EODistribution Framework

Objective–C API Reference

The EODistribution Layer

Framework:	System / Library / Frameworks / EOJavaClient.framework
Header File Directories:	System / Library / Frameworks / EOJavaClient.framework / Headers

Introduction

The EODistribution layer is used in Java Client applications. It consists of two parts: a Yellow Box framework for the server and a Java package for the client. The EODistribution (or, simply, "distribution") layer performs by-copy object distribution and synchronization. It is responsible for synchronizing the states of the object graphs on the client and on the application server. The distribution layer handles communication over a "channel" (which use transports such as HTTP or CORBA) and moves properties in both directions, that is, as objects are fetched and changes are saved. It encodes and decodes objects as they travel back and forth over the channel.

The classes in the server side of the EODistribution layer are provided by the EOJavaClient framework, and server side APIs are available in both Objective-C and Java (the Java package for the server side APIs is com.apple.yellow.eodistribution). The classes on the client side are implemented in pure Java and live in the com.apple.client.eodistribution package.

The following table summarizes each class in the EODistribution layer:

Class	Client	Server	Description
EODistributedDataSource	X		Fetches data using an EOEditingContext on the client as its source of objects.
EODistributedObjectStore	X		Handles interaction with the distribution layer's channel, incorporating knowledge of that channel so it can forward messages it receives from the server to its editing contexts and forward messages from its editing contexts to the server.
EODistributionChannel	X		Abstract class for distribution channels.
EODistributionContext		X	Encodes data to send to the client and decodes data it receives from the client; also tracks and communicates any changes on the server object graph to the client.
EOHTTPChannel	X		Implements a distribution channel using HTTP as the transport.
WOJavaClientApplet		X	Used to download and create the applet on the client.

In addition, `EOAccessAdditions.h` declares Objective-C categories on EOEntity, EOClassDescription, and EOEntityClassDescription. The methods in these categories return client-specific information stored in model files.

EOClassDescription Additions

Category of:	EOClassDescription
Declared in:	EODistribution/EOAccessAdditions.h

Category Description

This Objective-C category adds a number of methods to EOClassDescription (which is part of the EOControl framework). These methods are intended to return various properties bound to the client-side class that corresponds to the receiver's EOEntity. In this category implementation, however, all of these methods return `nil`. See the EOEntityClassDescription Additions reference for descriptions of implementations that return actual data.

Instance Methods

clientAttributeKeys

```
– (NSArray *)clientAttributeKeys
```

Intended to return an array containing the names of those attributes that are bound to the client-side class that corresponds to the receiver's EOEntity, this category implementation simply returns `nil`.

See Also: `– clientAttributeKeys` (EOEntityClassDescription Additions)

clientToManyRelationshipKeys

`- (NSArray *)clientToManyRelationshipKeys`

Intended to return an array containing the names of those to-many relationships that are bound to the client-side class that corresponds to the receiver's EOEntity, this category implementation simply returns `nil`.

See Also: `- clientToManyRelationshipKeys` (EOEntityClassDescription Additions)

clientToOneRelationshipKeys

`- (NSArray *)clientToOneRelationshipKeys`

Intended to return an array containing the names of those to-many relationships that are bound to the client-side class that corresponds to the receiver's EOEntity, this category implementation simply returns `nil`.

See Also: `- clientToOneRelationshipKeys` (EOEntityClassDescription Additions)

EODistributionContext

Inherits from: NSObject

Declared in: EOJavaClient/EODistributionContext.h

Class Description

An EODistributionContext object encodes data to send to the client and decodes data received from the client over the distribution channel. An EODistributionContext is also responsible for tracking the state of the server-side object graph and communicating any changes to the client, thus keeping the client and server object graphs in sync. EODistributionContext—or, if implemented, its delegate—validates remote invocations originating from client objects. The server-side EODistributionContext communicates with the EODistributedObjectStore on the client. See the EODistributionContext. Delegate protocol description for more information on security and validation.

Constants

The `EODistributionContext.h` header defines NSString constants for the names of the notifications it posts. For more information, see "Notifications" (page 10).

Instance Methods

delegate

```
- (id)delegate
```

Returns the receiver's delegate.

editingContext

```
- (EOEditingContext *)editingContext
```

Returns the receiver's editing context.

See Also: `- initWithSession:editingContext:`, `- initWithSession:`

initWithSession:

```
- (id)initWithSession:(WOSession *)session
```

Initializes a new EODistributionContext for use in *session* with *session*'s default editing context.

initWithSession:editingContext:

```
- (id)initWithSession:(WOSession *)session
   editingContext:(EOEditingContext *)editingContext
```

Initializes a new EODistributionContext for use in *session* and with *editingContext*.

invocationTarget

`- (id)invocationTarget`

Returns the target object to which client requests are sent for processing.

See Also: `- responseToClientMessage:`

responseToClientMessage:

`- (NSData *)responseToClientMessage:(NSData *)`*`message`*

Called to generate the response to a client request. The target object specified with `setInvocationTarget:` is invoked with the client request, and the response returned by the target object is returned from this method.

See Also: `- invocationTarget`

session

`- (WOSession *)session`

Returns the receiver's session.

See Also: `- initWithSession:editingContext:`, `- initWithSession:`

setDelegate:

`- (void)setDelegate:(id)`*`delegate`*

Specifies that *`delegate`* should be used by the EODistributionContext to validate method invocations and fetches requested by the client. For more information, see the EODistributionContext. Delegate protocol specification.

See Also: `- delegate`

setInvocationTarget:

`- (void)setInvocationTarget:(id)`*`invocationTarget`*

Specifies the target object to which client requests are sent for processing.

See Also: `- responseToClientMessage:`

Notifications

EOLoadUserDefaultsNotification

Posted whenever a distribution context receives a request for user default values from a client application. Receivers can load default values (from a database, for example) and add them to the mutable dictionary provided in the notification's userInfo.

Notification object	`this`
userInfo	An NSDictionary containing a single entry with the key "defaults" and an NSMutableDictionary as the value. The keys to the mutable subdictionary are the names of the user defaults and the corresponding values are the default values themselves.

EOSaveUserDefaultsNotification

Posted whenever the distribution context receives user default values from a client application. Receivers can use this notification to store the default values (in a database, for example).

Notification object	`this`
userInfo	An NSDictionary containing a single entry with the key "defaults" and another NSDictionary as the value. The keys to the mutable subdictionary are the names of the user defaults and the corresponding values are the default values themselves.

EOEntity Additions

Category of:	EOEntity
Declared in:	EODistribution/EOAccessAdditions.h

Category Description

This Objective–C category adds a number of methods to EOEntity (which is part of the EOAccess framework). These methods are used by the template generation code to obtain information about the client-side class corresponding to the receiving EOEntity object.

The information returned by these methods is stored in your model file. To change it use the EOJavaClientExtensions bundle for EOModeler.

Instance Methods

clientClassName

`- (NSString *)clientClassName`

Returns the name of the client-side enterprise object class associated with the receiver. If no client-side class name has yet been registered for the receiver, this method returns the name of the receiving class (either EOEntity or a subclass of EOEntity).

See Also: `- clientClassNameWithoutPackage`, `- clientClassPackage`, `- referenceClientClassName`, `— className` **(EOEntity class)**

clientClassNameWithoutPackage

`- (NSString *)clientClassNameWithoutPackage`

Returns the name of the client-side enterprise object class associated with the receiver, with the package name removed. If no client-side class name has yet been registered for the receiver, this method returns the name of the receiving class (either EOEntity or a subclass of EOEntity).

See Also: `- clientClassName`, `- clientClassPackage`, `- referenceClientClassName`, `— className` **(EOEntity class)**

clientClassPackage

`- (NSArray *)clientClassPackage`

If the string returned from `clientClassName` includes a package name, this method returns an array of one item, the package name. Otherwise, this method returns an empty array.

See Also: `- clientClassName`, `- clientClassNameWithoutPackage`

clientClassProperties

`- (NSArray *)clientClassProperties`

Returns an array containing the properties that are bound to the client-side class corresponding to the receiver. If no information about the client-side class's properties is available, this method returns the receiver's class properties. The properties returned by this method are the attributes and relationships that are used by the client. Only these attributes and relationships will be shipped to the client.

See Also: – `classProperties` (EOEntity class), `- clientClassPropertyNames`

clientClassPropertyAttributeNames

`- (NSArray *)clientClassPropertyAttributeNames`

Returns the names of those properties obtained using `clientClassProperties` that are attributes.

See Also: `- clientClassPropertyNames`, `- clientClassPropertyToManyRelationshipNames`, `- clientClassPropertyToOneRelationshipNames`

clientClassPropertyNames

`- (NSArray *)clientClassPropertyNames`

Returns an array containing the names of the properties that are bound to the client-side class corresponding to the receiver. If no information about the client-side class's properties is available, this method returns the names of the receiver's class properties. The property names returned by this method are the attributes and relationships that are used by the client. Only these attributes and relationships will be shipped to the client.

See Also: `- clientClassProperties`

clientClassPropertyToManyRelationshipNames

`- (NSArray *)clientClassPropertyToManyRelationshipNames`

Returns the names of those properties obtained using `clientClassProperties` that are to-many relationships.

See Also: `- clientClassPropertyAttributeNames`, `- clientClassPropertyNames`, `- clientClassPropertyToOneRelationshipNames`

clientClassPropertyToOneRelationshipNames

`- (NSArray *)clientClassPropertyToOneRelationshipNames`

Returns the names of those properties obtained using `clientClassProperties` that are to-one relationships.

See Also: `- clientClassPropertyAttributeNames`, `- clientClassPropertyNames`, `- clientClassPropertyToManyRelationshipNames`

referenceClientClassName

`- (NSString *)referenceClientClassName`

Returns the name of the client-side class that corresponds to the receiver. If the receiver doesn't have an associated client-side class, `referenceClientClassName` returns "EOEnterpriseObject".

See Also: `- clientClassName`

EOEntityClassDescription Additions

Category of:	EOEntityClassDescription
Declared in:	EODistribution/EOAccessAdditions.h

Category Description

This Objective-C category adds a number of methods to EOEntityClassDescription (which is part of the EOAccess framework). These methods return various properties bound to the client-side class that corresponds to the receiver's EOEntity.

The information returned by these methods is stored in your model file. To change it use the EOJavaClientExtensions bundle for EOModeler.

Instance Methods

clientAttributeKeys

```
- (NSArray *)clientAttributeKeys
```

Returns an array containing the names of those attributes that are bound to the client-side class that corresponds to the receiver's EOEntity.

See Also: `- clientClassProperties` (EOEntity Additions)

clientToManyRelationshipKeys

`- (NSArray *)clientToManyRelationshipKeys`

Returns an array containing the names of those to-many relationships that are bound to the client-side class that corresponds to the receiver's EOEntity.

See Also: `- clientClassPropertyToManyRelationshipNames` (EOEntity Additions)

clientToOneRelationshipKeys

`- (NSArray *)clientToOneRelationshipKeys`

Returns an array containing the names of those to-one relationships that are bound to the client-side class that corresponds to the receiver's EOEntity.

See Also: `- clientClassPropertyToOneRelationshipNames` (EOEntity Additions)

WOJavaClientApplet

Inherits from:	WOComponent
Declared in:	EOJavaClient/EODistributionContext.h

Class Description

WOJavaClientApplet is the web component used by Java Client applications to create and download to the client an applet of class com.apple.client.interface.EOApplet. This component passes several parameters to the applet, including the dimensions, code/codebase, and additional EOApplication-specific parameters—such as the initial EOInterfaceController subclass name and language.

WOJavaClientApplet is able to generate the HTML required by SunSoft's Java Plug-in for Microsoft's Internet Explorer and Netscape's browsers. The plug-in is usually required for Netscape, while Internet Explorer often works without it (whether or not the plug-in is required depends on the applet's contents).

Java Client applications can be started outside of a web browser using the following command-line syntax:

```
java -classpath path_list com.apple.client.eointerface.EOApplication application_url
```

When a Java Client application is started outside of a browser, the WOJavaClientApplet is still used on the server side to determine the additional EOApplication-specific parameters. Thus the bindings listed below can still apply even in the absence of a web browser.

The following tables lists those bindings used by WOJavaClientApplet:

Binding	Description
`width`	Width of applet in the HTML page.
`height`	Height of applet in the HTML page.
`useJavaPlugin`	If this flag is YES, the WOJavaClientApplet generates HTML that causes Internet Explorer and Netscape's browsers to use SunSoft's Java Plug-in.
`archive`	Standard applet parameter.
`code`	Standard applet parameter.
`codebase`	Standard applet parameter.
`distributionContext`	The EODistributionContext used by the applet to handle requests from the client. If the WOJavaClientApplet does not have a binding for the distribution context, it instantiates one with the session's `defaultEditingContext`, sets the session as the delegate of the distribution context, and itself as the invocation target.
`interfaceControllerClassName`	The class name of the initial EOInterfaceController subclass that becomes visible when an application is launched (in the applet if launched inside a browser).
`applicationClassName`	(Objective-C only) The class name of the EOApplication subclass used for the shared application object.
`language`	The preferred language for the application.
`channelClassName`	The class name of the distribution channel to be used by the client.

Constants

EODistribution defines the following NSString constants in `WOJavaClientApplet.h`. Each constant corresponds to a WOJavaClientApplet binding and is a key for use in the dictionary returned by `clientSideRequestApplicationParameters`.

Constant	Corresponding Binding
`EOWidthKey`	`width`
`EOHeightKey`	`height`
`EOUseJavaPluginKey`	`useJavaPlugin`
`EOArchiveKey`	`archive`
`EOCodeKey`	`code`
`EOCodebaseKey`	`codebase`
`EODistributionContextKey`	`distributionContext`
`EOInterfaceControllerClassNameKey`	`interfaceControllerClassName`
`EOApplicationClassNameKey`	`applicationClassName`
`EOLanguageKey`	`language`
`EOChannelClassNameKey`	`channelClassName`

The `WOJavaClientApplet.h` header defines the following additional NSString constants.

Constant	Description
`EOAllParameterNamesKey`	Used internally to collect the names of all HTML parameters passed to the client (the names of all bindings of the WOJavaClientApplet), including any additional bindings that you add to the applet.
`EOSessionIDKey`	Used internally to identify the session with which the server side EODistributionContext is associated.
`EOComponentURLKey`	Used internally to identify the WOJavaClientApplet component on the server side which corresponds to the EOApplet on the client side.

The `WOJavaClientApplet.h` header also defines NSString constants for the names of the notifications it posts. For more information, see "Notifications" (page 22).

Instance Methods

applicationClassName

`- (NSString *)applicationClassName`

Returns the value for the `applicationClassName` binding. This binding identifies the name of the EOApplication subclass used for the shared application object.

See Also: `- channelClassName`, `- clientSideRequestApplicationParameters`, `- interfaceControllerClassName`

archive

`- (NSString *)archive`

If the applet has a binding for `archive`, the value of that binding is returned. Otherwise, the default `archive` binding—"eojavaclient.jar"—is returned.

channelClassName

`- (NSString *)channelClassName`

Returns the string value bound to the `channelClassName` binding. The `channelClassName` identifies the class of the object that the client uses for a distribution channel.

See Also: `- applicationClassName`, `- interfaceControllerClassName`

clientSideRequestApplicationParameters

`- (NSDictionary *)clientSideRequestApplicationParameters`

Returns a dictionary with the values of all the bindings that have been set. This method is used by EOApplication on the client to warm up a Java application started outside of a browser.

See Also: `- applicationClassName`, `- interfaceControllerClassName`

code

`- (NSString *)code`

If the applet has a binding for `code`, the value of that binding is returned. Otherwise, the default `code` binding—“com.apple.client.eointerface.EOApplet”—is returned.

codebase

`- (NSString *)codebase`

If the applet has a binding for `codebase`, the value of that binding is returned. Otherwise, this method checks to see if the request came through a web server and, if so, returns a URL relative to `cgi-bin/WebObjects` for the resource request handler. If the request didn’t come through a web server, this method returns “/WebObjects/Java”.

distributionContext

`- (EODistributionContext *)distributionContext`

Returns the EODistributionContext used by this component to handle client requests.

handleClientRequest

`- (id)handleClientRequest`

Using the component's EODistributionContext, generates a response for a client request.

See Also: `- responseToClientMessage:` (EODistributionContext **class)**

interfaceControllerClassName

`- (NSString *)interfaceControllerClassName`

Returns the value bound to `interfaceControllerClassName`.

See Also: `- applicationClassName`, `- channelClassName`,
`- clientSideRequestApplicationParameters`

Notifications

WOJavaClientAppletDidVendComponentURLNotification

Posted after the WOJavaClientApplet vends a component URL. The notification contains:

Notification Object	The WOJavaClientApplet that vended a component URL.
Userinfo	None

WOJavaClientAppletWillDeallocNotification

Posted whenever the WOJavaClientApplet is about to be deallocated. The notification contains:

Notification Object	The WOJavaClientApplet that's about to be deallocated.
Userinfo	None

Deprecated API

This file enumerates those EODistribution classes and methods that have been deprecated and should no longer be used. Wherever possible, notes have been included to indicate what API should be used in place of the deprecated class or method.

EODistributionContext

An EODistributionContext is now strongly associated with a WOSession. Consequently, the initializer that takes only an editing context is deprecated.

initWithEditingContext:

`- (id)initWithEditingContext:(EOEditingContext *)`*`editingContext`*

Deprecated in Enterprise Objects Framework 4.5. Use `initWithSession:editingContext:` instead.

T H E   A P P L E   P U B L I S H I N G   S Y S T E M

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.