```
<glassfish-web-app>
   ...
   <jsp-config>
      <property name=keepgenerated value=true />
   </jsp-config>
</glassfish-web-app>
```

If you include this property when you deploy the WAR file, the generated source is kept in *domain-dir*`/generated/jsp/`*app-name*`/`*module-name* for an application, or *domain-dir*`/generated/jsp/`*module-name* for an individually-deployed web module.

For more information about JSP precompilation, see "`jsp-config`" on page 181.

- **Web Context Parameters.** You can set web context parameters after deployment. See the following sections:
  - "To Set a Web Context Parameter" on page 61
  - "To Unset a Web Context Parameter" on page 62
  - "To List Web Context Parameters" on page 63
- **Web Environment Entries.** You can set web environment entries after deployment. See the following sections:
  - "To Set a Web Environment Entry" on page 64
  - "To Unset a Web Environment Entry" on page 65
  - "To List Web Environment Entries" on page 65

EJB Module Deployment Guidelines

Note – The GlassFish Server Web Profile supports the EJB 3.1 Lite specification, which allows enterprise beans within web applications, among other features. The GlassFish Server Full Platform Profile supports the entire EJB 3.1 specification. For details, see JSR 318 (`http://jcp.org/en/jsr/detail?id=318`

The following guidelines apply to deploying an EJB module in GlassFish Server:

- **JNDI Name.** — If no JNDI name for the EJB JAR module is specified in the `jndi-name` element immediately under the `ejb` element in `glassfish-ejb-jar.xml`, or there is no `glassfish-ejb-jar.xml` file, a default, non-clashing JNDI name is derived. A warning message is logged, recording the JNDI name used to look up the EJB JAR module.

  Because the EJB 3.1 specification defines portable EJB JNDI names, there is less need for GlassFish Server specific JNDI names. By default, GlassFish Server specific default JNDI names are applied automatically for backward compatibility. To disable GlassFish Server specific JNDI names for an EJB module, set the value of the `<disable-nonportable-jndi-names>` element in the `glassfish-ejb-jar.xml` file to `true`. The default is `false`.

- **Stateful Session Bean and Timer State.** — Use the `--keepstate` option of the `redeploy`(1) subcommand or the `<keepstate>` element in the `glassfish-ejb-jar.xml` file to retain stateful session bean instances and persistently created EJB timers across redeployments. The `--keepstate` option of the `redeploy` subcommand takes precedence. The default for both is `false`. This option is not supported and ignored in a clustered environment.

  Some changes to an application between redeployments can ent this feature from working properly. For example, do not change the set of instance variables in the SFSB bean class. Other examples would be changes to EJB names, or adding or removing EJBs to or from an application.
- **Stubs and Ties.** — Use the `get-client-stubs`(1) subcommand in remote mode to retrieve stubs and ties.
- **Compatibility of JAR Visibility Requirements.** — Use the `compatibility` element of the `glassfish-application.xml` or `glassfish-ejb-jar.xml` file to specify the GlassFish Server release with which to be backward compatible in terms of JAR visibility requirements for applications. The current allowed value is `v2`, which refers to GlassFish Server version 2 or GlassFish Server version 9.1 or 9.1.1. The Java EE 6 platform specification imposes stricter requirements than Java EE 5 did on which JAR files can be visible to various modules within an EAR file. Setting this element to `v2` removes these Java EE 6 restrictions.

Deploying a Connector Module

Deploying a stand-alone connector module allows multiple deployed Java EE applications to share the connector module. A resource adapter configuration is automatically created for the connector module.

- "To Deploy and Configure a Stand-Alone Connector Module" on page 68
- "Redeploying a Stand-Alone Connector Module" on page 69
- "Deploying and Configuring an Embedded Resource Adapter" on page 70

▼ To Deploy and Configure a Stand-Alone Connector Module

As an alternative to Step 3 through Step 6, you can define application-scoped resources in the `glassfish-resources.xml` deployment descriptor. For more information, see "Application-Scoped Resources" on page 75.

1 Ensure that the server is running.

Remote commands require a running server.

2 Deploy the connector module by using the `deploy`(1) subcommand.