Sun GlassFish Web Stack 1.5 Getting Started Guide

Sun Microsystems Inc.

Part No: 821–0323
2009

DOCUMENTATION IS PROVIDED “AS IS” AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

090727@22510

Contents

CHAPTER 1

Getting Started with Sun GlassFish Web Stack 1.5

The Sun GlassFish Web Stack (Web Stack) software provides a collection of open source servers and interpreters that are bundled and made available for various platforms including Solaris 10 OS, OpenSolaris OS, and Red Hat Enterprise Linux (RHEL).

The following getting started guide has information for all the supported platforms.

Sun GlassFish Web Stack 1.5

The Sun GlassFish Web Stack (Web Stack) software provides a collection of open source servers and interpreters that are bundled and made available for various platforms including Solaris 10 OS, OpenSolaris OS, and Red Hat Enterprise Linux (RHEL).

The Web Stack software is supported on the following platforms:

1. **Solaris 10 OS** (Update 5 or higher).
2. **OpenSolaris OS** (2009.06).
3. **Linux** (Red Hat Enterprise Linux (RHEL) 5 Update 2 or higher (32-bit and 64-bit kernels)). Support is available only for x86 platforms.

This getting started guide provides information for the following platforms:

- OpenSolaris 2009.06 OS
- Solaris 10 OS
- Red Hat Enterprise Linux

Throughout this document examples are shown using commands that can be executed from a console (shell). The convention followed in this document is that for commands that needs to be run with super user privileges, # notation is used and for commands that can be run as any other user, $ notation is used.

For instance,

```
# <PERFORM_ADMIN_TASKS>
```

```
$ <PERFORM_USER_TASKS>
```

The Web Stack software includes the following software:

1. **Server Software** - Apache Web Server, Apache Tomcat, Squid Proxy Server, lighttpd.
2. **DB Software** - MySQL Server.
3. **Programming Environment** - PHP, Perl, Ruby, and Python.
4. **Caching Software** - memcached.

Changes in the Current Release

The following table lists the open source components included in the Web Stack software in this release:

Web Stack Component	Previous Release (1.4)	Current Release (1.5)
Apache HTTP Server	2.2.9	**2.2.11**
mod_jk	1.2.25	**1.2.27**
mod_perl	2.0.2	2.0.2
lighttpd	1.4.19	**1.4.21**
memcached	1.2.5	1.2.5
MySQL Server	5.0.67	**5.0.67 + 5.1.30**
PHP	5.2.6	**5.2.9**
Python	2.5.2	**2.6.1**
Ruby	1.8.6 p287	**1.8.7**
Ruby Gems	0.9.4	**1.3.1**
Apache Tomcat Server	5.5.27	**6.0.18**

Ruby and Python are not provided as part of the Web Stack software for Red Hat Enterprise Linux. You need to use the interpreters provided by the Red Hat platform.

Getting Web Stack

Note – After downloading the Web Stack software (For Red Hat Enterprise Linux and Solaris 10 OS), follow the instructions provided in the Web Stack Installation and Configuration Guide for information on installing and configuring the stack on a specific platform.

Read the following sections for getting the Web Stack software in OpenSolaris 2009.11 OS.

Installing Individual Web Stack Components

Note – Web Stack is fully integrated with OpenSolaris 2009.06 and can be easily installed by using the Package Manager GUI or `pkg install` commands.

OpenSolaris 2009.06 release allows you to install Web Stack components individually. The following table lists the packages and the commands necessary to install them.

Component Name	Package Name	Command to install the component
Apache 2.2 core, documentation, `mod_perl`, and `mod_auth_gss`	`SUNWapch22,SUNWapch22d`	`pkg install SUNWapch22 SUNWapch22d`

Apache 2.2 modules	`SUNWapch22m-security,`
	`SUNWapch22m-jk,`
	`SUNWapch22m-fcgid,`
	`SUNWapch22m-dtrace,`
	`SUNWapch22m-sed`
		`pkg install SUNWapch22m-security SUNWapch22m-jk SUNWapch22m-fcgid SUNWapch22m-dtrace SUNWapch22m-sed`

PHP5	`SUNWphp52, SUNWphp52d,`
	`SUNWphp52-mysql,`
	`SUNWphp52-pgsql,`
	`SUNWapch22m-php52`
		`pkg install SUNWphp52 SUNWphp52d SUNWphp52-mysql SUNWphp52-pgsql SUNWapch22m-php52`

MySQL 5.1 Server	`SUNWmysql51`	`pkg install SUNWmysql51`
MySQL 5.1 Client Libraries	`SUNWmysql51lib`	`pkg install SUNWmysql51lib`
MySQL 5.1 Regression Tests	`SUNWmysql51test`	`pkg install SUNWmysql51test`
MySQL 5.0 Server	`SUNWmysql5`	`pkg install SUNWmysql5`
MySQL 5.0 Client Libraries	`SUNWmysql5lib`	`pkg install SUNWmysql5lib`
MySQL 5.0 Regression Tests	`SUNWmysql5test`	`pkg install SUNWmysql5test`

Component Name	Package Name	Command to install the component
Squid	`SUNWsquid`	`pkg install SUNWsquid`
lighttpd	`SUNWlighttpd14`	`pkg install SUNWlighttpd14`
memcached	`SUNWmemcached`	`pkg install SUNWmemcached`
Web Stack user interface	`webstackui`	`pkg install webstackui`
Apache Tomcat Server	`SUNWtcat`	`pkg install SUNWtcat`

Note – You need to be the `root` user to run the previously mentioned commands.

You can install the components using the Package Manager graphical user interface for IPS. To install, update, and manage packages using the Package Manager, see this document.

For installing the Web Stack components in Red Hat Enterprise Linux and Solaris 10 OS, see the Installation Guide

Installing the Whole Web Stack Bundle

If you are an AMP developer, you can download the AMP cluster package, which is a super set of all individual AMP packages. To download the AMP package, execute the following command:

```
# pkg install amp-dev
```

Note – For executing the `pkg install` command, you should be a root user. You can also invoke the `pkg install` command with `pfexec` command as a non-root user. For instance, as a non-root user, execute `pfexec pkg install amp-dev`

Note – `amp-dev` package is the set of all the web stack packages. The approximate size of this package is 650 mega bytes (MB). You need to download and install this package for AMP development. If you are unable to download this package, you can download individual packages as shown in the following sections.

If you need just the AMP runtime components, you can download the `amp` package. This package does not include any development environment packages.

To download and install the amp package, execute the following command:

```
# pkg install amp
```

Available Packages

See the Installation Guide for more information.

The following table shows the list of all packages available through the Web Stack software in Solaris 10 OS:

Component Name	Package Name
Apache HTTP Server 2.2.11	`sun-apache22`
mod_dtrace 0.3.1	`sun-apache22-dtrace`
mod_fcgid 2.2	`sun-apache22-fcgid`
mod_jk 1.2.27	`sun-apache22-jk`
mod_perl 2.0.2	`sun-apache22-perl`
mod_sed	`sun-apache22-sed`
mod_security	`sun-apache22-security2`
mod*auth*gss	`sun-apache22-authgss`
-	-
lighttpd 1.4.21	`sun-lighttpd14`
memcached 1.2.5	`sun-memcached`
-	-
MySQL 5.1.30	`sun-mysql51`
MySQL 5.0.67	`sun-mysql50`
MySQL 5.1.30 client libraries	`sun-mysql51lib`
Regression tests for MySQL	`sun-mysql51test` and `sun-mysql50test`
-	-
PHP 5.2.9	`sun-php52`
APC Extension	`sun-php52-apc`
IDN Extension	`sun-php52-idn`
Memcache Extension	`sun-php52-memcache`
Suhosin Extension	`sun-php52-suhosin`
TCPWrap Extension	`sun-php52-tcpwrap`
XDebug Extension	`sun-php52-xdebug`

Component Name	Package Name
DTrace Extension	sun-php52-dtrace
MySQL Extension	sun-php52-mysql
PostgreSQL Extension	sun-php52-pgsql
-	-
Python 2.6	sun-python26
Ruby 1.8.7	sun-ruby18
Squid 2.7.STABLE5	sun-squid
Apache Tomcat 6.0.18	sun-tomcat6
Support libraries and registration tool	sun-wsbase

Note – Always use the Web Stack installer to install individual packages or the whole stack.

See the Web Stack Installation and Configuration Guide.

See the Installation Guide for more information.

The following table shows the list of all packages available through the Web Stack software for Red Hat Enterprise Linux:

Apache HTTP Server 2.2.11	sun-apache22-2.2.11
mod_fcgid 2.2	sun-apache22-fcgid-2.2
mod_jk 1.2.27	sun-apache22-jk-1.2.27
mod_perl	sun-apache22-perl-2.0.2
mod_sed	sun-apache22-sed-1.0
mod_security	sun-apache22-security2-2.5.9
-	-
lighttpd 1.4.21	sun-lighttpd14-1.4.21
memcached 1.2.5	sun-memcached-1.2.5
-	-
MySQL 5.0.67	sun-mysql50-5.0.67

MySQL 5.1.30	`sun-mysql51-5.1.30`
MySQL Libraries 5.1.30	`sun-mysql51lib-5.1.30`
Regression tests for MySQL 5.0.67	`sun-mysql50test-5.0.67`
Regression tests for MySQL 5.1.30	`sun-mysql51test-5.1.30`
-	-
PHP 5.2.9	`sun-php52-5.2.9`
APC Extension	`sun-php52-apc-3.0.19`
IDN Extension	`sun-php52-idn-0.2.0`
Memcache Extension	`sun-php52-memcache-2.2.5`
Suhosin Extension	`sun-php52-suhosin-0.9.27`
TCPWrap Extension	`sun-php52-tcpwrap-1.1.3`
XDebug Extension	`sun-php52-xdebug-2.0.4`
MySQL Extension	`sun-mysql-5.2.9`
PostgreSQL Extension	`sun-pgsql-5.2.9`
-	-
Squid 2.7.STABLE6	`sun-squid-2.7.STABLE6`
Apache Tomcat 6	`sun-tomcat6-6.0.18`
Support libraries and registration tool	`sun-wsbase-1.5`

Note – Always use the Web Stack installer to install individual packages or the whole stack.

The following list provides you with information on what is **not supported** by the Web Stack software on the Red Hat platform:

1. Ruby interpreter (use the OS-provided Ruby interpreter).
2. Python interpreter (use the OS-provided Python interpreter).
3. `mod_dtrace` plug-in module for Apache (Linux doesn't support DTrace).
4. Dtrace extension for PHP (Linux doesn't support DTrace).
5. `mod_auth_gss`

Web Stack UI

Before you can use Web Stack for web development, you need to initialize the environment for your login. Every new user logged into the system must initialize their own environment. After downloading and installing the amp-dev package, initialize the Web Stack environment. Click **Applications** > **Developer Tools** > **Web Stack Initialize** to start the initialization process.

To download and install the Web Stack UI, execute the following command:

```
# pkg install webstackui
```

Note – If you have already installed amp-dev package, webstackui package is also installed. You need to install webstackui only when you are installing the AMP packages separately or if you have installed the amp package.

Since webstackui package depends on other Apache packages, if you have not already installed the Apache Web Server, it will be automatically downloaded when you install the webstackui package. As mentioned earlier, Web Stack UI is just the GUI option to control the AMP components. You still need to download and configure AMP packages if you have not installed the amp-dev package.

After installing the Web Stack UI, you can create and run the co-packaged sample applications from the menu item. Go to **Applications** > **Developer Tools** > **Web Stack Admin** > **Sample App** to find these menu items.

The Web Stack menuitems provide a quick way to start/stop the Apache Web Server and the MySQL Server.

For instance, to start the Apache Web Server in OpenSolaris 2009.06:

1. Click **Applications** > **Developer Tools** > **Web Stack Admin** > **Initialize**
2. Click **Applications** > **Developer Tools** > **Web Stack Admin** > **Start Apache2/MySQL Servers**

After Initialization, you can customize several aspects of the Web Stack components.

From the Launch menu, select All Applications > Developer Tools > Web Stack Admin > Options. The Web Stack Admin Options panel is displayed. The General tab enables you to configure the Servers to start automatically at system start up, or each time when you log in.

For advanced options such as editing the configuration files, click Advanced Configuration the Apache, MySQL, and PHP tabs.

Note – You must restart the servers for the changes to take effect.

You can edit the `httpd.conf` file for Apache advanced configuration options. For information about how to configure Apache, refer to the documentation at `/usr/apache2/2.2/manual/configuring.html.en`.

For PHP advanced configuration, you can edit the `php.ini` file. For information about how to configure PHP using the `php.ini` file, see the documentation at `/usr/php/5.2/doc/html/configuration.html`.

For MySQL advanced configuration, you can edit the `my.cnf` file available at `/etc/mysql/5.1/my.cnf`.

Not applicable for Solaris 10 OS and Red Hat Enterprise Linux.

Web Stack IPS Repository

Web Stack IPS repository is an experimental repository hosting applications and tools pertaining to the Web Stack. For example, you can find applications like Drupal or PHPMyAdmin in the Web Stack repository, which you can download and install instead of downloading the applications directly from their sites. The applications that are available in the Web Stack repository are pre-configured to work with other components in the Web Stack.

The Web Stack repository is available at http://pkg.opensolaris.org/webstack.

You can set additional repositories in the OpenSolaris OS using `pkg set-publisher` command. For more information on adding the Web Stack repository, see this document.

For more information about the Web Stack IPS repository, read this document.

While you are free to install software from the Web Stack IPS repository, support will not be provided through the Sun GlassFish Portfolio subscription.

Not applicable for Solaris 10 OS and Red Hat Enterprise Linux.

Apache Web Server

In OpenSolaris 2009.06, if you have installed the optional package `amp-dev`, you don't need to download other Apache Web Server packages separately.

The Web Stack comes with Apache 2.2.11 Web Server fully configured and optimized for PHP deployment. The NetBeans IDE lets you add Apache web server in the servers list so the applications that you build including PHP based web applications can be deployed to the Apache web server. To set up Apache Web Server in the NetBeans IDE, refer to the later sections in this document.

To start the Apache Web Server in OpenSolaris 2009.06, execute the following command:

```
# svcadm enable -s apache22
```

For more information on the configuration file layout, see the Installation and Configuration Guide

The Web Stack software comes with Apache 2.2.11 Web Server fully configured and optimized for PHP deployment.

If you are using the NetBeans IDE, you can add the Apache Web Server in the servers list so the applications that you build including PHP based web applications can be deployed to the Apache Web Server directly from the IDE. To set up the Apache Web Server in the NetBeans IDE, refer to the later sections in this document.

To start the Apache Web Server on Solaris 10 OS, execute the following command:

```
# svcadm enable -s sun-apache22
```

You need to have the necessary permission to execute the previous command.

Note – The previous command starts the Apache Web Server in the 32-bit prefork MPM mode (the default mode).

To start the Apache Web Server on Red Hat Enterprise Linux, execute the following command:

```
# /sbin/service sun-apache22 start
```

To start the Apache Web Server (worker MPM) on Red Hat Enterprise Linux, execute the following command:

1. Stop `sun-apache22` service.

   ```
   $ /sbin/service sun-apache22 stop
   ```

2. Edit `/etc/opt/sun/webstack/apache2/2.2/envvars` file to set the `HTTPD` variable, as shown below:

   ```
   HTTPD=/opt/sun/webstack/apache2/2.2/bin/httpd.worker
   ```

3. Start `sun-apache22` service.

   ```
   $ /sbin/service sun-apache22 start
   ```

Automatic Startup of Servers

Refer to this section.

Using SMF to Manage Services

Apache Web Server's SMF properties are listed in the following table:

SMF Property Name	Value
httpd/server_type	prefork (default) or worker
httpd/enable_64bit	true or false(default)
httpd/startup_options	valid `apachectl` options

Enabling the server via SMF will keep the current server properties for addressing mode and MPM, which are initially 32-bit and prefork; these settings can be displayed and changed as follows:

1. **To list the SMF properties**

   ```
   # svccfg -s <service-name> listprop httpd/*
   ```

2. **To switch to a 32-bit-prefork server (one-time action)**

   ```
   # svccfg -s <service-name> setprop httpd/enable_64bit=false
   # svccfg -s <service-name> setprop httpd/server_type=prefork
   # svcadm refresh <service-name>
   ```

3. **To switch to a 64-bit-prefork server (one-time action)**

   ```
   # svccfg -s <service-name> setprop httpd/enable_64bit=true
   # svccfg -s <service-name> setprop httpd/server_type=prefork
   # svcadm refresh <service-name>
   ```

4. **To switch to a 32-bit-worker server (one-time action)**

   ```
   # svccfg -s <service-name> setprop httpd/enable_64bit=false
   # svccfg -s <service-name> setprop httpd/server_type=worker
   # svcadm refresh <service-name>
   ```

5. **To switch to a 64-bit-worker server (one-time action)**

   ```
   # svccfg -s <service-name> setprop httpd/enable_64bit=true
   # svccfg -s <service-name> setprop httpd/server_type=worker
   # svcadm refresh <service-name>
   ```

6. **To start the server**

   ```
   # svcadm enable -s <service-name>
   ```

7. **To stop the server**

```
   # svcadm disable <service-name>
```

In the OpenSolaris 2008.11 OS, replace the `<service-name>` with `apache22`.

In the Solaris 10 OS, replace the `<service-name>` with `sun-apache22`.

Note – `mod_perl` and `mod_php` cannot be used with the Apache Web Server running in 64-bit mode.

Not applicable for Red Hat Enterprise Linux.

Setting Up a Secure Web Site with a Self-Signed Certificate

You can set up a secure web server to process web requests over `https`. `mod_ssl`, included with the Apache web server, provides support for `https`.

To configure the SSL for your server, perform the following tasks:

1. Generate a Certificate
2. Modify the Apache configuration files

Generating a Certificate

For the purpose of this example, we will generate a (self-signed) test certificate. This certificate will not be accepted by a browser because it is not signed by a certificate authority (CA), which your browser is setup to trust.

To configure a real public secure server, you can pay a company such as Verisign to sign your certificate.

Note – You can only run one secure web server on a machine (if IP alias is not configured) because the certificates are server wide, and must contain the name of the site they represent. But you can let the server work on both secure and non-secure connections.

For OpenSolaris 2009.06, from a shell, run:

```
$ /usr/bin/openssl req -new -x509 -nodes -out
server.crt -keyout server.key
```

The `openssl` utility is available in OpenSolaris 2009.06 by default.

For **Solaris 10** OS, from a shell, run:

```
$ /usr/sfw/bin/openssl req -new -x509 -nodes
-out server.crt -keyout server.key
```

The `openssl` utility is available in Solaris 10 OS by default.

For Red Hat Enterprise Linux, from a shell, run:

```
$ /usr/bin/openssl req -new -x509 -nodes -out
server.crt -keyout server.key
```

The `openssl` utility is available in Red Hat Enterprise Linux by default.

You will be asked some questions to fill the certificate's attributes. Read this document for more information.

The previous command generates 2 files:

1. **server.crt** - SSL Certificate File
2. **server.key** - SSL Certificate Key File

Modifying `ssl.conf`

There is a sample `ssl.conf` file available under the `samples-conf.d` directory.

In OpenSolaris 2009.06, the `samples-conf.d` directory is under `/etc/apache2/2.2/`.

In Solaris 10 OS, the `samples-conf.d` directory is under `/etc/opt/webstack/apache2/2.2/`.

In Red Hat Enterprise Linux, the `samples-conf.d` directory is under `/etc/opt/sun/webstack/apache2/2.2/`.

Move the `ssl.conf` file to the `conf.d` directory.

In OpenSolaris 2009.06, the `conf.d` directory is under `/etc/apache2/2.2/`.

```
# cp /etc/apache2/2.2/samples-conf.d/ssl.conf
/etc/apache2/2.2/conf.d/ssl.conf
```

In Solaris 10 OS, the `conf.d` directory is under `/etc/opt/webstack/apache2/2.2/`.

```
$ cp /etc/opt/webstack/apache2/2.2/samples-conf.d/ssl.conf
/etc/opt/webstack/apache2/2.2/conf.d/ssl.conf
```

In Red Hat Enterprise Linux, the `conf.d` directory is under `/etc/opt/sun/webstack/apache2/2.2/`.

```
# cp /etc/opt/sun/webstack/apache2/2.2/samples-conf.d/ssl.conf
/etc/opt/sun/webstack/apache2/2.2/conf.d/ssl.conf
```

Then modify the `ssl.conf` file in order to change the values of the following parameters:

```
...
SSLCertificateFile "<path to server.crt>"
SSLCertificateKeyFile "<path to server.key>"
...
```

Configuration files are read from the `conf.d` directory automatically.

Restart the Apache Web Server in order for the changes to take effect.

From a browser window, invoke `https://localhost`. Some browsers complain of invalid security certificates. Add an exception and continue to test the secure connection.

Configuring the Solaris Kernel SSL Proxy (KSSL)

Not applicable for Red Hat Enterprise Linux.

The SSL-enabled Apache Web Server can use the services of its Kernel SSL proxy to improve the performance of the HTTPS packets processing. This is acheived by creating an instance of the Kernel SSL service, specifying the SSL proxy port and parameters, and by listening on the proxy port.

To configure the KSSL with the Apache Web Server using the `pkcs12` key type, export the key/cert. in a PKCS12 file as shown in the following command:

```
$ ksslcfg create -f pkcs12 -i /kssl-conf/cert.pk12
-p /kssl-conf/cert.pass -x 8443 443
```

You must run `ksslcfg` to configure your Kernel SSL proxy before you start the Apache Web Server.

To Configure the KSSL with the `pkcs11` key type provided the Server Cert. is in the "Sun Software PKCS#11 softtoken" database, execute the following commands:

```
$ cryptoadm disable metaslot
$ ksslcfg create -f pkcs11 -T "Sun Software PKCS#11 softtoken" -C "Server-Cert" -p ./password.conf -x 8444 444
$ cryptoadm enable metaslot
```

lighttpd Server

lighttpd web server is available through the Web Stack software bundle. lighttpd is an alternative to other web servers because its key features include security, low memory footprint, and ease of use. lighttpd 1.4.21 is available through the Web Stack.

For starting the lighttpd server in the OpenSolaris OS, execute the following command:

```
# svcadm enable -s lighttpd14
```

In OpenSolaris OS, the lighttpd configuration file `lighttpd.conf` and the configuration directory `conf.d` are available under the `/etc/lighttpd/1.4` directory.

For starting the lighttpd server in the Solaris OS, execute the following command:

```
# svcadm enable -s sun-lighttpd14
```

In Solaris OS, the lighttpd configuration file `lighttpd.conf` and the configuration directory `conf.d` are available under the `/etc/opt/webstack/lighttpd/1.4` directory.

For starting the lighttpd server in the Red Hat Enterprise Linux, execute the following command:

```
# /sbin/service sun-lighttpd14 start
```

In Red Hat Enterprise Linux OS, the lighttpd configuration file `lighttpd.conf` and the configuration directory `conf.d` are available under the `/etc/opt/sun/webstack/lighttpd/1.4` directory.

MySQL DB Server

The MySQL server is the most widely used relational database server and the MySQL package is available for download from the IPS repository.

The following table shows the locations where you can find the MySQL files:

Solaris 10 OS	`/opt/webstack/mysql/5.1`

To start the MySQL Server on Solaris 10, execute the following command:

```
# svcadm enable -s sun-mysql51
```

You need to have the necessary permission to execute the previous command.

Red Hat Linux Enterprise Linux	`/opt/sun/webstack/mysql/5.1`

To start the MySQL Server on Red Hat Enterprise Linux, execute the following command:

```
# /sbin/service sun-mysql51 start
```

OpenSolaris 2009.06	`/usr/mysql`

To start the MySQL Server on OpenSolaris 2009.06, execute the following command:

```
# svcadm enable -s mysql:version_51
```

Automatic Startup of Servers

Refer to this section.

Administering the Server

Not applicable for Red Hat Enterprise Linux.

MySQL Server's SMF properties are listed in the following table:

SMF Property Name	Value
mysql/data	/var/mysql/5.1/data
mysql/enable_64bit	false(default) or true

The following examples shows you how you can work with SMF:

1. **To list the SMF properties**

```
# svccfg -s mysql:version_51 listprop mys
l/*
```

1. **To run MySQL 5.1 server as 32-bit (default)**

```
# svccfg -s mysql:version_51 setprop mysq
/enable_64bit=false
# svcadm refresh mysql:version_51
# svcadm disable mysql:version_51
# svcadm enable -s mysql:version_51
```

1. **To run MySQL 5.1 server as 64-bit**

```
# svccfg -s mysql:version_51 setprop mysq
/enable_64bit=true
# svcadm refresh mysql:version_51
# svcadm disable mysql:version_51
# svcadm enable -s mysql:version_51
```

1. **To change the location of MySQL data directory (default is /var/mysql/5.1/data)**

```
# svccfg -s mysql:version_51 setprop mysq
/data=/zpool/data
# svcadm refresh mysql:version_51
# svcadm disable mysql:version_51
# svcadm enable -s mysql:version_51
```

Perform the following steps as a root user to start the MySQL Server:

```
# svcadm enable -s application/database/m
sql:version_51
```

The first command shows how you can import the MySQL Server manifest file. You need to execute this only once. To stop the server, execute the following command:

```
svcadm disable application/database/mysql
version_51
```

Note – The default installation of the MySQL Server creates a file `my.cnf` under `/etc/opt/webstack/mysql/5.1` (Solaris 10 OS) or `/etc/opt/sun/webstack/mysql/5.1` (Red Hat Enterprise Linux). This default configuration is only for very small deployments. There are other sample configurations under `mysql/5.1/my-*.cnf` that you can use for medium to large deployments. Copy one of these files to `my.cnf` before starting the MySQL service.

Setting the MySQL `root` Password

The default password for the MySQL Server `root` user is a blank string (no password). After starting the MySQL Server using the command provided in the previous section, you can connect to the MySQL Server using the following command:

For **Solaris 10** OS:

```
/opt/webstack/mysql/5.1/bin/mysql -u root
```

For **Red Hat Enterprise Linux** OS:

```
/opt/sun/webstack/mysql/5.1/bin/mysql -u
oot
```

For **OpenSolaris 2009.06** OS:

```
/usr/mysql/bin/mysql -u root
```

For changing the default `root` password, perform the following step:

For **Solaris 10** OS:

```
/opt/webstack/mysql/5.1/bin/mysqladmin -u
root password {{<new-pass>}}
```

For **Red Hat Enterprise Linux** OS:

```
/opt/sun/webstack/mysql/5.1/bin/mysqladmin
-u root password {{<new-pass>}}
```

Caution – Change the default MySQL Server root password before deploying it in a production environment.

After setting the MySQL `root` password, if you need to use the MySQL prompt as the MySQL `root` user, you will need to invoke the following command:

```
$ mysql -u root -p
```

Enter the password at the prompt.

Running Multiple Instances of the MySQL Server

You can run multiple versions of the MySQL Servers by altering the default MySQL port (3306) and the default socket file (/tmp/mysql.sock). The Web Stack software has both the MySQL 5.0 Server and the MySQL 5.1 Server. You can choose to run both the servers by making these changes in your configuration files.

In OpenSolaris 2009.06 OS, the MySQL 5.1 configuration file is `/etc/mysql/5.1/my.cnf`.

In Solaris 10 OS, the MySQL 5.1 configuration file is `/etc/opt/webstack/mysql/5.1/my.cnf`.

In Red Hat Enterprise Linux, the MySQL 5.1 configuration file is `/etc/opt/sun/webstack/mysql/5.1/my.cnf`.

MySQL-JDBC Driver

When you need to deploy Java-based web applications in the Apache Tomcat Server or the GlassFish Enterprise Server that uses MySQL DB connection you need to download the appropriate JDBC driver. The MySQL-JDBC driver is not bundled with the Web Stack software.

- Download the MySQL Connector/J (Binary Distribution).

MySQL Connector/J is distributed as a .zip or .tar.gz archive containing the sources, the class files, and the JAR archive named:

```
mysql-connector-java-[version]-bin.jar
```

and a debug build of the driver in a file named:

```
mysql-connector-java-[version]-bin-g.jar
```

Once you have extracted the distribution archive, you can install the driver by placing the:

```
mysql-connector-java-[version]-bin.jar
```

file in your classpath.

For more information, read this document.

MySQL Basics

Before you can use the MySQL DB, you need to create databases and tables. You can either use the PHPMyAdmin application or use the `mysql` command line client.

```
# mysql -u root -p
Enter the root password.
mysql>
```

To create a new database, execute the following command from the MySQL prompt:

```
mysql> CREATE DATABASE database1;
```

To create a new user with all privileges (**for test purpose only**), execute the following command from the MySQL prompt:

```
mysql> GRANT ALL PRIVILEGES ON *.* TO 'yourusername'@'localhost'
IDENTIFIED BY 'yourpassword' WITH GRANT OPTION;
```

To create a new user with lesser privileges (allowing access only to the database named "database1"), execute the following command from the MySQL prompt:

```
mysql> GRANT SELECT, INSERT, UPDATE, DELETE,
CREATE, DROP, INDEX, ALTER, CREATE TEMPORARY TABLES, LOCK TABLES ON database1.* TO 'yourusername'@'localhost' ID
```

Upgrading from MySQL 5.0 Server to MySQL 5.1 Server

Perform the following steps to migrate the data from MySQL 5.0 Server to MySQL 5.1 Server:

1. Stop the MySQL 5.1 Server (If running).
2. Start the MySQL 5.0 Server.

3. Create a backup directory to store the SQL dump files.
4. Execute the `mysqldump` script to obtain the dump of all the databases. For instance, execute the `mysqldump --all-databases -p > dump.sql` command.
5. Stop the MySQL 5.0 Server.
6. Start the MySQL 5.1 Server.
7. Restore the SQL dump using the `mysql -p < dump.sql` command.
8. Run the `mysql_upgrade` script.

In OpenSolaris 2009.06 OS, execute the following commands to upgrade the MySQL Server 5.0 to MySQL Server 5.1 assuming that you have installed both the DB versions from the IPS repository and their respective directories are under `/usr/mysql/5.0` and `/usr/mysql/5.1`.

```
# svcadm disable mysql:version_51
# svcadm enable mysql:version_50
# mkdir /mydbbackup
# cd /mydbbackup
# /usr/mysql/5.0/bin/mysqldump --all-databases -p > mydbdump.sql
# svcadm disable mysql:version_50
# svcadm enable mysql:version_51
# /usr/mysql/5.1/bin/mysql -p < mydbdump.sql
# /usr/mysql/5.1/bin/mysql_upgrade
```

Squid Proxy Server

The Squid Proxy Server is both a proxy server and a web cache daemon available through the Web Stack software. The Squid Proxy Server is primarily used to speed up a web server by caching repeated requests.

The Squid Proxy Server can be used in 2 ways:

1. The Squid Proxy Server can act in **Normal Mode** where it can cache the contents of an unlimited number of web servers for a limited number of clients.
2. The Squid Proxy Server can act in **Load Balancer Mode** (web server acceleration) where the cache serves an unlimited number of clients for a limited number of servers.

For instance, if your site is becoming slower because of heavy load there are several things you can do without buying additional hardware:

1. Configure your application to use `memcached`.
2. Configure your web server with a caching server like the Squid Proxy Server.

An example Squid-Apache Web Server setup could be:

1. Run the **Apache Web Server** on port **8080**

2. Run the **Squid** Proxy Server on port **80**
3. When a request from a client reaches port 80, the Squid Proxy Server will first check if it has the result stored. If so, Apache server will never be contacted to process the request thus reducing the server load.

To Configure the Squid Proxy Server

The following example shows how you can configure the Squid Proxy Server for a site `www.foo.com` on a web server.

For OpenSolaris 2009.06, the squid directory is `/usr/squid`

For Red Hat Enterprise Linux, the squid directory is `/opt/sun/webstack/squid`

For Solaris 10 OS, the squid directory is `/opt/webstack/squid`

Open the `squid.conf` file available under the squid directory, in a text editor. Make the following changes to specify the port for the web site, and to specify the IP address for the web server. By default Squid Proxy Server listens at port 3128. Also, specify access controls to allow access to your web site.

```
http_port 80 accel defaultsite=www.foo.co
cache_peer ip.of.server1 parent 80 0 no-query
acl sites_server_1 dstdomain www.foo.com
http_access allow sites_server_1
```

In OpenSolaris 2009.11 OS, to start the Squid Proxy Server, execute the following command:

```
# svcadm enable squid
```

In Solaris 10 OS, to start the Squid Proxy Server, execute the following command:

```
# svcadm enable sun-squid
```

In Red Hat Enterprise Linux, to start the Squid Proxy Server, execute the following command:

```
# /sbin/service sun-squid start
```

You need to start the Squid as a privileged user or use the `sudo` prefix.

You can also start the Squid Proxy Server in Red Hat Enterprise Linux as follows:

```
# /etc/init.d/sun-squid start
```

Note – For Red Hat Enterprise Linux, see the logs at /var/opt/sun/webstack/squid/logs (especially in the cache.log file) to verify if there are any errors in configuring the Squid Proxy Server.

Note – For Solaris 10 IS, see the logs at /var/opt/sun/webstack/squid/logs (especially in the cache.log file) to verify if there are any errors in configuring the Squid Proxy Server.

Using SMF to Manage Services

In OpenSolaris 2009.06, you can start the Squid Proxy Server by executing the following command:

```
# svcadm enable -s squid
```

In Solaris 10 OS, you can start the Squid Proxy Server by executing the following command:

```
# svcadm enable -s sun-squid
```

Not applicable for Red Hat Enterprise Linux

Sun GlassFish Enterprise Server

Sun GlassFish Enterprise Server (GlassFish Server) is a Java EE platform-compatible server for the development and deployment of Java EE applications and Java Web Services. GlassFish Server is free for development, deployment, and redistribution.

Sun GlassFish Web Stack software includes the GlassFish Server, which allows you do deploy and monitor your web applications.

Installing the GlassFish Enterprise Server

In OpenSolaris OS, you can install the GlassFish Enterprise Server by executing the following command:

```
# pkg install glassfish-2
```

In Red Hat Enterprise Linux and Solaris 10 OS, you can install the GlassFish Enterprise Server by running the installer available through the Web Stack software. Read this document for more information.

- You can download and read the GlassFish Enterprise Server documentation from this site.

Apache Web Server as a Load Balancer for the GlassFish Server (OpenSolaris OS)

Not applicable for Solaris OS and Red Hat Enterprise Linux.

The GlassFish Server has had `mod_jk` support since the first release; a common use case is where the Apache Web Server is serving static pages and delegating dynamic content to the GlassFish Server.

The GlassFish Server also supports the case where the Apache Web Server works as a load balancer to a cluster of the GlassFish Server instances.

The following steps shows you how you can make the Apache Web Server work as a load balancer along with the GlassFish Server:

- **Set up the Web Stack and the GlassFish Server**.

Set up the Web Stack software (if not setup already).

```
# pkg install amp
```

Set up the GlassFish Server.

```
# pkg install glassfishv2
```

The GlassFish Server is installed in the `/usr/appserver` directory.

-
    - Configure the GlassFish Server to use the Cluster Profile. See this blog entry for more information.
- Configure SSL for the Apache Web Server.

```
$ /usr/bin/openssl req -new -x509 -nodes -out
server.crt -keyout server.key
```

The 2 files 1. `server.crt` and 2. `server.key` are created.

```
# cp /etc/apache2/2.2/samples-conf.d/ssl.conf
/etc/apache2/2.2/conf.d/ssl.conf
```

Modify the `ssl.conf` file in order to change the values of the following parameters:

```
...
SSLCertificateFile "<path to server.crt>"
```

```
SSLCertificateKeyFile "<path to server.key>"
...
```

Restart the Apace Web Server

```
# svcadm restart apache22
```

- Download the GlassFish Load Balancer plug-in for the OpenSolaris OS from this site.
- Extract the file `aslb-9.1.1-b8.jar` to get the 2 files 1. `SUNWaslb.zip` and 2. `SUNWaspx.zip`.
- Create a Temp. dir. structure.

```
$ mkdir <TMP_PREFIX>/glassfish/lib/plu
in
```

where TMPP_PREFIX can be your home directory.

Extract the content of both the zip files `SUNWaslb.zip` and `SUNWaspx.zip` into the <TMPPREFIX>/glassfish/lib/plugin dir.

After extracting the content the <TMP_PREFIX>/glassfish/lib/plugin directory should look like:

```
+lib
----dtds
----install
----webserver-plugin
----lib*.so
```

Set the neccessary permission.

```
$ chmod -R 755 <TMP_PREFIX>/glassfish/
ib/plugin/lib
```

- Create the required directories in the Apache Web Server directory. Create the modules, resource, and the errorpages directory.

```
# mkdir /var/apache2/2.2/modules
# mkdir /var/apache2/2.2/modules/resource
# mkdir /var/apache2/2.2/modules/errorpages
```

- Populate the resource directory.

```
# cp <TMP_PREFIX>/glassfish/lib/plugin/lib/webserver-plugin/solaris/apache2.2/LBPluginDefault_root.res
/var/apache2/2.2/modules/resource
# cp <TMP_PREFIX>/glassfish/lib/plugin/lib/webserver-plugin/solaris/apache2.2/LBPlugin_root.res /var/apache2/2.2/modul
```

- Populate the modules directory.

```
# cp <TMP_PREFIX>/glassfish/lib/plugin/lib/webserver-plugin/solaris/apache2.2/mod_loadbalancer.so
/var/apache2/2.2/modules
```

- Populate the errorpages directory.

```
# cp <TMP_PREFIX>/glassfish/lib/plugin/lib/webserver-plugin/solaris/apache2.2/errorpages/*
/var/apache2/2.2/modules/errorpages
```

- Create and populate the configuration directory.

```
# mkdir /var/apache2/2.2/conf
$ cd /var/apache2/2.2/conf
# cp <TMP_PREFIX>/glassfish/lib/plugin/lib/install/templates/loadbalancer.xml.example /var/apache2/2.2/conf
# mv <TMP_PREFIX>/glassfish/lib/plugin/lib/install/templates/loadbalancer.xml.example <TMP_PREFIX>/glassfish/lib/
# cp <TMP_PREFIX>/glassfish/lib/plugin/lib/dtds/sun* /var/apache2/2.2/conf
```

- Create and populate the Certificate DB directory.

```
# mkdir /var/apache2/2.2/sec_db_files
# cp <TMP_PREFIX>/glassfish/lib/plugin/lib/webserver-plugin/solaris/apache2.2/*.db /var/apache2/2.2/sec_db_files
```

- Modify the `httpd.conf` file. Add the following lines to the `/etc/apache2/2.2/httpd.conf` file:

```
LoadModule apachelbplugin_module /var/apa
he2/2.2/modules/mod_loadbalancer.so
<IfModule mod_apache2lbplugin.cpp>
config-file "/var/apache2/2.2/conf/loadbalancer.xml"
locale en
</IfModule>
<VirtualHost 10.12.8.107>
DocumentRoot "/var/apache2/2.2"
ServerName vm07
</VirtualHost>
```

- Restart the Apache Web Server

```
# svcadm restart apache22
```

- Start the GlassFish Server domain

```
# svcadm enable domain1
```

- Deploy web applications in the GlassFish Server.

NOTE: Check if the previously mentioned procedure can be documented since the Web Stack-GlassFish issues are yet to be resolved

PHP Environment

In OpenSolaris 2009.06, if you have installed the optional package amp-dev, you don't need to download PHP runtime libraries separately. But if you have not installed the amp-dev package, you can install the PHP libraries by executing the following command:

```
# pkg install SUNWphp52 SUNWphp52-mysql SUNWphp52-pgsql
SUNWapch22m-php52
```

In Red Hat Enterprise Linux and Solaris 10 OS, if you have installed the Web Stack software through the most common amp configuration option, you don't need to install the PHP runtime libraries separately. But if you have not installed the complete amp stack, you can install the PHP libraries by providing the php option during installation. For more information, see the Web Stack Installation and Configuration Guide

For instance,

```
$ ./install php
```

Where is PHP Installed?

In OpenSolaris 2009.06, when you download and install the PHP packages mentioned in the preceding section, the binaries will be available from /usr/php/5.2 and other runtime configuration files will be available under the /etc/php/5.2 directory.

Note – Available Version - PHP 5.2.9.

In Solaris 10 OS, when you install PHP through the Web Stack software, the PHP configuration and runtime files will be available under the locations provided in the following table:

Platform	Install Location
Solaris 10 OS	1. Installation Location - /opt/webstack/php/5.2/ 2. Runtime files - /var/opt/webstack/php/5.2/ 3. Configuration files - /etc/opt/webstack/php/5.2

In Red Hat Enterprise Linux, when you install PHP through the Web Stack software, the PHP configuration and runtime files will be available under the locations provided in the following table:

Platform	Install Location
Red Hat Enterprise Linux	1. Installation Location - `/opt/sun/webstack/php/` 2. Runtime files - `/var/opt/sun/webstack/php/5.2/` 3. Configuration files - `/etc/opt/sun/webstack/php/5.2/`

Available PHP Extensions

The following is the list of PHP extensions available through the Web Stack:

```
APC, GD, LDAP, OpenSSL, PostgresSQL, XDebug
, Bzip2,
Gettext, Mcrypt, PDO, SQLite, Zlib, Curl, Iconv,
Memcache, PDO MySQL, Suhosin, DTrace (Solaris 10 OS and OpenSolaris OS),
IDN, Mysqli, PDO PostgreSQL, TCP Wrap, FTP, IMAP, MySQL,
PDO SQLite, and Tidy.
```

In OpenSolaris 2009.06, if you have installed the `amp-dev` package, some PHP modules are available as extensions. Each of the modules has a respective INI file under the `/etc/php/5.2/conf.d` directory. These PHP extensions are enabled by default. Some custom third-party PHP extensions are not enabled by default. However, you can edit the PHP extension specific INI file for any specific configuration changes.

Note – Not all co-packaged extensions are mentioned in the list provided. See the `/etc/php/5.2/conf.d` directory for the available extensions.

In Red Hat Enterprise Linux and Solaris 10 OS, if you have installed the Web Stack software using the `amp` installation option, some PHP modules are made available as extensions. Each of the modules has a respective INI file under the PHP `conf.d` directory.

For instance, INI files for Red Hat Enterprise Linux is available under `/etc/opt/sun/webstack/php/5.2/conf.d` directory.

For instance, INI files for Solaris 10 OS is available under `/etc/opt/webstack/php/5.2/conf.d` directory.

Enabling/Disabling Bundled PHP Extensions

You can enable or disable a PHP extension by commenting out a line in the `<extension>.ini` available under the PHP `/etc/php/5.2/conf.d` directory.

In OpenSolaris 2009.06, the PHP `conf.d` directory is available under the `/etc/php/5.2` directory.

In Solaris 10 OS, the PHP `conf.d` directory is available under the `/etc/opt/webstack/php/5.2/` directory.

In Red Hat Enterprise Linux, the PHP `conf.d` directory is available under the `/etc/opt/sun/webstack/php/5.2/` directory.

For example, to enable the Suhosin PHP extension, perform the following tasks:

1. Open the `conf.d/suhosin.ini` file in a text editor.
2. Un-comment the line enabling the Suhosin extension

```
extension=suhosin.so
```

To disable the extension, insert a comment character `;` at the beginning of the line.

Note – You must restart the web server for changes to take effect.

PHP Configuration through AMP GUI

Not applicable for Red Hat Enterprise Linux and Solaris 10 OS.

You can use the AMP GUI to modify PHP settings as supported by Apache Web Server. To invoke the GUI tool, click Applications > Developer Tools > WebStack Admin > Options.

The PHP Debugger is not enabled by default. You need to select the PHP Debugger check box to enable this option. The NetBeans IDE supports the use of `XDebug` for debugging PHP projects.

Debugging Support for PHP Applications

1. From the Launch menu of the Solaris operating system, select Applications > Developer Tools > Web Stack Admin > Options. The Options screen is displayed.
2. Select the PHP tab.
3. Select the Debug check box. By selecting the check box you are enabling the automatic debug option for all PHP applications.
4. You are prompted to restart the server.
5. Click Restart for the change to take effect.

Optionally, you can enable debugging by adding the following lines in the `xdebug.ini` file:

```
zend_extension=/usr/php/5.2/modules/xdebu
.so
xdebug.remote_enable=1
xdebug.remote_handler=dbgp
xdebug.remote_mode=req
xdebug.remote_host=localhost
```

where:

In OpenSolaris OS, the xdebug.ini file is available under the /etc/php/5.2/conf.d directory.

In Solaris 10 OS, the xdebug.ini file is available under the /etc/opt/webstack/php/5.2/ directory.

In Red Hat Enterprise Linux, the xdebug.ini file is available under the /etc/opt/sun/webstack/php/5.2/ directory.

Python Environment

Python is an object oriented scripting language and is available through the Web Stack software on all supported platforms. Python scripts can be invoked through the Apache Web Server using mod_python. There are various web frameworks available for python including Django, Pylons, TurboGears, and Zope that are widely used by the community.

In OpenSolaris 2009.06, Python is co-packaged. If SUNWpython is not already available, install the package through the IPS package manager.

Note – Python is not provided through the Web Stack software for Red Hat Enterprise Linux. You need to use the Python interpreter provided by the platform. Also mod_python is not part of the Web Stack software.

You can download mod_python from http://httpd.apache.org/modules/python-download.cgi.

In Solaris 10 OS, you can install Python by providing the python option during installation. For more information, see the Web Stack Installation and Configuration Guide.

For instance,

```
$ ./install python
```

When you install Python from the Web Stack software, the Python runtime files will be available under the locations provided in the following table:

Platform	Install Location
Solaris 10 OS	Installation Location - `/opt/webstack/python/`. Runtime files - `/var/opt/webstack/python`

Note – Python 2.5.2 has DTrace probes integrated. You can trace your Python-based applications for performance bottlenecks and other problems using the Solaris DTrace feature.

Ruby Environment

Ruby is available through the Web Stack software only for Solaris 10 OS and OpenSolaris OS. The Web Stack software does not include the Ruby interpretor on the Red Hat platform; use the OS-supplied Ruby instead.

In Solaris 10 OS, you can install the Ruby runtime by providing the `ruby` option during installation. For more information, see the Web Stack Installation and Configuration Guide

For instance,

```
$ ./install ruby
```

When you install Ruby from the Web Stack software, the Ruby configuration and runtime files will be available under the locations provided in the following table:

Platform	Install Location
Solaris 10 OS	Installation Location - `/opt/webstack/ruby/1.8`. Runtime files - `/var/opt/webstack/ruby/1.8`

In OpenSolaris 2009.06, to download and install Ruby packages, execute the following command:

```
# pkg install ruby-dev
```

When you download and install the `ruby-dev` package, you are also installing the NetBeans plug-in for Ruby. If you are not planning to use NetBeans IDE for your Ruby development, you can get only the ruby package as show below:

```
# pkg install SUNWruby18
```

When you download and install the Ruby package mentioned in the above section, the binaries will be available from `/usr/ruby/1.8`.

Ruby Gems

Ruby Gems is a system for managing Ruby software libraries. Ruby code packaged in this manner is called a gem. Gems enable downloading, installing, and managing your Ruby libraries. The Web Stack package in Solaris does not include a lot of packaged gems because you can use Ruby Gems to download and install additional gems. You can use the the NetBeans interface for managing Gems.

/var/ruby/1.8/gem_home contains the Ruby gems repository. GEM_HOME is /var/ruby/1.8/gem_home. /usr/ruby/1.8/bin contains the Ruby executable as well as other utility programs.

If you are running gem install and have no write access to the GEM_HOME path, the gem will be installed to $HOME/.gem. You can also add the GEM_HOME/bin directory to the PATH to easily access commands installed with gems (such as the rails command).

In Solaris 10 OS, /var/opt/webstack/ruby/1.8/gem_home contains the Ruby gems repository. Configure the GEM_HOME environment variable to point to /var/opt/webstack/ruby/1.8/gem_home to use this installation of Ruby gems. /var/opt/webstack/ruby/1.8/bin contains the Ruby executable as well as other utility programs. These programs are linked from /usr/bin.

In OpenSolaris 2009.11 OS, the installation of Ruby native extensions requires the SUNWgcc package. gem is preconfigured to use gcc for C language extensions. Set the CXX environment variable to:

```
/usr/sfw/bin/g++
```

when installing C++ language extensions, such as eventmachine, and thin.

For example:

```
$ CXX=/usr/sfw/bin/g++ gem install eventm
chine
```

When installing the mysql gem, the paths to the MySQL 5.1 client libraries and header files must be provided, as follows:

```
# gem install mysql --with-mysql-include=/usr/mysql/5.1/include
--with-mysql-lib=/usr/mysql/5.1/lib
```

Note – To use gem, make sure you have direct access to the Internet. If your system is behind a firewall or if it uses a proxy server, set the HTTP_PROXY variable.

The directory referenced by GEM_HOME should be writable. So make sure you execute the following command:

```
$ chmod -R a+w <GEM_HOME_PATH>
```

Caution - Do not perform the previously mentioned tasks, if `GEM_HOME` refers to `$HOME/.gem`.

Ruby on Rails (RoR)

Rails is a full-stack framework for developing database-backed web applications. Rails provide a pure-Ruby development environment. Ruby gems is the standard Ruby package manager. If you have successfully installed Ruby through the Web Stack software, Ruby gems are already loaded. Hence you can install Rails and its dependencies by executing the following command:

```
/usr/bin/gem install rails
```

You can also install rails directly from the NetBeans IDE rather than invoking the gem command through a command line. The Ruby on Rails environment is now ready for development. You can now use the NetBeans IDE to create a Ruby on Rails project. When you create a project, the IDE creates the same folders and files that a rails command would create.

Tomcat Servlet Container

The Web Stack software comes with the Tomcat 6.0 Servlet container for deploying JSP and Servlet based web applications. You can install the Tomcat Server by providing the `tomcat` option during installation. For more information, see the Web Stack Installation and Configuration Guide.

In OpenSolaris 2009.06*, Tomcat is available under `/usr/tomcat6`.

To start the Tomcat Server in **OpenSolaris 2009.06 OS**, perform the following task:

1. Execute the following command:

```
# svcadm enable -s tomcat6
```

To stop Tomcat, perform the following tasks:

```
# svcadm disable tomcat6
```

In Solaris 10 OS*, Tomcat is available under `/opt/webstack/tomcat/6.0`.

To start the Tomcat Server in **Solaris 10 OS**, perform the following task:

```
# svcadm enable -s sun-tomcat6
```

To stop the Tomcat Server, perform the following task:

```
# svcadm disable sun-tomcat6
```

In Red Hat Enterprise Linux*, Tomcat is available under /opt/sun/webstack/tomcat/6.0.

To start the Tomcat Server in **Red Hat Enterprise Linux**, perform the following task:

```
# /sbin/service sun-tomcat6 start
```

To stop the Tomcat Server, perform the following task:

```
# /sbin/service sun-tomcat6 stop
```

In Red Hat Enterprise Linux, you need to set the JAVA_HOME variable or modify the /opt/sun/webstack/tomcat/6.0/bin/setenv.sh file to point to the correct Java SE installation path.

Distributed Caching with memcached

memcached is a high-performance, distributed caching system most widely used by popular web sites. The main benefit of using memcached is to speed up dynamic Web applications by reducing database load. memcached is used on LiveJournal, Slashdot, Wikipedia, and other high-traffic sites.

In OpenSolaris 2009.11 and Solaris 10 OS, perform the following steps as a root user to start memcached:

```
# svcadm enable -s memcached
```

Execute the disable command to stop the server.

```
svcadm disable memcached
```

In Red Hat Enterprise Linux, as the root user you need to create the file /etc/opt/sun/webstack/sysconfig/memcached. You can modify the following default values for the parameters already available in the file:

```
PORT=11211
USER=memcached
MAXCONN=1024
CACHESIZE=64
OPTIONS=""
```

Starting the memcached server

This section describes how to start memcached.

- Log in as non-root user.

Note – You cannot execute memcached as a root user.

- Start the memcached server.

In OpenSolaris 2009.06, you can start the memcached server by executing the following command:

```
# svcadm enable -s memcached
```

In Solaris 10 OS, you can start the memcached server by executing the following command:

```
# svcadm enable -s sun-memcached
```

If you need to pass some options to memcached, you can use setprop option as shown below:

In Solaris 10 OS, execute the following command:

```
# svccfg -s sun-memcached setprop memcached/options=("-u"
"nobody" "-m" "512")
# svcadm refresh sun-memcached
# svcadm enable -s sun-memcached
```

In OpenSolaris 2009.06, execute the following command:

```
# svccfg -s memcached setprop memcached/options=("-u"
"nobody" "-m" "512")
# svcadm refresh memcached
# svcadm enable -s memcached
```

The previously mentioned command will start the server as user nobody with 512 MB as memory.

For information about the memcached command and its options, see the memcached man page.

In Red Hat Enterprise Linux, to the memcached server, execute the following command:

```
# /sbin/service sun-memcached start
```

Using memcached with Solaris Zones

Read this document .

Not applicable for Red Hat Enterprise Linux.

Automatic Startup of Servers

In OpenSolaris 2009.06, if you have started the servers through the Web stack menu, the servers will automatically start when the machine restarts.

In Solaris 10 OS, the Apache Web Server automatically starts when the machine restarts if you enable the `sun-apache22` service.

In Red Hat Enterprise Linux, startup script `sun-apache22` is available under `/etc/init.d`. To automatically start the service when the machine starts, use the `chkconfig` mechanism.

To start the Apache Web Server in Red Hat Enterprise Linux when the machine starts, execute the following command:

```
# /sbin/chkconfig sun-apache22 on
```

To start the MySQL DB in Red Hat Enterprise Linux when the machine starts, execute the following command:

```
# /sbin/chkconfig sun-mysql51 on
```

To start the MySQL DB in Red Hat Enterprise Linux when the machine starts, execute the following command:

```
# /sbin/chkconfig sun-mysql51 on
```

Note – `{}chkconfig` command only ensures that the start-on-boot feature is enabled in Red Hat Enterprise Linux. However, to start/stop the Apache Web Server or MySQL Server, you must use the `service` command in Red Hat Enterprise Linux.

For instance,

```
# /sbin/service sun-apache22 start
```

NetBeans IDE as a Development Environment

This section shows you how to use the NetBeans IDE to build web applications using the Web Stack components available through the Web Stack software. First let us enumerate the stages involved in developing an end to end enterprise grade web application. In order to use the IDE for development, you will need the following:

- **Server support** - The IDE should support managing the Server life cycle through an intuitive, and easy-to-use interface.
- **Languages Support** - The IDE should support working with the programming language or web application framework that your web application will be eventually built on.

- **Database support** - The IDE should be well integrated with the development, testing, and production database.

Typically, your development environment should enable you to build web applications with minimal effort, offering you extensive support for stack level configuration through property editors, wizards, and widgets. Server support, languages support, and database support are the three prime facets that any IDE can offer for web development.

The NetBeans IDE takes care of all these factors with an elegant interface thereby making web development easy. You can build your web applications in PHP or use Rails framework and work with databases like MySQL or PostgreSQL in the NetBeans IDE.

Note – If you are already using the NetBeans IDE, you can use the Web Stack components along with NetBeans IDE by following some steps. For instance, you can use **Add Server** option of NetBeans IDE to add the Apache Tomcat Server 6.0 provided through the Web Stack software.

You can also use the NetBeans IDE for Java/PHP/Ruby development. The NetBeans IDE supports integrated configuration windows for managing web servers and databases directly from the IDE. Note that the version of NetBeans available in the official pkg repository may be of an older version than the one available through the NetBeans web site. The following sections show how to configure the NetBeans IDE 6.5.1 for your development needs.

The NetBeans IDE supports various Java EE servers and other web servers for application deployment. You can have multiple servers configured in NetBeans irrespective of the server that will be used in your projects. Thus you can have the Apache 2 Web Server configured for deploying PHP files, the Apache Tomcat container configured for deploying Java based web applications or a WEBrick Server for deploying Rails applications.

Getting Started with the NetBeans IDE and the Web Stack Software

Not applicable for Red Hat Enterprise Linux and Solaris 10 OS.

- Using the NetBeans IDE with the Web Stack Software
- MySQL DB Integration
- Server Integration
- Languages and Application Support

Migrating From Development to Production Environment

It is not recommended to run mission-critical web sites without customization and tuning for performance, scalability, and security. When you have developed and debugged a working application in this environment, it is recommended to transfer the database data and PHP files to a production-quality OpenSolaris installation, running on high-availability Sun hardware that has been carefully tuned to your mission-critical needs.

File Layout of Web Stack Components in the OpenSolaris 2009.06 OS

Read this document .

Deploying Common Open Source Applications

Read this document .

DTrace Support for Web Stack

Not applicable for Red Hat Enterprise Linux.

Read this document .

Troubleshooting Web Stack

Read this document .