修平科技大學

資訊網路技術系

LINUX系統管理 PHP TQC證照

組長:BN97033 王偉名

組員:BN97012 曾子恩

BN97045 林鈺閔

BN97110 吳冠賢

指導教師：麥毅廷

評審老師:_____

_____

_____

中華民國101年6月

修平科技大學

資訊網路技術系

LINUX系統管理 PHP TQC證照

組長:BN97033 王偉名

組員:BN97012 曾子恩

BN97045 林鈺閔

BN97110 吳冠賢

指導教師：麥毅廷

中華民國101年6月

目錄

摘要

TQC 是全球首推為華人企業競爭力與個人職涯規劃量身訂作的電腦技能認證， 電腦的技能認證不應只是軟體的應用與操作，我們使用軟體的目的在於提昇工作效率與品質，在知識經濟的時代透過 e 化的過程讓個人的專業得以呈現與累積， 由於工作的不同職業的不同專業的不同當然在電腦應用上與深淺度上自有不同， 這樣的不同非一般國內常見的各別單科認證所能替代，更重要的是，一個職業的專業人士，應該具備的電腦技能應有的素養與能力將代表著工作者的工作品質與效率，甚至是資訊溝通的能力， TQC 為華人企業與個人應運而生。

TQC的認證不但能讓有志從事該項職務的人員掌握學習的方向，對求才企業也提供了更快速、更客觀、更簡化的人才徵選程序。目前包括7-11統一企業、鴻海等知名企業，皆已採用TQC做為晉升評估標準之一，TQC是全球首推為華人企業競爭力與個人職涯規劃量身訂作的電腦技能認證，已獲得國內知名企業機構認證，作為企業徵才時之參考標準。TQC人才認證可以讓 企業確保其員工擁有達到相當水準的現代化戰技-電腦技能，經由這項認證可使企業明確了解員工、或求職者確實具備可以獨立作業的專業技能。

第一章 TQC證照簡介

1. TQC認證由來

TQC為Techficiency Quotient Certification的簡稱，為財團法人中華民國電腦技能基金會依員工職務所規劃之整合性認證。這項認證是經過詳細調查、分析國內3,500家企業各職務用需求，確認從事該項職務應具備哪些電腦技能，再對所有電腦技能測驗項目重新歸類整合而成。

2. TQC怎麼來

企業人才技能認證(TQC）是財團法人中華民國電腦技能基金會針對企業用才需求，所提出來的一項整合性認證。這項認證是經過詳細調查、分析各職務工作需求，確認從事該項職務究應具備哪些電腦技能，再對所有電腦技能測驗項目重新歸類整合而成。不但能讓有志於從事該項職務的人員掌握學習的方向，對求才企業也提供了更快速、更客觀、更簡化的人才甄選程序。

LINUX系統管理認證

Linux是一種自由和開放源碼的類UNIX操作系統，使用Linux核心。目前存在著許多不同的Linux發行版，可安裝在各種各樣的電腦硬體設備，從手機、平板電腦、路由器和影音遊戲控制台，到桌上型電腦，大型電腦和超級電腦。 Linux作業系統也是自由軟體和開放原始碼發展中最著名的例子。只要遵循GNU通用公共許可證,任何人和機構都可以自由地使用Linux的所有底層原始碼，也可以自由地修改和再發布。嚴格來講，Linux這個詞本身只表示Linux核心，但在實際上人們已經習慣了用Linux來形容整個基於Linux核心,並且使用GNU 工程各種工具和數據庫的作業系統，通常情況下，Linux被打包成供桌上型電腦和伺服器使用的Linux發行版本。一些流行的主流Linux發行版本，包括Debian（衍及其衍生版本Ubuntu），Fedora和openSUSE等。

此認證項目相關人員：

職務別	電腦技能需求
專業 Linux 系統管理工程師	• 電子商務概論(EC3) • Linux 系統管理(LX3)

PHP 認證

PHP的全名為Hypertext Preprocessor，它是個被廣泛運用在網頁程式撰寫的語言，尤其是它能適用於網頁程式的開發及能夠嵌入HTML文件之中，它的語法和C、Java及Perl等語法相似，且學習起來更容易上手。PHP的目地是為了能使網站開發者可以快速地撰寫動態網頁。

此認證項目相關人員：

職務別	電腦技能需求
專業電子商務應用工程師	• 電子商務概論(EC1) • 多媒體設計(FL1) • 動態網頁程式設計(AN1)/(PH1) • 網頁設計(DW1) • 網頁標記語言(HT1)
專業互動式網頁設計工程師	• 電子商務概論(EC2) • 資料庫管理系統(AS1) • 動態網頁程式設計(AN2)/(PH2) • 大型資料庫管理系統(MY3)/(SQ3) • 網頁標記語言(HT2)
專業網站程式開發工程師	• 電子商務概論(EC3) • 動態網頁程式設計(AN2)/(PH2) • 網頁標記語言(HT2) • 軟體開發(B2)/程式設計(JV2)

第二章 LINUX-系統及開關機程序

2-1.作業系統基礎知識

一.完整的作業系統構成要件:

1 核心(管理所有的硬體資源)

電腦硬體、檔案系統、行程、記憶體及網路管理...

2 操作介面

Shell、GUI…

3 系統程式(API)

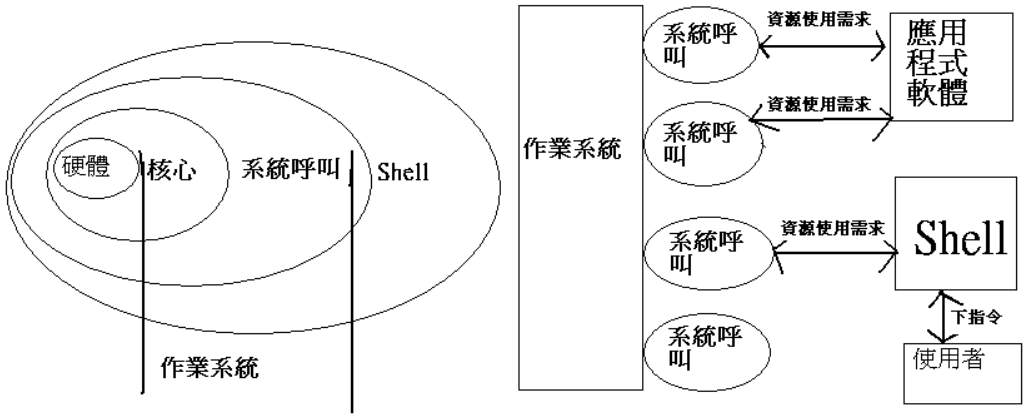
程式向作業系統提出”系統資源需求”時，所使用的”

功能函式”

4 應用程式

瀏覽器、文書編輯軟體...

以上幾點做成圖片關係:(如圖一)



(圖一) 作業系統的角色圖

二.boot loader 開機管理程式

1 開機管理程式的功能

載入核心，以啟動作業系統

提供選單功能，具有多重開機功能(Boot Manager)

2 Linux 常見開機管理程式

LILO (LInux LOader)，較早期的程式

grub 目前最常見的開機管理程式

3 開機管理程式可安裝的磁碟區塊

MBR (一顆磁碟只有一個MBR)只有446 bytes

只可以安裝一個Boot Manager(grub，lilo，spfdisk…)

Super block (每個filesystem 都有其first sector)

2-2.Linux的檔案系統

一.檔案系統

1 為一種目錄樹的結構，且將所有裝置視為”檔案”

2 需將裝置連結到目錄樹下的某個目錄，稱為『掛載』

/dev/hda5 掛載到/mnt ，表示進入/mnt 目錄，即可看到

/dev/hda5 磁碟中的檔案資料

3 每個目錄都有其特定的意義，常見的目錄：

/ 最上層目錄

/boot 核心與開機管理程式

/etc 系統與軟體的設定檔

/user unix software resource

/var 系統運作過程中會產生的資料

/home 使用者家目錄

/tmp 暫存檔案放置目錄

/etc, /lib, /bin, /sbin, /dev 不可與根目錄分開(一定得掛載在根目錄之下)

二.磁碟儲存結構

1 硬碟組成

磁區(sector)：最小物理儲存單位(512 Bytes)

磁柱(cylinder)：分割區的最小單位

2 最重要的磁區：整顆硬碟的第一個磁區

主要開機區(MBR)：446bytes，可安裝開機管理程式

分割表(partition table)：64bytes，可記錄四筆分割記錄

3 Super block (boot sector)

每個分割區的第一個磁區(sector)

三.磁碟分割區的類型

1 主要分割區(Primary)

最多四個(4P)，可被格式化使用

2 延伸分割區(Extended)

最多一個(1E)，不可被格式化使用

可再分出邏輯分割區

3 邏輯分割區(Logical)

四.initrd的任務

1 kernel file的功能

偵測硬體並載入適當的模組(驅動程式)

開機過程中『需要掛載根目錄』以載入模組

2 虛擬磁碟裝置(Initial Ram Dist)

提供開機時所必須要的模組(核心非內建的模組)

建立虛擬檔案系統，讓核心在開機過程中可使用

常見非內建模組：LVM, RAID, SCSI, USB…

可用mkinitrd 來建立新的initrd 檔案

五.Linux 的執行等級(run level):

共有七種執行等級

0 關機

1 單人維護模式

2 不含NFS 的多人文字模式

3 多人文字模式

4 保留

5 圖形介面模式

6 重新開機

六.關機指令

選項與參數：

-t sec ： 亦即『過幾秒後關機』的意思

-k ： 只發送警告訊息

-r ： 在將系統的服務停掉之後就重新開機。

-h ： 將系統的服務停掉後，立即關機。

-n ： 不經過init 程序，直接以shutdown 的功能來關機

-f ： 關機並開機之後，強制略過fsck 的磁碟檢查

-F ： 系統重新開機之後，強制進行fsck 的磁碟檢查

-c ： 取消已經在進行的shutdown 指令內容。

時間： 指定系統關機的時間

警告訊息： 通知使用者系統關機警告訊息。

shutdown 範例

shutdown -h now 立刻關機

shutdown -h 20:15 於當天的20:15 分會關機

shutdown -h +10 系統再過十分鐘後自動關機

shutdown -r now 系統立刻重新開機

shutdown -r +30 ‘The system will reboot!!’

第三章 **LINUX**-基本操作及指令

3-1.Linux 的操作

一.指令的操作

每行指令均以 Enter 為『開始執行』的依據，不過可以 \ 延續下達指令可接續參數(Options)來達成多樣化的工作大小寫是完全不同的藉由輸出訊息可瞭解問題、解決之

二.變數的功能

環境當中，一些必要指令的用途

HOME 家目錄

MAIL 用 mail 時取得的 mailbox

PS1 提示字元囉！

PATH 執行檔的搜尋路徑

? 上個指令的執行結果回傳值

三.進階指令操作--資料流重導向

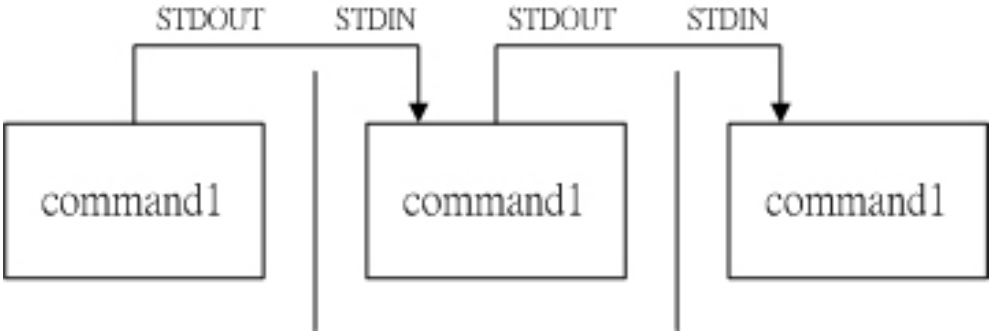
將指令執行後，應該由螢幕輸出的的訊息導向到裝置或檔案當中；

ls /home /vbird >> right 2>> error

最常見在『背景』工作中！例如系統的一堆背景工作紀錄(syslogd)

進階指令操作--管線命令 pipe

將輸出的訊息繼續進行處理做成程序圖:(如圖二)



(圖二) 資料流重導向處理程序圖

四.進階指令操作--管線命令 pipe

```
# last | grep root | wc -l
# cat /etc/passwd | cut -d ':' -f1 > accout
```

重點在於『僅』取得所需的資料，這與正規表示法常常離不開！

文書處理器 vi

管理員至少一定要會一種編輯器

vi 的使用：

一般模式：移動、複製、刪除、貼上

編輯模式：插入與取代文件

指令列模式：搜尋、自動取代、檔案存取等

3-2.Linux 基本指令介紹

一.登入/登出(login/logout)

1. 注意大小寫

2. 安全緣故，可系統管理者(root)預設無法用遠端(ssh)登入

3. logout, exit ：登出 linux 系統

4. su - ：轉換以 root 登入(會執行 root 環境)

5. su ：轉換以 root 登入(仍舊在使用者環境)

二.查看指令說明

1. man 指令名稱

2. 指令名稱 --help

3. man -k 指令名稱

三.命令列指令編輯

1. 『↑』：列出打過的上一個指令

2. history：列出之前打過的指令，每個指令前會有編號(預設記錄 1000 個指令，預設的數目可用$HISTSIZE 定義，而之前打過的指令則存在 ~/.bash_history 檔案中)

3. !N：執行之前打過的編號 N 的指令(!!：執行上一個指令)

4. 按『Tab』鍵，可自動補全檔案或目錄名稱

5. !N:s/原來字串/替代字串：執行之前打過的編號 N 的指令，但將該指令中的原來字串置換為替代字串

6. !字串：執行最近一個以字串開頭的指令

7. !?字串：執行最近一個包含字串的指令

四.列出目錄內容

1. ls：列出簡單的目錄內容

2. ls -l：列出詳細的目錄內容

3. ls -a：列出目錄的所有檔案，包括以"."開頭的檔案

4. ls -F：列出目錄內容，並在檔名的後面加下列的特殊字元"*/=@|"來區別檔案的屬性。*(綠色)：可執行檔、/(藍色)：目錄、無(白色)：一般檔案、@(水藍色)：symbolic link

五.檔案權限設定

1. chmod {a,u,g,o}{+-=}{rwxst}：依照選項執行權限更改

2. chmod {421}{777}：依照選項執行權限更改

3. chown 新的擁有者 檔案：將檔案的擁有者改變

4. chgrp 新的群組 檔案：將檔案的所屬群組改變

六.變更所在目錄

1. cd 路徑名稱：變更到指定目錄

2. cd ~, cd：回到使用者的 Home Directory

3. cd ..：回到上一層目錄

4. 可設定 CDPATH 環境變數

七.顯示目前所在目錄

pwd

八.複製檔案

1. cp 原始檔案 目標檔案

2. cp -R 原始目錄 目標目錄

3. cp -s 原始路徑 目標路徑

九.刪除檔案

1. rm 檔案名稱

2. rm -r 目錄名稱：連子目錄一起刪除

3. rm -f 檔案名稱：強制刪除

4. rm -i 檔案名稱：刪除前再做確認

十.移動檔案

1. mv 原始檔案 目標檔案

2. mv -f 原始檔案 目標檔案

十一.建立新目錄

mkdir 目錄名稱

十二刪除新目錄

rmdir 目錄名稱

十三.產生連結

1. ln -s 真正檔案 連結名稱：建立 Soft Link(軟連結，可指向一個存在或不存在的檔案)

2. ln 真正檔案 連結名稱：建立 Hard Link(硬連結，只能指向一個存在的檔案)

十四.檢查硬碟使用情形

1. df -km 檢查 Partition 可用空間

2. du -kms 檢查目錄已用空間

Job Control(程序控制)

1. 指令&：將指令丟到背景執行

2. CTRL+z：將執行中的指令暫停，並丟到背景

3. fg：將在背景的指令丟到前景執行

4. bg：執行在背景的指令

5. jobs：顯示目前在背景的指令

十五.檔案搜尋設定

1. whereis 檔案：顯示檔案所在位置

2. find 起始目錄 -name 搜尋字串 -print：從起始目錄開始找尋名稱符合搜尋字串的檔案並顯示出來

十六.掛載檔案系統

1. mount -t 檔案系統型態 裝置名稱 掛入點:將指定的裝置掛入指定的掛入點

2. mount -a ：使用/etc/fstab 掛入預設裝置

3. umount 原掛入點：卸載掛入的裝置

4. mount -t iso9660 /dev/cdrom /mnt/cdrom：掛入 CDROM 到/mnt/cdrom 目錄

5. mount -t msdos /dev/fd0 /mnt/floppy：掛入軟碟到/mnt/floppy 目錄

十七.查看登入的使用者

1. w [username]：查看登入的使用者及目前的使用狀態

2. who：查看登入的使用者

3. last [username]：查看曾登入系統的使用者(/var/log/wtmp)

十八.查看程序執行狀態

1. top：監看系統資源使用情形。預設 5 秒更新一次，可執行指令 m(以

記憶體使用排列), t(以執行時間排列), u(觀看特定使用者), k(刪除執行程序)

2. ps -aux：監看系統資源使用情形

3. ps -aux|grep 程序名稱：列出指定程序的執行狀態

十九.刪除執行程序

1. kill PID, kill -9 PID：刪除執行程序 PID

2. kill -HUP PID：重新執行程序 PID，讀入更改過的設定檔

3. top 的 k 指令

二十.更改程序執行優先順序

1. nice priority 指令：以指定的 priority 執行指令(-20/高~19/低, default=10)

2. renice priority PID：以改變程序的 priority

二十一.轉換登入名稱

1. su [usename]：切換到指定的使用者名稱但不改變使用者環境，不指定則為 root

2. su - [usename]：列出詳細的目錄內容且執行該使用者的 startup file

3. sudo 指令：以 root 身分執行指令，可使用的使用者在/etc/sudoers 定義

第四章 **LINUX**-檔案與目錄管理

4-1 檔案與目錄管理：

檔案與目錄的管理上，不外乎『顯示屬性』、『拷貝』、『刪除檔案』及『移動檔案或目錄』等等，由於檔案與目錄的管理在 Linux 當中是很重要的， 尤其是每個人自己家目錄的資料也都需要注意管理，所以介紹有關檔案與目錄的一些基礎管理部分。

一.Is 功能:

在 Linux 系統當中，這個 ls 指令可能是最常被執行的，因為我們隨時都要知道檔案或者是目錄的相關資訊，不過我們 Linux 的檔案所記錄的資訊實在是太多了，ls 沒有需要全部都列出來，所以當你只有下達 ls 時，預設顯示的只有：非隱藏檔的檔名、 以檔名進行排序及檔名代表的顏色顯示如此而已。舉例來說， 你下達『 ls /etc 』之後，只有經過排序的檔名以及以藍色顯示目錄及白色顯示一般檔案，如此而已，其實 ls 的用法還有很多，包括查閱檔案所在 i-node 號碼的 ls -i 選項，以及用來進行檔案排序的 -S 選項，還有用來查閱不同時間的動作的 --time=atime 等選項。 而這些選項的存在都是因為 Linux 檔案系統記錄了很多有用的資訊的緣故。那麼 Linux 的檔案系統中，這些與權限、屬性有關的資料放在哪裡呢？ 放在 i-node 裡面。無論如何， ls 最常

被使用到的功能還是那個 -l 的選項，為此，很多 distribution 在預設的情況中， 已經將 ll (L 的小寫) 設定成為 ls -l 的意思了！其實，那個功能是 Bash shell 的 alias 功能，也就是說我們直接輸入 ll 就等於是輸入 ls -l 是一樣的。

二.複製、刪除與移動： cp, rm, mv

要複製檔案，請使用 cp (copy) 這個指令即可，不過 cp 這個指令的用途可多了， 除了單純的複製之外，還可以建立連結檔 (就是捷徑)，比對兩檔案的新舊而予以更新， 以及複製整個目錄等等的功能，至於移動目錄與檔案，則使用 mv (move)， 這個指令也可以直接拿來作更名 (rename) 的動作，至於移除那就是 rm (remove) 這個指令，複製(cp) 這個指令是非常重要的，不同身份者執行這個指令會有不同的結果產生，尤其是那個-a, -p 的選項， 對於不同身份來說，差異則非常的大！底下的練習中，有的身份為 root 有的身份為一般帳號，請特別注意身份的差別！

三.Cp 功能:

這個 cp 的功能很多，由於我們常常會進行一些資料的複製，所以也會常常用到這個指令的。 一般來說，我們如果去複製別人的資料 (當然，該檔案你必須要有 read 的權限才行！) 時，總是希望複製到的資料最後是我們自己的，所以，在預設的條件中， cp 的來源檔與目的檔的權限是不同的，目的檔的擁有者通常會是指令操作者本身。舉例來說， 上面的範例二中，由於我是 root 的身份，因此複製過來的檔案擁有者與群組就改變成為 root 所有了！

由於具有這個特性，因此當我們在進行備份的時候，某些需要特別注意的特殊權限檔案， 例如密碼檔 (/etc/shadow) 以及一些設定檔，就不能直接以 cp 來複製，而必須要加上 -a 或者是 -p 等等可以完整複製檔案權限的選項才行！另外，如果你想要複製檔案給其他的使用者， 也必須要注意到檔案的權限(包含讀、寫、執行以及檔案擁有者等等)， 否則，其他人還是無法針對你給予的檔案進行修訂的動作。

四.-l 及 –s 的差別:

-l 及 -s 都會建立所謂的連結檔(link file)，但是這兩種連結檔卻有不一樣的情況。這是怎麼一回事？ 那個 -l 就是所謂的實體連結(hard link)，至於 -s 則是符號連結(symbolic link)， 簡單來說，bashrc_slink 是一個『捷徑』，這個捷徑會連結到 bashrc 去！所以你會看到檔名右側會有個指向(->)的符號。

至於 bashrc_hlink 檔案與 bashrc 的屬性與權限完全一模一樣，與尚未進行連結前的差異則是第二欄的 link 數由 1 變成 2 了！

五.rm 功能:

這是移除的指令(remove)，要注意的是，通常在 Linux 系統下，為了怕檔案被誤刪除，所以很多 distributions 都已經預設加入 -i 這個選項了！而如果要連目錄下的東西都一起刪除的話， 例如子目錄裡面還有子目錄時，那就要使用 -r 這個選項了！不過，使用『 rm -r 』這個指令之前要注意，因為該目錄或檔案『肯定』會被 root 刪除！因為系統不會再次詢問你是否要刪除，但如果你確定該目錄不要了，那麼使用 rm -r 來循環殺掉是不錯的方式！

六.檔案內容查閱：

如果我們要查閱一個檔案的內容時，該如何是好呢？ 最常使用的顯示檔案內容的指令可以說是 cat 與 more 及 less 了！此外，如果我們要查看一個很大型的檔案 (好幾百 MB 時)，但是我們只需要後端的幾行字而已，那麼該如何是好？用 tail 呀，此外， tac 這個指令也可以達到

七.各個指令的用途:

- cat 由第一行開始顯示檔案內容
- tac 從最後一行開始顯示，可以看出 tac 是 cat 的倒著寫！
- nl 顯示的時候，順道輸出行號！
- more 一頁一頁的顯示檔案內容
- less 與 more 類似，但是比 more 更好的是，他可以往前翻頁！
- head 只看頭幾行
- tail 只看尾巴幾行
- od 以二進位的方式讀取檔案內容！

4-2.目錄觀念

Linux 系統中以檔案為主，不會出現如 windows 系統中的 C 磁碟、D 磁碟等。

- /bin 系統有很多放置執行檔的目錄
- /boot 這個目錄主要在放置開機會使用到的檔案
- /dev 在 Linux 系統上，任何裝置與周邊設備都是以檔案的型態存在於這個目錄當中。
- /etc 系統主要的設定檔幾乎都放置在這個目錄內
- /home 這是系統預設的使用者家目錄
- /lib 系統的函式庫
- /media 這個/media 底下放置的就是可移除的裝置， 包括軟碟、光碟、DVD 等等裝置都暫時掛載於此
- /mnt 如果妳想要暫時掛載某些額外的裝置
- /opt 這個是給第三方協力軟體放置的目錄。
- /root 系統管理員(root)的家目錄
- /sbin Linux 有非常多指令是用來設定系統環境的
- /srv 一些網路服務啟動之後，這些服務所需要取用的資料目錄

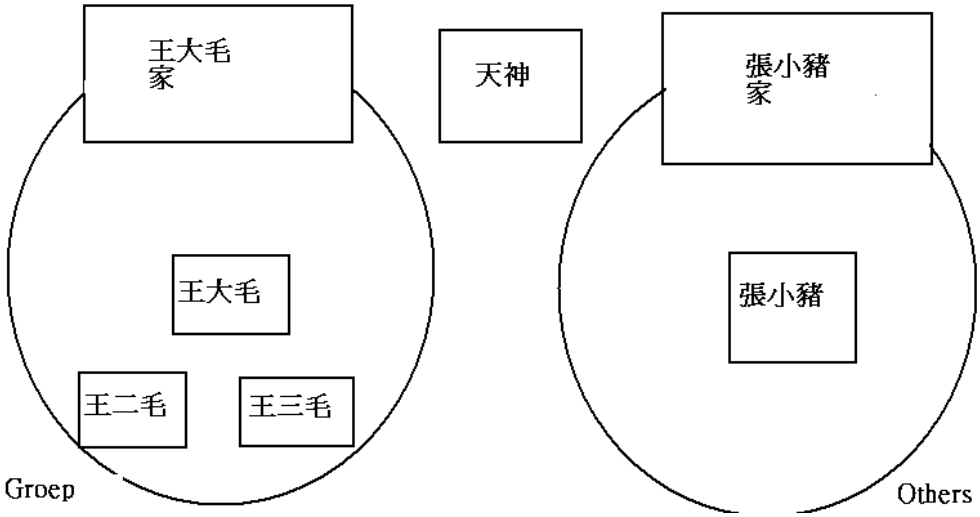
- /tmp 這是讓一般使用者或者是正在執行的程序暫時放置檔案的地方。
- /usr 作業系統軟體資源，可分享與不可變動的資料。
- /var 各項服務做儲存或記錄資料的空間。

4-3.權限觀念

Linux 是多人共用系統，在執行時會分成數種身份執行各稱服務。

在 Linux 裡面，任何一個檔案都具有『User, Group 及 Others』(**使用者本身、群組、其他人**)三種身份的個別權限。

(圖三) ：王三毛就擁有全部權限，而它的王三毛建立了檔案同一家人要看群組權限



(圖三) 家人群組關係圖

- 所以在除了個人目錄外，大多只提供給使用者讀取的權限，所以就不太可能會破壞系統，這是 linux 比較安全的主因。

- 在 Linux 系統中預設有一個超級管理員 root ，擁用全部的權利(就像天神)，可以對任何檔案做更動。
- 平常使用避免使用超級管理員身份工作，以免系統被破壞。當要對系統做更動，例如安裝、移除軟體，更動服務等動作時，再切換到管理員身份工作。

第五章 **LINUX**-程序管理

5-1.程序管理

程序之間是可以互相控制的！舉例來說，你可以關閉、重新啟動伺服器軟體，伺服器軟體本身是個程序， 你既然可以讓她關閉或啟動，當然就是可以控制該程序啦！那麼程序是如何互相管理的呢？其實是透過給予該程序一個訊號 (signal) 去告知該程序你想要讓她作什麼！因此這個訊號就很重要啦！

要給予某個已經存在背景中的工作某些動作時，是直接給予一個訊號給該工作號碼即可。那麼到底有多少 signal 呢？ 你可以使用 kill -l (小寫的 L) 或者是 man 7 signal 都可以查詢到！主要的訊號代號與名稱對應及內容**(如圖四)**

代號	名稱	內容
1	SIGHUP	啓動被終止的程序，可讓該 PID 重新讀取自己的設定檔，類似重新啓動
2	SIGINT	相當於用鍵盤輸入 [ctrl]-c 來中斷一個程序的進行
9	SIGKILL	代表強制中斷一個程序的進行，如果該程序進行到一半， 那麼尚未完成的部分可能會有『半產品』產生，類似 vim會有 .filename.swp 保留下來。
15	SIGTERM	以正常的結束程序來終止該程序。由於是正常的終止， 所以後續的動作會將他完成。不過，如果該程序已經發生問題，就是無法使用正常的方法終止時， 輸入這個 signal 也是沒有用的。
17	SIGSTOP	相當於用鍵盤輸入 [ctrl]-z 來暫停一個程序的進行

(圖四) 訊號代號與名稱對應及內容

上面僅是常見的 signal 而已，更多的訊號資訊請自行 man 7 signal

吧！一般來說，你只要記得『1, 9, 15』這三個號碼的意義即可。那麼我們如何傳送一個訊號給某個程序呢？就透過 kill 或 killall 吧！

5-2. kill -signal PID

一.kill

kill 可以幫我們將這個 signal 傳送給某個工作 (%jobnumber) 或者是某個 PID (直接輸入數字)。 要再次強調的是： kill 後面直接加數字與加上 %number 的情況是不同的！ 這個很重要喔！因為工作控制中有 1 號工作，但是 PID 1 號則是專指『 init 』這支程式！你怎麼可以將 init 關閉呢？ 關閉 init ，你的系統就當掉了啊！所以記得那個 % 是專門用在工作控制的喔！ 我們就活用一下 kill 與剛剛上面提到的 ps 來做個簡單的練習吧！

例題：

以 ps 找出 syslog 這個程序的 PID 後，再使用 kill 傳送訊息，使得 syslog 可以重新讀取設定檔。

答：

由於需要重新讀取設定檔，因此 signal 是 1 號。至於找出 syslog 的

二.PID 可以是這樣做：

ps aux | grep 'syslog' | grep -v 'grep'| awk '{print $2}'

接下來則是實際使用 kill -1 PID，因此，整串指令會是這樣：

kill -SIGHUP $(ps aux|grep 'syslog'|grep -v 'grep'|awk '{print $2}')

如果要確認有沒有重新啟動 syslog ，可以參考登錄檔的內容，使用如下指令查閱：

tail -5 /var/log/messages

如果你有看到類似『Mar 19 15:08:20 www syslogd 1.4.1: restart』之類的字樣，就是表示 syslogd 在 3/19 有重新啟動 (restart) 過了！

三.關於程序的執行順序

我們知道 Linux 是多人多工的環境，由 top 的輸出結果我們也發現，系統同時間有非常多的程序在運行中，只是絕大部分的程序都在休眠 (sleeping) 狀態而已。 想一想，如果所有的程序同時被喚醒，那麼 CPU 應該要先處理那個程序呢？也就是說，那個程序被執行的優先序比較高？ 這就得要考慮到程序的優先執行序 (Priority) 與 CPU 排程！

第六章 **PHP**-資料型態

一.PHP 所支援的資料型態:

PHP 所支援的資料型態 (data types) 有 下面 8 種

1. Boolean(布林)
2. integer(整數)
3. floating (浮點數)
4. string(字串)
5. array(陣列)
6. object(物件)
7. resource(資源)
8. NULL(空值資料型態)

第 1~4 種型別稱為 scalar types ()，第 5~6 種稱為 compound types (複合型別)，最後第 7 與第 8 種則是特殊型別。

PHP 的變數屬鬆散資料型別，雖然支援 8 種不同的型態，但在初始化變數時不必宣告型態，而是在計算時動態 (dynamic) 決定，而非由撰寫程式的人決定。如果要強制設定變數的資料型別的話，可以利用 settype() 函數，或利用 C 語言的強迫轉型方式 (type casting)。

範例(一) 使用變數型別:

PHP 的鬆散資料型別，即我們給定什麼值，該變數即為什麼型別，或是如何使用該變數，該變數即為適當的型別,

(範例一)

```
<?php
$foo = "0";   // $foo 為 string (ASCII 48)
$foo++;          // $foo 變成 string "1" (ASCII 49)
$foo += 1;     // $foo 變成 integer (2)
$foo = $foo + 1.3;   // $foo 變成 double (3.3)
$foo = 5 + "10 Little Piggies"; // $foo 為 integer (15)
$foo = 5 + "10 Small Pigs";        // $foo 為 integer (15)
?>
```

另外還有一點，PHP 的變數都是以 $ (dollar sign) 開頭，並且變數名稱有大小寫之分。

舉例： *$name 或$Name* 以上是兩個不同的變數。

當我們指定值 (value) 給變數時，事實上我們是指定 expression 最後的值給變數，舉例：*$a = 5+5*2;*　($a 的值為 5+5*2 最後的結果，即 15)

範例(二) 變數前加上 & (ampersand) 符號 :

在 PHP 裡，除了給定值給變數外，還可以給定 reference 給變數。也就是，該變數為另外一個變數的 reference，reference 的意義很像是 "becomes an alias for" 或是 "points to"，給定 reference 的方法為，在原來的變數前加上 & (ampersand) 符號，再指定給另外一個變數

(範例二)

```
<?php
$foo = 5;                    // $foo 為 integer 5
$num = &$foo;                // $num 為 $foo 的 reference
?>
```

範例(三) 變數名稱寫法 :

範例二中，&$foo 表示將 $foo 的 reference 指定給 $num 變數，當我們改變 $num 的值，等於改變 $foo 的值，也就是：*$num = 10; echo $foo; // 輸出 10* 使用參考時，來源必須是一個變數名稱

(範例三)

```
<?php
$foo = 10;
$bar = &$foo;            // 正確的寫法
```

```
$bar = &(1+2*2);        // 錯誤的寫法!!!

function test() {
    return 0;
}

$bar = &test();       // 錯誤的寫法!!!

?>
```

第七章 **PHP**-運算子

7-1 數學運算子

(表一) PHP 的數學運算子

運算子	範例	用途
+	$a+$b	加法
-	$a-$b	減法
*	$a*$b	乘法
/	$a/$b	除法
%	$a%$b	取 $a/$b 的餘數
++	$a++	$a = $a+1
--	$a--	$a = $a-1

(表一)中++ 與 -- 為單元運算子 (unary operator)，PHP/FI 2 並不支援這種寫法。這 2 個單元運算子和 C 語言一樣，有前置與後置的寫法：

```
$a = $b++; //第一種寫法

$a = ++$b; //第二種寫法
```

如果 $b 為 5，則第 1 種寫法產生的結果為：

```
$a = 5
$b = 6
```

第 2 種寫法產生的結果為：

```
$a = 6
$b = 6
```

範例(一) :++與--運算子寫法

```
<?php

$x = 0;

for ($i = 0; $i < 10; $i++) {
    echo $x++. " ";
}

for ($i = 0; $i < 10; $i++) {
    echo --$x. " ";
}
?>
```

輸出結果：

0 1 2 3 4 5 6 7 8 9 9 8 7 6 5 4 3 2 1 0

7-2 PHP 的位元運算子

表二:PHP 的位元運算子

運算子	範例	用途
&	$a&$b	做 $a AND $b 的運算
|	$a\|$b	做 $a OR $b 的運算
~	~$a	將 $a 的位元（bit）反相，~ 是一個單元運算子
^	$a^$b	做 $a XOR $b 的運算
>>	$a>>$b	將 $a 向右旋轉 $b 個位元
<<	$a<<$b	將 $a 向左旋轉

(表二中)中要注意的是，向右旋轉後的值仍保有原先的性質符號，例如：

-1 >> 2

-1 向右旋轉 2 個位元後結果仍為 -1，又如：

1 >> 2

1 向右旋轉 2 個位元後，結果為 0。

7-3 PHP 邏輯運算子

表三:PHP 的邏輯運算子

運算子	範例	用途
and	$a and $b	$a 與 $b 同為 true 時結果為 true
or	$a or $b	$a 或 $b 為 true 時結果為 true
xor	$a xor $b	$a 或 $b 為 true，但兩者不同時為 true 時結果為 true
!	!$a	$a 不為 true 時結果為 true
&&	$a && $b	同 and
||	$a || $b	同 or

邏輯運算子利用真值表來觀察會比較清楚：

1. AND

and	*0*	*1*
0	*0*	*0*
1	*0*	*1*

2. OR

or	*0*	*1*
0	*0*	*1*
1	*1*	*1*

3. XOR

xor	*0*	*1*
0	*0*	*1*
1	*1*	*0*

4. ! (NOT)

not	
0	*1*
1	*0*

7-4 PHP 的字串運算子

"." 用在字串上，表示字串連接運算子 (string concatenation operator)

舉例：

```
$a = "Hi! ";
$b = $a . "Dears.";
```

此時 $b 為 "Hi! Dears."，句號用來連接兩個字串的資料型態。

當 string concatenation operator 用在數值型態上時，數值型態會被自動轉型成字串。請看**範例(二)：**

範例(二)

```
<?php

$age = 20;
$str = "Martin is " . $age . " years old.";
echo $str;

?>
```

輸出結果：

Martin is 20 years old.

7-5PHP 的判斷運算子

(判斷運算子還可細分成 2 種：Comparison Operators 與 Condition Operators。)

表四:PHP 的判斷運算子

運算子	範例	用途
Comparison Operators		
==	$a==$b	判斷 $a 是否等於 $b
!=	$a!=$b	判斷 $a 是否不等於 $b
<>	$a<>$b	判斷 $a 是否不等於 $b
Condition Operators		
<	$a<$b	判斷 $a 是否小於 $b
>	$a>$b	判斷 $a 是否大於 $b
<=	$a<=$b	判斷 $a 是否小於等於 $b
>=	$a>=$b	判斷 $a 是否大於等

範例(三) :判斷運算子依結果傳回 true 或 false (真或假)

```
<?php

if ("abcd" > "abcdefg") {
    echo "abcd > abcdefg
";
} else {
    echo "abcd < abcdefg
";
}

?>
```

輸出結果：

abcd < abcdefg

第八章 **PHP**-控制結構

8-1 if 敘述撰寫

If是所有程式語言都有的條件判斷指令。If後面的條件判斷式,是個布林直。當判斷是的值為 true時,代表條件成立,就執行 if 後的敘述。如果 if 後面還有 else 指令,則是當 if 後的條件不成立時,就要執行 eles 後的敘述。

撰寫 if 敘述

if 的 3 種寫法：

第一種:

if (EXPRESSION) statement;
// 當 EXPRESSION 為 true 時，則執行 statement

當 statement 只有一行敘述時，可以省略大括弧。

第二種:

if (EXPRESSION) {
statement1; // 當 EXPRESSION 為 true 時，則執行這裡的
statement2; // statements 敘述 (statement block)。
...

第三種

```
if (EXPRESSION) :
        statement1;    // 當 EXPRESSION 為 true 時，則執行這裡的
        statement2;    // statements 敘述 (statement block)。
        ...
endif;
```

第 3 種語法是 PHP 特有的冒號寫法。

範例(一) if 敘述：

if 敘述的意義為，當 EXPRESSION 為 true 時，則執行 statement(s) 的敘述。

```
<?php
$a = 50;
$b = 10;
if ($a > $b) {
    echo "a is bigger than b";
    echo $a;
}
?>
```

範例(二) :對於只有一行的 statement，省略大括弧的使用

```
<?php

if ($name == 'Jollen')
    echo "Hi! Jollen";

?>
```

當變數 $name 的值等於 Jollen 這個字串時，則顯示 Hi! Jollen 字串。

if 敘述搭配 else 的寫法

else 與 if 語法搭配使用：

舉例一:

```
if (EXPRESSION) {
        statement1;    // 當 EXPRESSION 為 true 時，則執行這裡的
        statement2;    // statements 敘述 (statement block)。
        ...
} else {
        statement3;    // 當 EXPRESSION 不為 true 時，則執行這裡的
敘述。
        statement4;
        ...
}
```

舉例二:

```
if (EXPRESSION) :
        statement1;    // 當 EXPRESSION 為 true 時，則執行這裡的
        statement2;    // statements 敘述 (statement block)。
        ...
else :
        statement3;    // 當 EXPRESSION 不為 true 時，則執行這裡的
敘述。
        statement4;
        ...
endif;
```

範例(三) :加上 else 可以做「如果・・則・・否則・・」的邏輯判斷

```
<?php
if ($name == "Jollen") {
    echo "Man!";
else {
    echo "Woman!";
}
?>
```

8-2 while 敘述撰寫

在編寫程式時,會常遇到有些狀況,有的敘述必須一直重複執行直到要結束的條件成立為止。

撰寫 while

while 是一種迴圈的敘述，語法如下：

舉例一

```
while (EXPRESSION) {
        statement1;    // 當 EXPRESSION 當 true 時，則執行這裡的敘述。
        statement2;
        ...
}
```

舉例二

```
while (expr) :
        statements;
        ...
endwhile;
```

第 2 種格式則是第一種格式的冒號寫法。

while 執行的過程為：先判斷 EXPRESSION，如果 EXPRESSION 為 true，則執行 while 裡的敘述。請看以下(範例一):

範例(四):判斷 EXPRESSION，在 EXPRESSION 為 true，則執行 while 裡的敘述

```
<?php
$a = 1;
$sum = 0;
while ($a <= 10) {
    $sum = $sum+$a;
    $a = $a+1;
}
```

?>

最後 $sum 的值為 55，即 1+2+3+4+5+6+7+8+9+10 的結果。

8-3 do...while 敘述撰寫

do...while 迴圈與 while 迴圈都是重複執行迴圈內的敘述,直到判斷是為 false 為止。但差別在於 do...while 是先做回圈敘述,做完後才作是否還要繼續執行回圈敘述的判斷,所以至少會做到一次回圈敘述;而while迴圈則事先作判斷,判斷式是 ture 才做迴圈敘述,有可能剛開始判斷式就是 false 了,以致於迴圈敘述一次都沒執行到。

撰寫 do...while

範例(五) : do...while 的語法撰寫

```
do {
        statement1;
        statement2;  // 先執行這裡的 statements 一次，然後再判斷
                     // EXPRESSION，
        ...                          // 當 EXPRESSION 當 true 時，則繼續
執行這裡的敘述。
} while (expr);
```

do...while 迴圈的最大特色是，do 裡面的程式碼會至少被執行 1 次。另外，do...while 述敘並沒有冒號的寫法。

範例(六) : do...while 述敘並沒有冒號的寫法

```
<?php
$a = 1;
$sum = 0;
do {
    $sum = $sum+$a;
    $a = $a+1;
} while ($a <= 10);;
?>
```

與 while 的範例結果相同，範例二最後 $sum 的值一樣是 55。

那麼 while 與 do...while 不同的地方什麼場合比較明顯呢？

範例(七): while 與 do...while 在不同的地方場合比較

```
<?php

while ($a == true) {

    echo "Hello!";
    $a = false;
}

do {
    echo "Hello!";
    $a = false;
} while ($a == true);
?>
```

這是 while 與 do...while 的程式片斷，這 2 段程式最大的不同在於：

如果 $a 的初始值為 true，則 2 個程式片斷的輸出結果相同。但是如

果 $a 的初始值為 false，則 while 不會有任何輸出，但是 do...while 則一定會被執行 1 次，因此會輸出 1 個 Hello! 的字串。

8-4 for 敘述撰寫

for 迴圈敘述是一種有條件式的迴圈語法，for 與 while 或 do...while 不同的地方在於，for 可以指定迴圈開始與結束的條作，因此可以限定迴圈的次數。

for 迴圈敘述的語法：

```
for (EXPRESSION1; EXPRESSION2; EXPRESSION3) {
        statements;
        ...
}
```

範例(八)： EXPRESSION1 為最始條件，EXPRESSION2 為終止條件，EXPRESSION3 為迴圈結束後所要執行的 statement。

```
<?php
for ($i = 0; $i <= 10 ; $i++) {
    echo "$i
";
}
?>
```

執行時，$i = 0 表示 $i 的初始值為 0；$i <= 10 表示當 $i <= 10 時，迴圈繼續執行；$i++ 表示每次執行一次迴圈裡的敘述後 $i 的值加 1。

$i++ 的寫法等於 $i = $i+1。這裡有 1 個重要的觀念要說是：

EXPRESSION3 是在迴圈結束後才被執行。

範例(九): EXPRESSION3 觀念

```
for ($i = 0; $i <= 10; $i++) {
    ...
}
```

第九章 **PHP-**函數

9-1 函數定義

PHP 和其他程式語言一樣,提供許多內建的函數可供程式設計者直接使用。

範例(一): PHP 裡定義函數的語法是利用 function 關鍵字

```
<?php
function add($x, $y) {
    return $x+$y;
}
?>
```

這個範例宣告一個名為 add 的函數，並且有兩個傳入值與一個傳回值。要注意的是，因為 PHP 是直譯式 (Interrupt) 語言，所以函數必須在第一次被呼叫之前宣告，否則會出錯。

9-2 傳遞參考 (call by address)

在 PHP 裡如果要傳遞參考 (call by address) 的話，有 2 種做法：

範例(二) : 第一種呼叫函數時在變數前加上 &

```
add(&$x, $y);
```

此時 add() 的寫法沒有什麼不同：

```
function add($x, $y) {
    $x += $y;
}
```

範例(三)： 第二種: 在函數的參數加上 &

```
function add(&$x, $y) {
    $x += $y;
}
```

呼叫時的寫法： `add($x, $y);`

9-3 參數預設值

在設計函數時，為了避免呼叫函數時沒有傳入參數值，因此我們可以替函數的參數加上預設值

範例(四)： 避免呼叫函數時沒有傳入參數值替,函數的參數加上預設值

```
<?php
function add($x = 0, $y = 0) {
    return $x+$y;
}

echo add();
?>
```

因為呼叫 add() 時沒有傳入值，因此最後的輸出結果為：

```
0
```

範例(五)： 呼叫函數時，如果沒有加上傳入值，則使用預設值。重要的是，如果只為某些參數加上預設值，加上預設值的參數必須全部靠右

```
<?php
function add($x, $y = 1, $z = 2) {
    return $x+$y+$z;
}

echo add(5);
?>
```

結果：

8

範例(六)： 底下都是錯誤的寫法

```
unction add($x = 1, $y, $z = 2) {
    return $x+$y+$z;
}

function add($x = 1, $y = 2, $z) {
    return $x+$y+$z;
}
```

範例(七)： 函數是由左而右存放傳入值的關係，錯誤的寫法會導致

PHP 認為傳入的參數不足的錯誤

```
function add($x = 1, $y = 2, $z) {
    return $x+$y+$z;
}

echo add(10);
```

此時，$x = 10，不使用預設值，但因為沒有傳入值給 $z，所以會產生

錯誤。

9-4 變動長度參數

PHP 4 新支援了可變長度的參數用法 (Variable-length argument lists)

範例(八):變長度的參數用法

```
<?php

function foo() {
    $numargs = func_num_args();
    return $numargs;
}

$n = foo (10, 15, 20); //傳入 3 個參數

echo $n; //輸出 3

?>
```

程式裡的 func_num_args() 函數傳回呼叫該函數時所傳入的參數個數，以範例中的 foo(10, 15, 20) 為例，傳入了 3 個參數給 foo() 函數，因此 $numargs 的值為 3。此時 function foo() 不需要加上參數列表 (parameter lists)。

範例(九) : 取得傳入的參數，使用 func_get_arg() 函數

```
<?php
function foo() {
    $numargs = func_num_args();
    if ($numargs > 2) {
```

```
        echo "First: ". func_get_arg(0). "
\n";
        echo "Second: ". func_get_arg(1). "
\n";
        echo "Third: ". func_get_arg(2). "
\n";
    }
    return $numargs;
}

$n = foo (10, 15, 20); //傳入 3 個參數

echo $n;
?>
```

執行後的輸出結果為：

```
First: 10
Second: 15
Third: 20
3
```

結論

對於一個剛出社會的新鮮人要證明自己的能力，最好的辦法就是成績跟證照還有實做成品，證照作用在於如果你正在一家公司就職，你想要升職或是加薪，證照就可以讓您充滿信心，和你同時競爭的同事沒有此類證照，那麼你獲得成功率就會比他們高一些，你就可以有較高的機會實現升職或是加薪，加薪或是升職是為了讓自己更好的生活，實現自己更高的理想。證照是實現你進取心的一個必備的憑證。

此兩張證照有他一定的專業在，但只要認真要考取到不難並可以提升自己在社會中的競爭力，企業在徵才時也會多注意一點。