Mac OS X Tips

For Lasso Professional Server 8.

OMNIPILOT™
SOFTWARE

Trademarks

Third Party Links

Copyright

Fourth Edition: December 1, 2004

OmniPilot Software, Inc.
701 S. Federal Highway
Fort Lauderdale, Florida 33316 U.S.A.

Telephone: (954) 315-9400
Fax: (954) 315-9410
Email: info@omnipilot.com
Web Site: http://www.omnipilot.com

Contents

1

Chapter 1

Introduction

This document contains an introduction to the terminal command line in Mac OS X and an explanation of Mac OS X file permissions. Understanding how these features of Mac OS X work with Lasso enable you take advantage of advanced Lasso features and customize how Lasso interacts with other Mac OS X applications.

Standard administration of Lasso Professional Server can be performed by accessing the command files from the Tools folder in the Finder and through the Lasso Administration Web interface. These techniques are covered in the *Installation* and *Administration* chapters in the Lasso Professional Server Setup Guide.

This document contains the following chapters.

- *Using the Terminal* – Describes how to use the terminal command line to start and stop Lasso Service and Lasso MySQL and how to check to see if Lasso is running.
- *Mac OS X File Permissions* – Introduces Mac OS X file permissions and describes how they interact with Apache and WebSTAR V's configuration and with Lasso's file tags permissions.

Additional information about the Mac OS X terminal command line interface and file permissions can be found in the documentation for Mac OS X or in third-party books about Mac OS X or UNIX. See the documentation for Apache and WebSTAR V for more information about those products.

Most of the procedures documented in this document can also be accomplished using third-party applications such as batCHMOD from arbysoft or Super Get Info from Bare Bones Software.

2

Chapter 2

Using the Terminal

This chapter contains the following sections.

- *Terminal Application* provides an overview of the Mac OS X terminal command line interface and common commands.
- *Using Super User* describes how to use sudo to execute commands.
- *Moving and Creating Files* describes how to use the terminal for file manipulation including moving Lasso connector for Apache.
- *Starting and Stopping Lasso* describes how to start and stop Lasso Service and Lasso MySQL through the command line.
- *Checking to See if Lasso is Running* describes how to check that Lasso Service and Lasso MySQL are running through the command line.

Terminal Application

Mac OS X is built on top of an open source operating system named Darwin and includes the BSD (Berkeley System Distribution) subsystem. This subsystem provides a command line interface and all the traditional UNIX command line tools. The Terminal application is the user-interface for the UNIX command line.

To launch the terminal application:

The Terminal application can be found in the Utilities folder within the Applications folder in a standard Mac OS X installation. Launch the Terminal application by double clicking on it.

The initial window contains a welcome message and a prompt which includes the name of the current machine localhost, the current path ~ (which represents the user's home directory `/Users/Example/`), and the username of the current user `example`.

```
Welcome to Darwin!
[localhost:~] example%
```

Commands can be typed at the prompt and the results will be displayed immediately after. Common commands are summarized in the *Common Mac OS X Commands* table.

Note: The examples in this guide do not include the prompt, but it will always be similar to the prompt shown above. The commands shown in each example should be typed exactly as shown after the prompt.

Table 1: Common Mac OS X Commands

Command	Description
cd <directory>	Change directory opens the specified directory.
cd ..	Moves one directory up in the file system.
cp <file> <directory>	Copies a file to the specified directory.
ls	Lists the contents of the current directory.
ls <directory>	Lists the contents of the specified directory.
ls -la <directory>	Lists the contents of the specified directory with more detailed information.
man <command>	Shows documentation for the command.
mv <file> <directory>	Moves a file to the specified directory.
ps auxc	Lists the processes running on the machine.
rm <file>	Removes the specified file from the currrent directory.
sudo <command>	Executes the specified command as the super user.
./<file>	Executes the specified file from the current directory.

These commands can be used to navigate through the file system, list the contents of directories, and perform commands. For example, the following sequence of commands navigate to the Lasso Professional 8 application folder and list its contents.

```
cd /Applications/Lasso Professional 8/
ls
```

```
➜ Documentation    Extensions      JavaLibraries   JDBCDrivers
  Lasso8Service    LassoAdmin      LassoApps       LassoErrors.txt
  LassoLibraries   LassoModules    LassoSites      LassoStartup
  Library          SQLiteDBs       Tools
```

The tools in Mac OS X are case sensitive, but the file system is not. Typing `cd /applications/lasso professional 8/` in the above example would work, but it is recommended to use the proper case when specifying file and directory names.

Mac OS X will auto-complete directory and file names. In the above example it is possible to type `cd /A` and then the tab key, followed by `L` and the tab key. If there is only one item in the directory that starts with `A` then the name of that item will be auto-inserted. If there is more than one item that starts with `A` then those items will be listed.

Wildcards can be used any time a file or directory name is expected. `A*` in a file name will expand to be any number of characters. `ls a*` will list only files in the current directory that begin with the letter `a`.

Paths in Mac OS X are specified from the root of the main hard drive on the system. The *Common Mac OS X Paths* table that follows lists some commonly used paths and their definitions.

Table 2: Common Mac OS X Paths

Command	Description
/	The root of the main hard drive.
./	The current directory.
../	The directory one level up in the file system.
~/	The current user's home directory, e.g. /Users/Example/.
/Applications/	The standard location of Mac OS X applications.
/Applications/4DWebSTAR/	The WebSTAR V application folder.
/Applications/Lasso Professional 8/	The Lasso Professional application folder.
/Applications/Utilities/	Contains utilities such as Terminal and Activity Monitor.
/Library/	Shared system or application resources.
/Library/StartupItems/	Items which are launched when the system is booted.
/Library/WebServer/Documents/	The Apache Web server root.
/System/	The system folder. The contents of this folder should never need to be modified.
/Users/<username>/	The location of users' home directories.
/Volumes/<drive name>/	The location of additional hard drives, mounted file systems, and mounted disk images.
/bin/	Contains BSD executables and command line tools.Similar to /Applications/.
/etc/	Contains system configuration files. Similar to /System/.
/usr/	Contains shared application resources. Similar to /Library/.
/usr/local/bin/mysql/	The default location of an installation of MySQL.
/var/	Contains temporary files.

Using Super User

Mac OS X is a multi-user system. File system permissions prevent one user from accessing files which are owned by another user. In order to make configuration of the machine possible, Mac OS X allows one or more users to be designated as administrators in the Users pane of System Preferences.

Administrators can run installers by entering their username and password when prompted, can modify all system preferences, and can run commands as the super user. The first account created by Mac OS X is always an administrator account.

An administrator can run commands as the super user using `sudo` in the terminal in order to bypass all file system permissions on the machine. This allows any administrator to modify files owned by another user, reassign file permissions, execute protected applications, etc. Once a command has been run using `sudo`, additional commands can be run for five minutes without re-entering the administrator's password.

It is important to think twice before running a command as super user since the normal protections of Mac OS X are not in place. It is possible to delete all files on the current hard drive or to render a system unbootable. It is wise to first try a command using normal user permissions and only to use super user permissions if required.

To run a command as super user:

Use the `sudo` command in the terminal. This command prompts for the administrator's password and then runs the command as super user. The following command shows an attempt to list the home directory of another user. The command is denied permission.

```
ls /Users/OtherUser/
```

```
➜ ls: : Permission denied
```

Next, the command is run using `sudo`. Once the administrator's username has been entered the command runs, bypassing the file system permissions.

```
sudo ls /Users/OtherUser/
```

```
➜ Desktop          Documents        Library          Movies
  Music            Pictures         Public           Sites
```

Note: UNIX-based system define a user named `root` which can perform any commands on the system as super user. Mac OS X also defines a `root` user, but it is deactivated by default. Except for in extreme misconfigurations it is never necessary to activate the `root` user on a Mac OS X system. Any commands which require super user privileges should use `sudo` instead.

Moving and Creating Files

Files can be copied, moved, deleted, or created using the terminal command line. Many Mac OS X files are not shown in the Finder and must be accessed through the command line. Files which are located in the hidden BSD folders such as `/etc/` or `/usr/` also must be accessed through the command line. Finally, the command line allows use of `sudo` to access files which can not normally be accessed by the current user.

To create or edit a file in a hidden directory:

The file which specifies options for the Lasso connector for Apache is called `/etc/lasso/lasso8apache.conf`. This file cannot be seen in the Finder, but can be accessed through any of the following methods.

- Use the Mac OS X text editor `pico` to create and edit the file. The following command will edit the file if it exists or create it if it doesn't.

  ```
  sudo pico /etc/lasso/lasso8apache.conf
  ```

- BBEdit includes a command line tool that can be used to create the file and then edit the file using the BBEdit application. Consult the BBEdit documentation for details. The following command will edit the file if it exists or create it if it doesn't. Note the `-c` option.

  ```
  sudo bbedit -c /etc/lasso/lasso8apache.conf
  ```

To move a file from one directory to another:

Lasso connector for Apache is implemented in file named `Lasso8ConnectorforApache.so` which is stored in the hidden `/usr/libexec/httpd/` folder. In order to use Lasso's distributed architecture features (running Lasso Service on one machine and the Web server on another) this file would need to be moved to a different machine.

1. Use the following command to move the `LassoConnectorforApache.so` file to the current user's desktop folder on the machine will hosts Lasso Service.

   ```
   sudo mv /usr/libexec/httpd/Lasso8ConnectorforApache.so ~/Desktop/
   ```

2. Copy the file to the machine which will be running Apache using the Finder and file sharing. We assume the file is copied to the desktop of the current user on the second machine.
3. Use the following command to move the file into the `/usr/libexec/httpd/` folder.

   ```
   sudo mv ~/Desktop/Lasso8ConnectorforApache.so /usr/libexec/httpd/
   ```

Note: See the Lasso Professional Server Setup Guide for details about setting up Lasso using a distributed architecture.

Starting and Stopping Lasso

Lasso Service can be started and stopped in the terminal using `lasso8ctl` or the command files from the `Tools` folder. Follow the instructions below to access the `Tools` folder, then use one or more of the particular commands to start or stop Lasso Service.

The command files from the `Tools` folder can also be used from the Finder by double-clicking on them. See the *Mac OS X Installation* chapter in the Lasso Professional Server Setup Guide for more information.

There are two ways to start Lasso Service:

- *Service* – Lasso Service is normally started as a service (background application, faceless application, or dæmon). Lasso Service starts as a service when the system boots. MySQL, Apache, and WebSTAR are all similarly run as services.

  When started as a service the Lasso Watchdog script will automatically restart Lasso Service if it quits unexpectedly. Lasso Service will run even if no user is logged in to their account and will continue to run as multiple users log in and out.

- **Console** – Lasso Service can be started in console mode in order to view debugging and status information within the terminal window. When in console mode the startup messages of Lasso Service can be viewed, each SQL statement issued to Lasso MySQL can be viewed, and messages logged using `[Log_Always]` can be seen.

  When in console mode, the Lasso Watchdog script does not run so if Lasso Service quits unexpectedly it will not be restarted automatically. Console mode should only be used when an administrator is actually watching the console. For unattended servers Lasso Service should be started as a service.

To start Lasso Service:

Lasso Service starts automatically when the machine is booted so it is not normally necessary to start Lasso Service. However, these commands can be used to start or restart Lasso Service if needed.

- **Finder** – Double click the `startLasso8Service.command` file in the `Tools` folder for Lasso Professional Server. The command will prompt for an administrator password if needed and start Lasso Service as a service.
- **Terminal** – Type the following commands to start Lasso Service as a service.

  ```
  sudo lasso8ctl start
  ```

The command will prompt for an administrator password if needed. This command will not stop Lasso Service if it is already running. The following command can be used to restart an already running Lasso Service.

```
sudo lasso8ctl restart
```

To stop Lasso Service:

- **Finder** – Double click the stopLasso8Service.command file in the Tools folder for Lasso Professional Server. The command will prompt for an administrator password if needed and stop Lasso Service whether it is running as a service or in console mode.
- **Terminal** – Type the following command to stop Lasso Service.

  ```
  sudo lasso8ctl stop
  ```

  The command will prompt for an administrator password if needed. This command will stop Lasso Service running either as a service or in console mode. Use the instructions above to restart Lasso Service.

To start Lasso Service in console mode:

- **Finder** – Double click the consoleLasso8Service.command file in the Tools folder for Lasso Professional Server. The command will prompt for an administrator password if needed, stop Lasso Service if it is running, and start Lasso Service in the current window.
- **Terminal** – Type the following commands to start Lasso Service in console mode. Lasso MySQL should already be running when Lasso Service is started. Lasso Service will run in the terminal window and will be quit if the terminal window is closed or the current user logs out.

  ```
  cd /Applications/Lasso Professional 8/Tools/
  ./consoleLasso8Service.command
  ```

  The command will prompt for an administrator password if needed, stop Lasso Service if it is running, and start Lasso Service in the current window. To stop Lasso Service running in console mode either type `control-c` or use the instructions below.

Note: When running in console mode Lasso Service will not restart automatically if it quits unexpectedly. It is advisable to run Lasso Service as a service if the server is to be unattended.

Checking to See if Lasso is Running

Since Lasso Service is generally run as background processes it is necessary to use the Activity Monitor or commands in the terminal to check and see whether they are running. See the *Mac OS X Installation* chapter in the Lasso Professional Server Setup Guide for more information about using Activity Monitor.

Each site in Lasso Professional Server will show as a separate process in the listings described below. There should be one process for each site defined in Server Administration plus one additional process for the master site.

Note: Advanced users who are creating shell scripts to interact with Lasso Service should look at the source of the commands in the Tools folder and the Lasso8Service.sh script in the LassoAdmin folder for examples of how to check that Lasso Service is running. For best results, shell scripts should call the commands in the Tools folder rather than stopping and starting Lasso Service directly.

To check whether Lasso Service and Lasso MySQL are running:

- **Activity Monitor** – Open the Activity Monitor from the Utilities folder inside the Mac OS X Applications folder. Select All Processes from the pop-up menu and type lasso in the text input. Lasso8Service should be shown in the listing multiple times.
- **Terminal** – Type the following command to check whether Lasso Service is running. This is actually two commands. The first command ps returns a list of processes. The parameters auxc format the process list so it is easy to read. The output of this command is sent into a second command using the vertical bar symbol | called a pipe. The second command grep filters out all lines except those containing the word lasso.

  ```
  ps auxc | grep lasso
  ```

  ```
  ➜ lasso 1423 0.0 0.0 1732 152 p1 S+ 0:00.05 sh
    lasso 1429 0.0 3.3 212516 10664 p1 S+ 11:51.00 Lasso8Service
    lasso 1429 0.0 3.3 212516 10664 p1 S+ 11:51.00 Lasso8Service …
  ```

  The output should be similar to that shown above. The first line represents a shell script sh that is monitoring Lasso Service and restarting it if it quits unexpectedly. This line will only be present if Lasso Service is started in the background. The other line tells whether Lasso8Service is started. If Lasso Service is not listed then it can be restarted using the instructions above.

To check whether the Web server is running:

All access to Lasso Service is through a Web server. If the Web server isn't running then Lasso Service cannot be accessed even if it are operating perfectly.

- The easiest way to check whether a Web server is running is to use a Web browser to load a page on the Web server. Try loading the default page on the Web server, then a Lasso specific page. For example, a Web server on the same machine as the Web browser can be accessed using the following URLs.

  ```
  http://localhost/
  http://localhost/Lasso/
  ```

- **Apache** – The default Web server of Mac OS X is controlled through the Sharing pane of System Preferences. Check that Personal Web Sharing is on. Apache can be restarted by selecting the Stop button, waiting, then selecting the Start button. Apache can also be viewed in the Activity Monitor by typing http in the text input. One or more httpd processes should be visible. The following command will provide a list of httpd processes in the terminal.

  ```
  ps auxc | grep httpd
  ```

- **WebSTAR V** – Consult the documentation for WebSTAR V to best determine whether it is running properly or not. WebSTAR V includes applications to monitor and administer the Web server. WebSTAR V can also be viewed in the Activity Monitor by typing ws in the text input. A WSWebServer process and other W… processes should be visible. The following command will provide a list of WebSTAR V processes in the terminal.

  ```
  ps auxc | grep WS
  ```

3

Chapter 3

File Permissions

This chapter contains the following sections.

- *Introduction* provides an overview of Mac OS X file permissions and file permission commands.
- *File Tags and File Permissions* describes how Lasso interacts with Mac OS X file permissions when using the [File_...] and [Log] ... [/Log] tags.
- *Apache File Permissions* describes how the Mac OS X standard Apache interacts with Mac OS X file permissions.
- *WebSTAR V File Permissions* describes how WebSTAR V interacts with Mac OS X file permissions.

Introduction

Mac OS X is a multi-user operating system and includes a robust system of UNIX-style file permissions based on the underlying Darwin and BSD subsystems. This system of file permissions should be familiar to anyone who has administered a UNIX, Linux, or BSD operating system, but is probably unfamiliar to Macintosh users.

The sections that follow offer a brief introduction to viewing, understanding, and changing Mac OS X file permissions. Subsequent sections describe how to configure file permissions for use with Lasso file tags, Apache, and WebSTAR V.

This chapter assumes familiarity the Mac OS X terminal which is introduced in the *Using the Terminal* chapter in this document.

Terminology

Each file or directory in Mac OS X has two owners. The file is owned by a User and by a Group. File permissions can be assigned separately for a file's user, group, and for all Other users of the machine.

There are three types of file permissions. Read permission allows a user to read the contents of a file or list a directory. Write permission allows a user to alter files or delete files from a directory. Execute permission allows a user to run an application or command or cd into a directory.

The three permissions can be given to the user, group, and all other users separately. A given file could have all permissions granted for the user owner, only read permission granted for the group owner, and no permission granted for others to access the file. More details are given below in the section on viewing, understanding, and changing file permissions.

The user who owns a file does not need to belong to the group which owns a file. If either the user or the group (or all other users) have permission to perform an operation on the file then the operation will be allowed.

Default Groups

Users on Mac OS X belong to one or more groups. All users who can log into the machine typically belong to the `staff` group. All users who are allowed to administer the machine belong to the `admin` group. Other special purpose groups and standard UNIX groups like `wheel` are defined as well.

Default Users

A Mac OS X system starts with a few pre-defined users. `root` is the super user of the machine and is deactivated by default. When running commands using `sudo` an administrator is effectively running commands as root. `www` is a predefined user which Apache uses for file access. Other special purpose users and standard UNIX users are defined as well.

The Lasso Professional Server installer automatically creates a new user lasso which belongs to the staff group. The instructions for installing WebSTAR V have the administrator create a new webstar user which automatically belongs to the staff group.

Viewing File Permissions

Mac OS X file permissions can be viewed using the `Show Info` command in the Finder or through the terminal using the `ls` command. The terminal provides more information and allows the permissions for all the files within a folder to be viewed at once.

Note: The permissions for files and folders can also be viewed using third-party applications such as batCHMOD from arbysoft or Super Get Info from Bare Bones Software.

To show the permissions for the files in a folder:

Use the `ls` command with the `-al` parameter. The `-a` parameter instructs ls to show all files in the specified folder. The `-l` parameter instructs `ls` to provide longer output, including information about file permissions.

```
cd /Applications/Lasso Professional 8/
ls -al
```

```
➜ drwxrwxr-x 12 lasso admin 364 Feb 6 12:08 .
  drwxrwxrwx 33 root admin 1078 Feb 4 15:33 ..
  drwxrwxr-x 7 lasso admin 264 Feb 1 16:51 Admin
  drwxrwxr-x 8 lasso admin 264 Feb 1 16:52 Documentation
  drwxrwxr-x 10 lasso admin 296 Feb 1 16:52 LassoModules
  drwxrwxr-x 9 root admin 262 Feb 1 16:51 LassoMySQL
  -rwxrwxr-x 1 lasso admin 1718644 Feb 1 16:26 Lasso8Service
  drwxrwxr-x 4 lasso admin 264 Feb 1 16:52 LassoStartup
  drwxrwxr-x 10 lasso admin 296 Feb 1 16:52 Tools
```

The first column describes the file permissions of each file and directory. A leading `d` indicate that an item is a directory. The first two items listed are . for the current directory and .. for the directory which contains the current directory.

The next three letters define the permissions for the user owner of the directory; `r` for read permission, `w` for write permission, and `x` for execute permission. A hyphen in place of a letter means that a user does not have the given permission. The owner of each item is listed in the third column. Most of the items in this folder are owned by `lasso`.

The second set of three letters define the permissions for users who belong to the owner group. The owner group of each item is listed in the fourth column. Most of the items listed are owned by the `admin` group.

The third set of three letters define the permissions for all other users on the machine. The listing also includes the size of each item, the creation date, and the name.

Files and directories can also have various flags set. The most common is the immutable flag which prevents even the root user or `sudo` from deleting a file. See below for instructions on how to unset the immutable flag so files in the Finder's trash which refuse to be deleted when the `Empty Trash` command is used can be deleted.

Changing File Permissions

File permissions are changed using four commands which are summarized in the *File Permissions Commands* table. Each command is described in detail below.

Table 1: File Permissions Commands

Command	Description
chflags <flag> <file>	Used to change various settings for a file or directory.
chgrp <group> <file>	Used to change the group owner for a file or directory.
chmod <flags> <file>	Used to change the permissions for a file or directory.
chown <user> <file>	Used to change the user owner for a file or directory.

Note: File permissions can also be changed using third-party applications such as batCHMOD from arbysoft or Super Get Info from Bare Bones Software. These utilities support the same operations that are documented here using the terminal.

To change the group owner of a file or directory:

Use the `chgrp` command. This command does not change any of the permissions of a file or directory; it simply changes the group owner. The new group owner has the same permissions assigned as the previous group owner.

The following example creates a new folder named `mydirectory` in the current directory then changes the group owner of that directory to `staff`. The result shown include just the line from the listing which contains the permissions for the `mydirectory` directory.

```
mkdir mydirectory
sudo chgrp staff mydirectory
ls -al
```

```
➜ drwxr-xr-x 2 example staff 24 Feb 7 14:09 mydirectory
```

The directory is now owned by the current user and the group `staff`.

Multiple items can be changed at once using wild card characters. For example, `chgrp staff my*` would change the group owners of all items in the current directory whose names start with `my`. The `-R` parameter can be used to change all items within a directory. `chgrp -R staff mydirectory` would change the group owners of all items within the named directory including sub-directories.

To change the user owner of a file or directory:

Use the chown command. This command does not change any of the permissions of a file or directory; it simply changes the user owner. The new user owner has the same permissions assigned as the previous user owner.

The following example changes the user owner of the mydirectory directory from the previous example to lasso. The result shown includes just the line from the listing which contains the permissions for the mydirectory directory.

```
sudo chown lasso mydirectory
ls -al
```

```
➜ drwxr-xr-x 2 lasso staff 24 Feb 7 14:09 mydirectory
```

The directory is now owned by the lasso user and the group staff.

Multiple items can be changed at once using wild card characters. For example, chown lasso my* would change the user owners of all items in the current directory whose names start with my. The -R parameter can be used to change all items within a directory. chown -R lasso mydirectory would change the user owners of all items within the named directory including sub-directories.

To change the permissions for a file or directory:

Use the chmod command. This command can change the permissions for the user owner, group owner, or other users to any combination of read, write, and execute.

The format of a chmod command is chmod <flags> <items>. The flags determine what permissions the items are given. Flags are built up by specifying whose permission to change: u for the user owner, g for the group owner, or o for other users, or a for all three; how to change the permission: + to add a permission, - to remove a permission, or = to set a permission; and what permission to set: r for read, w for write, and x for execute.

For example, the flags a=rwx would grant all users read, write, and execute permissions. The flags u+x would add execute permission to whatever permissions the user owner already had. The flags o-rwx would remove read, write, and execute permissions from other users.

Compound flags are possible as in g-x+rw which would remove execute permission from the group owner while granting read and write permissions.

Finally, multiple sets of flags can be concatenated using a comma as in `ug+x,o-x` which would grant execute permission to the user and group owners of an item while removing execute permissions from all other users.

The following example changes the permissions of the `mydirectory` directory from the previous example to `u+rwx,go-rwx`. The user owner can perform any actions, and neither the group or other users can perform any actions.

The result shown include just the line from the listing which contains the permissions for the `mydirectory` directory.

```
sudo chmod u+rwx,go-rwx mydirectory
ls -al
```

```
➜ drwx------ 2 lasso staff  24 Feb 7 14:09 mydirectory
```

The directory can now be accessed only by the user owner `lasso` and not by users in the group owner `staff` or any other users of the machine.

Multiple items can be changed at once using wild card characters. For example, `chmod a+rwx my*` would change the permissions of all items in the current directory whose names start with `my`. The `-R` parameter can be used to change all items within a directory. `chmod -R a+rwx mydirectory` would change the permissions of all items within the named directory including sub-directories.

To change the immutable flag on a file or directory:

Use the `chflags` command. This command modifies various flags which can be set on files including the immutable flag. The following command will unset the immutable flag for all files in the directory mydirectory.

```
sudo chflags -R nouchg mydirectory
```

The immutable flag is somewhat similar to the Finder's file locking feature. It prevents files from being modified or deleted. If there are files in the Finder's trash which will not go away when the Empty Trash command is used then the following command can be used to unset the immutable bit so those files can be deleted.

```
cd ~/.Trash
sudo chflags -R nouchg *
```

File Tags and File Permissions

Lasso Professional Server automatically creates a user `lasso` on installation. When Lasso Service is launched the permissions of the user *lasso* and the group *staff* are used to determine what files Lasso can access when the file tags or log tag are used.

The *File Tags Permissions* table lists the LDML file tags and what permission Lasso requires on the directory or file being acted upon in order for the action to take place. The permission required can be granted to either the user owner if the lasso user owns the file, the group owner if the staff group own the file, or to all other users.

The results are summarized here and specific recommendations for setting up permissions are provided after the table.

- Execute permission is required for any directory in which Lasso is going to perform a file action. Execute permission is required for the directory that contains any files which are going to be created, deleted, read, or inspected and for any directories which are going to be created, deleted, listed, or inspected.
- Execute permission is required for both the source and destination directories for move, copy, and rename operations. In addition, the source files for these operations must have read permission.
- Read permission is required for any file which is going to be read or inspected and for any directory which is going to be listed.
- Write permission is required for any file which is going to be modified and for any directory in which files or directories are going to be created.
- Execute permission is never required for files.
- No read or write permission is required to delete a file, only execute permission on the directory that contains the file.
- With -FileOverwrite set in the [File_Copy], [File_Move], or [File_Rename] tags they do not require write access for the destination file, merely execute permission for the destination directory.

Note: In addition to the terminal based instructions provided in this chapter, file permissions can also be changed using third-party applications such as batCHMOD from arbysoft or Super Get Info from Bare Bones Software.

Table 2: File Tags Permissions

Tag	Permissions
[File_Copy]	Execute for the source directory. Execute and write for the destination directory. Read for the source file.
[File_Create]	Execute for the destination directory.
[File_Delete]	Execute for the file's directory.
[File_Exists]	Execute for the file's directory. Read for the file.
[File_GetLineCount]	Execute for the file's directory. Read for the file.
[File_GetSize]	Execute for the file's directory. Read for the file.
[File_IsDirectory]	Execute for the directory's parent directory.
[File_ListDirectory]	Execute for the directory's parent directory. Read for the directory.
[File_ModDate]	Execute for the file's directory. Read for the file.
[File_Move]	Execute for the source directory. Execute and write for the destination directory. Read for the source file.
[File_ReadLine]	Execute for the file's directory. Read for the file.
[File_Read]	Execute for the file's directory. Read for the file.
[File_Rename]	Execute for the source directory. Execute and write for the destination directory. Read for the source file.
[File_SetSize]	Execute for the file's directory. Write for the file.
[File_Write]	Execute for the file's directory. Write for the file.
[Log] ... [/Log]	Execute for the log file's directory.

To set the permissions so Lasso can operate in a directory:

There are two basic ways to establish permission for Lasso to perform file tags actions within a directory. The first method is preferred.

- Change the user owner of the directory to `lasso` and grant all permission to the user. The following example modifies a directory named `mydirectory` in the Web server root. The `-R` parameter ensures that all files in the directory and sub-directories can be accessed by Lasso.

  ```
  cd /Library/WebServer/Documents
  sudo chown -R lasso mydirectory
  sudo chmod -R u+rwx mydirectory
  ```

- Give all users permission to read, write, and execute within the directory. The following example modifies a directory named `mydirectory` in the Web server root. The `-R` parameters ensures that all files in the directory and sub-directories can be accessed by any users of the machine.

  ```
  cd /Library/WebServer/Documents
  sudo chmod -R o+rwx mydirectory
  ```

To set the permissions so Lasso can operate on a file:

There are two basic ways to establish permission for Lasso to perform file-tags actions on a specific file. The first method is preferred.

- Change the user owner of the file and the directory in which the file exists to `lasso` and grant all permission to the user. The following example modifies a file named `log.txt` inside a directory named `mydirectory` in the Web server root.

  ```
  cd /Library/WebServer/Documents
  sudo chown lasso mydirectory
  sudo chmod u+rwx mydirectory
  sudo chown lasso mydirectory/log.txt
  sudo chmod u+rwx mydirectory/log.txt
  ```

- Give all users permission to read, write, and execute for the file and the directory in which the file exists. The following example modifies a file named `log.txt` inside a directory named `mydirectory` in the Web server root.

  ```
  cd /Library/WebServer/Documents
  sudo chmod o+rwx mydirectory
  sudo chmod o+rwx mydirectory/log.txt
  ```

Apache File Permissions

Lasso Professional Server automatically creates a user `lasso` on installation. When Lasso Service is launched the permissions of the user lasso and the group staff are used to determine what files Lasso can access when the file tags or log tag are used. For Apache, the permissions of the user `www` and the group `www` are used to determine what files can be served to Web clients. In order to serve pages both through Lasso and through Apache, files must be readable by both programs.

One solution is to make all files which will be read by Apache and Lasso readable by all other users on the machine. Then the user owner and group owner of the files is unimportant.

To set the permissions so both Apache and Lasso can serve files:

Give all users permission to read the files and permission to read and execute the directory in which the files exist. The following example modifies all files inside a directory named `mydirectory` in the Web server root. The `-R` parameter ensures that all files in the directory and sub-directories can be accessed by Lasso and Apache.

```
cd /Library/WebServer/Documents
sudo chmod -R o+rx mydirectory
```

Note: In addition to the terminal based instructions provided in this chapter, file permissions can also be changed using third-party applications such as batCHMOD from arbysoft or Super Get Info from Bare Bones Software.

WebSTAR V File Permissions

Lasso Professional Server automatically creates a user `lasso` on installation. When Lasso Service is launched the permissions of the user `lasso` and the group `staff` are used to determine what files Lasso can access when the file tags or log tag are used. For WebSTAR V, the permissions of the user `webstar` and the group `webstar` are used to determine what files can be served to Web clients. In order to serve pages both through Lasso and through WebSTAR, files must be readable by both programs.

One common solution is to make all files which will be read by WebSTAR V and Lasso readable by either the `staff` group or all other users on the machine.

To set the permissions so both WebSTAR V and Lasso can serve files:

- Give all users permission to read the files and permission to read and execute the directory in which the files exist. The following example modifies all files inside a directory named `mydirectory` in the Web server root. The `-R` parameter ensures that all files in the directory and subdirectories can be accessed by any user of the machine.

  ```
  cd /Applications/4DWebSTAR/WebServer/DefaultSite/
  sudo chmod -R o+rx mydirectory
  ```

- Assign the files and the directory in which the files exist to the `staff` group. Give the group owner permission to read the files and permission to read and execute the directory. The following example modifies all files inside a directory named `mydirectory` in the Web server root. The `-R` parameter ensures that all files in the directory and sub-directories can be accessed by any users of the machine who belong to the group `staff`..

```
cd /Applications/4DWebSTAR/WebServer/DefaultSite/
sudo chgrp -R staff mydirectory
sudo chmod -R o+rx mydirectory
```

Note: In addition to the terminal based instructions provided in this chapter, file permissions can also be changed using third-party applications such as batCHMOD from arbysoft or Super Get Info from Bare Bones Software.