

Development environment

An integrated development environment (IDE) provides facilities for software development and typically consists of editors for editing source code, a compiler, project management tools, build automation tools and debugging tools. IDEs help in increasing the productivity by automating many tasks and often provide one-step process for creating a ready to use binary from source code. [Eclipse](#) and [NetBeans](#) are two of the most popular IDEs. IDEs that enable Java EE development also provide integration of Java EE Server runtime environments so that the developer can quickly see the effect of the changes being made to the application without having to create the binaries explicitly.

The Apache Geronimo Development Tools project is aimed at providing a rich set of development tools for Geronimo, focusing on application development, migration and IDE integration. The two major tools available currently are **Geronimo Eclipse Plugin (GEP)** and **JBoss to Geronimo Migration Tool (J2G)**. Also, a **Geronimo NetBeans Plugin** is under development in Geronimo Sandbox. This article on setting up a development environment is organized as given below:

- [Options and tools](#)
 - [Eclipse](#)
 - [NetBeans](#)
 - [Apache Maven](#)
 - [Web Tools Platform \(WTP\)](#)
 - [Geronimo Eclipse Plugin](#)
 - [JBoss to Geronimo Migration Tool \(J2G\)](#)
 - [Maven Integration for Eclipse](#)
- [Installing Eclipse](#)
- [Geronimo Server Runtimes and Servers](#)
 - [Defining a Geronimo Server Runtime](#)
 - [Defining a Geronimo Server](#)
 - [Creating a new project](#)
- [Configuring your development environment](#)
 - [Editing a Server configuration](#)
 - [Other configuration](#)

Options and tools

In this section we briefly discuss various tools available to set up a development environment.

Eclipse

Eclipse is an open source IDE for Java developers and consists of Java Development Tools. Eclipse is written primarily in Java. Eclipse community is focused on building an open development platform comprised of extensible frameworks, tools and runtimes for building, deploying and managing software across the lifecycle. Users can extend its functionality by installing plugins written for Eclipse. For more details visit [Eclipse web site](#).

NetBeans

NetBeans is an open source IDE written entirely in Java using the NetBeans platform. NetBeans IDE provides all the tools needed for creating desktop, enterprise web and mobile applications in Java. For more details visit [NetBeans web site](#).

Apache Maven

Apache Maven is a Java tool for software project management and automation for Java. It uses project object model to describe the software project being built, its dependencies on other external modules and components, and the build order. It comes with pre-defined targets for performing certain well defined tasks such as compilation of code and its packaging. Maven consists of a core engine which provides basic project-processing capabilities and build-process management, and a host of plugins which are used to execute the actual build tasks. Maven is primarily a command-line tool. Plugins to integrate Maven with Eclipse and NetBeans IDEs are also available. For more details visit [Apache Maven web site](#).

Web Tools Platform (WTP)

The Eclipse Web Tools Platform (WTP) project extends the Eclipse platform with tools for developing Web and Java EE applications. It includes source and graphical editors for a variety of languages, wizards and built-in applications to simplify development, and tools and APIs to support deploying, running, and testing applications. For more details visit [WTP Project web site](#).

Geronimo Eclipse Plugin

The Geronimo Eclipse Plugin (GEP) provides integration between Geronimo and the Web Tools Platform (WTP). With this plugin, users will be able to use the features in WTP to create, deploy and debug applications on Geronimo.

The Geronimo Eclipse Plugin (GEP) requires the following prerequisite software (all of which is platform specific):

- Sun JDK 5.0 (J2SE 5.0)
- Eclipse IDE for Java EE Developers

Sun JDK 5.0 can be downloaded from [Sun's J2SE 5.0 Downloads site](#). The Eclipse IDE for Java EE Developers can be downloaded from the [Eclipse Downloads site](#). Download and extract the **Eclipse IDE for Java EE Developers** archives to a directory of your choice (for e.g. C:\eclipse). The archive will be extracted to a directory named eclipse under the directory you specified (for e.g. C:\eclipse\eclipse), referred to as **<eclipse_home>** from now on. Make sure that the JDK in the PATH is Sun JDK 5.0. Launch Eclipse by running **<eclipse_home>/eclipse**.

The Geronimo server can be installed using the Geronimo Eclipse Plugin, or optionally you may install it manually. To install it manually, download the Geronimo 2.1.1 server from <http://www.apache.org/dist/geronimo/> and extract the archive to a directory of your choice (for e.g. C:\g). The archive will be extracted to a directory **geronimo-tomcat6-javaee5-2.1.1** (for e.g. C:\g\geronimo-tomcat6-javaee5-2.1.1) or **geronimo-jetty6-javaee5-2.1.1** depending on whether you downloaded Geronimo 2.1.1 distribution with Tomcat or Jetty as the web container. We will refer to this directory as **<geronimo_home>** from now on.

JBoss to Geronimo Migration Tool (J2G)

The JBoss to Geronimo Migration Tool (J2G) is an Eclipse plugin designed to assist in migrating the sources of an application written for the JBoss application server or written for Java Enterprise Edition (Java EE) to the Apache Geronimo platform. For more information on using the J2G tool follow [this link](#).

Maven Integration for Eclipse

Maven Integration for Eclipse provides tight integration for Maven into the IDE and providing the following features:

- Launching Maven builds from within Eclipse
- Dependency management for Eclipse build path based on Maven's pom.xml
- Resolving Maven dependencies from the Eclipse workspace without installing to local Maven repository
- Automatic downloading of the required dependencies from the remote Maven repositories
- Wizards for creating new Maven projects, pom.xml or to enable Maven support on plain Java project
- Quick search for dependencies in Maven remote repositories
- Quick fixes in the Java editor for looking up required dependencies/jars by the class or package name

For more details and installation instructions, visit <http://m2eclipse.codehaus.org/>.

Installing Eclipse

See [How to install Geronimo Eclipse Plugin v2.1.2](#)

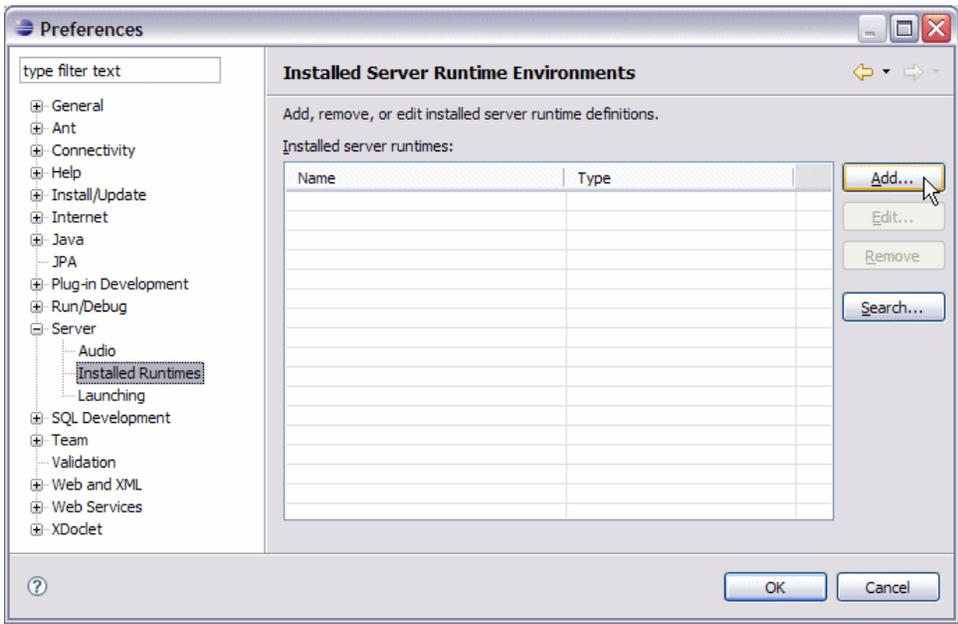
Geronimo Server Runtimes and Servers

A **Geronimo Server Runtime** in GEP associates a Geronimo 2.1 Server installation with a JRE to be used to run that server. A **Geronimo Server** in GEP associates a Geronimo Server Runtime with a profile consisting of Security settings, Publish Settings, Port Configuration, Java VM settings to be used with that Geronimo Server Runtime.

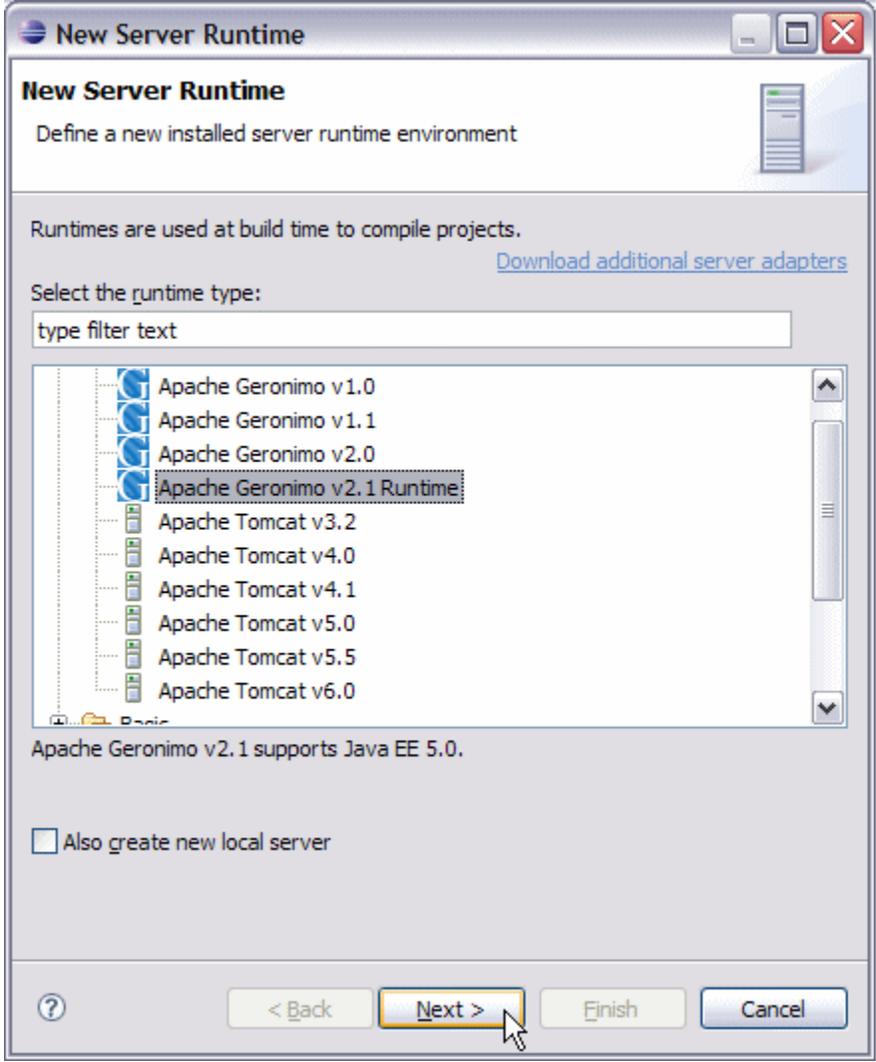
Defining a Geronimo Server Runtime

In order to define a new Geronimo Server v2.1 Runtime, follow the steps below:

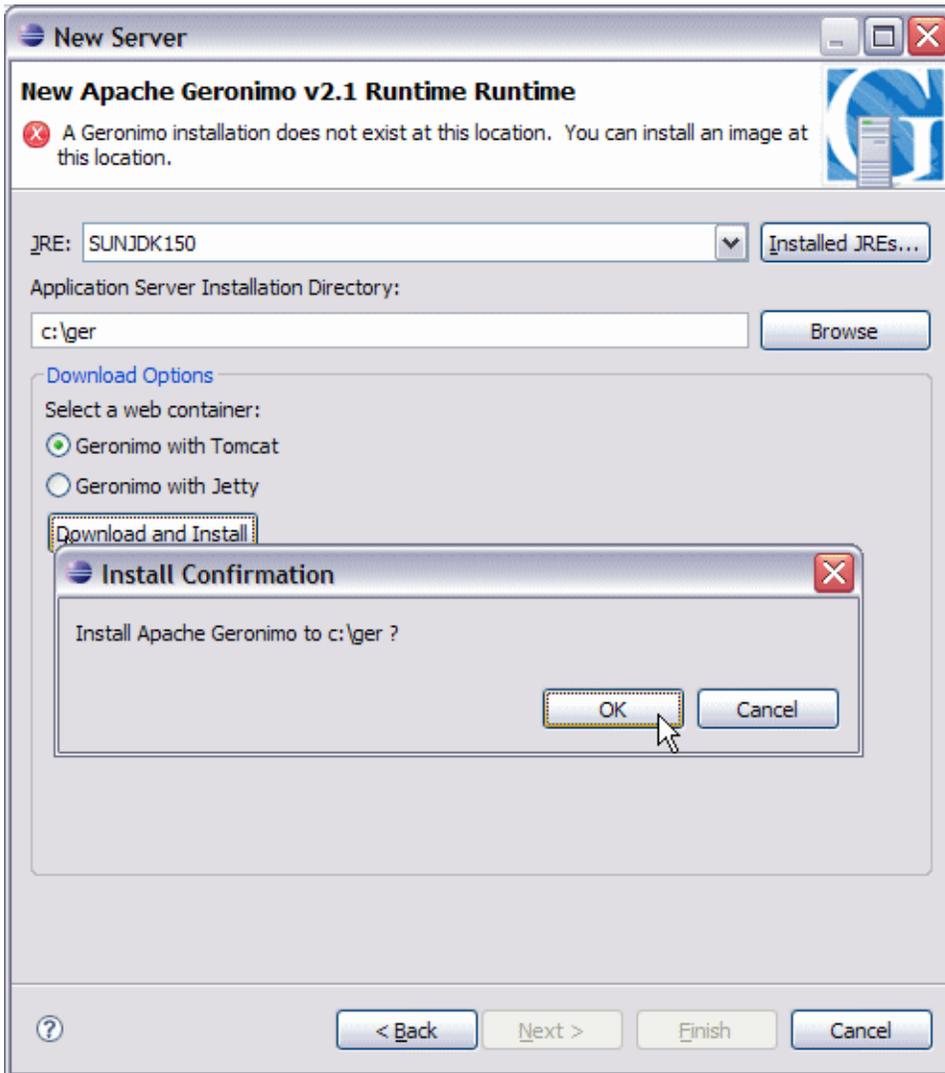
1. Click **Window -> Preferences -> Server -> Installed Runtimes**.



- 2. Click **Add** to launch **New Server Runtime** dialog.
- 3. Select **Apache Geronimo v2.1 Runtime**, uncheck **Also create new local server** (we will discuss about the use of this option later) and click **Next**



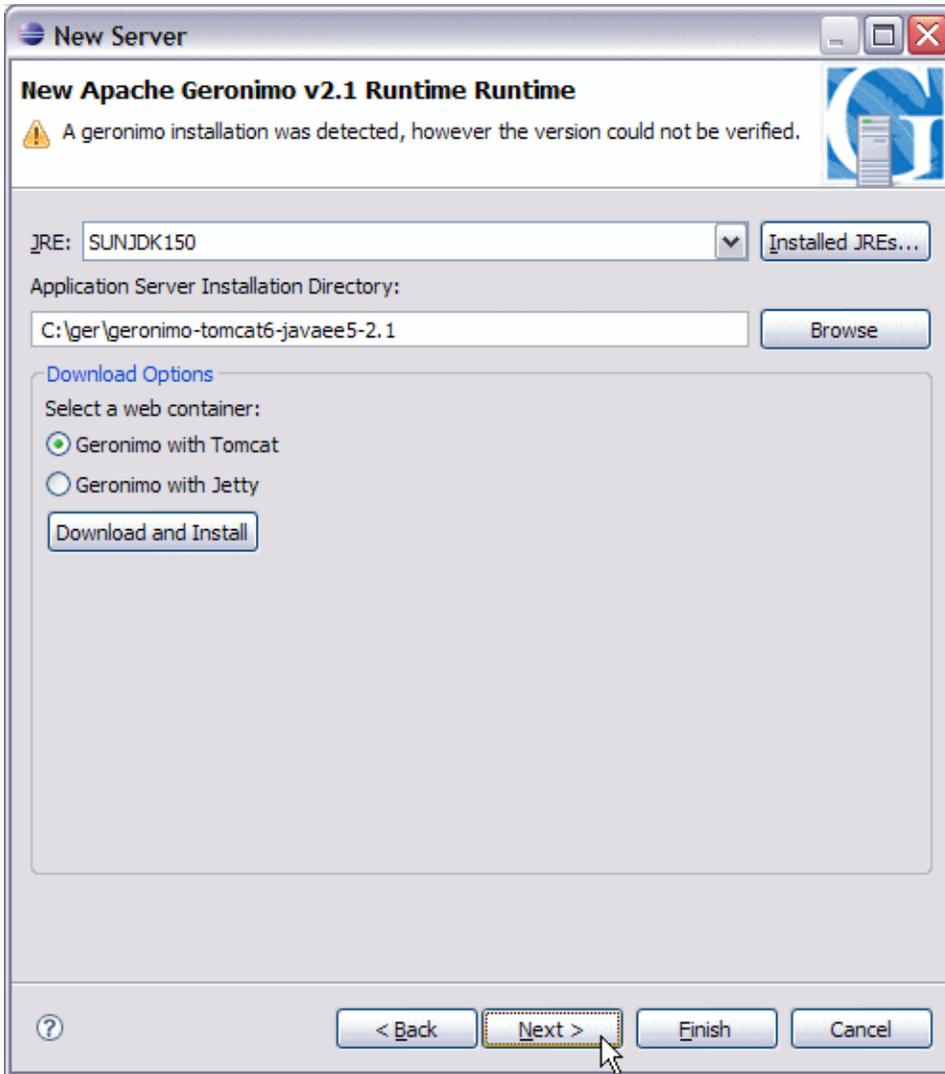
4. Select **Workbench default JRE** (click **Installed JREs** to add any new JREs and return to this dialog once done).
5. If you already have Geronimo v2.1 installation that you want to use with GEP, enter the directory name or browse to **<geronimo_home>** and click **Next**. Skip the next three steps and go to selecting source archive.
6. Enter a directory for **Application Server Installation Directory**, under **Download Options** select the web container and click **Download and Install**.



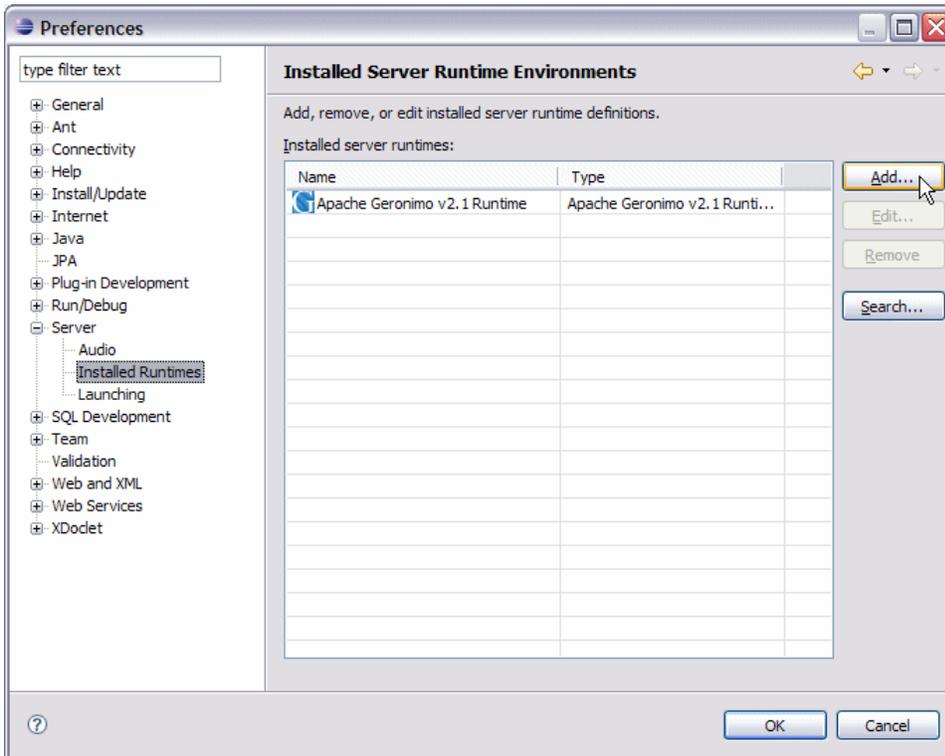
Important

The directory specified for **Application Server Installation Directory** must exist in order to select **Download Options**.

7. Click **OK** for Install Confirmation. This will install the selected Geronimo v2.1 server and fill the **Application Server Installation Directory** field accordingly.
8. Click **Next**.



9. Select the location of the archive containing Geronimo source. This is required only if you intend to debug into Geronimo source code.
10. Click **Finish** to return to **Installed Server Runtime Environments** dialog.

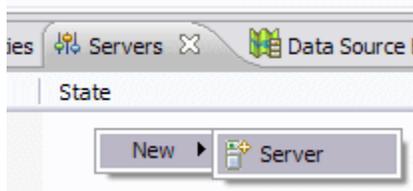


Additional Server Runtimes to use a different Geronimo server installation or a different JRE can be added in a similar manner.

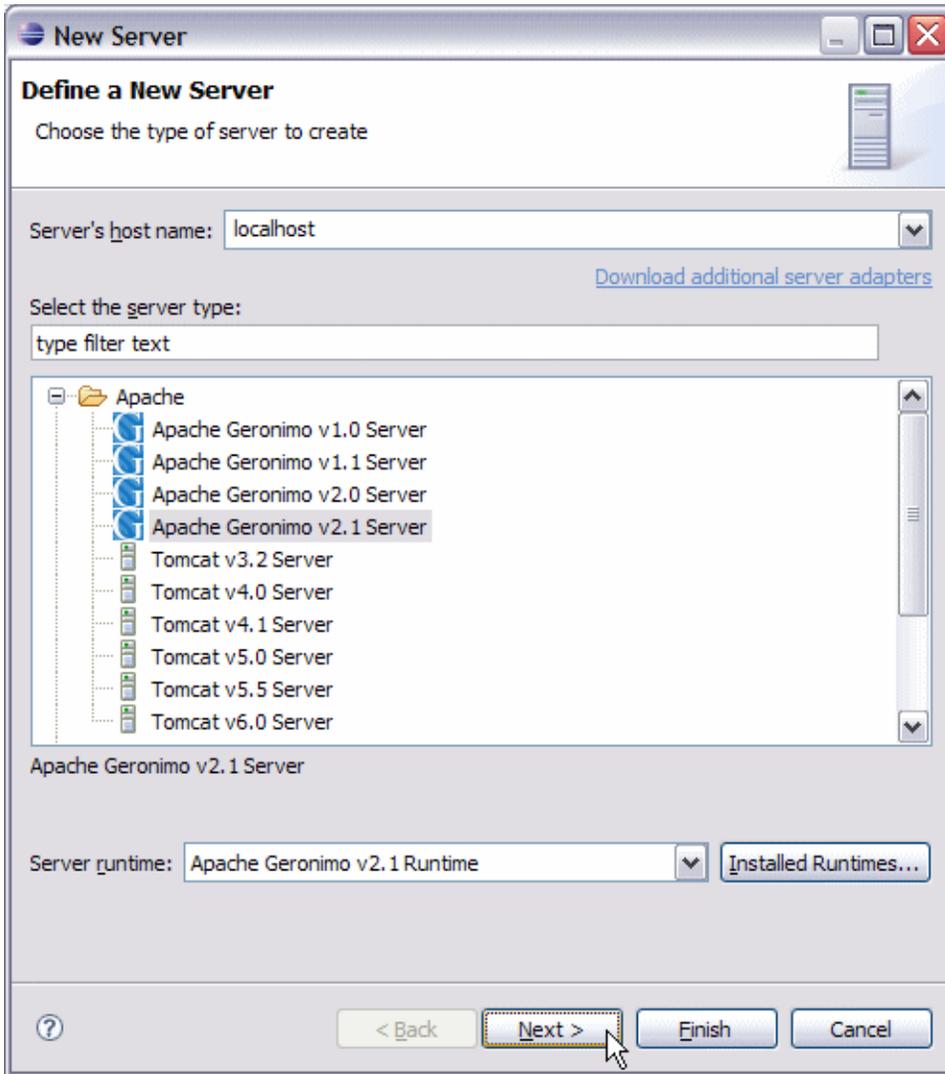
Defining a Geronimo Server

Follow the steps below to define a new Geronimo v2.1 Server:

1. Launch the **Define a New Server** dialog using one of the following:
 - From the **Servers** view in the Java EE Perspective: **Right-click** -> **New** -> **Server**.
 - From any Perspective: Select **File** -> **New** -> **Other** -> **Server**.

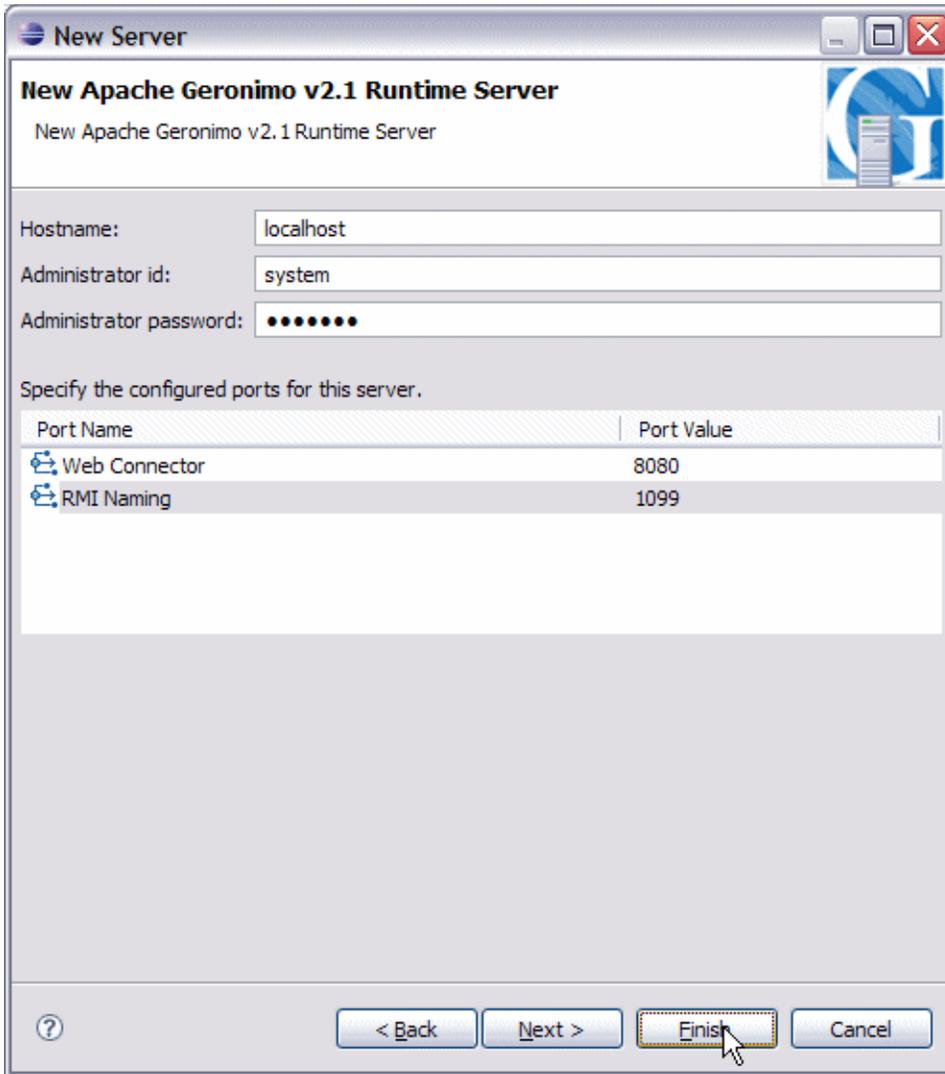


2. Select **Apache Geronimo v2.1 Server**, select an **Apache Geronimo v2.1 Runtime** from the **Server runtime** dropdown and click **Next**.

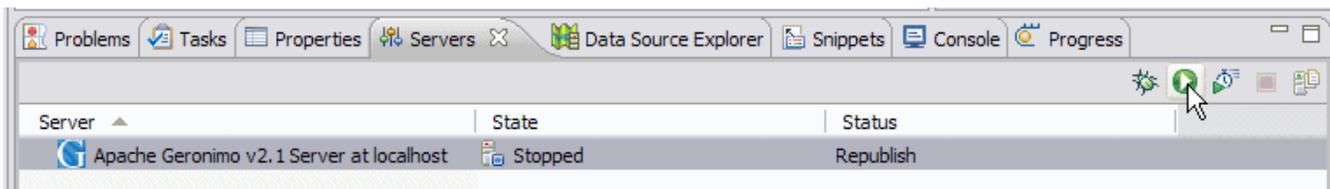


Additional Geronimo Server Runtimes can be installed by clicking on **Installed Runtimes** button and return to this step once done to select the newly added server runtime.

3. Modify **Hostname**, **Administrator Id**, **Administrator password**, **Web Connector port** and **RMI Naming port** if necessary and click **Finish**. Normally you will not need to change the default values for these fields for a newly installed Geronimo 2.1 server.



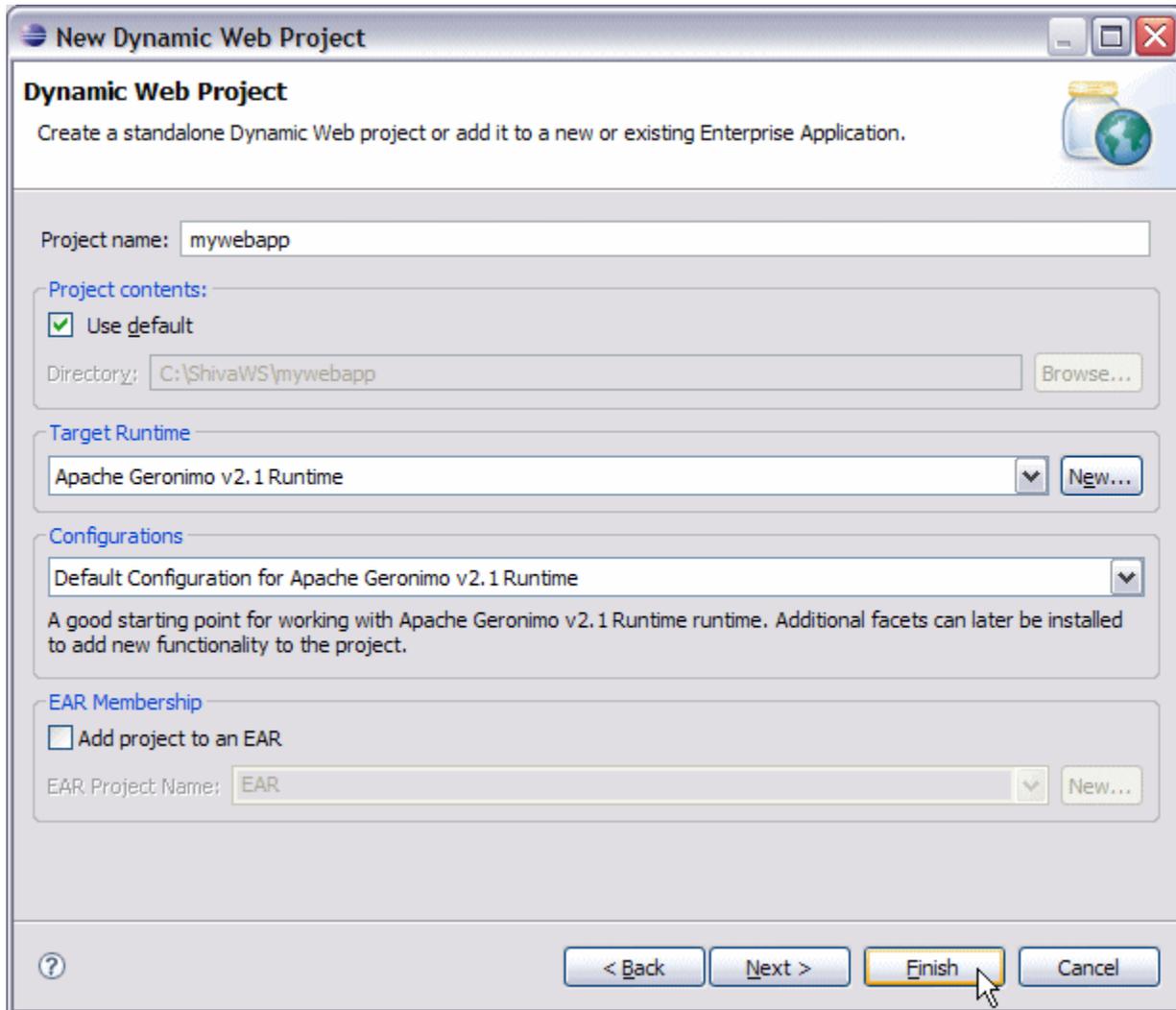
This completes defining a new Geronimo 2.1 Server. The newly added server will appear in the **Servers** view. The server can be started by selecting the server and clicking on the **Start server** button.



 A Geronimo Server can also be defined while creating a Geronimo Server Runtime by checking the option **Also create new local server**. This will present an additional dialog to configure the hostname, ports, etc.

Creating a new project

Once Geronimo Server Runtimes are defined, Eclipse provides these runtime for selecting as a target runtime while creating an EJB, Web and Java EE projects. Notice the **Target Runtime** field in the **Dynamic Web Project** dialog shown in the figure below.



Upon adding the Geronimo 2.1 Server Runtime as the target runtime, the Geronimo server libraries and Java EE specification libraries get automatically added to the build path of the project. GEP will also create Geronimo specific deployment plans.

In order to run an application on Geronimo, follow the steps below:

1. **Right-click** on the project and click **Run As -> Run on Server**.
2. Click **Choose and existing server**, select **Apache Geronimo v2.1 Server at localhost** and click **Finish**.

The application can also be run on Geronimo by adding the project to the Geronimo server explicitly as given below:

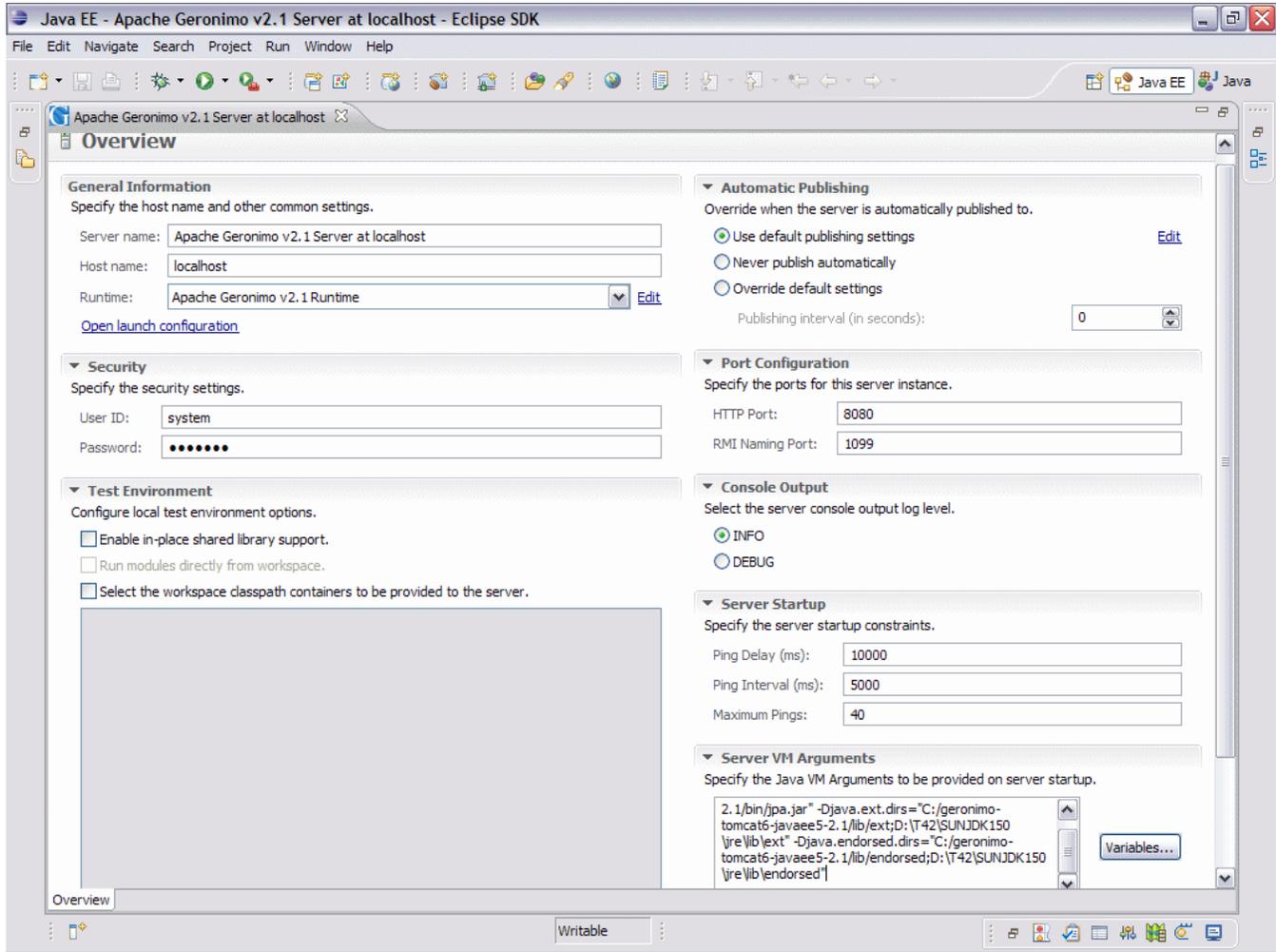
1. In the **Servers** view, **Right-click** on **Apache Geronimo v2.1 Server at localhost** and select **Add and Remove Projects**.
2. In the **Add and Remove Projects** dialog, select the project under **Available projects** and click on **Add** button. The project will now be listed under **Configured projects**.
3. Click **Finish**.
4. If the server status is shown as **Republish**, right-click on the server and select **Publish**.

Configuring your development environment

In this section we discuss various options to configure your development environment.

Editing a Server configuration

In order to edit the configuration of a server, double click on server in the **Servers** view to open the overview of the server as shown below.



- **General Information**
Edit the fields in this section to change the name of the server or to associate the **Geronimo Server** with a different **Apache Geronimo v2.1 Runtime** or to change the hostname with which the Geronimo Server is associated.
- **Security Credentials**
The default User ID is **system** with a password **manager**. If your server installation uses a different set of credentials, change these accordingly. These security credentials configured here are used to detect server status as well as deploying and undeploying of applications.
- **Automatic Publishing**
All servers use the default publishing settings. The server can be configured to not publish automatically by selecting the **Never publish automatically** option. The default publish setting can be overridden by selecting the **Override default settings** option and providing a new **Publish interval**.
- **Port Configuration**
The default HTTP and RMI ports are 8080 and 1099 respectively. If your server installation uses a different HTTP and/or RMI port, edit these values accordingly. Altering the values here does not change the corresponding ports in the associated Geronimo Server Runtime.
- **Console Output**
The default server console output log level is **INFO**. If you want to enable debug output to console, select **DEBUG**.
- **Server Startup**
The settings under this section control the ping thread that polls to the server to detect the server status when the server is started from within eclipse. **Ping delay** specifies the time the ping thread will wait before polling the server. **Ping interval** controls the time between successive pings. **Maximum Pings** is the number of times the ping thread will poll the server to detect a successful startup. If the server startup can not be confirmed, the ping thread will stop the server.
In case of a remote server, Ping interval controls the time between successive pings to update the server status.
- **Server VM Arguments**
Any additional arguments to be passed on to the server VM can be specified here.

Other configuration

- [Configuring Geronimo Eclipse Plugin to publish Maven Dependencies as Shared Library](#)
- [Configuring Geronimo Eclipse Plugin to publish POJO projects as Shared Libraries](#)
- [Using Eclipse XML tools in Apache Geronimo](#)

