plug-in Application Programming Interface
Appendix A, Part A

3. DATE REPORT PUBLISHED

MONTH	YEAR
June	2009

4. FIN OR GRANT NUMBER

Y6851

5. AUTHOR(S)

Ken Jones, John Rothe, William Dunsford

6. TYPE OF REPORT

Technical

7. PERIOD COVERED *(Inclusive Dates)*

8. PERFORMING ORGANIZATION - NAME AND ADDRESS *(If NRC, provide Division, Office or Region, U.S. Nuclear Regulatory Commission, and mailing address; if contractor, provide name and mailing address.)*

Applied Programming Technology, Inc
240 Market St., Suite 208
Bloomsburg, PA 17815-1951

9. SPONSORING ORGANIZATION - NAME AND ADDRESS *(If NRC, type "Same as above"; if contractor, provide NRC Division, Office or Region, U.S. Nuclear Regulatory Commission, and mailing address.)*

Division of System Analysis
Office of Nuclear Regulatory Research
U.S. Nuclear Regulatory Commission
Washington DC 20555-0001

10. SUPPLEMENTARY NOTES

C. Gingrich, NRC Project Manager

11. ABSTRACT *(200 words or less)*

Many of the analytical codes developed by the Office of Nuclear Regulatory Research (RES) rely on a text based input file to specify model parameters and computational options. The formats of the text based input files are often quite complex and usually require careful study before a user can create an input model that functions correctly. The Symbolic Nuclear Analysis Package (SNAP) is primarily a graphical user interface that was developed to simplify the analyst's task of creating input files for the analytic codes as well as helping to visualize code results. SNAP is a Java based computer application that runs on the most popular computer platforms including Windows XP and Vista, LINUX based systems, and Mac OS X. The code architecture used in SNAP is "plug-in" based and very flexible. Third party developers can implement their own user interfaces under SNAP without breaking the interfaces developed by other developers. The application programming interface (API) that is described in this document provides a short tutorial and some guidelines for developing a custom plug-in that works in the SNAP framework. This document also includes the actual API method and data-structure definitions needed to create such a custom interface.

12. KEY WORDS/DESCRIPTORS *(List words or phrases that will assist researchers in locating the report.)*

Graphical User Interface, Analysis, Analytic Code, Computation Code, Computation, Symbolic Nuclear Analysis Package, Graphical, User Interface, Java, plug-in, GUI, SNAP

13. AVAILABILITY STATEMENT

unlimited

14. SECURITY CLASSIFICATION

(This Page)
unclassified

(This Report)
unclassified

15. NUMBER OF PAGES

16. PRICE

Federal Recycling Program

UNITED STATES
NUCLEAR REGULATORY COMMISSION
WASHINGTON, DC 20555-0001

OFFICIAL BUSINESS