User's Guide

iPlanet Application Server Integration for VisualCafé

November 2000

Contents

Preface **5**
How This Book Is Organized 5
Typographic Conventions 5
Related Documentation 6

Chapter 1 Introducing the iPlanet Integration with VisualCafé **7**
About the iPlanet Integration with VisualCafé 7
Getting Started: Installation and Tutorial 8
Identifying the Plugin Version 8
To find the version number of the plugin 8

Chapter 2 Developing iPlanet Applications **9**
Creating and Building Your Project 9
To Create a Web Application 9
To Add a Servlet To a Project 10
Configuring for Deployment 11
To Define the Target Server 11
To Add Target Servers 11
About Role Mappings 13
To Set Role Mappings 13
About Web Application Deployment Descriptor Files 16
To Configure a Web Application for Deployment 16
About Servlet Deployment Descriptor Files 18
To Configure a Servlet for Deployment 19
Running Applications 20
Creating a Project with Database Access 20
About Data Sources 20
To Create a Data Source 20
To Configure a Servlet for Database Access 23
Enabling EJB to EJB Calls 25

To Create a Project with Two Enterprise Java Beans 25
About Registering an EJB 28
To Register an EJB 28
To Deploy the EJB 30
To Set the Classpath 30
To Build the Web Application 30
To Locate the EJB for the Build 32
To Configure the Servlet to Communicate with EJBs 32
To Deploy and Run a Project with Database Access 33
Debugging iPlanet Projects 35
To Debug an iPlanet Project 35

Chapter 3 Menu Reference 37
iPlanet Functionality-Added Menu Items 37
Execute 38
Run in Debugger 38
Step Into 38
Start EJB Server 39
Shut Down EJB Server 39
Deliver Enterprise Bean to EJB Server 39
Configure Deployment Descriptor (for EJBs) 40
New Menu Items 41
Start Application Server 41
Shut Down Application Server 42
Execute in iPlanet Server 42
Deployment Configuration Menu Items 42
Configure iPlanet Web Application 42
General Tab 43
Context Tab 45
Environment Tab 45
Errors Tab 46
Session Tab 46
MIME Tab 48
Refs Tab 48
SecRoles Tab 50
Security Tab 50
Servlets Tab 52
Configure Servlets 53
General Tab 54
iAS Tab 54
InitParams Tab 56

Preface

The *iPlanet Application Server Integration for VisualCafé User's Guide* describes how to use the plugin for the WebGain VisualCafé Enterprise Edition integrated development environment (IDE). This manual is intended for the software developer, but can be used by anyone involved in application development for the iPlanet Application Server 6.0; including application integrators, administrators, and release engineers.

This manual assumes some understanding of iPlanet Application Server™, WebGain VisualCafé™ and the Windows NT operating system.

How This Book Is Organized

Chapter 1, Chapter 1, "Introducing the iPlanet Integration with VisualCafé," gets you started with an overview of this product and its features.

Chapter 2, "Developing iPlanet Applications," describes the development cycle, using the iPlanet Application Server Integration for VisualCafé with the VisualCafé IDE.

Chapter 3, "Menu Reference," provides a reference guide to the menu items that are added or modified in the VisualCafé IDE. These aid in your application development for deployment to the iPlanet Application Server.

Typographic Conventions

File and directory paths are provided in Windows NT format (with backslashes separating directory names).

URLs are used in the form http://*server.domain*/*path*/*file*.html

Where:

- *server* is the name of the server where you are running the application.

- *domain* is your internet domain name.
- *path* is the directory structure on the server.
- *file* is an individual filename.

The following table shows the typographic conventions used throughout iPlanet documentation.

Table 1 Typographic Conventions

Typeface	Meaning	Examples
`AaBbCc123`	The names of files, directories, sample code, and code listings; and HTML tags	Open `Hello.html` file. `<HEAD1>` creates a top heading.
AaBbCc123	Book titles, variables and placeholders, words to be emphasized; words used in the literal sense	Read Chapter 8of the *Programmer's Guide*. Enter your *UserID*. Enter *Login* in the Name field.
AaBbCc123	glossary terms	**Templates** are page outlines.

Related Documentation

You can access for the latest, updated documentation on the iPlanet Web site at:

`http://docs.iplanet.com/docs/manuals`

The installed iPlanet Application Server on your local system or network also contains documentation in:

`<YourInstallDir>/ias/docs/index.htm`

where `YourInstallDir` is the directory in which you installed the iPlanet Application Server.

Chapter 1

Introducing the iPlanet Integration with VisualCafé

This chapter gets you started with iPlanet Application Server WebGain Plugin by giving you an overview of the features and functionality it adds to the WebGain VisualCafé IDE. It contains the following sections:

- About the iPlanet Integration with VisualCafé
- Getting Started: Installation and Tutorial

About the iPlanet Integration with VisualCafé

When used in conjunction with the iPlanet Application Server Integration for VisualCafé, WebGain's VisualCafé Enterprise Edition, which is one of the leading Java IDEs in the marketplace, provides a complete development path for the iPlanet Application Server.

The iPlanet Integration with VisualCafé extends the VisualCafé development environment so that you can set iPlanet Application Server options for the application you are developing. The iPlanet Integration with VisualCafé allows you to do things in the VisualCafé IDE such as, start and stop the application server, create an iPlanet Web Application project, create an EJB project, set up your application's deployment descriptors, configure a data source for access, deploy and execute your application, as well as do local and remote debugging, test EJBs using RMI/IIOP clients, and so forth.

Getting Started: Installation and Tutorial

Before you can begin using the iPlanet-enabled VisualCafé IDE, you must install iPlanet Application Server 6.0 sp1, and a Web server (the iPlanet Web Server ships with iPlanet Application Server 6.0 sp1). Then, after you install the VisualCafé IDE on your machine, you can look for the online instructions for installing the WebGain plugin at:

`http://docs/iplanet.com/docs/manuals/ide/install.html`

In addition, there is an online tutorial, which takes you through building, deploying, executing and debugging two simple Web applications projects: "Hello World" and "Sales Tax Calculator." Using the tutorial gives you some experience in using most of the iPlanet Integration with VisualCafé features. You can find the tutorial at:

`http://docs/iplanet.com/docs/manuals/ide/wgtutorial.htm`

Finally, the latest information about this release can be found at:

`http://docs/iplanet.com/docs/manuals/ide/releasenotes.html`

Identifying the Plugin Version

Because this product is a plugin extension to VisualCafé, use the following procedure to find its version number.

To find the version number of the plugin

1. From VisualCafé menus, choose Tools > Select Jar Viewer.
2. Select the `iplanet_wg.jar` file.

   This file is installed in <*VisualCafeInstalldir*>`\Bin\Components`.
3. Double click on the `WebAppProjectPlugin.class` file.

   The version number is in the Manifest Display area.

Chapter 2

Developing iPlanet Applications

This chapter describes what you can do using the iPlanet Application Server Integration with VisualCafé. The following sections take you through the process of developing an iPlanet Application Server- enabled application in VisualCafé:

- Creating and Building Your Project
- Configuring for Deployment
- Running Applications
- Debugging iPlanet Projects

Creating and Building Your Project

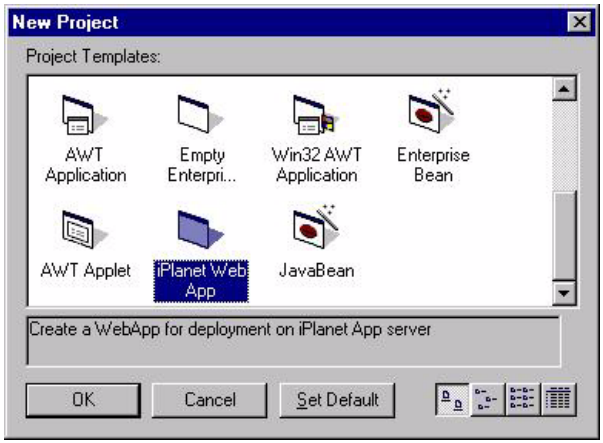
The first step in building an iPlanet Web-enabled application, is to create a project and add the necessary modules to it.

To Create a Web Application

1. From the File menu, select New Project.

The window in the following illustration appears.

Figure 2-1 The iPlanet Web App project icon is shown highlighted.



2. Double-click the iPlanet Web App icon and name your project `.vep` file.

   Your project template appears in a project window.

3. Choose File > Save As to update the title on the Project window title bar.

To Add a Servlet To a Project

After you have a project, you can create and insert a servlet into it.

1. Make sure that you have a project open.
1. From the VisualCafé IDE, choose Insert > Servlet.
2. Read the screen that describes what a Servlet wizard does and click Next.
3. When the Servlet window appears, enter a servlet name and a package name.
4. Click Finish.

   The Java template for the servlet is added to your project.

5. To edit the Java file, double-click the file name in the Project window.

   A code editor opens.

6. Type in any changes and save them.

Configuring for Deployment

The iPlanet Application Server Plugin extensions to VisualCafé allow you to configure your iPlanet Web applications, servlets, and EJBs for deployment by using dialogs to create the necessary deployment descriptor files (DTDs). You can also globally define the deployment targets, data sources, and role mapping attributes that can then be easily set from each project.

You can set multiple attributes of the deployment target from the Environment Options > UML/EJB tab's Manage Deployment Target dialog. The following procedures describe how:

- To Add Target Servers
- To Set Role Mappings
- To Create a Data Source

To Define the Target Server

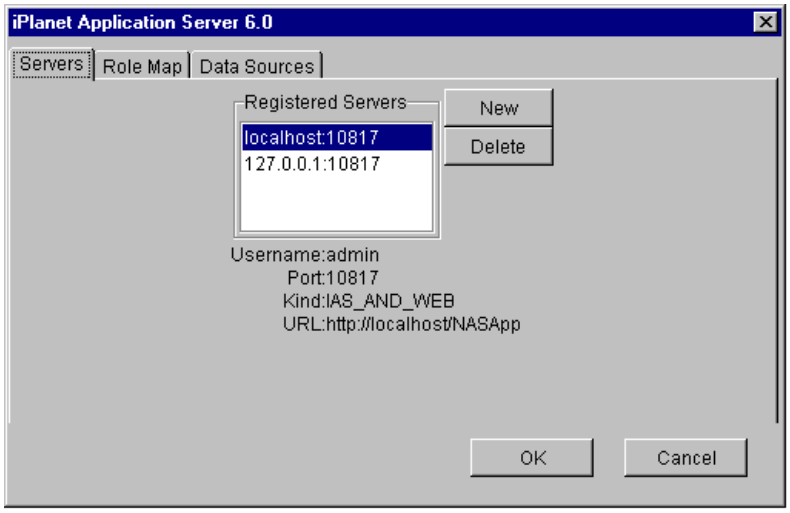
1. Choose Tools > Environment Options.
2. Select the UML/EJB tab.
3. Keep iPlanet Application Server 6.0 as the default in the Deployment Target drop-down menu.

To Add Target Servers

1. Choose Tools > Environment Options.
2. Select the UML/EJB tab.
3. Click the Manage Deployment Target button.

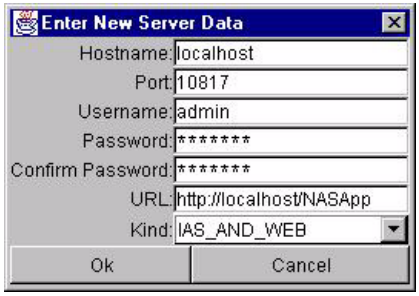
   The iPlanet Application Server 6.0 Deployment Target dialog appears.

Figure 2-2 You can identify servers with their names or IP addresses.



4. Click New.

The Enter New Server Data dialog appears.



5. Enter the following data in each field:

Field	Information to enter
Hostname	Server's name.
Port	The admin port of your application server; default is 10817.

Username	The iPlanet Application Server administrator username
Password	The administrator password
Confirm Password	Repeat administrator password
URL	The base URL for your server. The default is `http://hostname/NASApp`
Kind	IAS_AND_WEB

6. Click OK.

   This dismisses the dialog and enters your server in the list of registered servers.

7. Click OK again to close the Manage Deployment Target dialog.

   This records the global setting that identifies the selected server as the default server to be used for new projects. It also registers the servers, which are listed on the Servers tab of the Deployment Target dialog, with the VisualCafé IDE so that they are available to all your development projects.

About Role Mappings

If you are implementing security features in your Web application, then before you deploy an application, the security roles must be bound to groups and users that are known to the server. This is for authentication control of user access to a method. This role mapping information is not part of the overall application's deployment descriptors, since it can be different for different servers where the application might be deployed. This binding does not modify the application module.

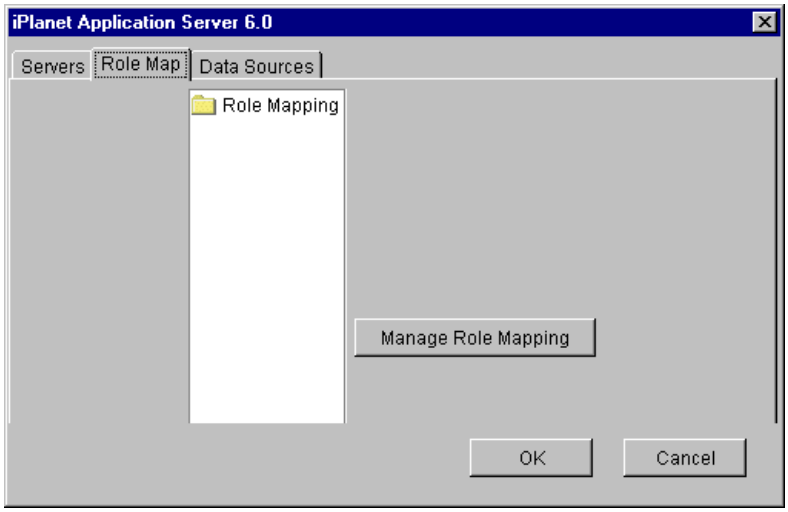
NOTE For more information about security features see, the J2EE specification on the Sun.com Web site.

To Set Role Mappings

1. Choose Tools > Environment Options.
2. Select the UML/EJB tab.
3. Click the Manage Deployment Target button.

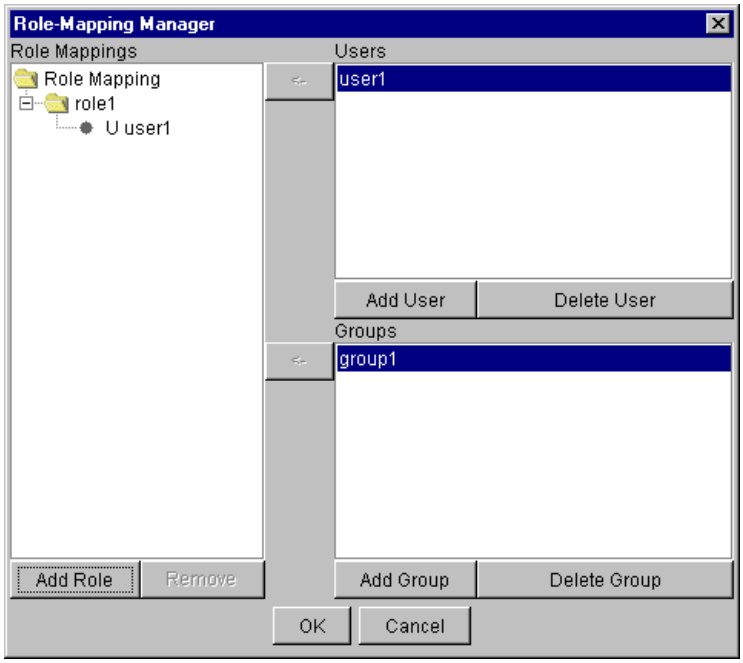
4. Select the Role Map tab.

   The Role Map panel appears.



5. Click Manage Role Mapping.

   The Role-Mapping Manager appears.



6. Click Add Role.
7. In the Input dialog that appears, enter the role you require and click OK.

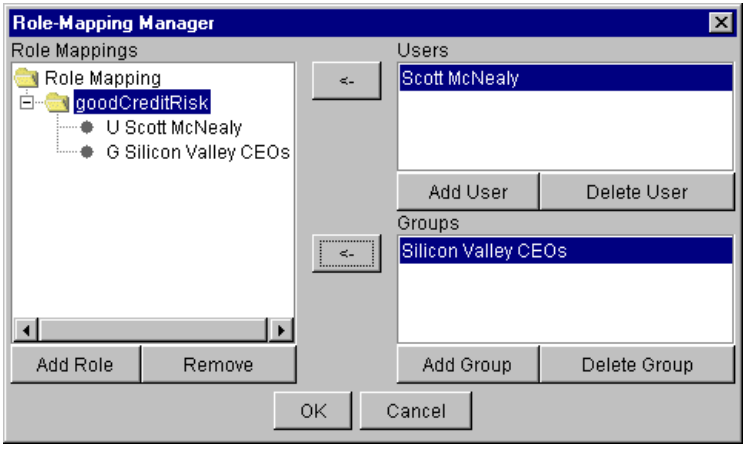
   The new role appears in the Role Mappings tree structure.
8. Similarly you add a user or group by clicking on Add User or Add Group.
9. To map a user or group to a role:

- Click on the role you want to add them to
- Select the user or group to add

NOTE This can only be done in the context of a project.

- Click the arrow button next to the selected user or group.

  The user and group you mapped to the selected role, are shown under the role they to which they are mapped, with a U or G next to the name to signify User or Group.



About Web Application Deployment Descriptor Files

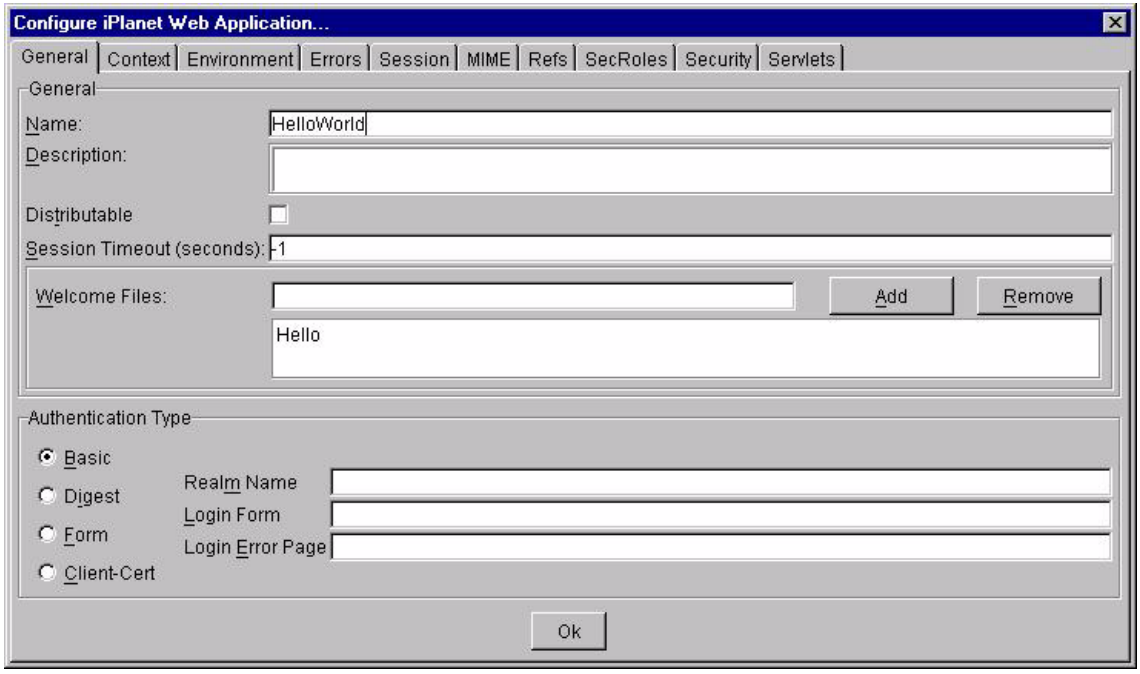
After you have specified the files for the Web Application module, you must specify deployment descriptor information at the module-level. You do this using the Configure iPlanet Web Application deployment descriptor (DD) graphical editor. The Configure iPlanet Web Application DD editor consists of the tabs shown in Figure 2-3.

To Configure a Web Application for Deployment

1. Select Project > Configure iPlanet Web Application.

The following dialog appears:

Figure 2-3 The tabbed panels allow you to set up the deployment descriptor files for a Web application.



2. Declare deployment descriptors as required:

- General - Specifies overall attributes for your Web Application.
- Context - Declares the Web Application's servlet context initialization parameters.
- Environment - Identifies environmental elements specific to your Web Application.
- Errors - Maps error codes to a resource within the Web Application.
- Session - Configures the Web Application targeted to your operational iAS environment.
- MIME - Identifies the mapping between MIME types and MIME Extensions.
- Refs - Identifies EJB and resource factory references in the source code of any Web Application component.
- SecRoles - Declares security roles that are actually used within the organization using this Web Application.
- Security - Specifies security constraints to protect web content by defining a resource collection.

3. Servlets - Specifies the mapping between servlets and the URL pattern that invokes them.

NOTE See “Configure iPlanet Web Application,” on page 42 for more information.

4. Choose File > Save and the deployment descriptor declarations are written to the XML files and your Web Application module is complete.

About Servlet Deployment Descriptor Files

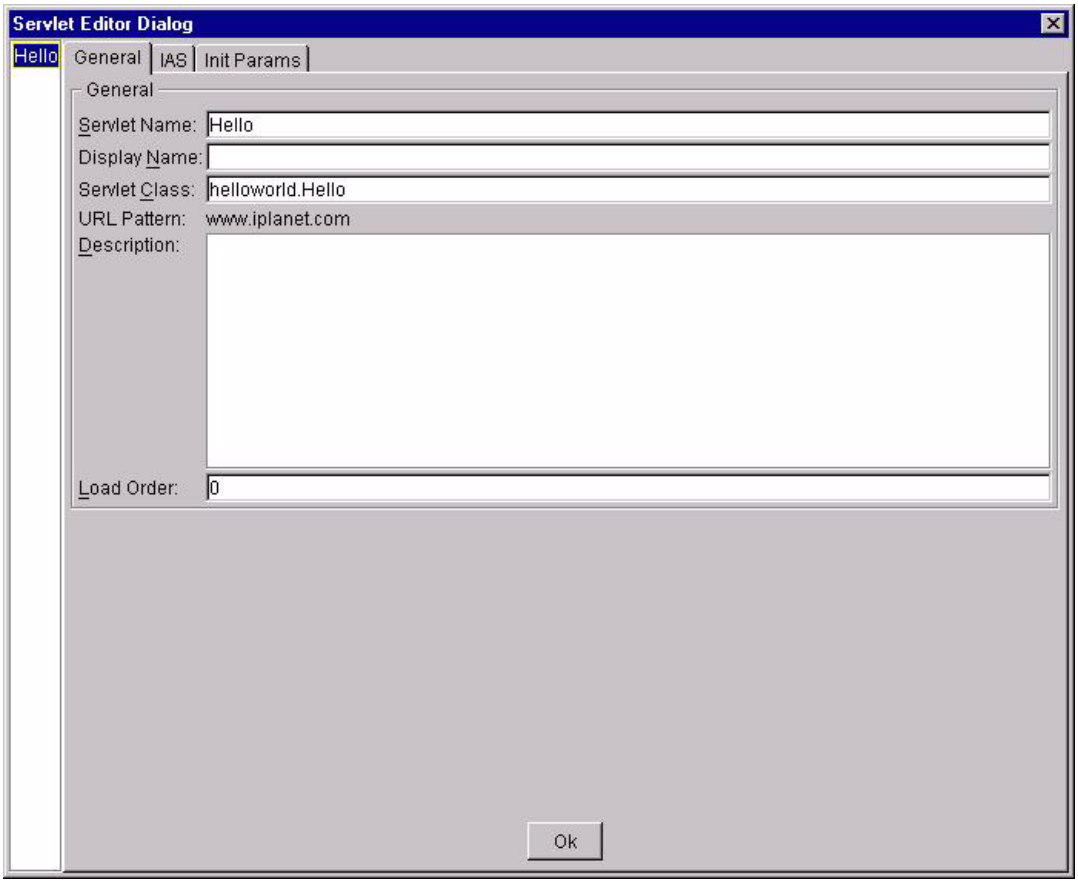
Once you have inserted the components into the Web Application module, you are ready to declare the deployment descriptors. Using the Configure iPlanet Web Application dialog, deployment descriptors are defined at the servlet component level, as well as, at the application module level of a Web Applications. The deployment descriptor declarations are written to a WAR files, as well as web.xml and an ias-web.xml file in the \WEB-INF directory at the root of the archive.

NOTE You need to only declare deployment descriptors for any servlets that a user can directly access. The iPlanet Integration creates a default deployment descriptor for each servlet in a module.

To Configure a Servlet for Deployment

1. From the VisualCafé IDE, select Project > Configure Servlets.

   The following dialog appears:



2. Declare the deployment descriptors for servlets as required:

- General tab panel - Specifies overall properties for a servlet in the Web Application.
- iAS tab panel - Define iAS-specific options for the servlet according to your targeted operational iAS environment.
- Init Params tab panel - Specifies initialization parameters for a servlet in the web application.

3. Choose File > Save and the deployment descriptor declarations are written to the XML files.

NOTE See the menu item “Configure Servlets,” on page 53 for more information.

Running Applications

The iPlanet Integration with the VisualCafé IDE enables you to create applications that can run locally or remotely.Applications must be configured correctly before you can deploy and run them on the iPlanet Application Server. The following examples describe:

- Creating a project in which you configuring a servlet for database access before deploying and running it
- Configuring to enable communication between two EJBs

Creating a Project with Database Access

The following procedures show you the basics of how to create a data source and how to configure your project to work with a database.

About Data Sources

You use a data source in an application to encapsulate the database connection. Before you can use a data source on the application server, it must be registered there when the Web application is deployed. When you add a data source to your Web app, the configuration is registered on the application server.

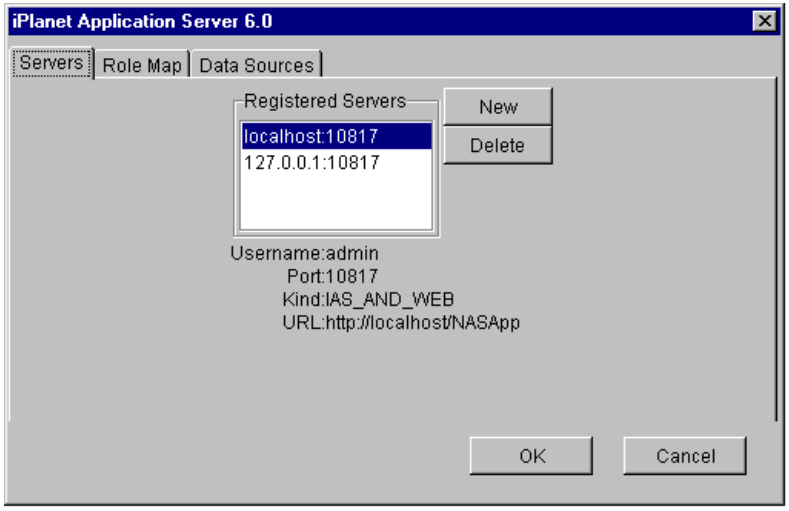
To Create a Data Source

This procedure creates a datasource referring to your database. It uses Oracle as the database.

1. Open a project that has a servlet already included.
2. Select Tools > Environment Options.
3. Click on the UML/EJB tab and select iPlanet Application Server 6.0 as the Deployment Target from the drop-down menu.

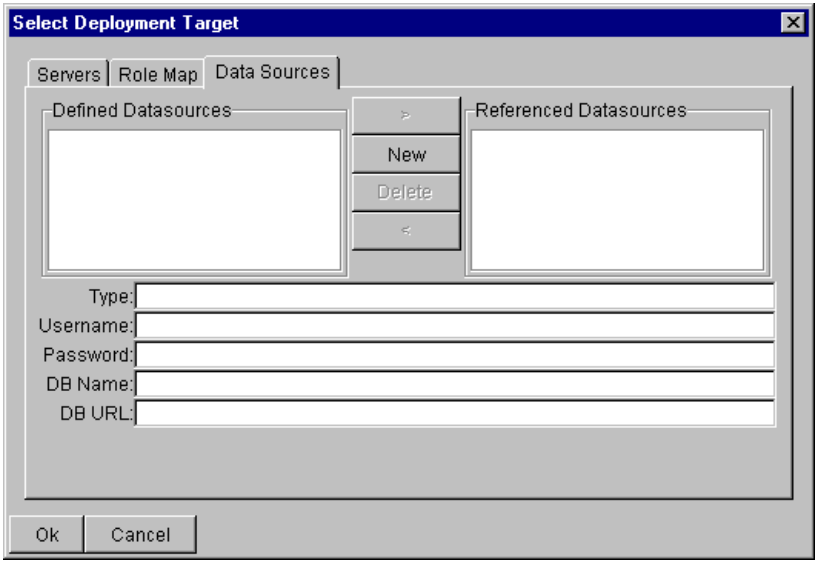
4. From the same panel, click Manage Deployment Target.

   The Manage Deployment Target dialog appears.



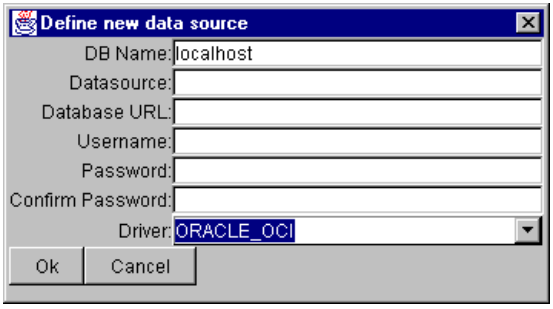
5. Click on the Data Sources tab.

   The following panel appears.



6. Click New on the Data Sources panel.

   The Define New Data Source dialog appears.



7. Enter the following information in the Define New Data Source dialog:

Field	Information to enter
Database Name	Database's name.
Datasource	Identifier you want to use to refer to the data source.
Database URL	JDBC URL for 3rd party driver. This is not needed for iPlanet Application Server database access engine.
Username	The username for the database.
Password	The password for the database username.
Confirm Password	Repeat username password
Driver	ORACLE_OCI (or name if third party used with DB URL)

NOTE iPlanet Application Server has built-in JDBC drivers, as well as providing support for third-party JDBC drivers.

8. Click OK.

   This defines the data source globally in the VisualCafé IDE and it appears in the Referenced Datasources list of the Select Deployment Target Data Source panel. Once you add the data source to a project and deploy that project, that data source is registered in the iPlanet Application Server globally. All applications that are subsequently deployed to that instance of the application server can also access that data source.

To Configure a Servlet for Database Access

1. Select the datasource from the Referenced Datasources list of the Select Deployment Target Data Source panel.
2. Click the Servers tab and select the server you want to deploy to.
3. Click OK.
4. Replace the code in the servlet with the following code. This code performs the JNDI lookup of the datasource.

Code Example 2-1 This code makes a connection to the database and includes code to make inserts and deletes to a row.

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;

import javax.naming.NamingException;
import javax.sql.*;
import javax.servlet.*;
import javax.servlet.jsp.JspException;
import javax.servlet.http.*;

public class webservlet extends HttpServlet
{
   static private Connection con;
   static private String dbName = "java:comp/env/cdx";


   public void doGet(HttpServletRequest req, HttpServletResponse
                                                          resp)

      throws ServletException, IOException

       {
            resp.setContentType("text/html");
            PrintWriter out = new
                            PrintWriter(resp.getOutputStream());
```

```
// to do: code goes here.

          try {
             makeConnection();
             insertRow("Test Product", 10.0, "B");
             }

          catch(Exception e){
                    e.printStackTrace();
                    }

             out.println("<HTML>");

out.println("<HEAD><TITLE>webservletOutput</TITLE></HEAD>");
             out.println("<BODY>");

// to do: your HTML goes here.
             out.println("</BODY>");
             out.println("</HTML>");
             out.close();

        }

private void makeConnection()
       throws NamingException, SQLException
       {InitialContext ic = new InitialContext();
          DataSource ds = (DataSource) ic.lookup(dbName);
                                        con =  ds.getConnection();
       }

private void insertRow (String desc, double price, String id )
       throws SQLException

        {
String insertStatement ="insert into product values ( ? , ? , ?)";
          PreparedStatement prepStmt =
                          con.prepareStatement(insertStatement);

          prepStmt.setString(1, desc);
          prepStmt.setDouble(2, price);
          prepStmt.setString(3, id);
          prepStmt.executeUpdate();
          prepStmt.close();
          }

private void deleteRow(String id) throws SQLException {
       String deleteStatement ="delete from product where id = ? ";
```

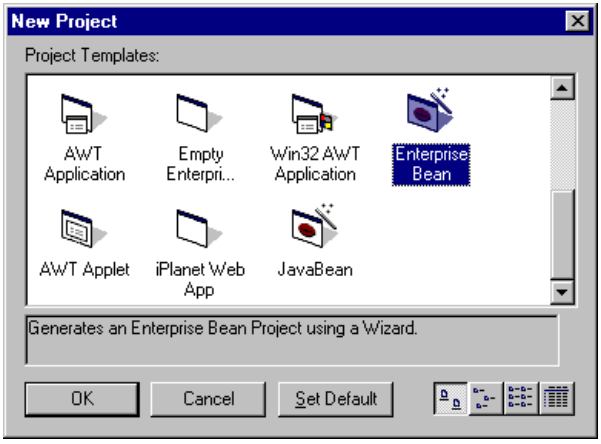
```
        PreparedStatement prepStmt =
        con.prepareStatement(deleteStatement);
        prepStmt.setString(1, id);
        prepStmt.executeUpdate();
        prepStmt.close();
    } }
```

Enabling EJB to EJB Calls

The following procedures enable an iPlanet Web Application project to call an EJB in another project. That EJB then calls another EJB in that project. To reproduce this example, you must create two projects, an EJB project which contains two Enterprise Java beans, and an iPlanet Web Application project, which has a servlet code that calls an EJB in the other project.

To Create a Project with Two Enterprise Java Beans

1. From VisualCafé, choose File > New Project.
2. In the New Project window, select the Enterprise Java Bean icon, as shown.



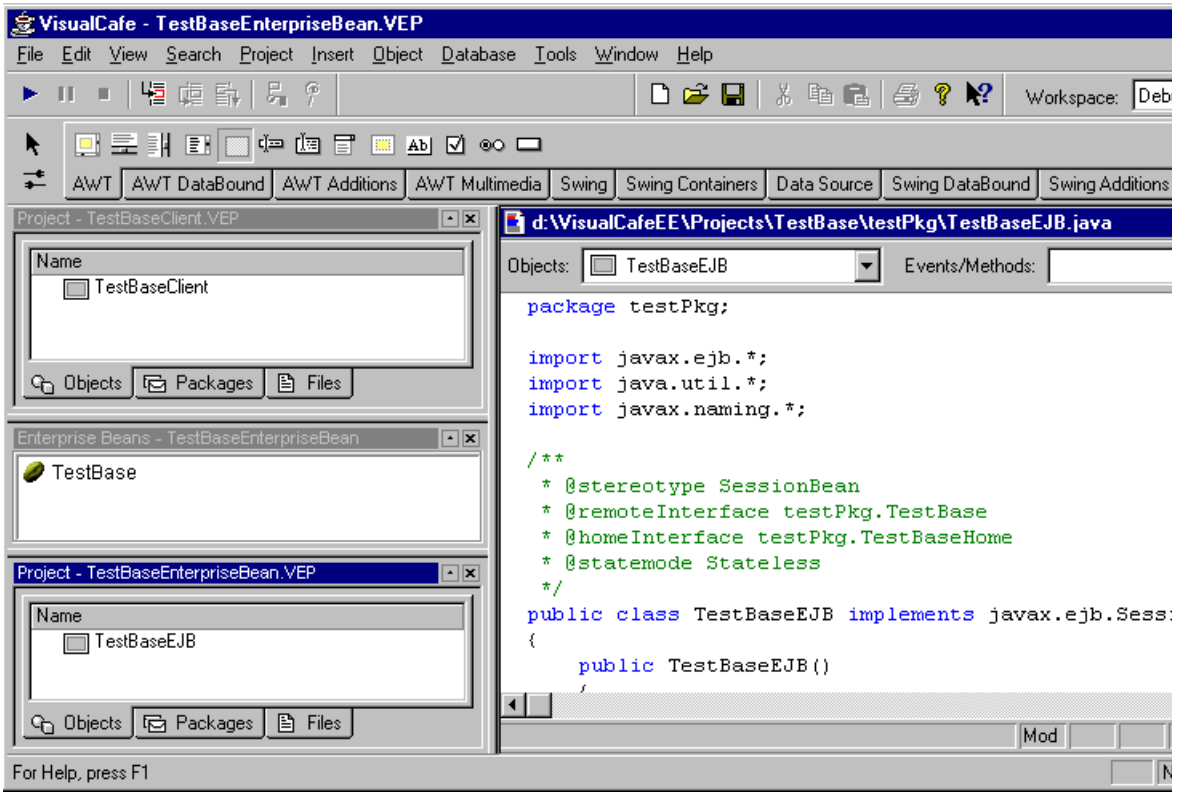
3. In the Enterprise Bean wizard, enter:

- `TestBase` in the Base Name field
- `testpkg` in the Package Information field

4. After clicking Next, check to see that the Session-Stateless radio button is selected.

5. Leave Session-Stateless selected and click Finish.

   The stateless session bean is created and shows in the VisualCafé IDE.



6. Click the Files tab of the `TestBaseEnterpriseBean.vep` project window and open the `TestBaseEJB.java` code window. Locate the `import` statements and add:

   ```
   import java.rmi.*;
   ```

7. In the same `TestBaseEJB.java` code window, create the method `testBaseHello` by entering the following code:

   ```
   public String testBaseHello() throws RemoteException
   {
      String Message = " This is the test message.";
      return Message;
   }
   ```

8. Add the following method code to the `TestBase` Enterprise Java beans' remote interface, `TestBase.java`:

```
// Make sure the remote interface also has the method
// testBaseHello.

public interface TestBase extends EJBObject
{
public string testBaseHello()throws RemoteException;
}
```

9. Choose Insert > Enterprise Bean to create and insert a second Enterprise Java bean into this project.

10. Enter the following information in the Enterprise Bean wizard:

- Base Name: `Caller`
- Package Information: `testpkg`

11. Click Next, check to make sure that the Session-Stateless radio button is selected, then click Finish.

    The second EJB, `Caller`, appears in the `TestBaseEnterpriseBean.vep` project.

12. Open the `CallerEJB.java` code window. Locate the `import` statements and add:

```
import java.rmi.*;
```

13. In the same code window, add the method, `callerHello`, by entering the following code:

```
public String callerHello() throws RemoteException

{
System.out.println("Now calling the testBaseHello method.");
String JNDI_NAME = "java:comp/env/TestBase";
Object beanObject = null;

 try
{
   Context ctx = new InitialContext();
   beanObject = ctx.lookup(JNDI_NAME);
   testpkg.TestBaseHome home =(testpkg.TestBaseHome) beanObject;
   testpkg.TestBase remote = home.create();
   String testBaseMessage = remote.testBaseHello();
   System.out.println("The TestBase Message is :
   " + testBaseMessage);
   return(testBaseMessage);
}

 catch(Exception ex)
{
   ex.printStackTrace();
}
```

```
    return ("this is not the Message I want after calling an ejb
    from ejb");
}
```

14. Add the following method code to the `Caller` Enterprise Java beans' remote interface, `Caller.java`:

```
public interface caller extends EJBObject

{
    public String callerHello() throws RemoteException;
}
```

15. Choose the menu item Project > Build Application.

    Look for the "Build Successful" message in the Messages window.

About Registering an EJB

For `Caller` to communicate with `testBase`, `testBase` must "know" about the other EJB. With the iPlanet Application Server Integration with VisualCafé, this is done by entering the name of the EJB you wish to register into the references list of the bean you want to register it with.

To Register an EJB

1. Click on the Objects tab of the Project panel in the VisualCafé IDE and select the `CallerEJB`.
2. With `CallerEJB` selected, choose menu item: Project > Configure Deployment Descriptor.

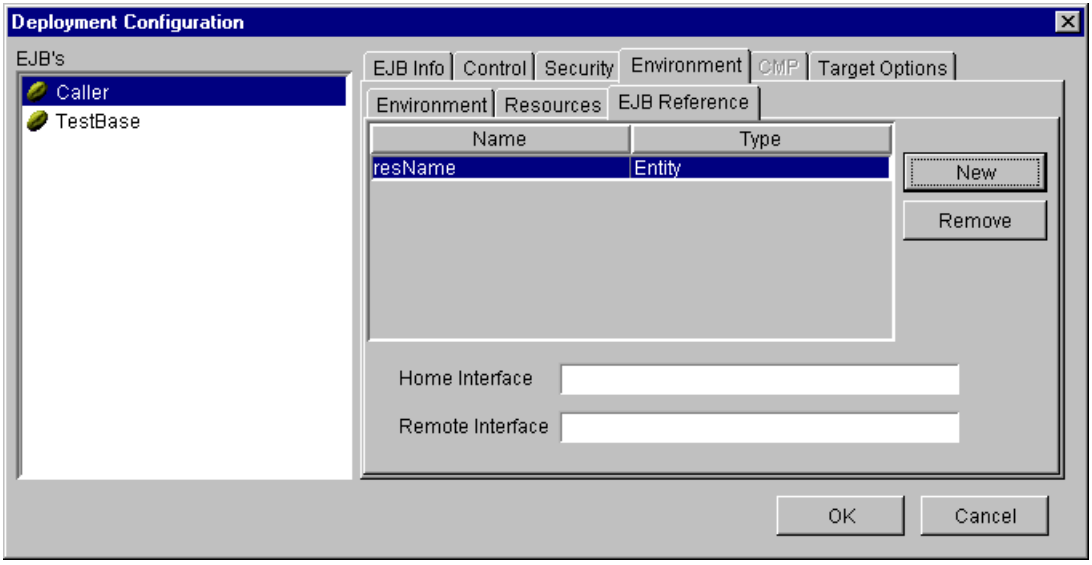
   The Deployment Configuration dialog box appears with `Caller` highlighted in the EJB's list.
3. Click on the Environment tab.

   Another row of tabs appears on the Environment panel, below the top row of tabs.
4. Click on the EJB Reference tab.

   The entries, which you use to register the EJB, appear on this panel.

5. Click New.

   A highlighted text entry appears in the EJB Reference list box, where you should replace the placeholder text with the real name and type of the EJB you're registering.



6. Enter the following information:

- Name (of the EJB you want to register): `TestBase`
- Type: `Session`
- Click the Home Interface text box and select, `testpkg.TestBaseHome`, from the drop-down menu that appears.
- Click the Remote Interface text box and select, `testpkg.TestBase`, from the drop-down menu that appears.

7. Select the Target Options tab, then click on the EJB Reference tab that appears under the Target Options tab.

8. Enter the following information:

- EJB Reference Name: `TestBase`
- JNDI Name: `ejb/TestBaseEnterpriseBean/TestBase`

9. Press Enter and click OK.

   The EJB is registered.

To Deploy the EJB

1. Check that your server is configured by choosing Tools > Environment Options.
2. Click the UML/EJB tab.

   If you installed and are running iPlanet Application Server, it is highlighted.
3. Click Manage Deployment.
4. Select the server configuration for a local host, which is assumed to be set up.
5. Click OK and Manage Deployment dialog closes. Click OK again, and the Environment Option dialog closes.
6. If the application server is not running, you can start it by selecting Project > Start EJB Server.
7. Choose Project > Deliver Enterprise Bean, to deploy `TestBaseEnterpriseBean.jar` to the application server.
8. Add `TestBaseEnterpriseBean.jar` to the CLASSPATH of the application server.

To Set the Classpath

1. Run `regedit` from a command prompt on the server to which you are deploying.
2. Find the entry:

   ```
   HKEY_LOCAL_MACHINE->SOFTWARE->iPlanet->Application
   Server->6.0->Java->ClassPath
   ```
3. Edit this entry by adding the path to the deployed servlet to the end of the classpath. The added path should look like this:
   `d:\`*`YourInstallDir`*`\iAS6\ias\JAR\`*`TestBaseEnterpriseBean.`*`jar;`
4. Click OK to commit your changes.
5. Restart the application server from the Windows Control Panel Services interface.

   If you followed all the steps to this point, you successfully deployed an Enterprise bean JAR file, which consists of two Enterprise beans, to the application server. The following procedure builds a servlet that has the sole purpose of calling one of the EJBs.

To Build the Web Application

1. From the VisualCafé IDE, close any currently open projects.
2. Select File > New Project.

3. In the New Project window, select the iPlanet Web App icon and Click OK.
4. Name the project: `TestBaseCall.vep`
5. Choose Insert > Servlet.
6. On the appropriate wizard panel, set the servlet name to: `TestBaseServlet`
7. Go through the rest of the wizard panels, keeping all of the default choices.
8. Add the following lookup code to the `TestBaseServlet.java` code file:

```
public void doPost(HttpServletRequest req, HttpServletResponse resp)
                  throws ServletException, IOException

   {
   resp.setContentType("text/html");
   PrintWriter out = new PrintWriter(resp.getOutputStream());
                  String JNDI_NAME = "java:comp/env/Caller";
                  Object beanObject = null;

   // to do: code goes here.

   out.println("<HTML>");
   out.println("<HEAD><TITLE>TestBaseServlet
                                          Output</TITLE></HEAD>");
   out.println("<BODY>");

   try
   {
      Context ctx = new InitialContext();
      beanObject = ctx.lookup(JNDI_NAME);
      testpkg.CallerHome home = (testpkg.CallerHome)beanObject;
      testpkg.Caller remote = home.create();
      String callerMessage = remote.callerHello();
      System.out.println("Caller's message is:" + callerMessage);
      out.println("<P><H2> An EJB is calling an EJB!<h2><p>");
      out.println("<p><h2>" + callerMessage + "<p><h2>");
   }
   catch(Exception ex)
   {
      ex.printStackTrace();
   }

   // to do: your HTML goes here.

      out.println("</BODY>");
      out.println("</HTML>");
      out.close();
   }
```

9. To build this servlet, you must make sure that the EJB class files are in the class path.

To Locate the EJB for the Build

1. Choose Project > Options.
2. Click the Options tab.
3. Click the Directories tab.
4. From the Show Directories For drop-down menu, select Input Class Files.
5. Add the directory which contains the EJB classes as the entry at the top of the list. In this example it is:

   ```
   d:\VisualCafeEE\projects\TestBase;
   ```

To Configure the Servlet to Communicate with EJBs

1. Choose Project > Configure iPlanet Web Application.
2. Click the Refs tab of the Configure iPlanet Web Application dialog.
3. Under References for EJBs Defined Elsewhere, click Add to enter the information for the first of the two bean references:

- Reference: `TestBase`
- Bean Type: `Session`
- Linked to Bean: `TestBase`
- Bean Home Interface: `testpkg.TestBaseHome`
- Bean Remote Interface: `testpkg.TestBase`
- JNDI Name: `ejb/TestBaseEnterpriseBean/TestBase`

4. Click Add to enter the information for the second of the two bean references:

- Reference: `Caller`
- Bean Type: `Session`
- Linked to Bean: `Caller`
- Bean Home Interface: `testpkg.CallerHome`
- Bean Remote Interface: `testpkg.Caller`
- JNDI Name: `ejb/TestBaseEnterpriseBean/Caller`

5. Select the menu item, Project > Configure Target Server.

   Check to make sure that your application server is the Target Server. If your server is not registersted, click New and enter the appropriate information.

6. Click OK to save the web application configuration file.

   Select Project > Execute in iPlanet Server, to deploy the Web Archive (WAR) file onto the application server and run the example.

TIP To troubleshoot your project:

- Check that the JAR file (`TestBaseEnterpriseBean.jar)` is in the server classpath.
- See that the browser has an entry of the form:

  `http://hostname/iASApp/webModuleName/servletName`

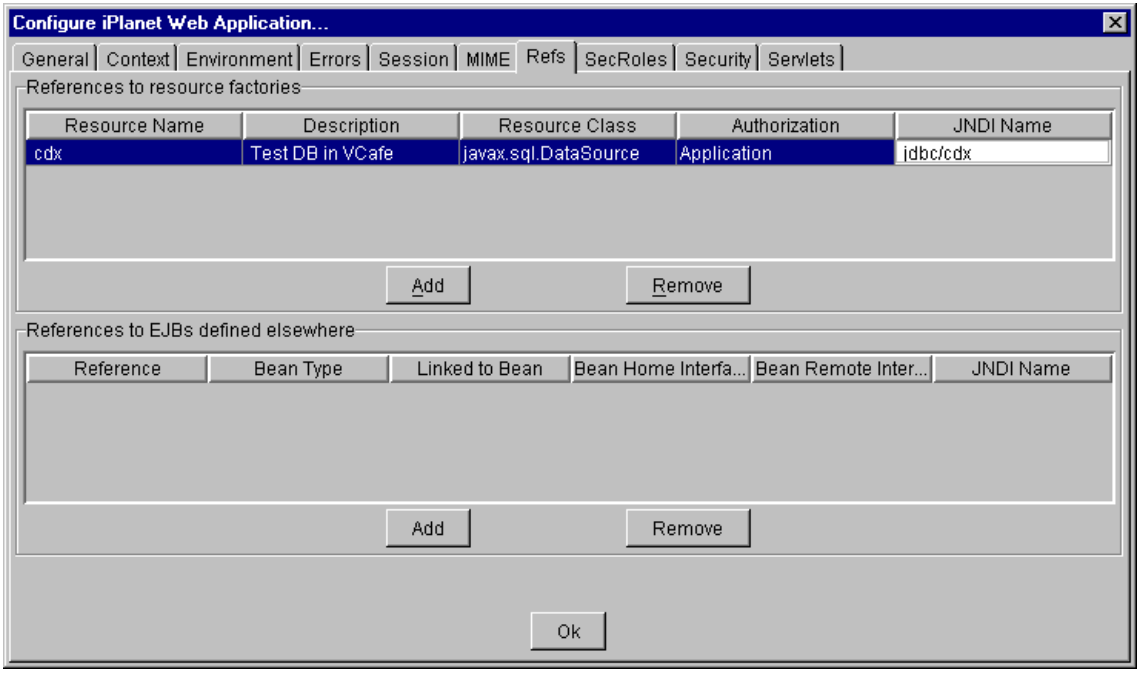
To Deploy and Run a Project with Database Access

1. Open the Web Application Project you configured for database access.
2. Select Project > Configure iPlanet Web Application.

   The Configure iPlanet Web Application dialog (for deployment settings) appears.

3. Select the Refs tab and click Add.

   The Add New Data Sources dialog appears.



4. Add the datasource definition:

- Resource Name: <DataSourceName>
- Description - Test DB in WebGain
- Resource Class- javax.sql.DataSource
- Authorization - Application
- JNDI Name - jdbc/<*DataSourceName*>

5. Click OK.
6. Choose Configure Servlet you see that the servlet name is there.
7. Select > Start Application Server. (If the menu item is disabled, this means that your server is running.)

   This starts the server selected in Managed Deploy Targets.

8. Select Project > Execute in iPlanet Server.

NOTE Make sure that the URL in the browser is of the form:
`http://host_name//NASApp/web_module_name/servlet_name`

Debugging iPlanet Projects

These instructions work for any Web Application.

To Debug an iPlanet Project

1. Make sure the iPlanet Application Server is running. Under Project, select Start Application Server. If the menu is disabled it means the server is already running.
2. Select Project > Run in Debugger (or press the F5 key) to launch the Java engine.
3. Select View > Messages

   The Messages window appears.
4. Look for the message:

   ready: <*port number*>

   This indicates that the server is ready to take requests.
5. Set at least one breakpoint or specify other debug options within VisualCafé.

NOTE From a source file you can set a breakpoint by selecting a line, then right clicking and selecting the Set > Breakpoint.

You can set a breakpoint in any project. If you want to debug an EJB you need to start the debugger from a Web App, but you can set a breakpoint in the EJB project.

6. Open your browser to the start page of your application and exercise the servlets you want to debug. For web applications the URL is of the form:

   `http://hostname/NASApp/projectname/servletname`
7. The debugger stops at your breakpoint. Use any of the debugging features of VisualCafé you would like, such as: Step Over, Continue to Cursor, or Continue.
8. Once you are done with the debugger, select Stop under the Debug menu to end your debugging session.

Chapter 3

Menu Reference

The iPlanet Application Server Integration with VisualCafé modifies the VisualCafé IDE through the menuing system. This chapter is a reference guide to the menus that are changed or added to that development environment. It includes the following sections:

- iPlanet Functionality-Added Menu Items
- New Menu Items
- Deployment Configuration Menu Items

iPlanet Functionality-Added Menu Items

The VisualCafé IDE menu functionality that is changed by the iPlanet Application Server Integration with VisualCafé, include:

- Execute
- Run in Debugger
- Step Into
- Start EJB Server
- Shut Down EJB Server
- Deliver Enterprise Bean to EJB Server
- Configure Deployment Descriptor (for EJBs)

Execute

If this is the first time you Execute an iPlanet Web App project, a dialog appears that asks you to enter the User Name and Password for the administrator of the local application server. It then registers your local iPlanet Application Server and makes it the default server for all your projects. In addition, this command packages the project in the `<appName>.war` file needed to run your application on the iPlanet Application Server. It then deploys the package to run on the application server.

This command kills the Java Server Process (kjs), if one is running, and then opens a new DOS window containing a Java Virtual Machine (JVM) which creates a new kjs process in the application server; that way only one process is running. A browser is then opened to the start of your Web App (as shown in the Welcome Files list that is on the Projects > Configure iPlanet Web Application dialog.)

NOTE This command uses the local server.

Run in Debugger

Choosing Project > Run in Debugger, switches VisualCafé into debug mode. While this mode is activated, you can debug your program by using the Source window, Class Browser, Breakpoints window, Variables window, Calls window, Threads window, and Messages window.

As you find and correct errors, you can leave debugger mode and rebuild the project. Once you have recompiled, you can restart the debugger. For more information on run-time editing with the VisualCafé Expert Edition, see Incremental debugging features in the VisualCafé online help.

The iPlanet Integration modifications cause this command to do the same as the Execute command with a couple of differences: instead of starting the application in a new window it starts your project in the VisualCafé debugger and no browser window opens.

NOTE This command uses the local server.

Step Into

Choosing Project > Step Into begins running a program in the debugger by stopping at the first line of source code. For an iPlanet project, this means that the application runs up to go to the first line of the first class in the project that starts the engine.

NOTE This command uses the local server.

Start EJB Server

If the registered server is not already running, this command starts iPlanet Application Server as a Windows NT Service on the local machine. This command is only active in an EJB project.

NOTE This command uses the local server.

Shut Down EJB Server

If the application server is started as a Windows Service on the local machine from an EJB project, this shuts down the server. This command is only active in an EJB project.

NOTE This command uses the local server.

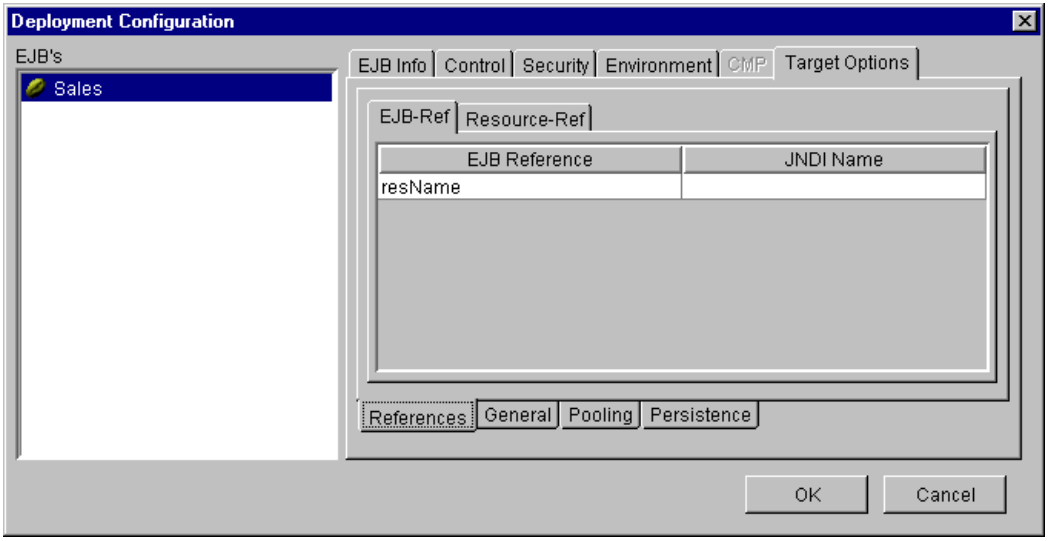
Deliver Enterprise Bean to EJB Server

This command compiles the EJB project, builds the stubs and skeletons, and creates and delivers the Enterprise Java bean JAR file, to the iPlanet Application Server. The WebGain progress panel informs you of where the application is in its process of deploying and delivering the JAR file and cleaning up. These activities are also reflected in the VisualCafé Message window as they occur. You can find the resulting JAR file in the JAR directory of your iPlanet Application Server.

NOTE This command uses the server which is specified in the current EJB project, instead of the server set in the Environment Options.

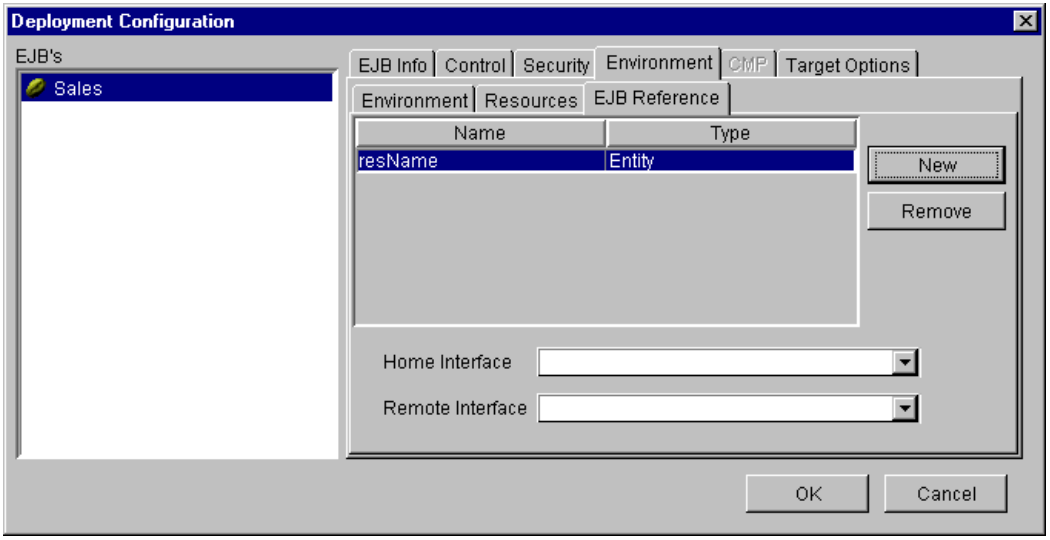
Configure Deployment Descriptor (for EJBs)

When a project is open that has an EJB implemented, the Project > Configure Deployment Descriptor dialog displays the Deployment Configuration dialog shown below



The Configure Deployment Descriptor dialog, is a dialog that is implemented by the VisualCafé IDE. The Target Options tab panel is added by the iPlanet Integration with VisualCafé. These tabbed panels allow you to set deployment descriptors specific to your EJB.

Before you can add an EJB Reference to the Target Options, you must add it to the VisualCafé Environment EJB Reference panel shown below. For more information, see the VisualCafé Online Help.



New Menu Items

Menus that are added to the VisualCafé IDE, specifically to enable iPlanet Application Server functionality include:

- Start Application Server
- Shut Down Application Server
- Execute in iPlanet Server

Start Application Server

If the registered application server is not running, this menu item is active. By default it is set to start the iPlanet Application Server.

Shut Down Application Server

If the registered application server is running, this menu item is active. By default it is set to shut down the iPlanet Application Server.

Execute in iPlanet Server

Builds, deploys, and executes the current project in the iPlanet Application Server.

NOTE This command uses the server which is specified in the current EJB project, instead of the server set in the Environment Options.

Deployment Configuration Menu Items

Web Applications and servlets have corresponding deployment configuration menu items that are added to the VisualCafé IDE by the iPlanet Integration, including:

- Configure iPlanet Web Application
- Configure Servlets

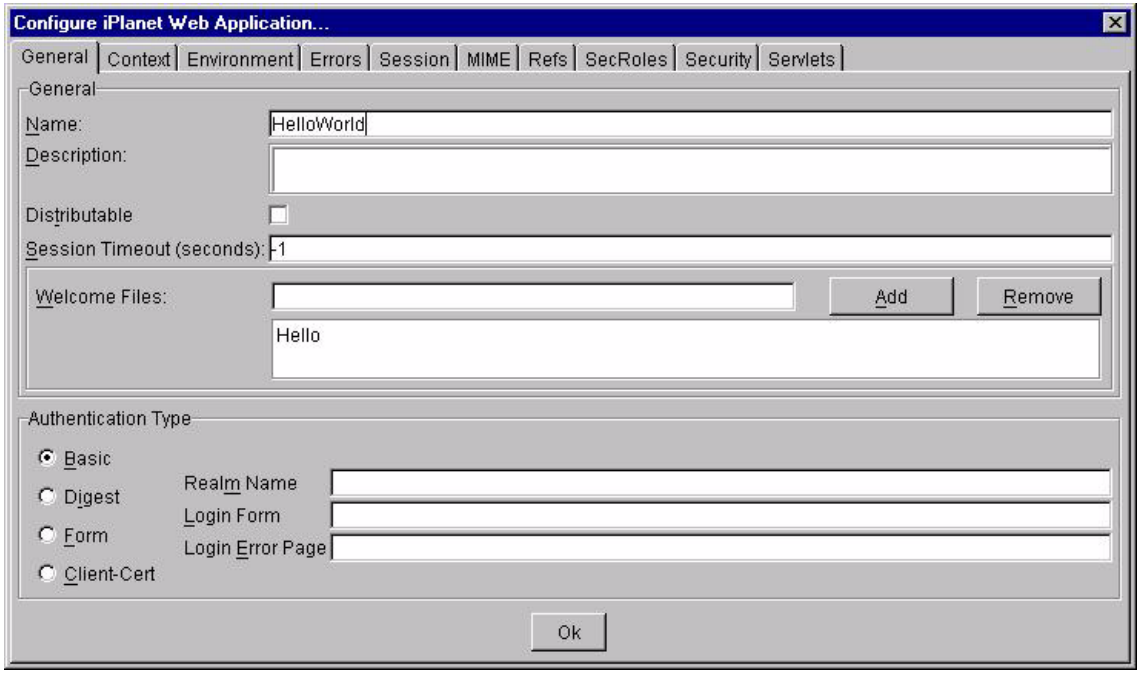
Configure iPlanet Web Application

Choose this menu item from the project menu to configure your iPlanet Web Application for deployment to the iPlanet Application Server. It allows you to configure deployment information on the following tabbed panels:

- General Tab; specifies overall attributes for your Web Application.
- Context Tab; declares the Web Application's servlet context initialization parameters.
- Environment Tab; identifies environmental elements specific to your Web Application.
- Errors Tab; maps error codes to a resource within the Web Application.
- Session Tab; configures the Web Application targeted to your operational iAS environment.
- MIME Tab; identifies the mapping between MIME types and MIME Extensions.
- Refs Tab; identifies EJB and resource factory references in the source code of any Web Application component.

- SecRoles Tab; declares security roles that are actually used within the organization using this Web Application.
- Security Tab; specifies security constraints to protect web content by defining a resource collection.
- Servlets Tab; specifies the mapping between servlets and the URL pattern that invokes them.

The graphical user interface to the deployment descriptor editor is shown in the following illustration.



General Tab

The General deployment descriptor tab allows you to set the following fields or columns:

Field/Column	Description
Name	Enter a short name that is intended to be displayed by GUI tools. For example, bank.war could be named bank.

Description	Enter descriptive text about the web application.
Distributable	If checked, specifies that the web application has been programmed appropriately so that the servlets can be deployed across multiple Java Virtual Machines running on the same or different hosts.
Session Timeout (seconds)	Enter the number of seconds to specify a default session timeout interval for all sessions created in this web application. The default is 500 seconds.
Welcome Files	Enter the path to welcome file(s) in the Web Application module. If your application starts with a page other than a logon, you can specify the start page here. The server tries to load the first file from this list, if the file is loaded successful, the server ignores the others. Otherwise, the server tries to load the next file until the end of the file list is reached.
Authentication Type	Specify the Authentication Type while logging on to the Web Application: None, Basic, Digest, Form, Client Cert.
Authentication Type: None	If selected, no authentication is required while logging on to the Web Application. This is the default.
Authentication Type: Basic	If selected, the HTTP Basic Authentication protocol is used for authentication. The authentication is based on a username and password. The web client obtains the username and password and transmits them to the web server using base64 encoding. The web server then authenticates the user.
Authentication Type: Digest	If selected, the HTTP Digest Authentication is used for authentication. The authentication is based on passing the username and password in an encrypted form. (Currently, digest is not used by iAS 6.0.)
Authentication Type: Form	If selected, the developer controls the look and feel of the login screens, unlike the HTTP browser's built-in authentication mechanisms. **Realm Name**: Name to be used for authentication with this Web Application. (Currently, realm names are not used by iPlanet Application Server 6.0.) **Login Form**: Enter the location of the login page (login.jsp). **Login Error Page**: Specify the location of the error page (error.jsp) that is used with this type of login. The error page is displayed when the authentication is not successful.

Authentication Type: Client-Cert	If selected, authentication is performed using HTTPS (HTTP over SSL). This secure type of authentication requires the user to possess a Public Key Certificate to logon. Public Key Certificates are useful in e-commerce applications and single-sign on applications. For more information about single sign-on applications, see the iAS tab for web applications.

Context Tab

You use the Context tab to declare the web application's servlet context initialization parameters. These are accessed by:

```
ServletContext context = getServletContext()
//returns context
```

```
context.get;InitParameterNames()
//returns enumeration of init parameters
```

```
config.getInitParameter(<paramName>)
returns value of a specific parameter
```

Field/Column	Description
Name	Enter the parameter's name.
Value	Enter the parameter's value.
Description	Enter a description of the parameter.
Add button	Click to add a row to the Context table for you to specify the web application's servlet context initialization parameters.
Remove button	Click to remove the selected row from the Context table.

Environment Tab

You use the Environment tab to identify environmental elements specific to your web application such as sales tax rates or zip codes.

Field/Column	Description
Key Name	Enter the name of the property. For example, if you are going to enter a zip code, enter the city name in this column.
Type	Click in the column and choose a datatype for the property, such as integer.

Value	Enter the value of the property in this column. Following the zip code example, enter the zip code corresponding to the city name, for example, 49770
Add button	Click to add a column in the table for you to define environment properties.
Remove button	Click to remove the selected row from the table.

Errors Tab

You use the Errors tab to map error codes to a resource within the web application. This defines what resource will be used (or HTML page to display) if an error occurs.

Field/Column	Description
Error	Enter an error code or exception type in this column. An example of an error code is HTTP error 404 (which indicates that the web server cannot find the requested file or script). When entering an exception type, enter a fully qualified class name of a Java exception type.
Path	Enter the location within the web application of the resource to use when this error or exception occurs, for example,. an HTML page to display.
Add button	Adds a row to the Errors table for you to map an error to a resource.
Remove button	Removes the selected row from the table

Session Tab

You use the Session tab to configure the web application targeted to your operational iPlanet Application Server environment.

For example, iPlanet Application Server supports single sign-on for multiple web applications. You can create multiple applications where the user needs to log on (be authenticated) only once. A scenario for this feature could be a consolidated airline booking service which searches all airlines and provides links to different airline web sites. Once the user is authenticated on the consolidated booking service, the log on information can be used by each of the individual airline sites, without requiring the user to log on again.

Field/Column	Description

Distributed Session	If checked, a session can be distributed across multiple iPlanet Application Server machines. A session can be either a distributed fault-tolerant session, or a lightweight local-only session. Local only sessions are available locally in one engine, and have to depend upon requests being sticky (See iPlanet Application Server deployment descriptors in the Online Help for more information.) In addition, local only sessions have no fault-tolerance if the engine crashes.
Secure Session	If checked, the session can only occur on a secure (HTTPS) server.
TimeOut(seconds)	Enter a period in seconds that specifies when the client is no longer active. If unspecified, a system-wide default session timeout of 30 minutes is used.
Timeout Type	Specify when the timeout count begins: since last access or since session creation.
Data Synchronization	If you specified a distributed session on this tab, then select the session type: dsync-local or dysnc-distributed. DSync distributed helps failover across instances of the application server participating in a cluster.
Domain (supports single sign-on)	Enter the domain to which the cookie is sent back from the web browser. By default, the domain of the URL that sets the cookie is assumed to be the domain where the cookie is sent back. You can specify a different domain here, if needed.

Path (supports single sign-on)	Enter the path for the session cookie (i.e. the minimum path that the URL must have for the cookie to be sent back from the browser. For example, setting the path to `/phoenix`, sends the cookie back when either of the following URLs are accessed:
	
	`http://my.foo.com/phoenix/bees.htm`
	`http://my.foo.com/phoenix/birds.htm`
	
	The path must begin with slash (/). If the path is not set, then the default path is assumed to be that of the URL setting the cookie.

Scope (supports single sign-on)	A grouping name that identifies applications participating in single sign-on. Each application participating in a single sign-on for a grouping of applications must use the same grouping name. For example applications such as login, homeBanking, `loanCenter` within the same application server cluster, can share a session by specifying the same scope variable, `myBank`.

MIME Tab

You use the Mime tab to identify the mapping between MIME types and MIME Extensions for your web application.

Field/Column	Description
MIME Type	Enter a MIME type for a file such as text/html or image/gif for files used within your web application.
MIME Extension	Enter the file extension in the column that corresponds to the MIME type.
Add button	Adds a row in the MIME Types table.
Remove button	Removes the selected row from the MIME Types table.

Refs Tab

You use the References tab to identify any of the following references in the source code of any of the web application components:

- References to resource factories.

  A resource factory is a provider of resources, such as a JDBC or a JMS driver. Resource factories are used within an EJB to acquire resources such as database connections.

- References to other EJBs Resource Class

.

Field/Column	Resource Factory References Description
Resource Name	Enter the name of the resource in this column that a web application component references, for example, a database.

Description	Enter a description of the resource, for example, a JDBC connection to a database.
Resource Class	Click in the Resource Class column and choose a data source from the list box. A resource class is a Java interface that specifies the type of data source, for example, javax.sql.DataSource.
Authorization	Choose the authorization type as follows: Application or Container. Application specifies that the enterprise bean code signs on programmatically to the resource manager. Container specifies that it will sign on to the resource manager on behalf of the enterprise bean.
JNDI Name	Enter the Java Naming and Directory Interface (JNDI) name by which the resource factory is known. The JNDI name becomes the name of the module and the reference name of the resource factory.

Field/Column	References to EJB Description
Reference	Enter the name of the EJB that is referenced in the source code of a component of the web application and should match the string used for EJB lookup.
Bean Type	Click in the Bean Type column and choose the Bean Type:Session or Entity.
Linked to Bean	Click in the Resource Class column and choose a data source from the list box. A resource class is a Java interface that specifies the type of data source, for example, javax.sql.DataSource.

Authorization	Choose the authorization type as follows: Application or Container. Application specifies that the enterprise bean code signs on programmatically to the resource manager. Container specifies that it will sign on to the resource manager on behalf of the enterprise bean.
JNDI Name	Enter the Java Naming and Directory Interface (JNDI) name by which the resource factory is known. The JNDI name becomes the name of the module and the reference name of the resource factory.

SecRoles Tab

You use the Security Roles tab to identify any security roles that the servlet references in its code.

Field/Column	Description
Referenced Name	Enter the name by which the servlet's code refers to a security role. The developer will often refer to roles in the servlet's code without knowing how these names fit into the security policies of the organization using the web application. For example, the developer could use a security role of: Credit-Worthy Customer.
Link to Role	Name of the corresponding security role used within the company. This maps the security role that the developer uses to a role that is actually used within the organization using this web application. For example, Credit-Worthy Customer could be linked to VIP Customer.
Reference Description	Enter a comment intended to assist the user who assembles the modules into an application.ear file when choosing a role.

Security Tab

When specifying a security constraint, first you define the resource collection and then you map the resource collections to roles. Then you specify the transport layer constraint of the user data. You define the resource collection in the first part of this tab.

Field/Column	Description
Collection	Enter a name for the resource collection.

URL Pattern	Enter the URL pattern(s) for this resource collection as a comma delimited list.
Method	Enter the HTTP method(s) as a comma delimited list for this resource collection. HTTP methods can include GET, POST, GET and POST, depending on what the user's web server supports. If no HTTP methods are specified, then the security constraint applies to all HTTP methods.

The following information is used to map resource collections to roles and specify the transport layer.

Field/Column	Description
Security Constraint	Enter a name for the security constraint or accept the default.
Collection	Enter a one or more resource collection names that you want apply access control. The resource collection names that you enter must be predefined in the top half of this tab.
Roles	Enter the roles that are allowed to access the resources described by the resource collection. Roles names are bound to actual users and groups when you deploy your application. If the user is not a member of an allowed role, the user is denied access to that resource. Note that all roles entered in this column must be defined in the Link to Role column on the Security Role Refs tab for servlet descriptors.
Transport Guarantee	Click in the column and choose NONE, INTEGRAL or CONFIDENTIAL to define the type of communication between the client and server as follows: NONE: The application does not required any transport guarantees. INTEGRAL: The application requires that data sent between the client and server is not changed in transit.(SSL is required). CONFIDENTIAL: The application requires that data sent between the client and server is not read during transit. (SSL is required).
Add button	Click to add a new row for you to specify access control for a resource collection.
Remove button	Click to remove the selected row from the table.

Edit Collection button	Select a row in the table and click the Edit Collection button to open a dialog where you select a new collection name. The dialog displays collection names that have been created in the first half of this dialog.
Edit Roles button	Select a row in the table and click the Edit Roles button to open a dialog where you can select one or more roles that are allowed to access the resource collection. The dialog displays role names that have been specified on the Roles tab for web applications.

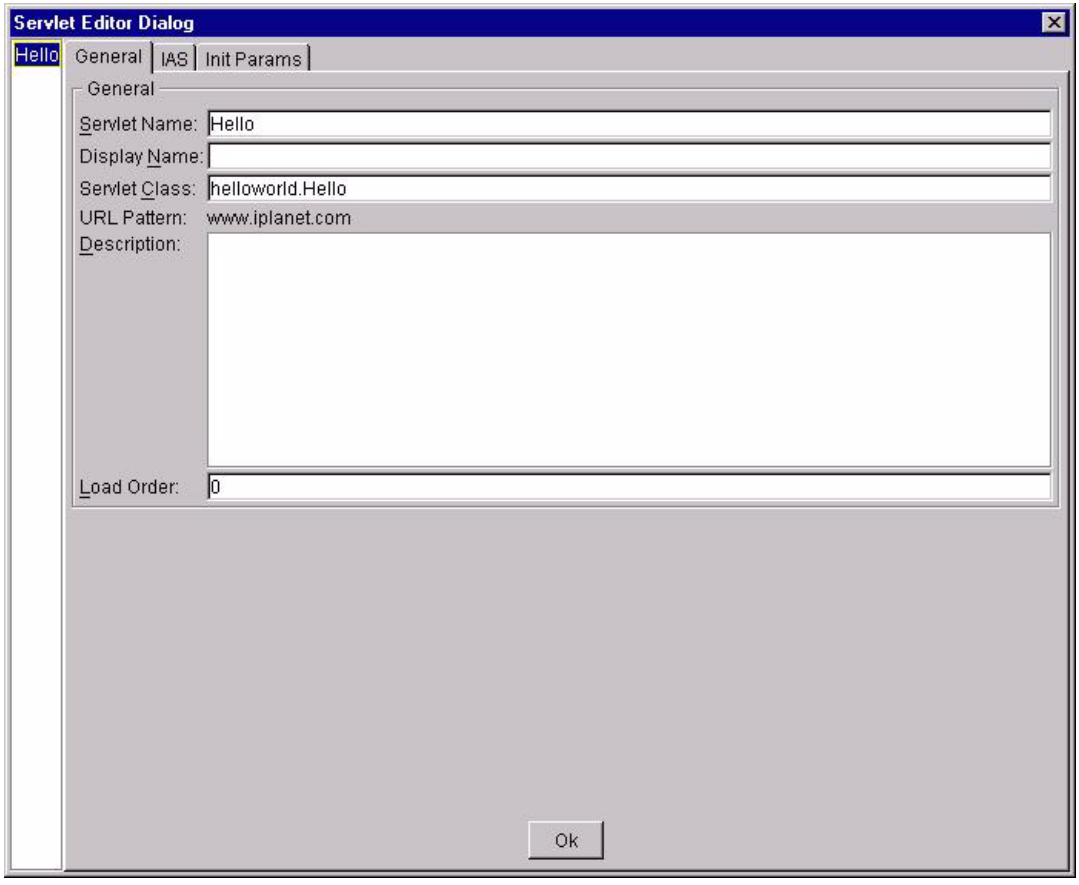
Servlets Tab

You use the Servlets tab to specify the mapping between servlets and the URL pattern that invokes them. You also use this tab to identify any JSP tag libraries that are used within the web application.

Field/Column	Description
Servlet Name	After adding a row to the Servlet Mapping table, click in the column and select a servlet that is included in this web application.
URL Pattern	Enter a comma delimited list of the URLs that will invoke the servlet.
JSP Tag Lib URI	Enter a URI, relative to the location of the web.xml file identifying a Tag Library used in the web application.
JSP TLD Location	Enter the location of the Tag Library Description file for the Tag Library relative to the root of the web application.

Configure Servlets

- Choose this menu item from the Project menu to configure your iPlanet Servlet project for deployment to the iPlanet Application Server. This command opens the Configure Servlets deployment descriptor editor.



The three tabbed panels allow you to specify the following deployment attributes, including:

- General Tab; Specifies overall properties for a servlet in the Web Application.
- iAS Tab; Define iAS-specific options for the servlet according to your targeted operational iAS environment.
- Init Params - Specifies initialization parameters for a servlet in the web application

General Tab

You use the General tab to specify overall properties for a servlet in the web application. You need only to specify deployment descriptors for web application components that the user can access directly.

Field/Column	Description
Servlet Name	Enter the name of the servlet.
Display Name	Enter a short name that will be displayed by the GUI tools accessing this servlet or JSP.
Servlet Class	For servlets, displays the fully qualified class name of the servlet.
URL Pattern	Displays a comma delimited list of the URLs that invoke the servlet. This information is read from the Servlet (WAR) tab shown in the URL Pattern field.
Description	Enter a description of the servlet.
Load Order	Enter a number to specify that this servlet should be loaded on the start up of the web application. Enter a positive number indicating the order in which the servlet should be loaded. Note that lower integers are loaded before higher integers. Enter 0 (or a negative number) to indicate that the container is free to load the servlet at any time in the start-up sequence.

iAS Tab

You use the iAS tab to define iPlanet Application Server-specific options for the servlet so you can configure the servlet to your targeted operational iPlanet Application server environment.

Field/Column	Description
GUID	This is the Globally Unique ID that is provided by the Deployment Tool assigned when you registered this servlet.

Number of Singles	Enter the number of objects in the servlet pool, when SingleThread mode is used. In the case of a servlet that implements the SingleThreadModel interface, the servlet container will instantiate multiple instances of that servlet (number of singles) so that it can handle a heavy request load while still serializing requests to a single instance. In all other cases where the servlet does not implement the SingleThreadModel interface, the container instantiates a single servlet instance per servlet definition per KJS and this option does not apply.
Error Handler	Enter the path to a servlet or JSP file to handle messages used when the Error Handler on the iAS Param tab is set to log.
Validation Required	If checked, input parameters of a servlet defined in the ias-web.xml file (or on Init Params tab of the DD editor) are validated.
Disable Reload	If checked, disables dynamic reloading of this servlet instance. Otherwise, recompilation or redeployment of this servlet will cause dynamic reloading of the corresponding servlet instance for new requests, without restarting KJS or the iAS instance.
Encryption	If checked, specifies communications to the servlet are encrypted for security considerations.
Cache timeout	Set a time-out value in seconds for the caching of the servlet. If you set this value to zero, caching is disabled.
Cache size	Set a cache size for the servlet. If you set this value to zero, caching is disabled.
Caching criterion	Express your caching criteria as a series of comma-delimited string descriptors. Each descriptor defines a match with one of the input parameters to the servlet.

The table below allows you to customize sticky load balancing for this servlet on each iPlanet Application Server machine targeted to deployment. If this table is blank, then the Sticky checkbox applies to all deployment machines.

Field/Column	Description
IP Address	Enter the IP address of the iPlanet Application Server instance you want to customize.
Port	Set to true/false to apply/disable sticky load balancing for this servlet on the iAS machine.
Sticky LoadB	Set to true/false to apply/disable sticky load balancing for this servlet on the iAS machine.
Sticky LoadB	Check to specify that sticky load balancing is activated for this server.
Enable	Click to remove the selected row from the Server Information table.
Add Serve	Click to add a row to the Server Information table.
Remove Server	Click to remove the selected row from the Server Information table.

InitParams Tab

You use the Init Params tab to specify initialization parameters that are set in the `init()` call when the servlet is loaded. The initialization parameters are accessed in the following manner:

`ServletConfig config= getServletConfig()` - returns configuration

`config.get;InitParameterNames()` - returns enumeration of init parameters

`config.getInitParameter(<parmName>)` - returns value of specific parameter

Field/Column	Description
Name	Enter the name of the parameter.
Value	Enter the value of the parameter.
Description	Enter a description of the parameter.
New Parameter button	Click to add a new row to the table where you can specify an initialization parameter.

Delete Parameter button	Click to remove the selected row from the table.