Reverse Options

Demo

Code Reverse

On this page

• Importing JAVA .JAR packages

The reverse is the opposite of code generation. Existing code can be converted to UML models with the help of the MagicDraw reverse feature. In order to perform a reverse operation, prepare the sets the same way you did for code generation, see Code Engineering Sets.

- Choose Reverse from the Code engineering sets item shortcut menu.
- Choose Reverse from the selected set shortcut menu.

The UML model for the component can be reversed in the same way. Simply select the component you are interested in from the browser and click Reverse on its shortcut menu. Models can be reversed without creating a set.

To reverse a model without creating a set

1. In the main menu, click Tools > Code Engineering > Quick Reverse and then choose from one of the needed options. The Round Trip Set dialog opens.

this menu item is unavailable from System Analyst and System Engineer perspectives.

- Select the files from the Round Trip Set dialog box, Add Files tab.
 Click OK. The Reverse Options dialog box appears.

| Reverse Options | × |
|---|--|
| Quick reverse properties A quick reverse generates a UML model from the s field creation, model refresh type, and visualization packages (or both) option if dependencies shall be | source code. Specify the class on options. Select classifiers or e created for it. |
| Create class fields as | Visualization |
| ◯ Attributes | Visualize reversed model |
| Associations | Launch Model Visualizer Create new Class Diagram |
| According to rules | Add to active diagram |
| Resolve collection generics | Analysis |
| Reset already created fields | Create dependencies between: |
| Model refresh type | Packages |
| O Merge model and code | |
| Change model according to code | |
| | OK Cancel Help |

The following table outlines Dialog descriptions:

| Area | Element name | Function | | |
|---|---|---|--|--|
| Create class | Attributes | Class fields are represented in a model as attributes. | | |
| tields as | Associations | Class fields are represented in a model as association ends. | | |
| | According to rules | Association or Attribute creation on the reverse is the ability to enter rules that help decide if an association or attribute must be created on reverse. | | |
| | Resolve collection generics | Reverse engineering can create associations when one class has a collection of other classes and uses Java generics (for example, <i>List<string></string></i>). If selected, types of collections will be resolved (property type will not be a collection, but a real type). | | |
| | | Predefined container types in Java language properties will be appended by all the same containers in this form: <i>.java</i> . <i>util.List</i> <\$\$type\$\$> where \$\$type\$\$ replaced by the value of "type" property when the code is generated. | | |
| | Reset already created fields | Select this option if you want to keep previously created UML representation (attribute or association) for class fields. | | |
| Model refresh type Merge Model and Code The model elements are updated by code. Element | | The model elements are updated by code. Elements that do not exist in the code will not be removed from the model. | | |
| | Change Model According to Code | The model will be created strictly by code. Entities in the model that do not match entities in the code will be discarded. | | |
| Vizualization | ation Visualize reserved model Classes created while reversing can be added to a diagram. | | | |
| | Launch Model Visualizer | After reversing, the Model Visualizer dialog box appears. It will assist you in creating a Class diagram or Sequence diagram (Java only) for newly created entities. | | |
| | Create new Class Diagram | After reversing, the Create Diagram dialog box appears. Create a new diagram where the created entities will be added. | | |
| | Add to active diagram | After reversing, all created entities will be added to the currently opened diagram. | | |
| Analysis. | Classifiers | Dependencies between classes will be analyzed and created. | | |
| Create dependencie s between | Packages | Dependencies only between packages will be created. | | |

Whod bodies are not parsed on dependency search. Only static information is used.

If you have a code set combined from several files, you may see changes you wish to model without reversing all the code. Only changed files should be reversed. You can do this type of reversing by clicking the Refresh button on the set shortcut menu, or by performing model refresh from the Code Engineering Sets dialog.

Importing JAVA .JAR packages

Apart from reversing code operation, Code Engineering Sets also allows you to import JAVA .JAR files and their Packages using Java Bytecode on the Code Engineering Sets shortcut menu.

To include (or import) a JAVA .JAR package with Java code generation

- Create a new UML project or open an existing one.
 In the Containment tree, right-click Code Engineering Sets. Select New > Java Bytecode.

| Be Conta | inmer | nt ^注 印 Diagrams | | | | |
|---------------|----------|---|-------------|--------------|-------|----|
| Containment | | | | @ # X | | |
| 📄 🏠 Q | ₿. | | | Ø - | | |
| ····· 🖾 Model | | | | | | |
| Code En | nainee | ring Sets | | | | |
| | | New | 2 | Java | | |
| | | Edit Source | \$. | Java Byteco | de | |
| | | Generate | c++ | C++ | | ~> |
| | | Check Syntax | CIL | CIL | | |
| | | Update C++ Language Properties and Profiles | CIL | CIL Disasser | mbler | |
| | | Update C# Language Properties and Profiles | • | CORBA IDL | | |
| | | Reverse | | | | |
| | | Refresh | | | | |
| | ~ | Show Code Engineering Sets | | | | |

3. Name your new Java Bytecode engineering set and click **OK**. The new Java Bytecode engineering set is created under **Code Engineering Sets** in the Containment tree.



- 4. Right-click your newly created Java Bytecode engineering set. Select Edit.
- 5. The Round Trip Set dialog opens. In the Working Directory box, specify the root directory of the .JAR file that you want to import.

| 🔀 Round Trip Se | t | | | × |
|---|---|--|---|--------------|
| Manage code en Specify where th reside (working p add model eleme | gineering set e source code resides (worki ackage). Add source code fil nts to the set using the Add i | ng directory) an es to the set us Data from Mode | d where model element ing the Add Files tab ar I tab. | ts nd |
| Working Directory: | D:\MyPlugin | | | |
| Working Package: | Model | | | |
| All files: D:\/MyPlugin d: com.nome d: myplugin d: com.n d: lib d: li | gic.myplugin omagic.myplugin) MyPluginTest.dass urce files (*.dass, * v | + + - | Set: 🍇 JAR Import (Ja | wa Bytecode) |
| | | | ОК | Cancel Help |

6. In the All files list, select the .JAR file you want to import.

the file nierarchy of the imported .JAR file in the Containment tree will start from the working directory you set. For example, if you set the

working directory as the MagicDraw installation directory, the whole file hierarchy that the imported .JAR file is in the installation directory will

be also imported into the Containment tree. elect and click **OK**. The imported .JAR file is now in the Containment tree under the **Model** Package, **File View** Package, and the 7. Select rest of the file hierarchy of the imported .JAR file.

| Containment 20 Diagrams | |
|---|----|
| Containment 🖪 🖗 | x |
| ଳ 🙀 Q 🛝 🔹 📢 | ŧ÷ |
| Model Image: Second state | 5 |

Related Pages:

- Rules of association or attribute creation on reverse

- Code Engineering Sets
 Generating Code
 Reverse Options
 Global options for Code Engineering
 Files of Properties
 Java Code Engineering
 C++ Code Engineering
 CORBA IDL Mapping To UML