

Rational Tau[®]

Java Tutorial



This edition applies to IBM® Rational® Tau® version 4.3 and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 2000, 2009.

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send written license inquiries to the following:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send written inquiries to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions. Therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for Rational Software|
IBM Corporation
1 Rogers Street
Cambridge, Massachusetts 02142
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Copyright license

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at www.ibm.com/legal/copytrade.html.

Third-party Trademarks

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product or service names may be trademarks or service marks of others.

Contents

Getting Started	1
Overview	1
Tutorial Objectives	1
Before You Begin	1
Documentation Conventions	2
About the Rational Tau Product	2
Tool Tips	3
Starting Rational Tau	3
Saving a Project in Rational Tau	3
Closing Rational Tau	4
Project Files and Directories	4
Model Element Names	4
Model Views	4
Lesson 1: Creating a Java Project	5
Goals for this Lesson	5
Exercise 1: Creating a Java Project	5
Task 1a: Creating a Java Project	5
Task 1b: Configuring the Standard Model View	7
Exercise 2: Creating Java Packages	7
Task 2a: Renaming the Default Package	7
Task 2b: Creating a New Package	8
Summary	8
Lesson 2: Creating a Use Case Diagram	9
Goals for this Lesson	9
Exercise 1: Creating a Use Case Diagram	9
Task 1a: Creating a Use Case Diagram	10
Task 1b: Renaming a Use Case Diagram	10
Task 1c: Adding an Actor and a Use Case	10
Task 1d: Adding an Association	11

Summary	11
Lesson 3: Creating an Activity Diagram	13
Goals for this Lesson	13
Exercise 1: Creating an Activity Diagram	14
Task 1a: Configuring UML Settings.	14
Task 1b: Creating an Activity Diagram.	14
Task 1c: Drawing Activity Nodes.	15
Summary	15
Lesson 4: Creating a Class Diagram	17
Exercise 1: Creating a Class Diagram	17
Task 1a: Creating a Class Diagram	17
Task 1b: Drawing a Class	18
Task 1c: Adding Attributes and Operations.	18
Summary	18
Lesson 5: Generating Java Code and More	19
Goals for this Lesson	19
Exercise 1: Generating and Editing Java Code	19
Task 1a: Exporting a Java Package	19
Task 1b: Adding Code for the compute operation.	20
Task 1c: Viewing Model Updates	20
Task 1d: Compiling Java Code	21
Task 1e: Running your Application	22
Summary	22
Conclusion	23
Technical Support and Documentation	25
Contacting IBM Rational Software Support	25
Prerequisites	25
Submitting problems.	26
Tau Documentation.	27
Index	29

Getting Started

Overview

This tutorial teaches you the basics of working with the Rational Tau product in a Java coding environment, and introduces the concepts of requirements analysis and project implementation. In the tutorial, you model a simple application that calculates the population growth projections of senior citizens residing in a city. The growth projections will help the city's planner determine the appropriate size of a new senior center. The calculation example used in this tutorial is based on Fibonacci numbers. For more information on Fibonacci numbers, see http://en.wikipedia.org/wiki/Fibonacci_number.

Tutorial Objectives

When you have completed this tutorial, you will have had experience with:

- ◆ Using the Rational Tau interface
- ◆ Creating a Java project
- ◆ Creating a use case diagram
- ◆ Creating an activity diagram
- ◆ Creating a class diagram
- ◆ Generating and editing Java code
- ◆ Running and compiling a Java application


Before You Begin

To complete the lessons in this tutorial, you must install the J2SE 5.0 Development Kit from Sun Microsystems. The kit is available at http://java.sun.com/javase/downloads/index_jdk5.jsp.

Note: After you install the development kit, be sure to set the PATH system environment variable to the correct path of **javac.exe** in the bin directory of your installation.

Documentation Conventions

This document uses the following conventions:

- ◆ **Boldface** for names of GUI objects and controls, including selection choices; and emphasis. Examples:
 - From the **Default model view** drop-down list box, select **Java View**.
 - Click the **Activity flow final** symbol  on the **Drawing** toolbar.
 - If the Rational Tau browser does not display, select **View > Browser**.
 - A project file, called **<project_name>.ttp**.
- ◆ Courier font in 10 point for pathnames, system messages, and items that you have to type. Examples:
 - The Output window displays the message `Animation session terminated`.
 - In the **Project name** box, replace the default project name with `<project name>`.
 - Type `show` for the function name, and press **Enter**.
- ◆ *Italics* for the first mention of a concept with an explanation.

About the Rational Tau Product

IBM® Rational® Tau® provides standards-based Model Driven Development™ (MDD™) of complex systems and robust software for enterprise IT applications, including those utilizing Service Oriented Architectures. Rational Tau's iterative requirements-based approach, comprehensive error-checking, and automated simulation increase developer productivity from initial requirements through final documentation and deployment.

Rational Tau offers a large feature set for developers to employ key enabling technologies in a natural, easy-to-use tool environment. Rational Tau makes a seamless and efficient environment for systems, software, and testability. It enables you to perform these tasks:

- ◆ **Analyze**, during which you can define, analyze, and validate the system requirements.
- ◆ **Design**, during which you can specify and design the architecture.
- ◆ **Implement**, during which you can automatically generate code, then build and run within the Rational Tau product.

Tool Tips

This section describes time saving features in Rational Tau that enable you to work more efficiently on a project. Wherever possible, the task-oriented procedures in this tutorial demonstrate the use of these features.

Using Shortcuts

Keyboard shortcuts (**CTRL + Arrow Key** for example) allow you to quickly navigate through the Rational Tau interface. For a complete list of available shortcuts, see the Help topic “Editor Shortcuts.”

Using the Auto Placement Feature

You can use Rational Tau’s auto placement feature to quickly add a series of elements in the drawing area. This feature is useful when drawing activity diagrams. For more information on this feature, see the Help topic “Add Symbols.”

Starting Rational Tau

Windows

To start the Rational Tau product in Windows, select **Start > Programs > IBM Rational > IBM Rational Tau *version number***.

Linux and Solaris

To start the Rational Tau product on Linux and Solaris, type the following command:

```
<installation path>/bin/tau
```



Saving a Project in Rational Tau

To save a project in Rational Tau, on the main menu bar, select **File > Save All**. To configure Rational Tau to automatically save changes to your project, follow these steps.

1. On the main menu bar, select **Tools > Options**.
2. On the **Save** tab, in the Auto-backup panel, select the **Activate** checkbox and specify an interval (every 5 minutes, for example.)
3. Click **OK**.

Closing Rational Tau

To exit the Rational Tau product, follow these steps.

1. Save your work. Do one of the following:
 - ◆ Press **CTRL+S**.
 - ◆ Click the Save button  to save your work.
2. Choose **File > Exit** or click the **Close** button .

Note

A red bar on the lefthand side of any file in your workspace indicates that you need to save your work before you exit Rational Tau.

Project Files and Directories

The Rational Tau product creates the following files and subdirectories in the project directory:

- ◆ A project file, called **<project_name>.ttp**
- ◆ A workspace, called **<project_name>.ttw** in which you can create a project and work on a project.
- ◆ Model files, called **<file_name.u2>** that contain the unit files for the project, including UML diagrams, packages, and code generation configurations.

Model Element Names

The names of model elements should follow conventions, such as these:

- ◆ Class names begin with an upper case letter, such as “System.”
- ◆ Operations and methods begin with lower case letters, such as “restartSystem.”
- ◆ Upper case letters to separate concatenated words, such as “checkStatus.”

Model Views

Rational Tau enables you to construct a model in several views, each representing different abstract characteristics of your model. For detailed information on model views, see the Help topic “Views.”

Lesson 1: Creating a Java Project

When you create a Java project in Rational Tau, it contains UML diagrams as well as Java packages, libraries, and add-ins. It also contains the required code generation configurations for the version of Java you are using. Rational Tau creates a directory containing the *project files* in a specified location. The name you choose for your new project is used to name project files and directories, and it appears at the top level of the project hierarchy in the Rational Tau browser.

Goals for this Lesson

In this lesson, you create a new Java project in Rational Tau, configure a project workspace, and add the necessary Java packages for the exercises in this tutorial. You will learn about the following concepts:

- ◆ Project configuration settings
- ◆ Project workspaces
- ◆ Project directories

Exercise 1: Creating a Java Project

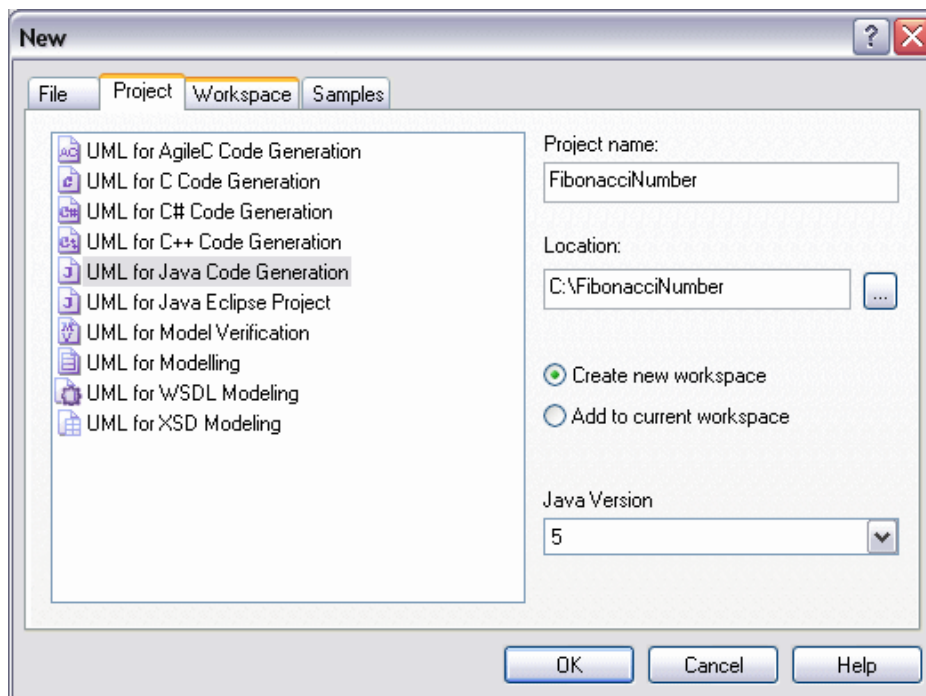
In this exercise, you create a new Java project and workspace in Rational Tau.

Task 1a: Creating a Java Project

To create a Java project, follow these steps:

1. Click **Start > Programs > IBM Rational > IBM Rational Tau *version number***.
2. Do one of the following:
 - ◆ Press **CTRL+N**, or
 - ◆ On the main toolbar, select **File > New**, or
 - ◆ On the IBM Rational Tau Welcome page, in the **New Project** panel, click **Proceed**.

3. In the **New** dialog box, in the **Project** tab, choose **UML for Java Code Generation**.
4. In the **Project name** box, type `FibonacciNumber`.
5. In the **Location** box, enter a new directory name or browse to an existing directory.
6. Accept the default option **Create new workspace**.
7. From the Java version drop-down list box, select version **5**. Your dialog box should resemble the following figure.



8. Click **OK**.
9. Accept the Developer Wizard defaults and click **Next**, then click **Finish**. Rational Tau creates a new project and workspace. In the Output window, the following message displays:

```
Java support loading ...  
Done.
```

Note: The amount of time required to create the project and load the necessary Java files may vary depending on memory usage and CPU size. In general, this process takes several seconds.

Task 1b: Configuring the Standard Model View

To complete the exercises in this tutorial, you must work with your project in the Standard model view. To make sure you are using the Standard model view, follow these steps.

1. In the Rational Tau browser, select the **.ttp** project file.
2. Do one of the following:
 - ◆ Press **ALT+4**.
 - ◆ Right click **Reconfigure Model View**.
 - ◆ From the menu bar, select **View > Reconfigure Model View**.
3. In the Reconfigure View dialog box, select **Standard View**.
4. Click **OK**.

Exercise 2: Creating Java Packages

In this exercise, you create two Java packages in your workspace. The first package is for analysis and requirements diagrams that provide a high level overview of your application. The second package is for a class diagram. In subsequent lessons, you create a use case diagram and an activity diagram in the analysis package, and a class diagram in the class package. Later in the tutorial, you *export* the class package when you generate Java files from your model.

Note: By default, when you create your project, Rational Tau adds a single Java package with the same name as the project. In the first task of this lesson, you rename the package included with the project. In the second task, you create a second Java package.

Task 2a: Renaming the Default Package

To rename the Java package, follow these steps.

1. In the Model view, select the **FibonacciNumber** package.
2. Press **F2**.
3. Type `Analysis`.

Task 2b: Creating a New Package

To create a new Java package, follow these steps:

1. In the Model view, right-click **Model**, then Select **New Model Element > Package**.
2. In the Create Model Root Element dialog box, in the **Element Name** field, type `Implementation`.
3. Right-click **Implementation**. On the drop-down menu, select **Convert to Java Package**. This will allow us to use this as a Java namespace and generate java code from this package.
4. Ensure that Java syntax is turned on for this package, by Right-clicking the package and making sure **Use Java syntax** has a check mark beside it.
5. On the main menu bar, select **File > Save All**.

Summary

In this lesson, you created a project and added Java packages to the project. You are ready to proceed to the next lesson, in which you begin your project by creating a use case diagram.

Lesson 2: Creating a Use Case Diagram

Use case diagrams show the behavior and capabilities of a system as it interacts with an external user or actor. A use case diagram also shows what a system will do and who will use it.

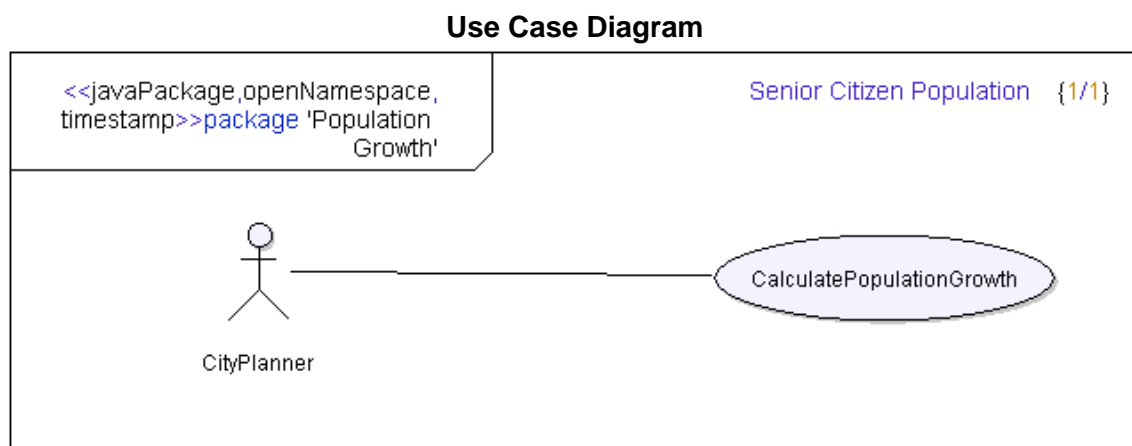
Goals for this Lesson

In this lesson, you create a simple use case diagram for the application you are modeling. As you work through this lesson, you will learn about the following elements in use case diagrams, and how to draw them:

- ◆ Actors
- ◆ Use Cases
- ◆ Associations

Exercise 1: Creating a Use Case Diagram

The following figure shows the use case diagram that you create in this exercise.




Task 1a: Creating a Use Case Diagram

To create a use case diagram, follow these steps.

1. Right-click **Analysis**.
2. Select **New Diagram > Use Case diagram**. Tau creates the use case diagram in your workspace and opens it in the drawing area.

Task 1b: Renaming a Use Case Diagram

In this task, you use the properties editor to rename the use case diagram. Follow these steps.



1. Right-click in the Model View **Usecase diagram1** and select **Properties**.
2. In the Edit Properties dialog box, in the **Name** field, replace the default name with `Senior Citizen Population`.
3. Click the **Close** button  to exit the Properties dialog box.

Task 1c: Adding an Actor and a Use Case

In this task, you add an actor and a use case to the diagram. An *actor* is an external element outside of the system that interacts with the system. A *use case* illustrates the capabilities of a system and shows why a user interacts with the system.

The actor in your diagram is a city planner who is using a system to gather data on the expected population growth of senior citizens in the city. The use case shows the system using a Fibonacci algorithm to compute the growth projections the planner needs to help determine the appropriate size of the new senior center.

To add an actor and a use case to your diagram, follow these steps.

1. Click the actor symbol  on the **Drawing** toolbar, then click in the drawing area. Tau adds an actor element in the drawing area.
2. Replace the default name with `CityPlanner`.
3. Click the use case symbol  on the **Drawing** toolbar. Rational Tau adds a use case element in the drawing area.
4. Replace the default name with `CalculatePopulationGrowth`.

Task 1d: Adding an Association

An *association line* shows a relationship between two elements in a use case diagram. In this task, you draw an association line that shows the interacting relationship between the city planner and the application use case. You can add an association line using the association symbol on the Drawing Tool menu, or by selecting the association “handle” on the **CityPlanner** element.


To draw an association line using the handle:

1. In the drawing area, select **CityPlanner**.
2. Click on the Association “handle” at the bottom of the **CityPlanner** element as shown in the following figure.



3. Click anywhere inside of the **CalculatePopulationGrowth** use case element. Rational Tau adds an association line that connects the two elements.

To draw an association line using the symbol:

1. Click the **Association** symbol  on the **Drawing** Toolbar.
2. Click the right edge of **CityPlanner** and the left edge of **ComputePopulationGrowth**. Rational Tau adds an association line that connects the two elements.
3. On the main menu bar, select **File > Save All**.

Your drawing should resemble the [Use Case Diagram](#) figure.

Summary

In this lesson, you created a use case diagram. You became familiar with the following elements of use case diagrams:

- ◆ Actors
- ◆ Use cases

- ◆ Associations

You are now ready to proceed to the next lesson, in which you create an activity diagram.

Lesson 3: Creating an Activity Diagram

An activity diagram shows a particular type of behavior based on a sequence of activities. An activity diagram consists of different activity items connected to each other with arrows. The arrows are used to show the direction of activity flow in the diagram.

Goals for this Lesson

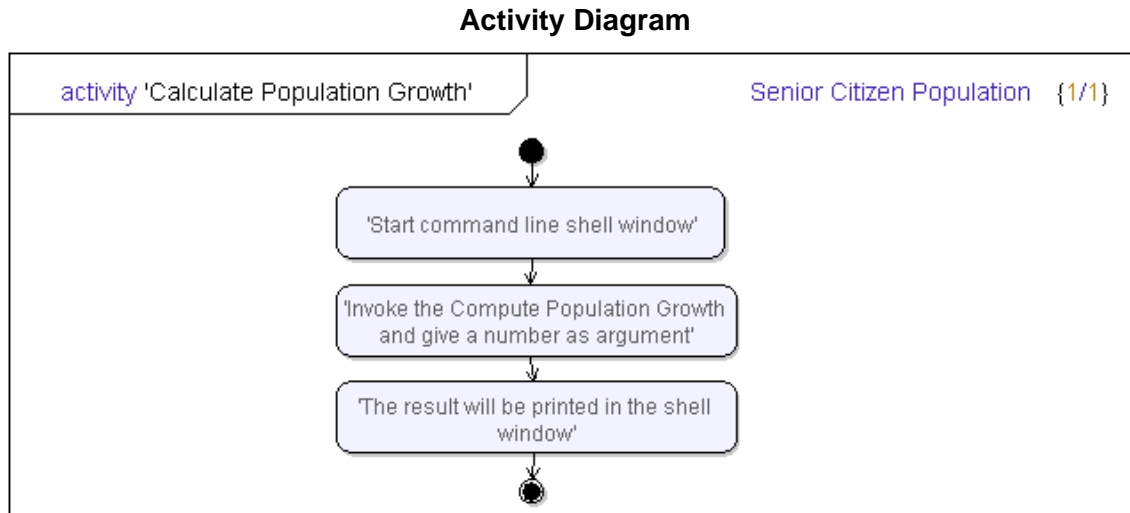
In this lesson, you create an activity diagram. Your activity diagram will show the sequence of activities that occur when the city planner uses the application to estimate the future population growth of senior citizens. The city planner starts the application, then enters a value that the application uses to calculate the population growth. In your activity diagram, the value supplied by the planner represents a number of years in the future. The response from the application is a population growth projection based on the number of years.

When you create an activity diagram using Rational Tau, you will learn how to draw

- ◆ An initial node
- ◆ An action node
- ◆ An activity line
- ◆ A final node

Exercise 1: Creating an Activity Diagram

The following figure shows the Activity Diagram that you create in this exercise.



Task 1a: Configuring UML Settings


In this task, you configure UML editing settings so you can draw the elements of your activity diagram as shown in the [Activity Diagram](#) figure. By default, Rational Tau is configured to draw the elements of an activity diagram horizontally. Follow these steps to change the setting to vertical so you can draw the elements as shown above.

1. On the menu bar, select **Tools > Options**, then select the **UML Advanced Editing** tab.
2. On the **UML Advanced Editing** tab, in the Activity diagrams panel, select **Vertical** from the **Autocreate** drop down list box.
3. Click **OK**.

Task 1b: Creating an Activity Diagram

To create an Activity diagram, follow these steps.




1. In the Rational Tau browser, expand **Model**, then expand the **Analysis** package.
2. Select **New Diagram > Activity Diagram**. Rational Tau creates the activity diagram in your workspace and opens it in the drawing area.

3. In the Edit Properties dialog box, in the **Name** field, replace the default name with Senior Citizen Population.
4. Click the Save button  to save your work.

Task 1c: Drawing Activity Nodes

Nodes show a specific unit of behavior within an activity flow. In this task, you draw an initial node, three action nodes and a final activity node in your diagram. The nodes show the units of behavior that occur when the user starts the application and enters a value that the application uses to produce a population projection figure.

To draw activity nodes, follow these steps:

1. Click the **initial node**  symbol on the **Drawing** toolbar. Rational Tau adds an initial node element in the drawing area.
2. Press **Shift-Spacebar** and on the pop-up menu, click the **Activity/action**  symbol, repeat this two times to result in three action nodes. Notice that each time you add an action node, Rational Tau automatically includes an *activity flow arrow* between each node. The arrows show the direction of flow in the diagram.
3. Continue to press **Shift-Spacebar** and click the **Activity final**  symbol on the **Drawing** toolbar.
4. Click each activity node and type the names as shown in the [Activity Diagram](#) figure.
5. On the main menu bar, select **File > Save All**.

You have finished drawing the activity diagram. Your diagram should resemble the [Activity Diagram](#) figure.

Summary

In this lesson, you created an activity diagram. You learned about the following elements in an activity diagram:

- ◆ Initial Nodes
- ◆ Action Nodes
- ◆ Final Nodes
- ◆ Associations

You are now ready to proceed to the next lesson, in which you create a class diagram.

Lesson 4: Creating a Class Diagram

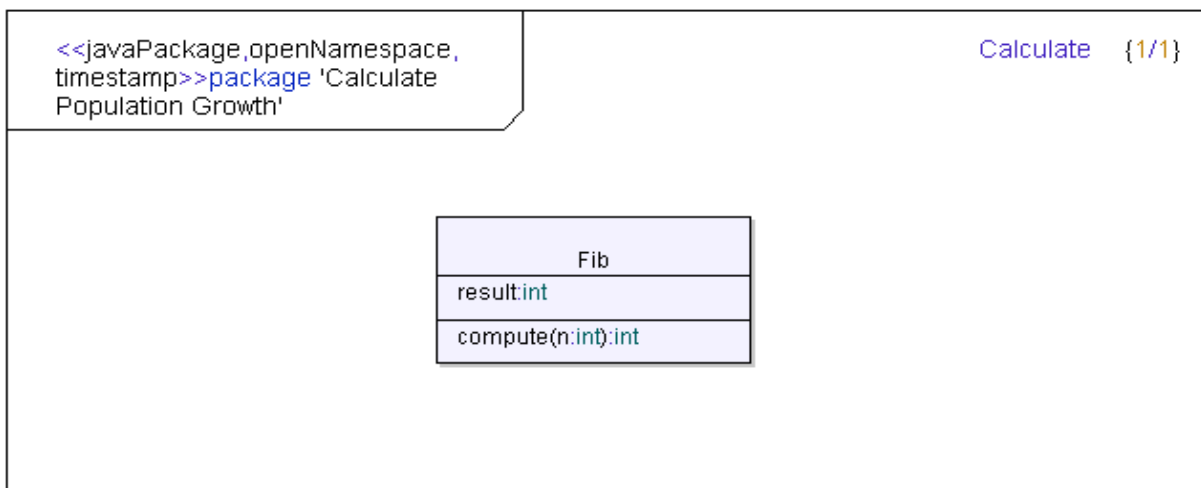
A class diagram shows the types of elements in a system and how they interact and relate to each other. Class relationships are typically shown with dependency or association lines.

Exercise 1: Creating a Class Diagram

In this exercise, you create a class diagram and draw a class in the diagram. The class contains an operation and an attribute for the application you are modeling.


Task 1a: Creating a Class Diagram

The following figure shows the class diagram that you create in this exercise.




To create a class diagram, follow these steps:

1. In the browser, expand **Model**.
2. Right-click the **Implementation** package and select **New Diagram > Class Diagram**. Rational Tau creates the class diagram in your workspace.

3. In the Edit Properties dialog box, in the **Name** field, replace the default name with `Calculate`.
4. Click the **Close** button  to exit the Properties dialog box.

Task 1b: Drawing a Class

To draw a class, follow these steps:

1. Click the Class button  on the **Drawing** toolbar, then click anywhere in the drawing area to add the class to the diagram.
2. Replace the default name of the class with `Fib`.

Task 1c: Adding Attributes and Operations

In this task, you add an attribute and an operation to the class you created in the previous task. The *attribute* you add will be the resulting number that is generated when the application performs a computation. The *operation* is the act of computing the number.

To add attributes to the **Fib** class, follow these steps:

1. Select the **Fib** class.
2. Place the cursor the middle panel of the class box and type `result:int` in the attribute text box.
3. Place the cursor in the bottom panel of the class box and type `compute(n:int):int` in the operation text box.
4. On the main menu bar, select **File > Save All**.

Summary

In this lesson, you created a class diagram. You learned about the following elements in a class diagram:

- ◆ Attributes
- ◆ Operations

You are now ready to proceed to the next lesson, in which you will generate Java code from the **Fib** class.

Lesson 5: Generating Java Code and More

Goals for this Lesson

In this lesson, you generate source code from the model you have created, and then you edit the source code. You learn about:

- ◆ Generating Java code
- ◆ Manually adding code
- ◆ Running a Java application

Exercise 1: Generating and Editing Java Code

In this exercise, you generate Java code from the **Fib** element you created in the previous lesson. After you generate the code from **Fib**, you manually add code to the `compute` operation. To generate the Java code, you export the package to a directory that you specify. Rational Tau creates a file in that directory that contains the Java code.

Task 1a: Exporting a Java Package

In this task, you export the **Implementation** package to Java source code. Follow these steps:

1. Select **Implementation** in the Rational Tau browser.
2. From the menu bar, choose **Java > Export > Package**.
3. In the Browse For Folder dialog box, do one of the following:
 - ◆ Click **OK** which will create a default folder named **Implementation** in the project directory, or
 - ◆ Use the scroll bar to specify an alternate directory, then click **Make New Folder**.

Rational Tau creates a file named **Fib.java** that contains the Java code generated from the **Implementation** package. Rational Tau also creates a java build artifact named **Artifact0000**.

Task 1b: Adding Code for the compute operation

In this task you add code that is the body of the `compute` operation that you created in [Lesson 4: Creating a Class Diagram](#). To add code manually for the `compute` operation, follow these steps:

1. In the Model browser, double-click the **Calculate** diagram.
2. In the drawing area, right-click **Fib**.
3. Select **Go to Source**.
4. Remove the following lines:

```
int result;
int compute( int n){}
```

5. Replace them with the following code:

```
int result;
int compute( int n ){
    if(n == 0)
        result = 1;
    else if(n == 1)
        result = 2;
    else
        result = n + compute(n - 1);
    return result;
}
public static void main(String[] argv)
{
    Fib myFib = new Fib();
    myFib.compute( new Integer(argv
[0]).intValue() );
    System.out.println( "The result is " +
myFib.result );
}
```

6. On the main menu bar, select **File > Save All**.

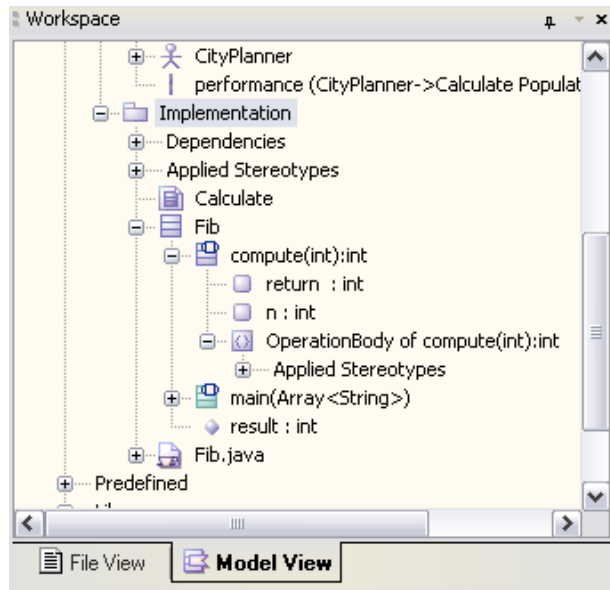
Task 1c: Viewing Model Updates

In this task, you view the updates that were made to the model when you added the code in the previous task. When you save your work, Rational Tau detects the updated source files in your project and updates your model.

To view the updates you made in the previous task, follow these steps.

1. In the Model view, expand **Model**.

2. Right-click **Artifact0000**. This file is a build artifact that contains the stereotypes needed to build your model.
3. Select **Build (Java) > Update**.
4. In the **Implementation** package directory, expand **Fib**. Notice that Rational Tau has added the `operationBody` element to the `compute(int):int` operation, as shown in the following figure. The `operationBody` is the code you added in the previous task.



Task 1d: Compiling Java Code

In this task, you compile the source code. To compile your code, follow these steps.

1. In the model browser, select **Fib**.
2. From the Java menu, select **Compile**.
3. Verify that a successful `javac` compilation command displays in the **Build** tab of the Output window.

Note

The output displayed in the **Build** tab may vary depending on your operating system.

Task 1e: Running your Application

The application runs an algorithm that computes Fibonacci numbers based on the code you added in the previous task. To run the application, follow these steps.

1. Open a command window and navigate to your project directory.
2. In the project directory, enter the following command:

```
java Implementation/Fib "n"
```

Where [n] is an integer representing a number of years in the future.

3. Press **Enter**. The result is total number of seniors expected to be residing in the city in the number of years you specify. For example, if you entered 22, the total number of senior expected to be residing in the city in 22 years would be approximately 254.

Summary

In this lesson, you generated Java code, manually added code, and ran your application. You learned how to

- ◆ Export a Java package
- ◆ Edit source code
- ◆ Test your model

Conclusion

This tutorial has introduced you to UML modeling with IBM Rational Tau in a Java environment. By completing the exercises in this tutorial, you have become familiar with the Tau product. You have learned how to:

- ◆ Create a project
- ◆ Use drawing tools and shortcut keys
- ◆ Draw diagrams
- ◆ Export a Java package
- ◆ Compile code
- ◆ Add external code to a model

Your knowledge of how to perform these tasks gives you a basic understanding of IBM Rational Tau. You will enhance your skills and product knowledge as you continue to work on UML modeling projects using this product.

Technical Support and Documentation

Contacting IBM Rational Software Support

If the self-help resources have not provided a resolution to your problem, you can contact IBM® Rational® Software Support for assistance in resolving product issues.

Prerequisites

To submit your problem to IBM Rational Software Support, you must have an active Passport Advantage® software maintenance agreement. Passport Advantage is the IBM comprehensive software licensing and software maintenance (product upgrades and technical support) offering. You can enroll online in Passport Advantage from <http://www.ibm.com/software/lotus/passportadvantage/howtoenroll.html>

- ◆ To learn more about Passport Advantage, visit the Passport Advantage FAQs at http://www.ibm.com/software/lotus/passportadvantage/brochures_faqs_quickguides.html.
- ◆ For further assistance, contact your IBM representative.

To submit your problem online (from the IBM Web site) to IBM Rational Software Support, you must additionally:

- ◆ Be a registered user on the IBM Rational Software Support Web site. For details about registering, go to <http://www.ibm.com/software/support/>.
- ◆ Be listed as an authorized caller in the service request tool.

Submitting problems

To submit your problem to IBM Rational Software Support:

1. Determine the business impact of your problem. When you report a problem to IBM, you are asked to supply a severity level. Therefore, you need to understand and assess the business impact of the problem that you are reporting.

Use the following table to determine the severity level.

Severity	Description
1	The problem has a <i>critical</i> business impact: You are unable to use the program, resulting in a critical impact on operations. This condition requires an immediate solution.
2	This problem has a <i>significant</i> business impact: The program is usable, but it is severely limited.
3	The problem has <i>some</i> business impact: The program is usable, but less significant features (not critical to operations) are unavailable.
4	The problem has <i>minimal</i> business impact: The problem causes little impact on operations or a reasonable circumvention to the problem was implemented.

2. Describe your problem and gather background information. When describing a problem to IBM, be as specific as possible. Include all relevant background information so that IBM Rational Software Support specialists can help you solve the problem efficiently. To save time, know the answers to these questions:

- ◆ What software versions were you running when the problem occurred?

To determine the exact product name and version, use the option applicable to you:

- ◆ Start the IBM Installation Manager and select **File > View Installed Packages**. Expand a package group and select a package to see the package name and version number.
- ◆ Start your product, and click **Help > About** to see the offering name and version number.
- ◆ What is your operating system and version number (including any service packs or patches)?
- ◆ Do you have logs, traces, and messages that are related to the problem symptoms?
- ◆ Can you recreate the problem? If so, what steps do you perform to recreate the problem?

- ◆ Did you make any changes to the system? For example, did you make changes to the hardware, operating system, networking software, or other system components?
 - ◆ Are you currently using a workaround for the problem? If so, be prepared to describe the workaround when you report the problem.
3. Submit your problem to IBM Rational Software Support. You can submit your problem to IBM Rational Software Support in the following ways:
- ◆ **Online:** Go to the IBM Rational Software Support Web site at <https://www.ibm.com/software/rational/support/> and in the Rational support task navigator, click **Open Service Request**. Select the electronic problem reporting tool, and open a Problem Management Record (PMR), describing the problem accurately in your own words.
 - ◆ For more information about opening a service request, go to <http://www.ibm.com/software/support/help.html>
 - ◆ You can also open an online service request using the IBM Support Assistant. For more information, go to <http://www.ibm.com/software/support/isa/faq.html>.
 - ◆ **By phone:** For the phone number to call in your country or region, go to the IBM directory of worldwide contacts at <http://www.ibm.com/planetwide/> and click the name of your country or geographic region.
 - ◆ **Through your IBM Representative:** If you cannot access IBM Rational Software Support online or by phone, contact your IBM Representative. If necessary, your IBM Representative can open a service request for you. You can find complete contact information for each country at <http://www.ibm.com/planetwide/>.

If the problem you submit is for a software defect or for missing or inaccurate documentation, IBM Rational Software Support creates an Authorized Program Analysis Report (APAR). The APAR describes the problem in detail. Whenever possible, IBM Rational Software Support provides a workaround that you can implement until the APAR is resolved and a fix is delivered. IBM publishes resolved APARs on the IBM Rational Software Support Web site daily, so that other users who experience the same problem can benefit from the same resolution.

Tau Documentation

The IBM Rational Tau documentation provides information on most of the topics covered in this tutorial. documentation is available from the following locations:

- ◆ From the Start menu, click **Programs > IBM Rational > IBM Rational Documentation > IBM Rational Tau *version number***.
- ◆ From the Help menu in the Tau interface.

The following table lists the Help topics that provide additional information on key concepts covered in this tutorial.

Help Topic	Reference Information
“UML and Java”	General information on UML modeling using Tau in a Java environment.
“Working with Diagrams”	Provides information on creating, saving, and printing diagrams, as well as other common diagram operations.
“UML Language Guide”	Provides a complete list of UML language constructs and model elements

Index

A

- Activity diagrams
 - creating 14
- Adding
 - actors 10
 - association lines 11
 - attributes 18
 - code 20
 - operations 18

C

- Class diagrams
 - creating 17
- Closing
 - Tau 4

D

- Diagrams
 - activity 14
 - class 17
 - usecase 10
- Directories 4
- Documentation 25
 - conventions 2

E

- Elements
 - names 4
- Exiting
 - Tau 4
- Exporting
 - java packages 19

F

- Files 4

I

- IBM Customer Support 25

J

- Java language
 - editing code 19
 - generating code 19
 - packages 7

L

- Launching
 - Linux Tau 3
 - Tau 3
 - Windows Tau 3
- Linux 3

M

- Model
 - views 4
- Models
 - element names 4

N

- Names
 - for elements 4

O

- Opening
 - Tau 3
- Operations 18

P

- Packages
 - creating 7
 - renaming 7
- Projects 4
 - creating 5
 - directories 5
 - files 5
 - saving 3

R

Renaming 7
renaming 10

S

Solaris
starting Tau on 3
Standard model view
configuring 7

T

Tau 2
closing 4

documentation 25
exiting 4
projects 4
starting 3
Tool Tips 3

U

Use case diagrams 9
creating 10

W

Windows
starting Tau 3