Web Stack Getting Started Guide

Sun Microsystems, Inc.

Part No: 820–3883–10
May 2008

080516@20209

Contents

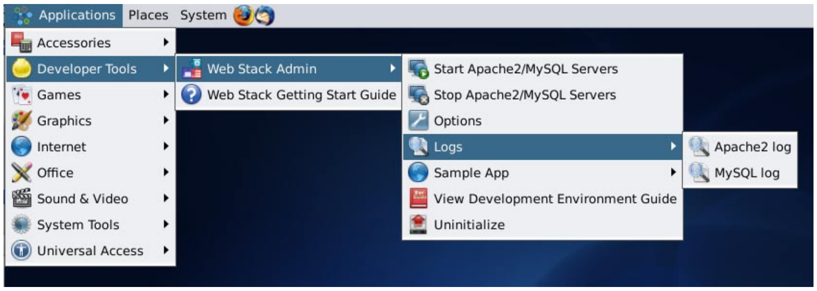
CHAPTER 1

About Web Stack

The OpenSolaris 2008.05 OS includes a set of popular web technologies that enables developers to easily build and deploy web applications. This guide describes the supported web technologies and how to use them on OpenSolaris 2008.05 OS. OpenSolaris also integrates the NetBeans™ IDE 6.0 for easy development and deployment of web applications.

About the User Interface

OpenSolaris OS provides an interface for using web stack components. The following figure shows the Web Stack Admin menu options on the screen.



The Launch menu provides menu options to start or stop the servers, configure the options for Apache, MySQL, and PHP. For more information, see "Customizing Web Stack Components" on page 17.

Web Stack Components

The following components are installed and optimized for use on the OpenSolaris 2008.05 platform:

Apache HTTPd Server 2.2.8	Apache is the Web Server that serves web pages in response to requests from users' web browsers.
MySQL 5.0.45	MySQL is a relational database engine used to store most dynamic data. Open Solaris OS includes 32bit architecture of MySQL database. MySQL supports the following standard APIs: ■ SQL92 ■ SQL99 ■ SQL2003 MySQL also has its own C client API which is delivered with the server.
PHP 5.2.4	PHP is an interpreted dynamic web page scripting language. A PHP language interpreter is integrated into the Apache Web Server. PHP module is integrated with MySQL and PostgreSQL that provides database support for Apache HTTP (32 and 64-bit), and Pre-fork MPM. This module is also integrated with the NetBeans IDE.

About PHP Extensions

PHP modules are integrated as extensions. Each of the modules have a respective INI file under `/etc/php5/5.2.4/conf.d` directory. These PHP extensions are enabled by default.

Custom third-party PHP extensions like `APC`, `Suhosin`, `IDN`, `Tcpwrap`, `XDebug`, `DTrace` are not enabled by default. However, you can edit the PHP extension specific INI file for any specific configuration changes.

Following is the list of PHP extensions available on the Open Solaris OS :

bcmath	exif	mbstring	pdo_sqlite	sysvsem	xmlrpc
bz2	ftp	mysql	pgsql	sysvshm	xmlwriter

calendar	gd	mysqli	posix	tcpwrap	xsl
ctype	gettext	openssl	shmop	tcpwrap	zip
curl	hash	pcntl	soap	tidy	zlib
dba	iconv	pdo	sockets	tokenizer	
dbase	n.so	pdo_mysql	sqlite	wddx	
dom	ldap	pdo_pgsql	sysvmsg	xmlreader	

PostgreSQL 8.2	PostgreSQL is an object-relational database management system (ORDBMS). PostgreSQL uses a client/server model. A PostgreSQL session consists of a server process, which manages the database files, accepts connections to the database from client applications, and performs actions on the database on behalf of the clients. The database server program is called `postgres` and the user's client application that wants to perform database operations.
Ruby on Rails	Ruby on Rails is the framework for web application development. Ruby is an object-oriented programming language. Rails is an open source Ruby framework for developing web-based, database-driven applications. Extensions to the Ruby language, and a packaging program called `Rubygems` are included to enable developers to install to Rails and to add functionality in a seamless manner. Ruby is bundled with extensions, similar to PHP, and they are `OpenSSL`, `curses`, `Tcl/Tk`, and `readline`.
Squid 2.6.STABLE16[1]	Squid is a fully-featured HTTP/1.0 proxy server. Squid offers a rich access control, authorization and logging environment to develop web proxy and content serving applications.
memcached 1.2.2	`memcached` is a high-performance, distributed memory object caching system. Helps in speeding up dynamic web applications by alleviating database load. The OpenSolaris 2008.05 includes 32 bit versions of the memcached daemon. For more information about APIs and how to configure, enable and disable the server, see `memcached` man pages. For more information on how to use NetBeans on OpenSolaris 2008.05, see Using NetBeans IDE for Development

◆ ◆ ◆ CHAPTER 2

Getting Started

This chapter describes how to use the Web Stack developer tools for developing and deploying web applications.

Installing Web Stack Components

OpenSolaris 2008.05 release allows you to install Web Stack components individually. The following table lists the packages and command to install them.

Note – Ensure to import SMF manifest file before setting up an AMP cluster or installing as an individual package. For more information see "Administering Apache Server" on page 11, "Administering MySQL Server" on page 11, and "Administering memcached" on page 11.

Note – The `pfexec` command allows the `non-root` user to install packages.

TABLE 2–1 Installing the Web Stack components

Component Name	Package Name	Command to install the component
Apache 2.2 core only	`SUNWapch22`,`SUNWapch22d`	Use the following command to install Apache core: `pfexec pkg install SUNWapch22 SUNWapch22d`
Apache 2.2 modules	`SUNWapch22m-security`, `SUNWapch22m-jk`, `SUNWapch22m-fcgid`, `SUNWapch22m-dtrace`	Use the following command to install Apache modules: `pfexec pkg install SUNWapch22m-security SUNWapch22m-jk SUNWapch22m-fcgid SUNWapch22m-dtrace`

TABLE 2–1 Installing the Web Stack components *(Continued)*

Component Name	Package Name	Command to install the component
PHP5	`SUNWphp524`, `SUNWphp524core`, `SUNWphp524man`, `SUNWphp524-mysql`, `SUNWphp524-pgsql`, `SUNWapch22m-php5`	Use the following command to install PHP5: `pfexec pkg install SUNWphp524 SUNWphp524core SUNWphp524man SUNWphp524-mysql SUNWphp524-pgsql SUNWapch22m-php5`
MySQL	`SUNWmysql5`	Use the following command to install MySQL: `pfexec pkg install SUNWmysql5` For more information on how to start the MySQL, see "Setting Up MySQL" on page 14
PostgreSQL	`SUNWpostgr-82-devel`	Use the following command to install PostgreSQL: `pfexec pkg install SUNWpostgr-82-devel`
Squid	`SUNWsquid`	Use the following command to install Squid: `pfexec pkg install SUNWsquid`
Web Stack user interface	`webstackui`	This package adds menu to operate Apache, PHP, MySQl and other components in the stack. Use the following command to install Web Stack user interface: `pfexec pkg install webstackui`

You can install the components using the PackageManager graphical user interface for IPS. To install, update, and manage packages using the PackageManager, see the screencast.

Installing AMP Cluster

The AMP cluster package is a super set of all individual AMP packages. For more information on how to install an AMP cluster, see Setting Up Your AMP Development Environment.

Note – Ensure to import SMF manifest file before setting up an AMP cluster. For more information on how to import the manifest, see "Administering Apache Server" on page 11, "Administering MySQL Server" on page 11, and "Administering memcached" on page 11.

Administering Apache Server

When you install the Apache Web Server package, the binaries are installed in the `/usr/apache2/2.2` and run time configuration files will be available in the following location `/etc/apache2/2.2`.

SMF Framework for Web Stack components are not initialized in the this release. Use either the Web Stack UI or start the server by running the startup command manually. Perform the following steps as `root` user to start Apache Web Server:

```
svccfg import /var/svc/manifest/network/http-apache22.xml
svcadm enable network/http:apache22
```

The `import` command allows you to import the apache manifest file and the enable command allows you to start the server.

Execute the `disable` command to stop the server.

```
svcadm disable network/http:apache22
```

Administering MySQL Server

SMF Framework for Web Stack components are not initialized in the this release. Use either the Web Stack UI or start the server by running the startup command manually. Perform the following steps as `root` user to start MySQL Server:

Perform the following steps as `root` user to start MySQL server.

```
svccfg import /var/svc/manifest/application/database/mysql.xml
svcadm enable application/database/mysql:version_50
```

The `import` command allows you to import the MySQL manifest file and the enable command allows you to start the server.

Execute the `disable` command to stop the server.

```
svcadm disable application/database/mysql:version_50
```

Administering memcached

SMF Framework for Web Stack components are not initialized in the this release. Use either the Web Stack UI or start the server by running the startup command manually. Perform the following steps as `root` user to start memcached:

Perform the following steps as `root` user to start memcached.

```
svccfg import /var/svc/manifest/application/database/memcached.xml
svcadm enable application/database/memcached:default
```

The `import` command allows you to import the memcached manifest file and the enable command allows you to start the server.

Execute the `disable` command to stop the server.

```
svcadm disable application/database/memcached:default
```

Initializing the Development Environment

Before you can use the Web application development environment, you must initialize the environment for your login. Every new user logged into the system must initialize their own environment.

Note – To use the Web Stack Admin user interface options, you must login to the GNOME desktop.

To initialize your Web application development environment, click on the Launch menu, and select All Applications > Developer Tools > Web Stack Admin > Initialize.

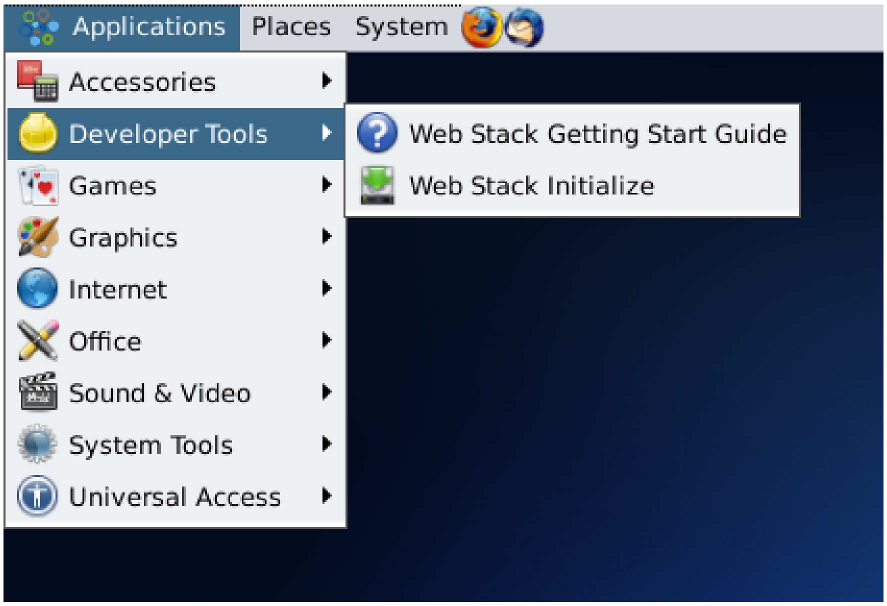


FIGURE 2–1 Initializing the development environment

Note – Changes to the Web Stack Admin options affects all users on the system.

For more information about customizing the Web Stack Admin components, see "Customizing Web Stack Components" on page 17.

Starting and Stopping Apache HTTPd and MySQL Servers

Note – Before you start the components, you must initialize the development environment for your login account.

To start or stop the servers on your development environment, click on the Launch menu, and select All Applications > Developer Tools > Web Stack Admin > Start Apache2/MySQL Servers or Stop Apache2/MySQL Severs.

This action starts the MySQL database and start the Apache 2 server.

You can configure the servers to be started automatically at system start up by customizing the start up options. For more information see "Customizing Web Stack Components" on page 17.

Setting Up MySQL

The procedure to register MySQL service with SMF and to start the database server instance is described below.

▼ To Start the Database Server Instance

1 **Check the status of the SMF service.**

The `mysql;version_50` service is disabled by default.

```
example% svcs mysql
STATE         STIME     FMRI
disabled      14:27:09  svc:/application/database/mysql:version_50
```

2 **Enable the** `mysql:version_50` **service.**

```
example%svcadm enable mysql:version_50
```

3 **Check the status of the service.**

```
example% svcs mysql
STATE          STIME    FMRI
online         14:30:08 svc:/application/database/mysql:version_50
```

The state of the service is `online`, and the database server is started successfully.

Note – To shutdown the database instance and prevent automatic restart, disable the SMF service.

```
example% svcadm disable mysql:version_50
```

Setting Up PostgreSQL

This section describes how to set up the PostgreSQL database with NetBeans 6.0 IDE.

▼ To Set Up PostgreSQL

1 **Create a user by name** `postgres`**.**

```
bash-3.2$ su
Password:
# chown postgres:postgres /var/postgres/8.2/data
# su - postgres
```

2 **Initialize the database.**

```
bash-3.2$/usr/postgres/8.2/bin/initdb -D /var/postgres/8.2/data
```

The files belonging to this database system are owned by user `postgres`. This user must also own the server process. The database cluster is initialized with locale C.

3 **Start the database server.**

```
bash-3.2$/usr/postgres/8.2/bin/postgres -D /var/postgres/8.2/data
```

or

```
bash 3.2$/usr/postgres/8.2/bin/pg_ctl -D /var/postgres/8.2/data -l logfile start
```

4 **Enable PostgreSQL version 8.2.**

```
bash-3.2$ su
Password:
# /usr/sbin/svcadm enable postgresql:version_82
```

5 **Configure the PostgreSQL database with NetBeans IDE 6.0.**

a. **Create a New Database Connection.**

In the New Connection Database dialog box, enter the following details:

- `Name – PostgreSQL`
- `Driver – org.postgresql.Driver`
- `Database URL – jdbc.postgresql://localhost.5432/postgres`
- `User – postgres`
- `Password – ******`

b. **Click the Advanced tab.**

c. **Select a Database Schema to use.**

d. **Create a Table.**

e. **View the Table Data.**

Starting memcached

This section describes how to start memcached.

▼ To Start memcached

Before You Begin Make sure that the `libevent` library (`SUNWlibevent`) is installed.

1 **Log in as non-root user.**

Note – You cannot execute `memecached` as a root user.

---

2 **At the command prompt, type the following command:**

```
#memcached --options
```

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

`memcached` package for Solaris (`SUNWmemcached`) includes files necessary to register with the Service Management Facility (SMF) for the Java™ platform. For more information, see `memcached.1m` man page.

Testing the Development Environment

To test whether the development environment you have set up is working fine, perform the following simple steps:

▼ To Test the Development Environment

Before You Begin

- Make sure that you have initialized the environment for your login account.
- Make sure that the required components are started. For more information, see "Initializing the Development Environment" on page 12

1 **Select Developer Tools > Web Stack Admin > Sample App > Create to create a web application.**

Alternatively, you can copy the following code into a text file.

```
<DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
    <title>My First Web App</title>
    </head>
<body>
<b>Hello, World!</b>
<p>The date and time are <?php echo date ("r"); ?> - isn't that nice? - isn't that nice?</p>
</body>
</html>
```

2 **Save the file to** `/var/apache2/2.2/htdocs/hello.php`**.**

3 **Open a web browser and enter the URL http://localhost/hello.php in the address bar.**

The following message is displayed in the web browser, where the date and time reflects your system's current date and time:

Hello, world!

The date and time are Thu, 20 Dec 2007 16:01:07 +0200

- **Select Developer Tools > Web Stack Admin > Sample App > Run.**

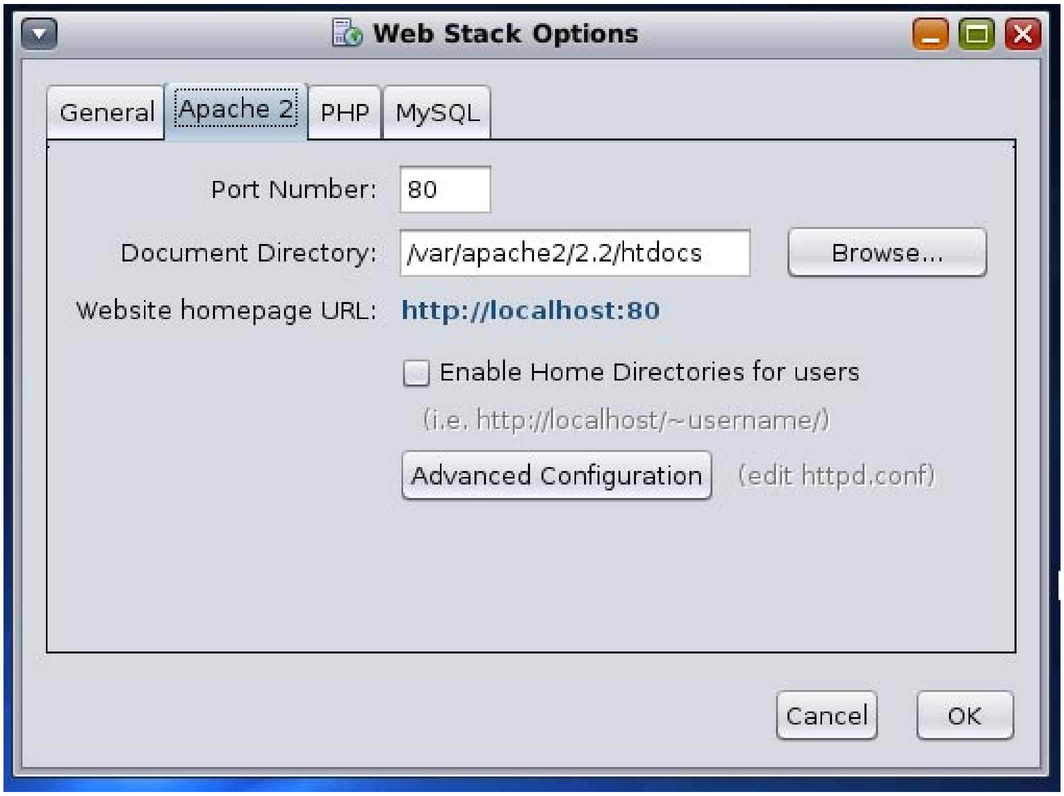
Next Steps Success! Note that the web page contents enclosed within the `<?php and ?>` de-limiters were interpreted by the PHP interpreter at the time the Apache Web Server served up that page. The output of those PHP language statements were substituted into the HTML stream at that location. All other HTML content is served up to the web browser as in static web pages.

Now, reload your browser but make sure that your browser does not use its cache; in most browsers, you can reload the browser by holding down the Alt key while clicking the Reload button. You see the same result, but with a later time. Thus you can see that a PHP page gets executed every time it is displayed (barring local browser caching, which you can suppress through PHP directives), and that you, as a developer, can generate highly dynamic content.

Customizing Web Stack Components

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.



Port Number: 80

Document Directory: /var/apache2/2.2/htdocs

Website homepage URL: http://localhost:80

☐ Enable Home Directories for users
(i.e. http://localhost/~username/)

Advanced Configuration (edit httpd.conf)

For advanced options such as to edit the configuration files, click `Advanced Configuration` on 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/php5/5.2.4/doc/html/configuration.html`.

For MySQL advanced configuration, you can edit the `my.cnf` file available at `/etc/mysql/5.0/my.cnf`.

Configuring Ruby with NetBeans 6.0

OpenSolaris 2008.5 installation includes Ruby 1.8 and NetBeans 6.0. To configure Ruby with NetBeans, see `http://www.netbeans.org/kb/60/ruby/setting-up.html`.

Configuring Ruby on Rails

Rails is a full-stack framework for developing database-backed web applications. Rails provides a pure-Ruby development environment, to the request and response in the controller, and to the domain model wrapping the database. RubyGems is the standard Ruby package manager. On the Solaris platform, RubyGems are already loaded. Hence you can install all of Rails and its dependencies through the command-line by typing the following command as a root user:

```
/usr/bin/gem install rails --include-dependencies
```

The Ruby on Rails environment is set up for the development. You can now use the NetBeans IDE to creating a Ruby on Rails project. When you create a project, the IDE creates the same folders and files that a rails command would create. For more information about how to create Ruby on Rails project in the NetBeans IDE, see the tutorial at `http://www.netbeans.org/kb/60/ruby/getting-started.html`.

Working with PHP, Apache, and NetBeans

For creating a PHP application using NetBeans and to deploy the application to Apache Web Server, here is what you need to do:

▼ To Configure PHP, Apache, and NetBeans

1. **Register Apache Web Server with NetBeans.**

   a. **In the NetBeans IDE, select the Services tab, select Web Servers > Add Host and select Local Web Server.**

   b. **Enter the correct path to Apache 2 configuration file.**

      The default path of the Apache configuration file, `httpd.conf` is: `/etc/apache2/2.2`.

   c. **Accept the default path for the Document Root field.**

   d. **Click Finish.**

      The Apache 2 web server is listed under the Web Server node.

2 **Create a PHP project using NetBeans.**

   a. **Select File > New Project.**

   b. **Select PHP Project from the PHP category.**

   c. **Enter a target server for your PHP project.**

   d. **Click Finish.**

3 **Write a PHP program.**

4 **Right click on the project and select Run Project to view the output on a browser.**

▼ To Enable Debug for PHP Applications

1 **From the Launch menu of the Solaris operating system, select All 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.**

Migrating From Development to Production Environment

The easy User Interface provided from the Launch menu for rapid and easy-to-learn web application development is intended for development purposes. 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

This section discusses the file layout of Web Stack components.

Note – The easy user interface provided from the Launch menu for rapid and easy-to-learn web application development is intended for development purposes. It is not recommended to run mission critical web sites without customizing and tuning of the components for performance, scalability, and security. When you have developed a working application, it is recommended to transfer the data from the database and PHP files from your development machine to a production-quality installation that has been carefully tuned to your mission critical needs.

You can access these components from the Launch menu of the Solaris platform. From the Launch menu, select All Applications > Developer Tools > Web Stack Admin.

Alternatively, the web stack tools can be accessed from the installation path described in the following sections.

Apache 2 HTTPd Server Files

The following list describes the file structure for Apache Web Server:

Note – The string `[version]` should be expanded to "`MAJOR.MINOR`". For example, `/usr/apache2/2.2/*`.

`/etc/apache2/[version]/httpd.conf`
: Contains server configuration files. A newly-installed server contains a default `httpd.conf` file. This is the main configuration file.

`/etc/apache2/[version]/conf.d`
: Contains additional server configuration files.

  By default, server will load all the `.conf` files placed under this directory. It also has 2 additional `.load` configuration files-`modules-32.load` and `modules-64.load` which contain `LoadModule` directives for loading the 32 and 64-bit bundled apache modules respectively. All the `.conf` files in this directory are included by the following line in `httpd.conf`:

  `Include /etc/apache2/2.2/conf.d/*.conf`. You can add the additional configuration here.

`/etc/apache2/[version]/envvars`
: Contains the environment settings that the server uses at startup.

When the `/usr` is mounted as read-only, you will not be able to modify the `envvars` file present in the following path `/usr/apache2/[version]/envvars`. To modify the environment settings you need to edit the `envvars` file present in the following path `/etc/apache2/[version]/envvars`.

`/etc/apache2/[version]/magic`
: Magic data for `mod_mime_magic` Apache module as documented in the

  `/usr/apache2/2.2/manual/mod/mod_mime_magic.html` file. Editing this file is not recommended.

`/etc/apache2/[version]/mime.types`
: Default MIME types file. This file sets the default list of mappings from filename extensions to content types, changing this file is not recommended. Use the `AddType` directive instead.

`/etc/apache2/[version]/original/`
: Contents under this directory are delivered as-is from the apache distribution and these files are not meant to be read by the server.

`/etc/apache2/[version]/sample-conf.d`
: Contains sample `.conf` files. These are not included in the main configuration file. To use the sample file, copy the file to `conf.d` directory and modify as per the need.

`/usr/apache2/[version]/bin`
: Contains the 32-bit `httpd` (Pre-fork MPM) and `httpd.worker` (Worker MPM) executables as well as other utility programs.

`/usr/apache2/[version]/bin/[isainfo]`
: Contains the 64-bit `httpd` (Prefork MPM) and `httpd.worker` (Worker MPM) executables as well as other utility programs.

`/usr/apache2/[version]/manual`
: Contains the Apache manual in HTML format.

`/usr/apache2/[version]/include`
: Contains the Apache header files, which are needed for building various optional server extensions with `apxs(8)`.

`/usr/apache2/[version]/libexec`
: Contains 32-bit loadable modules (DSOs) supplied with the server.

`/usr/apache2/[version]/libexec/[isainfo]`
: Contains 64-bit loadable modules (DSOs) supplied with the server.

`/usr/apache2/[version]/man`
: Contains man pages for the server, utility programs, and `mod_perl`.

  Add this directory to your MANPATH to read the Apache man pages.

`/usr/apache2/[version]/lib`
: Contains the 32-bit apache2 core libraries.

/usr/apache2/[version]/lib/[isainfo]
: Contains the 64-bit Apache 2 core libraries.

/usr/apache2/[version]/lib/perl
: Contains the 32-bit modules and library files used by the mod_perl extension to Apache.

/var/apache2/[version]/cgi-bin
: Default location for the CGI scripts.

  This can be changed by altering the httpd.conf file and restarting the server.

/var/apache2/[version]/htdocs
: Default document root.

  This can be changed by altering the httpd.conf file and restarting the server.

/var/apache2/[version]/icons
: Icons used by the server.

  This should not be changed.

/var/apache2/[version]/libexec
: Place holder for 32-bit user apache modules.

  Any 32-bit modules which are added using apxs(8) are copied into this directory.

/var/apache2/[version]/libexec/[isainfo]
: Place holder for 64-bit user Apache 2 modules.

  Any 64-bit modules which are added using apxs(8) are copied into this directory.

/var/apache2/[version]/logs
: Contains server log files.

  The formats, names, and locations of the files in this directory can be altered by various configuration directives in the httpd.conf file.

/var/apache2/[version]/proxy
: Directory used to cache pages if the caching feature of mod_proxy is enabled in the httpd.conf file.

  The location of the cache can also be changed by changing the proxy configuration in the httpd.conf file.

Modules mod_fcgid, mod_jk,mod_security, and mod_dtrace are integrated to Apache 2. For more information about these modules, see http://fastcgi.coremail.cn/doc.htm, http://tomcat.apache.org/connectors-doc/generic_howto/quick.html, http://www.modsecurity.org/documentation/index.html, and http://prefetch.net/projects/apache_modtrace/mod_dtrace.c

MySQL Database Files

The MySQL 5.0.45 software for Solaris is installed into a number of subdirectories of `/usr/mysql/5.0`. Symbolic links are created from all directories under `/usr/mysql/5.0` to `/usr/mysql` directories so that latest version of MySQL can also be accessed from `/usr/mysql`.

`/usr/mysql/5.0/bin`	Contains the binaries and scripts.
`/usr/mysql/5.0/lib`	Contains the libraries for the client API.
`/usr/mysql/5.0/include`	Contains the header files for the client API.
`/usr/mysql/5.0/man/man1`	Manual pages for client programs.
`/usr/mysql/5.0/man/man8`	Manual pages for server programs.
`/usr/mysql/5.0/share`	Shared data: locale, time zone
`/usr/mysql/5.0/docs`	Contains HTML documentation.
`/usr/mysql/5.0/mysql-test`	Contains MySQL test programs.
`/usr/mysql/5.0/sql-bench`	SQL benchmark test
`/usr/mysql/5.0/share/mysql`	Contains internationalization (I18N) files, sample configuration files, and utility scripts.
`/var/mysql/5.0/data`	Default database data directory.
`/etc/mysql/5.0`	Contains the MySQL configuration file.
`/usr/mysql/5.0/bin/mysql`	Client executable.
`/usr/mysql/5.0/bin/mysqld`	Server executable.

PHP 5.2.4 Files

The following list describes the file structure for PHP:

`/usr/php5/5.2.4/bin`	Contains the PHP configuration executables.
`/usr/php5/5.2.4/lib`	Contains PHP library files.
`/usr/php5/5.2.4/modules`	Contains PHP modules for PHP extensions.
`/usr/apache2/2.2/libexec`	Contains the PHP module for Apache 2 (32-bit) prefork MPM and the module name is `mod_php5.so`. This module is delivered as part of `SUNWapch22m-php5` package.
`/usr/php5/5.2.4`	Contains the default configuration file (`php.ini`).

`/usr/php5/5.2.4/conf.d`	Contains extension specific INI files for each PHP extension . Here, you can enable or disable various extensions that are integrated with the package.

Note – The PHP extensions like `APC`, `DTrace`, `XDebug`, `suhosin`, and`tcpwrap` are disabled by default. To enable these extensions, you need to uncomment the respective line (`extensions=<module-name>.so`) in the `conf.d` file.

PostgreSQL Database Files

The following list describes the file structure for the PostgreSQL database:

`/usr/postgres/8.2/bin`	Contains the PostgreSQL executables.
`/usr/postgres/8.2/etc`	Contains a sample database.
`/usr/postgres/8.2/jdbc`	Contains the JDBC drivers for PostgreSQL.
`/usr/postgres/8.2/lib`	Contains library files for OpenSolaris.
`/usr/postgres/8.2/man`	Contains the man pages for the commands.

Ruby Files

The following list describes the file structure for Ruby:

`/var/ruby/1.8/gem_home`	Contains the Rubygems repository. Configure the `GEM_HOME` environment variable to point to `/var/ruby/1.8/gem_home` to use this installation of Rubygems.
`/usr/ruby/1.8/bin`	Contains the Ruby executable as well as other utility programs, and Rubygems programs. These programs are linked from `/usr/bin`. For example: `/usr/ruby/1.8/bin/ruby` is linked from `/usr/bin/ruby1.8`, and may be linked from `/usr/bin/ruby` if 1.8 is the latest version of Ruby installed on this system.

Squid Files

The following list describes the file structure for Squid:

`/usr/squid/bin`	Contains the executable for the Squid client and to run the cache.
`/usr/squid/libexec`	Contains the libraries.
`/etc/squid/squid.conf`	The main configuration file. Modify this file for Squid to work.