



Apply a pattern

Contents

Apply a pattern	1	Lesson 4: Create an instance of the pattern	4
Introduction: Apply a pattern.	1	Lesson 5: Bind new elements to pattern instances . . .	5
Lesson 1: Create the model environment	2	Lesson 6: Bind existing elements as arguments . . .	6
Lesson 2: Import sample patterns	3	Lesson 7: Reapply the pattern	7
Lesson 3: Select the pattern	4	Summary: Apply a pattern	7

Apply a pattern

This tutorial guides you through the steps needed to apply a simple pattern.

Pattern authors can use the IBM® Rational® pattern capability to design patterns from the most simple to the very complex. Patterns are software solutions that solve a recurring problem within a given context. The limitations of a pattern are determined by the pattern design and the intent of the pattern author. You can share patterns within a project, within a company, or across many companies.

Learning objectives

The objective of this tutorial is to learn how to apply patterns.

Time required

30 minutes

Related information

[Applying patterns](#)

[Sample: Patterns to apply](#)

[View the PDF version](#)

Introduction: Apply a pattern

Using the Pattern Explorer view, you will learn how to create an instance of a pattern. Then, you are instructed to specify the UML type of arguments for the pattern's template parameters using four different methods.

This tutorial might require some optionally installable components. To ensure that you have installed the appropriate optional components, see the System requirements list.

This tutorial uses samples from the Samples Gallery to set up the required apply pattern environment and load some patterns. It is recommended that you complete each exercise in order.

Learning objectives

The tutorial is divided into seven exercises. The beginning exercises help you to set up the environment needed to apply patterns. After the environment is set, you will learn one way to instantiate a pattern. Next, you specify arguments for a pattern instance's parameters using four different methods. Last, you will make a final modification to one of the arguments and reapply the pattern. This modification helps you to see the power of including patterns in your software design.

In this tutorial, you will learn the following:

- Set up the environment needed to apply patterns
- How to instantiate a pattern
- How to specify arguments for a pattern instance's parameters using four different methods
- How to make a final modification to one of the arguments and reapply the pattern

Time required

This tutorial should take approximately 30 minutes to finish. If you explore other concepts related to this tutorial, it could take longer to complete.

Skill level

Advanced

Audience

The intended audience for this tutorial is developers.

System requirements

To complete this tutorial, you need to have the following tools and components installed:

- Pattern authoring

Prerequisites

To perform the steps in this tutorial, you should be familiar with the following concepts and tasks:

- Basic UML concepts and terminology such as binding, argument values, and collaborations
- Basic visual modeling functions such as creating a class in the Project Explorer view or in a diagram
- Some knowledge of programming concepts and terminology, preferably Java™
- How to navigate the Eclipse workbench
- Industry definitions and uses of design patterns

Lesson 1: Create the model environment

This lesson guides you through how to establish the environment you need to apply a pattern.

You apply patterns to models that contain UML elements so that you can modify these elements. To do this, you need an open UML project and a UML model. For this tutorial you also need an open freeform or class diagram. Although the diagram view is not essential, using it to apply patterns makes it easier to see the pattern results.

If you are an advanced user, you can create a UML project, a UML model, and a freeform or class diagram. For this exercise you need to add a UML class and a UML interface to the model. The interface must define one or more operations. Ensure that the Modeling perspective is open. An easier option is to perform the following instructions to create these requirements for you. Advanced users can review these instructions to ensure that their results are the same as the example.

To create the model environment:

1. Load the UML project and model
 - a. Import the simple UML project by clicking: Samples Gallery. The Samples Gallery window opens.
 - b. On the Simple UML model sample page, click **Import the sample**. The Sample Model window opens.
 - c. Click **Finish** to open the new project and model. The Project Explorer view opens.
 - d. Expand the project and double-click the model. The model and a freeform diagram open.
Identify the diagram view by the name of the model on its tab. A new diagram view is a blank surface used to add and manipulate model elements. In this example, you see the class, named AppFunction1, and the interface, named DoWork, are already displayed on to the freeform diagram. DoWork owns UML operations named doNothing and doSomething. The AppFunction class also owns two UML elements: an operation named Operation1 and an attribute named Attribute1.
2. Open the Modeling Perspective
 - a. Click **Window** → **Open Perspective** → **Other**.
 - b. In the Select Perspective window, select **Modeling Perspective** and click **OK**.

- 2 Apply a pattern

3. Adding a diagram view. In the Project Explorer view, do one of the following:
 - a. Right-click the model icon and click **Add Diagram** → **Class Diagram**
 - b. Right-click the model icon and click **Add Diagram** → **Freeform Diagram**.

Now that you know the basic requirements to create a pattern, you are ready to import the sample patterns. You will apply a sample pattern to elements in your new UML model.

Lesson 2: Import sample patterns

This lesson shows how to import the sample patterns. The samples are imported to a pattern samples repository and not to your workspace. Patterns only appear in your workspace if you select and apply them.

Before you begin, you must have created the UML project as detailed in Lesson 1: Creating the model environment.

You will select a sample pattern in this tutorial. After you complete this tutorial, you can get additional practice applying the other sample patterns. After import, the sample patterns remain listed in your Pattern Explorer view unless you choose to remove them.

Patterns are accessible in the Pattern Explorer view. You can use the Pattern Explorer view to organize, select, and apply patterns. You can see the pattern samples immediately after you import them, if the Pattern Explorer view is open.

If you would like to view this exercise before you perform the steps, click:

Show Me

To import sample patterns:

1. Open the Pattern Explorer view.
 - a. Click the Pattern Explorer button that appears in the bottom or right-hand Workbench frame to open the Pattern Explorer view. The Pattern Explorer view opens with a group of advanced patterns already installed: Design Patterns. You can apply any patterns that are listed in the Pattern Explorer view. If the Pattern Explorer is already open, skip this step.
 - b. Click the Pattern Explorer button again or click anywhere outside the Pattern Explorer view. The Pattern Explorer view disappears.
 - c. Repeat step one to open the Pattern Explorer view. Thus, you can quickly switch views and you may want to do this while you apply patterns.

If you do not see the Pattern Explorer button, then the Modeling perspective is not in focus. Return to exercise one to open the Modeling perspective, Exercise 1: Creating the model environment.
2. Import the pattern samples.
 - a. Check the Pattern Explorer view to see if the sample patterns group, Sample Patterns, has already been imported. If they are imported, skip performing the following import instructions but do read the instructions for general knowledge about the samples.
 - b. Open the Samples Gallery page by clicking: Samples Gallery. The Samples Gallery page opens.
 - c. Expand **Technology samples**, expand **Patterns**, and click **Patterns to apply**.
 - d. At the bottom of the Patterns to apply page, click **Import the sample**. The Sample Patterns window opens.
 - e. Accept the default project name and click **Finish**. The samples group, named Sample Patterns, appears in the Pattern Explorer view.
 - f. Expand **Sample Patterns** to see the subgroups organized into folders. Each folder represents a pattern.

The Sample Patterns is a group of pattern samples that you can use to practice applying patterns. You cannot apply a pattern from the Project Explorer view.

Lesson 3: Select the pattern

In this lesson, you locate the pattern in the Pattern Explorer view that you will apply to elements in the UML model.

Before you begin, you must complete Lesson 2: Importing sample patterns.

As a pattern applier, you can look at different types of pattern documentation to assist you in selecting and applying a pattern. You can locate the pattern documentation by using the Pattern Explorer view to examine each pattern. After you have selected a pattern, you can apply it. For this tutorial, you will apply the Interface pattern.

If you would like to view this exercise before you perform the steps, click: [Show Me](#)

To locate a pattern to apply:

1. In the Pattern Explorer view, expand the **Sample Patterns** group. Locate the **Interface** pattern.
2. Click the **Interface** pattern. A description of the pattern displays in the Pattern Explorer view's Short Description pane.
3. Click the **Overview** tab. A simple model of the relationship of the pattern elements reveals the potential use of the pattern.
4. Expand the **Interface** pattern to view the pattern's parameters. The pattern owns two parameters named Interface and Implementation.
5. Click each parameter and read the short description for each. The description explains the function that the parameter performs in the pattern.
6. Right-click the **Interface** pattern and click **Show Pattern Documentation**. The online Help opens with additional information about this pattern.

You can use the pattern documentation to understand the purpose of the pattern. Documentation varies according to what a pattern author chooses to provide.

You are ready to create an instance of the pattern.

Lesson 4: Create an instance of the pattern

This lesson explores a simple method of creating an instance of the Interface pattern on the class or freeform diagram.

Before you begin, you must complete Lesson 3: Selecting the pattern.

If you would like to view this exercise before you perform the steps, click: [Show Me](#)

When you apply a pattern, the first part of the sequence is creating the pattern instance. You can create more than one instance of a pattern and have them active at the same time. The easiest method to create an instance is to use the drag-and-drop method.

The pattern instance is a structure that is identifiable by the UML keyword Pattern Instance. Because the Interface pattern is based on a collaboration, it is contained by one of two unique user-selected shapes. Other types of patterns are package- or class-based, and these types of patterns mimic the shape of respective elements on the diagram view. The pattern's template parameter definitions are abbreviated in the instantiation. The UML type and the multiplicity for each template parameter are also visible in the instance to aid the pattern applier in selecting or creating appropriate arguments to bind to the pattern.

You can create more than one pattern instance for the same pattern in your workspace. To add the pattern instance to the model:

1. In the Pattern Explorer view, drag the Interface pattern to the freeform or class diagram. The pattern is instantiated and the resulting pattern instance appears on the diagram.
2. In the Project Explorer view, a pattern instance now appears as part of the same sample UML model.
3. Expand the pattern instance to see its contents. You will be able to see the changes that occur to the instance as you modify the pattern on the diagram in the next exercise.
4. To view the properties for any elements in the Project Explorer view or the diagram view, right-click the element and click **Properties**.

You are now ready to learn how to supply arguments for the pattern's parameters.

Lesson 5: Bind new elements to pattern instances

The lesson explores different methods of adding or binding arguments to the pattern instance.

Before you begin, you must complete Lesson 4: Creating an instance of the pattern.

If you would like to view this exercise before you perform the steps, click: [Show Me](#)

When binding occurs, the pattern instance's template parameters are replaced by the selected or newly created elements specified by the pattern applier.

Whether you select existing elements or create new elements as arguments depends on what you want the pattern to do. In this tutorial, different methods are explored so that you can learn about them.

On the freeform or class diagram, you can use the action bar to add elements. To see the action bar, move the cursor over the blank surface of the diagram editor and press the Space bar. You can hold the cursor over the template parameters in the pattern instance to display a smaller action bar with only the elements that are applicable to the UML type of the template parameter.

If the action bar disappears in a few seconds, tap the spacebar to re-display it.

To bind new elements to pattern instances:

1. Create a new element as an argument.
 - a. On the freeform or class diagram, hold the cursor over the pattern instance's Interface parameter. The action bar is displayed.
 - b. Click the interface icon on the action bar to create and bind an interface element to the template parameter. A binding symbol replaces the blank box next to the template parameter followed by the UML type of the element (interface), indicating that the new element is bound to the pattern instance.
 - c. In the Project Explorer view, note the addition of the new interface element in the UML model. The pattern instance's template binding structure also shows the Interface template parameter bound to an Interface element.
2. Create a custom-named element as an argument

Not all pattern templates may have this option because the pattern designer can suppress this option in design.

 - a. On the freeform or class diagram, hover the cursor over the Interface parameter in the pattern instance. On the action bar, the icons are available for selection indicating that the multiplicity for the template parameter allows the binding of another element.
 - b. Click the **Text** icon to the right of the interface icon on the action bar. A rectangular box appears next to the right-side of the template parameter.

- c. In the box, type `IMyInterface` to name the new element and click outside of the instance or press Enter to complete the name. A binding symbol replaces the blank box next to the template parameter followed by the UML type of the element (interface).
- d. Observe the changes to the template binding structure in the Project Explorer view. The binding structure now shows the Interface template parameter bound to an additional UML interface element named `IMyInterface`.

You are ready to learn two more methods of adding arguments to a pattern instance through binding new elements to pattern instances.

Lesson 6: Bind existing elements as arguments

The lesson explores two different ways to specify existing UML model elements as pattern template arguments.

If you would like to view this exercise before you perform the steps, click: [Show Me](#)

Instead of creating a new UML element, you can select existing elements by typing the name of an existing element in the pattern instance on the diagram view. Or you can drag an existing element from the Project Explorer view or the diagram view onto the pattern instance.

To bind existing elements as arguments, you can use different methods such as specifying the name of an existing element as argument or dragging an existing element as an argument.

1. To specify the name of an existing element as an argument:
 - a. On the freeform or class diagram, hold the cursor over the Implementation parameter in the pattern instance. The action bar is displayed and a **Class** icon and a **Text** icon are available.
 - b. Click the **Text** icon on in the action bar to type the name of an existing UML class element. A rectangular box appears next to the right-side of the template parameter.
 - c. In the box, type `AppFunction1` to specify the class from the sample model.
 - d. Click outside the instance or press Enter to complete the name. A binding symbol replaces the blank box next to the template parameter followed by the name of the element.
 - e. Observe the changes to the template binding in the Project Explorer view. The binding structure now shows the `AppFunction1` class bound to the Implementation template parameter.

The most important result is the two operations from the `DoWork` interface are now copied to the `AppFunction1` class.

2. To drag an existing element as an argument:
 - a. On the freeform or class diagram, drag and drop the `DoWork` interface onto the left side of the Interface row in the pattern instance.

Dragging an element onto a pattern instance will replace a previously bound element if the bound element was selected.
 - b. In the Project Explorer view, observe the addition of the `doNothing` and `doSomething` operations to the `AppFunction1` class resulting from the pattern binding.
 - c. In the Project Explorer view, expand the instance's template binding structure to see the addition of the new interface element. The Interface template parameter is bound to a third interface element named `DoWork`.

You are ready to reapply the pattern and observe its effects on the model elements.

Lesson 7: Reapply the pattern

The lesson explores reapplying the sample pattern after adding additional elements to an interface you will use as an argument.

If you would like to view this exercise before you perform the steps, click: [Show Me](#)

In lesson 6, you specified arguments for the Interface pattern. You saw that UML operations were added to the class AppFunction1 when you specified AppFunction1 as an argument, and it was bound to the assigned template parameter. Now you can modify some of the argument elements to observe how you can use reapplying patterns to update the other elements that participated in the pattern application.

So that you can see the power of using pattern, add an operation element to observe the results of reapplying the pattern. To add an operation to an interface:

1. In the Project Explorer view, right-click the IMyInterface interface and click **Add UML → Operation** . An operation is added to the interface and the default name is in focus.
2. Type myOperation over the default name.
3. In the diagram view, right-click the pattern instance and click **Patterns → Reapply Pattern** . Unless the pattern author specifically alters the reapply process in a pattern's design, all of the bound elements, as with this pattern, are now re evaluated by the pattern.
4. In the diagram view or in the Project Explorer view, locate the AppFunction1 class to observe the results of the reapplication of the pattern. The myOperation operation was added to IMyInterface.

Finish this tutorial by reviewing the materials in the [Apply a pattern](#) summary.

Summary: Apply a pattern

You have learned all of the basics needed to apply patterns, so you can select another one of the sample patterns and try it now on your own.

Provide extended summary information here.

Lessons learned

If you have completed all of the exercises, you should now be able to successfully apply any simple to complex pattern and perform the following tasks:

- Locate and open the Pattern Explorer view.
- Find the pattern short description, overview, and pattern documentation in the Pattern Explorer view.
- Identify the parts of an instantiation.
- Drag a pattern from the Pattern Explorer view to create an instantiation in the diagram editor.
- Select an existing element as a pattern argument and add it to the instantiation.
- Create a new element to be a pattern argument on the instantiation with or without a custom name.
- Add an argument to the instantiation using drag and drop.
- Modify the bound elements and reapply a pattern.

Additional resources

If you want to learn more about the topics covered in this tutorial, consider the following sources:

- White papers in developerWorks®
- Apply Patterns topics in the online Help system
- Pattern Author topics in the online Help system (for advanced pattern enthusiasts)

Related information

ibm.com
eclipse.org