Deploying and debugging applications using GEP

{scrollbar}

Deploying and debugging applications using GEP

The Geronimo Eclipse Plugin is a tool for deploying and testing applications to a Geronimo server. Define a new server and server runtime, before attempting to deploy your applications.

When you create or import your Java EE project in the Eclipse Integrated Development Environment (IDE), specify the Geronimo server runtime you defined previously as the target runtime. This action adds the server's class libraries to your project's build path.

Note
If you deploy your assets using Eclipse, it is recommended that you also undeploy and redeploy your applications using Eclipse. For example, if you deploy your application in Eclipse and then undeploy the application using the administrative console or the `deploy` command, Eclipse will not detect the change and the view will show that the asset is still deployed. If this happens to you, you can correct the view by removing the applications that have been published to the server but undeployed outside of the Eclipse framework.

Local deployment

Use this procedure to deploy applications to a local server.

1. In the **Java EE** perspective, select the **Project** view and right click the Java EE project you want to deploy. Select **Run As, Run on server**
2. On the **Run on server** panel, if you have an existing server, keep the **Choose an existing server** option and select the server. If you don't have a Geroimo server defined, select the **Manually define a new server** option and select the server.
3. Click **Finish**. The Geronimo Eclipse Plugin will deploy the applications shortly. If the server is not started, the Geronimo Eclipse Plugin will start the server and deploy the Java EE applications when the server has initialized.

Removing an asset that has been published to the server

Once an asset has been published to the server, you must use the tool's **Add/Remove projects** option if you want to remove it from the server. If you simply remove the asset without removing its project, the asset will remain deployed on the server.

1. In the **Java EE** perspective, the select the **Server** view.
2. On the **Server** panel, right click on the server where the asset was deployed.
3. On the resulting context menu, click **Add/Remove Projects**.
4. On the selection panel, click on the asset to undeploy and click the **<Remove** button to move it from the right to the left list. Click **Finish**.

Remote deployment

You can use the same steps discussed above to deploy a Java EE asset to a remote server, but there are additional considerations.

1. Make sure the target system can be contacted by ping command.
2. If there is a firewall between your system and the target server, it must be configured to permit both HTTP and RMI requests to flow between your system and the server's host. When the server is installed, the initial HTTP port is **8080** and the initial RMI port is **1099**. If the target server has been configured to use different ports, the firewall must be configured to use those ports instead.
3. On the target server, find the `RemoteDeployHostname` attribute in `<geronimo_home>\var\config\config-substitutions.properties`, and change the value to the name or IP address of the target server's host.
4. Define a local server and then change the **hostname** in Eclipse to the name of the remote server's host. This is required because the Eclipse framework must use the class libraries from the local server. You will not be able to use Eclipse to start, stop, or restart the remote server. You cannot use Eclipse to start the remote server in debug mode. Often, this limitation makes it more convenient to develop and debug your Java EE assets using a local server and then switch to a remote server when you need a common server to integrate your assets with assets from other developers.
5. The remote server must be running when you invoke the deploy command. When you deploy or refresh your Java EE asset, the files will be copied across the network, saved as temporary files, and then deployed.

Debugging a remote application on an already running server

Before debugging the remote application on a running server, you have to start the server under jpda debugger using geronimo.sh/bat:

geronimo jpda run

or

geronimo jpda start

Use this procedure to attach a remote application (a WAR file in this example) to an already running server started in debug mode.

1. On the Eclipse menu bar, click **File** and **Import**. On the *Import* panel, expand **Web**, select **WAR file** and click **Next**.
2. Specify the WAR file directory, and the target runtime. Click **Next**.
3. Import Web libraries if needed. Click **Finish**.

4. Left-click the down arrow following the debug icon on the main tool bar, and select **Debug Configurations...**.
5. Select **Remote Java Application**, and click the 'New' button to create a new debug configuration.
6. Fill in the information about the server:
    - Name: as desired
    - Project: the WAR file you want to bebug on the server
    - Host: the server's hostname
    - Port: 5005 (or the appropriate value if the server's debug port has been changed.)
7. Click **Debug**.