| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RelationSupport.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html)   [**NEXT CLASS**](http://docs.google.com/javax/management/relation/RelationSupportMBean.html) | [**FRAMES**](http://docs.google.com/index.html?javax/management/relation/RelationSupport.html)    [**NO FRAMES**](http://docs.google.com/RelationSupport.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

## **javax.management.relation**

Class RelationSupport

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.management.relation.RelationSupport**

**All Implemented Interfaces:** [MBeanRegistration](http://docs.google.com/javax/management/MBeanRegistration.html), [Relation](http://docs.google.com/javax/management/relation/Relation.html), [RelationSupportMBean](http://docs.google.com/javax/management/relation/RelationSupportMBean.html)

public class **RelationSupport**extends [Object](http://docs.google.com/java/lang/Object.html)implements [RelationSupportMBean](http://docs.google.com/javax/management/relation/RelationSupportMBean.html), [MBeanRegistration](http://docs.google.com/javax/management/MBeanRegistration.html)

A RelationSupport object is used internally by the Relation Service to represent simple relations (only roles, no properties or methods), with an unlimited number of roles, of any relation type. As internal representation, it is not exposed to the user.

RelationSupport class conforms to the design patterns of standard MBean. So the user can decide to instantiate a RelationSupport object himself as a MBean (as it follows the MBean design patterns), to register it in the MBean Server, and then to add it in the Relation Service.

The user can also, when creating his own MBean relation class, have it extending RelationSupport, to retrieve the implementations of required interfaces (see below).

It is also possible to have in a user relation MBean class a member being a RelationSupport object, and to implement the required interfaces by delegating all to this member.

RelationSupport implements the Relation interface (to be handled by the Relation Service).

It implements also the MBeanRegistration interface to be able to retrieve the MBean Server where it is registered (if registered as a MBean) to access to its Relation Service.

**Since:** 1.5

| **Constructor Summary** | |
| --- | --- |
| [**RelationSupport**](http://docs.google.com/javax/management/relation/RelationSupport.html#RelationSupport(java.lang.String,%20javax.management.ObjectName,%20javax.management.MBeanServer,%20java.lang.String,%20javax.management.relation.RoleList))([String](http://docs.google.com/java/lang/String.html) relationId, [ObjectName](http://docs.google.com/javax/management/ObjectName.html) relationServiceName, [MBeanServer](http://docs.google.com/javax/management/MBeanServer.html) relationServiceMBeanServer, [String](http://docs.google.com/java/lang/String.html) relationTypeName, [RoleList](http://docs.google.com/javax/management/relation/RoleList.html) list)            Creates a RelationSupport object. |
| [**RelationSupport**](http://docs.google.com/javax/management/relation/RelationSupport.html#RelationSupport(java.lang.String,%20javax.management.ObjectName,%20java.lang.String,%20javax.management.relation.RoleList))([String](http://docs.google.com/java/lang/String.html) relationId, [ObjectName](http://docs.google.com/javax/management/ObjectName.html) relationServiceName, [String](http://docs.google.com/java/lang/String.html) relationTypeName, [RoleList](http://docs.google.com/javax/management/relation/RoleList.html) list)            Creates a RelationSupport object. |

| **Method Summary** | |
| --- | --- |
| [RoleResult](http://docs.google.com/javax/management/relation/RoleResult.html) | [**getAllRoles**](http://docs.google.com/javax/management/relation/RelationSupport.html#getAllRoles())()            Returns all roles present in the relation. |
| [Map](http://docs.google.com/java/util/Map.html)<[ObjectName](http://docs.google.com/javax/management/ObjectName.html),[List](http://docs.google.com/java/util/List.html)<[String](http://docs.google.com/java/lang/String.html)>> | [**getReferencedMBeans**](http://docs.google.com/javax/management/relation/RelationSupport.html#getReferencedMBeans())()            Retrieves MBeans referenced in the various roles of the relation. |
| [String](http://docs.google.com/java/lang/String.html) | [**getRelationId**](http://docs.google.com/javax/management/relation/RelationSupport.html#getRelationId())()            Returns relation identifier (used to uniquely identify the relation inside the Relation Service). |
| [ObjectName](http://docs.google.com/javax/management/ObjectName.html) | [**getRelationServiceName**](http://docs.google.com/javax/management/relation/RelationSupport.html#getRelationServiceName())()            Returns ObjectName of the Relation Service handling the relation. |
| [String](http://docs.google.com/java/lang/String.html) | [**getRelationTypeName**](http://docs.google.com/javax/management/relation/RelationSupport.html#getRelationTypeName())()            Returns name of associated relation type. |
| [List](http://docs.google.com/java/util/List.html)<[ObjectName](http://docs.google.com/javax/management/ObjectName.html)> | [**getRole**](http://docs.google.com/javax/management/relation/RelationSupport.html#getRole(java.lang.String))([String](http://docs.google.com/java/lang/String.html) roleName)            Retrieves role value for given role name. |
| [Integer](http://docs.google.com/java/lang/Integer.html) | [**getRoleCardinality**](http://docs.google.com/javax/management/relation/RelationSupport.html#getRoleCardinality(java.lang.String))([String](http://docs.google.com/java/lang/String.html) roleName)            Returns the number of MBeans currently referenced in the given role. |
| [RoleResult](http://docs.google.com/javax/management/relation/RoleResult.html) | [**getRoles**](http://docs.google.com/javax/management/relation/RelationSupport.html#getRoles(java.lang.String%5B%5D))([String](http://docs.google.com/java/lang/String.html)[] roleNameArray)            Retrieves values of roles with given names. |
| void | [**handleMBeanUnregistration**](http://docs.google.com/javax/management/relation/RelationSupport.html#handleMBeanUnregistration(javax.management.ObjectName,%20java.lang.String))([ObjectName](http://docs.google.com/javax/management/ObjectName.html) objectName, [String](http://docs.google.com/java/lang/String.html) roleName)            Callback used by the Relation Service when a MBean referenced in a role is unregistered. |
| [Boolean](http://docs.google.com/java/lang/Boolean.html) | [**isInRelationService**](http://docs.google.com/javax/management/relation/RelationSupport.html#isInRelationService())()            Returns an internal flag specifying if the object is still handled by the Relation Service. |
| void | [**postDeregister**](http://docs.google.com/javax/management/relation/RelationSupport.html#postDeregister())()            Allows the MBean to perform any operations needed after having been unregistered in the MBean server. |
| void | [**postRegister**](http://docs.google.com/javax/management/relation/RelationSupport.html#postRegister(java.lang.Boolean))([Boolean](http://docs.google.com/java/lang/Boolean.html) registrationDone)            Allows the MBean to perform any operations needed after having been registered in the MBean server or after the registration has failed. |
| void | [**preDeregister**](http://docs.google.com/javax/management/relation/RelationSupport.html#preDeregister())()            Allows the MBean to perform any operations it needs before being unregistered by the MBean server. |
| [ObjectName](http://docs.google.com/javax/management/ObjectName.html) | [**preRegister**](http://docs.google.com/javax/management/relation/RelationSupport.html#preRegister(javax.management.MBeanServer,%20javax.management.ObjectName))([MBeanServer](http://docs.google.com/javax/management/MBeanServer.html) server, [ObjectName](http://docs.google.com/javax/management/ObjectName.html) name)            Allows the MBean to perform any operations it needs before being registered in the MBean server. |
| [RoleList](http://docs.google.com/javax/management/relation/RoleList.html) | [**retrieveAllRoles**](http://docs.google.com/javax/management/relation/RelationSupport.html#retrieveAllRoles())()            Returns all roles in the relation without checking read mode. |
| void | [**setRelationServiceManagementFlag**](http://docs.google.com/javax/management/relation/RelationSupport.html#setRelationServiceManagementFlag(java.lang.Boolean))([Boolean](http://docs.google.com/java/lang/Boolean.html) flag)            Specifies whether this relation is handled by the Relation Service. |
| void | [**setRole**](http://docs.google.com/javax/management/relation/RelationSupport.html#setRole(javax.management.relation.Role))([Role](http://docs.google.com/javax/management/relation/Role.html) role)            Sets the given role. |
| [RoleResult](http://docs.google.com/javax/management/relation/RoleResult.html) | [**setRoles**](http://docs.google.com/javax/management/relation/RelationSupport.html#setRoles(javax.management.relation.RoleList))([RoleList](http://docs.google.com/javax/management/relation/RoleList.html) list)            Sets the given roles. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Constructor Detail** |
| --- |

### RelationSupport

public **RelationSupport**([String](http://docs.google.com/java/lang/String.html) relationId,  
 [ObjectName](http://docs.google.com/javax/management/ObjectName.html) relationServiceName,  
 [String](http://docs.google.com/java/lang/String.html) relationTypeName,  
 [RoleList](http://docs.google.com/javax/management/relation/RoleList.html) list)  
 throws [InvalidRoleValueException](http://docs.google.com/javax/management/relation/InvalidRoleValueException.html),  
 [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html)

Creates a RelationSupport object.

This constructor has to be used when the RelationSupport object will be registered as a MBean by the user, or when creating a user relation MBean whose class extends RelationSupport.

Nothing is done at the Relation Service level, i.e. the RelationSupport object is not added to the RelationService and no checks are performed to see if the provided values are correct. The object is always created, EXCEPT if:

- any of the required parameters is null.

- the same name is used for two roles.

To be handled as a relation, the RelationSupport object has to be added to the Relation Service using the Relation Service method addRelation().

**Parameters:**relationId - relation identifier, to identify the relation in the Relation Service.

Expected to be unique in the given Relation Service.

relationServiceName - ObjectName of the Relation Service where the relation will be registered.

This parameter is required as it is the Relation Service that is aware of the definition of the relation type of the given relation, so that will be able to check update operations (set).

relationTypeName - Name of relation type.

Expected to have been created in the given Relation Service.

list - list of roles (Role objects) to initialize the relation. Can be null.

Expected to conform to relation info in associated relation type.

**Throws:** [InvalidRoleValueException](http://docs.google.com/javax/management/relation/InvalidRoleValueException.html) - if the same name is used for two roles. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if any of the required parameters (relation id, relation service ObjectName, or relation type name) is null.

### RelationSupport

public **RelationSupport**([String](http://docs.google.com/java/lang/String.html) relationId,  
 [ObjectName](http://docs.google.com/javax/management/ObjectName.html) relationServiceName,  
 [MBeanServer](http://docs.google.com/javax/management/MBeanServer.html) relationServiceMBeanServer,  
 [String](http://docs.google.com/java/lang/String.html) relationTypeName,  
 [RoleList](http://docs.google.com/javax/management/relation/RoleList.html) list)  
 throws [InvalidRoleValueException](http://docs.google.com/javax/management/relation/InvalidRoleValueException.html),  
 [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html)

Creates a RelationSupport object.

This constructor has to be used when the user relation MBean implements the interfaces expected to be supported by a relation by delegating to a RelationSupport object.

This object needs to know the Relation Service expected to handle the relation. So it has to know the MBean Server where the Relation Service is registered.

According to a limitation, a relation MBean must be registered in the same MBean Server as the Relation Service expected to handle it. So the user relation MBean has to be created and registered, and then the wrapped RelationSupport object can be created within the identified MBean Server.

Nothing is done at the Relation Service level, i.e. the RelationSupport object is not added to the RelationService and no checks are performed to see if the provided values are correct. The object is always created, EXCEPT if:

- any of the required parameters is null.

- the same name is used for two roles.

To be handled as a relation, the RelationSupport object has to be added to the Relation Service using the Relation Service method addRelation().

**Parameters:**relationId - relation identifier, to identify the relation in the Relation Service.

Expected to be unique in the given Relation Service.

relationServiceName - ObjectName of the Relation Service where the relation will be registered.

This parameter is required as it is the Relation Service that is aware of the definition of the relation type of the given relation, so that will be able to check update operations (set).

relationServiceMBeanServer - MBean Server where the wrapping MBean is or will be registered.

Expected to be the MBean Server where the Relation Service is or will be registered.

relationTypeName - Name of relation type.

Expected to have been created in the given Relation Service.

list - list of roles (Role objects) to initialize the relation. Can be null.

Expected to conform to relation info in associated relation type.

**Throws:** [InvalidRoleValueException](http://docs.google.com/javax/management/relation/InvalidRoleValueException.html) - if the same name is used for two roles. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if any of the required parameters (relation id, relation service ObjectName, relation service MBeanServer, or relation type name) is null.

| **Method Detail** |
| --- |

### getRole

public [List](http://docs.google.com/java/util/List.html)<[ObjectName](http://docs.google.com/javax/management/ObjectName.html)> **getRole**([String](http://docs.google.com/java/lang/String.html) roleName)  
 throws [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html),  
 [RoleNotFoundException](http://docs.google.com/javax/management/relation/RoleNotFoundException.html),  
 [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html)

Retrieves role value for given role name.

Checks if the role exists and is readable according to the relation type.

**Specified by:**[getRole](http://docs.google.com/javax/management/relation/Relation.html#getRole(java.lang.String)) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Parameters:**roleName - name of role **Returns:**the ArrayList of ObjectName objects being the role value **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if null role name [RoleNotFoundException](http://docs.google.com/javax/management/relation/RoleNotFoundException.html) - if:

- there is no role with given name

- the role is not readable.

[RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html) - if the Relation Service is not registered in the MBean Server**See Also:**[setRole(javax.management.relation.Role)](http://docs.google.com/javax/management/relation/RelationSupport.html#setRole(javax.management.relation.Role))

### getRoles

public [RoleResult](http://docs.google.com/javax/management/relation/RoleResult.html) **getRoles**([String](http://docs.google.com/java/lang/String.html)[] roleNameArray)  
 throws [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html),  
 [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html)

Retrieves values of roles with given names.

Checks for each role if it exists and is readable according to the relation type.

**Specified by:**[getRoles](http://docs.google.com/javax/management/relation/Relation.html#getRoles(java.lang.String%5B%5D)) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Parameters:**roleNameArray - array of names of roles to be retrieved **Returns:**a RoleResult object, including a RoleList (for roles successfully retrieved) and a RoleUnresolvedList (for roles not retrieved). **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if null role name [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html) - if the Relation Service is not registered in the MBean Server**See Also:**[setRoles(javax.management.relation.RoleList)](http://docs.google.com/javax/management/relation/RelationSupport.html#setRoles(javax.management.relation.RoleList))

### getAllRoles

public [RoleResult](http://docs.google.com/javax/management/relation/RoleResult.html) **getAllRoles**()  
 throws [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html)

Returns all roles present in the relation.

**Specified by:**[getAllRoles](http://docs.google.com/javax/management/relation/Relation.html#getAllRoles()) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Returns:**a RoleResult object, including a RoleList (for roles successfully retrieved) and a RoleUnresolvedList (for roles not readable). **Throws:** [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html) - if the Relation Service is not registered in the MBean Server

### retrieveAllRoles

public [RoleList](http://docs.google.com/javax/management/relation/RoleList.html) **retrieveAllRoles**()

Returns all roles in the relation without checking read mode.

**Specified by:**[retrieveAllRoles](http://docs.google.com/javax/management/relation/Relation.html#retrieveAllRoles()) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Returns:**a RoleList

### getRoleCardinality

public [Integer](http://docs.google.com/java/lang/Integer.html) **getRoleCardinality**([String](http://docs.google.com/java/lang/String.html) roleName)  
 throws [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html),  
 [RoleNotFoundException](http://docs.google.com/javax/management/relation/RoleNotFoundException.html)

Returns the number of MBeans currently referenced in the given role.

**Specified by:**[getRoleCardinality](http://docs.google.com/javax/management/relation/Relation.html#getRoleCardinality(java.lang.String)) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Parameters:**roleName - name of role **Returns:**the number of currently referenced MBeans in that role **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if null role name [RoleNotFoundException](http://docs.google.com/javax/management/relation/RoleNotFoundException.html) - if there is no role with given name

### setRole

public void **setRole**([Role](http://docs.google.com/javax/management/relation/Role.html) role)  
 throws [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html),  
 [RoleNotFoundException](http://docs.google.com/javax/management/relation/RoleNotFoundException.html),  
 [RelationTypeNotFoundException](http://docs.google.com/javax/management/relation/RelationTypeNotFoundException.html),  
 [InvalidRoleValueException](http://docs.google.com/javax/management/relation/InvalidRoleValueException.html),  
 [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html),  
 [RelationNotFoundException](http://docs.google.com/javax/management/relation/RelationNotFoundException.html)

Sets the given role.

Will check the role according to its corresponding role definition provided in relation's relation type

Will send a notification (RelationNotification with type RELATION\_BASIC\_UPDATE or RELATION\_MBEAN\_UPDATE, depending if the relation is a MBean or not).

**Specified by:**[setRole](http://docs.google.com/javax/management/relation/Relation.html#setRole(javax.management.relation.Role)) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Parameters:**role - role to be set (name and new value) **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if null role [RoleNotFoundException](http://docs.google.com/javax/management/relation/RoleNotFoundException.html) - if there is no role with the supplied role's name or if the role is not writable (no test on the write access mode performed when initializing the role) [InvalidRoleValueException](http://docs.google.com/javax/management/relation/InvalidRoleValueException.html) - if value provided for role is not valid, i.e.:

- the number of referenced MBeans in given value is less than expected minimum degree

- the number of referenced MBeans in provided value exceeds expected maximum degree

- one referenced MBean in the value is not an Object of the MBean class expected for that role

- a MBean provided for that role does not exist

[RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html) - if the Relation Service is not registered in the MBean Server [RelationTypeNotFoundException](http://docs.google.com/javax/management/relation/RelationTypeNotFoundException.html) - if the relation type has not been declared in the Relation Service [RelationNotFoundException](http://docs.google.com/javax/management/relation/RelationNotFoundException.html) - if the relation has not been added in the Relation Service.**See Also:**[getRole(java.lang.String)](http://docs.google.com/javax/management/relation/RelationSupport.html#getRole(java.lang.String))

### setRoles

public [RoleResult](http://docs.google.com/javax/management/relation/RoleResult.html) **setRoles**([RoleList](http://docs.google.com/javax/management/relation/RoleList.html) list)  
 throws [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html),  
 [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html),  
 [RelationTypeNotFoundException](http://docs.google.com/javax/management/relation/RelationTypeNotFoundException.html),  
 [RelationNotFoundException](http://docs.google.com/javax/management/relation/RelationNotFoundException.html)

Sets the given roles.

Will check the role according to its corresponding role definition provided in relation's relation type

Will send one notification (RelationNotification with type RELATION\_BASIC\_UPDATE or RELATION\_MBEAN\_UPDATE, depending if the relation is a MBean or not) per updated role.

**Specified by:**[setRoles](http://docs.google.com/javax/management/relation/Relation.html#setRoles(javax.management.relation.RoleList)) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Parameters:**list - list of roles to be set **Returns:**a RoleResult object, including a RoleList (for roles successfully set) and a RoleUnresolvedList (for roles not set). **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if null role list [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html) - if the Relation Service is not registered in the MBean Server [RelationTypeNotFoundException](http://docs.google.com/javax/management/relation/RelationTypeNotFoundException.html) - if the relation type has not been declared in the Relation Service. [RelationNotFoundException](http://docs.google.com/javax/management/relation/RelationNotFoundException.html) - if the relation MBean has not been added in the Relation Service.**See Also:**[getRoles(java.lang.String[])](http://docs.google.com/javax/management/relation/RelationSupport.html#getRoles(java.lang.String%5B%5D))

### handleMBeanUnregistration

public void **handleMBeanUnregistration**([ObjectName](http://docs.google.com/javax/management/ObjectName.html) objectName,  
 [String](http://docs.google.com/java/lang/String.html) roleName)  
 throws [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html),  
 [RoleNotFoundException](http://docs.google.com/javax/management/relation/RoleNotFoundException.html),  
 [InvalidRoleValueException](http://docs.google.com/javax/management/relation/InvalidRoleValueException.html),  
 [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html),  
 [RelationTypeNotFoundException](http://docs.google.com/javax/management/relation/RelationTypeNotFoundException.html),  
 [RelationNotFoundException](http://docs.google.com/javax/management/relation/RelationNotFoundException.html)

Callback used by the Relation Service when a MBean referenced in a role is unregistered.

The Relation Service will call this method to let the relation take action to reflect the impact of such unregistration.

BEWARE. the user is not expected to call this method.

Current implementation is to set the role with its current value (list of ObjectNames of referenced MBeans) without the unregistered one.

**Specified by:**[handleMBeanUnregistration](http://docs.google.com/javax/management/relation/Relation.html#handleMBeanUnregistration(javax.management.ObjectName,%20java.lang.String)) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Parameters:**objectName - ObjectName of unregistered MBeanroleName - name of role where the MBean is referenced **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if null parameter [RoleNotFoundException](http://docs.google.com/javax/management/relation/RoleNotFoundException.html) - if role does not exist in the relation or is not writable [InvalidRoleValueException](http://docs.google.com/javax/management/relation/InvalidRoleValueException.html) - if role value does not conform to the associated role info (this will never happen when called from the Relation Service) [RelationServiceNotRegisteredException](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html) - if the Relation Service is not registered in the MBean Server [RelationTypeNotFoundException](http://docs.google.com/javax/management/relation/RelationTypeNotFoundException.html) - if the relation type has not been declared in the Relation Service. [RelationNotFoundException](http://docs.google.com/javax/management/relation/RelationNotFoundException.html) - if this method is called for a relation MBean not added in the Relation Service.

### getReferencedMBeans

public [Map](http://docs.google.com/java/util/Map.html)<[ObjectName](http://docs.google.com/javax/management/ObjectName.html),[List](http://docs.google.com/java/util/List.html)<[String](http://docs.google.com/java/lang/String.html)>> **getReferencedMBeans**()

Retrieves MBeans referenced in the various roles of the relation.

**Specified by:**[getReferencedMBeans](http://docs.google.com/javax/management/relation/Relation.html#getReferencedMBeans()) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Returns:**a HashMap mapping:

ObjectName -> ArrayList of String (role names)

### getRelationTypeName

public [String](http://docs.google.com/java/lang/String.html) **getRelationTypeName**()

Returns name of associated relation type.

**Specified by:**[getRelationTypeName](http://docs.google.com/javax/management/relation/Relation.html#getRelationTypeName()) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Returns:**the name of the relation type.

### getRelationServiceName

public [ObjectName](http://docs.google.com/javax/management/ObjectName.html) **getRelationServiceName**()

Returns ObjectName of the Relation Service handling the relation.

**Specified by:**[getRelationServiceName](http://docs.google.com/javax/management/relation/Relation.html#getRelationServiceName()) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Returns:**the ObjectName of the Relation Service.

### getRelationId

public [String](http://docs.google.com/java/lang/String.html) **getRelationId**()

Returns relation identifier (used to uniquely identify the relation inside the Relation Service).

**Specified by:**[getRelationId](http://docs.google.com/javax/management/relation/Relation.html#getRelationId()) in interface [Relation](http://docs.google.com/javax/management/relation/Relation.html) **Returns:**the relation id.

### preRegister

public [ObjectName](http://docs.google.com/javax/management/ObjectName.html) **preRegister**([MBeanServer](http://docs.google.com/javax/management/MBeanServer.html) server,  
 [ObjectName](http://docs.google.com/javax/management/ObjectName.html) name)  
 throws [Exception](http://docs.google.com/java/lang/Exception.html)

**Description copied from interface:** [**MBeanRegistration**](http://docs.google.com/javax/management/MBeanRegistration.html#preRegister(javax.management.MBeanServer,%20javax.management.ObjectName)) Allows the MBean to perform any operations it needs before being registered in the MBean server. If the name of the MBean is not specified, the MBean can provide a name for its registration. If any exception is raised, the MBean will not be registered in the MBean server.

**Specified by:**[preRegister](http://docs.google.com/javax/management/MBeanRegistration.html#preRegister(javax.management.MBeanServer,%20javax.management.ObjectName)) in interface [MBeanRegistration](http://docs.google.com/javax/management/MBeanRegistration.html) **Parameters:**server - The MBean server in which the MBean will be registered.name - The object name of the MBean. This name is null if the name parameter to one of the createMBean or registerMBean methods in the [MBeanServer](http://docs.google.com/javax/management/MBeanServer.html) interface is null. In that case, this method must return a non-null ObjectName for the new MBean. **Returns:**The name under which the MBean is to be registered. This value must not be null. If the name parameter is not null, it will usually but not necessarily be the returned value. **Throws:** [Exception](http://docs.google.com/java/lang/Exception.html) - This exception will be caught by the MBean server and re-thrown as an [MBeanRegistrationException](http://docs.google.com/javax/management/MBeanRegistrationException.html).

### postRegister

public void **postRegister**([Boolean](http://docs.google.com/java/lang/Boolean.html) registrationDone)

**Description copied from interface:** [**MBeanRegistration**](http://docs.google.com/javax/management/MBeanRegistration.html#postRegister(java.lang.Boolean)) Allows the MBean to perform any operations needed after having been registered in the MBean server or after the registration has failed.

**Specified by:**[postRegister](http://docs.google.com/javax/management/MBeanRegistration.html#postRegister(java.lang.Boolean)) in interface [MBeanRegistration](http://docs.google.com/javax/management/MBeanRegistration.html) **Parameters:**registrationDone - Indicates whether or not the MBean has been successfully registered in the MBean server. The value false means that the registration phase has failed.

### preDeregister

public void **preDeregister**()  
 throws [Exception](http://docs.google.com/java/lang/Exception.html)

**Description copied from interface:** [**MBeanRegistration**](http://docs.google.com/javax/management/MBeanRegistration.html#preDeregister()) Allows the MBean to perform any operations it needs before being unregistered by the MBean server.

**Specified by:**[preDeregister](http://docs.google.com/javax/management/MBeanRegistration.html#preDeregister()) in interface [MBeanRegistration](http://docs.google.com/javax/management/MBeanRegistration.html) **Throws:** [Exception](http://docs.google.com/java/lang/Exception.html) - This exception will be caught by the MBean server and re-thrown as an [MBeanRegistrationException](http://docs.google.com/javax/management/MBeanRegistrationException.html).

### postDeregister

public void **postDeregister**()

**Description copied from interface:** [**MBeanRegistration**](http://docs.google.com/javax/management/MBeanRegistration.html#postDeregister()) Allows the MBean to perform any operations needed after having been unregistered in the MBean server.

**Specified by:**[postDeregister](http://docs.google.com/javax/management/MBeanRegistration.html#postDeregister()) in interface [MBeanRegistration](http://docs.google.com/javax/management/MBeanRegistration.html)

### isInRelationService

public [Boolean](http://docs.google.com/java/lang/Boolean.html) **isInRelationService**()

Returns an internal flag specifying if the object is still handled by the Relation Service.

**Specified by:**[isInRelationService](http://docs.google.com/javax/management/relation/RelationSupportMBean.html#isInRelationService()) in interface [RelationSupportMBean](http://docs.google.com/javax/management/relation/RelationSupportMBean.html) **Returns:**a Boolean equal to [Boolean.TRUE](http://docs.google.com/java/lang/Boolean.html#TRUE) if the object is still handled by the Relation Service and [Boolean.FALSE](http://docs.google.com/java/lang/Boolean.html#FALSE) otherwise.

### setRelationServiceManagementFlag

public void **setRelationServiceManagementFlag**([Boolean](http://docs.google.com/java/lang/Boolean.html) flag)  
 throws [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html)

**Description copied from interface:** [**RelationSupportMBean**](http://docs.google.com/javax/management/relation/RelationSupportMBean.html#setRelationServiceManagementFlag(java.lang.Boolean))

Specifies whether this relation is handled by the Relation Service.

BEWARE, this method has to be exposed as the Relation Service will access the relation through its management interface. It is RECOMMENDED NOT to use this method. Using it does not affect the registration of the relation object in the Relation Service, but will provide wrong information about it!

**Specified by:**[setRelationServiceManagementFlag](http://docs.google.com/javax/management/relation/RelationSupportMBean.html#setRelationServiceManagementFlag(java.lang.Boolean)) in interface [RelationSupportMBean](http://docs.google.com/javax/management/relation/RelationSupportMBean.html) **Parameters:**flag - whether the relation is handled by the Relation Service. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if null parameter

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RelationSupport.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/management/relation/RelationServiceNotRegisteredException.html)   [**NEXT CLASS**](http://docs.google.com/javax/management/relation/RelationSupportMBean.html) | [**FRAMES**](http://docs.google.com/index.html?javax/management/relation/RelationSupport.html)    [**NO FRAMES**](http://docs.google.com/RelationSupport.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).