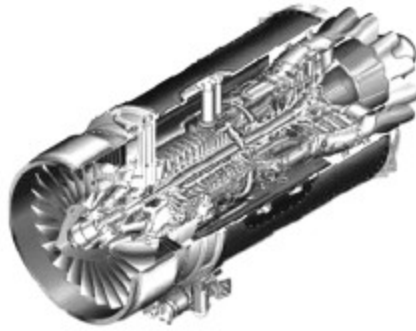


OVERVIEW of Facets required to Construct A Small Business Server (LINUX – APACHE)

*Requires a spare phone line at the business premises.



Written by Mr. Samuel A Marchant (--/5/2008)
*unfinished 14/06/2008 Will be introducing soon
some re-worded(for application into this article)
information that will require refering to the author
and origin of the informations work(journalising).*

www.nicephotog-jsp.net

nicephotog@yahoo.com.au

nicephotog@gmail.com

INDEX

- A. Why your own server**
- B. Some special points and rules before starting**
- C. Before installing the Operating System**
 - 1. Required Parts**
 - 2. Costs**
 - 3. *Getting Started**
 - 4. Setting Up the Server and Operating System**
 - 5. Post (after) Operating System Install**
 - 6. Configuring and/or Installing Server(s)**
 - 7. httpd.conf the Apache http Server configuration file**
 - 8. php.ini PHP's configuration file**
 - 9. Installing a Tomcat JSP/Servlet Server and a JDK/SDK**
 - 10. [Libraries] J2EE, Java Mail API .jar and JSP, JSF, Custom Tags, Beans**

*This document is intended for **small businesses**(1 premises) with **high volume requirements of public information supply by a net http service DNS site.***

*“tty0” tty followed by a number is Linux version of labeling main-board hardware port IRQ's
[there are other IRQ port names but are not particularly required here]*

Warning: this article presumes you are familiar with COM/tty ports and special PCI or ISA card connections and general main-board computer hardware handling and fitting requirements along with a comprehension of computer hardware board pre-install configuration requirements.

It is also a little extra extensive as an overview, but that is to point you in the right direction to solve your isolated individual problems (if they occur) of the personally created configuration that you make from this article, since beyond it there will be no help present.

A. Why your own server

One avoidance of using a hosting provider and dedicated server leasing is to connect a server box or PC converted box inside your business premises to the inter-net, but, it really is more for the point of personal control of data and its security of ownership. Outside that though, is the ability to

expand(scalability) with accord to business requirements by use of *in house application development to solve any mission critical requirements that manifest*.

This last reason is the actual inherent reason for operating your own business server more than pure data ownership and control. While hosting providers do their best inclusive at the level of supplying application and OS choices for dedicated services, they do not qualify for *true legal privacy* finally and are a hindrance to activities such as SOA architecture that drives much of service communication with customers(money/economics).

If we take a look at one particular type of business we all know so well in Australia, we can see how it comes to be anyone operating a small business would make the choice of using their own server and not a hosting provider.

Hypothetically(some parity to, why, and with what for a small business)

The *Newsagent* is the last small business that would possibly ever use its own server. For most because of its simplicity and rigid stocking principles, it could only quite have a use for its own server and not a hosting provider.

A newsagent will have between 200 to 500 different individual magazine and newspaper medias to sell and would probably like to show each reader the type of service each print offers. This means to best serve advertising purpose he would have at least 200 to 500 individual pages to show these each for their being aside to any customer processing and customer accounts.

This also means that at a reasonable guess he would have around 700 to 1000 images to entice his customers with the conceptual representation of the medias.

For any company that is not effectively able to present common stable(items that do not require changing information relating to them, or placing and removing them intermittently) items, such as print media by point of its individual idiosyncratic nature, in context to the business and market it belongs, will require to present such items carefully, and as exposed to availability as possible, hence, 24 hour inter-net as a covering part of the sales and promotion technique.

What it gives to have your own server though ultimately is flexibility and scalability of services by virtue of its total control over in house development and usage apart more reputable customer personal data privacy.

Alternately, this can be much the same in terms of cost outlay if you use a *dedicated server* as provided by a hosting provider. Dedicated server packages often include all of the below that is discussed in this document(not that you ever touch it or see it, except through a special remoting console, neither is the choice of daemon apps and Operating System flavor particularly bendable, that is why having your own business server box is often done. [FOR MEGALOMANIAL POWER AND CONTROL] , *i'd rather be a megalomaniac, what about you?*).

B. Some special points and rules before starting

To assist jumping straight to the actualities, there are a few points that will help you to not befall any managerial or process problems *of implementation of creating your own business DNS server*. There are some *do's and don'ts* to adhere as much as extra sensible information regarding outfitting and operation of such a server scheme.

Physical Environment for the Server Box and equipment.

A server is a physical piece of equipment that requires an environment physically in it can operate. For safety, be sure the floor upon which it is mounted by a table or shelves is solid so that no vibration will occur at a level that will cause damage

e.g. A wooden floor raised above ground level will flex whilst people walk over it and is unsuitable, it will require for such to bolt a set of shelves supported by the building frame uprights of the wall to avoid such flex and vibration, as it would locating such a shelf on a wall opposite a workshop to avoid excessive machine audio vibration.

A second feature of *server environment* is atmospherics of temperature and humidity. There are makers of computer equipment that build environment housings for servers to prevent overheating

and with filters on the humidity box to prevent excessive damage from condensation from the atmosphere. This is for most not needed until the atmosphere can contain dust or excessive extremes of temperature, but again can be assessed more to be whether you use an actual server box and board around \$1000(\$AUS) to \$2000(\$AUS) for a single CPU type server.

C. Before installing the Operating System

Think of all the applications that could be required from the administrators and site coders point of view. While user/coders should do their work on their own machine before uploading, it is not a mistake to have their applications present on the server at the administrators terminal in an operation this size (3 to 50 people company approx.).

There are many utilities also, and most found under K desktop for Linux, do not remove any menu items applied in the setup for K desktop or for Gnome Desktop(also: never remove any default Gnome or Desktop parts as redundant or petty, always add more [this last is a rule of thumb but can be quite deadly if any removal occurs]), always respond to dependency messages during setup.

Note: Any properly delivered IT system requires specialists to install and operate, but Anyone should be able to at least get the server OS in alone correctly with this article, ready for successful setup of the server system.

Decide the system of “CGI” and “server(Apache with PHP **or** Jakarta Tomcat[definitely requires a specialist programmer but has massive benefits])” before installing the Operating System on the server box. Tomcat requires a 1.5.x Java2 SDK and installation after Operating System install. Slackware 10.1 shipped with a Java SDK from Sun Microsystems Inc. but any others i do not know of and would be unlikely to be present in the Distro. DO NOT try to use a JRE to operate Tomcat, its an extreme experts job (your unlikely to obtain one that can among the \$100/p/hour Java admin. and programmers that can configure and advise it).

Why PHP is the chosen CGI language(for Apache 2) is because if you can configure and/or program PERL, then you probably do not need to have bothered with this guide. PHP is chosen here for its effective web popularity, *and*, **PHPs' supporting function features/features functions in server side web scripting**. It from this will not do any disservice to the installer(person/organisation/group).

php.ini requires configuring after install and effectively only web programming setup and usage will be assisted to get it all as close to rolling along together.

1. Required Parts *Approx 14 requirements.*

Some of these below **should be bought new**, because they are the brains/vital organs of the network memory and operation. **They are**, the server NIC card , the aDSL modem , Router/HUB-NIC-Firewall-card.

ISP Address allocation(**static** ip address) & **connection(aDSL)**, + **time** (for the router)
You get this from your ISP telephone/inter-net-connection provider company.

Kernel Operating System (if appl.) Any main Linux Distro. Is acceptable e.g. SUSE , Slackware , Fedora , Red Hat...

You can either buy this as a whole from a *large chain book store* or use an ordinary free PC one processor kernel from a promotional DVD Disc in a newsagent sold magazine e.g. APC , or Linux Format (what we are all about here because the job is not so critical and huge and if this operates for us we would want to potentially expand on this).

Note: A “**Packages list**” exists in the main Linux Distributions internet home sites.

“Http Server” Application (Apache 2 or Jakarta Tomcat 5.0 or 6.0)
www.apache.org (Jakarta Tomcat) (Apache 2 http server)

CGI Language binaries PHP or Java2 SDK(jakarta Tomcat)
www.java.sun.com (J2EE) , www.php.net (PHP binaries)

Domain Name - Registration & Mapping (for the www internet ICANN names to machine address mapping database *your registrar company* will place that *after domain name registration payment*)

PC or Server **Box**(**main-board ram, Disc, Monitor, keyboard, mouse e.t.c.**)
last generation X86 P3 256MB-RAM at least or a cheap Single CPU type *Server*.

UPS(temporary power-source, backup power device)
[Pro Office Australian connectors type](#)

IP address (“**the gateway**”) **Router/Firewall – aDSL Modem**(**usually a router contains a modem and less commonly also a firewall card with address aging*) **You buy this from your ISP company usually because of your payment plans “Data transfer speed” rating of service, however some NICs and drivers are multi speed (Baud rate / Bit rate) configurable.**

10/100Mbps “**NIC**” **PCI Card** (if required by the Server box and router [note: Alone to modem will be “**the gateway**”])

(Exclusive line) Available Telephone Connection line (Data, same as audio but agreed to with telephone services provider company / ISP provider)

Telephony service package (rated 10Mbps) [often includes: *ISP Address allocation(static ip address) & connection, + time*]

[3 X RJ45 plug-end network cable] **or** [2 x RJ45 and 1 X **Serial**(COM port/**tty**) cable-connectors]
or [1 X RJ45 and 2 X **Serial**(COM port/**tty**) cable-connectors]



Note: You will be required to buy some power leads if you use a normal UPS power protection system.

Web Design
Custom Programming
Web Architecture
Ongoing Management

2. Costs

These are only “new” prices in AUS dollars. They are a low end price range but with some step to mid range for better pricing cover(more accurate pricing blow out possibility) by availability.

single CPU server **\$1000 to \$2000** (new) e.g. **Proliant ML Hpackard** [servers are cheaper than they were, also check they have all the requires extras, sometimes its only the mainboard and CPU]

BE SURE THE FOLLOWING ARE LINUX COMPATIBLE: talk to the sales rep. Before buying.
NIC card 10/100 (\$40 - \$80) *note: For IP address reasons it is best to **only have one 10/100 card among all the network hardware because of address conflict reasons.***

Consider to buy the router-firewall-4portHUB-adsIModem-10/100 as an all in one and connect it to the server by a serial port.

Router/HUB adsl-firewall-router \$100 approx

aDSL Modem and plan *\$100modem -\$80-first setup - \$50 - \$80 24h-p/month*

UPS power protection \$200 - \$500 (approx. rough)

Domain registration (.com.au type \$130 approx, 2years)

OS Kernel e.g. SUSE enterprise Server (Novell SUSE 10 1yr approx \$500 to \$1500) – or some Kernel offerd by a Linux Magazine promotion DVD, **free (what were really about here in this article)**

* **nominal:** A second hard disc (for **RAID [Redundant Array of Inexpensive Discs]**) identical to the one in the server box \$100 - \$200 approx (**if you do this put any RAID software in that you find during setup of the Linux OS*)

3. Getting Started

A word of warning before continuing, **If your box is new** (either server or PC), be sure of “one particular requirement, before attempting to install anything” , be sure that the main board of your computer and its BIOS chip-set , its board jumpers or switches have been set correctly (particularly) for the CPU oscillation frequencies and that it **was done by a technician and is ready** and inclusive of the BIOS disc setup disc(CD/DVD) has also been setup. ******* If this has not been done, You cannot proceed [For many, that has been done before sale and while you could do it yourself, it is a little bit of a shuffle of choices with the setup from the BIOS-setup DISC, that will require a more educated comprehension of “why” to make some choices].**

First, let us assume we understand *what we require to build this public inter-net domain server* at the level of that we would find if we *fulfilled the list requirements above physically* and were sitting in the room we intend to house the server and components.

- Four major pieces of hard-ware equipment will be present themselves in front of us.
 1. the Server or PC box containing a ready Main-board and disc with CD/DVD ROM, and Monitor.
 2. the Router box, a small box with a quantity of connection sockets that match the male ends of the RJ45/COM(tty) networking cable.
 3. A Large box similar to the server or PC housing box that is the UPS
 4. The kernel Operating System Disc/s for installation with the Application Binaries disc/s

**A quick note of warning here, this document and article is not absolute description of connection process and all the required parts or extras (such as a print server PC e.t.c.) to setup a DNS public*

inter-net server, but a chosen set of required pieces of hardware and parts to attempt to best represent a minimum safe set of hardware to operate a constant 24 hour DNS service.

Assuming a safe place for all of these physical parts is prepared and ready as a permanent location for these items and is sensibly in range/reach of the “exclusive telephone DATA socket” to be used by one of the 3 of RJ45 cables and the same sensibility has been applied between the UPS and the server box and again for the power supply cords to all of the 3 boxes requiring power from the building power source socket.

We can now connect the RJ45 or /COM(tty) connectors cables.

First Cable

One RJ45 network cable (specified as the telephone output socket) from the Router to the wall telephone socket.

Second Cable

**this could also potentially require a joint to the aDSL modem from your ISP provider*

Another from the Server/PC box to the router(as determined in the router documentation) service socket (is **either an RJ45 or COM port type cabling and connectors** M to M or M to F e.t.c as are, **you need to check this first**).

Third cable Is connected from the server/PC box to the UPS (special data line).

Next, UPS (uncommon, unknown as to availability) (Pro Office UPS **Australian connectors**)



Normal UPS connectors (**4 x UPS sockets**) below (what you need, and account physically to the parts of the system here)



Now connect the power cords for the router and server box to the UPS power source output connection sockets, and any middle man hardware requirements (Service provider aDSL modem). Then connect the UPS to the power source (* it would be a good idea to use a voltage surge

power socket board between the power point and the UPS for this).

Assuming all are correctly in place in their respective sockets for their designed and allocated purpose by the manufacturers handbook diagrams we can now switch on the power and start the server box after making any UPS manufacturers specified adjustments after the UPS powers up. Power up the server and monitor. If your PC or Server box is new you possibly will at worst require to place the server manufacturers CD into the drive and restart the server/PC box again to install its BIOS for the main-board chip-set for the first time before proceeding.

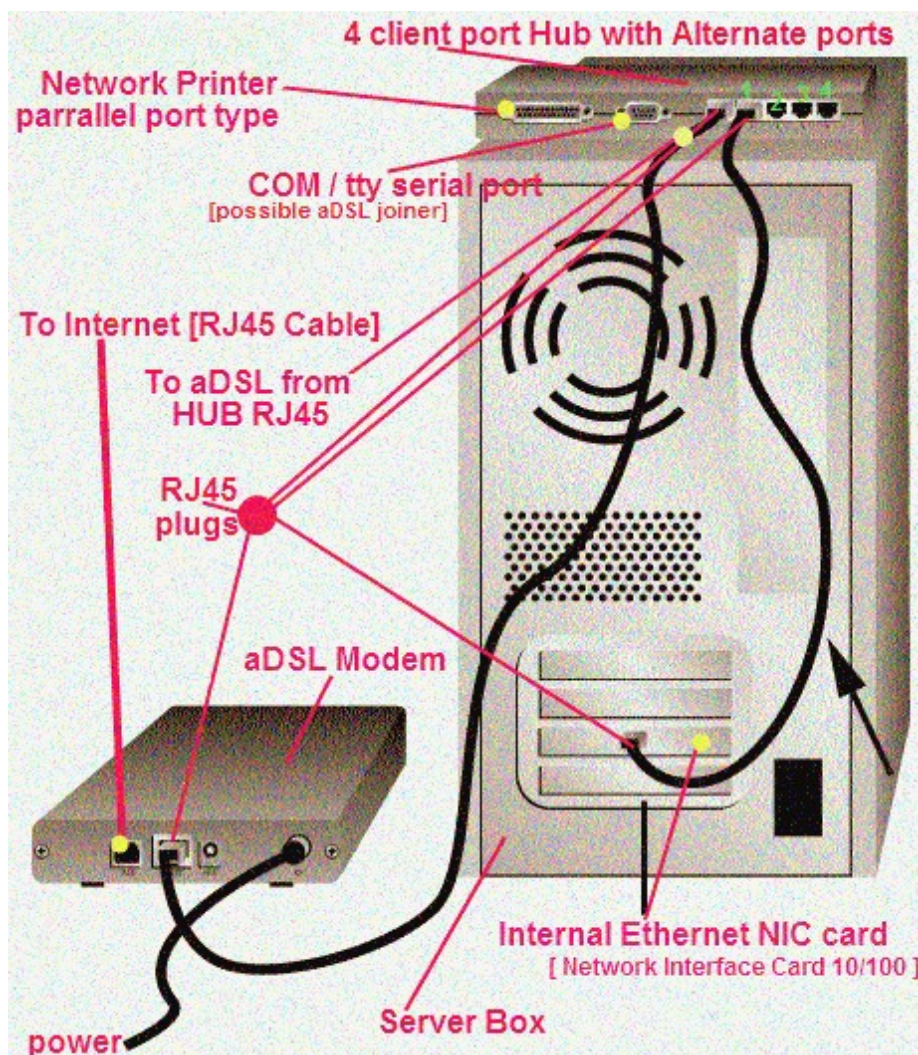
4. Setting Up the Server and Operating System

Assuming your server/PC box "is" ready(*For any server box you use you could require to enter the BIOS section in the start-up screen that occurs always before Operating System boot whether there is an OS present or not, to set the BIOS to allow booting from CD/DVD ROM then restart(power)again), place the first install CD or DVD into the CD/DVD ROM bay and restart the server box.

Before making an idiot of yourself, did you think of everything that should be put into the server.

Linux setups have many choices usually. Previously i told you about **K Desktop** and **Gnome Desktop**, but that's related the *driver/module dependencies* for these programs at install.

One particular point, **use(choose) K Desktop** as the default to boot into. If you want to move around on the inter-net(browse) after install to test the connection to your ISP, use (and probe with if required) the *connector utility of K Desktop called "KPPP"*.



The "**Packages**" section requires you to expand all your choices , or go through each choice set

individually to install more, Not simply in chunks. To remove some pieces such as unwanted servers can usually be done at the main choice heading of the packages list.

When you reach the “**packages**” screen in setup attempt to load every language and disc utility and configuration utility and hardware utility you can find supplied.

Any extra *kernel shells* such as zsh , ksh e,t,c load those too. Also load every **terminal** program you can find too, they operate in some shells only and some have features for that shell that may be beneficial after you know Linux OS better.

A critical factor of loading **packages** is a logistical problem of their operation (**'dependency'**), not particularly relevant from a setup disc *but later when applications are added* by either use of *GNU Make utility* or by a *Linux package manager interface*.

“**dependencies**” is the common term used for “extra runtime libraries(application binaries or extension binaries)” that are often required to be installed with the application in progress of being installed to support its integration and operation, and commonly are at least present by foresight of the Linux distribution manufacturers if not installed and lying in wait.

If such are not present but required, and application install failure message is given. You then would require to find and download the given **dependencies** and install those first. note: their could also be *dependencies for the dependencies too*.

DO NOT load any server if you intend to use Tomcat, Database server such as MySQL daemon however is up to as much as you know to bother with it. That requires restarting the server box after making a special “root” user file for MySQL servers first run admin configuration(NOTE: If you put MySQL daemon in during setup **DO NOT** remove it. Get someone who knows MySQL under Linux to modify it for you).

You could want to also load the supplied Linux NFS (Network File Server) server if your router has a hub and are extending this articles scheme beyond the DNS setup to some extra computers, but if you also are using MYSQL, your NFS will be required to have 127.0.0.1 , while you could need to give MySQL 127.0.0.2 IP address + port (MySQL has a default port). 127.0.0.x(x represents a numeral from 1 to 7 inclusive) is the internal machine loop-back address system of the host.

*Note: **Samba Server** was not decided for use in this article (unless it is installed default as the NFS in the Linux distro you use)because it is a little fiddley and complex to configure.*

Remove Games A serious point about Samba is its tout of compatibility with windows on a network, but whatever your choice, check you can communicate with your windows machines. Its not particularly difficult, simply you must know your NFS will allow that.

Remove Games and other superfluous applications such as chat apps e.t.c.

DO NOT remove text and binary editors, add those if you find them.

Some common applications to help assist you decide what(type) others are installed and colleagues could require to use in emergency for content are next.

The Gimp (Image editor)

Bluefish (programming editor)

OpenOffice.org (office document editor)

StarOffice (office document editor)

download: Netbeans IDE (Java2 [All frameworks and All Application architectures], plus: [HTML , XML , XSL , CSS , C++ , others])

download: MySQL Workbench (from mysql.com dev. Downloads)

download: MySQL Administration Console

download: MySQL JDBC or ODBC or bridge drivers for the PHP or Java2 server app.

K develop (various development editors and tools for Linux and GTK,C++)

These before mentioned applications *require secondary install after first boot of the installed Linux Operating System.*

You should then from this point here be able to go to a [LINUX INSTALLATION run through tutorial.](#)

e.g. (date: --/5/2008)

here at the bottom of this URL's page , list of Linux installation
“graphic imaged walk throughs” “Some Install guides”

<http://www.nicephotog-jsp.net/linux-install.html>

Inclusive in the setup procedure are a number actions that the setup disc will commit to assist finding the hardware recently connected while its power is on and their data lines are appropriately connected also.

The action of the setup disc attempting to find hardware is known as “**probing**”. During Operating install, the setup will be able to install bundled software of applications(if supplied as standard) such as databases, OpenSSL some times ships with distros , PHP4/5, email server(*sendmail* utility) and deliberate machine configuration for the box to be an HTTP DNS Server for public inter-net.

Partitioning the Hard Drive.

After formatting a 256MB size partition called “**the swap partition**”(the swap partition is not elective for most)

Allocate all of the left over drive space to “/”(the symbol used for the root partition), this symbol is known as the “root partition” symbol *akin to “MyComputer” in MS Windows*. Format it in “**ext2**” (extension 2).

Your *primary hard disk* is called “**hda**” and a second would be called “**hdb**”.

An important feature of Linux is that there are always two users for the machine,

1. username: root
2. username: someuser

someuser is **lower permissions** level user that will be whoever/whatever name you wish to give during the install setup wizard.

root though is different, “root” is the username in UNIX operating systems for the **superuser** , the top ranking administrator of the machine **with all permissions**. root has an alias on the **terminal prompts** for logging into the prompt of “su -” (lower case su space dash). Root can go anywhere and do anything it wants including destroying the filing completely as a klutz, so it requires being careful using it.

When you run Linux normally you will be in normal mode using the “someuser” user. This is to both prevent irreversible accidents and to prevent viruses from achieving much the same.

Both “root” and “someuser” will require a **password of letters and numbers**, and as many things inside it will require some **non electronic references for safe keeping**, get a pen and paper to write these down before setup of the Linux OS.

When logging in at the boot prompt, **always have your caps lock off**, thats another standard behaviour of UNIX.

Things to put in if they are on the setup disc for the Linux Distro Flavour.

OpenSSL(You can make self signed certificates and signatures with this or use **-import** signing chains. Note: SSL is alike **https://** but runs through port 443 by default, traditionally/usually)

OpenSSH

Webmin

PostNuke(depends if PHP is with it)

Sun Microsystems Inc. J2EE **SDK** , be sure you have the JRE if it does not but if you require to download it your better getting the Java2-6.0 SDK(if you do not use *Tomcat*) , if you use Tomcat then get the J2EE **SDK6** .

All of the **gcc GNU compiler** and tools(You NEVER want/should to do this yourself or look for dependencies relating it)

...The Network Hardware systematics

During installation and as was mentioned before, you will find that the setup interface will enter a point where it asks you what you want to do relating your hardware. It should find both your router

and your Apache http server . It expects you to tell it some information relating its use in contacting the internet(the **static IP address is for your router**, if you directly connect by the 10/100 and use no other networking, then the **static IP address will be for the machine and the server internal 10/100 card** between the server and aDSL modem[*the hardware joint-filter forTCP/UDP IP signal system and packet data scheme handler of the ISP transmission (frame-signal) scheme service type*]).

With this, there are some small complications to comprehend about receiving and transmitting data to and from the internet.

1st, your router is *not in its technicality a 10/100 Ethernet NIC*. Second, to communicate with the DNS server, you require to assign the DNS server a subnet IP address(the subnet mask) with that you tell the router to send the requests it gets “**from the router port xxxx**” to “**the DNS server port 80**”. That action is called “**port forwarding**”, *the router itself has the static IP address* for the mapping for the **ICANN (Internet Consortium for Assigned Names and Numbers)** database registration. Setting the particular servers port (*e.g. http(80) or mail(25) ftp(21)*) to the type of request occurs by setting up the router driver software.

Port forwarding is the fundamentals and reason for and being of a “router”. A *router* is the single common physical connector point in your joint to the internet for everything, it is a *software controlled bridge-switch combination*.

When **http serving** , **telnet-ing** , **mailing** , **ftp-ing** , **surfing/proxy-ing** , each of these functions usually(and for most) have a *traditional port number assignment* behind the **router machine address**(your address on the internet) that the router must send and receive for *each of these types of service*.

Your router will probably have an Ethernet card embedded in it (particularly if its part of a HUB). An Ethernet card is a data flow control device. Many aDSL plans are 8 - 10Mbps rating and should be sufficient for the bandwidths/speeds required at this level of inter-net server service(remember this is for most only an http server service, *not an intranet or combined to an intranet*, although some of that is mentioned and mixed into this because http serving is often not the primary reason for a business server or inter-net services as part of a small business).

******What you should have decided upon** is the actual way by which you want to connect this http service physically****.

[Depending whether the router has the aDSL scheme built in also, is dependent whether to place an aDSL modem in front of your router.]

Either:

With a **serial port COM/tty port to a router directly**, of which the 10/100 card could be present in the router,

or

by **10/100 card in the server(not the serial port COM/tty) to the router(no 10/100 in the router)** directly

or

by a **combination router-HUB-10/100-firewall** from the server boxes' COM/tty to the HUBs' server input port.

or

by **directly using the RJ45 cable to an Ethernet NIC 10/100 card in the server to the aDSL modem then to the phone socket** , using the **the server machine and its card assigned the inter-networking address** that the router(not used in this) would have had.

5. Post (after) Operating System Install

Familiarising and Customising the Server box.

After install and the first restart, the server operating system will be in *first boot mode*, and the various applications , directories and supplied configuration utilities you should familiarise yourself to, particularly the HTTP server configuration equipment of there is a link to a configuration GUI for

the http server on the start menu (NOTE: Apache.org Jakarta Tomcat 5 or 6 and the Sun Microsystems Inc. Java2 SDK will not be present at this time, you install that after first boot), as much also its actual location in the file-system of its binaries, documentation and document serving host directory zone (*htdocs* or *httpsdocs* in an Apache server usually, for the Tomcat its in *webapps/docs*).

As stated before, you need to install all the required users emergency applications completely so navigate to their position on the “start menu” and its section and install them.

If you need to write anything, use KWRITE not KEDIT, kwrite is effectively a supered up comparative version of MS Windows notepad.

Use the **internal browser of Linux, it's “Konqueror”**, it sees html and some javascript, but is effectively the default window file browser also. For proper web browsing Firefox will probably be installed.

Among other things you will require to write down will be the **names of programs folders locations** and **environment variables** , and their **value settings** inside the special file called **/etc/proc**.

Another interesting and useful point of *UNIX and Linux*, is that “**sendmail**” is a **utility** run from the command line(terminal prompt) originally, but now also, is **configurable to a server application installed in the OS, it will have an inexorable link with the servers domain name** now. Its configuration file can be found in **/etc/sendmail.cf**.(back to it)

The thing to remember here when connecting a machine to the inter-net is that while you and i can recognise a name such as www.somedomain-somewhere.com.au , a machine cannot. So the **IP address(4 digits of 4 sets up to 3 digit length each, not exceeding 255)** is the important part of connection to the internet if we had had the server operational, the machines do not and cannot actually communicate using the **human readable** domain name, they must use **networking IP address masks**. That is why the **10/100 card** must have its software given:

A- the ISP Gateway machine address

B- the ISP address of the server behind the card on *the cards' network* , known as the **subnet mask**.

The card is bound to the server by a **static IP address(while Dynamic Host Control Protocol [DHCP] is possible, for purposeful http servers, it is not acceptable for a business purpose)**. When a request is made over the inter-net the *browser address-bar call name* is used to look up the machine address at the serving ISP(Internet Service Provider).

-----[About /etc/proc environment variables disc partition /usr/bin /usr/local/sbin]-----

6. Configuring and/or Installing Server(s)

Apart from some distro's of Linux having both an “*HTTP Server*” and “*BIND*” GUI to service the configuration of your server (and such are available for MySQL) there is more than the **httpd.conf** and many other ***.conf** files that will configure and operate Apache.

One important feature of DNS name servers is there are usually 2 that are named in **BIND** as “**ns1**” and “**ns2**”.

[Important note: Downloaded Apache setup and “*mainly integration, is far different internally*” to an OS integrated version while it will seem more complex to use the Linux OS integrated version, the control we will have and services are much more effective and efficient. **THIS IS WHY WE ARE USING THE INTEGRATED SYSTEM. It is more slogging it out doing configurations than actual complexity and will bring better safe efficient management since we want some flexibility and scalability of the services from it. One such flexibility you will probably want is to be able to “multicast IPv6 and by HTTP 1.1” from the server as both the public serving DNS name server host and to the internal**

network as localhost 127.0.0.1:8080 server. To do this will require the full suite of ARPANET and BIND 8 or 9 (as is supplied) configuration files, [Note: during setup , by some Distro/Flavour setups, you will potentially be asked if your machine is to be an HTTP server for extra levels of dependencies to be suggested for integration]. When supplying .html or other files in this system, never use the “human readable domain name address” in any of the documents links [markup syntax], that should be added dynamically or be inserted at by the client browser(software) end by “implication”(implication at the directory level the document came from also) of the host-name. Other non related domains can be inserted]

Inside the `./Apache-name-version/bin` are the executables for *starting* and *stopping* Apache http server, and others integral, to, *this must have set of services*, these also have **command-line** uses.

Below is an output dump of an important part of configuration and use of Apache add-ins. These are the **modules (*.so) loaded at startup** listed by a command to the binary “**httpd.exe**”.

(Sorry about the fact its in windows at this time)

```
C:\Apache2.2\bin>httpd -t -D DUMP_MODULES
```

Loaded Modules:

```
core_module (static)
win32_module (static)
mpm_winnt_module (static)
http_module (static)
so_module (static)
actions_module (shared)
alias_module (shared)
asis_module (shared)
auth_basic_module (shared)
authn_default_module (shared)
authn_file_module (shared)
authz_default_module (shared)
authz_groupfile_module (shared)
authz_host_module (shared)
authz_user_module (shared)
autoindex_module (shared)
cgi_module (shared)
dir_module (shared)
env_module (shared)
include_module (shared)
isapi_module (shared)
log_config_module (shared)
mime_module (shared)
negotiation_module (shared)
setenvif_module (shared)
```

Syntax OK

*Although the version used here contained a lite version of SSL embedded with it, it was not named and thus not loaded at the time. The ssl .conf file had not been edited by me either, not that i had any scripts for it to run(that being its principal joint into the system apart **https:** protocol) The SSL binary supplied with it can create and sign .crt certificates after creating private/public key pairs (PKCS #12), or imported from .crt to create a certificate chain, or make email digest messages or signatures.*

You can **download a list of commands** and their switches/uses taken from a windows prompt here(note: This version appears somewhat experimental):

<http://www.nicephotog-jsp.net/2.2.8-Apache-bin-commands.txt>

One of the best lessons of installation for putting in applications in Linux i found, was the **read-me file for MySQL server 4 alpha in the tar.gz download**. You learn two things. 1. A **daemon** is a **UNIX background program**, 2. a special problem occurs allocating permissions to binaries and application program folders of using the “chown” command with flags as **root** to assign operation allowance to “**someuser**” if startup goes directly to the non root user.

A quick note also about getting a MySQL server operating in Linux, download the MySQL Administrator browser from www.mysql.com and connect by localhost address numerics / port TCP/IP(not unix socket) after joining it correctly to the application MySQL program folder.

Much of Linux and UNIX installation of applications is done with a **utility** program called “**make**”, “make” is a system of issuing commands from special scripts called “**makefiles**”, these makefiles are interpreted and executed by the “**make utility program**” upon a C/C++ compiler embedded into the OS (in Linux usually “**gcc** the GNU compiler”).

Never voluntarily(by update or special install downloaded zip [by customised non-setup install context]) put one of these in unless you are a programmer it requires an immense quantity of time, However though, *if you can see your way clear to it you should*, **put in the gcc GNU compiler “and all its extras” from the setup disc.**

Note that when servers such as Apache or MySQL are **installed by the OS setup disk**, it has a different directory structure than the server when setup from its .tar.gz download from apache.org or mysql.com, and, the parts are in two or three different directories under /usr , /opt or /etc .

Also when committing this, they should be setup as a “**startup daemon**”, meaning when the computer is booted into the linux they will automatically start after login. These servers will also be running in non root mode, but will need to have their permission to do such assigned to the daemon in the configuration utility after install.

All inter-net http document servers ports on the internet are set as 80 by tradition and default. On localhost servers they are often domain-name.com:8080 or more usually localhost:8080 or 127.0.0.1:8080 , and requiring the port(the **:8080**) in syntax supplied with each http request.

That must be **listed in the server configuration as “80”**, because *it then does not require to be present on a browser/link url to call the location in the server.*

The MySQL daemon should be running as a localhost with a 4 digit port number(probably the MySQL server default *http* port :3306).

Some things that servers worry about , **MIME types of files**, inside the Apache server configuration file called **httpd.conf**(for Apache 2) or inside **server.xml**(Jakarta J2EE Tomcat(an Apache *special variant*) it also has a **web.xml** configuration file for the server) there are a number of words on lines to look for “mime” and “accept” are two of them relating file types to serve.

Going back to what i mentioned before, *a server is in the context of software* an application that *relies upon/is based in* the Operating System to support its operation making it an application. A feature of this interactivity requiring a compatible **environment** is a *reference with a unique name* that is a permanent(usually and for most here in this document) setting for both the Operating systems' reference and for the Applications it has operating(applying).

The **reference with a unique name** is called an “**environment variable**”, and *in the Linux OS* many of these reside in **/etc/proc** . The “**proc**” file is one of the main Linux OS configuration scripts to run at boot/startup. It contains the “**PATH**” environment variable, this one is used to retrieve all the possible places major default **.bin** application programs could be if named to variable as important in its configuration.

Some examples of custom inserted (editing of /etc/proc) environment variables are:
MYSQL_HOME = /usr/sbin/mysql_server

```
MYSQL_TCP_PORT = 3306
JAVA_HOME = /usr/bin/Sun_JDK_150_6/Java_SDK
CATALINA_HOME = /usr/sbin/Apache-Server-http/bin
TOMCAT_HOME = /home/theuser/Tomcat5.5
```

When an environment variable is assigned it is then handed to the machine using the **export** shell script keyword.

The export action allows the reference to be recognised when named on the command-line. The command-line usage is the difference between these types of variable and the path variable. PATH only uses *the set of* filing location paths to sort through if it gets a name on the command-line. PATH is iterated until *a/the first* .bin executable is found matching the name given to the command on the command-line.

7. httpd.conf the Apache http Server configuration file

While there is a GUI to do most things for configuration if you installed the server in *using the Linux Operating System setup to integrate it with the OS*, it will help to explain it and some of its more vital contents.

One point here is that of PHP. This language while it acts the same as PERL is not configured that same way PERL is configured. PERL uses a cgi wrapper system through a redirection to a **directory** part of the server called **cgi-bin**(traditionally though it can be setup much similar to PHP for call triggering). In the Apache httpd.conf a number of lines are used to specify that it will require redirection to it and the location of the PERL perl.bin main program executable. Much the same occurs with PHP, but the aim of using PHP has some special differences, and later some special differences again relating the file **php.ini(PHP'S configuration file)**.

PHP requires the httpd.conf to recognise its use for two major differences as CGI, 1. It requires to serve files by a PHP mime-type (**.php** usually), but also, outside of the CGI bin directory and in fact any directory of the servers http documents directory(note: it is possible to do that with PERL but its not traditional).

Here its best to tell you of the “#” symbol in the httpd.conf, that symbol is generally found at the beginning of a line. **It comments out the line from the configuration files' usage**, and when some discriminations are added at setup or tuning the server, it requires uncommenting the line(s) to use them.

This should not be allowed to be confused by you with the “#!” *shebang* found at the top of either PERL or PHP(if required to be present) *scripts* as the first line(or in Unix Shell scripts). While scripts use that symbol as a “one line commenter” all alike with the httpd.conf and other configuration files inside Apache Server you have moved a little to fast for yourself if that(shebang) is found present.

As stated before, much of what we are attempting here is to setup an http inter-net server with as little actual knowledge as possible, but these types of concepts requiring to explain here to prevent mis-configuration and give you some chance to survive if anything occurs as a problem.

```
-----[
gnu make make,make install, ./configure , make bootstrap , --with target= ,Addresses, disc zones e.t.c. :8080 and duplex :80, :443 e.t.c.
--with SSL e.t.c.
Config. The perl.bin and services , main – install + configure PHP to the apache in the httpd.conf , httpd.conf tags and permissions ,
management of direct users and .htaccess , .so start in correct order
/etc/sendmail.cf , /etc/ftpaccess
OpenSSL commandline , Self signed Cert .crt .cer – self CA , PKCS #0 - #12 keystore types
ifconfig , lspci , tracer , ping
]-----
```

8. php.ini PHP's configuration file

<http://www.php.net/> PHP binaries runtime and documentation

One particular feature(more a requirement) of PHP is its configuration file called **php.ini**. The file is originated in PHP's requirement to 'supersede PERL' as a shell and scripting language by point of integration with web servers as *its name more than suggest*(PHP – Pre Processor Hypertext). Many of the settings of the **php.ini** file are “default settings”, they do not require uncommenting because they assume the folders location and special internal folders locations, or are machine

related settings by assumption the machine has enough power and that the technicians of the php binaries have produced settings inbuilt that are sufficient.

However, there are limitations to what can be pre-known of the final install, and that's where choosing some settings after install commits perfection, but, another feature of Linux (UNIX derivative OS) is a special utility called "**GNU Make**". With "make" it is possible to change features of the installation before install by specifying the retracted or added features desired on the command line when the setup of the PHP binaries occur.

Generally you can find the allowed "*names of features*" (that can be added or retracted) in the **INSTALL** file and the **README** file that are often present in UNIX installations (as many or most UNIX installs) that use the "make" command utility system.

Usually inside the distribution (application pre setup) folder is a special ".sh" shell script (or unnamed by extension because it operates almost as a "make" command *by name*) called "**configure.sh**". Configure is the first script to be run and can have either default settings or added *switches* to customise and refine the binaries pre compilation process.

Common commands and sequence for a "make" setup are (roughly of hand):

1. **./configure [switches here rarely]**
2. **make bootstrap [switches here]**
3. **make [switches here]**
4. **make install [--with=whatever-specified --target=/somewhere switches here]**
5. **install [--target=/somewhere switches here]**

Be sure to always read through the **README** file and the **INSTALL** file.

It's also not a bad idea to look in the **makefiles** and the **.sh shell scripts** themselves at the settings too.

The "**php.ini**" file controls many "**settings**" (constant or dynamic) of the interaction of both web pages and requests/responses of the Apache web server. Initially, it is the server that is configured to use PHP, not PHP altered to use the server (if you didn't realise its significance), the server is configured to respond to **pages with the extension of ".php"** by sending them through the php.bin (usually called php.bin, there are other main binary names dependent the distributions' version) executable.

Inside php.ini there are many explanations of the settings, but will only be relevant to a developer or administrator with experience of programming and the machine, but, however there will be some explained here and some you can guess.

These following settings are important and a good example of the type of setting diversity inside

php.ini:

```
implicit_flush = Off  
safe_mode_protected_env_vars = LD_LIBRARY_PATH  
open_basedir =  
realpath_cache_size=16k
```

Next are more important and as many are more the admin./developers decision to reset - set

```
max_execution_time = 30  
max_input_time = 60  
memory_limit = 128M
```

Note: that php.ini settings can be controlled inside scripts by a special function for the duration of the scripts activities.

```
;  
; Temporary directory for HTTP uploaded files (will use system default if not  
; specified).  
;upload_tmp_dir =
```

FOPEN WRAPPERS (some settings for opening files)

```
;  
; Whether to allow the treatment of URLs (like http:// or ftp://) as files.  
;allow_url_fopen = On
```

; Whether to allow include/require to open URLs (like http:// or ftp://) as files.
;allow_url_include = Off

; Define the anonymous ftp password (your email address)
from="john@doe.com"

; auto_detect_line_endings = Off

Because of the placement of the binaries and SSL not carried with the distributions of php for Linux the **.so module** must be built an output to its target location.

when it is in place it is configured in php.ini like this "**because there is a specified /ext directory for php(for modules .so) you require to configure in php.ini**"

extension_dir=".I" note: this location appears to be unexplicit but is its symbolisis for the default modules location.

openssl-0.9.6 example:

extension=nsopenssl.so

To make and put it into a linux **"/wherverto/php/ext"** directory...

\$./config --prefix=/usr/local --openssldir=/wherverto/php/ext
normally untarring it will be found in **/usr/local/ssl "unbuilt"**.

;extension=php_ssh2.dll is a windows example

With distributions of PHP or any other alike applications, it is not possible to know if modules and "extras"(as are called in PHP distribution) will be supplied and **supported** each update and change of version.

One particular point to note of PHP is its "**ext**" directory for extension modules and its "**extras**" directory that contains a special joiner utility for SSL. Included in the "extras" directory is a file for assistance to setting up SSL throughput with PHP. The file is aptly named **README-SSL.txt** and contains configuration information for environment variable setup. When PHP is in operation it will require to know the location of the **openssl.cnf configuration file**.

Note that for OpenSSH and for these to be added to apache, the same type of activity is required to install them in apache.

9. Installing a Tomcat JSP/Servlet Server and a JDK/SDK

<http://tomcat.apache.org> (Tomcat server documentation and downloads)

<http://java.sun.com/javase/downloads/?intcmp=1281> (Sun Microsystems Inc. **JDK 6 Update 6** with Java EE)

Something that gives using a Tomcat server have a great quantity more meaning...

<http://lenya.apache.org/> ("Lenya" Site Content Managemet System [Tomcat / Java2])

<http://lenya.apache.org/screenshots.pdf> (Screenshots PDF)

<http://java-source.net/open-source/content-managment-systems> (List of Various Java CMS)

Before explaining Tomcat server here, note that what you would want from it is not the full potential of the machine.

You require to make it alike any first or alternative hosted site you have with the minimum requirements to have a sensible business web server that you can add more services to and grow in its processing power and abilities as required through time.

To do this simply trust that you only require the first generation of JSP's EL language available, the JSP level in the server to be JSP2.0 and single thread model servlets. This means that whether

you use the latest Tomcat server or a Tomcat5 these before mentioned are all you will use. While you possibly would use AJAX it is not relevant except as a choice. What you truly should not use but would be able to, is JSF(Java Server Faces) or Struts or alike. While these are Web-GUI(Graphic User Interface)systems that bolt to the internal processing systems of both the server and the web application running on the machine for the Java J2EE server systems, they are too complex for what you want to achieve here, and a Java server requires more than a good amateur Java web programmer (generally, because there are some, although, do not bet about it) to program a web application.

It requires some type of experienced Java Web Programmer. Bringing this to another great and purposeful feature that will also be explained a little here but not intended for use at this level of stage is the EJB(Enterprise Java Bean) bean. This (EJB) is far beyond having a good server with room to expand operations, It is a complex beastie as are JSF beans.

Tomcat is being used here in this document only because Java J2EE servers and the Java Language framework/API systems make a sensible choice because it contains "everything in a programming language for any purpose inside or outside of web development" whether this can only result in someone paying for a programmer to have this type of server(quite likely to only be it).

The Tomcat Server is a special type of "**framework driven**" (to coin a phrase) server that uses a special engine(apart from the context of a server as software integrated to the OS) called a "**Virtual Machine**"(the Virtual machine is integrated to the OS). There are many types, brands and contexts of the word "**virtual**" in computing but this one is relating to a system that is *first installed into the Operating System* and then used alike other applications that are unseen daemons but is also *programmable by a computer language* and this last point being its main purpose.

In this system although, the virtual machine is the (**JVM**) **Java Virtual Machine** that was created as an in house solution by **Sun Microsystems Inc.** as a language originally known as "**Oak**".

Later it required a more commercial name and then was given specifications for its creation at assembly/machine level p/ versioning by other commercial companies and was publicly known as **Java**.

Java as a language is used to program the integral framework systems Tomcat uses in its method of use as a server, and the Java Virtual Machines' (**JRE**) **Java Runtime Environment** for most operates the server as it does its duties. But, When the server is in use it will require at some time to commit "**compiling**" of the (**JSP**) **Java Server Pages** pages and/or **Servlets** (mainly when they are updated/changed).

This action of compiling can only be carried out by a (**JDK**) **Java Development Kit** or (**SDK**) **Software Development Kit**, and thus an SDK or JDK is installed into the Computer OS to operate the server, *not a JRE*.

Tomcat Server is a derivative of the **Apache Web Server** and part of a project originally called "**Jakarta**". It is a **J2EE** specifications compliant server. (**J2EE**) **Java 2 Enterprise Environment** is the name given to the Sun Microsystems specification for **Java2 web server framework** programming and usage systems. It is incorporated now into a new **J2EE** project system known as "**Glassfish**".

While it is possible to get versions that will run in Linux by untarring (unzipping) straight onto the disc location you want, it does require knowing some configuration points and pieces to get it operating successfully.

It is *not unlike* the above explained relating OS configuration files to integrate into the system, but it does also require the JDK or SDK properly installed and configured before installing and configuring the Tomcat server.

As stated before, Tomcat is effectively an Apache Web Server by fundamentals of its construction, and it does also have Environment Variable declarations of its origin such as

CATALINA_HOME(.../Tomcat_server_version_top_folder/bin).

TOMCAT_HOME(The top folder of Tomcat server in your OS) is its application folder setting.

As you can expect by this point the name of the top location folder for an SDK , JDK or JRE is

called the **JAVA_HOME** as an environment variable. Another feature of this system of configuration is the the location of the .bin files for e.g. java.bin , is placed on the **PATH**(OS command line relevant folders to search for the executable named on the command line) environment variable in /etc/proc as it is in MS Windows of the variety with the autoexec.bat configuration file.

These environment variable setting are often used by one of the J2EE specified services called **(JNDI) Java Naming and Directories Interface**. The JNDI interface is a method of path-filename substitution nomenclature for the file and directory locations by *aliasing directories' and files' names*. This action is purely for use in configuration of the server and for the use of the Java programming language (not related of any way to ARPANET or IANNA naming). The purpose of this inter-twined architecture of directories , files and **unique names** is to make it easier to commit *robust and definitively clear naming* for runtime operation through better and easier programming technique.

The server has two main J2EE compliant configuration files.

(note that at the top of these XML files is either a DTD or Schema *declaration* that will vary by server versions and application versions)

1. **web.xml** (application configuration)
2. **server.xml** (*The Servers'* configuration for ARPANET and IANNA type things)

While these are/could (respectively) be found in many other parts of the server, *the two* configuration files themselves (of the server) are *unique to their job and location*.

Mainly you will find only web.xml files present at various directory levels, but these others are for applications that have been *pre integrated* into the Tomcat server such as WEBDAV.

-----[various xml elements and allowances , decomment for servlets and CGI add ins , explain context.xml , meaning of databases in server conf context]-----

10. [Libraries] J2EE, Java Mail API jar and JSP, JSF, Custom Tags, Beans

Unlike using **sendmail** (the UNIX command line utility) , **Java mail** is packaged as a special set of class file libraries in a .jar package archive (almost the same as a zip codec archive). Whereas the **sendmail** mailing program for SMTP is part of the UNIX OS, the **Java mail** is part of the **J2EE specification** for java servers and is one of its API(Application Programming Interface) frameworks. The **Java Mail** framework contains only the *base class file libraries* for for the SMTP protocol (*a binary protocol*) and the programmer must then create the final servlets/jsp's service for the site server, but, the libraries *contained in the JavaMail systems' .jar* are for **everything** including construction of a major mail SMTP server (newsgroup e.t.c.) including, handling mail box permissions and folders, attachments, accounts and their various limitations , alike any of the major companies or internet companies such as Yahoo! Or Google mail.

There are two places it is possible to place the JavaMail .jar in the Tomcat server, much the same as any of the J2EE framework libraries.

1. TOMCAT_HOME/**common/lib** (The best place for them all unless otherwise instructed)
2. TOMCAT_HOME/webapps/**my_web_application_war_name/WEB-INF/lib**

The first above, is in a special location allowing all web applications (.war or .ear packages) running on the machines' server program (Tomcat) to access the libraries inside the .jar framework binaries package, The second only allows the application it(the .jar) resides in to use those libraries (for .war , .ear is likely to take up all of the machine and require all libraries in ../common/lib for more efficient use(programmer) and access(services – applications)).

With other “Java2 J2EE frameworks” also, the system is much the same for them unless specified. As a quick example of “unless specified” JDBC (Java DataBase Connectivity) and (JNDI(Java Naming and Directory Interfacing) Data source “Connection Pooling”) is a good example. What we will have is a *required DB connection driver .jar* file from the Database vendor e.g. MySQL or Oracle. The .jar will then be placed in the *common(best for server JNDI Connection*

Pooling) folder system or the *WEB-INF* folder system. But when we use the *JNDI Connection Pooling*, you require to make available the driver to the server directly by integration to operate with the JNDI system .jar of the server because you will use JNDI server services, not write your own connection pool(although if your a heavy duty J2EE programmer you well could, or if you simply wanted to chunk that along to writing your own eventually).

A side effect of using a “*server managed*” service is that the *context.xml* of your application(.war) will require configuring for the association name, as much the server.xml has to be adjusted for the configuration of the *named naming service Connection Pool(JNDI connection pool admin section)*.

->Fortunately, not much reconfiguration is required in Tomcat because it has a JNDI configuring(adverb/adjective) interface administration page and the most you may need is a server software restart after naming the data source(in context.xml) and nominating the connection quantities(configuring the Connection pool in admin console of the server JNDI).

JSP(Java Server Page) and JSF(Java Server Faces) .jar libraries are placed much the same way as the JavaMails'. Servlets are for most , configured in your custom application folder web.xml. Newest Java servers conforming to the J2EE specifications do not(should not) require the JSP 2.x system installed and run EL(JSP page “*Expression Language*”) and JSP usage assumed.

<http://www.nicephotog-jsp.net/>

(custom tag class configuration tutorial for Tomcat 5.0 BY Login Only)

---[.tag]---

Finally, **Custom Tags** are another great feature of java servers, and by principle *are what a JSP library or the JSF library is exactly*, They are “custom tag libraries”, the heart beat of JSP.

To use any of these requires having special configuration files that register/bind each tag in a library to the application or machine. Its file extension is “*library_name.tld*” and is stored in the WEB-INF directory itself. You should *copy these from any .jar files to the WEB-INF directory* before starting your application (there are various) so the server does not overstep attempting to load them to memory while it is in the process of unloading the .jar's structure from WEB-INF/lib (for much it usually is ok and not required to copy them).

True Custom Tags are a special association *between a mark-up tag(similar to an HTML or XML tag) placed in a JSP page and associated to a class file in WEB-INF/classes with the same name as the JSP page embedded markup-tag*.

More complexly *but ignore this next statement for most*, Any classes found in WEB-INF/classes directory are “servlets” effectively, and that is where the server unpacks the .jar libraries to from WEB-INF/lib of a custom application whether they are JSP or JSF or Struts .jar libraries of available frameworks (note: /common/lib and common are special server folders designed to commit the task of the individual WEB-INF custom application systems by an alternate method of configuration).

Servlets (the metaphorical akin to CGI scripts) are kept in WEB-INF/classes or subdirectories.

Onto another standard feature of the Java Server, “Beans”, basic standard beans or EJB(Enterprise Java Beans) beans, they always are kept in a sub directory of the WEB-INF/classes directory.

Beans are a little different to servlets or custom tags for their operative points of service. These class files are designed to commit “storage of data and operations on data during surfing sessions of client nodes” if connected to the Java server site.They also can communicate with some of the server program for environment variables and for the basic bean were designed before EL language were existent but (more aptly to say) were designed inside the standard J2SE application programming language system.

When the Java server became more effectively developed for as a web server solely, a new type of bean was created and introduced with the J2EE specification. This is the EJB(Enterprise Java Bean). The EJB has a huge quantity of server and web application interfacing ability for program, configuration, communication and control allowing it to more easily communicate inside or across the frameworks libraries operating in a server that have been developed recently in past 3 or 4

years for J2EE compliant servers specialist usage such as SOA (Service Oriented Architecture).

other information and articles related

<http://www.linuxformat.co.uk/modules.php?op=modload&name=PNphpBB2&file=index>

Linux Format Magazine forums for “!help” relating Linux OS

GET A LOGIN ID &PASSWORD

Linux Forum at www.nixcraft.com

<http://www.nixcraft.com>

<http://webscripts.softpedia.com/script/Security-Systems/OpenSSL-PHP-Webfrontend-37300.html>

<http://www.openssl.org/source>

<http://www.openssh.org> (with sftp and sshd)

BIND

<http://www.isc.org/index.pl>

<http://www.yolinux.com/TUTORIALS/LinuxTutorialWebSiteConfig.html>

DNS nameserver registry

<https://www.domainregistration.com.au/>

Networking Configuration

<http://www.builderau.com.au/program/unix/soa/Configure-Linux-networking-manually/0,339024638,320265871,00.htm>

<http://www.debian.org/doc/manuals/reference/ch-gateway.en.html>

Hardware Products

http://www.myshopping.com.au/PG--1_Computers

<http://h18004.www1.hp.com/products/servers/linux/suse/oneprocessor.html>

Applications

<http://www.sourceforge.net>

<http://www.mysql.com>

<http://www.php.net>

<http://postnuke.com>

MultiCasting addresses

<http://tomcat.apache.org/tomcat-6.0-doc/config/cluster-membership.html>

<http://www.zeroconf.org>

<http://www.linuxhq.com/IPv6/linux-ipv6.faq-3.html>

<http://www.debian.org/doc/manuals/reference/ch-gateway.en.html>

<http://tomcat.apache.org/tomcat-6.0-doc/config/cluster-membership.html>

journalings for assistant sites to go here

Nicephotogs (Script/Markup editor) SUcommanderXer Last Pre Beta +... (SCREEN SHOT)

<http://www.nicephotog-jsp.net/SUcommanderXer-last-pre-beta-screenshots.html>

<http://www.nicephotog-jsp.net/SUcommanderXer-last-pre-beta.zip> (Linux/Win)

<http://www.nicephotog-jsp.net/SUcommanderXer-last-pre-beta.exe> (Windows 2000/XP/Vista)

<http://www.nicephotog-jsp.net/nicephotogs-site-blaster.js.html>