MASSEY UNIVERSITY
TE KUNENGA KI PŪREHUROA
UNIVERSITY OF NEW ZEALAND

Software Design and Construction
159.251

Operating Systems
file systems and their environments

Amjed Tahir
a.tahir@massey.ac.nz

Overview

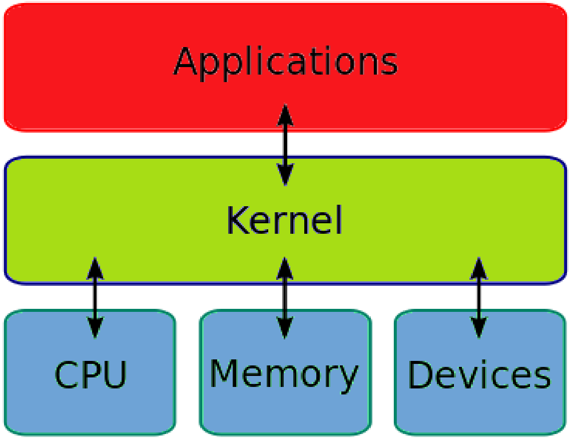
- We tend to ignore the operating systems (e.g. Windows)
  - It's just “setting there"
  - details of OS implementation affect the user
  - not all OS offer the same facilities
  - Different OS have different structure/hierarchy
- OS are everywhere …. Probably in most electronic device we see daily….
- This topic provides an overview of these differences
  - giving commands
  - navigating the filesystem
  - configuring the system environment

Key points

- As you develop a software you will need to deal directly with the OS (e.g., retrieve data from the hard drive).
- In many cases, the software you develop needs to be portable (i.e., works on multiple platforms/OS)
- There are two main types of Software:
  - Software that manages the operations of a computer (called System Software)
  - Software that help the user in performing particular tasks (called Application Software).

What does the Operating System (OS) do?

- Act as an interface between the hardware and the programs requesting Input/output (I/O)
- provides a way to start programs
- manages the file system
- manages resources (e.g. *processor time, memory, , access to I/O devices etc…)*
- restricts access: *prevents unauthorized access to user's files*



Other (key) OS Responsibilities

- Hiding the complexities of hardware from the user.
- Managing between the hardware's resources which include the processors, memory, data storage and I/O devices.
- Handling "interrupts" generated by the I/O controllers.
- Sharing of I/O between many programs using the CPU.

OS examples

- Think about different operating system you've seen
  - Windows distributions
  - Mac OS/X
  - DOS
  - Unix OS (e.g., Solaris and FreeBSD)
  - Linux (e.g. Ubuntu and Red Hat Enterprise Linux)
  - Smart phones and tablets (Android, iPhone and BlackBerry OS)
  - "Smart" TVs
  - eBook readers (e.g. Amazon FireOS)

How do operating systems differ?

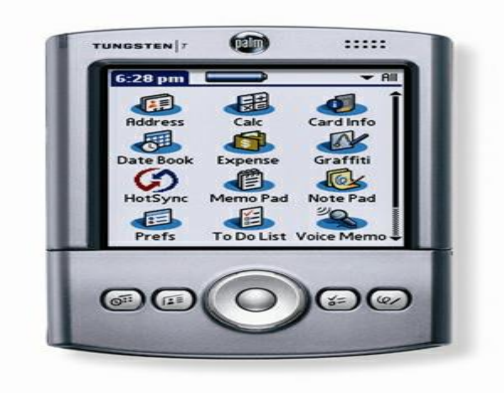
- **An OS can be intended to:**
  - provide good (interactive) response
  - use minimal memory
  - provide enhanced security
  - provide hard real-time response
  - be resilient
  - manage extremely large (terabyte) files
  - server roles?

There's a lot of common functionality

  - your cellphone/tablet
  - your TV
  - laptop & desktop PCs
  - ATMs
  - webservers (e.g. that running Trademe)
  - a Wifi router
- **external view, quite different - internally not so much …**

OS Types

- Single – multi tasking OS
  - Examples: single ➔ Palms OS , multi ➔ Windows 7 or OS X.



Palm OS on a PDA

- Single – multi users OS
  - Examples: single ➔ Windows 3X or DOS , multi ➔ Linux Ubuntu or windows 7.

OS Types

cont’d

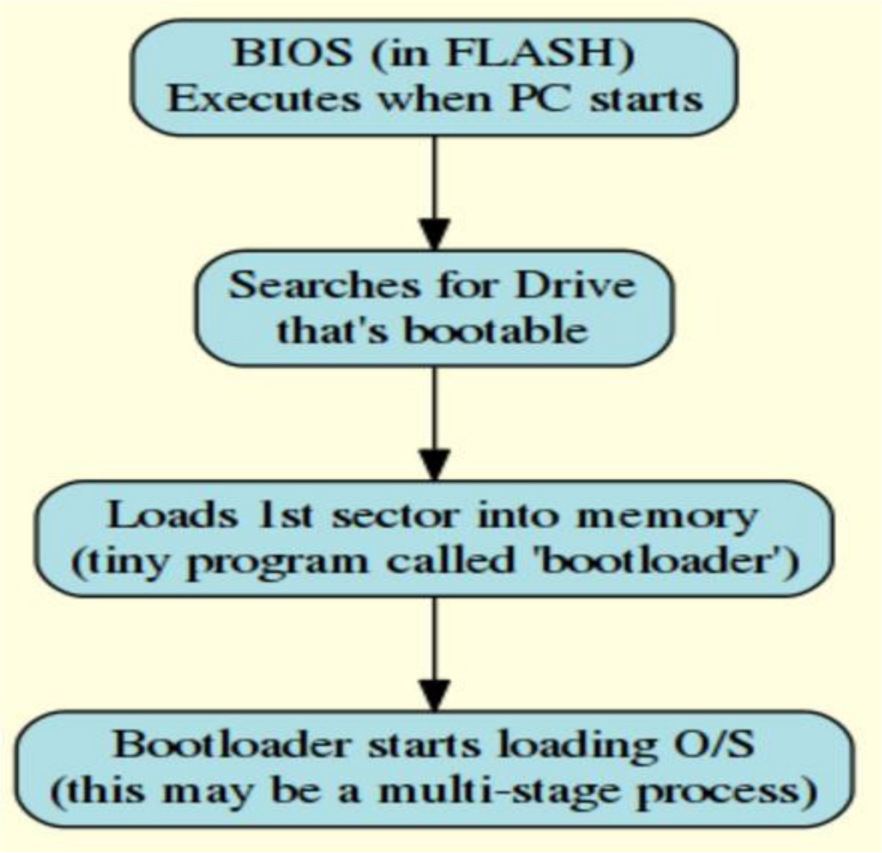
- Distributed OS
  - Example ➔ Amoeba and Windows Server
- Embedded OS
  - Example ➔ Symbian (widely used on Nokia phones!) and Windows CE
- Real-time operating system (event-driven)
  - Example ➔ FreeRTOS (also embedded!)

The Boot Loader

- The BIOS (Basic Input Output System) is stored in Flash memory
- detects devices (mainly storage devices)
- BIOS doesn't know what type of system is installed
  - *can load a block of data and execute it*
  - *this block is the first sector (512 bytes) from a drive*
  - *it contains the **bootloader** (a small program that loads an operating system) …*
- Recently a new standard UEFI (Unified Extensible Firmware Interface) is replacing the BIOS, but conceptually, its function is similar.

PC Boot sequence

- What happens when a PC boots



BIOS (in FLASH)
Executes when PC starts

Searches for Drive
that's bootable

Loads 1st sector into memory
(tiny program called 'bootloader')

Bootloader starts loading O/S
(this may be a multi-stage process)

Operating systems - what's needed

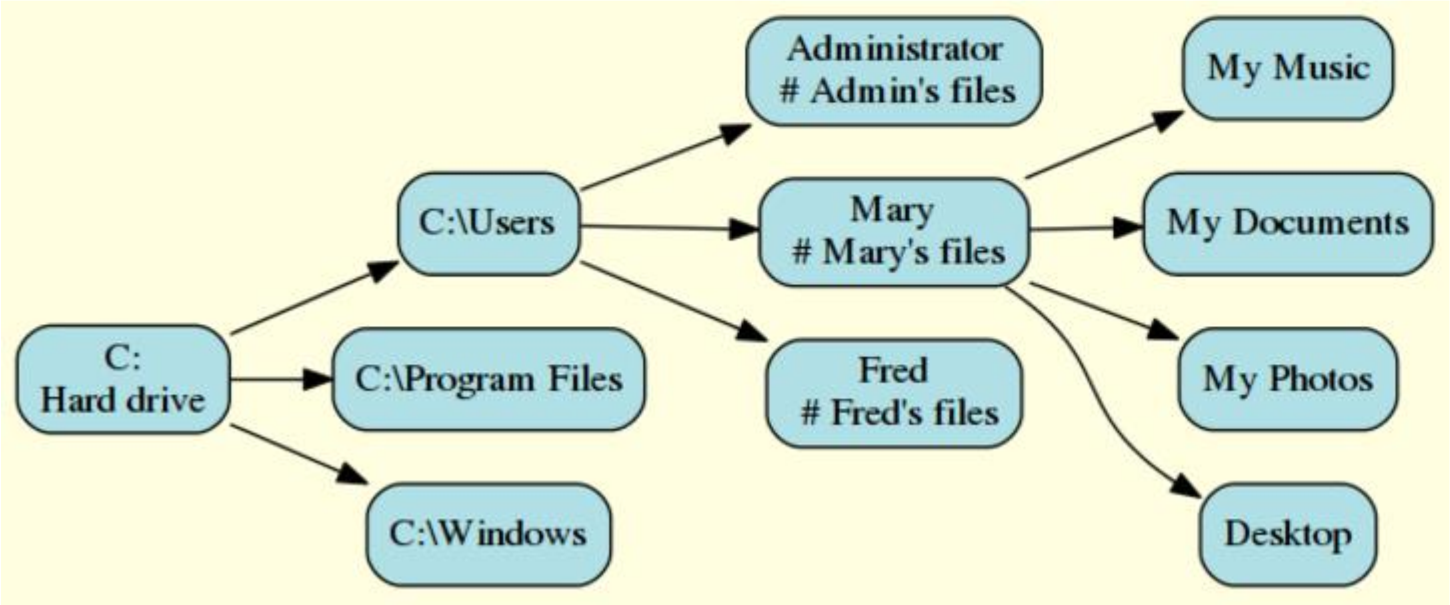
- **OS core**
  - device drivers (interfaces OS to device hardware)
  - file system managers
  - user interface controls
  - command interpreters
  - system libraries
- **System & User programs**
  - system applications (access may be restricted)
  - application programs
  - user-installed programs

OS Configuration data

- user names/passwords …
- the system configuration options
  - *machine ID*
  - *network configuration*
  - *hardware/disk configuration*
- user configuration
  - templates for new users
  - user's personalisation
- Access control
  - keep user data private
  - protect system from users

Windows files system layout

- Windows has separate directories for the system (C:\Windows) and user data (C:\Users):



Windows files system layout

- A Windows file system is not a tree

```
C:\
+  \Windows
+  \Program Files
+  \Users
|        \userX
|        |     \Desktop
|        |     \My Photos
|        |     \My Documents
|        |     \My Music
|        \Other documents# Fred's files
|        \Administrator    # Admin's files
|
D:\       # DVD drive
H:\       # network drive
```

Linux OS layout

- The Linux OS also has separate directories for the system and user data:

  There's a single 'root' directory

  - Linux filesystems have a single root directory called **/**
  - everything is somewhere in a subdirectory of **/**
  - ***all*** devices appear as subdirectories somewhere in its hierarchy

Linux OS layout

cont’d

/	The root directory
/bin	most used commands (binaries)
/boot	files needed to start the operating system (kernel ...)
/etc	system configuration directory (incl passwd file)
/home	*** User directories - one directory per user in here
/lib	system libraries
/media	mounted drives (eg USB sticks) appear here
/opt	Commands installed manually
/sbin	sys-admin commands (binaries)
/tmp	a temporary directory for intermediate files ...
/usr	for user installed binaries, libraries ...

Some Linux directories are 'virtual'

/dev	pseudoDirectory: devices are made to look like files
/proc	pseudoDirectory: for system status
/sys	another pseudoDirectory: for system status

  - these provide uniform access to I/O devices, system status …
  - Windows does similar renaming with the "My Documents" folder
- **Benefits of virtual directories**
  - This can be leveraged to simplify access to
    - system status
    - I/O

Benefits of virtual directories

- There are many commands for dealing with
  - **file** - listing, sorting, filtering, searching …
  - **directories** - useful for organising hierarchical information
- This can be leveraged to simplify access to
- system status
- I/O

FILES AND DIRECTORIES

AND THEIR CHARACTERISTICS

File & directory naming conventions

- What are files and directories?
- **Files - a name given to a block of data**
  - location is irrelevant
  - may be on local drive
  - may be somewhere on a network
- **Directories**
  - A hierarchical structuring of files and folders
    - also called *folders*
    - these form trees (usually)
    - for a given directory hierarchy, there's one 'root' node
- **Directory & file name syntax depends on OS**

Filesystems

- A *filesystem* is the methods and data structures that an operating system uses to keep track of files on a disk or partition.
- Associating a name with a chunk of data seems simple, but file systems can be incredibly complex.
- **File system features:**
  - included meta-data
    - *timestamps: created, last accessed, last modified …*
    - *type of data*
  - Journalling (keeps track of the all transactions - log), to prevent data corruption)
  - deduplication
  - compression
  - sparse files
  - quotas
  - links
- **Skim read this** http://en.wikipedia.org/wiki/File_system

Windows File systems

- Most commonly used are **FAT32** (File Allocation Table) and **NTFS**.
- **NTFS** (New Technology File System) is now the preferred Windows file system
- NTFS filenames: allowed characters
- File names/var are NOT case-sensitive
- FAT32 file names: allowed characters
  - Any except char values 0-31, 127 (DEL) " * / : < > ? \ | + , . ; = [ ]

Mac OS/X & Linux file systems

- **HFS+ file system**
  - permissions: Unix permissions, ACLs (Windows Access Control Lists)
- **Linux ext4 file system**
  - **ext4** filesystem is one of most popular
  - is part of Linux kernel
  - also used on Android operating system
- **ext4 characteristics:**
  - journalled
  - filenames – up to 256 chars, all except "/" and NULL (0)
  - date resolution: 1nS (10^{-9} second)

FILE SYSTEM DIFFERENCES

Filesystems- how do they differ?

- file name length limits
- the case sensitivity
- the path
- Access permissions
- environment variables

Filesystem limits

- The sizes of filenames, pathnames, files and volumes have limits
  - **FAT32** is the oldest – but still widely used
  - **NTFS is** modern file system for Windows.
  - **ext4** is latest Linux file system – also used on Android

	Pathname	**Max**	**Max**	
Filesystem type	**len (max)**	**len (max)**	**filesize**	**volume size**
FAT32	255	no def.limit	4GB	16TB
NTFS	256	32K chars	8EB	8EB
ext4	255	no def. limit	16TB	8EB

1 Exabyte (EB) = 1 x 10^9GB

Undeletable and immovable files in Windows

- While NTFS has 32K pathname length, Win7 Windows Explorer has much lower limit
  - it's possible to create files and then be unable to rename/delete them
  - necessary to:
    - *shorten the path (abbreviate the parent directory names)*
    - *Boot from a Linux Live-CD, mount the Windows filesystem, then delete/rename …*

File system Case-sensitivity

- Window filesystems are (usually) case-insensitive:
  - *data.txt Data.txt DATA.TXT* refer to the same file
- Mac OS/X, Linux and Unix filesystems are CASE-SENSITIVE
  - files called **data.txt Data.txt** and **DATA.TXT** are different
    - *they can exist at the same time in the same directory*
- This complicates moving files from Linux -> Windows
  - *data.txt* and *Data.txt* will map to the same Windows file

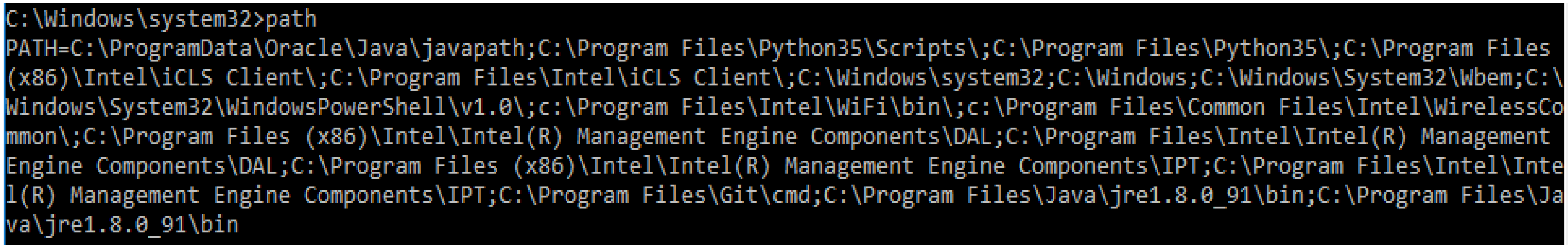
A file's "path"

- The filename by itself doesn't uniquely identify a file as same file may exist in different directories
- A complete specification of a file's location (it's **path**) may need to include:
  - machine
  - drive (if on Windows)
  - list of nested directories
  - filename

```
C:\users\Belinda\My Documents\letter.doc

/home/users/belinda/Documents/letter.doc
```

a file's path vs. the path

- there is a system variable called the "path"
- it's a list of directories to be searched when looking for a program
- more details later …



```
C:\Windows\system32>path
PATH=C:\ProgramData\Oracle\Java\javapath;C:\Program Files\Python35\Scripts\;C:\Program Files\Python35\;C:\Program Files
(x86)\Intel\iCLS Client\;C:\Program Files\Intel\iCLS Client\;C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\
Windows\System32\WindowsPowerShell\v1.0\;c:\Program Files\Intel\WiFi\bin\;c:\Program Files\Common Files\Intel\WirelessCo
mmon\;C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\DAL;C:\Program Files\Intel\Intel(R) Management
Engine Components\DAL;C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\IPT;C:\Program Files\Intel\Inte
l(R) Management Engine Components\IPT;C:\Program Files\Git\cmd;C:\Program Files\Java\jre1.8.0_91\bin;C:\Program Files\Ja
va\jre1.8.0_91\bin
```

The concept of a "Current Working Directory"

- **Absolute paths**
- These contain all intermediate levels from a top-level "device" to destination file.
- **Windows absolute paths will start with a device or machine**

```
C:\Windows
```

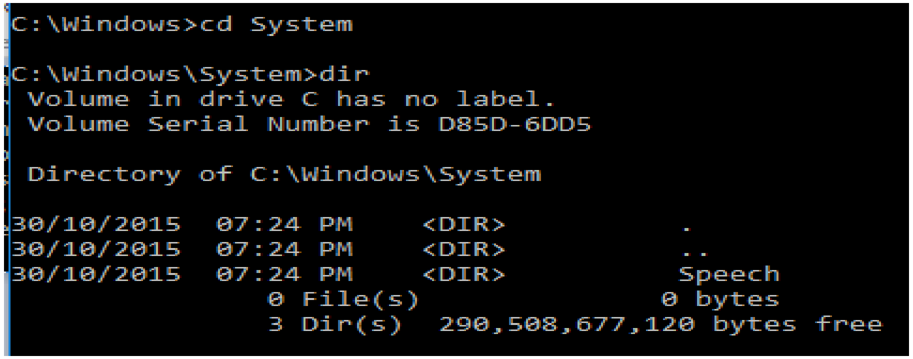
- **Linux absolute paths start with /**

```
/tmp/tmpfile.txt /home/bill
```

The concept of a "Current Working Directory“

cont’d

- **Relative paths**
  - It's tedious to repeatedly specify the whole path to a file/dir.
  - a directory can be nominated as "current working directory" (CWD)
    - *for a user*
    - *for a running program (a process)*



```
C:\Windows>cd System

C:\Windows\System>dir
 Volume in drive C has no label.
 Volume Serial Number is D85D-6DD5

 Directory of C:\Windows\System

30/10/2015  07:24 PM    <DIR>          .
30/10/2015  07:24 PM    <DIR>          ..
30/10/2015  07:24 PM    <DIR>          Speech
               0 File(s)              0 bytes
               3 Dir(s)  290,508,677,120 bytes free
```

The current and parent directories

- The directory names . and .. are special:

  . refers to the current directory

  .. refers to the parent (directory above) current directory

- If we have a directories:

```
C:\Users\Mary\
       |    |pics\
       |    |   beach.jpg
       |    \Docs\                          <<<<< Current Working Directory
       |        letter.doc
       |        jobApplication.doc
       |        ASSIGNMENT\
       |                A1-draft.doc
       \Tom\
           photo.jpg
```

The current and parent directories

cont’d

- If **C:\Users\Mary\Docs** is CWD (current working directory):

letter.doc	refers to the letter
./letter.doc	also refers to the letter.doc in current directory (same as letter)
ASSIGNMENT/A1-draft.doc	refers to a file in the subdir ASSIGNMENT
../pics/beach.jpg	go up one directory (to Mary), down intoPics, get beach.jpg
../../Tom/photo.jpg	go up to Users then down …

Determining the 'type' of a file

- It's useful to know the type of a file:
  - separates *data* from *programs*
  - ensures only programs are executed
- **Windows uses file extension**
  - part of the filename after the last dot
  - used to link an extension to a program
- **Alternate methods:**
  - embed the *handler program* in the file's **meta-data**
    - *'meta-data' is data about the data!*
    - *encode the handler program in line 1 of the file*
      e.g.
      #! /usr/bin/python

File and directory security

- It's important to control access to files and directories (and programs)
- **Users shouldn't be able to alter each others files**
  - default should be private
- Generally, there are two classes of users
  - **"normal" users:** unfamiliar with the technicalities of the O/S
    - should only be able to affect their own files and processes (programs)
    - shouldn't be able to see other user's files
- **privileged (admin) users:** can affect the running system
    - install/remove software
    - view/delete other user's files
- **partitioning helps protect against malicious users**

File and directory permissions - overview

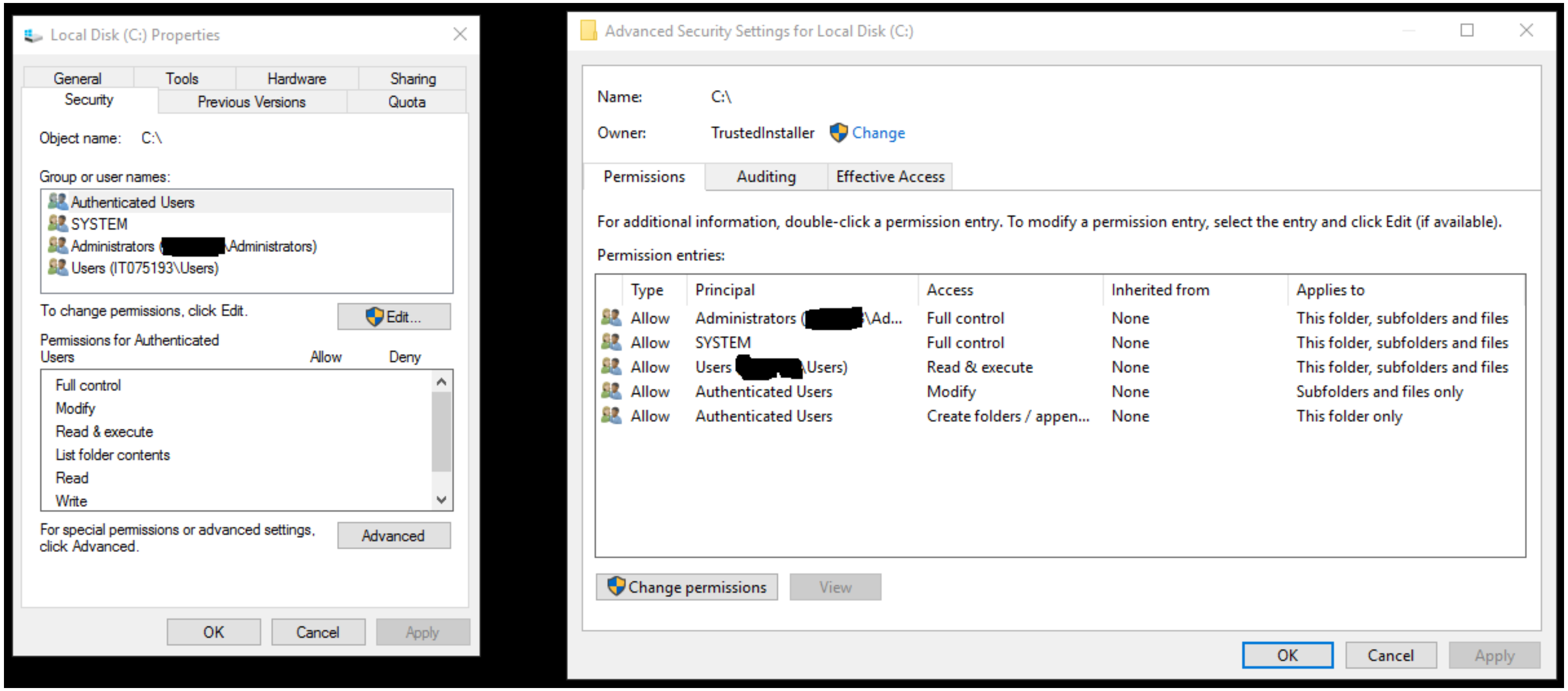
- Files and directories can have associated permissions (access categories):
  - it's important to keep files private
  - protect the operating system
- **Files can be:**
  - read-only
  - write-only (unusual but possible)
  - executable:
    - *binary (e.g. a .EXE file in Windows)*
    - *a text file containing a script*
- **Directories have more possibilities:**
  - open to all users
  - specified users can: *read, scan file list, write* to directory
    - *possible to read a file from a subdirectory but not list the files in that directory*
- **It's important that backup programs restore permissions correctly**

File access permissions

- **Permissions can be defined**
  - per **user:** this lets a user have full control over an item (file/dir)
  - per **group:** where a users are added to nominated groups (e.g. staff, students, administrators)
- **The protection mechanisms can be:**
- Simple
  - Linux **User/Group/Other** model
- Complex
  - Windows **ACLs** (access control lists)
- **more complex (e.g. role-based) are possible**

Windows file permissions

- Windows has a very comprehensive (and complex) system for managing permissions to files and directories.
-



Object name: C:\

Group or user names:

- Authenticated Users
- SYSTEM
- Administrators ([redacted]\Administrators)
- Users (IT075193\Users)

To change permissions, click Edit.

Permissions for Authenticated Users	Allow	Deny
Full control		
Modify		
Read & execute		
List folder contents		
Read		
Write		

For special permissions or advanced settings, click Advanced.

Name: C:\

Owner: TrustedInstaller

For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

Permission entries:

Type	Principal	Access	Inherited from	Applies to
Allow	Administrators ([redacted]\Ad...	Full control	None	This folder, subfolders and files
Allow	SYSTEM	Full control	None	This folder, subfolders and files
Allow	Users [redacted]\Users)	Read & execute	None	This folder, subfolders and files
Allow	Authenticated Users	Modify	None	Subfolders and files only
Allow	Authenticated Users	Create folders / appen...	None	This folder only

Hide files

- Not all the files may be visible, for example:
  - per/user application settings
  - Temporary Internet files (web cache)
- **Linux - start name with a dot**
  - There are many hidden files in a users home directory, to remember settings … e.g.

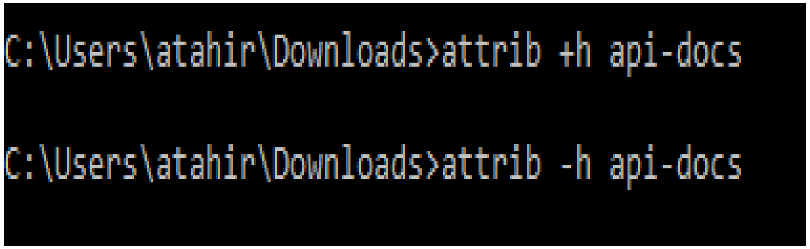
.xsession - hidden file with Ubuntu desktop settings
.ssh - hidden directory containing ssh configuration data

- You can use either the GUI File Properties dialog or command line to set or change file properties.

Hide files

cont’d

- Windows
  - You can use either the GUI File Properties dialog or command line to set or change file properties.
  - **command line**: use the ***attrib*** command



```
C:\Users\atahir\Downloads>attrib +h api-docs

C:\Users\atahir\Downloads>attrib -h api-docs
```

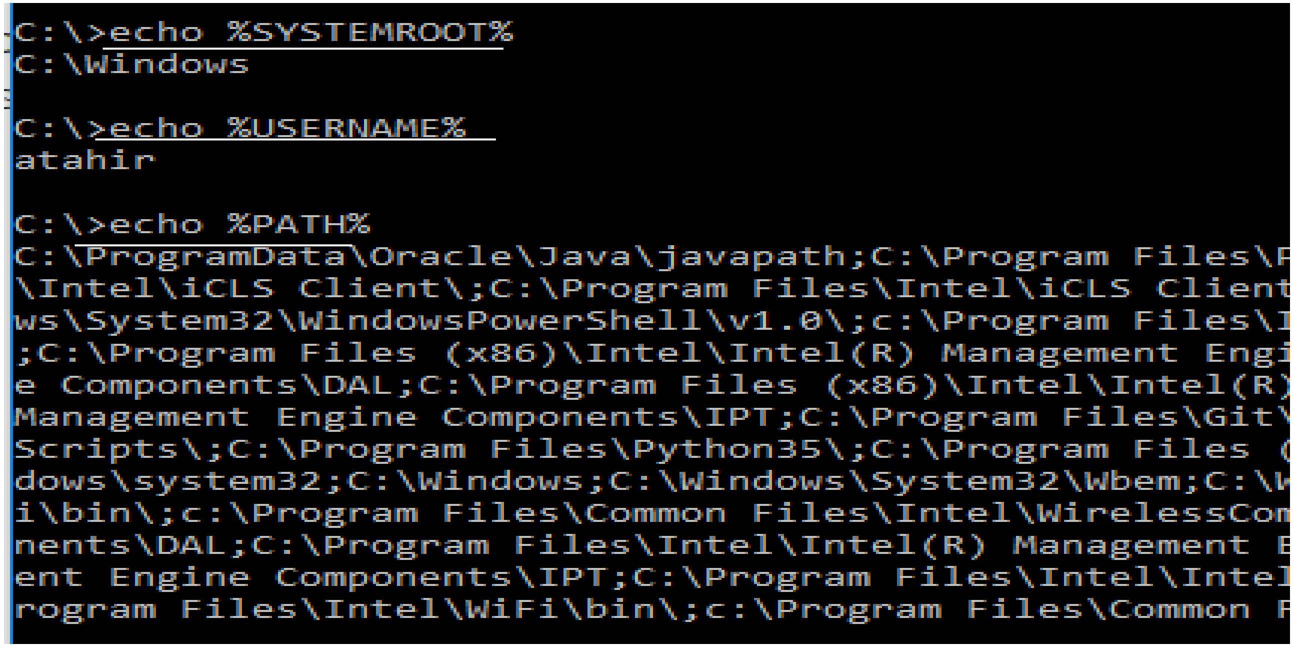
Environment Variables

- An **environment variable** is a dynamic "object" on a computer that stores a value, which in turn can be referenced by one or more software programs in the OS.
- **Environment variables have many uses:**
  - they're like *global variables* in programming
  - define system settings
  - used to exchange information between running programs
- **provide a method for processes to get input or leave results:**
  - **where is the Windows directory on this PC?**
    (%SYSTEMROOT%)
  - **what's the login name of the current user?**
    (%USERNAME%)
  - **where is the user's home directory?**
    (%HOME%)
  - **which directories should be searched for executable programs?**
    (%Path%)

Environment Variables

cont’d

- Windows EV: Display EV from command line
  - Use ***echo*** command (echo a line of text (string) on standard output or a file.)

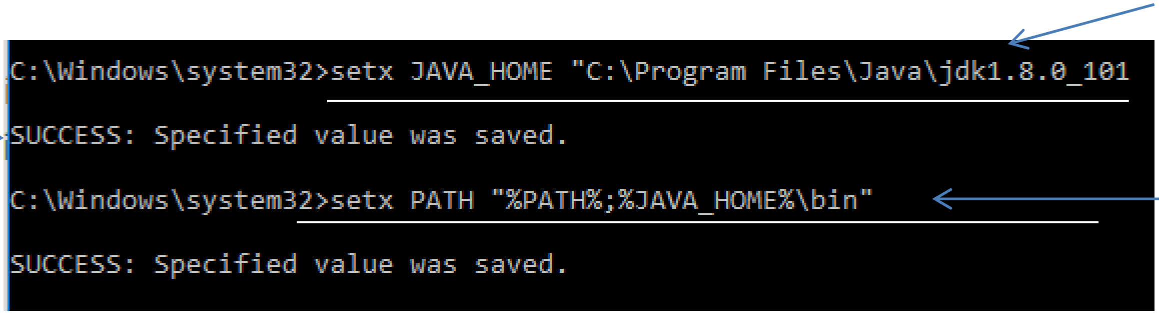


```
C:\>echo %SYSTEMROOT%
C:\Windows

C:\>echo %USERNAME%
atahir

C:\>echo %PATH%
C:\ProgramData\Oracle\Java\javapath;C:\Program Files\P
\Intel\iCLS Client\;C:\Program Files\Intel\iCLS Client
ws\System32\WindowsPowerShell\v1.0\;c:\Program Files\I
;C:\Program Files (x86)\Intel\Intel(R) Management Engi
e Components\DAL;C:\Program Files (x86)\Intel\Intel(R)
Management Engine Components\IPT;C:\Program Files\Git\
Scripts\;C:\Program Files\Python35\;C:\Program Files (
dows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\W
i\bin\;c:\Program Files\Common Files\Intel\WirelessCom
nents\DAL;C:\Program Files\Intel\Intel(R) Management E
ent Engine Components\IPT;C:\Program Files\Intel\Intel
rogram Files\Intel\WiFi\bin\;c:\Program Files\Common F
```

Environment Variables

cont’d

- The ***setx*** command permanently updates the environment variables.
- To add/update system environment variables, you must use the ***-m*** switch and open the command prompt using Administrator privilege.
- Example: setting a new directory called Java_Home ➔ to point to the directory where the JRE is installed.



```
C:\Windows\system32>setx JAVA_HOME "C:\Program Files\Java\jdk1.8.0_101

SUCCESS: Specified value was saved.

C:\Windows\system32>setx PATH "%PATH%;%JAVA_HOME%\bin"

SUCCESS: Specified value was saved.
```

Set a new value for Java_Home var (i.e., the directory of the applicati)

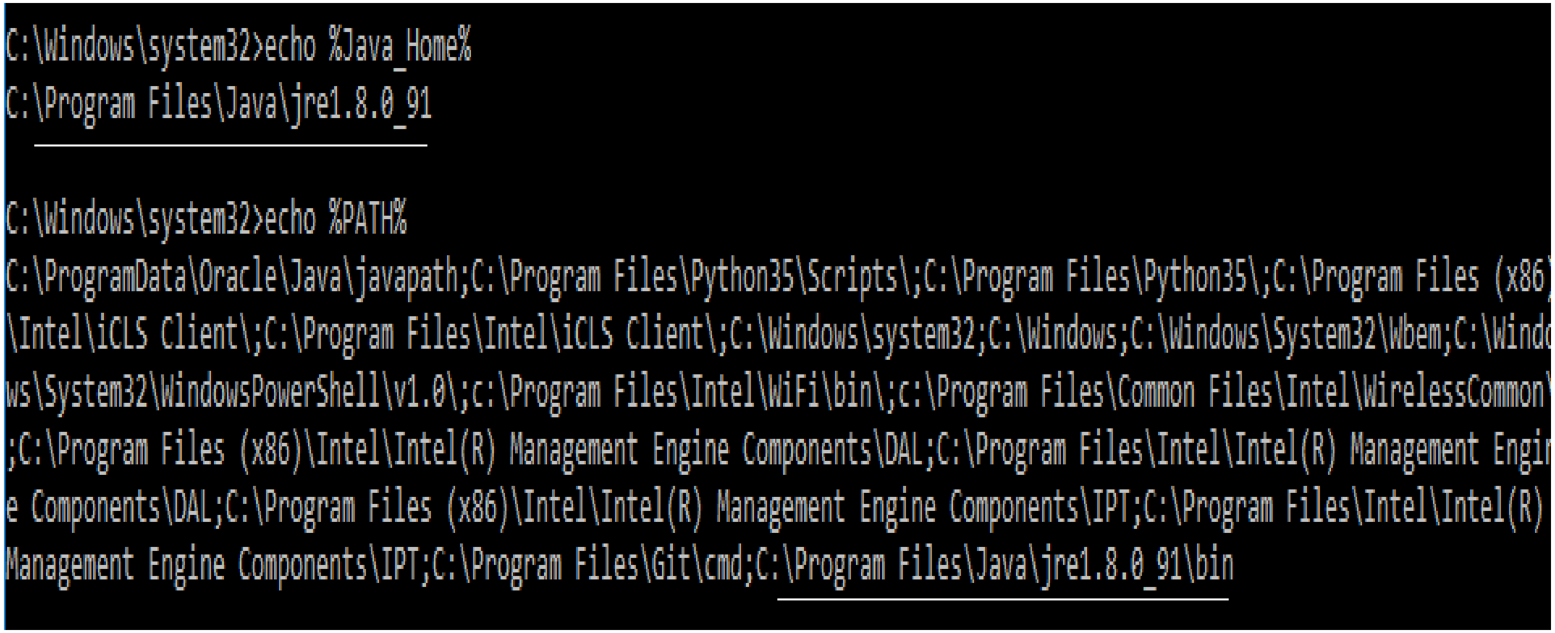
The results

Set a new value for F var

Environment Variables

cont’d

- Check that the variables were correctly added/changed

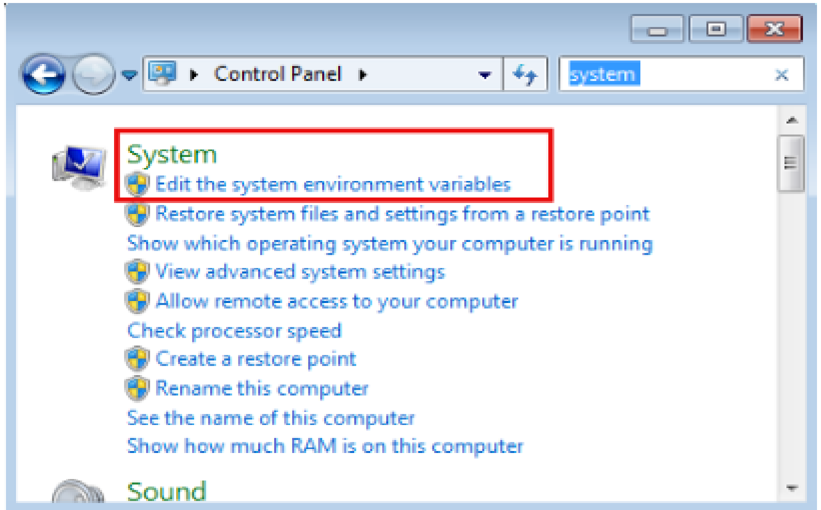


```
C:\Windows\system32>echo %Java_Home%
C:\Program Files\Java\jre1.8.0_91

C:\Windows\system32>echo %PATH%
C:\ProgramData\Oracle\Java\javapath;C:\Program Files\Python35\Scripts\;C:\Program Files\Python35\;C:\Program Files (x86)
\Intel\iCLS Client\;C:\Program Files\Intel\iCLS Client\;C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windo
ws\System32\WindowsPowerShell\v1.0\;c:\Program Files\Intel\WiFi\bin\;c:\Program Files\Common Files\Intel\WirelessCommon\
;C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\DAL;C:\Program Files\Intel\Intel(R) Management Engin
e Components\DAL;C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\IPT;C:\Program Files\Intel\Intel(R)
Management Engine Components\IPT;C:\Program Files\Git\cmd;C:\Program Files\Java\jre1.8.0_91\bin
```

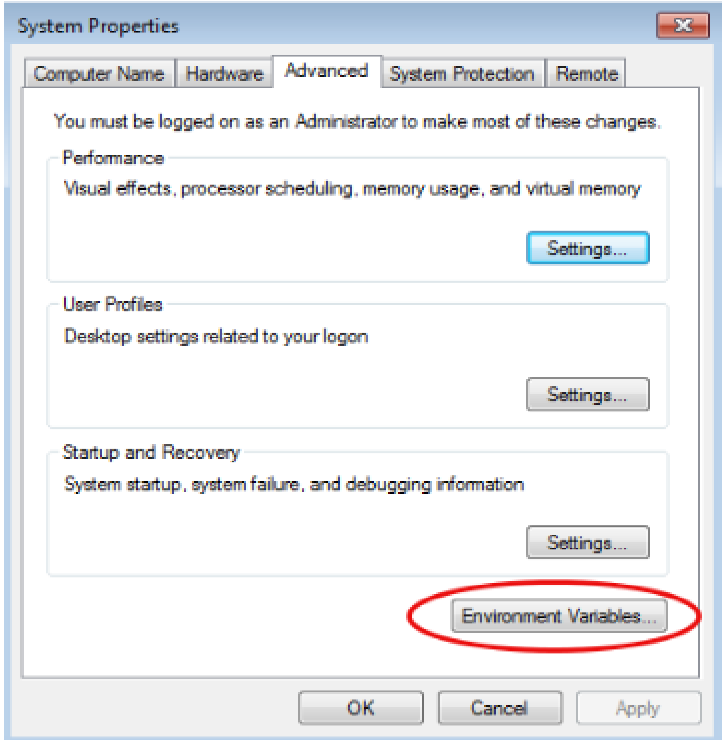
Changing the system-wide path in Windows

- **Changing the system-wide path in Windows**

  System/Advanced/EnvironmentVariables/Path



System
Edit the system environment variables
Restore system files and settings from a restore point
Show which operating system your computer is running
View advanced system settings
Allow remote access to your computer
Check processor speed
Create a restore point
Rename this computer
See the name of this computer
Show how much RAM is on this computer
Sound



You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory

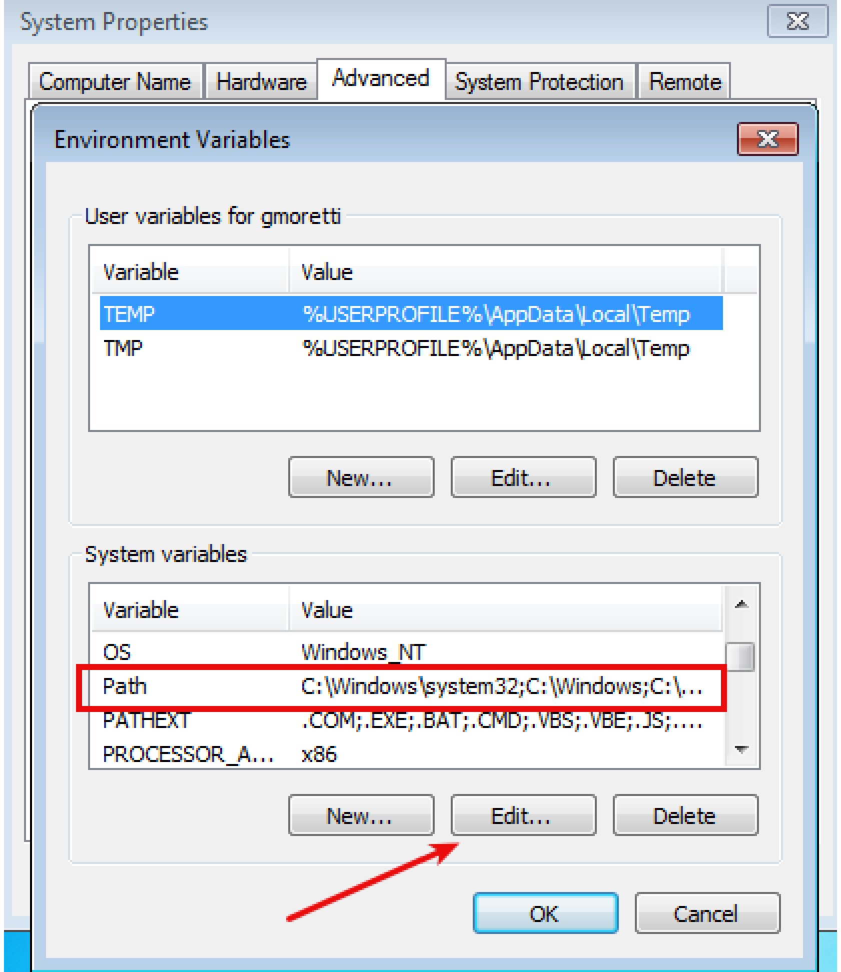
User Profiles
Desktop settings related to your logon

Startup and Recovery
System startup, system failure, and debugging information

Changing the system-wide path in Windows

cont’d

- **select Path and Edit**



User variables for gmoretti

Variable	Value
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
OS	Windows_NT
Path	C:\Windows\system32;C:\Windows;C:\...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;....
PROCESSOR_A...	x86

More on Windows CMD

- Here is a list of tasks that you can do with Command Lines
- http://www.computerhope.com/issues/chusedos.htm
- A reference guide to Command Lines
  - http://ss64.com/nt/