NSS

Intro to BASH

So why do I have to learn Terminal?

Most web servers run Linux

Most of these servers do not have a GUI

As web professionals you will need to know how to interact with these servers

Later in the program you will use CLI to write code and interact with different programs such as MySQL, Coldfusion, and ActionScript

Command Line Syntax

Syntax - the grammar that the commands must follow

- Different between different operating systems

Commands usually follow one of the following forms:

- [doSomething] [how] [toFiles]
- [doSomething] [how] [sourceFile] [destinationFile]
- [doSomething] [how] < [inputFile]
- [doSomething] [how] > [outputFile]
- [doSomething] [how] | [doSomething] [how] | [do Something] [how] > [outputFile]

The command line does not treat you like an idiot. It assumes that you know what you are doing.

Case sensitivity matters!!!

Terminal commands

pwd - print working directory

Shows your current location in the file structure

ls - list

Lists the contents of a directory

cd - change directory

Moves you to another directory in the file structure

Navigation Commands

cd	change directory
ls	List the files and folders in the current directory
pwd	Current directory (print working directory)
.	current directory
..	Move to the parent directory
~	Move to the home directory

File Commands:

touch	creates a file
mv	moves and/or rename a file
cp	copies a file
rm	removes a file
awk	find and replace text within files
cmp	compare two files
find	search for a file
open	Opens a file/folder/URL/Application
man	Displays help (q to quit)

Directory Commands

mkdir- makes a directory

rmdir- removes a dire

Access Commands

chmod- change permissions

chown- change owner

Terminal Commands

clear	clears the terminal
exit	exits the terminal
less	displays output one screen at a time
nano	text editor
ssh	connect to a remote server securely

Process Commands

Top	lists top processes
Kill	kills a process
ps	process status

User Commands

passwd	change a user password
su	substitute user
sudo	perform an action as another user
who	prints name of users currently logged in
whoami	prints currently user id and name
useradd	creates a new user
userdel	deletes user
usermod	modifies a user

Network Commands

ifconfig- pulls networking information about the current machine

netstat- shows network resources

ping- sends a test packet to a remote device

tcpdump- dump all traffic on a network

Variables

```
file="./file"
echo $file
```

comments

#comments

Well known commands

if <condition>; then

<commands>

else

<commands>

fi

Checks

String Comparisons

=	equal
!=	not equal
<	less then
>	greater then
-n s1	string s1 is not empty
-z s1	string s1 is empty

Arithmetic Comparisons

-lt	<	Less than
-gt	>	Greater than
-le	<=	Less than or equal to
-ge to	>=	Greater than or equal
-eq	==	Equal to
-ne	!=	Not equal to

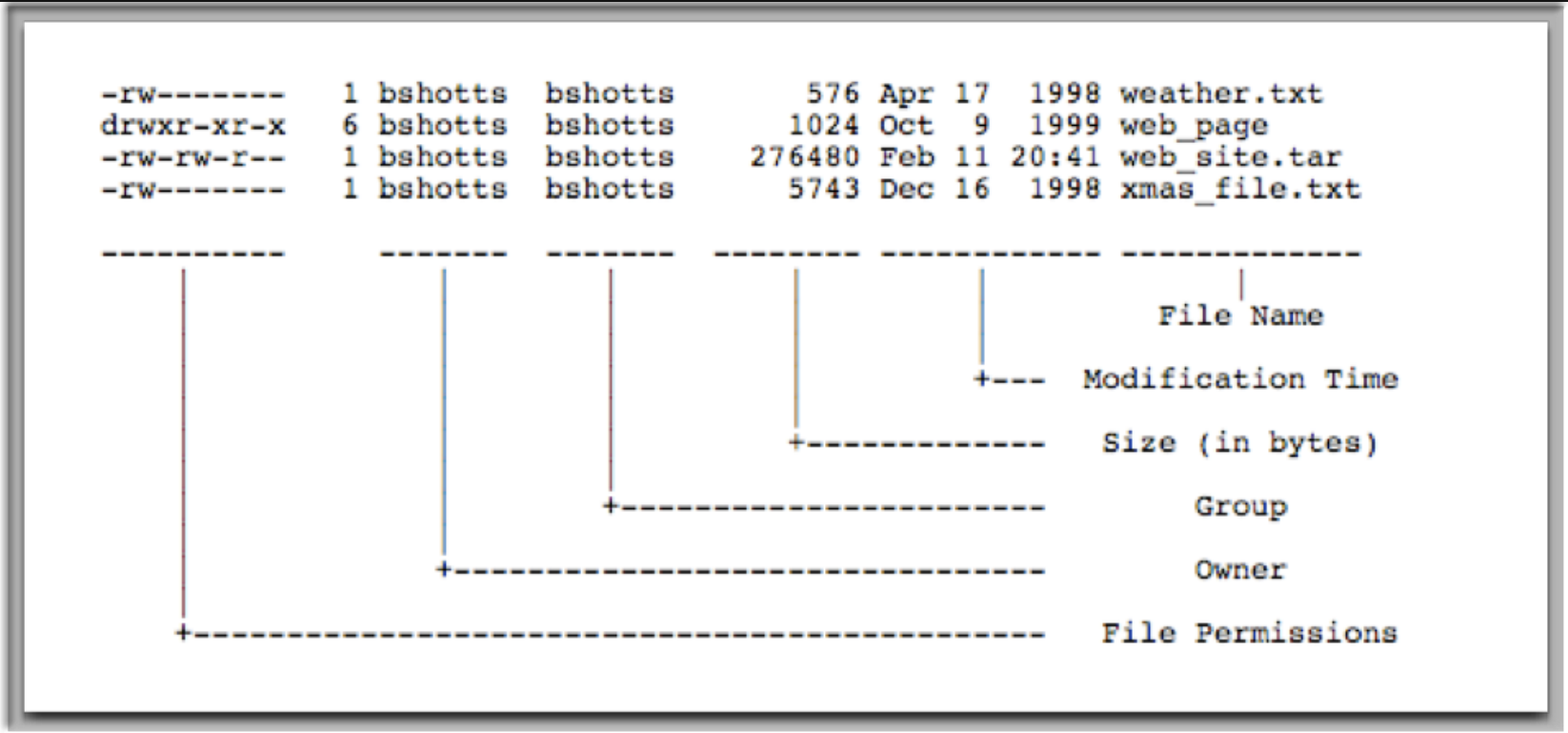
Bash File Testing

-b filename	Block special file	-g filename	true if file exists and is set-group-id.
-c filename	Special character file	-O filename	True if file exists and is owned by the effective user id.
-d directoryname	Check for directory existence	-G filename	Check if file exists and is owned by effective group ID.
-e filename	Check for file existence	-r filename	Check if file is a readable
-w filename	Check if file is writable	-s filename	Check if file is nonzero size
-L filename	Symbolic link	-u filename	Check if file set-ser-id bit is set
-k filename	Sticky bit	-x filename	Check if file is executable
-S filename	Check if file is socket	-f filename	Check for regular file existence not a directory

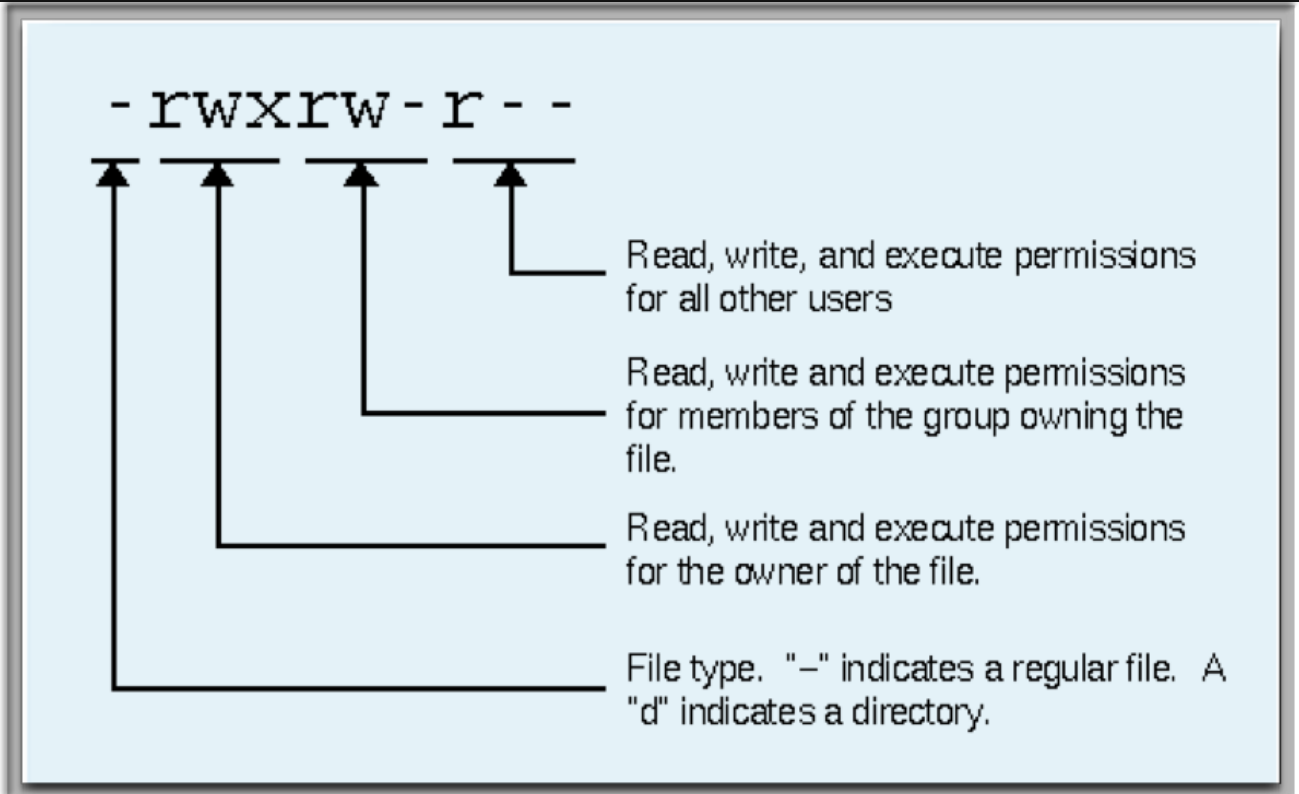
Bash File Testing Script

```
#!/bin/bash
file="./file"
if [ -e $file ]; then
	echo "File exists"
else
	echo "File does not exists"
fi
```

File Permissions



File Permission



File Permissions

```
rwx rwx rwx = 111 111 111
rw- rw- rw- = 110 110 110
rwx --- --- = 111 000 000

and so on...

rwx = 111 in binary = 7
rw- = 110 in binary = 6
r-x = 101 in binary = 5
r-- = 100 in binary = 4
```

File Permissions

Value	Meaning
777	**(rwxrwxrwx)** No restrictions on permissions. Anybody may do anything. Generally not a desirable setting.
755	**(rwxr-xr-x)** The file's owner may read, write, and execute the file. All others may read and execute the file. This setting is common for programs that are used by all users.
700	**(rwx------)** The file's owner may read, write, and execute the file. Nobody else has any rights. This setting is useful for programs that only the owner may use and must be kept private from others.
666	**(rw-rw-rw-)** All users may read and write the file.
644	**(rw-r--r--)** The owner may read and write a file, while all others may only read the file. A common setting for data files that everybody may read, but only the owner may change.
600	**(rw-------)** The owner may read and write a file. All others have no rights. A common setting for data files that the owner wants to keep private.

File Permissions

Value	Meaning
777	**(rwxrwxrwx)** No restrictions on permissions. Anybody may list files, create new files in the directory and delete files in the directory. Generally not a good setting.
755	**(rwxr-xr-x)** The directory owner has full access. All others may list the directory, but cannot create files nor delete them. This setting is common for directories that you wish to share with other users.
700	**(rwx------)** The directory owner has full access. Nobody else has any rights. This setting is useful for directories that only the owner may use and must be kept private from others.

chmod

Used to change permissions on files or directories

chmod [permission] [filename]

Example

chmod 644 file2

Changes the permission for file 2 to -rw-r--r--

chmod

Alternate Method

- u = owner
- g = group
- o = others (everyone else)
- a = all

chmod

“**+**” adds the specified modes to the specified classes

“**-**” removes the specified modes

chmod ug+rwx, o+r *[file]*

Sets permissions to -rwxrwx-r--

While loop

```
#!/bin/bash
COUNT=6
# bash while loop
while [ $COUNT -gt 0 ]; do
    echo Value of count is: $COUNT
    let COUNT=COUNT-1
done
```

Until loop

```
#!/bin/bash
COUNT=0
# bash until loop
until [ $COUNT -gt 5 ]; do
        echo Value of count is: $COUNT
        let COUNT=COUNT+1
done
```