EOApplication Java Framework

API Reference

Apple Computer, Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

The eoapplication Package

Package: com.webobjects.eoapplication

Introduction

Documentation for this package is forthcoming. For information on using this package, see the book *Getting Started with Direct to Java Client*.

The most important classes in this package are:

Class	Description
EOController	A representation of controller objects responsible for managing and sometimes generating the user interface of a Java Client application
EOComponentController	A controller that manages user interface components
EOEntityController	A controller that displays enterprise objects in a user interface
EODocumentController	A controller that displays and edits enterprise objects in a user interface
EOInterfaceController	A controller that represents an Interface Builder nib file
EOApplication	The Java Client application
EOAppletController	A representation of an EOApplet as a controller in the controller hierarchy

Class	Description
EOApplet	The default Applet class used in Java Client applications
EOAction	An abstract representation of operations the user can invoke from the user interface
EOXMLUnarchiver	An object containing the parameters from an XML specification used to create the controllers in the controller hierarchy

Rule System and XML Description

For Direct to Java Client applications, the controller hierarchy for the client side of the application is created from an XML description. The XML description is created by a rule system on the server side of the application. The details of this process are described in the book *Getting Started With Direct to Java Client*.

There are three pieces of information associated with an EOController class that are used to generate a controller hierarchy with the rule system: the controller's `controllerType`, its corresponding XML tag, and its corresponding XML attributes. Each controller class specification identifies this information in a section titled "Rule System and XML Description".

controllerType

You use the `controllerType` key to define custom rules that should fire only for certain kinds of controllers. For example, suppose you want to set the minimum width of all an application's windows. To do so, you write a rule whose condition specifies that the `controllerType` is 'windowController'. Then the rule only fires for controllers that control windows. Each controller falls into one of the following controller types:

- windowController
- modalDialogController
- entityController
- widgetController
- tableController
- groupingController
- dividingController

- actionWidgetController

XML Tag and Attributes

As an XML description is parsed, an EOXMLUnarchiver maps XML tags to particular EOController classes. All concrete controller classes—classes whose instances can actually be used in a controller hierarchy—have an XML tag.

XML attributes tell the EOXMLUnarchiver how to configure the controllers. XML attributes are inherited. For example, there are three XML attributes defined for EOController—className, disabledActionNames, and typeName. These attributes can be used by any controller, however, because all controllers are subclasses of EOController.

BeansAppletContext

Inherits from: Object

Implements: java.applet.AppletContext

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

- `getApplet`
- `getApplets`
- `getAudioClip`
- `getImage`
- `showDocument`

showDocument

showStatus

Instance Methods

getApplet

```
public java.applet.Applet getApplet(String aString)
```

getApplets

```
public java.util.Enumeration getApplets()
```

getAudioClip

```
public java.applet.AudioClip getAudioClip(java.net.URL anURL)
```

getImage

```
public synchronized java.awt.Image getImage(java.net.URL anURL)
```

showDocument

```
public void showDocument(
    java.net.URL anURL,
    String aString)
```

showDocument

```
public void showDocument(java.net.URL anURL)
```

showStatus

```
public void showStatus(String aString)
```

BeansAppletStub

Inherits from: Object

Implements: java.applet.AppletStub

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

`appletResize`

`getAppletContext`

`getCodeBase`

`getDocumentBase`

`getParameter`

isActive

Instance Methods

appletResize

```
public void appletResize(
    int anInt,
    int anInt)
```

getAppletContext

```
public java.applet.AppletContext getAppletContext()
```

getCodeBase

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

getDocumentBase

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

getParameter

```
public String getParameter(String aString)
```

isActive

```
public boolean isActive()
```

BeansCallback

Inherits from: Object

Implements: _EOTimer.Callback

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Instance Methods

timerFired

```
public void timerFired(
    _EOTimer a_EOTimer,
    java.util.Date aDate,
    java.util.Date aDate)
```

EOAction

Inherits from: javax.swing.AbstractAction

Package: com.webobjects.eoapplication

Class Description

EOAction objects are abstract representations of operations the user can invoke from the user interface. An action does not specify how it appears in the user interface—it can appear as a button, a menu item, or both.

Each action defines a method called the action name, that is invoked when the action triggers. An action also has a description path, which describes the category of the action and its name. For example, a Quit action's description path might be "Document/Quit". In addition, the action can have a short description that differs from the last element of the description path, for example, "Quit the Application".

Actions can have icons for buttons in the application and small icons for minor buttons in the user interface. To allow users to trigger actions with "hot-keys," each action has a menu accelerator, a javax.swing.KeyStroke the user can type on the keyboard.

Actions often appear in groups in the user interface: buttons in the same group are rendered close together and menu items in the group are rendered in separate menus like the Document, Edit, Tools, or Window menus. To group actions, EOAction defines a category priority. All actions in the same group have the same category priority. An additional parameter, the action priority defines the order in which actions appear within a group (for example, the order menu items appear within a menu).

An action triggers when the user clicks the corresponding user interface widget. In most cases, the action's method is dispatched to the subcontrollers of the controller that displays the action. Methods whose names end with ...ForControllerHierarchy return such actions. In some cases, the action's method is dispatched to the active widget, like the text field containing the cursor. Methods whose names end with ...ForFocusComponent return such actions. In other cases, the action's method is dispatched to a particular object, usually the EOApplication at the root of the controller hierarchy.

EOAction defines methods to create actions, access an action's parameters, manage groups of actions, and accessing shared actions used in Direct to Java Client applications.

Constants

Constant	Description
ActivatePreviousWindowActionPriority	Description forthcoming.
AppendActionPriority	Description forthcoming.
CancelActionPriority	Description forthcoming.
ClearActionPriority	Description forthcoming.
CloseWindowActionPriority	Description forthcoming.
DeleteActionPriority	Description forthcoming.
DescriptionPathSeparator	Description forthcoming.
DocumentCategoryPriority	Description forthcoming.
EditCategoryPriority	Description forthcoming.
FindActionPriority	Description forthcoming.
HelpActionPriority	Description forthcoming.
HelpCategoryPriority	Description forthcoming.

Constant	Description
InfoActionPriority	Description forthcoming.
InsertActionPriority	Description forthcoming.
ModalActionsCategoryPriority	Description forthcoming.
NavigationActionPriority	Description forthcoming.
NavigationActionsCategoryPriority	Description forthcoming.
NewMultipleWindowActionPriority	Description forthcoming.
NewSingleWindowActionPriority	Description forthcoming.
ObjectActionPriority	Description forthcoming.
ObjectActionsCategoryPriority	Description forthcoming.
OkayActionPriority	Description forthcoming.
OpenActionPriority	Description forthcoming.
PasteboardCopyActionPriority	Description forthcoming.
PasteboardCutActionPriority	Description forthcoming.
PasteboardDeleteActionPriority	Description forthcoming.
PasteboardPasteActionPriority	Description forthcoming.
PasteboardSelectAllActionPriority	Description forthcoming.
QuitActionPriority	Description forthcoming.
RedoActionPriority	Description forthcoming.
RefreshActionPriority	Description forthcoming.
RevertActionPriority	Description forthcoming.
SaveActionPriority	Description forthcoming.
SaveAllActionPriority	Description forthcoming.
SearchActionPriority	Description forthcoming.

Constant	Description
ToolsCateogryPriority	Description forthcoming.
UndoActionPriority	Description forthcoming.
WindowCategoryPriority	Description forthcoming.

Method Types

Accessing action parameters

actionName

actionPriority

actionTitle

categoryPriority

descriptionPath

descriptionPathComponents

icon

menuAccelerator

setActionName

setActionPriority

setCategoryPriority

setDescriptionPath

setIcon

setMenuAccelerator

setShortDescription

setSmallIcon

shortDescription

smallIcon

Creating actions

EOAction

actionForControllerHierarchy

actionForFocusComponent

actionForObject

standardActionForFocusComponent

standardDocumentActionForControllerHierarchy

standardDocumentActionForApplication

standardDocumentActionForControllerHierarchy

standardEditActionForControllerHierarchy

Creating menu accelerators

keyStrokeWithKeyCode

keyStrokeWithKeyCodeAndModifiers

keyStrokeWithKeyCodeAndShiftModifier

keyStrokeWithString

Accessing specific shared actions

standardActivatePreviousWindowActionForApplication

standardAddActionForControllerHierarchy

standardAppendActionForControllerHierarchy

standardCancelActionForControllerHierarchy

standardClearActionForControllerHierarchy

standardCloseWindowActionForControllerHierarchy

standardDeleteActionForControllerHierarchy

standardDeselectActionForControllerHierarchy

standardEditActionsForFocusComponent

standardFindActionForControllerHierarchy

standardInsertActionForControllerHierarchy

standardInsertWithTaskActionForControllerHierarchy

standardOkActionForControllerHierarchy

standardOkAndSaveActionForControllerHierarchy

standardOpenWithTaskActionForControllerHierarchy

standardQuitActionForApplication

standardRedoActionForControllerHierarchy

standardRefreshActionForApplication

standardRemoveActionForControllerHierarchy

standardRevertActionForControllerHierarchy

standardSaveActionForControllerHierarchy

standardSaveAllActionForApplication

standardSelectActionForControllerHierarchy

standardUndoActionForControllerHierarchy

Managing actions

actionCanBePerformedInContextOfController

actionPerformed

mergedActions

sortedActions

Managing the property change listener

addPropertyChangeListener

firePropertyChange

removePropertyChangeListener

Methods inherited from Object

equals

toString

Constructors

EOAction

```
public EOAction(
    String actionName,
    String descriptionPath,
    String shortDescription,
    javax.swing.Icon icon,
    javax.swing.Icon smallIcon,
    javax.swing.KeyStroke menuAccelerator,
    int categoryPriority,
    int actionPriority)
```

Returns a new action (an EOAction object) as specified by the arguments.

See Also: actionName, descriptionPath, shortDescription, icon, smallIcon, menuAccelerator, categoryPriority, and actionPriority.

Static Methods

actionForControllerHierarchy

```
public static EOAction actionForControllerHierarchy(
    String actionName,
    String descriptionPath,
    String shortDescription,
    javax.swing.Icon icon,
    javax.swing.Icon smallIcon,
    javax.swing.KeyStroke menuAccelerator,
```

```
    int categoryPriority,
    int actionPriority,
    boolean sendsActionToAllControllers)
```

Returns a new action (an EOAction object) as specified by the arguments. When this action triggers, it is dispatched to the subcontrollers of the controller that displays it. If sendsActionToAllControllers is true, the action is dispatched to the subcontrollers of the controller that displays the action. Otherwise, the action is dispatched to the first subcontroller that responds to it.

See Also: EOAction, actionName, descriptionPath, shortDescription, icon, smallIcon, menuAccelerator, categoryPriority, and actionPriority.

actionForFocusComponent

```
public static EOAction actionForFocusComponent(
    String actionName,
    String descriptionPath,
    String shortDescription,
    javax.swing.Icon icon,
    javax.swing.Icon smallIcon,
    javax.swing.KeyStroke menuAccelerator,
    int categoryPriority,
    int actionPriority)
```

Returns a new action (an EOAction object) as specified by the arguments. When this action triggers, it is dispatched to the active widget (for example, the text field containing the cursor). The other parameters are identical to the EOAction constructor parameters.

See Also: EOAction, actionName, descriptionPath, shortDescription, icon, smallIcon, menuAccelerator, categoryPriority, and actionPriority.

actionForObject

```
public static EOAction actionForObject(
    String actionName,
    String descriptionPath,
    String shortDescription,
    javax.swing.Icon icon,
    javax.swing.Icon smallIcon,
    javax.swing.KeyStroke menuAccelerator,
```

```
    int categoryPriority,
    int actionPriority,
    Object object)
```

Returns a new action (an EOAction object) as specified by the arguments. When this action triggers, it is dispatched directly to `object`. To create an action that gets dispatched to the application, set `object` to the EOApplication at the top of the controller hierarchy. The other parameters are identical to the EOAction constructor parameters.

See Also: `EOAction`, `actionName`, `descriptionPath`, `shortDescription`, `icon`, `smallIcon`, `menuAccelerator`, `categoryPriority`, and `actionPriority`.

keyStrokeWithKeyCode

```
public static javax.swing.KeyStroke keyStrokeWithKeyCode(int keyCode)
```

Returns a KeyStroke given the numerical key code `keyCode` with the appropriate modifier for the client operating system (usually `CTRL_MASK`). See Sun's javax.swing.KeyStroke documentation for more information.

keyStrokeWithKeyCodeAndModifiers

```
public static javax.swing.KeyStroke keyStrokeWithKeyCodeAndModifiers(
    int keyCode,
    int modifiers)
```

Returns a KeyStroke given the numerical key code `keyCode` and the modifier mask `modifiers`. This method adds the appropriate modifier for the client operating system (usually `CTRL_MASK`). See Sun's javax.swing.KeyStroke documentation for more information.

keyStrokeWithKeyCodeAndShiftModifier

```
public static javax.swing.KeyStroke keyStrokeWithKeyCodeAndShiftModifier(int keyCode)
```

Returns a KeyStroke given the numerical key code `keyCode` with the SHIFT modifier. This method also adds the appropriate modifier for the client operating system (usually `CTRL_MASK`). See Sun's javax.swing.KeyStroke documentation for more information.

keyStrokeWithString

```
public static javax.swing.KeyStroke keyStrokeWithString(String keyStrokeDescription)
```

Returns a KeyStroke for the String keyStrokeDescription. This method adds the appropriate modifier for the client operating system (usually CTRL_MASK). See Sun's javax.swing.KeyStroke documentation for more information.

mergedActions

```
public static NSArray mergedActions(
    NSArray actionArray1,
    NSArray actionArray2)
```

Returns an NSArray containing all of the actions in actionArray1 and actionArray2 with duplicate actions removed.

sortedActions

```
public static NSArray sortedActions(NSArray actionArray)
```

Returns a sorted NSArray containing the actions in actionArray. The actions are sorted first on the category priority, then on the action priority, and finally on the description path.

See Also: categoryPriority, actionPriority, and descriptionPath.

standardActionForFocusComponent

```
public static EOAction standardActionForFocusComponent(
    String actionName,
    javax.swing.KeyStroke menuAccelerator,
    int actionPriority)
```

Returns a shared action as specified by the arguments. When the action triggers, it is dispatched to the focus component (for example, a text field). The action's category priority is the edit action priority so the action is grouped with the other edit actions.

See Also: actionName, menuAccelerator, and actionPriority.

standardActivatePreviousWindowActionForApplication

```
public static EOAction standardActivatePreviousWindowActionForApplication()
```

Returns a shared action (an EOAction object) for the `activatePreviousWindow` method. When this action triggers, it is dispatched to the EOApplication at the top of the controller hierarchy. The action's category priority is the window action priority so the action is grouped with the other window actions. This action appears as the Activate Previous Window item in the Window menu in Direct to Java Client applications.

standardAddActionForControllerHierarchy

```
public static EOAction standardAddActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `add` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardAppendActionForControllerHierarchy

```
public static EOAction standardAppendActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `append` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the edit action priority so the action is grouped with the other edit actions.

standardCancelActionForControllerHierarchy

```
public static EOAction standardCancelActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `cancel` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the modal dialog action priority so the action is grouped with the other modal dialog actions.

standardClearActionForControllerHierarchy

```
public static EOAction standardClearActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `clear` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the edit action priority so the action is grouped with the other edit actions.

standardCloseWindowActionForControllerHierarchy

```
public static EOAction standardCloseWindowActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `close` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the window action priority so the action is grouped with the other window actions.

standardDeleteActionForControllerHierarchy

```
public static EOAction standardDeleteActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `delete` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardDeselectActionForControllerHierarchy

```
public static EOAction standardDeselectActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `deselect` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardDocumentActionForApplication

```
public static EOAction standardDocumentActionForApplication(
    String actionName,
    javax.swing.KeyStroke menuAccelerator,
    int actionPriority)
```

Returns a shared action with the method name `actionName`, menu accelerator `menuAccelerator`, and action priority `actionPriority`. When this action triggers, it is dispatched to the EOApplication at the top of the controller hierarchy. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardDocumentActionForControllerHierarchy

```
public static EOAction standardDocumentActionForControllerHierarchy(
    String actionName,
    javax.swing.KeyStroke menuAccelerator,
    int actionPriority)
```

Returns a shared action with the method name `actionName`, menu accelerator `menuAccelerator`, and action priority `actionPriority`. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardDocumentActionForControllerHierarchy

```
public static EOAction standardDocumentActionForControllerHierarchy(
    String actionName,
    String baseTitle,
    javax.swing.KeyStroke menuAccelerator,
    int actionPriority)
```

Returns a shared action as specified by the arguments. The `baseTitle` parameter is the name of the action as it appears in the user interface and is used for both the short description and the action title. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

See Also: `actionName`, `actionTitle`, `shortDescription`, `menuAccelerator`, and `actionPriority`.

standardEditActionForControllerHierarchy

```
public static EOAction standardEditActionForControllerHierarchy(
    String actionName,
    javax.swing.KeyStroke menuAccelerator,
    int actionPriority)
```

Returns a shared action as specified by the arguments. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the edit action priority so the action is grouped with the other edit actions.

See Also: actionName, menuAccelerator, and actionPriority.

standardEditActionsForFocusComponent

```
public static NSArray standardEditActionsForFocusComponent()
```

Returns an NSArray containing shared actions for the cut, copy, and paste methods. When these actions trigger, they are dispatched to the focus component. Sets the category priorities for the actions to the edit category priority so the actions are grouped with the other edit actions.

standardFindActionForControllerHierarchy

```
public static EOAction standardFindActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the find method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the edit action priority so the action is grouped with the other edit actions.

standardInsertActionForControllerHierarchy

```
public static EOAction standardInsertActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the insertWithTask method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardInsertWithTaskActionForControllerHierarchy

```
public static EOAction standardInsertWithTaskActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `insertWithTask` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardOkActionForControllerHierarchy

```
public static EOAction standardOkActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for an OK button in a modal dialog box. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the modal dialog action priority so the action is grouped with the other modal dialog actions.

standardOkAndSaveActionForControllerHierarchy

```
public static EOAction standardOkAndSaveActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for an OK and Save button in a modal dialog box. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the modal dialog action priority so the action is grouped with the other modal dialog actions.

standardOpenWithTaskActionForControllerHierarchy

```
public static EOAction standardOpenWithTaskActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `openWithTask` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardQuitActionForApplication

```
public static EOAction standardQuitActionForApplication()
```

Returns a shared action (an EOAction object) for the `quit` method. When this action triggers, it is dispatched to the EOApplication at the top of the controller hierarchy. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardRedoActionForControllerHierarchy

```
public static EOAction standardRedoActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `redo` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the edit action priority so the action is grouped with the other edit actions.

standardRefreshActionForApplication

```
public static EOAction standardRefreshActionForApplication()
```

Returns a shared action (an EOAction object) for the `refresh` method. When this action triggers, it is dispatched to the EOApplication at the top of the controller hierarchy. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardRemoveActionForControllerHierarchy

```
public static EOAction standardRemoveActionForControllerHierarchy()
```

The action's category priority is the document action priority so the action is grouped with the other document actions.Returns a shared action (an EOAction object) for the `remove` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it.

standardRevertActionForControllerHierarchy

```
public static EOAction standardRevertActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `revert` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardSaveActionForControllerHierarchy

```
public static EOAction standardSaveActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `save` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardSaveAllActionForApplication

```
public static EOAction standardSaveAllActionForApplication()
```

Returns a shared action (an EOAction object) for the `saveAll` method. When this action triggers, it is dispatched to the EOApplication at the top of the controller hierarchy. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardSelectActionForControllerHierarchy

```
public static EOAction standardSelectActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `select` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the document action priority so the action is grouped with the other document actions.

standardUndoActionForControllerHierarchy

```
public static EOAction standardUndoActionForControllerHierarchy()
```

Returns a shared action (an EOAction object) for the `undo` method. When the action triggers, it is dispatched to the subcontrollers of the controller that displays it. The action's category priority is the edit action priority so the action is grouped with the other edit actions.

Instance Methods

actionCanBePerformedInContextOfController

```
public boolean actionCanBePerformedInContextOfController(EOController controller)
```

Returns whether or not an action can trigger, which depends on the state of controllers in the controller hierarchy. For example, a Save action for an unedited document can not trigger.

actionName

```
public String actionName()
```

Returns the name of the method that executes when the receiver triggers.

actionPerformed

```
public void actionPerformed(java.awt.event.ActionEvent actionEvent)
```

This method is called when an action is triggered, that is, the user presses the action's button or selects its menu item.

actionPriority

```
public int actionPriority()
```

Returns the receiver's action priority, which determines the order in which its button or menu item appears within a category.

See Also: categoryPriority

actionTitle

```
public String actionTitle()
```

Returns the receiver's title, the last component of the receiver's description path.

addPropertyChangeListener

```
public void addPropertyChangeListener(java.beans.PropertyChangeListener listener)
```

See the method description for addPropertyChangeListener in Sun's documentation for javax.swing.AbstractAction.

categoryPriority

```
public int categoryPriority()
```

Returns the receiver's category priority, which determines the order in which the group of buttons or menu items that contains the receiver appears.

descriptionPath

```
public String descriptionPath()
```

Returns the receiver's menu hierarchy path. For example, the Quit menu item description path is Document/Quit.

descriptionPathComponents

```
public NSArray descriptionPathComponents()
```

Returns an NSArray containing the separate components of the receiver's menu hierarchy path.

equals

```
public boolean equals(Object anObject)
```

Indicates whether some object "is equal to" this one.

firePropertyChange

```
protected void firePropertyChange(
    String propertyName,
    Object oldValue,
    Object newValue)
```

See the method description for `firePropertyChange` in Sun's documentation for javax.swing.AbstractAction.

icon

```
public javax.swing.Icon icon()
```

Returns the receiver's icon.

menuAccelerator

```
public javax.swing.KeyStroke menuAccelerator()
```

Returns the KeyStroke the user can type to invoke the receiver instead of selecting it from the menu.

removePropertyChangeListener

```
public void removePropertyChangeListener(java.beans.PropertyChangeListener listener)
```

See the method description for `removePropertyChangeListener` in Sun's documentation for javax.swing.AbstractAction.

setActionName

```
public void setActionName(String actionName)
```

Sets the name of the method that executes when the receiver triggers.

setActionPriority

```
public void setActionPriority(int actionPriority)
```

Sets the receiver's action priority, which determines the order in which its button or menu item appears within a category.

setCategoryPriority

```
public void setCategoryPriority(int categoryPriority)
```

Returns the receiver's category priority, which determines the order in which the group of buttons or menu items containing the receiver appears.

setDescriptionPath

```
public void setDescriptionPath(String descriptionPath)
```

Sets the receiver's menu hierarchy path to `descriptionPath`.

setIcon

```
public void setIcon(javax.swing.Icon icon)
```

Sets the receiver's icon to `icon`.

setMenuAccelerator

```
public void setMenuAccelerator(javax.swing.KeyStroke menuAccelerator)
```

Sets the KeyStroke the user can type to invoke the receiver instead of selecting it from a menu.

See Also: keyStrokeWithKeyCode, keyStrokeWithKeyCodeAndModifiers, keyStrokeWithKeyCodeAndShiftModifier, and keyStrokeWithString.

setShortDescription

```
public void setShortDescription(String shortDescription)
```

Sets the action's short description to shortDescription. The short description appears in buttons and menu items. If shortDescription is null, the receiver's title is displayed instead.

See Also: actionTitle

setSmallIcon

```
public void setSmallIcon(javax.swing.Icon anIcon)
```

Sets the receiver's small icon used for some small buttons in the user interface (the Select button in a Form window's to-one relationship editor is an example).

shortDescription

```
public String shortDescription()
```

Returns the receiver's short description, which is displayed in buttons and menu items. If the short description is set to null or has not been assigned, shortDescription returns the action's title.

See Also: actionTitle

smallIcon

```
public javax.swing.Icon smallIcon()
```

Returns the receiver's small icon used for some small buttons in the user interface (the Select button in a Form window's to-one relationship editor is an example). By default, the small icon is not displayed for such buttons; the short description is displayed instead.

See Also: shortDescription

toString

```
public String toString()
```

Returns the receiver as a string that states the receiver's method name, description path, category priority, and action priority.

EOActionButtonsController

Inherits from: EOActionWidgetController :
EOComponentController :
EOController :
Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
ACTIONBUTTONSCONTROLLER	actionWidgetController

Method Types

All methods

EOActionButtonsController

actionWidget

actionWidgetToSubcontrollerAreaDistance

createWidgetForActionsAndPlaceInContainer

disposeActionWidget

setUsesLargeButtonRepresentation

updateActionWidgetEnabling

usesLargeButtonRepresentation

Constructors

EOActionButtonsController

```
public EOActionButtonsController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOActionButtonsController()
```

Description forthcoming.

Instance Methods

actionWidget

```
public javax.swing.JComponent actionWidget()
```

Description forthcoming.

actionWidgetToSubcontrollerAreaDistance

```
protected int actionWidgetToSubcontrollerAreaDistance()
```

Description forthcoming.

createWidgetForActionsAndPlaceInContainer

```
protected void createWidgetForActionsAndPlaceInContainer(
    NSArray aNSArray,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

disposeActionWidget

```
protected void disposeActionWidget()
```

Description forthcoming.

setUsesLargeButtonRepresentation

```
public void setUsesLargeButtonRepresentation(boolean aBoolean)
```

Description forthcoming.

updateActionWidgetEnabling

```
protected void updateActionWidgetEnabling()
```

Description forthcoming.

usesLargeButtonRepresentation

```
public boolean usesLargeButtonRepresentation()
```

Description forthcoming.

EOActionMenuController

Inherits from: EOActionWidgetController :
EOComponentController :
EOController

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
ACTIONMENUCONTROLLER	actionWidgetController

Method Types

All methods

EOActionMenuController

actionWidget

actionWidgetToSubcontrollerAreaDistance

createWidgetForActionsAndPlaceInContainer

disposeActionWidget

updateActionWidgetEnabling

Constructors

EOActionMenuController

```
public EOActionMenuController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOActionMenuController()
```

Description forthcoming.

Instance Methods

actionWidget

```
public javax.swing.JComponent actionWidget()
```

Description forthcoming.

actionWidgetToSubcontrollerAreaDistance

```
protected int actionWidgetToSubcontrollerAreaDistance()
```

Description forthcoming.

createWidgetForActionsAndPlaceInContainer

```
protected void createWidgetForActionsAndPlaceInContainer(
    NSArray aNSArray,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

disposeActionWidget

```
protected void disposeActionWidget()
```

Description forthcoming.

updateActionWidgetEnabling

```
protected void updateActionWidgetEnabling()
```

Description forthcoming.

EOActionWidgetController

Inherits from: EOComponentController : EOController : Object

Implements: EOActionWidgetController.ActionCollector

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
None (abstract class)	`actionWidgetController`

Method Types

All methods

`EOActionWidgetController`

`actionWidget`

`actionWidgetContainer`

actionWidgetPosition

actionWidgetToSubcontrollerAreaDistance

collectedActions

componentDidBecomeVisible

createWidgetForActionsAndPlaceInContainer

dispose

disposeActionWidget

generateComponent

resetActions

setActionWidgetContainer

setActionWidgetPosition

subcontrollerActionsDidChange

subcontrollerConnectionDidChange

updateActionWidgetEnabling

Constructors

EOActionWidgetController

```
public EOActionWidgetController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOActionWidgetController()
```

Description forthcoming.

Instance Methods

actionWidget

```
public abstract javax.swing.JComponent actionWidget()
```

Description forthcoming.

actionWidgetContainer

```
public javax.swing.JComponent actionWidgetContainer()
```

Description forthcoming.

actionWidgetPosition

```
public int actionWidgetPosition()
```

Description forthcoming.

actionWidgetToSubcontrollerAreaDistance

```
protected abstract int actionWidgetToSubcontrollerAreaDistance()
```

Description forthcoming.

collectedActions

```
public NSArray collectedActions()
```

Description forthcoming.

componentDidBecomeVisible

```
protected void componentDidBecomeVisible()
```

Description forthcoming.

createWidgetForActionsAndPlaceInContainer

```
protected abstract void createWidgetForActionsAndPlaceInContainer(
    NSArray aNSArray,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

dispose

```
public void dispose()
```

Description forthcoming.

disposeActionWidget

```
protected abstract void disposeActionWidget()
```

Description forthcoming.

generateComponent

```
protected void generateComponent()
```

Description forthcoming.

resetActions

```
public void resetActions()
```

Description forthcoming.

setActionWidgetContainer

```
public void setActionWidgetContainer(javax.swing.JComponent aJComponent)
```

Description forthcoming.

setActionWidgetPosition

```
public void setActionWidgetPosition(int anInt)
```

Description forthcoming.

subcontrollerActionsDidChange

```
public void subcontrollerActionsDidChange(EOController anEOController)
```

Description forthcoming.

subcontrollerConnectionDidChange

```
public void subcontrollerConnectionDidChange(EOController anEOController)
```

Description forthcoming.

updateActionWidgetEnabling

```
protected abstract void updateActionWidgetEnabling()
```

Description forthcoming.

EOActionWidgets

Inherits from: Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

`actionMenuWithActions`

`buttonRowWithActions`

`disposeActionMenu`

`disposeButtonRow`

`disposeMenuBar`

`menuBarWithActions`

updateEnablingOfActionMenu

updateEnablingOfButtonRow

updateEnablingOfMenuBar

Constructors

EOActionWidgets

```
public EOActionWidgets()
```

Description forthcoming.

Static Methods

actionMenuWithActions

```
public static javax.swing.JComboBox actionMenuWithActions(
    NSArray aNSArray,
    EOController anEOController,
    boolean aBoolean,
    String aString)
```

Description forthcoming.

buttonRowWithActions

```
public static javax.swing.JComponent buttonRowWithActions(
    NSArray aNSArray,
    EOController anEOController,
    boolean aBoolean,
    boolean aBoolean,
    boolean aBoolean)
```

Description forthcoming.

disposeActionMenu

```
public static void disposeActionMenu(javax.swing.JComboBox aJComboBox)
```

Description forthcoming.

disposeButtonRow

```
public static void disposeButtonRow(javax.swing.JComponent aJComponent)
```

Description forthcoming.

disposeMenuBar

```
public static void disposeMenuBar(javax.swing.JMenuBar aJMenuBar)
```

Description forthcoming.

menuBarWithActions

```
public static javax.swing.JMenuBar menuBarWithActions(
    NSArray aNSArray,
    EOController anEOController,
    boolean aBoolean)
```

Description forthcoming.

updateEnablingOfActionMenu

```
public static void updateEnablingOfActionMenu(javax.swing.JComboBox aJComboBox)
```

Description forthcoming.

updateEnablingOfButtonRow

```
public static void updateEnablingOfButtonRow(javax.swing.JComponent aJComponent)
```

Description forthcoming.

updateEnablingOfMenuBar

```
public static void updateEnablingOfMenuBar(javax.swing.JMenuBar aJMenuBar)
```

Description forthcoming.

EOApplet

Inherits from: javax.swing.JApplet

Package: com.webobjects.eoapplication

Class Description

EOApplet is the default Applet class embedded in WebObjects pages containing a WOJavaClientApplet dynamic element. EOApplet's only task is to read all the application's arguments (passed as HTML parameters) and forward them to the initialization code in EOApplication. For maximum flexibility, any application specific code should be implemented in EOApplication's `finishInitialization` method rather than EOApplet's `init` method.

In the controller hierarchy, the applet is represented by an EOAppletController, which becomes a client subcontroller of the EOApplication object.

EOApplet is used in Java Client applications only; there is no equivalent class on the server.

Constructors

EOApplet

```
public EOApplet()
```

Description forthcoming.

Instance Methods

destroy

```
public void destroy()
```

Description forthcoming.

init

```
public void init()
```

Instantiates an EOAppletController for the controller hierarchy and invokes EOApplication's `startApplication` using parameters retrieved via Applet's `getParameter`.

EOAppletController

Inherits from: EOComponentController : EOController : Object

Package: com.webobjects.eoapplication

Class Description

This class represents an EOApplet as a controller in the controller hierarchy. When the application is running as an applet, this controller is the direct descendent of the EOApplication in the controller hierarchy. It performs the analogous function as a EOWindowController when the application is running as a Java application.

XML Tag	Default Rule System Controller Type
None (abstract class)	None

Method Types

All methods

`EOAppletController`

`applet`

setApplet

setVisible

showInSupercontroller

Constructors

EOAppletController

public EOAppletController(EOApplet **applet**)

Creates an applet controller for applet.

Instance Methods

applet

public EOApplet applet()

Returns the receiver's applet.

setApplet

protected void setApplet(EOApplet **applet**)

Sets the receiver's applet to applet.

setVisible

public void setVisible(boolean **flag**)

Sets the visibility of the applet according to flag. Since applets can not be made invisible, this method does nothing if visible is false.

showInSupercontroller

```
public boolean showInSupercontroller()
```

Properly integrates the content of the applet (usually a component of a EOInterfaceController).

EOApplication

Inherits from: EOController : Object

Package: com.webobjects.eoapplication

Class Description

Java Client applications typically execute from the command line (often referred to as a "Java application") or as an applet running in a browser. EOApplication insulates the developer from this distinction by serving as an execution-mode-independent repository for application-level client-side logic. The provided JApplet subclass EOApplet simply invokes EOApplication with the HTML arguments as parameters.

Each application has a window observer which keeps track of all of the windows in the application, which window is active, and whether all windows have been closed. The window observer has two notifications: `ActiveWindowChangedNotification` and `LastWindowClosedNotification`, which the `finishInitialization` method binds to the `activeWindowDidChange` and `lastWindowDidClose` methods, respectively.

Each application also has a defaults manager, an EODefaults object, which maintains two dictionaries for application defaults: a transient dictionary whose values are forgotten when the application exits, and a persistent dictionary whose values are stored on the server. The defaults manager implements `valueForKey` to read the defaults and `setPersistentValueForKey` and `setTransientValueForKey` to store the defaults.

EOApplication is used in Java Client application only; there is no equivalent class on the server side.

XML Tag	Default Rule System Controller Type
None	None

Constants

Constant	Description
ApplicationDidStart	Description forthcoming.
ApplicationWillQuit	Description forthcoming.

Interfaces Implemented

NSDisposable

dispose

EOKeyValueCodingAdditions (com.webobjects.eocontrol)

EOAction.Enabling

canPerformActionNamed

EOKeyValueCoding (com.webobjects.eocontrol inherited from EOKeyValueCodingAdditions)

NSKeyValueCoding (inherited from EOKeyValueCoding)

Method Types

Accessing the shared instance

sharedApplication

Entering the application

main

startApplication

Initializing and terminating the application

canQuit

finishInitialization

quitsOnLastWindowClose

setCanQuit

setQuitsOnLastWindowClose

Managing the application

arguments

defaults

languages

Managing documents

documents

documentsForGlobalID

editedDocuments

hasEditedDocuments

Managing the window observer

activeWindowDidChange

lastWindowDidClose

setWindowObserver

windowObserver

Methods inherited from Object

toString

Performing main menu operations

activatePreviousWindow

collectChangesFromServer

defaultActions

saveAll

quit

Constructors

EOApplication

```
public EOApplication()
```

Description forthcoming.

Static Methods

main

```
public static void main(String[] args[])
```

This is the standard entry point for applications started from the command line (not in an applet). The args array contains the application's command-line arguments (for example, `-key1 value1 -key2 value2 ...`), which are stored in a parameter dictionary (NSDictionary). The user must specify an application URL (using the `-applicationURL <application URL>` argument), the name of a distribution channel class (using the `-channelClassName <channel class name>` argument), or both depending on the specific distribution channel. If the user specifies the application URL, he can optionally specify any initial entry page other than Main.

After instantiating an EODistributionChannel on the basis of these two parameters, `main` simply invokes `startApplication`.

sharedApplication

```
public static EOApplication sharedApplication()
```

Returns the EOApplication instance initialized via the `startApplication` method, throwing an IllegalStateException if `startApplication` has not yet been invoked.

startApplication

```
public static EOApplication startApplication(
    NSDictionary parameterDictionary,
    EOComponentController initialTopComponentController,
    boolean remoteRequestArguments)
```

Creates an EOApplication. An application can execute from the command line or as an applet.

EOApplication's parameters are specified using `parameterDictionary`. If the application is a Java application, the EOApplication's `main` method reads and parses the parameters from the command line. In addition, it sets `remoteRequestArguments` to `true`, which triggers `startApplication` to read additional parameters from the applet at the URL specified on the command line. If the application is started in an applet, all parameters are contained in the HTML.

The `initialTopComponentController` parameter specifies the top-most EOComponentController in the controller hierarchy. For applets, this controller is an EOAppletController. For command line applications, the `main` method sets `initialTopComponentController` to `null`, which causes a new EOFrameController to be instantiated and used as the top-most EOComponentController.

Instance Methods

activatePreviousWindow

```
public void activatePreviousWindow()
```

Activates the previously active window. The user can invoke this method from the Window menu.

activeWindowDidChange

```
public void activeWindowDidChange(NSNotification aNSNotification)
```

This method is invoked when the user changes the active window in the receiver (usually by clicking in an inactive window). It is invoked via a notification from the receiver's window observer.

arguments

```
public NSDictionary arguments()
```

Returns all of the receiver's arguments in a dictionary. If the application is a Java application (and not an Applet), the arguments can come from both the command line and the applet at the URL specified on the command line.

canPerformActionNamed

```
public boolean canPerformActionNamed(String actionName)
```

Conformance to EOAction.Enabling. See the method description of `canPerformActionNamed` in the interface specification for EOAction.Enabling. An action may be disallowed if it is disabled or is an `activatePreviousWindow` action and the first window is active.

canQuit

```
public boolean canQuit()
```

Returns whether or not the receiver has a Quit item in the File submenu. Defaults to `true` if the application is run from the command line and `false` if it is started in an applet.

collectChangesFromServer

```
public void collectChangesFromServer()
```

Updates the receiver's Enterprise Objects to reflect the changes sent to the server from other client applications. By default, the application does not automatically update its objects, however, the user can update the objects manually from the Document menu in Direct to Java Client applications.

defaultActions

```
protected NSArray defaultActions()
```

Returns an NSArray containing the actions (EOAction objects) the receiver can perform.

defaults

```
public EODefaults defaults()
```

Returns the receiver's defaults manager (an EODefaults object). If your application requires the user to log in, you should override this method so it returns `null` until the user logs in.

dispose

```
public void dispose()
```

Prepares the receiver so it is disposed when Java performs garbage collection.

documents

```
public NSArray documents()
```

Returns an NSArray containing the receiver's visible documents (EODocument objects).

documentsForGlobalID

```
public NSArray documentsForGlobalID(
    com.webobjects.eocontrol.EOGlobalID globalID,
    String entityName)
```

Returns an NSArray containing the receiver's visible documents (EODocument objects) that edit Enterprise Objects with an entity name matching `entityName` and global ID matching `globalID`.

editedDocuments

```
public NSArray editedDocuments()
```

Returns an NSArray containing the receiver's visible documents (EODocument objects) that have been edited.

finishInitialization

```
protected void finishInitialization()
```

This method is invoked after the final event thread is guaranteed to be running. If you subclass EOApplication, use this method to initialize anything relating to the user interface or event-handling. Do not perform such initialization using EOApplication's constructor.

hasEditedDocuments

```
public boolean hasEditedDocuments()
```

Returns `true` if any of the receiver's documents has been edited. Otherwise returns `false`.

languages

```
public NSArray languages()
```

Returns an NSArray containing the language names (Strings) for which the application is localized. An example language is `English`.

lastWindowDidClose

```
public void lastWindowDidClose(NSNotification aNSNotification)
```

This method is invoked when the user closes the last window in the receiver. It is invoked as a notification from the receiver's window observer.

See Also: `quitsOnLastWindowClose`

quit

```
public void quit()
```

Causes the receiver to quit (provided `canQuit` is `true`).

See Also: `canQuit`

quitsOnLastWindowClose

```
public boolean quitsOnLastWindowClose()
```

Returns whether or not the receiver quits when the user closes all of its windows. Defaults to `true`.

saveAll

```
public boolean saveAll()
```

Attempts to save all of the receiver's edited documents and returns `true` if it succeeds.

sessionDidTimeOut

```
public void sessionDidTimeOut()
```

Description forthcoming.

setCanQuit

```
public void setCanQuit(boolean flag)
```

Sets whether or not the application has a quit item in the File menu.

setQuitsOnLastWindowClose

```
public void setQuitsOnLastWindowClose(boolean flag)
```

Sets whether or not the receiver quits when the user closes all of its windows.

setWindowObserver

```
public void setWindowObserver(EOWindowObserver anEOWindowObserver)
```

Sets the receiver's window observer to `anEOWindowObserver`. The window observer manages the application's windows: which window is active, how many there are, etc.

toString

```
public String toString()
```

Returns the receiver as a string that contains the results of the EOController's `toString` method, the languages the receiver supports, and the status of the `canQuit` and `quitsOnLastWindowClose` flags.

windowObserver

```
public EOWindowObserver windowObserver()
```

Returns the receiver's window observer.

EOArchive

Inherits from: Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

- `EOArchive`
- `loadArchiveNamed`
- `loadArchiveNamed`
- `debug`
- `disposableRegistry`
- `namedObjects`

Constructors

EOArchive

```
public EOArchive(
    Object anObject,
    NSDisposableRegistry aNSDisposableRegistry)
```

Static Methods

loadArchiveNamed

```
public static NSDictionary loadArchiveNamed(
    String aString,
    Object anObject,
    String aString,
    NSDisposableRegistry aNSDisposableRegistry)
```

loadArchiveNamed

```
public static boolean loadArchiveNamed(
    String aString,
    Object anObject,
    String aString)
```

Instance Methods

debug

```
protected void debug(String aString)
```

disposableRegistry

```
public NSDisposableRegistry disposableRegistry()
```

namedObjects

```
public NSDictionary namedObjects()
```

EOBeanSupport

Inherits from: Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

- `beanBases`
- `beanClassLoader`
- `beanClassName`
- `beanCodeBase`
- `beanDocBase`
- `beanReadyToRun`

beanReadyToUse

beanSuperclassName

looksInstantiable

looksSerializable

Constructors

EOBeanSupport

```
public EOBeanSupport()
```

Description forthcoming.

Static Methods

beanBases

```
public static java.net.URL[] beanBases(Object anObject)
```

Description forthcoming.

beanClassLoader

```
public static ClassLoader beanClassLoader(Object anObject)
```

Description forthcoming.

beanClassName

```
public static String beanClassName(Object anObject)
```

Description forthcoming.

beanCodeBase

```
public static java.net.URL beanCodeBase(Object anObject)
```

Description forthcoming.

beanDocBase

```
public static java.net.URL beanDocBase(Object anObject)
```

Description forthcoming.

beanReadyToRun

```
public static Object beanReadyToRun(Object anObject)
```

Description forthcoming.

beanReadyToUse

```
public static Object beanReadyToUse(Object anObject)
```

Description forthcoming.

beanSuperclassName

```
public static String beanSuperclassName(Object anObject)
```

Description forthcoming.

looksInstantiable

```
public static boolean looksInstantiable(Object anObject)
```

Description forthcoming.

looksSerializable

```
public static boolean looksSerializable(Object anObject)
```

Description forthcoming.

EOBoxController

Inherits from: EOComponentController : EOController : Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
BOXCONTROLLER	groupingController

Constants

Constant	Description
EtchedBorder	Description forthcoming.
LineBorder	Description forthcoming.
LoweredBezelBorder	Description forthcoming.
NoBorder	Description forthcoming.
RaisedBezelBorder	Description forthcoming.

Method Types

All methods

EOBoxController

borderType

generateComponent

highlightsTitle

horizontalBorder

setBorderType

setHighlightsTitle

setHorizontalBorder

setTitleColor

setTitleFont

setTitlePosition

setUsesTitledBorder

setVerticalBorder

titleColor

titleFont

titlePosition

usesTitledBorder

verticalBorder

Constructors

EOBoxController

```
public EOBoxController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOBoxController()
```

Description forthcoming.

Instance Methods

borderType

```
public int borderType()
```

Description forthcoming.

generateComponent

```
protected void generateComponent()
```

Description forthcoming.

highlightsTitle

```
public boolean highlightsTitle()
```

Description forthcoming.

horizontalBorder

```
public int horizontalBorder()
```

Description forthcoming.

setBorderType

```
public void setBorderType(int anInt)
```

Description forthcoming.

setHighlightsTitle

```
public void setHighlightsTitle(boolean aBoolean)
```

Description forthcoming.

setHorizontalBorder

```
public void setHorizontalBorder(int anInt)
```

Description forthcoming.

setTitleColor

```
public void setTitleColor(java.awt.Color aColor)
```

Description forthcoming.

setTitleFont

```
public void setTitleFont(java.awt.Font aFont)
```

Description forthcoming.

setTitlePosition

```
public void setTitlePosition(int anInt)
```

Description forthcoming.

setUsesTitledBorder

```
public void setUsesTitledBorder(boolean aBoolean)
```

Description forthcoming.

setVerticalBorder

```
public void setVerticalBorder(int anInt)
```

Description forthcoming.

titleColor

```
public java.awt.Color titleColor()
```

Description forthcoming.

titleFont

```
public java.awt.Font titleFont()
```

Description forthcoming.

titlePosition

```
public int titlePosition()
```

Description forthcoming.

usesTitledBorder

```
public boolean usesTitledBorder()
```

Description forthcoming.

verticalBorder

```
public int verticalBorder()
```

Description forthcoming.

EOComponentController

Inherits from:	EOController : Object
Implements:	NSDisposable (Inherited from EOController) com.webobjects.eocontrol.EOKeyValueCodingAdditions (Inherited from EOController) EOAction.Enabling (Inherited from EOController) com.webobjects.eocontrol.EOKeyValueCoding (Inherited from EOKeyValueCodingAdditions) com.webobjects.foundation.NSKeyValueCoding (Inherited from EOKeyValueCoding)
Package:	com.webobjects.eoapplication

Class Description

The EOComponentController class provides behavior for controllers that manage user interface components. A component controller has a **component** , that represents the user interface for the controller itself (not for its subcontrollers), a **subcontroller area** for displaying the user interfaces for its subcontrollers, and an **integration component** —a component that represents the controller when its shown in its supercontrollers user interface.

By default, a controller's integration component is simply the controller's component. In other words, a supercontroller adds its subcontrollers' components to the subcontroller area of its component. However, the integration component can be a completely separate component. For example, the integration component for a window controller is a button that, when pushed, opens the window controller's window.

Also by default, a controller's subcontroller area is simply the controller's component. In the simplest case, a component controller doesn't have its own user interface, but only serves to display the user interfaces of its subcontrollers. For example, EOComponentController's component is simply an EOView. It puts nothing in the view except its subcontrollers' user interfaces. Thus, the subcontroller area is the controller's component—the EOView. However, the subcontroller area can be a subcomponent of the controller's component. For example, an EOBoxController's component contains a border (etched or bezel, for example) which is the box controller's user interface. Its subcontroller area is a component located inside the border. This is where the box controller displays its subcontrollers.

Managing the Component

To access a component controller's component, use the method `component`. If the component hasn't yet been created, `component` creates it by invoking `prepareComponent`. And `prepareComponent`, in turn, invokes `generateComponent` to dynamically create the component. Subclasses should override `generateComponent`.

To see if a controller's component has been created, use the method `isComponentPrepared`. Sometimes you need to know if a component has been created, because you can't configure its behavior after its creation. For example, if you want to set a component's alignment behavior, you have to set it with the EOComponentController method `setAlignsComponents` before the component controller creates its component.

Visibility

A component controller is visible when its component is visible on screen. When a controller becomes visible, it ensures that it's connected to its supercontroller. However, a controller that's connected to its supercontroller isn't necessarily visible. For example, you might connect an invisible controller when you need to prepare it with data before making it visible.

Similarly, a controller can be "shown" or "hidden" in its supercontroller without changing the controller's visibility. The method `showInSupercontroller` ensures that the receiver's integration component is displayed in its supercontroller's component. This doesn't necessarily change the visibility of the controller. For example, a tab switch controller might switch to another view, but if the switch controller isn't visible when the change occurs, the subcontroller doesn't become visible.

Component Appearance

A component controller's component can have an icon and a label. The component can be represented in the user interface with icon only, label only, or with both icon and label. A component specifies which representation it prefers. A controller can *prefer* to be represented with an icon only, but can't require it. This is because the controller might not have an icon. If the controller prefers icon only and has an icon, then the controller is represented with the icon only. If the controller doesn't prefer icon only and has an icon, then the controller is represented with its icon and label. If the controller doesn't have an icon, the controller is represented with the label only.

A controller always has a label. If the controller's label hasn't been explicitly set, the controller derives one from its subcontrollers.

Layout

Subclasses of EOComponentController have complete control over how they lay out their subcontrollers. EOComponentController's implementation can lay out subcontrollers in a row or a column (the default). To change the layout direction, the method `setUsesHorizontalLayout`.

In addition to horizontal/vertical layout behavior, a component can align its components or not. For example, consider a controller that uses vertical layout and contains several EOTextFieldControllers. If the controller aligns components, it left aligns the text fields. The default alignment behavior aligns components by making their corresponding labels identically sized. The width of the labels is known as the **alignment width** .

To specify a component's alignment behavior, use the method `setAlignsComponents`. To set the alignment width, use `setAlignmentWidth`.

Resizing

EOComponentController implements complex resizing behavior. For example, if a controller's component changes in a way that might affect its minimum size, the controller's supercontroller is notified and the supercontroller ensures that its subcontroller area is at least as big as the minimum size required to show all its subcontrollers.

Using the default behavior, the user interface doesn't automatically shrink. EOComponentController only resizes up to meet the minimum requirements. As much as possible it resizes components to fill the available space. A component controller can specify both horizontal and vertical resizing behavior for its component to accommodate this scheme.

Rule System and XML Description

The following tables identify the `controllerType`, XML tag, and XML attributes used by the rule system and EOXMLUnarchiver to generate a controller hierarchy. For more information, see the section "Rule System and XML Description" (page 8) in the package introduction.

Default Rule System Controller Type

`groupingController`

XML Tag

`COMPONENTCONTROLLER`

XML Attribute	Value	Description
`alignmentWidth`	integer	See "Layout" (page 97).
`alignsComponents`	"true" or "false"	See "Layout" (page 97).
`horizontallyResizable`	"true" or "false"	See "Resizing" (page 97).
`iconName`	string	The filename of the component's icon. Uses standard resource location behavior to find the icon by name. See "Component Appearance" (page 97) for more information.
`iconURL`	string	The URL from which the icon is downloaded. See "Component Appearance" (page 97) for more information.
`label`	string	See "Component Appearance" (page 97).
`minimumHeight`	integer	The minimum height of the controller's component, not including its subcontroller area.
`minimumWidth`	integer	The minimum width of the controller's component, not including its subcontroller area.
`prefersIconOnly`	"true" or "false"	See "Component Appearance" (page 97).
`usesHorizontalLayout`	"true" or "false"	See "Layout" (page 97).
`verticallyResizable`	"true" or "false"	See "Resizing" (page 97).

Constants

Constant	Description
Bottom	Description forthcoming.
BottomLeft	Description forthcoming.
BottomRight	Description forthcoming.
Center	Description forthcoming.
Left	Description forthcoming.
Right	Description forthcoming.
Top	Description forthcoming.
TopLeft	Description forthcoming.
TopRight	Description forthcoming.

Interfaces Implemented

NSDisposable (Inherited from EOController)

- dispose

EOKeyValueCodingAdditions (Inherited from EOController)

EOAction.Enabling (Inherited from EOController)

EOKeyValueCoding (Inherited from EOKeyValueCodingAdditions)

- handleQueryWithUnboundKey
- handleTakeValueForUnboundKey
- unableToSetNullForKey

NSKeyValueCoding (Inherited from EOKeyValueCoding)

Method Types

Constructors

- EOComponentController

Managing the component

- generateComponent
- prepareComponent
- setComponent
- component
- isComponentPrepared

Managing the integration component

integration ComponentDidBecomeInvisible

integrationComponentDidBecomeVisible

integrationComponent

Managing the subcontroller area

setSubcontrollerArea

subcontrollerArea

addComponentOfSubcontroller

removeComponentOfSubcontroller

Managing component visibility

showInSupercontroller

makeVisible

componentDidBecomeVisible

showSubcontroller

hideInSupercontroller

makeInvisible

componentDidBecomeInvisible

hideSubcontroller

setVisible

isVisible

Setting component appearance

setPrefersIconOnly

prefersIconOnly

setIcon

icon

setLabel

label

Layout behavior

setUsesHorizontalLayout

usesHorizontalLayout

setAlignsComponents

alignsComponents

setAlignmentWidth

alignmentWidth

Resizing behavior

setCanResizeHorizontally

canResizeHorizontally

setCanResizeVertically

canResizeVertically

Configuring user interface sizes

setDefaultComponentSize

defaultComponentSize

ensureMinimumComponentSizeWithoutSubcontrollers

ensureMinimumSubcontrollerAreaSize

subcontrollerMinimumSizeDidChange

minimumComponentSize

minimumComponentSizeWithoutSubcontrollers

minimumIntegrationComponentSize

minimumSubcontrollerAreaSize

Determining the root component controller

isRootComponentController

Methods inherited from EOController

canBeTransient

removeTransientSubcontroller

subcontrollerWasAdded

subcontrollerWasRemoved

Methods inherited from Object

toString

Constructors

EOComponentController

```
public EOComponentController(EOXMLUnarchiver unarchiver)
```

Creates a new component controller. For information on how these constructors are used and on what they do, see the method description for the `EOController` constructors in the EOController class specification.

```
public EOComponentController()
```

Description forthcoming.

Instance Methods

addComponentOfSubcontroller

```
protected void addComponentOfSubcontroller(EOComponentController controller)
```

Adds the integration component for the receiver's subcontroller, `controller`, to the user interface for the receiver.

alignmentWidth

```
public int alignmentWidth()
```

Returns the receiver's alignment width.

See Also: "Layout" (page 97)

alignsComponents

```
public boolean alignsComponents()
```

Returns `true` if the receiver aligns its components, `false` otherwise.

See Also: "Layout" (page 97)

canBeTransient

```
public boolean canBeTransient()
```

Returns `true` if the controller can be transient, `false` otherwise. By default, a component controller is transient only if it's an instance of EOComponentController, not an instance of a subclass.

See Also: `canBeTransient` **(EOController)**

canResizeHorizontally

```
public boolean canResizeHorizontally()
```

Returns `true` if the receiver can resize its component horizontally, or `false` otherwise.

See Also: "Resizing" (page 97)

canResizeVertically

```
public boolean canResizeVertically()
```

Returns `true` if the receiver can resize its component vertically, or `false` otherwise.

See Also: "Resizing" (page 97)

component

```
public javax.swing.JComponent component()
```

Returns the receiver's component, creating and preparing it first if it doesn't already exist.

See Also: "Managing the Component" **(page 96)**, prepareComponent, generateComponent

componentDidBecomeInvisible

```
protected void componentDidBecomeInvisible()
```

Invoked by the receiver's supercontroller when the receiver's component becomes invisible, giving the receiver a chance to respond. EOComponentController's implementation invokes breakConnection to break the receiver's connection to the controller hierarchy.

componentDidBecomeVisible

```
protected void componentDidBecomeVisible()
```

Invoked by the receiver's supercontroller when the receiver's component becomes visible, giving the receiver a chance to respond. EOComponentController's implementation invokes establishConnection to ensure the receiver is connected to the controller hierarchy.

defaultComponentSize

```
public java.awt.Dimension defaultComponentSize()
```

Returns the default size for the receiver's component. This is the size the component is set to when it's created.

See Also: "Resizing" **(page 97)**

dispose

```
public void dispose()
```

Conformance to NSDisposable. See the method description of dispose in the interface specification for NSDisposable.

ensureMinimumComponentSizeWithoutSubcontrollers

```
public void ensureMinimumComponentSizeWithoutSubcontrollers(
    int width,
    int height)
```

Ensures that the size of the receiver's component, not including the subcontroller area, is at least as large as the area specified by `width` and `height`. If it isn't, the receiver resizes its component to `width` and `height`. This method is invoked by the receiver itself when its component is changed in a way that might affect the component's minimum size. For example, suppose a label is changed and requires a larger space.

See Also: "Resizing" **(page 97)**

ensureMinimumSubcontrollerAreaSize

```
public void ensureMinimumSubcontrollerAreaSize(
    int width,
    int height)
```

Ensures that the size of the receiver's subcontroller area is at least as large as the area specified by `width` and `height`. If it isn't, the receiver resizes its subcontroller area to `width` and `height`. This method is invoked when a subcontroller's component changes in a way that might affect its minimum size.

See Also: "Resizing" **(page 97)**

generateComponent

```
protected void generateComponent()
```

Creates the receiver's component, including setting up the subcontroller area. Implementations of these methods usually invoke `setComponent` and if necessary `setSubcontrollerArea`. EOComponentController creates an EOView.

See Also: "Managing the Component" **(page 96)**

handleTakeValueForUnboundKey

```
public void handleTakeValueForUnboundKey(
    Object value,
    String key)
```

Conformance to EOKeyValueCoding. See the method description of `handleTakeValueForUnboundKey` in the interface specification for EOKeyValueCoding.

hideInSupercontroller

```
public boolean hideInSupercontroller()
```

Invokes `hideSubcontroller` on the receiver's supercontroller to hide the receiver's component if the component (or integration component) appears in the supercontroller's user interface. Returns `true` on success, `false` otherwise. If the receiver doesn't have a supercontroller, then this method simply makes the receiver invisible. For example, a window controller which is the root component controller simply closes.

This method is invoked automatically (for example, from `makeInvisible`). You should never need to invoke it yourself.

See Also: "Visibility" **(page 96)**

hideSubcontroller

```
protected boolean hideSubcontroller(EOComponentController controller)
```

Hides `controller`'s user interface in the interface of the receiver. Returns `true` if the subcontroller was successfully hidden, `false` otherwise. EOComponentController's implementation simply returns `false`. This is because most controllers can't hide their subcontrollers. Examples of controllers that can hide their subcontrollers are tab view controllers, which hide a subcontroller by making another subcontroller visible. Don't invoke this method directly; invoke `hideInSupercontroller` instead.

See Also: "Visibility" **(page 96)**

icon

```
public javax.swing.Icon icon()
```

Returns the receiver's icon, or `null` if it has none.

See Also: "Component Appearance" **(page 97)**

integrationComponent

```
public javax.swing.JComponent integrationComponent()
```

Returns the component used as the integration component in the receiver's supercontroller to represent the receiver. EOComponentController returns its component by default.

See Also: "Class Description" **(page 95)**

integrationComponentDidBecomeInvisible

```
protected void integrationComponentDidBecomeInvisible()
```

Invoked by the receiver's supercontroller when the receiver's integration component becomes invisible, giving the receiver a chance to respond. EOComponentController's implementation sets the receiver's visibility to be `false`, because by default the integration component is identical to the component.

integrationComponentDidBecomeVisible

```
protected void integrationComponentDidBecomeVisible()
```

Invoked by the receiver's supercontroller when the receiver's integration component becomes visible, giving the receiver a chance to respond. EOComponentController's implementation sets the receiver's visibility to be `true`, because by default the integration component is identical to the component.

isComponentPrepared

```
protected boolean isComponentPrepared()
```

Returns `true` if the receiver is prepared, `false` otherwise.

See Also: "Managing the Component" **(page 96)**

isRootComponentController

```
public boolean isRootComponentController()
```

Returns `true` if the receiver is a root component controller, `false` otherwise. A component controller is the root component controller if its supercontroller is not an instance of EOComponentController.

isVisible

```
public boolean isVisible()
```

Returns `true` if the receiver is visible, `false` otherwise. A component controller is visible if its component is on the screen. Note, showing a subcontroller in its supercontroller doesn't necessarily mean that it is visible. For example, you can show a component in a tab view, but the component won't be visible unless the tab view is visible.

See Also: "Visibility" **(page 96)**

label

```
public String label()
```

Returns the receiver's label. If the label is not explicitly set, EOComponentController's implementation attempts to derive a label from it's subcontrollers.

See Also: "Component Appearance" **(page 97)**

makeInvisible

```
public boolean makeInvisible()
```

Makes the receiver's user interface invisible. If the receiver's supercontroller is a component controller, makes the receiver invisible by making the receiver's supercontroller invisible. Otherwise, invokes `hideInSupercontroller`. Returns `true` if the method succeeds in making the receiver invisible, `false` otherwise.

makeVisible

```
public boolean makeVisible()
```

Makes the receiver's user interface visible. Establishes the receiver's connection to its supercontrollers and invokes `showInSupercontroller`. If the receiver's supercontroller is a component controller, it also attempts to make the supercontroller visible. Returns `true` if the method succeeds in making the receiver visible, `false` otherwise.

See Also: "Visibility" **(page 96)**

minimumComponentSize

```
public java.awt.Dimension minimumComponentSize()
```

Returns the current minimum size required to display the receiver's component, including the size required for its subcontroller area.

See Also: "Resizing" **(page 97)**

minimumComponentSizeWithoutSubcontrollers

```
public java.awt.Dimension minimumComponentSizeWithoutSubcontrollers()
```

Returns the current minimum size required to display the receiver's component, excluding the subcontroller area.

See Also: "Resizing" **(page 97)**

minimumIntegrationComponentSize

```
public java.awt.Dimension minimumIntegrationComponentSize()
```

Returns the minimum size required to display the receiver's integration component.

See Also: "Resizing" **(page 97)**

minimumSubcontrollerAreaSize

```
public java.awt.Dimension minimumSubcontrollerAreaSize()
```

Returns the minimum size of the subcontroller area to display the receiver's subcontrollers.

See Also: "Resizing" **(page 97)**

prefersIconOnly

```
public boolean prefersIconOnly()
```

Returns `true` if the receiver prefers to represent itself with only an icon, `false` otherwise.

See Also: "Component Appearance" **(page 97)**

prepareComponent

```
protected void prepareComponent()
```

If the receiver's component is not already prepared, it generates the component.

See Also: "Managing the Component" **(page 96)**

removeComponentOfSubcontroller

```
protected void removeComponentOfSubcontroller(EOComponentController controller)
```

Removes the user interface for the specified subcontroller, `controller`, from the receiver's user interface and informs `controller` that its integration component became invisible.

removeTransientSubcontroller

```
protected boolean removeTransientSubcontroller(EOController controller)
```

See the method description for `removeTransientSubcontroller` in the EOController class specification.

setAlignmentWidth

```
public void setAlignmentWidth(int alignmentWidgth)
```

Sets the receiver's alignment width to `alignmentWidth`. Throws an IllegalStateException if the receiver is already prepared. In other words, you can only set the alignment width before the component is generated.

See Also: "Layout" **(page 97)**

setAlignsComponents

```
public void setAlignsComponents(boolean flag)
```

Sets according to `flag` whether the receiver aligns the components in its user interface. Throws an IllegalStateException if the receiver is already prepared. In other words, you can only set the alignment behavior before the component is generated.

See Also: "Layout" **(page 97)**

setCanResizeHorizontally

```
public void setCanResizeHorizontally(boolean flag)
```

Sets according to `flag` whether the receiver's component can resize horizontally. Throws an IllegalStateException if the receiver is already prepared. In other words, you can only set the horizontal resizing behavior before the component is generated.

See Also: "Resizing" **(page 97)**

setCanResizeVertically

```
public void setCanResizeVertically(boolean flag)
```

Sets according to `flag` whether the receiver's component can resize vertically. Throws an IllegalStateException if the receiver is already prepared. In other words, you can only set the vertical resizing behavior before the component is generated.

See Also: "Resizing" **(page 97)**

setComponent

```
public void setComponent(java.awt.Component component)
```

Sets the receiver's component to `component`.

See Also: "Managing the Component" **(page 96)**

setDefaultComponentSize

```
public void setDefaultComponentSize(java.awt.Dimension dimension)
```

Sets the default size of the receiver's component to `dimension`.

See Also: "Resizing" **(page 97)**

setIcon

```
public void setIcon(javax.swing.Icon icon)
```

Sets the receiver's icon to `icon`.

See Also: "Component Appearance" **(page 97)**

setLabel

```
public void setLabel(String label)
```

Sets the receiver's label to `label`.

See Also: "Component Appearance" **(page 97)**

setPrefersIconOnly

```
public void setPrefersIconOnly(boolean flag)
```

Sets according to `flag` whether the receiver prefers to represent itself with only an icon or with an icon and a label.

See Also: "Component Appearance" **(page 97)**

setSubcontrollerArea

```
public void setSubcontrollerArea(javax.swing.JComponent component)
```

Sets the component that holds the user interface for the receiver's subcontrollers to `component`.

See Also: "Class Description" **(page 95)**

setUsesHorizontalLayout

```
public void setUsesHorizontalLayout(boolean flag)
```

Sets according to `flag` whether the receiver uses horizontal layout. Throws an IllegalStateException if the receiver is already prepared. In other words, you can only set the layout direction before the component is generated.

See Also: "Layout" **(page 97)**

setVisible

```
public void setVisible(boolean flag)
```

Sets the visibility of the receiver according to `flag`. Invokes `componentDidBecomeVisible` or `componentDidBecomeInvisible` to notify the receiver that its visibility changed and to give the receiver the opportunity to respond appropriately. Also notifies the receiver's ancestors that a subcontroller's visibility has changed, giving the supercontrollers the opportunity to respond.

If `flag` is `true`, this method disposes of transient receivers after making them visible.

See Also: "Visibility" **(page 96)**

showInSupercontroller

```
public boolean showInSupercontroller()
```

Invokes `showSubcontroller` to add the receiver's user interface to its supercontroller's receiver. Returns `true` on success, `false` otherwise. If the supercontroller is `null`, this method also makes the receiver visible.

Note: Invoking this method doesn't necessarily change the visibility of the receiver. For example, a switch controller might switch the component it displays, but if the switch controller isn't visible, the subcontroller doesn't become visible when it's shown.

This method is invoked automatically (for example, from `makeVisible`). You should never need to invoke it yourself.

See Also: "Visibility" **(page 96)**

showSubcontroller

```
protected boolean showSubcontroller(EOComponentController controller)
```

Adds `controller`'s user interface to the interface of the receiver. Returns `true` if the subcontroller was successfully shown, `false` otherwise. EOComponentController's implementation simply returns `true`: Since the integration components for subcontrollers are added to a controller's user interface automatically, the subcontrollers are already shown. EOTabSwitchController is an example of a subclass that overrides this method in a meaningful way. To show one subcontroller, the tab switch controller hides another.

See Also: "Visibility" **(page 96)**

subcontrollerArea

```
public javax.swing.JComponent subcontrollerArea()
```

Returns the component that holds the user interface for the receiver's subcontrollers.

See Also: "Class Description" **(page 95)**

subcontrollerMinimumSizeDidChange

```
public void subcontrollerMinimumSizeDidChange(
    EOComponentController controller,
    javax.swing.JComponent component,
    java.awt.Dimension dimension)
```

Updates the receiver's user interface to accommodate a change to the subcontroller's minimum size. This method is invoked by subcontrollers when they change in a way that might affect their component's minimum size. A subcontroller sends this method with itself, its integration component, and its new minimum size as the arguments. The expectation is that the supercontroller will make space for the subcontroller if it needs to.

See Also: "Resizing" **(page 97)**

subcontrollerWasAdded

```
protected void subcontrollerWasAdded(EOController controller)
```

Invokes `addComponentOfSubcontroller` to add the integration component (if any) for the receiver's subcontroller, `controller`, to the receiver's user interface. Invoked from `addSubcontroller` to notify the receiver that its subcontroller `controller` has been added to the controller hierarchy.

subcontrollerWasRemoved

```
protected void subcontrollerWasRemoved(EOController controller)
```

Invokes `removeComponentOfSubcontroller` to remove the integration component (if any) for the receiver's subcontroller, `controller`, from the receiver's user interface. Invoked from `removeSubcontroller` to notify the receiver that its subcontroller `controller` has been removed from the controller hierarchy.

toString

```
public String toString()
```

Returns the receiver as a string that states the receiver's class name and type name, whether the receiver is connected, the number of subcontrollers, whether or not the receiver has been prepared, whether or not the receiver is visible, information about widget sizing and alignment behavior, and so on.

usesHorizontalLayout

```
public boolean usesHorizontalLayout()
```

Returns `true` if the receiver uses a horizontal layout, `false` otherwise.

See Also: "Layout" **(page 97)**

EOController

Inherits from:	Object
Implements:	NSDisposable NSKeyValueCodingAdditions EOAction.Enabling NSKeyValueCoding.ErrorHandling NSKeyValueCoding
Package:	com.webobjects.eoapplication

Class Description

The EOController class defines basic behavior for controller objects that are responsible for managing and sometimes generating the user interface for the client side of a Java Client application. An application's controllers are arranged in a hierarchy, which describes the complete functionality of an application.

The controller hierarchy mirrors the hierarchy of windows and widgets that make up the client application's user interface. The root of the hierarchy is an EOApplication object. The EOApplication's subcontrollers are usually window or applet controllers, which themselves have subcontrollers.

The most significant functionality provided by the EOController class is managing the controller hierarchy (building, connecting, and traversing the hierarchy) and handling actions.

Building the Controller Hierarchy

EOController defines methods for building the controller hierarchy. You can add and remove controllers (addSubcontroller, removeFromSupercontroller), be notified when the controller hierarchy changes (subcontrollerWasAdded and subcontrollerWasRemoved), and inquire about the relationships controllers have to one another (subcontrollers, supercontroller, isAncestorOfController, and isSupercontrollerOfController).

You might need to directly invoke the methods addSubcontroller and removeFromSupercontroller to programmatically manipulate the controller hierarchy. The base implementations of these methods are sufficient for most subclasses. They set and unset a controller's supercontroller (setSupercontroller) and notify that supercontroller that a subcontroller was added or removed.

If you write a custom controller and you need to do something special when a subcontroller is added to or removed from the controller hierarchy, override the methods subcontrollerWasAdded and subcontrollerWasRemoved to put your customizations there. Taking this approach, you shouldn't have to override the add and remove methods.

Traversing the Controller Hierarchy

EOController defines numerous methods for traversing the controller hierarchy, but a single method provides the basic traversal functionality. The method controllerEnumeration creates and returns an enumeration that includes all the descendents of a controller (not including the controller), all the ancestors of a controller (not including the controller), or a controller and its descendants. You can further restrict the controllers included in an enumeration by specifying an interface the controllers must implement in order to be included. For more information, see the EOController.Enumeration interface specification and the method description for controllerEnumeration.

Other methods that traverse the controller hierarchy use a controller enumeration to perform the traversal. There are methods that return controllers in an enumeration that match one or more key-value pairs. Methods that use key-value coding on the controllers in an enumeration, returning the first controller that has a specified key or returning the value for that key. Also, there's a method (invokeMethod) that invokes a particular method on the controllers in an enumeration.

Connecting Controllers

A controller in the controller hierarchy can be connected to its supercontroller or not. Controllers are connected when they're performing their duties, and they are disconnected when they become idle. Generally controllers are connected only when their user interface is visible. For example, the controllers associated with a window are connected when the window is visible, and they're disconnected when the window becomes invisible.

When a controller *connects* to its supercontroller, it gets from its supercontroller whatever resources or information it needs, and it prepares itself in whatever way necessary to perform its duties (for example, setting delegates). Similarly, when a controller breaks its connection to its supercontroller, it cleans up its resources for an idle period.

The EOController class defines methods for connecting controllers. There are methods for connecting and disconnecting a controller from its supercontroller (`establishConnection` and `breakConnection`), and also methods that make connections all the way up the controller hierarchy (`establishConnectionToSupercontrollers`) and break connections all the way down (`breakConnectionToSubcontrollers`). Generally you use the latter methods that connect or disconnect an entire branch of a tree. EOController's implementations of all these methods is generally sufficient for subclasses. They set the connection status of a controller (`setConnected`), and notify the controller that its connection has been established or broken. You shouldn't have to override these methods.

If you do need to do something when a controller is connected or disconnected, you should override the methods `connectionWasEstablished` and `connectionWasBroken`. These methods are invoked automatically by `establishConnection` and `breakConnection`.

Accessing and Enabling Actions

Controllers define actions that users can perform (such as quitting the application) and they know how to respond to those actions when they're performed. EOController defines methods that manage a controllers actions.

A controller has a set of actions. It also keeps track of which of those actions are enabled and which are disabled. For performance reasons, EOController's method implementations cache some of this information. Thus, whenever you do something that changes a controller's actions (such as adding a new subcontroller or enabling or disabling an action), the caches must be reset. Most of the time they're reset automatically, but subclasses might need to explicitly reset them with the method `resetActions`.

To specify the actions a subclass understands, override the method defaultActions. However, to find out what actions a controller understands, use actions. This method simply manages and returns a cache of the methods returned by defaultActions. Some implementations of a defaultActions method are potentially costly to invoke over and over again, because they dynamically build their collections of actions. The actions method is simply an optimization. EOController's implementation of actions should be sufficient for subclasses; you should never need to override it.

To find out what actions a controller can perform at a specific point in time, use the method enabledActions. This method returns only the controller's actions that aren't explicitly disabled. As with actions, enabledActions manages and returns a cache of methods, and EOController's implemenation should be sufficient for subclasses.

Transience

Some controllers are needed only to dynamically generate the user interface and don't serve any purpose after the user interface has been created and connected. For example, an EOTextFieldController creates a widget and a corresponding association and then is no longer needed. Controllers such as EOTextFieldController can be **transient** , because after their work is done, they can sometimes be removed from the controller hierarchy and disposed of (with disposeIfTransient). This keeps the controller hierarchy simple, which makes user interface management more efficient.

Controllers specify whether or not they can be transient by overriding the method canBeTransient. Some controllers can be transient sometimes and not other times, so not all implementations simply return true or false. For example, an EOTableController can be transient if the double click action is unassigned. If the action is assigned, however, the controller must listen for a double click and react when one occurs.

Subclasses that can be transient should invoke the method disposeIfTransient as soon as their work is done and they can be disposed of. Sometimes a controller's supercontroller doesn't allow the controller to be disposed of. For example, the EOTabSwitchComponent doesn't allow its subcontrollers to be disposed of even if they're transient.

Rule System and XML Description

The following tables identify the `controllerType`, XML tag, and XML attributes used by the rule system and EOXMLUnarchiver to generate a controller hierarchy. For more information, see the section "Rule System and XML Description" (page 8) in the package introduction.

Default Rule System Controller Type

None

XML Tag

None

XML Attribute	Value	Description
`className`	string	Name of a class to instantiate instead of the default class.
`disabledActionNames`	array of strings	Names of actions to explicitly disable.
`typeName`	string	This is usually a textual representation of the specification used to generate the controller, for example "question = window, task = query". The type name is used by the controller factory to identify which windows are the same so that it can reuse resources. The typeName is also used by the defaults system to specify per-window defaults.

Constants

EOController defines the following `int` constants to identify types of enumerations returned by the method `controllerEnumeration`:

Constant	Description
`SubcontrollersEnumeration`	An enumeration object that enumerates over a controller's subcontrollers, not including the controller itself.
`SupercontrollersEnumeration`	An enumeration object that enumerates over a controller's supercontrollers, not including the controller itself.
`ControllerAndSubcontrollersEnumeration`	An enumeration object that enumerates over a controller and its subcontrollers.

Interfaces Implemented

NSDisposable

dispose

NSKeyValueCoding (Inherited from EOKeyValueCoding)

takeValueForKey

valueForKey

EOKeyValueCoding (Inherited from EOKeyValueCodingAdditions)

handleQueryWithUnboundKey

handleTakeValueForUnboundKey

unableToSetNullForKey

EOKeyValueCodingAdditions

takeValueForKeyPath

valueForKeyPath

EOAction.Enabling

canPerformActionNamed

Method Types

Constructors

EOController

Managing the controller hierarchy

addSubcontroller

subcontrollerWasAdded

removeFromSupercontroller

removeSubcontroller

subcontrollerWasRemoved

setSupercontroller

removeTransientSubcontroller

canBeTransient

subcontrollers

supercontroller

isAncestorOfController

isSupercontrollerOfController

Traversing the controller hierarchy

controllerEnumeration

controllersInEnumeration

controllerWithKeyValuePair

controllerWithKeyValuePairs

controllersWithKeyValuePair

controllersWithKeyValuePairs

hierarchicalControllerForKey

hierarchicalValueForKey

invokeMethod

Connecting controllers

establishConnectionToSupercontrollers

establishConnection

connectionWasEstablished

breakConnectionToSubcontrollers

breakConnection

connectionWasBroken

setConnected

isConnected

Accessing and enabling actions

actions

defaultActions

enabledActions

actionWithName

actionNames

disableActionNamed

enableActionNamed

isActionNamedEnabled

resetActions

Reusing controllers

prepareForNewTask

Accessing the type name

typeName

setTypeName

Accessing keys

canAccessFieldsDirectly

Disposing

disposeIfTransient

disposableRegistry

Methods inherited from Object

toString

Constructors

EOController

```
public EOController()
```

Description forthcoming.

```
public EOController(EOXMLUnarchiver unarchiver)
```

Creates and returns a new controller. The no argument constructor is used when you create a controller programmatically, whereas the version taking an unarchiver is used in a Direct to Java Client applications to create controllers from an XML description.

Controller subclasses should implement both constructors. Most commonly, controllers are created with the assistance of an unarchiver. For more information on this unarchiving, see the book *Getting Started with Direct to Java Client*.

Static Methods

canAccessFieldsDirectly

```
public static boolean canAccessFieldsDirectly()
```

Returns `true` if the receiver accesses its instance variables directly or `false` otherwise.By default, controllers don't access instance variables directly and return `false`.

See Also: `accessInstanceVariablesDirectly` **(EOCustomObject)**

Instance Methods

actionNames

```
public NSArray actionNames()
```

Returns an array of strings naming the actions the controller defines and responds to.

See Also: "Accessing and Enabling Actions" **(page 121)**

actionWithName

```
public EOAction actionWithName(String actionName)
```

If the receiver has an action named `actionName`, this method returns that action; otherwise, the method returns `null`.

See Also: "Accessing and Enabling Actions" **(page 121)**

actions

```
public NSArray actions()
```

Returns an array containing the receiver's actions. EOController's implementation caches the result of `defaultActions` and returns that. The cache is cleared with the method `resetActions`.

See Also: "Accessing and Enabling Actions" **(page 121)**

addSubcontroller

```
public void addSubcontroller(EOController subcontroller)
```

Adds `controller` as a subcontroller of the receiver and sets the receiver as `controller`'s supercontroller (first removing `controller` from it's supercontroller if it already has one). Invoke this method to programmatically add a subcontroller to the hierarchy.

EOController's implementation sets subcontroller's supercontroller and notifies the receiver that a subcontroller was added. It does nothing if controller is a supercontroller of the receiver. The default implantation of this method should be sufficient for most subclasses; you shouldn't have to override it. If you need to do something special when a subcontroller is added, override subcontrollerWasAdded.

See Also: "Building the Controller Hierarchy" **(page 120)**

breakConnection

```
public void breakConnection()
```

Breaks the receiver's connection to its supercontroller. Invokes connectionWasBroken to give the receiver a chance to clean up, and informs all its ancestors that a subcontroller's connections have changed so the ancestors can respond appropriately. Use this method to programmatically disconnect a single controller (and not its subcontrollers).

EOController's implementation is sufficient for most subclasses, so you don't ordinarily override this method.

See Also: "Connecting Controllers" **(page 121)**

breakConnectionToSubcontrollers

```
public void breakConnectionToSubcontrollers()
```

Breaks the connections the receiver's subcontrollers have to their subcontrollers, and then breaks the receiver's connections to its subcontrollers. This method is invoked recursively down the subcontroller chain until the receiver and all its subcontrollers are disconnected. Use this method to programmatically disconnect a branch of the controller hierarchy from a particular controller down.

EOController's implementation is sufficient for most subclasses, so you don't ordinarily override this method.

See Also: "Connecting Controllers" **(page 121)**

canBeTransient

```
public boolean canBeTransient()
```

Returns `true` if the controller can be transient, `false` otherwise. EOController's implementation returns `false`.

See Also: "Transience" (page 122)

canPerformActionNamed

```
public boolean canPerformActionNamed(String actionName)
```

Conformance to EOAction.Enabling. See the method description of `canPerformActionNamed` in the interface specification for EOAction.Enabling.

See Also: `isActionNamedEnabled`, "Accessing and Enabling Actions" (page 121)

connectionWasBroken

```
protected void connectionWasBroken()
```

Invoked from `breakConnection` to notify the receiver that its connection to its supercontroller has been broken, giving the receiver the opportunity to clean up after its become idle.

See Also: "Connecting Controllers" (page 121)

connectionWasEstablished

```
protected void connectionWasEstablished()
```

Invoked from `establishConnection` to notify the receiver that its connection to the controller hierarchy has been established, giving the receiver the opportunity to prepare itself (for example, setting delegates).

See Also: "Connecting Controllers" (page 121)

controllerEnumeration

```
public EOController.Enumeration controllerEnumeration(
    int enumerationType,
    Class controllerInterface)
```

Returns an enumeration object of the specified type and interface. For example, invoking this method with SubcontrollersEnumeration as the enumerationType and with MyControllerInterface as the controllerInterface returns an enumeration object that returns the receiver's subcontrollers that implement the interface MyControllerInterface.

The enumerationType argument can be one of:

- SubcontrollersEnumeration
- SupercontrollersEnumeration
- ControllerAndSubcontrollersEnumeration

The controllerInterface argument can be the name of an interface or null to specify no interface, which returns all the controllers specified by enumerationType.

See Also: "Traversing the Controller Hierarchy" **(page 120)**, EOController.Enumeration **interface specification**

controllersInEnumeration

```
public NSArray controllersInEnumeration(
    int enumerationType,
    Class controllerInterface)
```

Returns the controllers in an enumeration specified by enumerationType and controllerInterface.

See Also: controllerEnumeration

controllersWithKeyValuePair

```
public NSArray controllersWithKeyValuePair(
    int enumerationType,
    Class controllerInterface,
    String key,
    Object value)
```

Traverses the controller hierarchy, and returns the controllers in the hierarchy whose values for `key` match `value`. This method uses a controller enumeration specified by `enumerationType` and `controllerInterface` to find the controllers. The method tests the controllers returned by the enumeration for matches and returns them. Matches are determined with the method `valueForKeyPath`.

See Also: "Traversing the Controller Hierarchy" **(page 120)**, `controllerEnumeration`

controllersWithKeyValuePairs

```
public NSArray controllersWithKeyValuePairs(
    int enumerationType,
    Class controllerInterface,
    NSDictionary keyValuePairs)
```

Traverses the controller hierarchy, and returns the controllers in the hierarchy whose key-value pairs match those specified in `keyValuePairs`. This method uses a controller enumeration specified by `enumerationType` and `controllerInterface` to find the controllers. The method tests the controllers returned by the enumeration for matches and returns them. Matches are determined with the method `valueForKeyPath`.

See Also: "Traversing the Controller Hierarchy" **(page 120)**, `controllerEnumeration`

controllerWithKeyValuePair

```
public EOController controllerWithKeyValuePair(
    int enumerationType,
    Class controllerInterface,
    String key,
    Object value)
```

Traverses the controller hierarchy, and returns the first controller in the hierarchy whose value for key matches value. This method uses a controller enumeration specified by enumerationType and controllerInterface to find the controller. The method tests the controllers returned by the enumeration for a match and returns the first that it matches. Matches are determined with the method valueForKeyPath.

See Also: "Traversing the Controller Hierarchy" **(page 120)**, controllerEnumeration

controllerWithKeyValuePairs

```
public EOController controllerWithKeyValuePairs(
    int enumerationType,
    Class controllerInterface,
    NSDictionary keyValuePairs)
```

Traverses the controller hierarchy, and returns the first controller in the hierarchy whose key-value pairs match those specified in keyValuePairs. This method uses a controller enumeration specified by enumerationType and controllerInterface to find the controller. The method tests the controllers returned by the enumeration for a match and returns the first that it matches. Matches are determined with the method valueForKeyPath.

See Also: "Traversing the Controller Hierarchy" **(page 120)**, controllerEnumeration

defaultActions

```
protected NSArray defaultActions()
```

Returns an array of the receiver's default actions (EOAction objects). A subclass of EOController should override this method to return the action it defines merged with the actions of its supercclass. Never invoke this method directly. Instead, invoke actions, which caches the results of defaultActions and is therefore more efficient.

See Also: "Accessing and Enabling Actions" **(page 121)**

disableActionNamed

```
public void disableActionNamed(String actionName)
```

Disables the action specified by the name `actionName` and resets the receiver's actions.

See Also: "Accessing and Enabling Actions" **(page 121)**

disposableRegistry

```
public NSDisposableRegistry disposableRegistry()
```

Returns the receiver's disposable registry. This registry contains objects that will be disposed of together with the receiver. Subclasses can use the registry to register objects that should be disposed when their controller is disposed.

dispose

```
public void dispose()
```

Conformance to NSDisposable. See the method description of `dispose` in the interface specification for NSDisposable.

disposeIfTransient

```
protected boolean disposeIfTransient()
```

Disposes the receiver if it's transient, first removing it from its supercontroller with `removeTransientSubcontroller`. Returns `true` if the receiver is transient and has been disposed, `false` otherwise. If the receiver's supercontroller is non-`null`, this method also attempts to dispose of the supercontroller if it's transient.

See Also: "Transience" **(page 122)**, `removeTransientSubcontroller`

enableActionNamed

```
public void enableActionNamed(String actionName)
```

Enables the action named `actionName` and resets the receiver's actions.

See Also: "Accessing and Enabling Actions" **(page 121)**

enabledActions

```
public NSArray enabledActions()
```

Returns an array of the receiver's enabled actions—those of the receiver's EOAction objects that aren't explicitly disabled. This method caches the enabled actions to enhance performance. The cache is cleared with the method `resetActions`.

See Also: "Accessing and Enabling Actions" **(page 121)**

establishConnection

```
public void establishConnection()
```

Connects the receiver to the controller hierarchy. Invokes `connectionWasEstablished` to give the receiver a chance to ready the user interface. After connecting the receiver, this method disposes of it if it's transient and is therefore no longer needed. Use this method to programmatically connect a single controller (and not its ancestors).

EOController's implementation is sufficient for most subclasses, so you don't ordinarily override this method.

See Also: "Connecting Controllers" **(page 121)**

establishConnectionToSupercontrollers

```
public void establishConnectionToSupercontrollers()
```

Connects the receiver's supercontroller to the controller hierarchy, and then establishes the receiver's connection to the controller hierarchy. This method is invoked recursively up the supercontroller chain until the receiver and all its ancestors are connected. Use this method to programmatically prepare a branch of the controller hierarchy from a controller up to the root controller.

EOController's implementation is sufficient for most subclasses, so you don't ordinarily override this method.

See Also: "Connecting Controllers" **(page 121)**

handleQueryWithUnboundKey

```
public Object handleQueryWithUnboundKey(String key)
```

Conformance to EOKeyValueCoding. See the method description of `handleQueryWithUnboundKey` in the interface specification for EOKeyValueCoding.

handleTakeValueForUnboundKey

```
public void handleTakeValueForUnboundKey(
    Object value,
    String key)
```

Conformance to EOKeyValueCoding. See the method description of `handleTakeValueForUnboundKey` in the interface specification for EOKeyValueCoding.

hierarchicalControllerForKey

```
public EOController hierarchicalControllerForKey(
    Class controllerInterface,
    String key)
```

Starting at the receiver, searches up the controller hierarchy for the first controller that implements `controllerInterface` and has a non-`null` value for `key`. Returns that controller or `null` if none of the controllers have a non-`null` value for `key`.

See Also: "Traversing the Controller Hierarchy" **(page 120)**, `controllerEnumeration`

hierarchicalValueForKey

```
public Object hierarchicalValueForKey(
    Class controllerInterface,
    String key)
```

Starting at the receiver, searches up the controller hierarchy for the first controller that implements controllerInterface and has a non-null value for key. Returns the value or null if none of the controllers have a non-null value for key.

See Also: "Traversing the Controller Hierarchy" **(page 120),** controllerEnumeration

invokeMethod

```
public void invokeMethod(
    int enumerationType,
    Class controllerInterface,
    String methodName,
    Class[] parameterTypes[],
    Object[] parameters[])
```

Traverses the controller hierarchy, invoking the method specified by methodName and parameterTypes on the appropriate controllers. This method uses a controller enumeration specified by enumerationType and controllerInterface to find the controllers on which to invoke the specified method. For each controller in the enumeration, this method invokes the methodName method with the values in parameters as arguments to the method.

See Also: "Traversing the Controller Hierarchy" **(page 120),** controllerEnumeration

isActionNamedEnabled

```
public boolean isActionNamedEnabled(String actionName)
```

Returns true if the action specified by actionName isn't specifically disabled, false otherwise.

See Also: "Accessing and Enabling Actions" **(page 121)**

isAncestorOfController

```
public boolean isAncestorOfController(EOController controller)
```

Returns `true` if `controller` is a subcontroller of the receiver, of the receiver's subcontrollers, or their subcontrollers, and so on; `false` otherwise.

isConnected

```
public boolean isConnected()
```

Returns `true` if the receiver is connected, `false` otherwise.

See Also: "Connecting Controllers" **(page 121)**

isSupercontrollerOfController

```
public boolean isSupercontrollerOfController(EOController controller)
```

Returns `true` if `controller` is an immediate subcontroller of the receiver, `false` otherwise.

prepareForNewTask

```
public void prepareForNewTask(boolean prepareSubcontrollersForNewTask)
```

Prepares the receiver for performing a new task by resetting any data. If `prepareSubcontrollersForNewTask` is `true`, this method also sends `prepareForNewTask` to each of the receiver's subcontrollers. This method is invoked to prepare a branch of the controller hierarchy to be reused. Subclasses should override this method to get rid of data and perform any additional clean up.

removeFromSupercontroller

```
public void removeFromSupercontroller()
```

Removes the receiver from its supercontroller's set of subcontrollers. Invoke this method when you need to programmatically remove a controller from the controller hierarchy.

EOController's implementation simply invokes `removeSubcontroller` on the receiver's supercontroller. This method is a convenience so you don't have to look up a controller's supercontroller. The default implementation should be sufficient for subclasses; you shouldn't have to override it.

See Also: "Building the Controller Hierarchy" **(page 120)**

removeSubcontroller

```
protected void removeSubcontroller(EOController subcontroller)
```

Removes `subcontroller` from the controller hierarchy. EOController's implementation disconnects `subcontroller` from the controller hierarchy, and invokes `subcontrollerWasRemoved` on the receiver to give the receiver a chance to respond appropriately. Never invoke this method directly; use `removeFromSupercontroller` instead. The default implementation should be sufficient for subclasses; you shouldn't have to override it. If you need to do something when a subcontroller is removed, implement `subcontrollerWasRemoved`.

See Also: "Building the Controller Hierarchy" **(page 120)**

removeTransientSubcontroller

```
protected boolean removeTransientSubcontroller(EOController subcontroller)
```

Removes `subcontroller` from the controller hierarchy if `subcontroller` can be transient and if the receiver allows it. Returns `true` if the subcontroller could be removed, `false` otherwise. This method is invoked from `disposeIfTransient`, which is invoked in various situations to remove controllers as soon as they can become transient.

See Also: "Transience" **(page 122)**

resetActions

```
public void resetActions()
```

Destroys the receiver's cache of actions and enabled actions, and destroys the action caches of the receiver's supercontrollers. This method is generally invoked automatically when the receiver's set of actions changes or when an action's enabled state is changed, but you can invoke it yourself to clear the caches as needed. EOController's implementation of this method is sufficient for most subclasses. You shouldn't have to override it.

See Also: "Accessing and Enabling Actions" **(page 121)**

setConnected

```
protected void setConnected(boolean flag)
```

Sets the receiver's connected status according to `flag`. EOController's implementation is sufficient for most subclasses; you don't normally override this method. Nor should you ever need to invoke it; `establishConnection` and `breakConnection` set the controller's connection status automatically.

See Also: "Connecting Controllers" **(page 121)**

setSupercontroller

```
protected boolean setSupercontroller(EOController controller)
```

Sets the receiver's supercontroller to `controller` and resets the receiver's actions. Returns `true` on success or `false` otherwise. It fails if `controller` is unacceptable as the receiver's supercontroller. Also, `controller` can be `null` to unset the receiver's supercontroller.

EOController's implementation is sufficient for most subclasses; you don't normally override this method. Nor should you ever need to invoke it; `addSubcontroller` sets the supercontroller automatically.

See Also: "Building the Controller Hierarchy" **(page 120)**

setTypeName

```
public void setTypeName(String typeName)
```

Sets the receiver's type name to typeName.

See Also: "Rule System and XML Description" **(page 123)**

subcontrollers

```
public NSArray subcontrollers()
```

Returns the receiver's immediate subcontrollers. Use controllerEnumeration of controllersInEnumeration to return all the controllers in the hierarchy under the receiver.

subcontrollerWasAdded

```
protected void subcontrollerWasAdded(EOController subcontroller)
```

Invoked from addSubcontroller to notify the receiver that its subcontroller subcontroller has been added to the controller hierarchy, giving the receiver the opportunity to prepare the subcontroller for use.

See Also: "Building the Controller Hierarchy" **(page 120)**

subcontrollerWasRemoved

```
protected void subcontrollerWasRemoved(EOController subcontroller)
```

Invoked from removeSubcontroller to notify the receiver that its subcontroller subcontroller has been removed from the controller hierarchy, giving the receiver the opportunity to perform any necessary clean up.

See Also: "Building the Controller Hierarchy" **(page 120)**

supercontroller

```
public EOController supercontroller()
```

Returns the receiver's supercontroller, or null if the receiver has no supercontroller.

```
public EOController supercontroller(Class controllerInterface)
```

Searching from the receiver's immediate supercontroller, returns the first supercontroller that implements the interface controllerInterface. Returns null if the receiver has no supercontroller or if none of the supercontrollers implement controllerInterface. Returns the receiver's immediate supercontroller if controllerInterface is null.

takeValueForKey

```
public void takeValueForKey(
    Object value,
    String key)
```

Conformance to NSKeyValueCoding. See the method description of valueForKey in the interface specification for NSKeyValueCoding.

takeValueForKeyPath

```
public void takeValueForKeyPath(
    Object value,
    String keyPath)
```

Conformance to EOKeyValueCodingAdditions (com.webobjects.eocontrol). See the method description of takeValueForKeyPath in the interface specification for EOKeyValueCodingAdditions.

toString

```
public String toString()
```

Returns the receiver as a string that states the receiver's class name and type name, whether the receiver is connected, and the number of subcontrollers.

typeName

```
public String typeName()
```

Returns the receiver's type name—a string that uniquely identifies the receiver as a node in the controller hierarchy. EOController's implementation returns `null`. The type name is used to identify controllers that have the same task. It is used to configure a controller with user defaults and also to reuse controllers when possible.

See Also: "Rule System and XML Description" **(page 123)**

unableToSetNullForKey

```
public void unableToSetNullForKey(String key)
```

Conformance to EOKeyValueCoding. See the method description of `unableToSetNullForKey` in the interface specification for EOKeyValueCoding.

valueForKey

```
public Object valueForKey(String key)
```

Conformance to NSKeyValueCoding. See the method description of `valueForKey` in the interface specification for NSKeyValueCoding.

valueForKeyPath

```
public Object valueForKeyPath(String keyPath)
```

Conformance to EOKeyValueCodingAdditions (com.webobjects.eocontrol). See the method description of `valueForKeyPath` in the interface specification for EOKeyValueCodingAdditions.

EODefaultResourceBundle

Inherits from: java.util.ResourceBundle

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Constructors

EODefaultResourceBundle

```
public EODefaultResourceBundle()
```

Description forthcoming.

Instance Methods

getKeys

```
public java.util.Enumeration getKeys()
```

Description forthcoming.

handleGetObject

```
protected Object handleGetObject(String aString) throws java.util.MissingResourceException
```

Description forthcoming.

EODefaults

Inherits from: Object

Implements: NSDisposable

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

`clearAllValues`

`dispose`

`loadPersistentValues`

`savePersistentValues`

`setPersistentValueForKey`

setTransientValueForKey

valueForKey

Constructors

EODefaults

```
public EODefaults()
```

Description forthcoming.

Instance Methods

clearAllValues

```
public void clearAllValues()
```

Description forthcoming.

dispose

```
public void dispose()
```

Description forthcoming.

loadPersistentValues

```
public void loadPersistentValues()
```

Description forthcoming.

savePersistentValues

```
public void savePersistentValues()
```

Description forthcoming.

setPersistentValueForKey

```
public void setPersistentValueForKey(
    Object anObject,
    String aString)
```

Description forthcoming.

setTransientValueForKey

```
public void setTransientValueForKey(
    Object anObject,
    String aString)
```

Description forthcoming.

valueForKey

```
public Object valueForKey(String aString)
```

Description forthcoming.

EODialogController

Inherits from: EOSimpleWindowController :
EOWindowController :
EOComponentController :
EOController :
Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
DIALOGCONTROLLER	windowController

Method Types

All methods

EODialogController

runControllerInNewDialog

generateBorderSize

newWindow

setWindowResizable

setWindowTitle

Constructors

EODialogController

```
public EODialogController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EODialogController()
```

Description forthcoming.

Static Methods

runControllerInNewDialog

```
public static void runControllerInNewDialog(
    EOComponentController anEOComponentController,
    String aString)
```

Description forthcoming.

Instance Methods

generateBorderSize

```
protected java.awt.Dimension generateBorderSize()
```

Description forthcoming.

newWindow

```
protected java.awt.Window newWindow(javax.swing.JComponent aJComponent)
```

Description forthcoming.

setWindowResizable

```
protected void setWindowResizable(
    java.awt.Window aWindow,
    boolean aBoolean)
```

Description forthcoming.

setWindowTitle

```
protected void setWindowTitle(
    java.awt.Window aWindow,
    String aString)
```

Description forthcoming.

EODialogs

Inherits from: Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Constants

Constant	Description
`CancelOperation`	Description forthcoming.
`DefaultOperation`	Description forthcoming.
`OptionalOperation`	Description forthcoming.

Method Types

All methods

runChooseOperationDialog

runConfirmOperationDialog

runConfirmOperationDialog

runErrorDialog

runInformationDialog

Constructors

EODialogs

```
public EODialogs()
```

Description forthcoming.

Static Methods

runChooseOperationDialog

```
public static int runChooseOperationDialog(
    String aString,
    String aString,
    String aString,
    String aString)
```

Description forthcoming.

runConfirmOperationDialog

```
public static boolean runConfirmOperationDialog(
    String aString,
    String aString,
    String aString,
    String aString)
```

Description forthcoming.

runConfirmOperationDialog

```
public static boolean runConfirmOperationDialog(
    String aString,
    String aString,
    String aString)
```

Description forthcoming.

runErrorDialog

```
public static void runErrorDialog(
    String aString,
    String aString)
```

Description forthcoming.

runInformationDialog

```
public static void runInformationDialog(
    String aString,
    String aString)
```

Description forthcoming.

EODisplayUtilities

Inherits from: Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Constants

Constant	Description
`ComponentAlignmentContainerIndicatorKey`	Description forthcoming.
`ComponentAlignmentMinimumWidthKey`	Description forthcoming.
`ComponentAlignmentTitleIndicatorKey`	Description forthcoming.

Method Types

All methods

activateWindow

activateWindowIfVisible

componentPrefersHorizontalResizing

componentPrefersVerticalResizing

displayLabelForString

fillTargetSizeWithUnionSize

fillTargetSizeWithUnionSize

integrateTransientSubcontrollerComponentForController

localizedDisplayLabelForString

locateWindow

locateWindow

minimumComponentSizeWithIntegratedComponents

minimumComponentSizeWithIntegratedComponents

minimumSubcontrollerAreaSizeWithIntegratedComponents

relocateWindow

removeComponentFromParentContainer

tryToRemoveComponent

unionSize

unionSize

updateComponentInContainer

updateComponentInController

Static Methods

activateWindow

```
public static void activateWindow(java.awt.Window aWindow)
```

Description forthcoming.

activateWindowIfVisible

```
public static boolean activateWindowIfVisible(java.awt.Window aWindow)
```

Description forthcoming.

componentPrefersHorizontalResizing

```
public static boolean componentPrefersHorizontalResizing(javax.swing.JComponent aJComponent)
```

Description forthcoming.

componentPrefersVerticalResizing

```
public static boolean componentPrefersVerticalResizing(javax.swing.JComponent aJComponent)
```

Description forthcoming.

displayLabelForString

```
public static String displayLabelForString(String aString)
```

Description forthcoming.

fillTargetSizeWithUnionSize

```
public static void fillTargetSizeWithUnionSize(
    java.awt.Dimension aDimension,
    java.awt.Dimension aDimension,
    java.awt.Dimension aDimension)
```

Description forthcoming.

fillTargetSizeWithUnionSize

```
public static boolean fillTargetSizeWithUnionSize(
    java.awt.Dimension aDimension,
    int anInt,
    int anInt)
```

Description forthcoming.

integrateTransientSubcontrollerComponentForController

```
public static void integrateTransientSubcontrollerComponentForController(
    EOComponentController anEOComponentController,
    EOComponentController anEOComponentController,
    java.awt.Dimension aDimension,
    boolean aBoolean)
```

Description forthcoming.

localizedDisplayLabelForString

```
public static String localizedDisplayLabelForString(String aString)
```

Description forthcoming.

locateWindow

```
public static void locateWindow(
    java.awt.Window aWindow,
    java.awt.Dimension aDimension,
    java.awt.Point aPoint)
```

Description forthcoming.

locateWindow

```
public static void locateWindow(
    java.awt.Window aWindow,
    java.awt.Dimension aDimension,
    int anInt,
    int anInt)
```

Description forthcoming.

minimumComponentSizeWithIntegratedComponents

```
public static java.awt.Dimension minimumComponentSizeWithIntegratedComponents(
    EOComponentController anEOComponentController,
    java.awt.Dimension aDimension,
    java.awt.Dimension aDimension,
    NSArray aNSArray,
    boolean aBoolean)
```

Description forthcoming.

minimumComponentSizeWithIntegratedComponents

```
public static java.awt.Dimension minimumComponentSizeWithIntegratedComponents(
    EOComponentController anEOComponentController,
    java.awt.Dimension aDimension,
    java.awt.Dimension aDimension,
    NSArray aNSArray)
```

Description forthcoming.

minimumSubcontrollerAreaSizeWithIntegratedComponents

```
public static java.awt.Dimension minimumSubcontrollerAreaSizeWithIntegratedComponents(
    java.awt.Dimension aDimension,
    NSArray aNSArray,
    boolean aBoolean)
```

Description forthcoming.

relocateWindow

```
public static void relocateWindow(
    java.awt.Window aWindow,
    java.awt.Dimension aDimension,
    int anInt)
```

Description forthcoming.

removeComponentFromParentContainer

```
public static void removeComponentFromParentContainer(java.awt.Component aComponent)
```

Description forthcoming.

tryToRemoveComponent

```
public static void tryToRemoveComponent(java.awt.Component aComponent)
```

Description forthcoming.

unionSize

```
public static java.awt.Dimension unionSize(
    java.awt.Dimension aDimension,
    java.awt.Dimension aDimension)
```

Description forthcoming.

unionSize

```
public static java.awt.Dimension unionSize(
    int anInt,
    int anInt,
    int anInt,
    int anInt)
```

Description forthcoming.

updateComponentInContainer

```
public static void updateComponentInContainer(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    java.awt.Dimension aDimension,
    boolean aBoolean,
    boolean aBoolean,
    javax.swing.JComponent aJComponent,
    boolean aBoolean)
```

Description forthcoming.

updateComponentInController

```
public static void updateComponentInController(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    java.awt.Dimension aDimension,
    boolean aBoolean,
    boolean aBoolean,
    boolean aBoolean,
    boolean aBoolean)
```

Description forthcoming.

EODocumentController

Inherits from:	EOEntityController : EOComponentController : EOController
Implements:	EODocument EOEditable EOAssociationConnector (Inherited from EOEntityController) EOComponentController.EndEditing (Inherited from EOEntityController) EOObserving (Inherited from EOEntityController) EOObjectDisplay (Inherited from EOEntityController) NSInlineObservable (Inherited from EOController) NSDisposable (Inherited from EOController) EOKeyValueCodingAdditions (Inherited from EOController) EOAction.Enabling (Inherited from EOController) EOKeyValueCoding (Inherited from EOKeyValueCodingAdditions) NSKeyValueCoding (Inherited from EOKeyValueCoding)
Package:	com.webobjects.eoapplication

Class Description

The EODocumentController class provides behavior for displaying and editing enterprise objects in a user interface. EODocumentController's API is mostly specified by the interfaces EODocument and EOEditable. Additionally, much of the way that EODocumentController works is set up by its superclass, EOEntityController. Since EOEntityControllers use EOEditingContexts and EODisplayGroups to manage and display a set of enterprise objects;

EODocumentControllers use them as well. However, in addition to displaying enterprise objects, document controllers can also edit their objects. You can insert, update, and delete enterprise objects; undo and redo unsaved changes; and save and revert.

EODocumentController provides several methods that interact with a user. For example, the methods `revert` and `saveIfUserConfirms` open dialogs to confirm that a user wants to revert or save before performing the action. Also, many of the methods open dialogs when an error occurs, telling the user what happened.

Root Document Controller Responsibilities

EODocumentController defines the concept of a root document controller. A document controller is the root if none of its ancestors are EODocuments. A root document controller usually provides the editing context for all its descendent document controllers—they typically don't have their own. Consequently, the root document controller has responsibilities that non-root document controllers don't have. For example, only the root document controller provides save and revert behavior.

Rule System and XML Description

The following tables identify the `controllerType`, XML tag, and XML attributes used by the rule system and EOXMLUnarchiver to generate a controller hierarchy. For more information, see the section "Rule System and XML Description" (page 8) in the package introduction.

Default Rule System Controller Type

`entityController`

XML Tag

`DOCUMENTCONTROLLER`

XML Attribute	Value	Description
`editability`	string	One of "Never", "Always", or "IfSupercontroller". See the `EOEditable` interface specification for more information on these settings.

Interfaces Implemented

EODocument

- isDocumentForGlobalID
- isEdited
- save
- saveIfUserConfirms
- setEdited

EOEditable

- editability
- isEditable
- setEditability
- supercontrollerEditabilityDidChange
- takeResponsibilityForEditabilityOfAssociation

EOAssociationConnector (Inherited from EOEntityController)

EOComponentController.EndEditing (Inherited from EOEntityController)

EOObserving (Inherited from EOEntityController)

EOObjectDisplay (Inherited from EOEntityController)

NSInlineObservable (Inherited from EOController)

NSDisposable (Inherited from EOController)

- dispose

EOKeyValueCodingAdditions (Inherited from EOController)

EOAction.Enabling (Inherited from EOController)

canPerformActionNamed

EOKeyValueCoding (Inherited from EOKeyValueCodingAdditions)

NSKeyValueCoding (Inherited from EOKeyValueCoding)

Method Types

Constructors

EODocumentController

Inserting, updating, and deleting

insertObject

deleteSelectedObjects

wasEdited

Saving

canSave

saveChanges

saveAndMakeInvisible

saveIfUserConfirms

saveIfUserConfirmsAndMakeInvisible

saveIfUserConfirmsAndMakeInvisible

saveFailed

Reverting

canRevert

revert

revertAndMakeInvisible

revertChanges

revertFailed

Undoing and Redoing

canUndo

undo

canRedo

redo

Determining the root document controller

isRootDocumentController

Methods inherited from EOEntityController

handleEditingContextNotification

Methods inherited from EOController

connectionWasEstablished

defaultActions

prepareForNewTask

Methods inherited from Object

toString

Constructors

EODocumentController

```
public EODocumentController()

public EODocumentController(EOXMLUnarchiver unarchiver)
```

Creates a new document controller. For information on how these constructors are used and on what they do, see the method description for the `EOController` constructors in the EOController class specification.

Instance Methods

canPerformActionNamed

```
public boolean canPerformActionNamed(String actionName)
```

Conformance to EOAction.Enabling. EODocumentController's implementation uses the methods `canRedo`, `canRevert`, `canSave`, and `canUndo` to determine the enabling state of the corresponding actions.

See Also: `canPerformActionNamed` (EOAction.Enabling)

canRedo

```
public boolean canRedo()
```

Returns `true` if the receiver can redo, `false` otherwise. A document controller can redo as long as its editing context's undo manager can redo and as long as it (or one of its subcontrollers) is editable.

canRevert

```
public boolean canRevert()
```

Returns `true` if the receiver can revert, `false` otherwise. A document controller can revert only if it's been edited and only if it's the root document controller.

canSave

```
public boolean canSave()
```

Returns `true` if the receiver can save, `false` otherwise. A document controller can save only if it's been edited and only if it's the root document controller.

canUndo

```
public boolean canUndo()
```

Returns `true` if the receiver can undo, `false` otherwise. A document controller can undo as long as its editing context's undo manager can undo and as long as it (or one of its subcontrollers) is editable.

connectionWasEstablished

```
protected void connectionWasEstablished()
```

See the method description for `connectionWasEstablished` in the EOController class specification. EODocumentController's implementation additionally updates its editability.

defaultActions

```
protected NSArray defaultActions()
```

Adds actions for handling editing to the default actions defined by the superclass, EOEntityController. More specifically, it adds save and revert actions. However, note that `defaultActions` only adds save and revert if the receiver is the root document controller, if it's editable, and if it's not modal.

deleteSelectedObjects

```
public void deleteSelectedObjects()
```

Deletes the objects selected in the receiver's display group and then sets the receiver's edited state to `true`.

dispose

```
public void dispose()
```

Conformance to NSDisposable. See the method description of `dispose` in the interface specification for NSDisposable.

editability

```
public int editability()
```

Conformance to EOEditable. See the method description of `editability` in the interface specification for EOEditable.

handleEditingContextNotification

```
public void handleEditingContextNotification(NSNotification notification)
```

See the method description for `handleEditingContextNotification` in the EOEntityController class specification. EODocumentController's implementation additionally updates its edited state if the receiver is a root document controller.

insertObject

```
public void insertObject()
```

Creates a new enterprise object, inserts it into the receiver's display group, and sets the receiver's edited status to `true`.

isDocumentForGlobalID

```
public boolean isDocumentForGlobalID(
    com.webobjects.eocontrol.EOGlobalID globalID,
    String entityName)
```

Conformance to EODocument. See the method description of `isDocumentForGlobalID` in the interface specification for EODocument.

isEditable

```
public boolean isEditable()
```

Conformance to EOEditable. See the method description of `isEditable` in the interface specification for EOEditable.

isEdited

```
public boolean isEdited()
```

Conformance to EODocument. See the method description of `isEdited` in the interface specification for EODocument.

isRootDocumentController

```
public boolean isRootDocumentController()
```

Returns `true` if none of the supercontrollers are EODocuments, `false` otherwise.

prepareForNewTask

```
public void prepareForNewTask(boolean flag)
```

See the method description for `prepareForNewTask` in the EOController class specification. EODocumentController's implementation additionally sets its edited state to `false`.

redo

```
public void redo()
```

Tells the receiver's editing context to redo.

revert

```
public boolean revert()
```

Reverts the receiver's unsaved changes upon user confirmation. If the receiver has been edited, opens a dialog to verify that the user wants to revert. Upon confirmation, invokes `revertChanges` requesting an error dialog upon failure. Returns `true` on success, `false` upon failure or if the user cancels the revert.

revertAndMakeInvisible

```
public boolean revertAndMakeInvisible()
```

Reverts the receiver's unsaved changes and makes the receiver invisible. Reverts by invoking `revertChanges`, requesting an error dialog upon failure Returns `true` if changes are successfully reverted, `false` if the receiver can't be reverted or if the revert fails.

revertChanges

```
public boolean revertChanges(boolean showErrorDialog)
```

Tells the receiver's editing context to revert, refetches if necessary, and sets the receiver's editing state to `false`. If the revert fails, catches the exception and, if `showErrorDialog` is true, invokes `revertFailed` to show the reason for failure. Returns `true` if the revert succeeds, `false` otherwise.

revertFailed

```
protected void revertFailed(
    Exception exception,
    boolean showErrorDialog)
```

If `showErrorDialog` is `true`, brings the receiver's user interface to the front and opens a dialog displaying `exception`'s class name and exception message. Invoked from `revertChanges`.

save

```
public boolean save()
```

Saves the receiver's changes. Saves by invoking `saveChanges`, requesting an error dialog upon failure Returns `true` if changes are successfully saved, `false` if the receiver can't save or if the save fails.

saveAndMakeInvisible

```
public boolean saveAndMakeInvisible()
```

Saves the receiver's changes and makes the receiver invisible. Saves by invoking `saveChanges`, requesting an error dialog upon failure Returns `true` if changes are successfully reverted, `false` if the receiver can't be reverted or if the revert fails.

saveChanges

```
public boolean saveChanges(
    boolean showErrorDialog,
    String saveOperationTitle)
```

Tells the receiver's editing context to save changes and sets the receiver's editing state to `false`. If the save fails, catches the exception and, if `showErrorDialog` is true, invokes `saveFailed` to show the reason for failure. Returns `true` if the save succeeds, `false` otherwise.

saveFailed

```
protected void saveFailed(
    Exception showErrorDialog,
    boolean showErrorDialog,
    String saveOperationTitle)
```

If `showErrorDialog` is `true`, brings the receiver's user interface to the front and opens a dialog displaying `exception`'s class name and exception message. Invoked from `saveChanges`.

saveIfUserConfirms

```
public boolean saveIfUserConfirms(
    String operationTitle,
    String message)

public boolean saveIfUserConfirms()
```

Saves the receiver's unsaved changes upon user confirmation. If the receiver has been edited, opens a dialog to verify that the user wants to save. If `operationTitle` and `message` are provided, they are used as the dialog title and message; otherwise, "Save" and "Save Changes?" are used. Upon confirmation, invokes `saveChanges` requesting an error dialog upon failure. Returns `true` on success, `false` upon failure or if the user cancels the save.

saveIfUserConfirmsAndMakeInvisible

```
public boolean saveIfUserConfirmsAndMakeInvisible(
    String operationTitle,
    String message)

public boolean saveIfUserConfirmsAndMakeInvisible()
```

Saves the receiver's unsaved changes upon user confirmation and makes the receiver invisible. Saves by invoking `saveIfUserConfirms`, requesting an error dialog upon failure. The arguments `operationTitle` and `message` are used as the title and message of the confirmation panel. "Save" and "Save changes?" are substituted for `null`. If the no-argument form of this method is invoked, then the title of the confirmation dialog is "Close" and the dialog has no message. Returns `true` if changes are successfully saved, `false` if the receiver can't be saved or if the save fails.

setEditability

```
public void setEditability(int editability)
```

Conformance to EOEditable. See the method description of `setEditability` in the interface specification for EOEditable.

setEdited

```
public void setEdited(boolean flag)
```

Conformance to EODocument. See the method description of `setEdited` in the interface specification for EODocument.

supercontrollerEditabilityDidChange

```
public void supercontrollerEditabilityDidChange()
```

Conformance to EOEditable. See the method description of `supercontrollerEditabilityDidChange` in the interface specification for EOEditable. EODocumentController's implementation updates the receiver's editability and resets its actions.

takeResponsibilityForEditabilityOfAssociation

```
public void takeResponsibilityForEditabilityOfAssociation(
    com.webobjects.eointerface.EOAssociation association)
```

Conformance to EOEditable. See the method description of `takeResponsibilityForEditabilityOfAssociation` in the interface specification for EOEditable.

toString

```
public String toString()
```

Returns the receiver as a string, including the receiver's editability and whether or not it has unsaved edits.

undo

```
public void undo()
```

Tells the receiver's editing context to redo.

wasEdited

```
protected void wasEdited()
```

Invoked from `setEdited` to notify the receiver that edited status has changed, giving the receiver the opportunity to respond.

EOEntityController

Inherits from:	EOComponentController : EOController : Object
Implements:	EOObserving EOObjectDisplay EOAssociationConnector EOComponentController.EndEditing
Package:	com.webobjects.eoapplication

Class Description

The EOEntityController class provides behavior for displaying enterprise objects in a user interface that can optionally be loaded from an archive (a nib file). EOEntityController's most basic API is specified by the interface EOObjectDisplay, which identifies an implementation strategy that uses EOEditingContexts and EODisplayGroups to manage an entity controller's enterprise objects. An entity controller has an entity name, which identifies the kind of enterprise objects the controller works with. Additionally it has an editing context that manages the controller's enterprise objects, a display group that displays the enterprise objects and manages a selection, and a controller display group that connects controller methods to the user interface. For more information, see the EOObjectDisplay interface specification.

User Interface Archive

As a subclass of EOComponentController, EOEntityController manages a user interface component. However, whereas component controllers dynamically generate their components, entity controllers have the ability to load their components from an archive. An entity controller

has an archive name, which specifies the archive from which to load the controller's component. If, however, a controller doesn't have an archive name, the controller can fall back on dynamically generating its component (an empty EOView).

Managing the Editing Context

As mentioned earlier, EOEntityController uses an editing context to manage its enterprise objects. By default, an entity controller attempts to get its editing context from a supercontroller. An entity controller looks up the controller hierarchy for the first EOObjectDisplay ancestor that has an editing context. If it finds one, the entity controller uses that supercontroller's editing context. If it doesn't find one, it creates one.

You can change the way an entity controller gets its editing context by specifying a **provider method** with `setEditingContextProviderMethodName`. If an entity controller has an editing context provider method, it gets its editing context by invoking that method.

The provider method name is a string, which can be a key path or the name of an arbitrary class's static method. For an example of setting the method name to a key path, consider a subclass of EOEntityController that implements the method `customizedEditingContext` to return an editing context for the controller to use. In this case, the provider method name could be set to "customizedEditingContext".

If the provider method name is the name of a static method, the format of the string is "<class name>:<static method name>". For example, suppose that you've written a subclass of EOApplication that implements a static method, `customizedEditingContextForAllControllers`, to return an editing context for all an application's controllers to share. Then the editing context provider method name for all entity controllers could be set to "CustomApplicationClass: customizedEditingContextForAllControllers".

EOEntityController provides two methods that you can use as provider methods: `newEditingContext` and `nestedEditingContext`. The former simply creates a new editing context and is a convenience for setting the provider method. The latter attempts to create a new editing context that's nested inside an ancestor's editing context. If no ancestors provide an editing context to be a parent, `nestedEditingContext` simply creates a new editing context.

Managing the Display Group

EOEntityController uses a display group to display its enterprise objects. By default, an entity controller attempts to get its display group from a supercontroller. An entity controller looks up the controller hierarchy for the first EOObjectDisplay ancestor. If that supercontroller has the

same entity name and a display group, the entity controller uses that supercontroller's display group. If it doesn't find one, it invokes `loadArchive` to see if a display group is provided in the archive. If the controller still doesn't have a display group, it simply creates one.

You can change the way an entity controller gets its display group by specifying a **provider method** with `setDisplayGroupProviderMethodName`. If an entity controller has a display group provider method, it gets its display group by invoking that method. The display group provider method name works the same way the editing context provider method name works. For more information, see "Managing the Editing Context" (page 182).

EOEntityController provides two methods that you can use as provider methods: `newDisplayGroup` and `newDisplayGroupUsingOptimisticRefresh`. The simply create new display groups and are convenience methods for setting the provider method.

Rule System and XML Description

The following tables identify the `controllerType`, XML tag, and XML attributes used by the rule system and EOXMLUnarchiver to generate a controller hierarchy. For more information, see the section "Rule System and XML Description" (page 8) in the package introduction.

Default Rule System Controller Type

`entityController`

XML Tag

`ENTITYCONTROLLER`

XML Attribute	Value	Description
`archive`	string	The name of a nib file from which the controller loads its component (instead of dynamically creating it).

XML Attribute	Value	Description
displayGroupProviderMethodName	string	A key path or string of the form "<class name>:<method name>" that names a method the controller uses to create its display group.
editingContextProviderMethodName	string	A key path or string of the form "<class name>:<method name>" that names a method the controller uses to create its editing context.
entity	string	Name of the controller's entity.

Constants

Constant	Description
NestedEditingContextProviderMethodName	Description forthcoming.
NewDisplayGroupProviderMethodName	Description forthcoming.
NewEditingContextProviderMethodName	Description forthcoming.

Interfaces Implemented

EOObserving

- objectWillChange

EOObjectDisplay

- controllerDisplayGroup
- displayGroup
- editingContext
- entityName

EOAssociationConnector

- takeResposibilityForConnectionOfAssociation

EOComponentController.EndEditing

- endEditing

NSDisposable (Inherited from EOController)

- dispose

EOKeyValueCodingAdditions (Inherited from EOController)

EOAction.Enabling (Inherited from EOController)

EOKeyValueCoding (Inherited from EOKeyValueCodingAdditions)

NSKeyValueCoding (Inherited from EOKeyValueCoding)

Method Types

Constructors

EOEntityController

Setting the entity

setEntityName

Loading an archive

prepareComponent

loadArchive

controllerDidLoadArchive

objectForOutletPath

setArchiveName

archiveName

Managing the editing context

newEditingContext

setEditingContext

setEditingContextProviderMethodName

editingContextProviderMethodName

nestedEditingContext

startListeningToEditingContext

stopListeningToEditingContext

handleEditingContextNotification

setResetsEditingContextWhenPreparingForNewTask

resetsEditingContextWhenPreparingForNewTask

Managing the controller display group

setControllerDisplayGroup

hasControllerDisplayGroup

Managing the objects display group

newDataSource

newDisplayGroup

newDisplayGroupUsingOptimisticRefresh

setDisplayGroup

startListeningToDisplayGroup

stopListeningToDisplayGroup

setObjectWithGlobalID

setObjectsWithFetchSpecification

displayGroupSortOrderings

setDisplayGroupProviderMethodName

displayGroupProviderMethodName

Accessing selected objects

selectedObject

selectedObjectGlobalID

selectedObjects

selectedObjectsGlobalIDs

Fetching

fetchesOnConnect

setFetchesOnConnect

Determining the root document controller

isRootEntityController

Notifying observers of change

willChange

Methods inherited from EOController

connectionWasBroken

connectionWasEstablished

establishConnection

prepareForNewTask

Methods inherited from Object

toString

Constructors

EOEntityController

```
public EOEntityController(EOXMLUnarchiver unarchiver)
```

Description forthcoming.

```
public EOEntityController()
```

Description forthcoming.

Instance Methods

archiveName

```
public String archiveName()
```

Description forthcoming.

connectionWasBroken

```
protected void connectionWasBroken()
```

Description forthcoming.

connectionWasEstablished

```
protected void connectionWasEstablished()
```

Description forthcoming.

controllerDidLoadArchive

```
protected void controllerDidLoadArchive(NSDictionary aNSDictionary)
```

Description forthcoming.

controllerDisplayGroup

```
public com.webobjects.eointerface.EODisplayGroup controllerDisplayGroup()
```

Description forthcoming.

displayGroup

```
public com.webobjects.eointerface.EODisplayGroup displayGroup()
```

Description forthcoming.

displayGroupProviderMethodName

```
public String displayGroupProviderMethodName()
```

Description forthcoming.

displayGroupSortOrderings

```
protected NSArray displayGroupSortOrderings()
```

Description forthcoming.

dispose

```
public void dispose()
```

Description forthcoming.

editingContext

```
public com.webobjects.eocontrol.EOEditingContext editingContext()
```

Description forthcoming.

editingContextProviderMethodName

```
public String editingContextProviderMethodName()
```

Description forthcoming.

endEditing

```
public boolean endEditing()
```

Description forthcoming.

entityName

```
public String entityName()
```

Description forthcoming.

establishConnection

```
public void establishConnection()
```

Description forthcoming.

fetchesOnConnect

```
public boolean fetchesOnConnect()
```

Description forthcoming.

handleEditingContextNotification

```
public void handleEditingContextNotification(NSNotification aNSNotification)
```

Description forthcoming.

hasControllerDisplayGroup

```
public boolean hasControllerDisplayGroup()
```

Description forthcoming.

isRootEntityController

```
public boolean isRootEntityController()
```

Description forthcoming.

loadArchive

```
protected boolean loadArchive()
```

Description forthcoming.

nestedEditingContext

```
public com.webobjects.eocontrol.EOEditingContext nestedEditingContext()
```

Description forthcoming.

newDataSource

```
protected com.webobjects.eocontrol.EODataSource newDataSource()
```

Description forthcoming.

newDisplayGroup

```
public com.webobjects.eointerface.EODisplayGroup newDisplayGroup()
```

Description forthcoming.

newDisplayGroupUsingOptimisticRefresh

```
public com.webobjects.eointerface.EODisplayGroup newDisplayGroupUsingOptimisticRefresh()
```

Description forthcoming.

newEditingContext

```
public com.webobjects.eocontrol.EOEditingContext newEditingContext()
```

Description forthcoming.

objectForOutletPath

```
public Object objectForOutletPath(
    EOArchive anEOArchive,
    String aString)
```

Description forthcoming.

objectWillChange

```
public void objectWillChange(Object anObject)
```

Description forthcoming.

prepareComponent

```
protected void prepareComponent()
```

Description forthcoming.

prepareForNewTask

```
public void prepareForNewTask(boolean aBoolean)
```

Description forthcoming.

resetsEditingContextWhenPreparingForNewTask

```
public boolean resetsEditingContextWhenPreparingForNewTask()
```

Description forthcoming.

selectedObject

```
public com.webobjects.eocontrol.EOEnterpriseObject selectedObject()
```

Description forthcoming.

selectedObjectGlobalID

```
public com.webobjects.eocontrol.EOGlobalID selectedObjectGlobalID()
```

Description forthcoming.

selectedObjects

```
public NSArray selectedObjects()
```

Description forthcoming.

selectedObjectsGlobalIDs

```
public NSArray selectedObjectsGlobalIDs()
```

Description forthcoming.

setArchiveName

```
public void setArchiveName(String aString)
```

Description forthcoming.

setControllerDisplayGroup

```
public void setControllerDisplayGroup(com.webobjects.eointerface.EODisplayGroup anEODisplayGroup)
```

Description forthcoming.

setDisplayGroup

```
public void setDisplayGroup(com.webobjects.eointerface.EODisplayGroup anEODisplayGroup)
```

Description forthcoming.

setDisplayGroupProviderMethodName

```
public void setDisplayGroupProviderMethodName(String aString)
```

Description forthcoming.

setEditingContext

```
public void setEditingContext(com.webobjects.eocontrol.EOEditingContext anEOEditingContext)
```

Description forthcoming.

setEditingContextProviderMethodName

```
public void setEditingContextProviderMethodName(String aString)
```

Description forthcoming.

setEntityName

```
public void setEntityName(String aString)
```

Description forthcoming.

setFetchesOnConnect

```
public void setFetchesOnConnect(boolean aBoolean)
```

Description forthcoming.

setObjectWithGlobalID

```
public void setObjectWithGlobalID(com.webobjects.eocontrol.EOGlobalID anEOGlobalID)
```

Description forthcoming.

setObjectsWithFetchSpecification

```
public void
    setObjectsWithFetchSpecification(com.webobjects.eocontrol.EOFetchSpecification anEOFetchSpecification
    )
```

Description forthcoming.

setResetsEditingContextWhenPreparingForNewTask

```
public void setResetsEditingContextWhenPreparingForNewTask(boolean aBoolean)
```

Description forthcoming.

startListeningToDisplayGroup

```
protected void startListeningToDisplayGroup()
```

Description forthcoming.

startListeningToEditingContext

```
protected void startListeningToEditingContext()
```

Description forthcoming.

stopListeningToDisplayGroup

```
protected void stopListeningToDisplayGroup()
```

Description forthcoming.

stopListeningToEditingContext

```
protected void stopListeningToEditingContext()
```

Description forthcoming.

takeResposibilityForConnectionOfAssociation

```
public void
    takeResposibilityForConnectionOfAssociation(com.webobjects.eointerface.EOAssociation anEOAssociation)
```

Description forthcoming.

toString

```
public String toString()
```

Description forthcoming.

willChange

```
public void willChange()
```

Description forthcoming.

EOFrameController

Inherits from: EOSimpleWindowController :
EOWindowController :
EOComponentController :
EOController :
Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
FRAMECONTROLLER	windowController

Method Types

All methods

EOFrameController

runControllerInNewFrame

dispose

generateBorderSize

newWindow

setWindowResizable

setWindowTitle

verifyContentMinimumSize

Constructors

EOFrameController

```
public EOFrameController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOFrameController()
```

Description forthcoming.

Static Methods

runControllerInNewFrame

```
public static void runControllerInNewFrame(
    EOComponentController anEOComponentController,
    String aString)
```

Description forthcoming.

Instance Methods

dispose

```
public void dispose()
```

Description forthcoming.

generateBorderSize

```
protected java.awt.Dimension generateBorderSize()
```

Description forthcoming.

newWindow

```
protected java.awt.Window newWindow(javax.swing.JComponent aJComponent)
```

Description forthcoming.

setWindowResizable

```
protected void setWindowResizable(
    java.awt.Window aWindow,
    boolean aBoolean)
```

Description forthcoming.

setWindowTitle

```
protected void setWindowTitle(
    java.awt.Window aWindow,
    String aString)
```

Description forthcoming.

verifyContentMinimumSize

```
protected java.awt.Dimension verifyContentMinimumSize(
    java.awt.Window aWindow,
    java.awt.Dimension aDimension)
```

Description forthcoming.

EOInspectorController

Inherits from: EOWindowController :
EOComponentController :
EOController :
Object

Implements: EOComponentController.ResetUserInterface

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
INSPECTORCONTROLLER	windowController

Method Types

All methods

EOInspectorController

activateWindow

addComponentOfSubcontroller

dispose

generateBorderSize

generateComponent

inspectorIdentifier

integrationComponentDidBecomeInvisible

integrationComponentDidBecomeVisible

resetUserInterface

setInspectorIdentifier

subcontrollerMinimumSizeDidChange

Constructors

EOInspectorController

```
public EOInspectorController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOInspectorController()
```

Description forthcoming.

Instance Methods

activateWindow

```
public void activateWindow()
```

Description forthcoming.

addComponentOfSubcontroller

```
protected void addComponentOfSubcontroller(EOComponentController anEOComponentController)
```

Description forthcoming.

dispose

```
public void dispose()
```

Description forthcoming.

generateBorderSize

```
protected java.awt.Dimension generateBorderSize()
```

Description forthcoming.

generateComponent

```
protected void generateComponent()
```

Description forthcoming.

inspectorIdentifier

```
public String inspectorIdentifier()
```

Description forthcoming.

integrationComponentDidBecomeInvisible

```
protected void integrationComponentDidBecomeInvisible()
```

Description forthcoming.

integrationComponentDidBecomeVisible

```
protected void integrationComponentDidBecomeVisible()
```

Description forthcoming.

resetUserInterface

```
public void resetUserInterface()
```

Description forthcoming.

setInspectorIdentifier

```
public void setInspectorIdentifier(String aString)
```

Description forthcoming.

subcontrollerMinimumSizeDidChange

```
public void subcontrollerMinimumSizeDidChange(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    java.awt.Dimension aDimension)
```

Description forthcoming.

EOInterfaceController

Inherits from: EODocumentController :
EOEntityController :
EOComponentController :
EOController :
Object

Package: com.webobjects.eoapplication

Class Description

EOInterfaceController serves as a convenient base class for logic related to the interface of client-side applications. When the WebObjectsApplication wizard in Project Builder creates a new client-side interface, it adds (to the client-side subproject) an Interface Builder nib file representing this interface and a skeletal EOInterfaceController subclass defined as the nib file's root object or "owner."

In an application constructed in conformance to the Model-View-Controller paradigm, EOInterfaceController plays the role of controller. It has four special outlets (defined in the EOEntityController superclass): its `editingContext`, its `component`, its `displayGroup`, and its `controllerDisplayGroup`, all of which you can configure using Interface Builder. The object identified by component is an AWT JComponent that functions as the view, since it is the main entry point into the user interface. Because an enterprise object must always inhabit an editing context, `editingContext` and its contents serve as the "model." The `displayGroup` is an EODisplayGroup

containing the enterprise objects manipulated by the controller's user interface (which may will involve other display groups). The controllerDisplayGroup is a convenience instance containing nothing but the interface controller itself.

XML Tag	Default Rule System Controller Type
INTERFACECONTROLLER	entityController

Method Types

All methods

EOInterfaceController

archiveName

collectChangesFromServer

generateComponent

Constructors

EOInterfaceController

```
public EOInterfaceController()

public EOInterfaceController(com.webobjects.eocontrol.EOEditingContext editingContext)
```

Initializes a new instance then attempts to load the nib file matching the class name. The one-argument and two-argument constructors allow you to specify the editing context used as the nib file's substitution editing context during the load.

```
public EOInterfaceController(
    com.webobjects.eocontrol.EOEditingContext editingContext,
    String archiveName)
```

Initializes a new instance then attempts to load the associated nib file identified by `archiveName`. The `editingContext` argument is used as the nib file's substitution editing context during the load.

```
public EOInterfaceController(EOXMLUnarchiver unarchiver)
```

Initializes a new instance with the contents of the `unarchiver` EOXMLUnarchiver.

Instance Methods

archiveName

```
public String archiveName()
```

Returns the name of the nib file that specifies the receiver's user interface. Defaults to the receiver's class name.

collectChangesFromServer

```
public void collectChangesFromServer()
```

Updates the receiver's editing context to reflect any changes to enterprise objects pending on the server.

generateComponent

```
protected void generateComponent()
```

Since an EOInterfaceController requires a nib file, this method is overridden to raise a NSInternalInconsistencyException.

EOMenuSwitchController

Inherits from: EOSwitchController :
EOComponentController :
EOController :
Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
MENUSWITCHCONTROLLER	dividingController

Method Types

All methods

EOMenuSwitchController

addBorderComponentForControllerToDisplayComponent

borderSize

dispose

newDisplayComponent

removeBorderComponentForControllerFromDisplayComponent

selectedBorderComponentInDisplayComponent

showBorderComponentAtIndexInDisplayComponent

Constructors

EOMenuSwitchController

```
public EOMenuSwitchController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOMenuSwitchController()
```

Description forthcoming.

Instance Methods

addBorderComponentForControllerToDisplayComponent

```
protected void addBorderComponentForControllerToDisplayComponent(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent)
```

Description forthcoming.

borderSize

```
public java.awt.Dimension borderSize()
```

Description forthcoming.

dispose

```
public void dispose()
```

Description forthcoming.

newDisplayComponent

```
protected javax.swing.JComponent newDisplayComponent()
```

Description forthcoming.

removeBorderComponentForControllerFromDisplayComponent

```
protected void removeBorderComponentForControllerFromDisplayComponent(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

selectedBorderComponentInDisplayComponent

```
protected javax.swing.JComponent
    selectedBorderComponentInDisplayComponent(javax.swing.JComponent aJComponent)
```

Description forthcoming.

showBorderComponentAtIndexInDisplayComponent

```
protected void showBorderComponentAtIndexInDisplayComponent(
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

EOModalDialogController

Inherits from: EODialogController :
EOWindowController :
EOComponentController :
EOController :
Object

Implements: EOComponentController.Modal

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
MODALDIALOGCONTROLLER	modalDialogController

Method Types

All methods

EOModalDialogController

runControllerInNewModalDialog

activateWindow

closeWindow

isModal

newWindow

Constructors

EOModalDialogController

```
public EOModalDialogController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOModalDialogController()
```

Description forthcoming.

Static Methods

runControllerInNewModalDialog

```
public static void runControllerInNewModalDialog(
    EOComponentController anEOComponentController,
    String aString)
```

Description forthcoming.

Instance Methods

activateWindow

```
public void activateWindow()
```

Description forthcoming.

closeWindow

```
public boolean closeWindow()
```

Description forthcoming.

isModal

```
public boolean isModal()
```

Description forthcoming.

newWindow

```
protected java.awt.Window newWindow(javax.swing.JComponent aJComponent)
```

Description forthcoming.

EOProgrammaticSwitchController

Inherits from: EOSwitchController :
EOComponentController :
EOController :
Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
PROGRAMMATICSWITCHCONTROLLER	dividingController

Method Types

All methods

EOProgrammaticSwitchController

addBorderComponentForControllerToDisplayComponent

newDisplayComponent

removeBorderComponentForControllerFromDisplayComponent

selectedBorderComponentInDisplayComponent

showBorderComponentAtIndexInDisplayComponent

Constructors

EOProgrammaticSwitchController

```
public EOProgrammaticSwitchController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

Instance Methods

addBorderComponentForControllerToDisplayComponent

```
protected void addBorderComponentForControllerToDisplayComponent(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent)
```

Description forthcoming.

newDisplayComponent

```
protected javax.swing.JComponent newDisplayComponent()
```

Description forthcoming.

removeBorderComponentForControllerFromDisplayComponent

```
protected void removeBorderComponentForControllerFromDisplayComponent(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

selectedBorderComponentInDisplayComponent

```
protected javax.swing.JComponent
    selectedBorderComponentInDisplayComponent(javax.swing.JComponent aJComponent)
```

Description forthcoming.

showBorderComponentAtIndexInDisplayComponent

```
protected void showBorderComponentAtIndexInDisplayComponent(
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

EOSimpleWindowController

Inherits from:	EOWindowController : EOComponentController : EOController : Object
Implements:	WindowListener (java.awt.event package) ComponentListener (java.awt.event package) EOComponentController.ResetUserInterface
Package:	com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
None (abstract class)	`windowController`

Method Types

All methods

EOSimpleWindowController

activateWindow

addComponentOfSubcontroller

closeWindow

componentDidBecomeInvisible

componentDidBecomeVisible

componentHidden

componentMoved

componentResized

componentShown

deactivateWindow

dispose

disposeIfDeactivated

integrationComponentDidBecomeInvisible

integrationComponentDidBecomeVisible

makeVisible

newWindow

newWindow

resetUserInterface

setDisposeIfDeactivated

setLabel

setWindow

setWindowResizable

setWindowTitle

startListeningToWindow

stopListeningToWindow

subcontrollerEditedDidChange

subcontrollerMinimumSizeDidChange

verifyContentMinimumSize

window

windowActivated

windowClosed

windowClosing

windowDeactivated

windowDeiconified

windowIconified

windowOpened

Constructors

EOSimpleWindowController

```
public EOSimpleWindowController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOSimpleWindowController()
```

Description forthcoming.

Instance Methods

activateWindow

```
public void activateWindow()
```

Description forthcoming.

addComponentOfSubcontroller

```
protected void addComponentOfSubcontroller(EOComponentController anEOComponentController)
```

Description forthcoming.

closeWindow

```
public boolean closeWindow()
```

Description forthcoming.

componentDidBecomeInvisible

```
protected void componentDidBecomeInvisible()
```

Description forthcoming.

componentDidBecomeVisible

```
protected void componentDidBecomeVisible()
```

Description forthcoming.

componentHidden

```
public void componentHidden(java.awt.event.ComponentEvent aComponentEvent)
```

Description forthcoming.

componentMoved

```
public void componentMoved(java.awt.event.ComponentEvent aComponentEvent)
```

Description forthcoming.

componentResized

```
public void componentResized(java.awt.event.ComponentEvent aComponentEvent)
```

Description forthcoming.

componentShown

```
public void componentShown(java.awt.event.ComponentEvent aComponentEvent)
```

Description forthcoming.

deactivateWindow

```
public void deactivateWindow()
```

Description forthcoming.

dispose

```
public void dispose()
```

Description forthcoming.

disposeIfDeactivated

```
public boolean disposeIfDeactivated()
```

Description forthcoming.

integrationComponentDidBecomeInvisible

```
protected void integrationComponentDidBecomeInvisible()
```

Description forthcoming.

integrationComponentDidBecomeVisible

```
protected void integrationComponentDidBecomeVisible()
```

Description forthcoming.

makeVisible

```
public boolean makeVisible()
```

Description forthcoming.

newWindow

```
protected abstract java.awt.Window newWindow(javax.swing.JComponent aJComponent)
```

Description forthcoming.

newWindow

```
protected java.awt.Window newWindow()
```

Description forthcoming.

resetUserInterface

```
public void resetUserInterface()
```

Description forthcoming.

setDisposeIfDeactivated

```
public void setDisposeIfDeactivated(boolean aBoolean)
```

Description forthcoming.

setLabel

```
public void setLabel(String aString)
```

Description forthcoming.

setWindow

```
public void setWindow(java.awt.Window aWindow)
```

Description forthcoming.

setWindowResizable

```
protected abstract void setWindowResizable(
    java.awt.Window aWindow,
    boolean aBoolean)
```

Description forthcoming.

setWindowTitle

```
protected abstract void setWindowTitle(
    java.awt.Window aWindow,
    String aString)
```

Description forthcoming.

startListeningToWindow

```
protected void startListeningToWindow()
```

Description forthcoming.

stopListeningToWindow

```
protected void stopListeningToWindow()
```

Description forthcoming.

subcontrollerEditedDidChange

```
public void subcontrollerEditedDidChange(EOController anEOController)
```

Description forthcoming.

subcontrollerMinimumSizeDidChange

```
public void subcontrollerMinimumSizeDidChange(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    java.awt.Dimension aDimension)
```

Description forthcoming.

verifyContentMinimumSize

```
protected java.awt.Dimension verifyContentMinimumSize(
    java.awt.Window aWindow,
    java.awt.Dimension aDimension)
```

Description forthcoming.

window

```
public java.awt.Window window()
```

Description forthcoming.

windowActivated

```
public void windowActivated(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowClosed

```
public void windowClosed(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowClosing

```
public void windowClosing(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowDeactivated

```
public void windowDeactivated(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowDeiconified

```
public void windowDeiconified(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowIconified

```
public void windowIconified(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowOpened

```
public void windowOpened(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

EOSwitchController

Inherits from:	EOComponentController : EOController : Object
Implements:	EOComponentController.ResetUserInterface
Package:	com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
None (abstract class)	None

Method Types

All methods

`EOSwitchController`

`addBorderComponentForControllerToDisplayComponent`

`addComponentOfSubcontroller`

borderComponents

borderSize

borderedSizeForComponentSize

componentDidBecomeInvisible

componentDidBecomeVisible

componentShouldChange

componentSizeForBorderedSize

componentSwitched

dispose

generateComponent

hideSubcontroller

inset

minimumComponentSize

newDisplayComponent

removeBorderComponentForControllerFromDisplayComponent

removeComponentOfSubcontroller

removeTransientSubcontroller

resetUserInterface

selectedBorderComponentInDisplayComponent

showBorderComponentAtIndex

showBorderComponentAtIndexInDisplayComponent

showSubcontroller

subcontrollerMinimumSizeDidChange

subcontrollerVisibilityDidChange

switchedControllers

visibleBorderComponentIndex

Constructors

EOSwitchController

```
public EOSwitchController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOSwitchController()
```

Description forthcoming.

Instance Methods

addBorderComponentForControllerToDisplayComponent

```
protected abstract void addBorderComponentForControllerToDisplayComponent(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent)
```

Description forthcoming.

addComponentOfSubcontroller

```
protected void addComponentOfSubcontroller(EOComponentController anEOComponentController)
```

Description forthcoming.

borderComponents

```
protected NSArray borderComponents()
```

Description forthcoming.

borderSize

```
public java.awt.Dimension borderSize()
```

Description forthcoming.

borderedSizeForComponentSize

```
public java.awt.Dimension borderedSizeForComponentSize(java.awt.Dimension aDimension)
```

Description forthcoming.

componentDidBecomeInvisible

```
protected void componentDidBecomeInvisible()
```

Description forthcoming.

componentDidBecomeVisible

```
protected void componentDidBecomeVisible()
```

Description forthcoming.

componentShouldChange

```
public boolean componentShouldChange(int anInt)
```

Description forthcoming.

componentSizeForBorderedSize

```
public java.awt.Dimension componentSizeForBorderedSize(java.awt.Dimension aDimension)
```

Description forthcoming.

componentSwitched

```
public void componentSwitched(int anInt)
```

Description forthcoming.

dispose

```
public void dispose()
```

Description forthcoming.

generateComponent

```
protected void generateComponent()
```

Description forthcoming.

hideSubcontroller

```
protected boolean hideSubcontroller(EOComponentController anEOComponentController)
```

Description forthcoming.

inset

```
protected int inset()
```

Description forthcoming.

minimumComponentSize

```
public java.awt.Dimension minimumComponentSize()
```

Description forthcoming.

newDisplayComponent

```
protected abstract javax.swing.JComponent newDisplayComponent()
```

Description forthcoming.

removeBorderComponentForControllerFromDisplayComponent

```
protected abstract void removeBorderComponentForControllerFromDisplayComponent(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

removeComponentOfSubcontroller

```
protected void removeComponentOfSubcontroller(EOComponentController anEOComponentController)
```

Description forthcoming.

removeTransientSubcontroller

```
protected boolean removeTransientSubcontroller(EOController anEOController)
```

Description forthcoming.

resetUserInterface

```
public void resetUserInterface()
```

Description forthcoming.

selectedBorderComponentInDisplayComponent

```
protected abstract javax.swing.JComponent
    selectedBorderComponentInDisplayComponent(javax.swing.JComponent aJComponent)
```

Description forthcoming.

showBorderComponentAtIndex

```
protected void showBorderComponentAtIndex(int anInt)
```

Description forthcoming.

showBorderComponentAtIndexInDisplayComponent

```
protected abstract void showBorderComponentAtIndexInDisplayComponent(
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

showSubcontroller

```
protected boolean showSubcontroller(EOComponentController anEOComponentController)
```

Description forthcoming.

subcontrollerMinimumSizeDidChange

```
public void subcontrollerMinimumSizeDidChange(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    java.awt.Dimension aDimension)
```

Description forthcoming.

subcontrollerVisibilityDidChange

```
public void subcontrollerVisibilityDidChange(EOComponentController anEOComponentController)
```

Description forthcoming.

switchedControllers

```
protected NSArray switchedControllers()
```

Description forthcoming.

visibleBorderComponentIndex

```
protected int visibleBorderComponentIndex()
```

Description forthcoming.

EOTabSwitchController

Inherits from: EOSwitchController :
EOComponentController :
EOController :
Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
TABSWITCHCONTROLLER	dividingController

Method Types

All methods

EOTabSwitchController

addBorderComponentForControllerToDisplayComponent

borderSize

inset

newDisplayComponent

removeBorderComponentForControllerFromDisplayComponent

selectedBorderComponentInDisplayComponent

showBorderComponentAtIndexInDisplayComponent

Constructors

EOTabSwitchController

```
public EOTabSwitchController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOSTabSwitchController()
```

Description forthcoming.

Instance Methods

addBorderComponentForControllerToDisplayComponent

```
protected void addBorderComponentForControllerToDisplayComponent(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent)
```

Description forthcoming.

borderSize

```
public java.awt.Dimension borderSize()
```

Description forthcoming.

inset

```
protected int inset()
```

Description forthcoming.

newDisplayComponent

```
protected javax.swing.JComponent newDisplayComponent()
```

Description forthcoming.

removeBorderComponentForControllerFromDisplayComponent

```
protected void removeBorderComponentForControllerFromDisplayComponent(
    EOComponentController anEOComponentController,
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

selectedBorderComponentInDisplayComponent

```
protected javax.swing.JComponent
    selectedBorderComponentInDisplayComponent(javax.swing.JComponent aJComponent)
```

Description forthcoming.

showBorderComponentAtIndexInDisplayComponent

```
protected void showBorderComponentAtIndexInDisplayComponent(
    javax.swing.JComponent aJComponent,
    javax.swing.JComponent aJComponent,
    int anInt)
```

Description forthcoming.

EOUserInterfaceParameters

Inherits from: Object

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

`actionTitleFont`

`actionTitlePosition`

`allowsActionIcons`

`allowsIcons`

`allowsSmallActionIcons`

`disabledTextBackgroundColor`

editableTextBackgroundColor

highlightLabelColor

highlightLabelFont

highlightTitleColor

highlightTitleFont

labelColor

labelFont

largeBorder

localizedIcon

localizedString

makeIconBackgroundsTransparent

mediumBorder

minimumActionButtonSize

minimumSmallActionButtonSize

minimumSpecialActionButtonSize

optimizesMenuAccelerators

queryTextBackgroundColor

setActionTitleFont

setActionTitlePosition

setAllowsActionIcons

setAllowsIcons

setAllowsSmallActionIcons

setBorders

setDisabledTextBackgroundColor

setEditableTextBackgroundColor

setHighlightLabelColor

setHighlightLabelFont

setHighlightTitleColor

setHighlightTitleFont

setLabelColor

setLabelFont

setMakeIconBackgroundsTransparent

setMinimumActionButtonSize

setMinimumSmallActionButtonSize

setMinimumSpecialActionButtonSize

setOptimizesMenuAccelerators

setQueryTextBackgroundColor

setSmallActionTitleFont

setSmallActionTitlePosition

setSpecialActionTitleFont

setSpecialActionTitlePosition

setStandardResourceBundle

setTitleColor

setTitleFont

setUsesBorderWithActionIcons

setUsesBorderWithSmallActionIcons

setUsesSpecialColors

setUsesSpecialFonts

setUsesTitleWithActionIcons

setUsesTitleWithSmallActionIcons

setUsesWindowMenus

setWidgetFont

smallActionTitleFont

smallActionTitlePosition

smallBorder

specialActionTitleFont

specialActionTitlePosition

standardActionIcon

standardResourceBundle

standardSmallActionIcon

titleColor

titleFont

usesBorderWithActionIcons

usesBorderWithSmallActionIcons

usesSpecialColors

usesSpecialFonts

usesTitleWithActionIcons

usesTitleWithSmallActionIcons

usesWindowMenus

widgetFont

Constructors

EOUserInterfaceParameters

```
public EOUserInterfaceParameters()
```

Description forthcoming.

Static Methods

actionTitleFont

```
public static java.awt.Font actionTitleFont()
```

Description forthcoming.

actionTitlePosition

```
public static int actionTitlePosition()
```

Description forthcoming.

allowsActionIcons

```
public static boolean allowsActionIcons()
```

Description forthcoming.

allowsIcons

```
public static boolean allowsIcons()
```

Description forthcoming.

allowsSmallActionIcons

```
public static boolean allowsSmallActionIcons()
```

Description forthcoming.

disabledTextBackgroundColor

```
public static java.awt.Color disabledTextBackgroundColor()
```

Description forthcoming.

editableTextBackgroundColor

```
public static java.awt.Color editableTextBackgroundColor()
```

Description forthcoming.

highlightLabelColor

```
public static java.awt.Color highlightLabelColor()
```

Description forthcoming.

highlightLabelFont

```
public static java.awt.Font highlightLabelFont()
```

Description forthcoming.

highlightTitleColor

```
public static java.awt.Color highlightTitleColor()
```

Description forthcoming.

highlightTitleFont

```
public static java.awt.Font highlightTitleFont()
```

Description forthcoming.

labelColor

```
public static java.awt.Color labelColor()
```

Description forthcoming.

labelFont

```
public static java.awt.Font labelFont()
```

Description forthcoming.

largeBorder

```
public static int largeBorder()
```

Description forthcoming.

localizedIcon

```
public static javax.swing.Icon localizedIcon(String aString)
```

Description forthcoming.

localizedString

```
public static String localizedString(String aString)
```

Description forthcoming.

makeIconBackgroundsTransparent

```
public static boolean makeIconBackgroundsTransparent()
```

Description forthcoming.

mediumBorder

```
public static int mediumBorder()
```

Description forthcoming.

minimumActionButtonSize

```
public static java.awt.Dimension minimumActionButtonSize()
```

Description forthcoming.

minimumSmallActionButtonSize

```
public static java.awt.Dimension minimumSmallActionButtonSize()
```

Description forthcoming.

minimumSpecialActionButtonSize

```
public static java.awt.Dimension minimumSpecialActionButtonSize()
```

Description forthcoming.

optimizesMenuAccelerators

```
public static boolean optimizesMenuAccelerators()
```

Description forthcoming.

queryTextBackgroundColor

```
public static java.awt.Color queryTextBackgroundColor()
```

Description forthcoming.

setActionTitleFont

```
public static void setActionTitleFont(java.awt.Font aFont)
```

Description forthcoming.

setActionTitlePosition

```
public static void setActionTitlePosition(int anInt)
```

Description forthcoming.

setAllowsActionIcons

```
public static void setAllowsActionIcons(boolean aBoolean)
```

Description forthcoming.

setAllowsIcons

```
public static void setAllowsIcons(boolean aBoolean)
```

Description forthcoming.

setAllowsSmallActionIcons

```
public static void setAllowsSmallActionIcons(boolean aBoolean)
```

Description forthcoming.

setBorders

```
public static void setBorders(
    int anInt,
    int anInt,
    int anInt)
```

Description forthcoming.

setDisabledTextBackgroundColor

```
public static void setDisabledTextBackgroundColor(java.awt.Color aColor)
```

Description forthcoming.

setEditableTextBackgroundColor

```
public static void setEditableTextBackgroundColor(java.awt.Color aColor)
```

Description forthcoming.

setHighlightLabelColor

```
public static void setHighlightLabelColor(java.awt.Color aColor)
```

Description forthcoming.

setHighlightLabelFont

```
public static void setHighlightLabelFont(java.awt.Font aFont)
```

Description forthcoming.

setHighlightTitleColor

```
public static void setHighlightTitleColor(java.awt.Color aColor)
```

Description forthcoming.

setHighlightTitleFont

```
public static void setHighlightTitleFont(java.awt.Font aFont)
```

Description forthcoming.

setLabelColor

```
public static void setLabelColor(java.awt.Color aColor)
```

Description forthcoming.

setLabelFont

```
public static void setLabelFont(java.awt.Font aFont)
```

Description forthcoming.

setMakeIconBackgroundsTransparent

```
public static void setMakeIconBackgroundsTransparent(boolean aBoolean)
```

Description forthcoming.

setMinimumActionButtonSize

```
public static void setMinimumActionButtonSize(java.awt.Dimension aDimension)
```

Description forthcoming.

setMinimumSmallActionButtonSize

```
public static void setMinimumSmallActionButtonSize(java.awt.Dimension aDimension)
```

Description forthcoming.

setMinimumSpecialActionButtonSize

```
public static void setMinimumSpecialActionButtonSize(java.awt.Dimension aDimension)
```

Description forthcoming.

setOptimizesMenuAccelerators

```
public static void setOptimizesMenuAccelerators(boolean aBoolean)
```

Description forthcoming.

setQueryTextBackgroundColor

```
public static void setQueryTextBackgroundColor(java.awt.Color aColor)
```

Description forthcoming.

setSmallActionTitleFont

```
public static void setSmallActionTitleFont(java.awt.Font aFont)
```

Description forthcoming.

setSmallActionTitlePosition

```
public static void setSmallActionTitlePosition(int anInt)
```

Description forthcoming.

setSpecialActionTitleFont

```
public static void setSpecialActionTitleFont(java.awt.Font aFont)
```

Description forthcoming.

setSpecialActionTitlePosition

```
public static void setSpecialActionTitlePosition(int anInt)
```

Description forthcoming.

setStandardResourceBundle

```
public static void setStandardResourceBundle(java.util.ResourceBundle aResourceBundle)
```

Description forthcoming.

setTitleColor

```
public static void setTitleColor(java.awt.Color aColor)
```

Description forthcoming.

setTitleFont

```
public static void setTitleFont(java.awt.Font aFont)
```

Description forthcoming.

setUsesBorderWithActionIcons

```
public static void setUsesBorderWithActionIcons(boolean flag)
```

Description forthcoming.

setUsesBorderWithSmallActionIcons

```
public static void setUsesBorderWithSmallActionIcons(boolean flag)
```

Description forthcoming.

setUsesSpecialColors

```
public static void setUsesSpecialColors(boolean aBoolean)
```

Description forthcoming.

setUsesSpecialFonts

```
public static void setUsesSpecialFonts(boolean aBoolean)
```

Description forthcoming.

setUsesTitleWithActionIcons

```
public static void setUsesTitleWithActionIcons(boolean aBoolean)
```

Description forthcoming.

setUsesTitleWithSmallActionIcons

```
public static void setUsesTitleWithSmallActionIcons(boolean aBoolean)
```

Description forthcoming.

setUsesWindowMenus

```
public static void setUsesWindowMenus(boolean aBoolean)
```

Description forthcoming.

setWidgetFont

```
public static void setWidgetFont(java.awt.Font aFont)
```

Description forthcoming.

smallActionTitleFont

```
public static java.awt.Font smallActionTitleFont()
```

Description forthcoming.

smallActionTitlePosition

```
public static int smallActionTitlePosition()
```

Description forthcoming.

smallBorder

```
public static int smallBorder()
```

Description forthcoming.

specialActionTitleFont

```
public static java.awt.Font specialActionTitleFont()
```

Description forthcoming.

specialActionTitlePosition

```
public static int specialActionTitlePosition()
```

Description forthcoming.

standardActionIcon

```
public static javax.swing.Icon standardActionIcon(String aString)
```

Description forthcoming.

standardResourceBundle

```
public static java.util.ResourceBundle standardResourceBundle()
```

Description forthcoming.

standardSmallActionIcon

```
public static javax.swing.Icon standardSmallActionIcon(String aString)
```

Description forthcoming.

titleColor

```
public static java.awt.Color titleColor()
```

Description forthcoming.

titleFont

```
public static java.awt.Font titleFont()
```

Description forthcoming.

usesBorderWithActionIcons

```
public static boolean usesBorderWithActionIcons()
```

Description forthcoming.

usesBorderWithSmallActionIcons

```
public static boolean usesBorderWithSmallActionIcons()
```

Description forthcoming.

usesSpecialColors

```
public static boolean usesSpecialColors()
```

Description forthcoming.

usesSpecialFonts

```
public static boolean usesSpecialFonts()
```

Description forthcoming.

usesTitleWithActionIcons

```
public static boolean usesTitleWithActionIcons()
```

Description forthcoming.

usesTitleWithSmallActionIcons

```
public static boolean usesTitleWithSmallActionIcons()
```

Description forthcoming.

usesWindowMenus

```
public static boolean usesWindowMenus()
```

Description forthcoming.

widgetFont

```
public static java.awt.Font widgetFont()
```

Description forthcoming.

EOWindowController

Inherits from: EOComponentController : EOController : Object

Implements: ActionListener (java.awt.event package)
EOComponentController.Activation

Package: com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

XML Tag	Default Rule System Controller Type
None (abstract class)	`windowController`

Method Types

All methods

`EOWindowController`

`generateBorderSizeForRootPaneContainerClass`

actionPerformed

activate

activateWindow

borderSize

borderedSizeForComponentSize

componentDidBecomeInvisible

componentShouldBeResizable

componentSizeForBorderedSize

defaultActions

dispose

generateBorderSize

generateComponent

integrationComponent

minimumIntegrationComponentSize

removeTransientSubcontroller

setUsesActivationAction

setUsesActivationButton

setUsesUserDefaultsWindowLocation

setUsesUserDefaultsWindowSize

setWindowPosition

usesActivationAction

usesActivationButton

usesUserDefaultsWindowLocation

usesUserDefaultsWindowSize

windowPosition

Constructors

EOWindowController

```
public EOWindowController(EOXMLUnarchiver anEOXMLUnarchiver)
```

Description forthcoming.

```
public EOWindowController()
```

Description forthcoming.

Static Methods

generateBorderSizeForRootPaneContainerClass

```
protected static java.awt.Dimension generateBorderSizeForRootPaneContainerClass(
    Class aClass,
    boolean aBoolean)
```

Description forthcoming.

Instance Methods

actionPerformed

```
public void actionPerformed(java.awt.event.ActionEvent anActionEvent)
```

Description forthcoming.

activate

```
public boolean activate()
```

Description forthcoming.

activateWindow

```
public abstract void activateWindow()
```

Description forthcoming.

borderSize

```
public java.awt.Dimension borderSize()
```

Description forthcoming.

borderedSizeForComponentSize

```
public java.awt.Dimension borderedSizeForComponentSize(java.awt.Dimension aDimension)
```

Description forthcoming.

componentDidBecomeInvisible

```
protected void componentDidBecomeInvisible()
```

Description forthcoming.

componentShouldBeResizable

```
protected boolean componentShouldBeResizable(javax.swing.JComponent aJComponent)
```

Description forthcoming.

componentSizeForBorderedSize

```
public java.awt.Dimension componentSizeForBorderedSize(java.awt.Dimension aDimension)
```

Description forthcoming.

defaultActions

```
protected NSArray defaultActions()
```

Description forthcoming.

dispose

```
public void dispose()
```

Description forthcoming.

generateBorderSize

```
protected java.awt.Dimension generateBorderSize()
```

Description forthcoming.

generateComponent

```
protected void generateComponent()
```

Description forthcoming.

integrationComponent

```
public javax.swing.JComponent integrationComponent()
```

Description forthcoming.

minimumIntegrationComponentSize

```
public java.awt.Dimension minimumIntegrationComponentSize()
```

Description forthcoming.

removeTransientSubcontroller

```
protected boolean removeTransientSubcontroller(EOController anEOController)
```

Description forthcoming.

setUsesActivationAction

```
public void setUsesActivationAction(boolean aBoolean)
```

Description forthcoming.

setUsesActivationButton

```
public void setUsesActivationButton(boolean aBoolean)
```

Description forthcoming.

setUsesUserDefaultsWindowLocation

```
public void setUsesUserDefaultsWindowLocation(boolean aBoolean)
```

Description forthcoming.

setUsesUserDefaultsWindowSize

```
public void setUsesUserDefaultsWindowSize(boolean aBoolean)
```

Description forthcoming.

setWindowPosition

```
public void setWindowPosition(int anInt)
```

Description forthcoming.

usesActivationAction

```
public boolean usesActivationAction()
```

Description forthcoming.

usesActivationButton

```
public boolean usesActivationButton()
```

Description forthcoming.

usesUserDefaultsWindowLocation

```
public boolean usesUserDefaultsWindowLocation()
```

Description forthcoming.

usesUserDefaultsWindowSize

```
public boolean usesUserDefaultsWindowSize()
```

Description forthcoming.

windowPosition

```
public int windowPosition()
```

Description forthcoming.

EOWindowObserver

Inherits from:	Object
Implements:	WindowListener (java.awt.event package) NSDisposable
Package:	com.webobjects.eoapplication

Class Description

Documentation for this class is forthcoming. For information on using this class, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

- `activateBestWindow`
- `activatePreviousWindow`
- `activeWindow`
- `blockActiveWindowChangedNotification`
- `controllerForActiveWindow`

controllerForLatestDeactivatedWindow

controllerForWindow

dispose

latestDeactivatedWindow

previousWindowToActivate

registerWindow

registerWindow

registeredWindows

unblockActiveWindowChangedNotification

unregisterWindow

unregisterWindowOfController

visibleWindows

windowActivated

windowClosed

windowClosing

windowDeactivated

windowDeiconified

windowDidBecomeActive

windowDidBecomeInactive

windowDidBecomeInvisible

windowDidBecomeVisible

windowForController

windowIconified

windowOpened

Constructors

EOWindowObserver

```
protected EOWindowObserver()
```

Description forthcoming.

Instance Methods

activateBestWindow

```
public void activateBestWindow()
```

Description forthcoming.

activatePreviousWindow

```
public void activatePreviousWindow()
```

Description forthcoming.

activeWindow

```
public java.awt.Window activeWindow()
```

Description forthcoming.

blockActiveWindowChangedNotification

```
public void blockActiveWindowChangedNotification()
```

Description forthcoming.

controllerForActiveWindow

```
public EOController controllerForActiveWindow()
```

Description forthcoming.

controllerForLatestDeactivatedWindow

```
public EOController controllerForLatestDeactivatedWindow()
```

Description forthcoming.

controllerForWindow

```
public EOController controllerForWindow(java.awt.Window aWindow)
```

Description forthcoming.

dispose

```
public void dispose()
```

Description forthcoming.

latestDeactivatedWindow

```
public java.awt.Window latestDeactivatedWindow()
```

Description forthcoming.

previousWindowToActivate

```
public java.awt.Window previousWindowToActivate()
```

Description forthcoming.

registerWindow

```
public void registerWindow(
    java.awt.Window aWindow,
    EOController anEOController)
```

Description forthcoming.

registerWindow

```
public void registerWindow(java.awt.Window aWindow)
```

Description forthcoming.

registeredWindows

```
public NSArray registeredWindows()
```

Description forthcoming.

unblockActiveWindowChangedNotification

```
public void unblockActiveWindowChangedNotification()
```

Description forthcoming.

unregisterWindow

```
public void unregisterWindow(java.awt.Window aWindow)
```

Description forthcoming.

unregisterWindowOfController

```
public void unregisterWindowOfController(EOController anEOController)
```

Description forthcoming.

visibleWindows

```
public NSArray visibleWindows()
```

Description forthcoming.

windowActivated

```
public void windowActivated(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowClosed

```
public void windowClosed(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowClosing

```
public void windowClosing(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowDeactivated

```
public void windowDeactivated(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowDeiconified

```
public void windowDeiconified(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowDidBecomeActive

```
protected void windowDidBecomeActive(java.awt.Window aWindow)
```

Description forthcoming.

windowDidBecomeInactive

```
protected void windowDidBecomeInactive(java.awt.Window aWindow)
```

Description forthcoming.

windowDidBecomeInvisible

```
protected void windowDidBecomeInvisible(java.awt.Window aWindow)
```

Description forthcoming.

windowDidBecomeVisible

```
protected void windowDidBecomeVisible(java.awt.Window aWindow)
```

Description forthcoming.

windowForController

```
public java.awt.Window windowForController(EOController anEOController)
```

Description forthcoming.

windowIconified

```
public void windowIconified(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

windowOpened

```
public void windowOpened(java.awt.event.WindowEvent aWindowEvent)
```

Description forthcoming.

EOXMLUnarchiver

Inherits from: Object

Package: com.webobjects.eoapplication

Class Description

EOXMLUnarchiver objects contain the parameters used to create controllers (objects of the EOController class and its descendents) in the controller hierarchy. The parameters are determined from an XML specification sent from server.

For more information on using this class, see the book *Getting Started with Direct to Java Client*.

Constants

EOXMLUnarchiver defines the following String constants:

Constant	Description
ActionKeyParameter	
ActionNameParameter	
ActionPriorityParameter	

Constant	Description
ActionWidgetPositionParameter	
AlignmentParameter	
AlignmentWidthParameter	
AlignsComponentsParameter	
AllowsMultipleSelectionParameter	
ArchiveNameParameter	
BorderTypeParameter	
ButtonPositionParameter	
CategoryNameParameter	
CategoryPriorityParameter	
ClassNameParameter	
ColorParameter	
DescriptionPathParameter	
DisabledActionNamesParameter	
DisplayGroupProviderMethodNamePara meter	
DisplaysLabelInWidgetParameter	
DisposeIfDeactivatedParameter	
EditabilityParameter	
EditingContextProviderMethodNamePa rameter	
EnabledDisplayGroupProviderMethodN ameParameter	
EnabledKeyParameter	
EntityParameter	

Constant	Description
FontParameter	
FormatAllowedParameter	
FormatClassParameter	
FormatPatternParameter	
HighlightParameter	
HorizontalBorderParameter	
HorizontallyResizableParameter	
IconNameParameter	
IconURLParameter	
ImageScalingParameter	
IsQueryWidgetParameter	
LabelAlignmentParameter	
LabelComponentPositionParameter	
LabelParameter	
MandatoryRelationshipPathsParamete r	
MaximumValueKeyParameter	
MenuAcceleratorParameter	
MinimumHeightParameter	
MinimumValueKeyParameter	
MinimumWidthParameter	
MultipleWindowsAvailableParameter	
PrefersIconOnlyParameter	
QuickTimeCanvasResizingParameter	

Constant	Description
RelationshipPathParameter	
ScalingHintsParameter	
SendsActionToAllControllersParameter	
SharedIdentifierParameter	
ShortDescriptionParameter	
SmallIconNameParameter	
SmallIconURLParameter	
SortsByColumnOrderParameter	
TaskParameter	
TitleKeysParameter	
TitlePositionParameter	
TitlesDisplayGroupProviderMethodNameParameter	
TitlesEntityParameter	
TypeNameParameter	
URLKeyParameter	
UsesActionParameter	
UsesButtonParameter	
UsesHorizontalLayoutParameter	
UsesLabelComponentParameter	
UsesLargeButtonRepresentationParameter	
UsesTitledBorderParameter	

Constant	Description
UsesUserDefaultsWindowLocationParameter	
UsesUserDefaultsWindowSizeParameter	
ValueKeyParameter	
VerticalBorderParameter	
VerticallyResizableParameter	
WindowPositionParameter	

Method Types

Decoding objects

- decodeAlignmentForKey
- decodeArrayForKey
- decodeBooleanForKey
- decodeClassForKey
- decodeColorForKey
- decodeDictionaryForKey
- decodeEditabilityForKey
- decodeFontForKey
- decodeIntForKey
- decodePositionForKey
- decodeStringForKey
- decodeValueForKey

Other methods

EOXMLUnarchiver

decodeRootObject

decodeChildren

Constructors

EOXMLUnarchiver

```
public EOXMLUnarchiver(NSDictionary values)
```

Creates an XML archiver based on the values NSDictionary.

Static Methods

decodeRootObject

```
public static Object decodeRootObject(NSDictionary aNSDictionary)
```

Decodes the top controller in an XML description, which is represented by an NSDictionary.

Instance Methods

decodeAlignmentForKey

```
public int decodeAlignmentForKey(
    String key,
    int defaultAlignment)
```

Returns an alignment specification (JTextField.LEFT, JTextField.CENTER, or JTextField.RIGHT) for the key XML attribute. If no value for key is specified, returns defaultAlignment.

```
public int decodeAlignmentForKey(String key)
```

Returns an alignment specification (JTextField.LEFT, JTextField.CENTER, or JTextField.RIGHT) for the key XML attribute. If no value for key is specified, returns JTextField.LEFT.

decodeArrayForKey

```
public NSArray decodeArrayForKey(
    String key,
    NSArray defaultArray)
```

Returns an NSArray for the key XML attribute. If no value for key is specified, returns defaultArray.

```
public NSArray decodeArrayForKey(String key)
```

Returns an NSArray for the key XML attribute. If no value for key is specified, returns null.

decodeBooleanForKey

```
public boolean decodeBooleanForKey(
    String key,
    boolean defaultBoolean)
```

Returns a boolean for the key XML attribute. If no value for key is specified, returns defaultBoolean.

```
public boolean decodeBooleanForKey(String key)
```

Returns a `boolean` for `key` XML attribute. If no value for `key` is specified, returns `false`.

decodeChildren

```
public NSArray decodeChildren()
```

Returns an NSArray containing the receiver's decoded children. The children are the objects created from XML tags contained in the receiver's XML description.

decodeClassForKey

```
public Class decodeClassForKey(
    String key,
    Class defaultClass)
```

Returns a Class for the `key` XML attribute. If no value for `key` is specified, returns `defaultClass`.

```
public Class decodeClassForKey(String key)
```

Returns a Class for the `key` XML attribute. If no value for `key` is specified, returns `null`.

decodeColorForKey

```
public java.awt.Color decodeColorForKey(
    String key,
    java.awt.Color defaultColor)
```

Returns a color (a java.awt.Color object) for the `key` XML attribute. If no value for `key` is specified, returns `defaultColor`.

```
public java.awt.Color decodeColorForKey(String key)
```

Returns a color (a java.awt.Color object) for the `key` XML attribute. If no value for `key` is specified, returns `null`.

decodeDictionaryForKey

```
public NSDictionary decodeDictionaryForKey(
    String key,
    NSDictionary defaultDictionary)
```

Returns a NSDictionary for the key XML attribute. If no value for key is specified, returns defaultDictionary.

```
public NSDictionary decodeDictionaryForKey(String key)
```

Returns a NSDictionary for the key XML attribute. If no value for key is specified, returns null.

decodeEditabilityForKey

```
public int decodeEditabilityForKey(
    String key,
    int defaultEditability)
```

Returns an editability specification (EOEditable.IfSupercontrollerEditable, EOEditable.AlwaysEditable, or EOEditable.NeverEditable) for the key XML attribute. If no value for key is specified, returns defaultEditibility.

```
public int decodeEditabilityForKey(String key)
```

Returns an editability specification (EOEditable.IfSupercontrollerEditable, EOEditable.AlwaysEditable, or EOEditable.NeverEditable) for the key XML attribute. If no value for key is specified, returns EOEditable.IfSupercontollerEditable.

decodeFontForKey

```
public java.awt.Font decodeFontForKey(
    String key,
    java.awt.Font defaultFont)
```

Returns a font specification (a java.awt.Font object) for the key XML attribute. If no value for key is specified, returns defaultFont.

```
public java.awt.Font decodeFontForKey(String key)
```

Returns a font specification (a java.awt.Font object) for the key XML attribute. If no value for key is specified, returns null.

decodeIntForKey

```
public int decodeIntForKey(
    String key,
    int defaultInt)
```

Returns an int for the key XML attribute. If no value for key is specified, returns defaultInt.

```
public int decodeIntForKey(String key)
```

Returns an int for the key XML attribute. If no value for key is specified, returns 0.

decodePositionForKey

```
public int decodePositionForKey(
    String key,
    int defaultPosition)
```

Returns a position specification (EOComponentController.Top, EOComponentController.Bottom, EOComponentController.Left, EOComponentController.Right, EOComponentController.TopLeft, EOComponentController.TopRight, EOComponentController.BottomLeft, or EOComponentController.BottomRight) for the key XML attribute. If no value for key is specified, returns defaultPosition.

```
public int decodePositionForKey(String key)
```

Returns a position specification (EOComponentController.Top, EOComponentController.Bottom, EOComponentController.Left, EOComponentController.Right, EOComponentController.TopLeft, EOComponentController.TopRight, EOComponentController.BottomLeft, or EOComponentController.BottomRight) for the key XML attribute. If no value for key is specified, returns EOComponentController.Center.

decodeStringForKey

```
public String decodeStringForKey(
    String key,
    String defaultString)
```

Returns a String for the key XML attribute. If no value for key is specified, returns defaultString.

```
public String decodeStringForKey(String key)
```

Returns a String for the key XML attribute. If no value for key is specified, returns null.

decodeValueForKey

```
public Object decodeValueForKey(String key)
```

Returns an Object for the key XML attribute. If no value for key is specified, returns `null`.

EOAction.ActiveWindowDependentAction

Package: com.webobjects.eoapplication

Interface Description

Documentation for this interface is forthcoming. For information on using this interface, see the book *Getting Started with Direct to Java Client*.

Instance Methods

updateInContextOfActiveWindowController

```
public abstract void updateInContextOfActiveWindowController(EOController anEOController)
```

EOAction.Enabling

Implemented by: EOController

Package: com.webobjects.eoapplication

Interface Description

The EOAction.Enabling interface defines a method, `canPerformActionNamed`, which allows you to tell if an action (an EOAction object) is enabled for the receiver.

Instance Methods

canPerformActionNamed

```
public abstract boolean canPerformActionNamed(String actionName)
```

Returns `true` if the receiver can perform an action (an EOAction object) named `actionName`, `false` otherwise. An EOController's implementation of this method generally returns `false` if the receiver doesn't have an action named `actionName` or if the `actionName` action is disabled.

See Also: `isActionNamedEnabled` (EOController)

EOActionWidgetController.ActionCollector

Package: com.webobjects.eoapplication

Interface Description

Documentation for this interface is forthcoming. For information on using this interface, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

`collectedActions`

Instance Methods

collectedActions

```
public abstract NSArray collectedActions()
```

EOAssociationConnector

Implemented by: EOAssociationController, EOEntityController, EORangeValueController, EOTableController

Package: com.webobjects.eoapplication

Interface Description

EOAssociationConnector is an interface that defines an object that can assume the responsibilities for connecting and disconnecting the associations of a transient subcontroller.

Instance Methods

takeResposibilityForConnectionOfAssociation

```
public abstract void
    takeResposibilityForConnectionOfAssociation(com.webobjects.eointerface.EOAssociation association)
```

Invoked when one of the receiver's subcontrollers is disposed as a transient controller. This method instructs the receiver to assume responsibility for managing the subcontroller's EOAssociation, association.

EOComponentController.Activation

Package: com.webobjects.eoapplication

Interface Description

Documentation for this interface is forthcoming. For information on using this interface, see the book *Getting Started with Direct to Java Client*.

Instance Methods

activate

```
public abstract boolean activate()
```

EOComponentController.EndEditing

Package: com.webobjects.eoapplication

Interface Description

Documentation for this interface is forthcoming. For information on using this interface, see the book *Getting Started with Direct to Java Client*.

Instance Methods

endEditing

```
public abstract boolean endEditing()
```

EOComponentController.Modal

Package: com.webobjects.eoapplication

Interface Description

Documentation for this interface is forthcoming. For information on using this interface, see the book *Getting Started with Direct to Java Client*.

Method Types

All methods

`isModal`

Instance Methods

isModal

```
public abstract boolean isModal()
```

EOComponentController.ResetUserInterface

Package: com.webobjects.eoapplication

Interface Description

Documentation for this interface is forthcoming. For information on using this interface, see the book *Getting Started with Direct to Java Client*.

Instance Methods

resetUserInterface

```
public abstract void resetUserInterface()
```

EOController.Enumeration

Implements: java.util.Enumeration

Package: com.webobjects.eoapplication

Interface Description

EOController.Enumeration is an interface that defines an enumeration that iterates over a set of EOController objects. It adds one method to the java.util.Enumeration interface: `nextController`, which simply returns the next controller in the enumeration's set. The `nextController` method saves you from having to cast the returned object to an EOController.

Use the EOController method `controllerEnumeration` to get an EOController.Enumeration. You can create three types of enumerations:

`SubcontrollersEnumeration`
: Includes all the descendants of a controller—the controller's subcontrollers, their subcontrollers, and so on down the controller hierarchy—not including the controller itself.

`SupercontrollersEnumeration`
: Includes all the ancestors of a controller—the controller's supercontroller, its supercontroller, and so on up the controller hierarchy—not including the controller itself.

`ControllerAndSubcontrollersEnumeration`
: Includes a controller and all its descendants.

You can further restrict the controllers included in an enumeration by specifying an interface the controllers must implement in order to be included. For more information, see the method description for `controllerEnumeration` in the EOController class specification.

Instance Methods

nextController

```
public abstract EOController nextController()
```

Returns the next controller in the enumeration. Use this method instead of `nextElement` because it saves you a cast and because it's implementation is more efficient.

See Also: `controllerEnumeration` (EOController)

EOModalDialogController.ModalActions

Package: com.webobjects.eoapplication

Interface Description

Documentation for this interface is forthcoming.

Method Types

All methods

- `cancel`
- `modalDialogShouldClose`
- `ok`

Instance Methods

cancel

```
public abstract void cancel()
```

Description forthcoming.

modalDialogShouldClose

```
public abstract boolean modalDialogShouldClose()
```

Description forthcoming.

ok

```
public abstract boolean ok()
```

Description forthcoming.

EODocument

Implemented by: EODocumentController

Implements: EOObjectDisplay

Package: com.webobjects.eoapplication

Interface Description

EODocument is an interface that defines the behavior of a controller that displays and edits enterprise objects.

Instance Methods

isDocumentForGlobalID

```
public abstract boolean isDocumentForGlobalID(
    com.webobjects.eocontrol.EOGlobalID globalID,
    String entityName)
```

Returns `true` if the receiver is a document for the enterprise object associated with `globalID` and `entityName`, `false` otherwise. Typically implementations return true if the receiver's display group is displaying the specified enteprise object.

isEdited

```
public abstract boolean isEdited()
```

Returns `true` if the receiver has unsaved edits, `false` otherwise.

save

```
public abstract boolean save()
```

Saves the receiver's edits, returning `true` on success or `false` otherwise.

saveIfUserConfirms

```
public abstract boolean saveIfUserConfirms(
    String operationTitle,
    String message)
```

If the receiver's enterprise object has been edited, opens an alert panel that allows the user to save the edits, discard the edits, or cancel the save operation. The `operationTitle` argument is used as the title of the alert panel, and `message` is used as the message in the panel. Returns `true` if the save succeeds, `false` upon failure or if the user cancels.

setEdited

```
public abstract void setEdited(boolean flag)
```

Sets the receiver's edited status according to `flag`.

EOEditable

Implemented by: EOAssociationController, EODocumentController, EORangeValueController

Package: com.webobjects.eoapplication

Interface Description

EOEditable is an interface that defines an API for managing the editability of a branch of the controller hierarchy. EOEditable controllers usually base the editability of their user interfaces on the editability of their supercontrollers. Thus, by default all the EOEditable subcontrollers of an editable controller are also editable. To enable or disable a portion of an application's user interface, you need only message the highest level controller associated with that user interface.

Constants

EOEditable defines the following `int` constants to identify the editability of an EOEditable controller:

Constant	Description
`NeverEditable`	The controller is never editable.
`AlwaysEditable`	The controller is always editable.
`IfSupercontrollerEditable`	The controller is editable only if its supercontroller is editable. If none of the controller's ancestors implement EOEditable, then its the same as if the controller is `AlwaysEditable`.

Instance Methods

editability

```
public abstract int editability()
```

Returns the editability of the receiver, one of `NeverEditable`, `AlwaysEditable`, or `IfSupercontrollerEditable`. The default behavior should be to return `IfSupercontrollerEditable`.

isEditable

```
public abstract boolean isEditable()
```

Returns `true` if the receiver is editable, and `false` otherwise. The default behavior should be to return `true` if the receiver is currently editable. The receiver is editable if:

- The receiver's editability is `AlwaysEditable`.

- The receiver's editability is IfSupercontrollerEditable and sending isEditable to the first EOEditable ancestor of the receiver returns true.

setEditability

```
public abstract void setEditability(int editability)
```

Sets the receiver's editability to editability, one of NeverEditable, AlwaysEditable, or IfSupercontrollerEditable.

supercontrollerEditabilityDidChange

```
public abstract void supercontrollerEditabilityDidChange()
```

Invoked to notify the receiver that the editability of its supercontroller changed, giving the receiver the opportunity to update its user interface to match the editability of the supercontroller.

takeResponsibilityForEditabilityOfAssociation

```
public abstract void takeResponsibilityForEditabilityOfAssociation(
    com.webobjects.eointerface.EOAssociation association)
```

Invoked when one of the receiver's subcontrollers is disposed as a transient controller. This method instructs the receiver to assume responsibility for managing the editability of the subcontroller's EOAssociation, association.

EOObjectDisplay

Implemented by: EOEntityController

Package: com.webobjects.eoapplication

Interface Description

EOObjectDisplay is an interface that defines the behavior of a controller that displays enterprise objects using an EODisplayGroup.

Instance Methods

controllerDisplayGroup

```
public abstract com.webobjects.eointerface.EODisplayGroup controllerDisplayGroup()
```

Returns a display group containing the receiver—an EOController or subclass. This display group can be used to connect controller methods to the user interface.

displayGroup

```
public abstract com.webobjects.eointerface.EODisplayGroup displayGroup()
```

Returns the display group the receiver uses to display and edit the properties of its enterprise objects.

editingContext

```
public abstract com.webobjects.eocontrol.EOEditingContext editingContext()
```

Returns the editing context the receiver uses to manage the graph of its enterprise objects.

entityName

```
public abstract String entityName()
```

Returns the name of the entity that describes the enterprise objects the receiver displays with its display group.

This Apple manual was written, edited, and composed on a desktop publishing system using Apple Macintosh computers and FrameMaker software.

Line art was created using Adobe™ Illustrator and Adobe Photoshop.

Text type is Palatino® and display type is Helvetica®. Bullets are ITC Zapf Dingbats®. Some elements, such as program listings, are set in Adobe Letter Gothic.