User’s Guide

VERSION 4.0

Inprise Application Server™

Inprise Corporation, 100 Enterprise Way
Scotts Valley, CA 95066-3249

Chapter

2

Starting the Inprise Application Server

The Inprise Application Server is a set of services and tools that enable your enterprise to build, deploy, and manage web applications. These applications provide dynamic content by using java, servlets, and Enterprise Java Beans (EJB) technologies.

This chapter gives an overview of starting and stopping the server, changing server configurations, and managing top-level services.

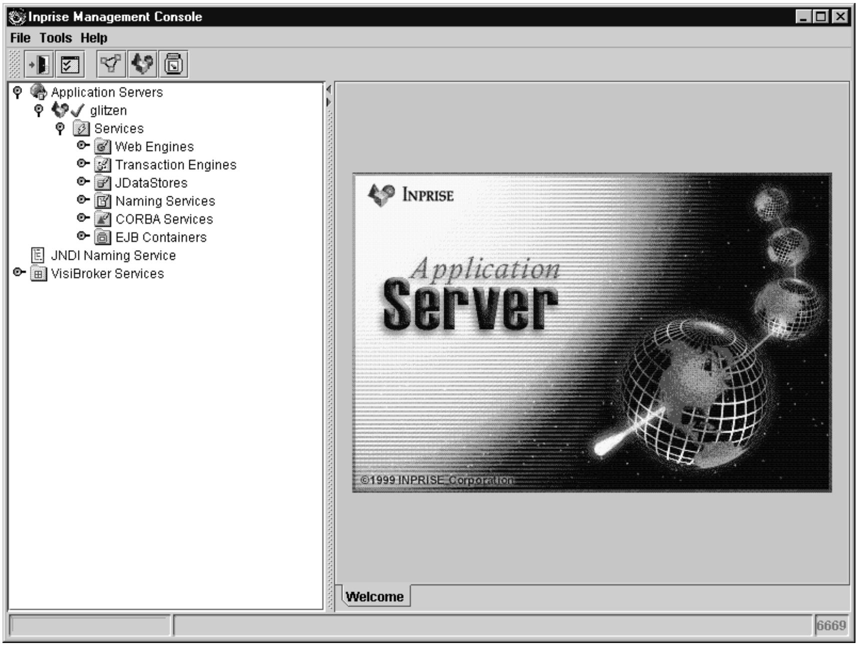
About the top-level services

Services included with the Inprise Application Server are:

- **EJB Containers:** Use this service to deploy EJBs and monitor EJB performance. Tools include a Deployment wizard and a Deployment Descriptor Editor. You can create and manage as many EJB Containers as desired.
- **CORBA services:** A collection of Inprise CORBA services including the VisiBroker Object Request Broker (ORB) and Smart Agent (osagent).
- **Naming Services:** The Naming Service associates meaningful names with objects, and then uses those names in a lookup facility. The Inprise Application Server supports two naming services and includes a JNDI interface for EJB applications.
- **JDataStores:** A database service written entirely in Java for embedded, web, and mobile database applications. You can create and manage as many JDataStores as desired.
- **Transaction engines:** Transaction services provide tools to develop and manage transactional applications, including the ability to access a database within a transaction. The Inprise Application Server supports two transaction engines, JTS and ITS.

- **Web Service Engines (or Web Engines):** Designed to support development and deployment of web applications, this web service and underlying HTTP engine includes a graphical interface for server configuration and administration. The server includes support for both standard (http) and secure (https) Web Service Engines.

Figure 2.1 Using the Inprise Application Server Console to view and manage services

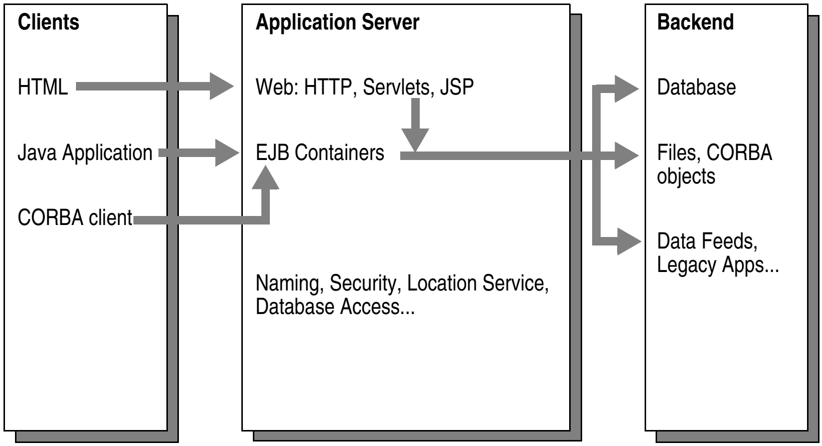


Architectures supported

Designed specifically for EJB development, deployment, and management, the Inprise Application Server supports a variety of server architectures for Java and web applications. Common ways enterprise beans can be accessed include:

- Web browser client can invoke a JSP or servlet that invokes the enterprise bean that communicates to a backend process (such as a database).
- Applet, stand-alone Java application, or CORBA client can invoke the enterprise bean directly.

Figure 2.2 Common architectures



Separation of development and deployment

The Inprise Application Server supports the design model described in version 1.1 of the EJB Specification published by Sun Microsystems. In this model, development tasks are separated from deployment to maximize engineering resources.

The following is an overview of the steps needed to develop and deploy EJBs using the Inprise Application Server. Deployment steps are described in detail in the remaining chapters.

Developing the beans

The application developer creates enterprise beans and servlets using JBuilder (or a text editor). All three types of beans are supported: Stateless Session, Stateful Session, and Entity.

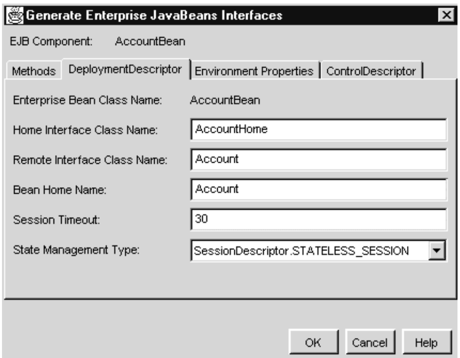
- For every enterprise bean, development includes the following three pieces:
  - Enterprise bean class
  - Home interface
  - Remote interface

  The home interface is an EJB component interface that enables clients to look up and create EJB objects. The remote interface is an EJB component interface that enables clients to interact with an EJB object on a server.
- The application developer compiles the remote interface, home interface, and implementation class for each enterprise bean.

- Once the appropriate beans are created and compiled, the application developer
  - Creates a deployment descriptor for each enterprise bean
  - Defines a primary key for each Entity bean
  - Bundles the deployment descriptor and enterprise bean classes (bean class, remote interface, home interface, and primary key) into a JAR file

Inprise provides a variety of tools to aid the development of EJBs. For more information, see the Borland JBuilder *User's Guide* and the Inprise Application Server *Enterprise JavaBeans Programmer's Guide*.

Figure 2.3 Using JBuilder to create enterprise bean interfaces



Deploying the beans

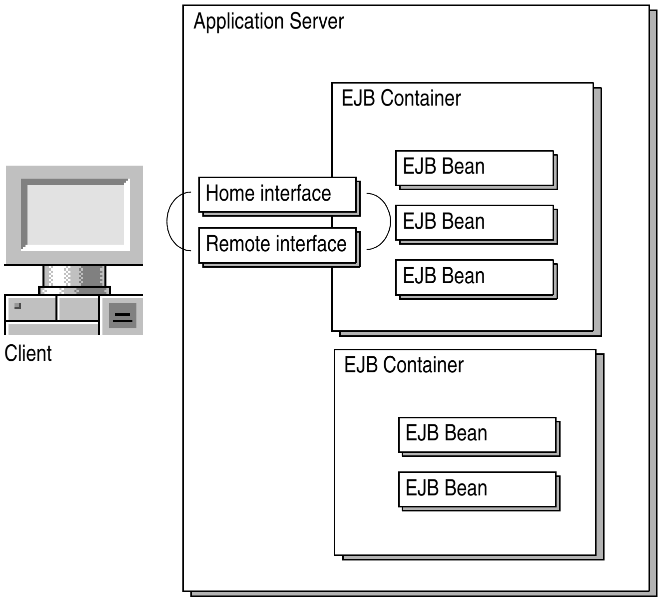
Attributes for transactions, security, and mapping to EJB-Container data sources all can be handled at deployment. Developers don't have to worry about these details when developing EJBs.

- Deploying an enterprise bean includes the following tasks:
  - Specifying the environment properties that the EJBs require at runtime
  - Deploying the JAR file to an EJB Container
  - Monitoring the deployment
- The Inprise Application Server provides a default EJB Container for deployment. You can create additional Containers as needed. (See "Creating a new Container" on page 3-20.)
- Deployment tools include a Deployment wizard for deploying beans to the Container and Deployment Descriptor Editor for setting or changing information in the deployment descriptor.
- The Deployment wizard creates the Inprise-specific stubs and skeletons necessary for deploying the enterprise bean to the EJB Container.

Writing and running clients

- Once the beans are in place on the server, developers write clients to access the beans. The Inprise Application Server supports EJB clients written using RMI or CORBA.
- Users run the client as needed.

Figure 2.4 Inprise Application Server and EJB Containers



Starting the server

The Inprise Application Server can be installed and started either locally or on a shared machine. The server, in turn, starts all services configured for the server. These include the Web Service Engine, database, Smart Agent, EJB Containers, and so on.

Starting a default server

To start a default version of the server:

WinNT
- Click the Start button and choose Application Server from the Inprise Application Server program group, or

WinNT/ UNIX
- Open a command window and type the following:

```
ias
```

Note To recognize the `ias` command, you must have your path updated to include the ias bin directory (*install_dir*/bin), or you can enter the path explicitly.

Once the server has started, you can specify which services are started with the server. See "Managing top-level services" on page 2-10 for more information.

Important The `ias` command includes command options you can use to start the server in something other than the default configuration. For more information, see "Server configurations" on page 2-14.

Starting the Console

The Inprise Application Server includes a GUI-based Console which functions as the main control point for the Application Server. The Console enables you to view servers on the network, start and stop services, and so on. The Console also enables you to view EJB JAR files and Containers, set deployment properties, and monitor performance.

The Inprise Application Server typically runs on a large shared UNIX or Windows NT machine, while the Console runs on any machine from which users want to view or modify the distributed system. Once the Console is installed, you can deploy to any Application Server on your network.

To start the Console:

1. Make sure an Inprise Application Server is started.

   You must start an Application Server before you start the Console.

2. Use one of the following methods to start the Console:

WinNT
- Click the Start button and choose Console from the Inprise Application Server program group.

UNIX
- Open a command window and enter the following command:

```
console.sh
```

Note To recognize the console command, you must have your path updated to include the Console bin directory (*install_dir*/console/bin), or you can enter the path explicitly.

The Console window appears. The Console enables you to view all instances of the Inprise Application Server that share the same Smart Agent port.

For more information on using the Console to view servers and services, See "Managing top-level services" on page 2-10.

Setting Console preferences

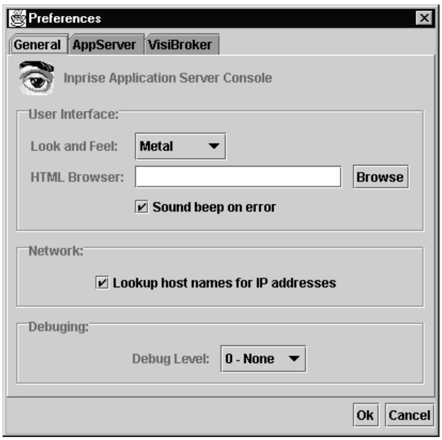
Console preferences enable you to specify the Smart Agent port used by the Console, the default polling interval for performance information displayed in the Console, and so on.

To set Console preferences:

1 Start the Console and choose Preferences from the File menu.

A dialog box appears with a list of preferences.

Figure 2.5 General preferences



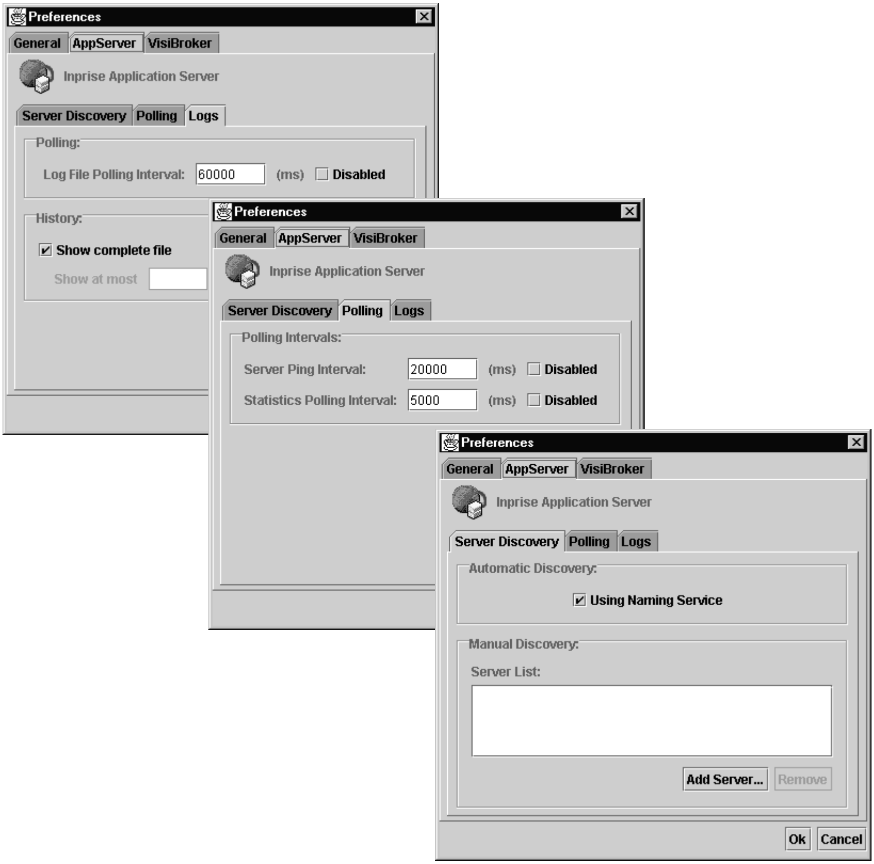
2 Set preferences as desired and click OK.

General preferences include:

- **Look and Feel:** Sets the display format and behavior of the Console windows (for example, Windows or CDE/Motif).
- **HTML browser:** Specifies the browser you wish to use with the Application Server.
- **Sound beep on error:** Enables an audible beep if an error occurs.
- **Debug Level:** Sets the level of debug messages displayed in the Console. There are 4 debug levels (0 to 3) with 3 being the most messages displayed.
- **Lookup host names for IP addresses:** Uses host names as well as IP addresses. If you are not running a DNS to resolve host names on your network, you may wish to uncheck this setting to enhance system performance.

There are three panels of Inprise Application Server preferences.

Figure 2.6 Inprise Application Server preferences



Log file preferences include:

- **Log File Polling Interval:** Sets the time interval (in milliseconds) for how often log files are updated. The Disabled checkbox lets you disable polling.
- **Show complete file:** Enables you to view all log entries since the file was created.
- **Show lines:** Sets the maximum number of message lines (log entries) shown. For example, if you specify 500 lines, the last 500 lines in the log file are shown.

Polling preferences include:

- **Server Ping Interval:** Sets the time interval (in milliseconds) for how often Inprise Application Servers are checked to verify they are active. The Disabled checkbox lets you disable server pinging.
- **Statistics Polling Interval:** Sets the time interval (in milliseconds) for how often objects are sampled for performance information. The Disabled checkbox lets you disable polling.

Server Discovery preferences include:

- **Automatic Discovery:** Enables the Location Service to automatically locate Inprise Application Servers on the network and display them in the Console.
- **Manual Discovery:** Enables you to manually add the servers you wish to display in the Console. Use this setting to selectively specify which servers appear in the Console.

Figure 2.7 VisiBroker preferences

VisiBroker preferences include:

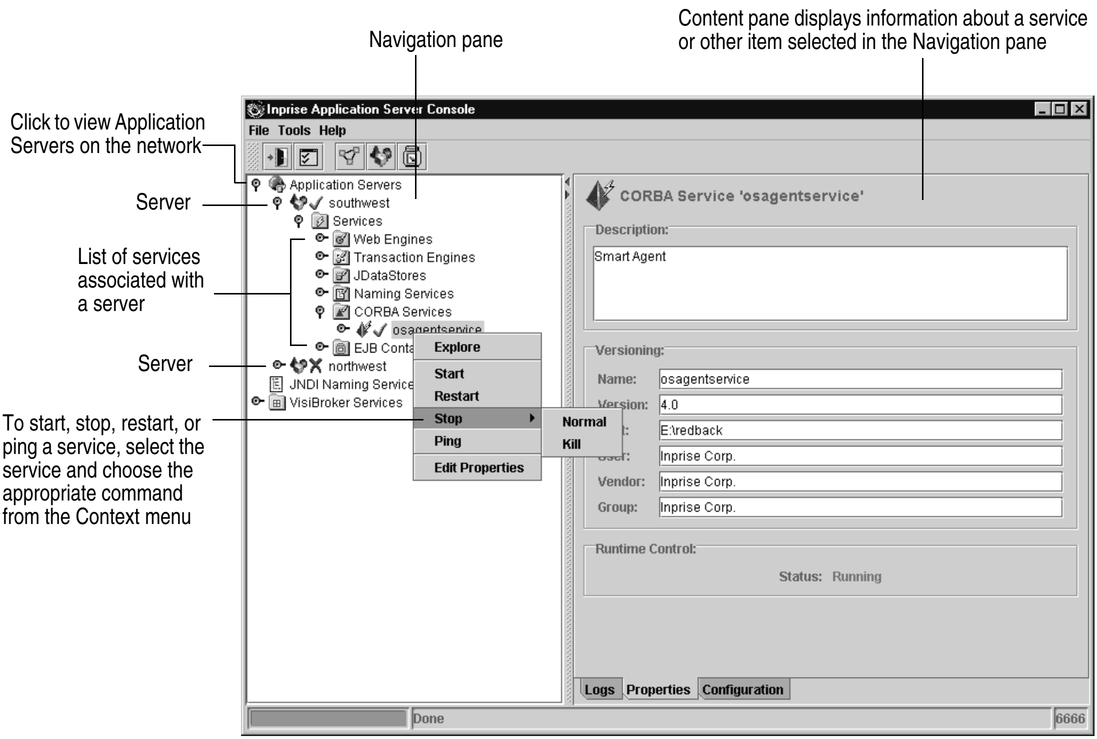
- **Smart Agent Port:** Sets the port the Console uses to communicate with the Smart Agent.

You may have to restart the Console for changes in some preferences to take effect (for example, the Smart Agent port).

Managing top-level services

Use the Inprise Application Server Console to view servers on the network and manage services associated with a server.

Figure 2.8 Managing top-level services



Menu commands you can use with servers and services include

Table 2.1 Managing top-level services

Command	Description
Start	Starts a stopped server or service.
Restart	Shuts down and restarts the server or service.
Stop | Normal	Exits the server or service normally. Writes data to disk as needed.
Stop | Kill	Terminates the sever or service immediately.
Ping	Checks to see if the server or service is active.
Edit Properties	Opens a dialog box with a list of server or service properties you can edit. (See "Editing service properties" on page 2-12 for more information.)

Note In addition to viewing and managing Inprise Application Servers, the Console enables you to manage VisiBroker services such as Gatekeeper and the Server Manager. (For more information, see the VisiBroker for Java documentation.)

Viewing log files

Log files provide a collection of error and status messages associated with a service. If problems occur, these files are useful for troubleshooting.

To view the log file for a service:

1. Use the Navigation pane to display the service you wish to examine. Then, open the folder for the service.

   A list of folders and objects associated with the service appears.

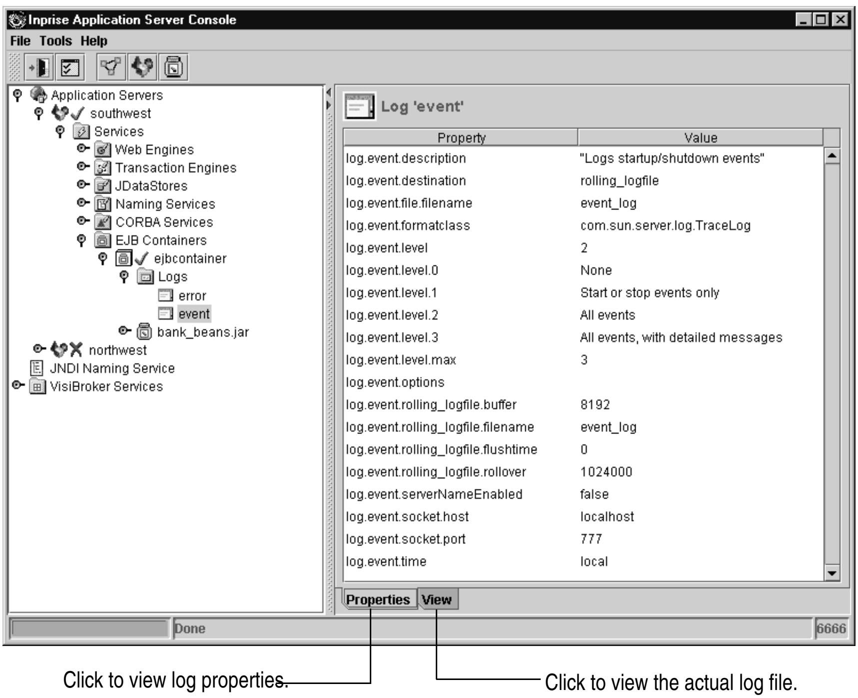
2. Open the Logs folder for the service and select the log you wish to view.

   - Error log displays all error messages produced by the service.
   - Event log displays all events recorded by the service.

   A list of log file properties appears in the Content pane.

3. Click the View tab to view the log file.

Figure 2.9 Viewing log files



Note You can use Console preferences to control how many message lines are stored in the log files and how often the logs are updated.

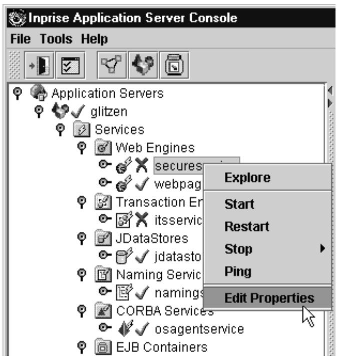
Editing service properties

Each service included with the Inprise Application Server has a set of properties you can edit to customize the service. The properties you can edit vary depending on the service.

To view and edit service properties:

1. Open the Application Server Console and select the service whose properties you wish to edit.
2. Choose Edit Properties from the Context menu.

Figure 2.10 Editing Service properties

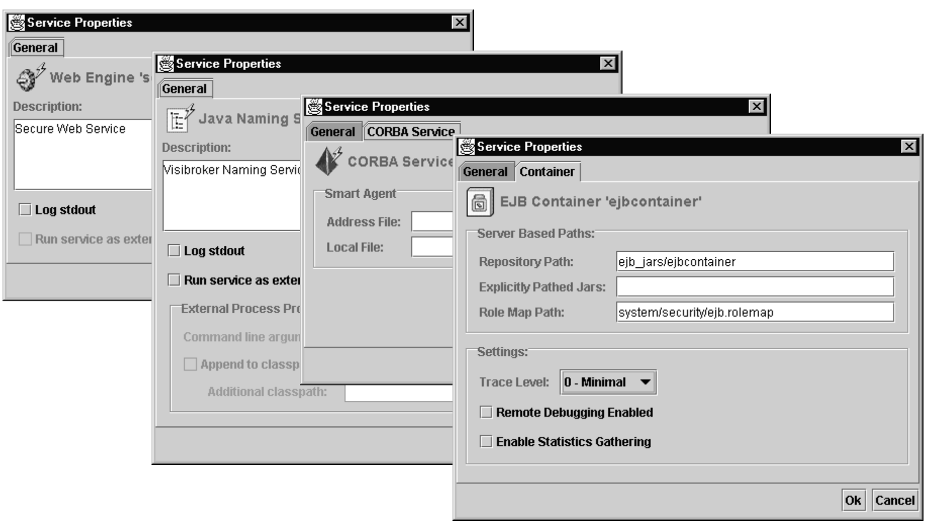


A dialog box appears with a list of Service properties.

3. Make changes as desired and click OK.

The following sections describes the properties you can edit for each service.

Figure 2.11 Examples of Service properties



Chapter

3

Using EJB Containers

This chapter describes how to use the Inprise Application Server Console to view and manage EJB Containers and related services.

About EJB Containers

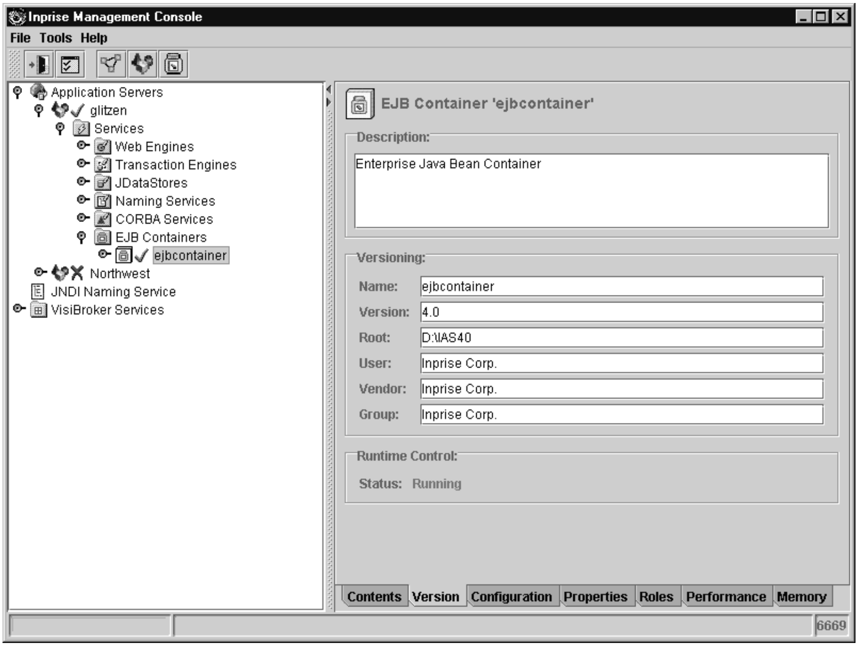
The Inprise Application Server Console enables you to view and edit the contents of Containers and enterprise beans.

The Console includes an EJB Deployment wizard. During deployment, the contents of the EJB JAR file are verified and Inprise-specific information is added. The Deployment wizard:

- Creates the required stubs and skeletons for the enterprise bean
- Deploys the JAR to a Container
- Validates the deployment descriptor

After the enterprise bean is deployed, you can use the Deployment Descriptor Editor to change deployment information as desired. For example, you can use the editor to specify or change the Transaction Policies and Security Roles for the enterprise bean.

Figure 3.1 EJB Container in the Inprise Application Server Console



Persistence support

Modeled after version 1.1 of the EJB specification published by Sun Microsystems, the Inprise EJB Container supports built-in persistence. This persistence can be bean-managed or container-managed.

For bean-managed persistence, the developer overrides the ejbLoad and ejbStore methods and provides custom code (for example, using JDBC) to transfer the data from the object's fields to permanent storage and vice versa. Bean-managed persistence ties the enterprise bean to one method of persistence and requires the developer to write and maintain persistence code.

For container-managed persistence, Inprise supplies mapping tools to map the Entity bean's field to fields in the JDataStore. The Entity bean can be persisted into a different data source by remapping the fields in the deployment descriptor.

For more information, see the Inprise Application Server *Enterprise JavaBeans Programmer's Guide*.

Viewing EJB Containers

To view EJB Containers with the Inprise Application Server Console:

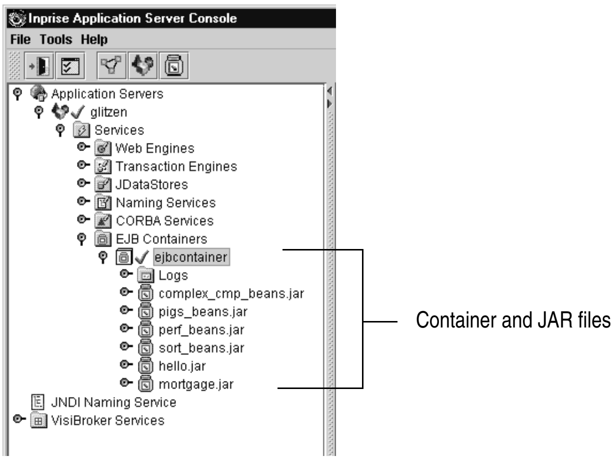
1 Start the Application Server and Console.

For details, see Chapter 2, "Starting the Inprise Application Server."

2 Open the Services folder in the Navigation pane and select EJB Containers.

A list of EJB Containers and JAR files appears in the Navigation pane.

Figure 3.2 EJB Containers



The EJB server includes a default Container that appears in the Console window.

An EJB server can have multiple Containers. You define and configure Containers as needed to deploy objects on the network. You can create custom EJB Containers for various data sources (such as Oracle) and support them separately from enterprise bean development. (For more information, see "Creating a new Container" on page 3-20.)

Viewing container attributes

Containers include a variety of attributes you can view and edit. These attributes include container contents, configuration information, and so on.

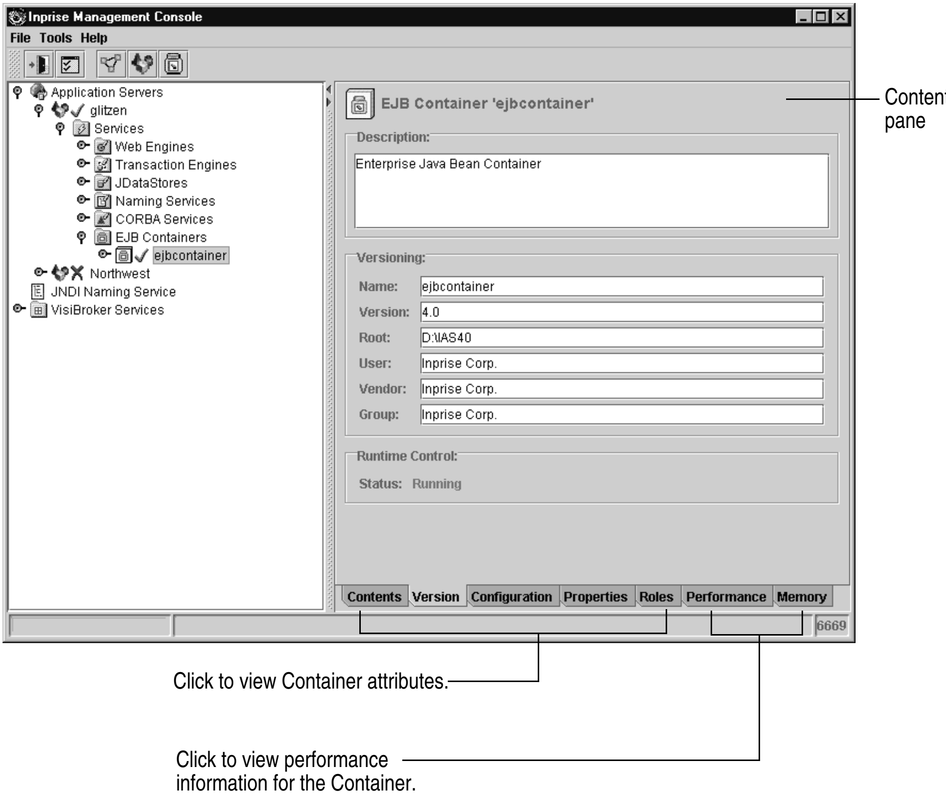
Note This section includes information about container attributes. For information about enterprise bean attributes, see "Viewing bean attributes" on page 3-10.

To view container attributes:

- Open the Navigation pane in the Inprise Application Server Console and select the EJB Container you wish to view.

  Attributes for the Container appear in the Content pane.

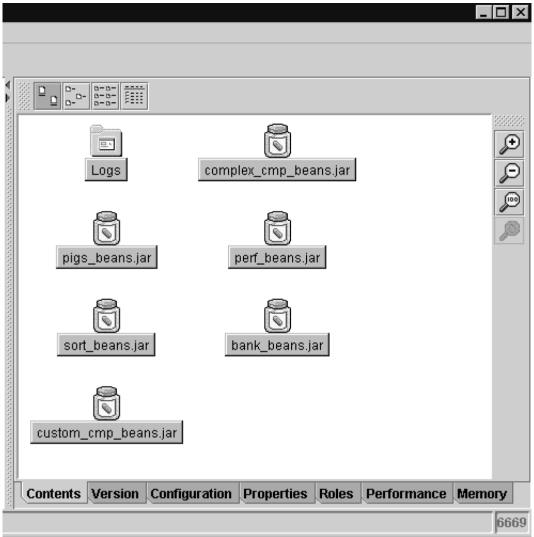
Figure 3.3 Container attributes



Contents panel

The Contents panel shows the JAR files included in the Container.

Figure 3.4 Contents panel

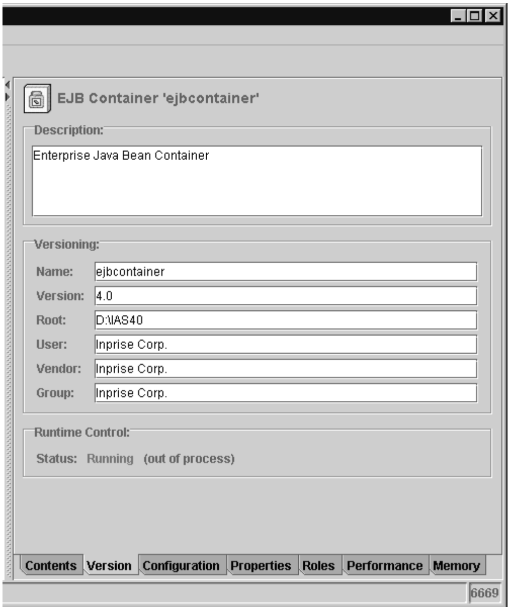


Version panel

The Version panel includes:

- General description of the Container
- Version number and root installation directory of the Inprise Application Server running the Container
- Container vendor
- Container runtime information

Figure 3.5 Version information

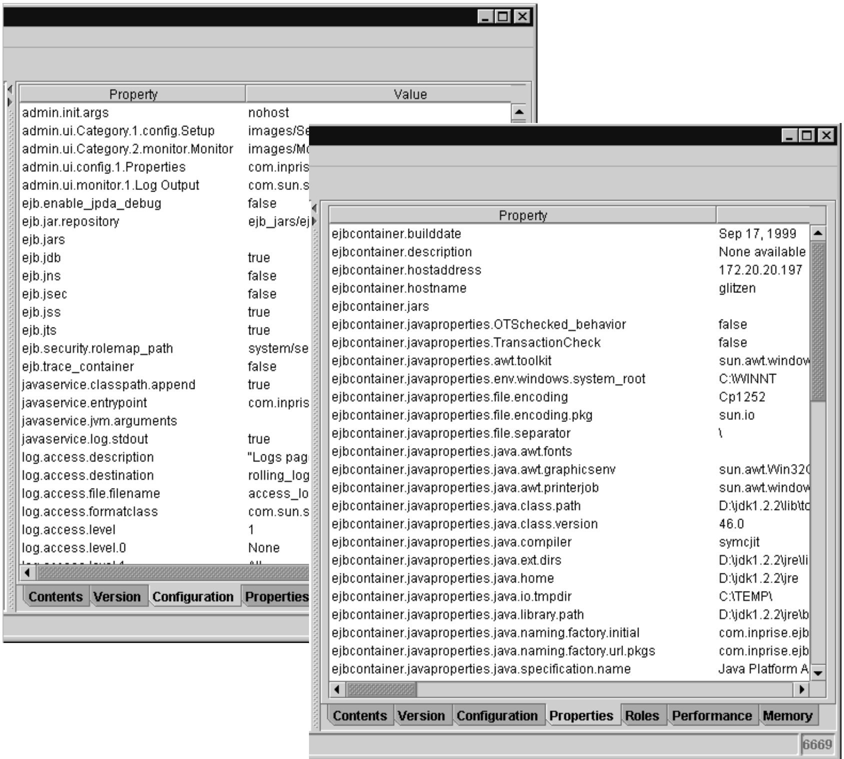


Configuration and Properties panels

The Configuration panel lists server configuration information defined for the Container. This information includes log file locations, JAR repository locations, service enabling attributes, and so on.

The Properties panel lists Java system properties for the Container. This information controls the Java Virtual Machine used to run the Container.

Figure 3.6 Configuration and Properties panels



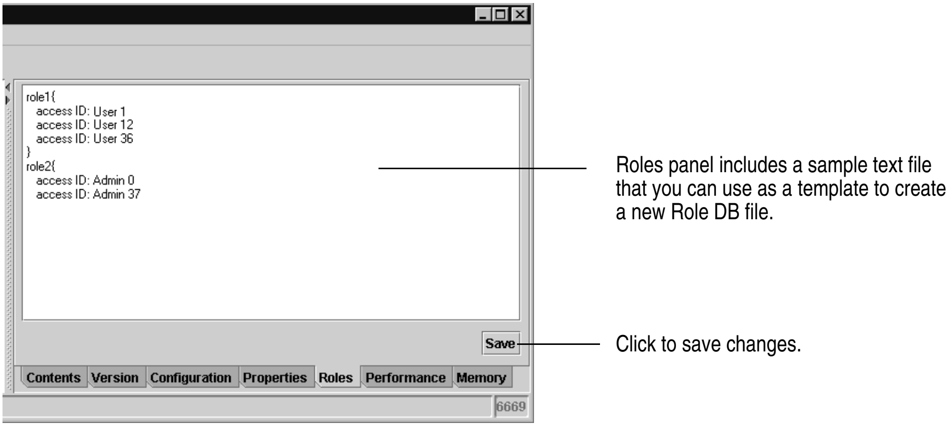
Roles panel

The Roles panel displays the contents of the Roles DB file defined for the Container. You can use the Roles panel to create a new Roles DB file or edit the contents of an existing file. For more information about this file, see the Inprise Application Server *Security Service Guide*.

To edit the file:

1. Open the Roles panel, and click to set an insertion point.
2. Enter new information as desired.
3. Click Save to save the changes.

Figure 3.7 Roles panel



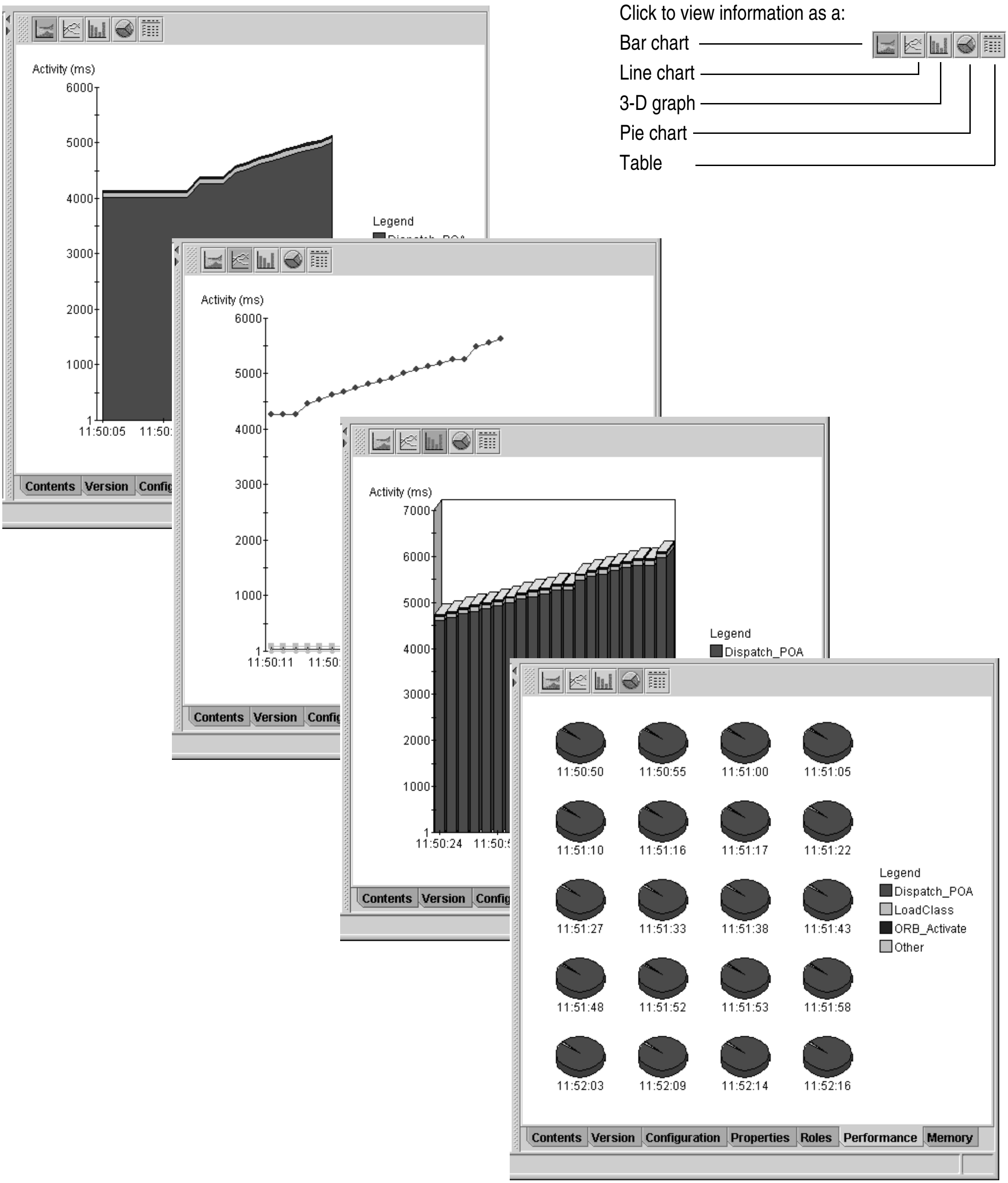
Viewing performance information

In addition to container attributes, you can use the Console to view performance information about Containers. Use the Performance and Memory tabs in the Content pane to view this information.

Performance information is useful for monitoring the amount of time the Container spends on deployment activities such as Dispatching or Activation. Use this information to troubleshoot network performance problems (for example, when a Container spends a much larger percentage of time on dispatching instead of business logic).

You can view performance information in a variety of formats including: bar charts, line charts, 3D charts, and numerical tables.

Figure 3.8 Formats for performance information

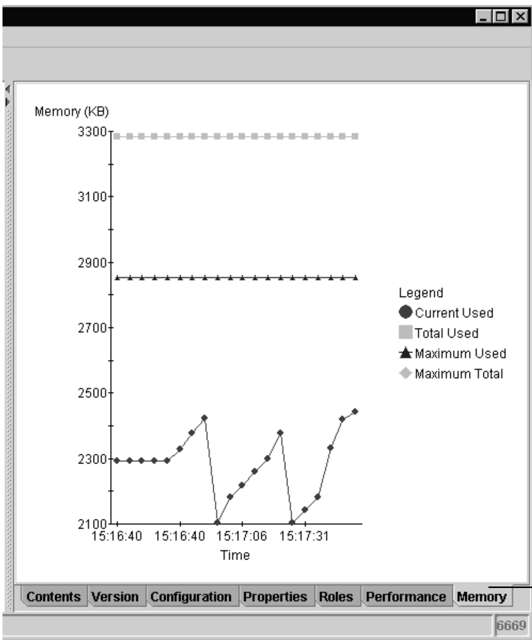


Use the Memory tab to display memory used by the Container. Memory usage includes:

Table 3.1 Memory usage information

Activity	Description
Current used	Memory used at a given point in time.
Total used	Total memory available at a given point in time.
Maximum used	Maximum amount of memory used by the Container since it started.
Maximum total	Maximum amount of total memory available since the Container started.

Figure 3.9 Memory usage



Click to display memory usage for the Container.

Viewing bean attributes

The Inprise EJB Container supports three types of enterprise beans:

- **Stateless Session beans:** In these beans, any state (if required) is maintained by the client or in an external location such as a database. Because no state is maintained, stateless Session beans aren't tied to any specific client; any available instance of a stateless Session bean can be used to service a client.
- **Stateful Session beans:** In these beans, state is maintained by the enterprise bean. This means the application server manages client-bean pairs. Each instance of the enterprise bean is created on behalf of a client and is intended to be a private resource to that client. Stateful Session beans can access persistent resources (such

as databases and files) but unlike Entity beans, they don't actually represent the data.

- **Entity beans:** These beans are persistent objects and represent an object view of data stored in permanent storage. An Entity bean lives in an EJB Container in much the same way a record lives in a database. Unlike Stateful Session beans, Entity beans can be accessed by multiple clients concurrently. This concurrency is managed by the EJB Container.

Beans include a variety of attributes you can view. These attributes include bean identity information, environment settings, and so on.

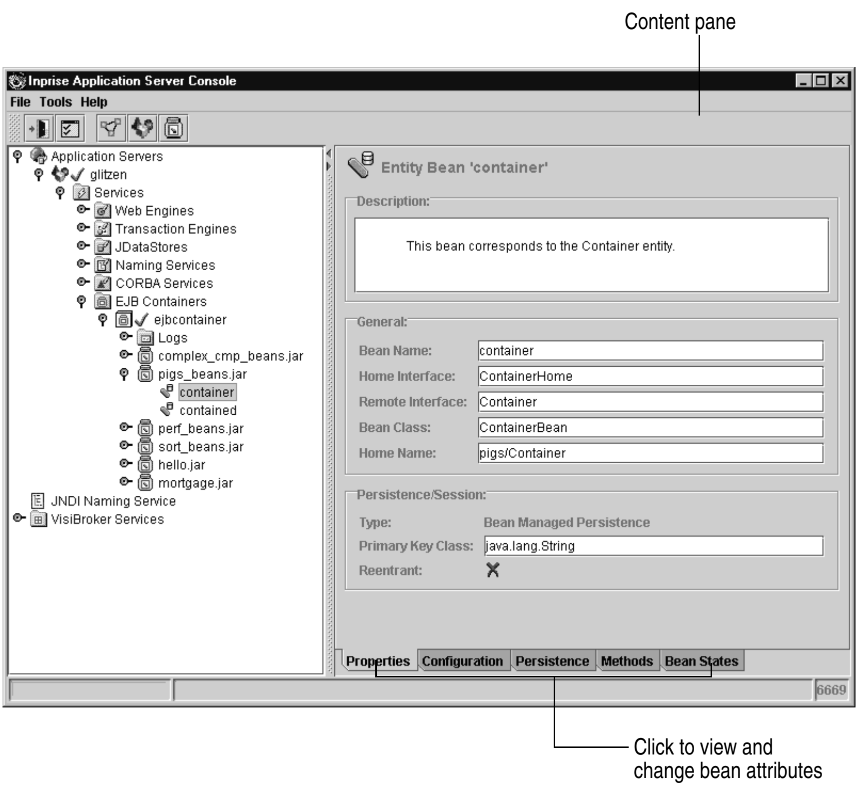
Note This section includes information about bean attributes. For information about container attributes, see "Viewing container attributes" on page 3-3.

To view bean attributes:

- Open the Navigation pane in the Inprise Application Server Console and select the enterprise bean you wish to view.

  Attributes for the enterprise bean appear in the Content pane.

Figure 3.10 Bean attributes



Note Not all beans display all the attributes specified. For example, the Persistence property only appears if you have selected an Entity bean.

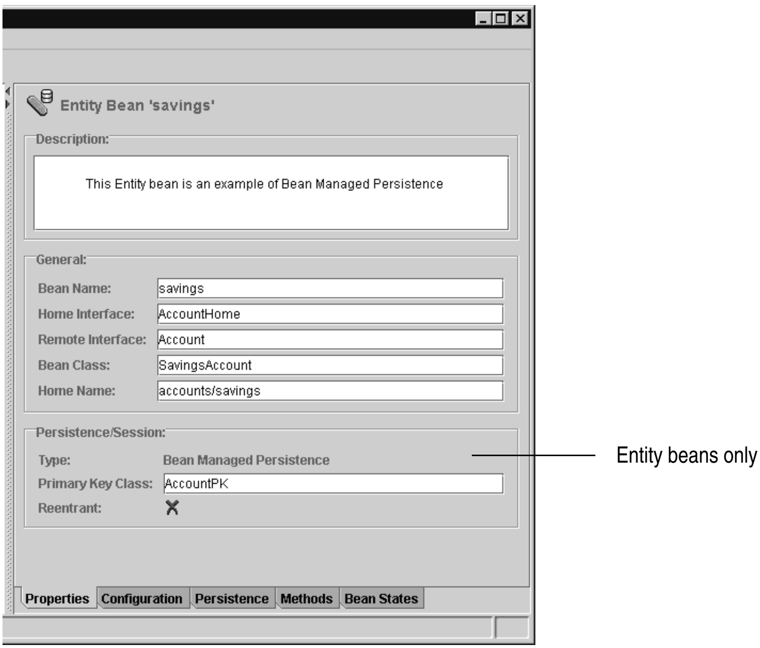
Properties panel

The Properties panel displays baseline information about the enterprise bean. This includes the bean's class name and the class names for the bean's home and remote interface.

For Entity beans, it also indicates whether the enterprise bean is reentrant. A reentrant bean can accept more than one call in the same transaction context. (Inprise recommends you avoid making a bean reentrant.)

For Stateful Session beans, timeout information is also provided.

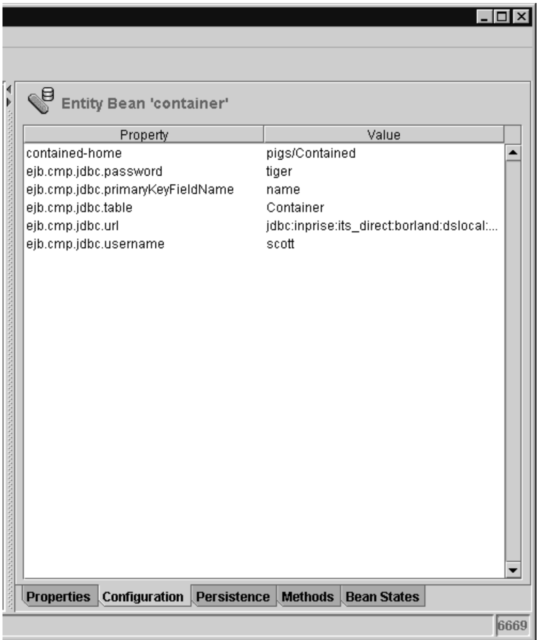
Figure 3.11 Properties panel



Configuration panel

The Configuration panel lists Inprise-specific environment and configuration information defined for the enterprise bean. This information includes persistence storage information, database location and login information, and so on.

Figure 3.12 Configuration information



Persistence panel

This panel appears for Entity beans only. It does not appear for Session beans.

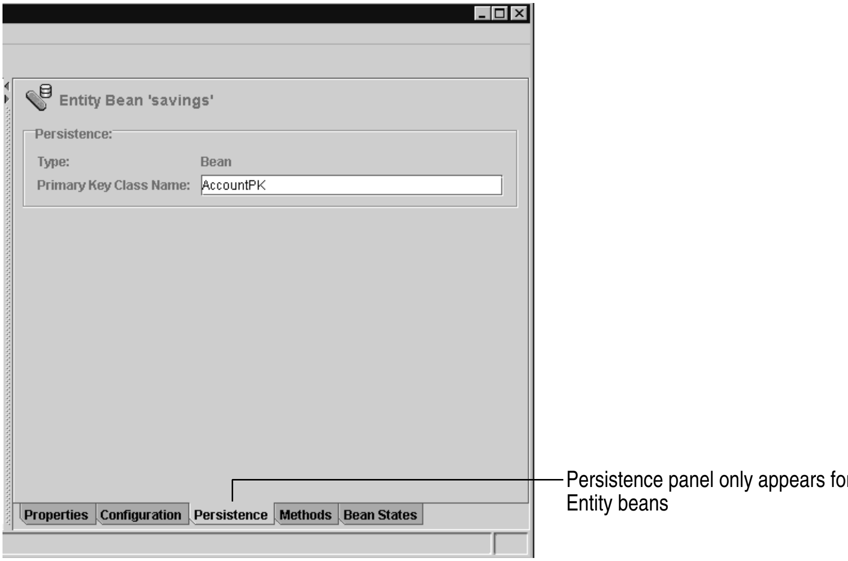
For container-managed persistence, the panel displays:

- Names of database fields managed by the Container
- Class name of the enterprise bean's primary key

  For Entity EJBs, a primary key is an object of user-defined type that can be used to look up a reference to the enterprise bean.

For bean-managed persistence, the panel displays the bean's primary key.

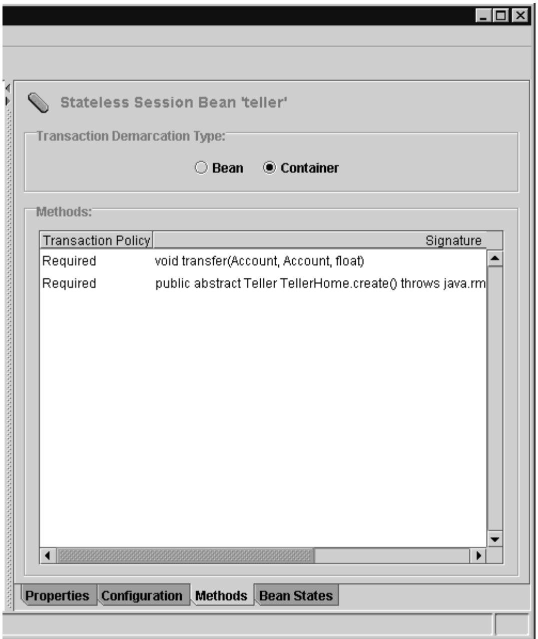
Figure 3.13 Persistence information



Methods panel

The Methods panel lists all the methods included in the enterprise bean and referenced by the bean. It also displays the Transaction Policies assigned to each method.

Figure 3.14 Method information



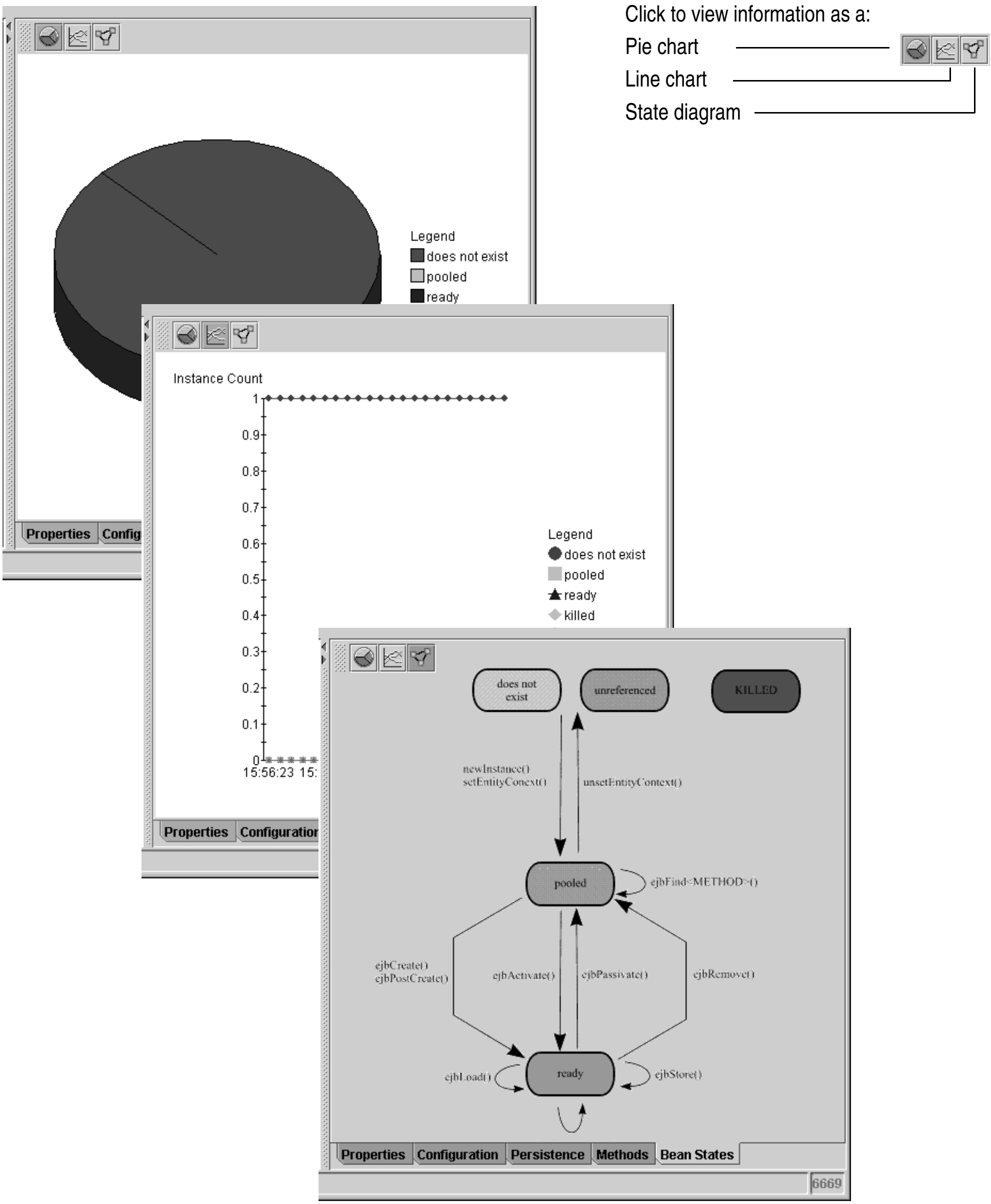
State panel

The State panel displays an overview of the enterprise bean's life cycle and the proportion of time a bean spends in each phase. For example, the State panel shows how much time a bean spends on invocations of its business methods vs. other tasks. This information is useful for troubleshooting potential network and performance problems.

You can view State information as a pie chart or line chart. To help you interpret the information provided, state diagrams of the bean's life cycle are provided for each type of bean: Stateless Session, Stateful Session, and Entity.

For more information on EJB state diagrams and life cycle, see the Inprise Application Server *Enterprise JavaBeans Programmer's Guide*.

Figure 3.15 State information



Deploying EBJs

Deploying an enterprise bean means installing the EJB JAR file in a Container. Installation includes:

- Merging support files that make up the enterprise bean
- Registration of the enterprise bean with a naming service
- Executing the Transaction Policies and Security Roles
- Adding Inprise-specific information

You can install any number of beans in a Container.

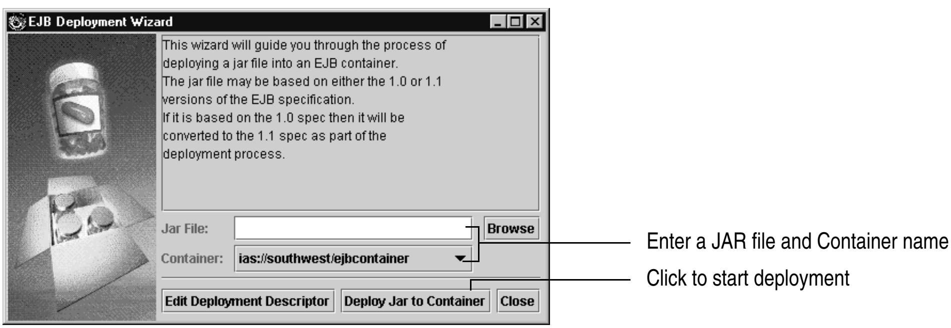
You can use the Deployment wizard included with the Inprise Application Server Console to deploy the enterprise bean, or you can use commands in the EJB toolkit. The following section describes how to use the Deployment wizard to deploy beans. For more information on the EJB toolkit, see the Inprise Application Server *Enterprise JavaBeans Programmer's Guide*.

To deploy an EJB:

1 Open the Application Server Console and choose Deployment Wizard from the Tools menu.

The EJB Deployment wizard starts.

Figure 3.16 Deploying an EJB JAR file



2 Enter the name of the JAR file that contains the beans you want to deploy.

You can type in the name or use the Browse button to locate the JAR file.

3 Select the Container you want to deploy the enterprise bean to.

The Container pop-up menu includes a list of Containers that are running.

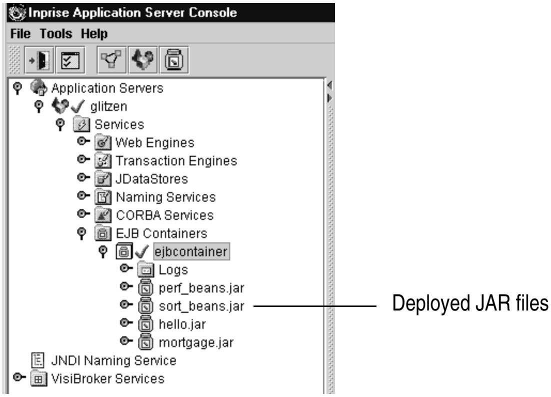
4 Click the Deploy JAR to Container button and follow the onscreen instructions.

During deployment, the contents of the JAR file is verified and Inprise-specific information is added. The Deployment wizard:

- Creates the required stubs and skeletons for the enterprise bean
- Deploys the JAR to a Container
- Validates the deployment descriptor

When the enterprise bean is deployed successfully, the JAR file appears in the Container folder.

Figure 3.17 Viewing the JAR file in the Container



After the enterprise bean is deployed, you can use the Deployment Descriptor Editor to change deployment information as desired. For example, you can use the Deployment Descriptor Editor to specify or change the Transaction Policies and Security Roles for the enterprise bean. For more information, see Chapter 4.

Note If errors are found during deployment, the Deployment Descriptor Editor opens automatically to enable you to correct problems.

Converting EJB 1.0 to EJB 1.1

The Deployment wizard converts an EJB 1.0 serialized deployment descriptor (or a set of descriptors) to an EJB 1.1 XML deployment descriptor. This wizard can be used to migrate enterprise beans from other EJB vendors to the Inprise EJB Container.

The wizard generates the DTD embedded within the XML deployment descriptor, which means that the Inprise Deployment Descriptor Editor (or any standard XML editor) can edit the generated XML deployment descriptor.

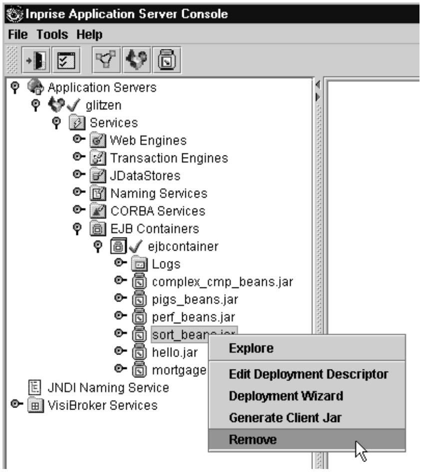
See the Inprise Application Server *Enterprise JavaBeans Programmer's Guide* for more information about changing EJB 1.0 deployment descriptors to EJB 1.1 deployment descriptors.

Removing EJBs from a Container

To remove an EJB JAR file from a Container:

1. Open the Container folder in the Console.
2. Select the JAR file you wish to delete and choose Remove from the Context menu.

Figure 3.18 Removing a JAR file



The JAR file is removed from the Container.

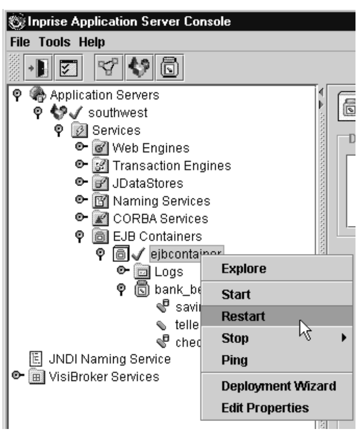
Restarting a Container

Occasionally, you may need to restart the Container for changes to take effect (for example, to make sure changes to the EJB JAR file are updated on the server).

To restart a Container:

1. Locate the Container in the Navigation pane.
2. Right-click the Container and choose Restart from the Context menu.

Figure 3.19 Restarting the Container



After a few moments, the Container restarts. When the Container has completely restarted, a check mark appears next to the Container.

Creating a new Container

If desired, you can create additional versions of the Inprise Application Server's EJB Container and run these Containers out of process.

To create a new Container:

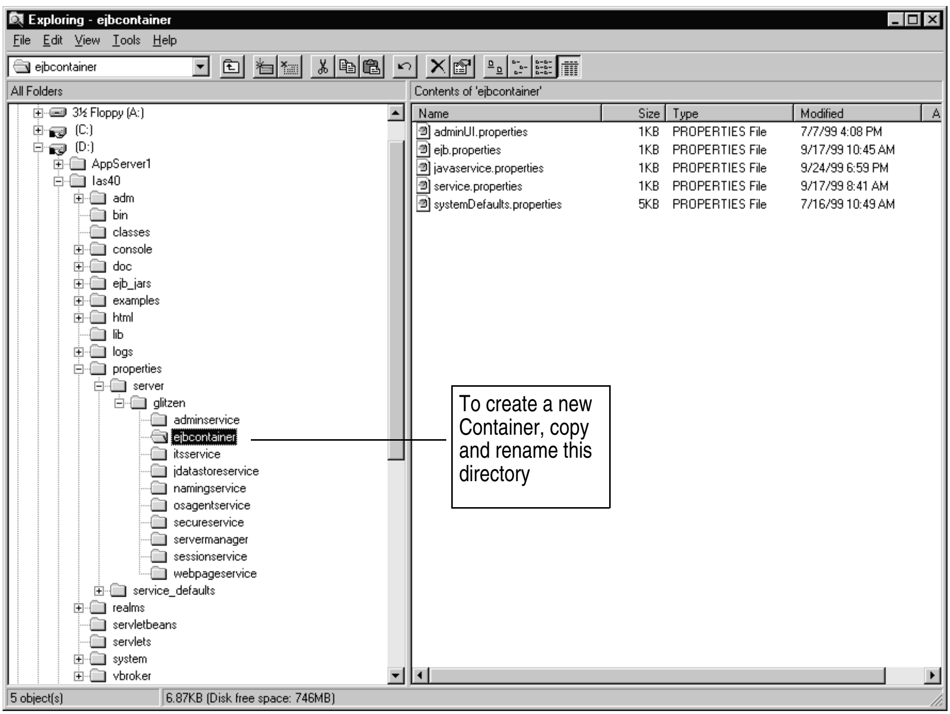
1 Using a file system navigation tool, locate the Server properties directory for the Inprise Application Server

```
install_dir/properties/server/server_name/
```

where *install_dir* is the Inprise Application Server installation directory and *server_name* is the name of the server.

2 Copy the ejbcontainer directory and give the directory a new name. (This is the name of the new Container.)

Figure 3.20 Container properties folder



3 In the directory you just created, locate the server.properties file and use a text editor to change the following:

- Change the service.name entry to the name of the new Container.
- Change the service.description entry to a description of the new Container.

The following is a partial example of a server.properties file.

```
#Service information
service.name=mycontainer
service.description=Custom Container for EJBs
service.vendor=Inprise Corp.
service.version=4.0
...
```

4. Locate the administer_services.properties file (*install_dir*/properties/server/*server_name*/administer_services.properties) and add the new container name to the server.service.administer entry.
5. Restart the server. Then, use the Console (or Web Administration tool) to start the new Container.

Note After you create a new Container, you can use the Console to change container attributes, if desired. See "Viewing container attributes" on page 3-3.