Bash shell

CS 2XA3

Term I, 2018/19

Outline

Shells

- A Unix shell is a command-line interpreter
- Interface to Unix/Linux OS
- Bourne **bash** shell; default on many systems
- **`which bash`**, answer **`/bin/bash`**
  only searches the directories listed in the **`PATH`** variable
- Also C shell **csh**, **tcsh**, Korn shell **ksh**, ...

Shell scripts

- Text file containing shell commands
- First line specifies the shell to be used
  - **`#!/bin/bash`** bash shell
  - **`#!/bin/sh -x`** calls bash in debug mode; prints each line
  - **`#!/bin/csh`** C shell

Shell variables

- `name=value`
  no spaces around =
- To access the value, use `$`, e.g. `$PATH`
- Examples
  - `name=/usr/lib/orb`
    `echo $name`
    will show `/usr/lib/orb`
  - `x=`ls``
    `echo $x`
    will show all items in the current directory, as if you typed `ls`

`ls` – see 3.5.4. command substitution in Bash manual in Help section.

Command-line arguments

- `$0` name of the script/command
- `$1` first argument
- `$2` second argument, ... and so on

Try the following script:

```
#!/bin/bash
echo "My␣name␣is␣$0"
echo "First␣argument␣is␣$1"
echo "Second␣argument␣is␣$2"
```

␣ *indicates space (blank)*

`"My name is $0"` – see 3.1.2. quoting in Bash manual in Help section. Why `'My name is $0'` would not work?

For loops

```
for i in list
do
  # command(s)
done
```

# indicates comment, try the following script

```
#!/bin/bash
files=`ls`
for i in $files
do
  echo "Echoing file name: " $i
done
```

For loops

```
for i
do
  # command(s)
done
```

default is the list of command-line arguments

Conditionals

```
if cmd
then
  # command(s)
fi
```

```
if cmd
then
  # command(s)
else
  # command(s)
fi
```

- ***cmd*** is any command or command sequence
- ***true*** is when it returns 0
- ***false*** is when it returns $\neq 0$

Tests

- **test** evaluates any conditional expression
  0 if true, 1 if false
- same as `[ args ]`, **`args`** is an expression

```
if [ args ]
then
   # command(s)
else
   # command(s)
fi
```

Tests

expression	true if
`str1 = str2`	**str1** equals **str2**
`str1 != str2`	**str1** does not equal **str2**
`-r file`	**file** exists and is readable
`-w file`	**file** exists and is writable
`-d file`	is a directory
`-f file`	is a regular file
`-s file`	is a file of size > 0
`expr1 -a expr2`	**expr1** and **expr2** are both true
`expr1 -o expr2`	**expr1** or **expr2** is true

Examples

- ```
  #!/bin/bash
  if [ "$1" = "foo" ]
  then
    echo "First argument is foo"
  else
    echo "First argument is not foo"
  fi
  ```
- `if [ -r file.txt ]` if `file.txt` is readable
- `if [ "$1"= "foo" -a -r file.txt ]`
  if the first argument is `foo` and `file.txt` is readable
  ```

Shell customization

- **`.bashrc`** executes when Unix starts a new shell
- **`.bashrc_profile`** executes on login

  `.bashrc` runs first
- C shell
  - **`.cshrc`**
  - **`.login`**

Environment variables

- **`PATH`** specifies where the shell searches for commands
- **`export`** defines a variable
- to add /usr/local/bin to the path
  `export PATH=$PATH:/usr/local/bin`
- to add *current directory* to the path
  `export PATH=$PATH:.`
- **`HOME`** home directory
- to see all environment variables, use **`printenv`**

Aliases

- Stored in `.bash_profile`
- `alias newname='command'`

  examples:
  - `alias rm='rm -i'` to prompt before removing a file
  - `alias cp='cp -i'` to prompt before copying a file