

initWithName:

Getting the entity setEntity:

entity

Setting the name setName:

name

+ isValidName:

Setting date information+ defaultCalendarFormat

setCalendarFormat:

calendarFormat

setClientTimeZone:

clientTimeZone

setServerTimeZone:

serverTimeZone

Setting external definitions setColumnName:

Checking type information referencesProperty:
isDerived
isFlattened
setReadOnly:
isReadOnly

Setting SQL statement formats setInsertFormat:
insertFormat
setSelectFormat:
selectFormat
setUpdateFormat:
updateFormat

Setting the user dictionary setUserDictionary:
userDictionary

setCalendarFormat:

(NSString *)calendarFormat

Returns the format used when an NSDate object is initialized from a string (typed by the user of a example), or nil if it hasn't been set. A calendar format applies only to attributes that represent date class specification in the Foundation Kit for more information.

valueClassName

(NSTimeZone *)clientTimeZone

Returns the time zone used for NSDate objects in the client application, or the local time zone if one hasn't been set. EOAdaptorChannel automatically converts dates between the time zones used by the database server for fetching and saving values. Applies only to attributes that represent dates.

serverTimeZone, + localTimeZone (NSTimeZone)

(NSString *)columnName

Returns a derived or flattened attribute's definition, or nil if the attribute is simple. An attribute's definition is a value expression defining a derived attribute, such as `^salary * 12^`, or a data path for a flattened attribute, such as `^toAuthor.name^`.

columnName, externalType

(EOEntity *)entity

Returns the entity that contains the attribute.

(NSString *)externalType

Returns the name the adaptor uses for the attribute's type as it's stored in the database for example an Oracle `^NUMBER^`.

columnName

initWithName:(NSString *)name

Initializes a newly allocated EOAttribute with name as its name. The EOAttribute needs to belong to an entity and have a column name or definition before it's usable. This is the designated initializer for the EOAttribute class.

addAttribute: (EOEntity), setColumnName:, setDefinition:

(NSString *)insertFormat

Returns the format string used for including the attribute's value in a SQL INSERT statement. A %s is replaced by the attribute's value when an INSERT statement is generated.

selectFormat, updateFormat

(BOOL)isDerived

Returns YES if the attribute corresponds exactly to one column in the table associated with its entity. For example, an attribute with a definition of `^otherAttributeName + 1^` is derived. A derived attribute is read-only.

Note that flattened attributes are also considered as derived attributes.

isReadOnly, isFlattened, setDefinition:

(BOOL)isFlattened

Returns YES if the value of the attribute can't be modified, NO if it can. Derived and flattened attributes only.

isDerived

(NSString *)name

Returns the attribute's name.

columnName, definition

(BOOL)referencesProperty:aProperty

Returns YES if aProperty (an EOAttribute or EORelationship) is used in the attribute's external definition. Only derived attributes reference other properties.

An attribute references a relationship if it's flattened through that relationship. For example, the flattened employee's department name (as described in the class description) references the

isDerived

(NSString *)selectFormat

Returns the format string used for including the attribute's value in a SQL SELECT statement. A % is replaced by the attribute's external name when a SELECT statement is generated.

insertFormat, updateFormat

(NSTimeZone *)serverTimeZone

Returns the time zone assumed for NSDate objects in the database server, or the local time zone if one has not been specified. EOAdaptorChannel automatically converts dates between the time zones used by the server and the client and saving values. Applies only to attributes that represent dates.

clientTimeZone, + localTimeZone (NSTimeZone)

(void)setCalendarFormat:(NSString *)format

Sets the format used when an NSDate object is initialized from a string (typed by the user, for example). This only applies to attributes that represent dates. See the NSDate class specification for more information.

(void)setClientTimeZone:(NSTimeZone *)aTimeZone

Sets to name the name of the attribute used in communication with the database server. An adaptor identifies the column corresponding to the attribute this name must match the name of a column in the database corresponding to the attribute's entity.

This method makes a derived or flattened attribute simple the definition is released and the column is ready for use with the server.

setDefinition:

(void)setDefinition:(NSString *)definition

Sets to format the attribute's definition as recognized by the database server. format should be either defining a derived attribute, such as `^salary * 12^`, or a data path for a flattened attribute, such as `^table.column^`.

This method converts a simple attribute into a derived or flattened attribute the column name is released and takes its place for use with the server.

setColumnName:

(void)setEntity:(EOEntity *)anEntity

Sets the entity of the attribute to anEntity. You only need to use this method when creating a flattened attribute. Use EOEntity's addAttribute: to associate an existing attribute with an entity.

setDefinition:

(BOOL)setExternalType:(NSString *)typeName

Sets to typeName the type used for the attribute in the database adaptor for example, a Sybase `^varchar^` or `^NUMBER^`. Each adaptor defines the set of types that can be supplied to setExternalType:. The external types specified for a given attribute must correspond to the type used in the database server.

(void)setInsertFormat:(NSString *)aString

Sets to aString the format string used for including the attribute's value in a SQL INSERT statement. The value is replaced by the attribute's value when an INSERT statement is generated. For example, an attribute with a format of `^%V + 5^` has its value increased by 5 when any enterprise object or row is inserted into the database.

setSelectFormat:, setUpdateFormat:

(BOOL)setName:(NSString *)name

Sets the attribute's name to name. Returns YES if successful, NO if name is already in use by another attribute in the relationship of the same entity.

(void)setSelectFormat:(NSString *)aString

Sets to aString the format string used for including the attribute's value in a SQL SELECT statement. The format string is replaced by the attribute's external name when a SELECT statement is generated. For example,

setInsertFormat:, setUpdateFormat:

(void)setServerTimeZone:(NSTimeZone *)aTimeZone

Sets to aTimeZone the time zone used for NSDate objects in the database server. If aTimeZone is nil then the default time zone is used. An EOAdaptorChannel automatically converts dates between the time zones used by the client when fetching and saving values. Applies only to attributes that represent dates.

setClientTimeZone:

(void)setUpdateFormat:(NSString *)aString

Sets to aString the format string used for including the attribute's value in a SQL UPDATE statement. The format string is replaced by the attribute's value when an UPDATE statement is generated. For example, an attribute with a format of "%V + 5" has its value increased by 5 when it's updated in the database.

setInsertFormat:, setSelectFormat:

(void)setUserDictionary:(NSDictionary *)aDictionary

Sets the dictionary of auxiliary data, which your application can use for whatever it needs. The dictionary must contain only NSString, NSData, NSArray, or NSDictionary objects.

(void)setValueClassName:(const char *)name

Sets the attribute's value class name to name. When an EOAdaptorChannel fetches data for the attribute, the application must create an instance of this class.

The class need not exist in the run-time system when this message is sent, but it must exist when an EOAdaptorChannel performs a fetch if the class isn't present. The result depends on the adaptor. See your adaptor's documentation for information on how absent value classes are handled.

setValueType:

(void)setValueType:(NSString *)typeName

Sets to typeName the format type for custom value classes, such as "TIFF" or "RTF". This type name is used by the EODatabaseCustomValues protocol to identify data formats for custom values.

setValueClassName:

(NSDictionary *)userDictionary

Returns a dictionary of user data. Your application can use this to store any auxiliary information in a property list (that is, it contains only NSString, NSData, NSArray, or NSDictionary objects).

(const char *)valueClassName

Returns the name of the class for value types (standard or custom). When data is fetched for the attribute, the application as an instance of this class. For example, if a column from the database is represented by an UIImage this method returns "UIImage".

This class must be present in the run-time system when an EOAdaptorChannel fetches data for the attribute. If the class isn't present the result depends on the adaptor. See your adaptor's documentation for information on which classes are handled.

valueType

(NSString *)valueType

Returns the format type for custom value classes, such as "TIFF" or "RTF". This type name is used by the EODatabaseCustomValues protocol to identify data formats for custom values.

valueClassName