Adding, moving, deleting model elements

There are two ways to add/move/delete model elements:

- Use *com.nomagic.magicdraw.openapi.uml.ModelElementsManager*
- Call direct functions on *Elements*

The *ModelElementsManager* is the singleton utility class which checks *com.nomagic.uml2.ext.magicdraw.classes.mdkernel.Element* editing permissions and the session existence before executing the function. A lso, it performs a check, if an element can be added to a parent. If the *ModelElementsManager* is not used, a programmer must perform these checks in the code explicitly.
Adding and moving functions are similar with mostly one - move functions does more checking such as checking for a name conflict if *com.nomagic.uml2.ext.magicdraw.classes.mdkernel.NamedElement* is moved. The general rule would be to use an adding function for newly created *Element(s)* and use a moving function for already existing elements in the model.
More advanced model refactoring functions are described in the Refactoring model elements page.

Adding element to owner with *ModelElementsManager*

For adding a new model *Element* into a model, use the *addElement(Element, Element)* method provided by the *ModelElementsManager*.

```
    Project project = ...;
    Class classA = ...;
    Package package = ...;
    // create a new session
    SessionManager.getInstance().createSession(project, "Add class into
package");
    try
    {
        // add a class into a package
        ModelElementsManager.getInstance().addElement(classA, package);
    }
    catch (ReadOnlyElementException e)
    {
    }
    // apply changes and add a command into the command history.
    SessionManager.getInstance().closeSession(project);
```

If a given model element cannot be added into a given parent, *java.lang.IllegalArgumentException* is thrown. For example, an *com.nomagic.uml2.ext.magicdraw.classes.mdkernel.Operation* cannot be added into a *com.nomagic.uml2.ext.magicdraw.classes.mdkernel.Package* or an *Operation* cannot be added into a not locked for editing *com.nomagic.uml2.ext.magicdraw.classes.mdkernel.Class* (in the server project). If an element or parent is null, *java.lang.IllegalArgumentException* also is thrown. If a given element is not editable (read-only), *com.nomagic.magicdraw.openapi.uml.ReadOnlyElementException* is thrown.

Adding element to owner using a direct function

```
    Element parent = ...;
    Class classA = ...;
    Project project = ...;
    // create a new session
    SessionManager.getInstance().createSession(project, "Add class into
parent");
    if (parent.canAdd(classA))
    {
        classA.setOwner(parent);
    }
    // apply changes and add a command into the command history.
    SessionManager.getInstance().closeSession(project);
```

Moving element with *ModelElementsManager*

For moving an existing model *Element*, use the *moveElement(Element, Element)* method provided by the *ModelElementsManager*.

On this page

```
    Project project = ...;
    Class classA = ...;
    Package package = ...;
    // create a new session
    SessionManager.getInstance().createSession(project, "Add class into
package");
    try
    {
        // add a class into a package
        ModelElementsManager.getInstance().moveElement(classA, package);
    }
    catch (ReadOnlyElementException e)
    {
    }
    // apply changes and add a command into the command history.
    SessionManager.getInstance().closeSession(project);
```

If a given model element cannot be moved into a given parent, *java.lang.IllegalArgumentException*is thrown. For example, an *Operation* cannot be moved into a *Package* or an *Operation* cannot be moved into a not locked for editing *Class* (in the server project). If an element or parent is null, *java.lang.IllegalArgumentException* also is thrown. If a given element is not editable (read-only), *com.nomagic.magicdraw.openapi.uml.ReadOnlyElementException* is thrown.

Moving element using a direct function

```
    Element parent = ...;
    Class classA = ...;
    Project project = ...;
    // create a new session
    SessionManager.getInstance().createSession(project, "Add class into
parent");
    if (com.nomagic.uml2.ext.jmi.helpers.ModelHelper.canMoveChildInto
(parent, classA))
    {
        classA.setOwner(parent);
    }
    // apply changes and add a command into the command history.
    SessionManager.getInstance().closeSession(project);
```

Removing element with *ModelElementsManager*

```
    Class classA = ...;
    Project project = ...;
    // create a new session
    SessionManager.getInstance().createSession(project, "Remove class");
    try
    {
         // remove a class
        ModelElementsManager.getInstance().removeElement(classA);
    }
    catch (ReadOnlyElementException e)
    {
    }
    // apply changes and add a command into the command history.
    SessionManager.getInstance().closeSession(project);
```

Removing element using direct call

```
Class classA = ...;
Project project = ...;
// create a new session
SessionManager.getInstance().createSession(project, "Remove class");
if (classA.isEditable()))
{
     classA.dispose();
}
// apply changes and add a command into the command history.
SessionManager.getInstance().closeSession(project);
```

You can find the code examples in <*modeling tool installation directory*>*openapi**examples**hierarchyremover*

Related pages

- Session management
- Checking element editing permissions
- Refactoring model elements