INSTALLING SOFTWARE ON LINUX

One of the most difficult things to get used to in the Linux world is installing new software packages. In the world of Windows, every program comes with a `Setup.exe` program that asks you some very easy questions and takes care of the job for you. While Linux software can be almost that easy to install, you will sometimes find software that seems to fight every step of the way. I can't cover all the problems you might run into, but I'll try to give you the basics and a few pointers to help get you over the rough spots.

Software tends to come in "packages". In the Windows world a package is a `Setup.exe` or a `program.zip` file. On a Mac a package is a `program.dmg` or a `program.sit` file. In the Linux world, there are several kinds of packages, and each distribution has its own preferred package format.

The standard Linux package format (according to the **Linux Standard Base**) is *RPM*. RPM is a packaging system originally developed by Red Hat and widely used in the Linux community. Distributions using it include Fedora, Mandriva, Red Hat (naturally), and SUSE. An RPM package file normally will be named something like `program-version-other.rpm`

Another popular package format is *DEB*, the Debian software package. Debian packages and the Advanced Packaging Tool (APT) were the first to introduce several advanced features that are now common, such as automatic dependency resolution and signed packages. Debian packages are used by Debian GNU/Linux (naturally), and distributions based on it, including Ubuntu, Knoppix, and Mepis. A Debian package file normally will be named something like `program-version-other.deb`

Remember, you will need to ***become SuperUser*** to install software.

Debian, Ubuntu: APT

There is a broad array of tools for working with DEB packages, but the one you will commonly use is `apt-get`, arguably the easiest of Linux package management tools. `apt-get`is so easy because it not only keeps track of what packages are installed, but also what other packages are available. It will even download them from the Internet for you (if properly configured).

```
apt-get install ${packagename}
```

To remove software is just as easy.

```
apt-get remove ${packagename}
```

Although the repositories that contain installable packages might live on the Internet or on a disc somewhere, APT keeps a local database on your hard drive with a list of all available packages and where to find them. This database needs to be explicitly updated. To update the APT database:

```
apt-get update
```

A common idiom is to update your package database, and then upgrade all the packages that have patches or security updates to install. The following command will do this all at once.

```
apt-get update; apt-get upgrade
```

For a more indepth `apt-get` tutorial and other resources, see ***Managing Software with APT and dpkg***.

Fedora, Red Hat: `yum`

`yum` does for RPM packages roughly what `apt-get` does for Debian packages. Like `apt-get`,`yum` can download and install packages from a configured repository.

```
yum install ${packagename}
```

To remove software is just as easy.

```
yum remove ${packagename}
```

`yum` does not keep a local copy of your package database by default, so normally there is no need to update it. To install all available security patches and bug fixes, use this command:

```
yum update
```

You can also explicitly update a single package with:

```
yum update ${packagename}
```

For a more indepth `yum` tutorial and other resources, see ***Managing Software with yum and rpm***.

Mandriva: `urpm`

Mandriva Linux (formerly Mandrake and Connectiva) has a toolset similar to APT called`urpmi`. To install software:

```
urpmi ${packagename}
```

To remove software:

```
urpme ${packagename}
```

To update the local package database:

```
urpmi.update -a
```

To install security updates and bug fixes:

```
urpmi --auto-select
```

For a more indepth `yum` tutorial and other resources, see ***Managing Software with urpm***.

Tar Balls

No, this is not a naughty term! A *tar ball* is a (usually compressed) archive of files, similar to a Zip file on Windows or a Sit on the Mac. Tar balls come in files that end in `.tar`, `.tar.gz`,`.tgz`, or something along these lines. To unpack a tar ball, use this command.

```
tar -xzvf ${filename}.tar.gz
```

The parameters are `x` to extract files, `z` to filter through gzip for decompression (leave this off if the file does not have a `gz` extension), `v` for verbose mode so you can tell what's going on, `f` indicating there will be a filename to follow. You may want to create an alias called "untar" that feeds in these options if you have a hard time remembering command line options as I do.

This command will not install the software, it will only extract the archived files. It is your job then to find the README file or INSTALL file and read its instructions for installation.

If the archive contains binaries there will usually be a setup script (often called `install.sh`) that you must execute as ***SuperUser***.

Very often, software delivered in tar balls is not in executable form, but in source code, which must first be compiled before it can be installed. For more details on this, see***Installing Software from Source Code***.

Other Systems

Some other Linux distributions have their own way of managing packages, notably SUSE. SUSE uses RPM as its native package format, but has its own high level tool to manage system software installation.

SUSE Linux uses a tool called `yast` (which allegedly is an acronym for Yet Another Setup Tool) to perform all kinds of system administration tasks, including installing software. Having no experience with it, I cannot give you more details. `man yast` for help.

Source : http://www.control-escape.com/linux/lx-swinstall.html