

Chapter 9

Quoting Mechanisms

Overview

- Quoting metacharacters
- Here documents

Lesson: Quoting metacharacters

- \ Backslash
- ' Single Quotes
- " Double Quotes

Quoting - \

- \ removes the special meaning of the next character.

```
$ echo the \\ escapes the next character
the \ escapes the next character

$ color=red\ white\ and\ blue
$ echo the value of \color is $color
the value of $color is red white and blue

$ echo one two \
> three four
one two three four
```

Quoting - '

- ' removes the special meaning of all characters surrounded by the single quotes.

```
$ color='red white and blue'
$ echo 'the value of \color is $color'
the value of \color is $color

$ echo 'the value of $color is' $color
the value of $color is red white and blue

$ echo 'this doesn't work'
>

$ echo '*****'
*****
```

Quoting - "

- " removes the special meaning of all characters surrounded by the double quotes except ", \ and \$, including \${*variable*} and \$(*command*).

```
$ color="red white and blue"
$ echo "the value of \color is $color"
the value of $color is red white and blue

$ curdir="$LOGNAME - Your current dir is $(pwd) "
$ echo $curdir
stu03 - Your current dir is /home/user3/tree

$ echo "they're all here: \\, ', \" "
they're all here: \, ', "
```

Lesson: Here documents

- What is a Here Document
- Using a Here Document
- Here Document Quoting
- Ignoring Leading Tabs

What is a Here Document

command << EOF

line 1
line 2
line 3
line 4
EOF

■ Example:

```
mail $( who | cut -f1 -d" " | sort -u) << EOF
This a mail message to say that I know you
are logged on.
Regards, Rick
EOF
```

Using a Here Document

- `${variable}` and `$(command)` expansion works in here documents.

```
mail $( who | cut -f1 -d" " | sort -u) << EOF
This a mail message to say that I know you
are logged on at $(date +"Date: %D Time: %T")
Regards, $LOGNAME
EOF
```

```
. . .
ftp -in $sysname << FTP_CMD
user "$uname" "$passwd"
put $filename /tmp/$filename
bye
FTP_CMD
```

Here Document Quoting

- To include the special characters, quote them:

```
$ cat << TEST
> Do I see a \$, a \', and a \\?
> TEST
Do I see a $, a ', and a \
```

- To remove all shell interpretation, just quote any part of the here document's marker:

```
$ cat << \TEST2
> Do I see a $, a ', and a \ ?
> TEST2
Do I see a $, a ', and a \ ?

$ write denise << 'x'
> Hey, what's the value of your $PATH?
> And what does 'uname -n' tell you?
> x
```


Ignoring Leading Tabs

- When you follow the << with a dash (-), then any leading tabs on lines of the here document will be ignored.

- Example:

```
for recipient
do
  mail $recipient <<- END_of_MEMO
    This a mail message to say that I know you
    are logged on at $(date +"Date: %D Time: %T")
    Regards, $LOGNAME
  END_of_MEMO
done
```

Review Exercises



- Complete the exercises from the Learning Guide

Topics for Review