TUTORIALS

version 17.0

No Magic, Inc.
2011
CONTENTS

MAGICDRAW BASICS 5
STEP #1 Create a class diagram 5
STEP #2 Create a new class element in diagram 7
STEP #3 Create package element in diagram 9
STEP #4 Create new class element from browser 11
STEP #5 Draw relationships 11
STEP #6 Presentation options 13
STEP #7 Browser options 13

SEQUENCE DIAGRAM CREATION 15
STEP #1 Create a Robustness diagram 15
STEP #2 Create sequence diagram 15
STEP #3 Create lifelines 15
STEP #4 Link messages and alternative fragment 17

JAVA REVERSE TO SEQUENCE DIAGRAM 19
STEP #1 Create code engineering set 19
STEP #2 Create sequence diagram from java source 21
STEP #3 Extend sequence diagram by method 25

APPLYING DIFFERENT COLORS 27
STEP #1 Select all analogous shapes 28
STEP #2 Open the Properties dialog box 28
STEP #3 Open the color dialog box 30
STEP #4 Change properties using Project Options dialog box 31
STEP #5 Extend properties by implementation diagram 31

APPLYING IMAGES TO STEREOTYPES 34
STEP #1 Organize Stereotypes in the Browser, Model Extensions Tree 34
STEP #2 Choose an icon for the stereotype 35
STEP #3 Applied Stereotype Property 36
STEP #4 Suppress attributes and operations 36

DATA PARTITIONING 38
STEP #1 Export a new Module 38
STEP #2 Open module as a new project 39
STEP #3 Reload the module 40

PERFORMING ROUND TRIP 41
STEP #1 Add elements to the newly created code engineering set 41
STEP #2 Generate code 42
STEP #3 Add a new LuggageException class to the model 43
STEP #4 Create class element in the project using reverse 44
STEP #5 Separate luggage into two types 45

CODE GENERATION AND REVERSE 46
STEP #1 Create a new code engineering set 46
## CONTENTS

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Add data you wish to generate</td>
<td>46</td>
</tr>
<tr>
<td>3</td>
<td>Generate code</td>
<td>47</td>
</tr>
<tr>
<td>4</td>
<td>Edit generated source</td>
<td>47</td>
</tr>
<tr>
<td>5</td>
<td>Reverse modified source code</td>
<td>48</td>
</tr>
</tbody>
</table>

### INTEGRATION WITH CVS

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Define main CVS options</td>
<td>49</td>
</tr>
<tr>
<td>2</td>
<td>Check out from scratch a new module on your disk</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>Add project to CVS</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>Commit project to CVS after making changes</td>
<td>51</td>
</tr>
<tr>
<td>5</td>
<td>Update CVS project</td>
<td>51</td>
</tr>
</tbody>
</table>

### USING TEAMWORK SERVER

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Login to the teamwork server</td>
<td>52</td>
</tr>
<tr>
<td>2</td>
<td>Create teamwork server users</td>
<td>53</td>
</tr>
<tr>
<td>3</td>
<td>Create teamwork project</td>
<td>54</td>
</tr>
</tbody>
</table>

#### Editing Teamwork Projects

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Login to the teamwork server</td>
<td>54</td>
</tr>
<tr>
<td>2</td>
<td>Open teamwork project</td>
<td>55</td>
</tr>
<tr>
<td>3</td>
<td>Commit Project to Teamwork Server</td>
<td>55</td>
</tr>
<tr>
<td>4</td>
<td>Lock elements for editing</td>
<td>56</td>
</tr>
<tr>
<td>5</td>
<td>Update teamwork project</td>
<td>57</td>
</tr>
</tbody>
</table>

### XML SCHEMA MODEL CREATION

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Create XML Schema Diagram Model</td>
<td>58</td>
</tr>
<tr>
<td>2</td>
<td>Create Model Types</td>
<td>59</td>
</tr>
<tr>
<td>3</td>
<td>Specify Sequence Order</td>
<td>60</td>
</tr>
<tr>
<td>4</td>
<td>Code Generation from Schema Model</td>
<td>62</td>
</tr>
</tbody>
</table>
STEP #1 Create a class diagram

1. Create a new project in MagicDraw. From the File menu, choose New Project and then select the Blank Project icon. Name the project or leave it untitled by default and click OK.
2. In the diagrams toolbar, click on the **Class Diagram** button. The **Create Diagram** dialog box appears.

3. In the **Type Class Diagram Name** field, enter the name of the new class diagram or leave the default name.

4. From the package tree, select the package where to contain the diagram (default is **Data** or create the new one by clicking the **Create Owner** button and choosing **Package** from the list.)
STEP #2 Create a new class element in diagram

1. In the diagram toolbar, click the Class button . Click again on the diagram pane in order to place the class. Type the name for a class. Click anywhere on the diagram pane to close the class title line.
2. The class that you see on the diagram pane is only the class symbol. All the class data are contained in the Browser. You can see that the class *Customer* has been created in the browser when you have drawn it on the diagram pane.

3. Double click on a class. The **Class Specification** dialog box appears. It contains various features of a class. For more detailed information about this dialog box, see MagicDraw Help.
4. Select the **Documentation/Hyperlinks** group, add some documentation about this class. Click **Close**.

---

**STEP #3 Create package element in diagram**

1. On the diagram tool bar, click the **Package** button.
2. Place the package on the diagram pane and name it *System*. Drag any corner of a package in order to reach its desired size.
3. Move a class into the package. A blue border around the package appears when the class is dragged into it and it shows that the class is included into package.
3. Right-click on a class to access its shortcut menu. Choose the **Select in Containment Tree** command.

4. The class name becomes marked by dark blue fill in the Browser tree.
STEP #4 Create new class element from browser

1. Right-click the System package symbol in the Browser tree. From the shortcut menu, choose the New Element option and then Class.

2. A new class in the package will appear in the Browser tree. Name the class Ticket.

3. Select the class in Browser and drag it onto the Diagram pane into the package.

STEP #5 Draw relationships

1. In the diagram toolbar, click on the Association button \( \text{Association} \). Connect the two classes with this path. If it is allowed to draw the link from one element to another, a blue border appears around it.

NOTE: You can draw some paths by clicking handler buttons on the class symbol.

2. To change the style of the path, click on special buttons


NOTE: For more detailed information about this dialog box, see MagicDraw Help.
4. In the Name field, enter the path name *owns*. Click Close.
5. Drag the association name link around and leave it on the diagram pane. Select the link. From the main toolbar, using a special **Reset Labels Positions** button, put the path's name into place.

**NOTE:** The same action can be performed using handler on the link.
STEP #6 Presentation options

1. Click on a class to select it. On the toolbar, go to the symbol editing buttons
   ![Symbol Editing Buttons]
   and click the Fill Color. Choose a color and click on it. The class fill color has been changed.

   **Note:** If not all symbol editing buttons are displayed on the diagram toolbar, from the toolbar shortcut menu, select the Expert Mode check box.

2. Click on Text Color. Choose a color and click on it. The class text color has been changed.
3. Click on an association to select it. Go to symbol editing toolbar and click Line Color. Choose a color and click on it. The association line color has been changed.

STEP #7 Browser options

1. In the left down Browser, click on the Documentation tab. The Documentation window will appear. Select a class containing the documentation. The documentation of a class will appear in the Documentation window.

   ![Documentation Window]

2. Click on the Zoom tab. The Zoom window will appear.

   ![Zoom Window]
3. Zoom in/out the picture by dragging the slider of the sliding scale to over/under 100% or zoom in/out the picture by dragging the blue border around the diagram in Zoom window.

**NOTE:** To add slider to the Zoom tab, from the Options main menu, choose Environment and in the opened dialog box, Browser group, select the Show diagram zoom slider check box.
SEQUENCE DIAGRAM CREATION

This guide contains step-by-step instructions, showing how to create a sequence diagram.

For creating a sequence diagram, an example of the Magic Test system will be given.

STEP #1 Create a Robustness diagram

1. Create a new project.
2. In the Browser tree, from the Data package shortcut menu, choose New Diagram > Custom Diagrams > Robustness Diagram. Name the Robustness diagram as Magic Test.
3. In the diagram draw elements, displayed in the Figure 1 on page 15.

![Figure 1 -- The Design robustness diagram](image)

STEP #2 Create sequence diagram

1. In the Browser tree, from the Data package shortcut menu, choose New Diagram > Sequence Diagram.
2. Name diagram as Change Password.

STEP #3 Create lifelines

i. Try below listed lifelines creation ways:
   - Drag and drop the Instructor actor from the Browser to the diagram pane (Note: you can also perform this action for multiple selection).
   - Draw lifeline from the diagram toolbar:
     1.) In the sequence diagram toolbar, click the Lifeline button .
     2.) Click on the diagram pane in order to draw a shape.
     3.) From the lifeline shortcut menu select Type drop down list and select lifeline type Instructor.
ii. In the same way in the sequence diagram create lifelines for the LoginDialog, SystemAccessManager, ProfileManager, InstructorProfile elements.

iii. You can display Image instead of lifeline shape. To change the lifeline representation on the diagram pane:

1. On the diagram pane from the lifeline shortcut menu select Symbol(s) Properties. The Properties dialog opens.
2. Change the Show Stereotypes property to Shape Image.
3. Change the same property for the rest of lifelines.
STEP #4 Link messages and alternative fragment

i. Create Instructor message to LoginDialog:
   1. Click the Message button and draw message from Instructor to LoginDialog.
   2. Type in message name as Type in new password twice.

ii. Create LoginDialog message to itself:
   1. Click the Message to self button and click on LoginDialog lifeline.
   2. Click on created message name and click the green balloon on the right - the Operation dialog will appear.
   3. Type in operation name as isPasswordsEquals and set Type value to boolean. Close dialog box.
   4. Click on operation name on the message and press Spacebar to edit text field. Edit message name to match equals = isPasswordEquals().

iii. Create alternative fragment.
   1. Click the Alternatives button and draw alternatives fragment from LoginDialog to InstructorProfile.
   2. Click on empty brackets and type in equals = true.

![Image of sequence diagram]

Figure 5 -- The Change Password sequence diagram

iv. Create changePassword() call message from LoginDialog to SystemAccessManager.
   1. Click the Call Message button and draw message from LoginDialog to SystemAccessManager.
   2. Create new operation changePassword() with type boolean.
v. Create the rest of the messages:

1. Repeat the first step to create `changePassword()` call message from `SystemAccessManager` to `ProfileManager` and `setPassword()` call message from `ProfileManager` to `InstructorProfile`.
2. Create reply messages to return changed variable to `LoginDialog`.
3. Create `showPasswordChangeMessage` and `showErrorMessage`.

![Figure 6 -- The Change Password sequence diagram](image-url)
JAVA REVERSE TO SEQUENCE DIAGRAM

NOTE: This functionality is available in Enterprise edition only.

A particular java source code file will be used to show how java reverse to sequence diagram is performed - extracted src.zip file from java home directory.

STEP #1 Create code engineering set

1. Create a new project in MagicDraw.
2. In the Browser tree, from the Code Engineering Sets shortcut menu, choose New. Type the name of the java set in Browser.
3. From the newly created set shortcut menu, choose Edit. The Round Trip Set dialog box appears.
4. In the Add Files tree, select java file and add it to the Set list. Click OK. Class source files is added to the java set in Browser.
5. From the java source file shortcut menu, choose Reverse, define reverse options and click OK. A new model with packages and classes will be created in the project.
STEP #2 Create sequence diagram from java source

1. In the Code Engineering Sets tree, expand the java file structure and from the class method shortcut menu, choose the Reverse Implementation command.
2. The Sequence Diagram from Java Source Wizard appears.

- The same wizard can be opened from the Diagram Wizards main menu by choosing Diagram Wizards command, from the Tools main menu, choosing Model Visualizer, or from the method shortcut menu in the Browser, choosing the Reverse Implementation command.

3. In the Step 1, specify the name of the sequence diagram and select the package, where this diagram will be created. By default, name of the class and method will be given to the sequence diagram. For example: Class.Method implementation. Click Next.

4. In the Step 2, select method (operation), which will be displayed in the sequence diagram. Specify the Java source file, if this file cannot be found automatically. Click Next.
5. In the Step 3, select other referenced classes, which you want to be displayed in the diagram.

Click **Next**.
Select the **Analyze and split long expressions in diagram** check box if expression contains calls and cannot be displayed as call message. Then every call will be shown as separate call message with temporary variable initialization.

Select the **Create reply messages** check box, if you want to display reply message for every call message.

Select the **Wrap messages text** check box and specify the maximum message text length in pixels, to wrap longer message.
6. In the Step 4, specify the presentation style for elements in created diagram.
7. Click Finish. The sequence diagram is created.

**STEP #3 Extend sequence diagram by method**

1. In the diagram pane, select method (message), which you want to be displayed more detail including referenced classes and other defining methods.
2. From the message shortcut menu, choose **Reverse Implementation** command.

3. The **Sequence Diagram from Java Source Wizard** appears. Following instruction in Step 2, define wizard options and click **Finish**. Additional new elements, which define the selected method, are added to the sequence diagram.
APPLYING DIFFERENT COLORS

Diagrams may have a set of different types of elements. For example, in the class diagram you may draw packages and classes. All those elements can be colored in the same color that you chose for your views. To make diagrams more clear, you may color elements of different types with their own color.

An example of two implementation diagrams, created in one project, will be used in this tutorial.

Implementation diagram, called Workstations:

This example was taken from "UML Toolkit" by Hans-Erik Eriksson and Magnus Penker
Implementation diagram, called Java Files:

```
<<component>>

  home.html

  <<component>>
  animlogo.java

  <<component>>
  animator.java

  <<component>>
  animlogo.doc

  <<component>>
  animator.doc
```

This example shows a typical dependency among few components: html file with embedded Java applet and software documentation files.

**STEP #1 Select all analogous shapes**

1. In the Workstations diagram, press and keep holding the ALT key.
2. On the Diagram pane, click the component instance Client Program element symbol. All analogous shapes are selected.

**STEP #2 Open the Properties dialog box**

1. When you have components selected, right-click the diagram and from the shape shortcut menu, choose Symbol(s) Properties.
2. The **Properties** dialog box appears:
STEP #3 Open the color dialog box

1. In the Properties dialog box, the Fill Color field, click the ‘...’ button. The Color dialog box appears:

![Color dialog box](image)

2. Choose the color you wish to be assigned to the element. Click OK.

3. In the Properties dialog box, click OK. The colors of the selected shapes will be changed.

**NOTE**
If you want to make the selected color as default for the newly created shapes, in the Properties dialog box select the Make Default check box.
STEP #4 Change properties using Project Options dialog box

1. From the Options main menu, choose Project. The Project Options dialog box appears.

   ![Project Options dialog box](image)

2. In the General Project Options tree, expand the Shapes section and select the Component element.

STEP #5 Extend properties by implementation diagram

1. From the Component shortcut menu, choose Implementation diagram. Style properties that may be applied to component element only in implementation diagram, appears.
2. Change the Fill Color and Pen Color properties values, by clicking the '...' button. The Color dialog box appears. Submit changes.

3. In the Project Options dialog box click the Apply button. The Select Diagrams dialog box appears.

NOTE! Only Implementation diagrams are displayed in this list, because properties were extended by this type of diagram only.
4. Select the Java Files diagram and click OK. The Select Properties to Apply dialog box appears. By default, all properties are added to the Selected list.

![Select Properties to Apply dialog box]

*NOTE!* To apply not all properties, use the '<' button and move irrelevant properties to the All list. Only selected properties existing in the Selected list will be applied.

5. To submit changes and close dialog boxes, click OK. Changes were made to component elements but only in Java Files implementation diagram.

![Diagram showing dependency among components]

This example shows a typical dependency among few components: html file with embedded Java applet and software documentation files.

*NOTE!* Component element color in Workstations diagram remains the same without any changes because only Java Files diagram was selected in the Select Diagrams dialog box to apply changed colors.
APPLYING IMAGES TO STEREOTYPES

STEP#1 Organize Stereotypes in the Browser, Model Extensions Tree

1. In the Browser, Model Extensions Tree tab, there are all project stereotypes listed. Use buttons to group them by profile or metaclass.
2. You may choose an existing stereotype from the tree or add a new one from the metaclass shortcut menu, by choosing **New** and then **Stereotype**. The **Select Package** dialog box opens:

3. Select package, where newly created stereotype will be saved and click **OK**. Type the name of the stereotype in the Browser.

4. Press **Enter**, double-click or from the stereotype shortcut menu, choose **Specification**. The **Stereotype Specification** dialog box opens.

**STEP#3 Choose an icon for the stereotype**

1. Click the “...” button near the **Icon** field. The **Open** dialog box appears.

2. Select the image to be attached to the stereotype, and click **Open**. An icon appears in the **Icon** field:

---

**NOTE:** GIF, JPG, JPEG, SVG, PNG, and MDICO files are attachable.
3. In two opened dialog boxes click Close and OK.

**STEP#4 Applied Stereotype Property**

1. Right-click the class shape and from the shortcut menu, choose Specification. The Class Specification dialog box opens.
2. Choose the Applied Stereotype property and click the “...” button to apply a newly created stereotype from the list. Click Apply. Click Close. The icon is added on the class element symbol.

**STEP#5 Suppress attributes and operations**

1. Select a class and from its shortcut menu, choose Presentation Options.
2. Select the Suppress Attributes check box.
3. Go again to the Presentation Options menu and select the Suppress Operations check box.
4. A new notation for your element appears on a diagram:

   ![Diagram showing class symbol with suppress attributes and operations]

   ```
   <<hardware actor>>
   ```

**TIP** You can suppress attributes and operations using a smart manipulation feature. Select a class and click on the buttons with ‘minus’ sign.
DATA PARTITIONING

NOTE This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

For starting work with a data partitioning, from the MagicDraw installation directory, samples folder, case studies subfolder, open the Inventory Control System.xml.zip project.

STEP #1 Export a new Module

1. In the Browser Tree, from the Data package shortcut menu, choose New Element and then Package. Name this package directly in the Browser as Inventory Control System.

2. Select the Static View package and drag it to the newly created package. A class diagram with all contained elements now is an inner element of the Inventory Control System package.

3. From the Inventory Control System package shortcut menu, choose Modules, and then choose Export Module. The Export Module dialog box appears.
4. The **Selected packages** list shows, which package will be exported as a module. Click **OK**.

**NOTE**
If you want to check for dependencies and module you are exporting depends on external project elements, the **Package Dependencies** dialog box appears. Using this dialog box, solve errors. Then you will be able to export module.

5. The **Save** dialog box appears. Select the destination directory, where you want to save the project module, type the file name **Static View** and click **Save**. The name of the module appears in the Browser, near the package name.

**NOTE**
The exported module is not editable after exportation. If you want to make some changes, from the module shortcut menu, choose **Modules**, and then choose **Import Module**. The module will be merged with a project model.

**STEP #2 Open module as a new project**

1. In the Browser Tree, from the **Inventory Control System** package shortcut menu, choose **Modules**, and then choose **Open Module As Project**. A new project is opened.

2. In the **Inventory Control System** package, the **Static View** subpackage, select the **Top Level class diagram** and open it from the diagrams shortcut menu choosing **Open**. The Diagram pane with element symbols appears.

3. Add two attributes to the class **Customer**. Select this class in the Diagram pane. From the class shortcut menu, select the **Presentation Options** command and clear the **Suppress Attributes**
check box. In the Diagram pane, press CTRL+ALT+A. A new attribute is added to the class. Type the name for the attribute Name. Add one more attribute, name it Address.

4. Save the changes, made to this project.

STEP #3 Reload the module

1. From the File main menu, choose Opened Projects, and then choose Inventory Control System.mdzip. The previous project will be opened.

2. In the Browser, select the Inventory Control System module. From its shortcut menu, choose Modules, and then choose Reload Module.

3. To see changes were applied to the module, open the Top Level class diagram. The class Customer has two attributes.
PERFORMING ROUND TRIP

**NOTE**

Code Engineering is available in Professional, Architect and Enterprise editions only.

Three classes, presenting the system for controlling the luggage storage in an aircraft will be used in this tutorial:

- **Luggage** – represents a piece of luggage;
- **LuggageCompartment** – represents the luggage compartment of an aircraft during a flight;
- **FlightSegment** – represents a flight segment.

**STEP#1 Add elements to the newly created code engineering set**

1. In the Browser tree, from the Code engineering sets shortcut menu, choose New, and then Java. Type the name of a new set in Browser. A new java set is created.

2. From this newly created set shortcut menu, choose Edit. The Round Trip Set dialog box appears.
3. Open the **Add Data from Model** tab and from the **All Data** list to the **Set** list, add the selected classes.

![Round Trip Set](image)

4. Click **OK**.

**STEP#2 Generate code**

1. From the code engineering set shortcut menu, choose **Generate**. The **Code Generation Options** dialog box appears.

2. Define options of the generation and click **OK**. (For more detailed information about how to generate code, see Tutorial “Code Generation and Reverse”).

The java code is generated:

```java
/**
 * @(#) Luggage.java
 */
public class Luggage
{
    private float weight;
    /**
     * Return the weight of this piece of luggage in kg.
     */
    protected float getWeight( )
    {
        return 0;
    }
}

/**
 * @(#) LuggageCompartment.java
```
STEP#3 Add a new LuggageException class to the model

When coding according these constraints, it is learned that exceptions need to be provided in case luggage checking fails. To account for this, a new class called LuggageException is introduced:

1. Create a new java code file and write the source code for a new class LuggageException addition:

```java
/**
 * An instance of this is thrown when a requested operation on luggage
 * may not be performed.
 */
public class LuggageException extends Exception {
    /**
     * Constructor
     */
    public LuggageException(String msg) { super(msg); }
}
```

2. In the MagicDraw, select the created New (Java) set and from its shortcut menu, choose Edit. The Round Trip Set dialog box appears.
3. In the **Add Files** tab, select the created *LuggageException.java* code file and add this file to the **Set** files list.

![Round Trip Set dialog box](image)

4. Click **OK**. The code file is added to the set.

**STEP#4 Create class element in the project using reverse**

1. From the code engineering set shortcut menu choose **Reverse**. The **Reverse Options** dialog appears:

![Reverse Options dialog box](image)
2. Define options of the reverse and click **OK**. A new class element is added to the project and class symbol appears on the Diagram pane.

### STEP#5 Separate luggage into two types

Next, a new requirement for the software is introduced. Luggage must be separated into two types: normal and dangerous.

1. According to these new requirements, create two new classes, `DangerousLuggage` and `NormalLuggage` (subclasses of `Luggage`) in the model. Link them with `Luggage` class by Generalization relationship.

### Additional Code for Luggage Class

```java
/**
 * @(#) Luggage.java
 */
public class Luggage {
    private float weight;
    /**
     * Return the weight of this piece of luggage in kg.
     */
    protected float getWeight() {
        return 0;
    }
}
public class NormalLuggage extends Luggage {
}
public class DangerousLuggage extends Luggage {
}
```
NOTE Code Engineering is available in Professional, Architect and Enterprise editions only.

STEP #1 Create a new code engineering set
1. Right-click the Code Engineering Sets in the Browser tree. From the shortcut menu choose New and then language.
2. Type the name for set directly in the Browser. A new java set is created.

STEP #2 Add data you wish to generate
1. From the created code engineering set shortcut menu, choose Edit. The Message dialog box appears. Click OK.
2. Open the Add Data from Model tab in the Round Trip Set dialog box. From the All Data list to the Set list, add classes, interfaces, packages, or components (we have two classes already created in the model). Click OK.

TIP! To add data for generation, you can drag-and-drop elements to the created set in the Browser tree.
STEP #3 Generate code

1. From the code engineering set shortcut menu, choose Generate. The Code Generation Options dialog box appears.

2. Define options of the generation:
   - If the classes for which you wish to generate code were not reversed, neither the Reverse before generation option, nor the If element deleted from model options, will influence the generated code.
   - If classes are reversed and you click the Reverse before generation check box, the file will be reversed once more before generation.
   - If there are any deleted elements in the model since the last reverse, the If element deleted from model options group box determine the code generation behavior.
   - To use spaces instead of tabs in the code file, click the Use spaces in place of tabs check box.

3. After you set these options, click OK.

STEP #4 Edit generated source

1. From the code engineering set shortcut menu, choose Edit Source.

NOTE: In the Options main menu, choose Environment, select Launchers group and click the “...” button to set a default tool for editing source code text.

2. The generated source will be opened for editing. Edit source and save changes.
STEP #5 Reverse modified source code

1. From the modified code engineering set shortcut menu, choose Reverse. The Reverse Options dialog appears.

2. Define options of the reverse:
   - To create attributes instead of associations, choose the Attributes option button. Class may have a collection of other classes and use Java generics (e.g. List<String>). If the Resolve collection generics check box is selected, types of collection will be resolved (property type will be not collection, but real type). Select the check box to reset an already created fields.
   - Select the Visualize reversed model check box and choose how the reversed data will be applied. The possible choices are: create a new diagram by launching Model Visualizer and starting Diagram Wizards, just create a new class diagram, or place data in an active class diagram.
   - To merge all the differences between the elements in the model and the ones in the file, click the Merge model and code option button. The model elements will be updated by code. Elements that do not exist in the code will not be removed from the model. To ensure that the elements in the model will be exactly the same as in the source file after the reverse, click the Change model according to code option button.
   - To create dependencies between classifiers and/or packages in model, select the appropriate check boxes.

3. Click OK.

NOTE! If you have only a source code and want to create a model, from the newly created code engineering set shortcut menu, choose Edit. Add files you wish to reverse from your hard disk to the set, in the Round Trip Set dialog box, the Add Files tab.

TIP! You may reverse the desired files using the Quick Reverse option: from the Tools menu, choose Quick Reverse, select a language and select a files you want to reverse.
INTEGRATION WITH CVS

NOTE Integration with CVS is available in Professional, Architect and Enterprise editions only.

STEP #1 Define main CVS options

1. From the Options main menu, choose Environment. The Environment Options dialog box appears.
2. Choose the CVS pane.

3. Define options for CVS:
   - If you want to add project to the CVS server every time after saving, select the **Add project to CVS after saving** check box.
   - If you want to commit project to the CVS server every time after saving, select the **Commit project to CVS after saving** check box.
   - If you want to update project when there is a new version on the server, select the **Update project from CVS before loading** check box.

4. Specify the path where the passwords storing file `.cvspass` is located and path of local folder where the module will be saved on checkout action.

   **NOTE** Grouped directories, that contain subfolders with the together related files are called modules.

5. Click OK.
INTEGRATION WITH CVS

STEP #2 Check out from scratch a new module on your disk

1. From the Tools menu, choose CVS, and then choose Checkout Module. The Checkout Module dialog box appears.

   ![Checkout Module dialog box](image)

2. In the CVS Root field, enter the path of the CVS repository, which stores a complete copy of all the files and directories that are under version control.

3. In the Module name field, enter the path where the module is stored.

4. In the Local folder field enter the path of the local folder where the module will be saved on checkout action.
   - You may select the Prune empty directories check box, if you want automatically remove empty folders when you update a module.
   - For checking out only the last project version, select the Reset sticky tags check box.
   - If you don’t want to check out the subdirectories, select the Non-recursive check box.

5. Click OK.

STEP #3 Add project to CVS

1. Create a new MagicDraw project.

2. Save it in already checked out directory (how to check out CVS directory, see CVS help.)

3. From the Tools menu, choose CVS, and then choose the Add Project to CVS command. The Add Project to CVS dialog box appears.

4. NOTE: If the Add Project to CVS after Saving command in the Environment Options dialog box, CVS pane is selected, the Add Project to CVS dialog box appears every time when the new to CVS project is being saved.

   ![Add Project to CVS dialog box](image)

   - If the project is saved as XML.ZIP format, select the Add as binary data [-kb] check box.
5. If needed, type the Log message.
6. Click OK.

### STEP #4 Commit project to CVS after making changes

1. From the Tools menu, choose CVS, and then choose Commit Project to CVS. The Commit Project to CVS dialog box appears.

   **NOTE:** Project is committed to the CVS server every time after saving it, if the option Commit Project to CVS After Saving is selected in the Environment Options dialog box, CVS pane.

2. In the Log message field, type the message about the project that can be important for other users.
3. Click OK.

### STEP #5 Update CVS project

1. From the Tools menu, choose CVS, and then choose Update CVS Project. The Update CVS Project dialog box appears.

   - If the Reset sticky tags check box is selected, the last version of the project that is on the server will be updated. If no check box is selected, you will be informed that project is already opened but you can reload it and loose changes.
   - Type the tag or version number you want to update to the Retrieve rev./tag/branch field (it becomes enabled when the Reset sticky tags check box is cleared). You may click the ‘…’ button and in the Revision dialog box choose the desirable version.

2. Click OK.
Getting Started with Teamwork Server

STEP #1 login to the teamwork server

1. From the Teamwork main menu, choose Login. The Login dialog box appears.

   ![Login Dialog Box]

   - **Login**: Administrator
   - **Password**: ********
   - **Teamwork server name**: kolbis:1130
   - **Auto login to server**: deselected

2. To login to the server as administrator, in the Login and Password fields, type “Administrator”. In the Teamwork Server Name text box, enter the server name and port.

3. If you want every time when starting MagicDraw to connect to the server, select the Auto Login to Server check box.

4. Click OK. On the status bar the note "Logged in as: Administrator" appears. The Administrator has administrator’s permissions (all available permissions) and is allowed to create new users.

   ![Logged in as: Administrator]
STEP #2 create teamwork server users

1. From the Teamwork main menu, choose Users. The Edit Users dialog box appears.

2. To add a new user, click the Add button. The Edit User dialog box appears.

3. Enter the login, name, and password for the user. Click OK. A new user is created.

4. In the Edit Users dialog box, Permissions list, set specific permissions for the user by selecting corresponding check boxes.

5. The More>> button allows to set permissions for the users on particular projects. If there are teamwork projects available, you can set permissions for the users on these projects.

6. When all users are added and permissions are set, click Close.
STEP #3 create teamwork project

1. If there are no teamwork projects created or you need a new teamwork project, from the Teamwork main menu, select Projects. The Edit Projects dialog box appears.

2. Select from the existing teamwork projects or add a new project using the Add button. When adding a new project, the Project Name dialog box appears. Enter the name for a new project or leave the default one. Click OK.

3. In the Permissions list, set project permissions for the users. Click Close.

Editing Teamwork Projects

STEP #1 Login to the teamwork server

1. From the Teamwork main menu, choose Login. The Login dialog box appears.
2. Enter the user login, password, and teamwork server name. Click OK.
STEP #2 Open teamwork project

1. From the Teamwork main menu, open a teamwork project by choosing Open Teamwork Project. The Open Teamwork Project dialog box appears. If no projects are available, you can add a new one by clicking the Add button. Click Open.

2. Look at the Browser tree. The items in the Browser tree are blank and they can be read, but no editing is allowed. A user may add new elements and edit only those elements that he/she has locked for editing.

STEP #3 Commit Project to Teamwork Server

1. Create a new class diagram and name it. Draw any elements and relationships in the class diagram.
2. Commit the diagram to the Teamwork Server. From the Teamwork menu, choose Commit Project (you can use CTRL+K shortcut key instead). The Commit Settings dialog box appears. If needed, you may enter any commit comments or tags.

NOTE! If you have some elements locked for editing, the list of them is displayed in the Locked Elements tab. Select elements if you want to unlock them after committing project.

If your project uses some local modules, list of them is displayed in the Local Modules tab.
The Locked Modules tab displays list of modules, which are locked in this project.
3. Click Commit.
4. After committing the diagram for the first time, it becomes read-only and it is blank in the Browser tree.
STEP #4 Lock elements for editing

1. In the Browser tree, from the class diagram shortcut menu, choose the Lock for Edit command and then Lock for Edit.

2. In the Browser tree, appears a name of a user who has locked the class diagram for editing.

3. To lock any class from a diagram for edit, from its shortcut menu, choose the Lock Class for Edit command and then Lock for Edit.
4. Perform some changes for a class. Click CTRL+K to commit changes. The diagram that was edited once again does not become read-only, and it is left editable for the user who has locked it.

5. In order to unlock the diagram, in the Browser tree, from the class diagram shortcut menu, choose the Unlock command and then Unlock.

NOTE! If you select the Discard Changes command, you may unlock a diagram and discard any changes that were made by the user who has locked the diagram.

6. In the same way classes and other elements of a diagram are unlocked.

STEP #5 update teamwork project

NOTE! You have to update a teamwork project from time to time. It is necessary to receive changes made by other users who have edited a teamwork project.

- From the Teamwork main menu, choose Update Project. All the changes made by other users will appear in the project version that you have.
This guide contains step-by-step instructions, showing how to create a basic XML schema model (and then generate the schema from this model) from scratch.

**STEP #1 Create XML Schema Diagram Model**

1. Create a new project.
2. In the Browser tree, from the **Data** package shortcut menu, choose **New Diagram** and then **XML Schema**. Specify the name of this diagram directly in the Browser - e.g. **Mail Tracking System** as in our example.
   - You may also create XML Schema diagram from the Diagram toolbar menu by clicking the **XML Schema Diagram** button or from the **Diagrams** main menu, choosing **XML Schema Diagram**.
3. To create a namespace in this diagram, on the diagram elements toolbar, click the XSD-Namespace button (or press **P**) and then click anywhere on the diagram pane. Type the name (**http://system/** in our example) for the namespace. This namespace will be the main container for the types of the schema.
4. To speed up further development, in the Browser tree, drag created XML Schema diagram to the namespace package.
5. Add the **http://www.w3.org/2001/XMLSchema** namespace package to the diagram - expand the XML Schema Profile, select this package and drag it into the diagram pane.
6. Draw the XSDSchema element in diagram and link both namespaces with xmlns relationship.
7. Specify names for relationships. These names will be mapped to the namespace prefix in the resulting .xsd file. Double-click on the relation to open the **Element Import Specification** dialog box and in the **Alias** field type the name, which will appear on diagram pane. You can leave one of these relationships without name - this will be the default namespace of the schema.
8. Double click the schema element and in the opened **Specification** dialog, click on the **Tags** tab. Expand the **<<XSDSchema>>** branch and select **targetNamespace** tag. Click the **Create**
Value button. In the Select Elements dialog, select http://system/ namespace and add it to the list by clicking Add.

9. Such a model will result (after code generation) in schema like this:

   <schema targetNamespace=http://system/
           xmlns=http://www.w3.org/2001/XMLSchema
           xmlns:my=http://system/>

STEP #2 Create Model Types

1. In the elements toolbar, click on the XSDcomplexType button and draw new type on the diagram pane. Name it as mailType.

2. Right-click on this type and from the shortcut menu, choose XML Schema and then New XSDAttribute.

3. Type the id name for new attribute. Double-click to open the Property Specification dialog box and assign integer type in the Type field.

4. Select mailType element and from the shortcut menu, choose Stereotype. Apply <<XSDFsequence>> stereotype for this element. This will enable inner element creation for complex type.
5. Right-click on the type again and from the shortcut menu, choose XML Schema and then New XSDElement. Name this element as e-mail. In the same manner, add another element, named envelope.

6. Draw another XSDcomplexType element on the diagram and name it as mailsType.

7. When choosing types for elements, you may use either primitive typed for XML schemas from the XML Schema Profile or your modeled types. Add new XSDElement type to mailsType and double-click to open the Property Specification dialog box. In the Type field assign previously created mailType element as a type.

8. It is possible to put the name of the element on the appropriate end of the association relation. Draw composition between mailsType and mailType, add a name and assign <<XSDElement>> stereotype to the association end. From the end shortcut menu, choose Edit Name and then on the diagram pane, type mail.

9. After code generation, this model will result in the following source code:

```xml
<complexType name="mailsType">
    <sequence>
        <element name="type" type="my:mailType"/>
        <element name="mail" type="my:mailType"/>
    </sequence>
</complexType>
<complexType name="mailType">
    <sequence>
        <element name="e-mail"/>
        <element name="envelope"/>
    </sequence>
    <attribute name="id" type="integer"/>
</complexType>
```

**STEP #3 Specify Sequence Order**

1. Create two more XSDcomplexType elements attachmentType and bodyType. Add appropriate XSDElements to them - attachment and text.
2. Draw aggregation link from the mailType element to these two newly created elements.
3. Double-click the **bodyType** element to open its Specification dialog box. Select the **Tags** tab and in the `<XSDSequence>` branch add new tagged value “1” for sequenceOrder tag.

4. Repeat the same action for **attachmentType** element, just add tagged value “2”.

5. Such model after all these actions is created:

6. Such source will be generated from this example:

```xml
<complexType name="attachmentType">
  <sequence>
    <element name="attachment"/>
  </sequence>
</complexType>
<complexType name="bodyType">
  <sequence>
    <element name="text"/>
  </sequence>
</complexType>
```
STEP #4 Code Generation from Schema Model

1. Right-click the Code Engineering Sets in the Browser tree. From the shortcut menu choose New and then XML Schema language. Type the name for set directly in the Browser. A new XML Schema set is created.

2. From the created code engineering set shortcut menu, choose Edit. Open the Add Data from Model tab in the Round Trip Set dialog box. From the All Data list to the Set list, add XML Schema elements for code generation. Click OK.

3. From the code engineering set shortcut menu, choose Generate. The Code Generation Options dialog box opens:

4. Specify Output Directory for source code and click OK. The schema.xsd file containing XML Schema code will be generated there.