1.109.2 Customize or write simple scripts Weight 3

Linux Professional Institute Certification — 102

Nick Urbanik <nicku@nicku.org>
This document Licensed under GPL—see section 18

2005 November

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

executable

script SUID?

rue and False

Shell Variable

Special Variables

uoting

Command Substitution

ie ii statement

hile <mark>statement</mark>

ne for statemen

ne test program

rithmetic

nout & Output

Alerting about problems by ema

icense Of This Document



Outline

Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement while statement The for statement The test program Conditions Arithmetic Input & Output Output with echo Input with read Alerting about problems by email License Of This Document

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

xecutable

script SUID?

rue and Faise

nell Variables

_4:__

Command Substitution

ie ii Statement

hile statement

ie ioi statement

The test program

Arithmetic

Input & Output

Alerting about problems by emails

License Of The Document

Topic 109 Shells, Scripting, Programming and Compiling [8] Where we are up to

1.109.1 Customize and use the shell environment [5]

1.109.2 Customize or write simple scripts [3]

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

he shebang: #!

Making the script executable

Should you make script SUID?

ue and False

Shell Variables

Special Variables

uoting

Command Substitution

he if statement

nile statement

.

test program

Arithmetic

Innut & Outnu

Alerting about

Alerting about problems by emains

icense Of This Occument



Description of Objective

1.109.2 Customize or write simple scripts [3]

Candidate should be able to customize existing scripts, or write simple new (ba)sh scripts. This objective includes using standard sh syntax (*loops*, *tests*), using command substitution, testing command return values, testing of file status, and conditional *mailing to the superuser*. This objective also includes making sure the correct interpreter is called on the first (#!) line of scripts. This objective also includes managing *location*, *ownership*, execution and *suid-rights* of scripts.

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The sheba

Making the scri executable

cript SUID?

Shell Variables

uoting

.

he if statement

while statemer

The test prod

The test prog

Arithmeti

Input & Outpu

Alerting about problems by email

License Of This

Key files, terms, and utilities include:

1.109.2 Customize or write simple scripts [3]

while — shell builtin: does things repetively while a condition is true

for — shell builtin: does things repetively, once with each element of a list

test — used to construct a condition

chmod — an external command, to change the permission on a file

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script

script SUID?

rue and False

Shell Variables

.....

Command Substitution

hile statement

he for statemen

he test program

rithmetic

nnut & Outnut

Alerting about problems by ema

License Of The Document



The Shebang: #!

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objective

The shebang: #!

Making the script executable

cript SUID?

True and False

oricii variabics

oting

ommand Substitution

The if statement

hile statement

ne for statement

he test program

.

Alerting about problems by em

License Of Th Document

- You ask the Linux kernel to execute the shell script
- kernel reads first two characters of the executable file
 - If first 2 chars are "#!" then
 - kernel executes the name that follows, with the file name of the script as a parameter
- ▶ Example: a file called find.sh has this as the first line:
 - #! /bin/sh
- then kernel executes this:

/bin/sh find.sh

- What will happen in each case if an executable file begins with:
 - #! /bin/rm
 - ▶ #! /bin/ls

Making the script executable

Customize or write simple scripts Weight 3

1.109.2

Nick Urbanik

Context

The chebana: #1

Making the script executable

script SUID?

True and False

Sileii variables

uoting

Command Substitution

he if statement

hile statement

e test progran

ino cest progr

Arithmetic

Input & Output

Alerting about problems by email

License Of Thi Document

To easily execute a script, it should:

- ▶ be on the PATH
- have execute permission.

How to do each of these?

Red Hat Linux by default, includes the directory ~/bin on the PATH, so create this directory, and put your scripts there:

\$ mkdir ~/bin ←

If your script is called script, then this command will make it executable:

\$ chmod +x script \hookleftarrow

Should you make a script SUID?

► Normally, when *you* run a script, the process is owned by *you*, and has the *same access rights as you*

- If a script has the SUID permission, then:
 - ▶ it does not matter who executes it!
 - the owner of the process is the owner of the file
 - This is very dangerous, especially if the owner of the file is root!
- Never make a shell script SUID, unless you really, really know what the risks are and how to avoid them
- ▶ Instead, write it in a language such as Perl, with taint checking, and make it as simple as possible.
- ► See Topic 1.114.1 Perform security administration tasks for details of manipulating SUID/SGID permissions.

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

Making the script executable

Should you make a script SUID?

rue and False

Shell Variables

ecial Variable

oting

ommand Substitution

he if statement

he for statemen

The test progr

Arithmeti

Input & Output

Alerting about problems by ema

License Of Th Document



True and False

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The Shebang. #

executable

script GOID:

True and False

Shell Variables

_41__

mmand Substitution

in if statement

i Le statemen

itte Statemen

e ioi statement

ne test progra

Arithmetic

nout & Output

Alerting about problems by emains

License Of TI Document

- Shell programs depend on executing external programs
- When any external program execution is successful, the exit status is zero, 0
- ► An error results in a non-zero error code
- ▶ To match this, in shell programming:
 - ► The value 0 is true
 - any non-zero value is false
- ▶ This is opposite from other programming languages

Variables—1

simple scripts Weight 3 Nick Hrhanik

1.109.2

Customize or write

Shell Variables

- Variables not declared; they just appear when assigned to
- Assignment:
 - no dollar sign
 - no space around equals sign
 - examples:

 $\dot{S} = 10$ # correct

\$ x = 10 # wrong: try to execute program called "x"

- Read value of variable:
 - put a '\$' in front of variable name
 - example:
 - \$ echo "The value of x is \$x"

Variables—Assignments

- ➤ You can put *multiple assignments* on one line: i=0 j=10 k=100
- You can set a variable temporarily while executing a program:
 - \$ echo \$EDITOR
 emacsclient.
 - \$ EDITOR=gedit crontab -e
 - \$ echo \$EDITOR
 emacsclient.

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The Shebang. #:

Should you make

cript SUID?

True and False

Shell Variables

intina

Name of Colorations

Jiiiialia Substitutioi

nile statement

ne test program

Arithmotic

onut 9 Outnut

Alerting about

License Of This Document



Variables—Local to Script

- Variables disappear after a script finishes
- Variables created in a sub shell disappear
 - parent shell cannot read variables in a sub shell
 - example:

```
$ cat variables
```

#! /bin/sh

echo \$HOME

HOME=happy

echo \$HOME

\$./variables

/home/nicku

happy

\$ echo \$HOME

/home/nicku

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objective

The shebang: #

Making the script executable

script SUID?

rue and False

Shell Variables

Special Variables

uoting

Command Substitution

hile statement

ne ioi statement

ine test progra

Arithmeti

Input & Outpu

Alerting about problems by ema

License Of Th Document



Variables—unsetting Them

► You can make a variable hold the null string by assigning it to nothing, but it does not disappear totally: \$ VAR= ←

```
$ env | grep '^VAR' ← VAR=
```

You can make it disappear totally using unset:

```
$ unset VAR ←
```

\$ env | grep '^VAR' ←

1.109.2 Customize or write simple scripts Weight 3

Nick Hrhanik

Context

Objectives

The shebang: #

Making the script xecutable

script SUID?

Shell Variables

.......

Command Cubatitution

ommanu Substitution

ile statement

a tast program

le test progra

Arithmetic

nput & Output

Alerting about problems by ema

License Of Th Document



Command-line Parameters

1.109.2 Customize or write simple scripts Weight 3

Nick Hrbanik

Context

Objectives

laking the scrip

cript SUID?

Shell Variables
Special Variables

uoting

Command Substitution

ne if statement

nile statement

e test prograi

Arithmetic

Input & Output

lerting about roblems by ema

License Of This Document

► Command-line parameters are called \$0, \$1, \$2, ...

Example: when call a shell script called "shell-script" like this:

\$ shell-script param1 param2 param3 param4 \hookleftarrow

variable	value
\$0	shell-script
\$1	paraml
\$2	param2
\$3	param3
\$4	param4
\$#	number of parameters to the program, e.g., 4

Note: these variables are read-only.

Special Built-in Variables

1.109.2 Customize or write simple scripts Weight 3

Nick Hrhanik

ontext

Objectives

Making the script executable

script SUID?

Shell Variables

Special Variables

uoting

Command Substitution

The if statement

hile statement

he for statement

The test progra

Arithmetic

Alerting about

License Of This

▶ Both \$@ and \$* are a list of all the parameters.

➤ The only difference between them is when they are quoted in quotes—see manual page for bash

\$? is exit status of last command

\$\$ is the process ID of the current shell

► Example shell script:

```
#! /bin/sh
echo $0 is the full name of this shell script
echo first parameter is $1
echo first parameter is $2
echo first parameter is $3
echo total number of parameters is $#
echo process ID is $$
```

Outline

Context
Objectives
The shebang: #!
Making the script executable
Should you make a script SUID?
True and False
Shell Variables
Special Variables
Quoting

Quoting and Funny Chars

The if statement
while statement
The for statement
The test program
Conditions
Arithmetic
Input & Output
Output with echo
Input with read
Alerting about problems by email
License Of This Document

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Contex

Objective

The shebang: #!

Making the script

Should you make script SLIID?

rue and False

hell Variables

Special Variables

otina

Quoting and Funny Chars Quoting

Command Substitution

The if statemen

THO II OLULOTIO

The for statemen

he test progran

Arithmetic

7 11 11 11 11 11 11 11 11

Alerting about problems by ema

Special Characters
Many characters have a special meaning to the shell

Character	Meaning
~	Home directory
`	Command substitution. Better: \$ ()
#	Comment
\$	Variable expression
&	Background Job
*	File name matching wildcard
	Pipe
(Start subshell
)	End subshell
[Start character set file name matching
]	End character set file name matching
{	Start command block
;	Command separator
\	Quote next character
,	Strong quote
"	Weak quote
<	Redirect Input
>	Redirect Output
/	Pathname directory separator
?	Single-character match in filenames
!	Pipline logical NOT
⟨space or tab⟩	shell normally splits at white space

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Quoting and Funny Chars

Outline

Quoting

Quoting

Output with echo Input with read

1.109.2 Customize or write simple scripts Weight 3

Nick Hrhanik

Quotina

Quoting

- Sometimes you want to use a special character literally;
 i.e., without its special meaning.
- Called quoting
- ► Suppose you want to print the string: 2 * 3 > 5 is a valid inequality?
- ▶ If you did this:
 - \$ echo 2 * 3 > 5 is a valid inequality
 the new file '5' is created, containing the character '2',
 then the names of all the files in the current directory, then
 the string "3 is a valid inequality".

1.109.2 Customize or write simple scripts Weight 3

Nick Hrhanik

Context

Objectives

Making the script

hould you make a cript SUID?

frue and False

onell variables

pecial Variables

Quoting

Quoting and Funny Chars

Quoting

ioting

Sommand Substitution

ne ii statement

- ·

THE IOI Statemen

.

Alerting about

Quoting—2

- ➤ To make it work, you need to protect the special characters '*' and '>' from the shell by quoting them. There are three methods of quoting:
 - Using double quotes ("weak quotes")
 - Using single quotes ("strong quotes")
 - Using a backslash in front of each special character you want to quote
- This example shows all three:

```
$ echo "2 * 3 > 5 is a valid inequality"
```

- \$ echo '2 * 3 > 5 is a valid inequality'
- \$ echo 2 \times 3 > 5 is a valid inequality

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

Making the script executable

script SUID?

ue and raise

ieii variabies

Quoting Quoting and Funny Chars

Quoting

Sommand Substitution

e ii statement

while statement

he for statemer

The + - - + pregr

rithmetic

Inner d O Out

Alerting about problems by ema

Quoting—When to use it?

- Use quoting when you want to pass special characters to another program.
- Examples of programs that often use special characters:
 - ▶ find, locate, grep, expr, sed **and** echo
- Here are examples where quoting is required for the program to work properly:

```
$ find . -name \*.jpg
```

- \$ locate '/usr/bin/c*'
- \$ grep 'main.*(' *.c
- \$ i=\$(expr i * 5)

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

laking the perint

Should you r script SUID?

rue and False

hell Variables

ecial Variables

oting and Funny

Quoting

Command Substitution

he if statement

.

he for statemer

-

Arithmetic

.

problems by em

More about Quoting

- Double quotes: "..." stop the special behaviour of all special characters, except for:
 - variable interpretation (\$)
 - ▶ backticks (`) see slide 6
 - ▶ the backslash (\)
- ▶ Single quotes ' . . . ' :
 - stop the special behaviour of all special characters
- Backslash:
 - preserves literal behaviour of character, except for newline; see slides §5, §10
 - Putting "\" at the end of the line lets you continue a long line on more than one physical line, but the shell will treat it as if it were all on one line.

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

ne snebang: #

Should vou m

.... and Calas

Shell Variables

nocial Variables

luoting Quoting and Funny Chars

Quoting

Command Substitution

e if statement

o ii statomont

he for statemen

rithmetic

i iti ii ii ii ii

input & Output

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

ontext

Objectives

The shebang: #

Making the script executable

script SUID?

e and False

onell variables

Special Variables

oting

Command Substitution

The it statement

while statement

he test program

rie test progra

Alerting about problems by em

License Of Thi Document

- ► Enclose command in \$ (...) or backticks: `... `
- ► Means, "Execute the command in the \$ (...) and put the output back here."
- ► Here is an example using expr:

```
$ expr 3 + 2
```

5

\$ i=expr 3 + 2 # error: try execute command '3'

\$ i=\$(expr 3 + 2) # correct

\$ **i='expr** 3 + 2' # also correct

Command Substitution—Example

We want to put the output of the command hostname into a variable:

```
$ hostname
nicku.org
```

\$ h=hostname

\$ echo \$h

hostname

- ➤ Oh dear, we only stored the *name* of the command, not the *output* of the command!
- Command substitution solves the problem:

```
$ h=$(hostname)
```

\$ echo \$h
nicku.org

▶ We put \$ (...) around the command. You can then assign the output of the command.

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objective:

The chelesian

Making the script

Should you n cript SUID?

e and False

hell Variables

_41__

.

Command Substitution

ine ii statement

while statement

ne ioi statement

me cest program

Arithmetic

Input & Outpu

lerting about roblems by ema

License Of T Document



if Statement

Syntax:

```
if \langle test-commands \rangle then
    \statements-if-test-commands-1-true \rangle elif \langle test-commands-2 \rangle then
    \statements-if-test-commands-2-true \rangle else
    \langle statements-if-all-test-commands-false \rangle fi
```

Example:

```
if grep nick /etc/passwd > /dev/null 2>&1
then
    echo Nick has a local account here
else
    echo Nick has no local account here
fi
```

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objective:

The shebang: #!

Making the script executable

Should you make script SUID?

rue and False

Shell Variables

ecial Variables

oting

Command Substitution

The if statement

hile statemen

The for statement

he test progra

Arithmet

Input & Outp

Alerting about problems by ema

License Of This Document



while Statement

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Object

```
while \langle test-commands \rangle do \quad \langle loop-body-statements \rangle done
```

Example:

Syntax:

```
i=0
while [ "$i" -lt 10 ]
do
    echo -n "$i "  # -n suppresses newline.
    let "i = i + 1"  # i=$(expr $i + 1) also works
done
```

ontext

Objectives

The shebang: # !

Making the script

Should you make

rue and False

hell Variables

nacial Variables

ıotina

ommand Substitution

ie ii stateriierit

while statement

The for statement

he test progran

Arithmetic

Input & Output

Alerting about problems by ema

icense Of This Document

for Statement

Syntax:

```
for (name) in (words)
do
     (loop-body-statements)
done
```

Example:

```
for planet in Mercury Venus Earth Mars \
    Jupiter Saturn Uranus Neptune Pluto
do
    echo $planet
```

done

► The backslash "\" quotes the newline. It's just a way of folding a long line in a shell script over two or more lines.

1.109.2 Customize or write simple scripts Weight 3

Nick Hrhanik

The for statement



Making the script executable

Should you make script SUID?

rue and False

Juotina

mmand Substitution

-- :/ ------

il - atatament

The for statement

The test program

A -: 41---- - 4:-

lancat 0 Octo

Alerting about problems by ema

License Of This Document

Here the shell turns *.txt into a list of file names ending in ".txt":

```
for i in *.txt
do
     echo $i
     grep 'lost treasure' $i
done
```

You can leave the in ⟨words⟩ out; in that case, ⟨name⟩ is set to each parameter in turn:

```
i=0
for parameter
do
    let 'i = i + 1'
    echo "parameter $i is $parameter"
done
```

Outline

Context
Objectives
The shebang: #!
Making the script executable
Should you make a script SUID'
True and False
Shell Variables
Special Variables
Quoting
Quoting and Funny Chars

The if statement
while statement
The for statement
The test program
Conditions
Arithmetic
Input & Output
Output with echo
Input with read
Alerting about problems by email
License Of This Document

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Contex

Objective

The shehand: #1

Making the script

Should you make

True and False

hell Variables

Special Variables

oting

Command Substitution

ne it statement

while statement

he for statemer

The test program

Arithmetic

nout & Outr

Alerting about

cense Of Ti

statements and for while loops

Examples of string comparisons:

either test or [...]

"\$USER" = root 1

-z "SUSER" 1

"\$USER" != root]

true if the string "\$USER" has zero length string1 \< string2 | # true if string1 sorts less than string2 string1 \> string2] # true if string1 sorts greater than string2

Note that we need to quote the '>' and the '<' to avoid</p> interpreting them as file redirection.

All programming languages depend on conditions for if

true if the value of \$USER is "root"

true if the value of \$USER is not "root"

Shell programming uses a built-in command which is

- Note: the spaces after the "[" and before the "]" are essential.
- Also spaces are essential around operators

Conditions

Conditions—Integer Comparisons

Examples of *numeric* integer comparisons:

```
[ "xx" -eq 5 ] # true if the value of x is 5 [ "xx" -ne 5 ] # true if integer x is not 5 [ "xx" -lt 5 ] # true if integer x is x 5 [ "xx" -gt 5 ] # true if integer x is x 5 [ "xx" -le 5 ] # true if integer x is x 5 [ "xx" -ge 5 ] # true if integer x is x 5
```

- Note again that the spaces after the "[" and before the "]" are essential.
- Also spaces are essential around operators

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

ne snebang: # !

executable

script SUID?

ilue aliu raise

Shell Variables

ıotina

Command Substitutio

ommand Substitution

ile statement

.

The test program

Conditions

rithmetic

Alerting about

Making the script executable

cript SUID?

Shell Variables

Quoting

Command Substitution

hile statement

hile statement

ho toot statement

The test progra Conditions

rithmetic

Innuit 0 Ou

Alerting about problems by email

License Of The Document

- ▶ The shell provides many tests of information about *files*.
- ▶ Do man test to see the complete list.
- ▶ Some examples:

```
[ -f file ]
              # true if file is an ordinary file
 ! -f file | # true if file is NOT an ordinary file
[ -d file ]
              # true if file is a directory
[ -u file ]
              # true if file has SUID permission
[ -a file ]
              # true if file has SGID permission
[ -x file ]
              # true if file exists and is executable
[ -r file ]
              # true if file exists and is readable
[ -w file ]
              # true if file exists and is writeable
[ file1 -nt file2 ] # true if file1 is newer than file2
```

- Note again: the spaces after the "[" and before the "]" are essential.
- Also spaces are essential around operators

ne shebang: # aking the scrip

Should you make

True and False

Shell Variables

luoting

Command Substitution

The if statement

hile statement

he test program

Conditions

Arithmetic

Alerting about

License Of T

Examples of combining comparisons with AND: -a and OR: -o, and grouping with \ (...\)

```
# true if the value of \$x is 5 AND \$USER is not equal to root: [ "\$x" -eq 5 -a "\$USER" != root ] # true if the value of \$x is 5 OR \$USER is not equal to root: [ "\$x" -eq 5 -o "\$USER" != root ] # true if (the value of \$x is 5 OR \$USER is not equal to root ) AND # ( \$y > 7 OR \$HOME has the value happy ) [ \ ( "\$x" -eq 5 -o "\$USER" != root \ ) -a \ \ ( "\$y" -gt 7 -o "\$HOME" = happy \ ) ]
```

- Note again that the spaces after the "[" and before the "]" are essential.
- Do man test to see the information about all the operators.

Arithmetic Assignments

- Can do with the external program expr
 - ...but expr is not so easy to use, although it is very standard and portable: see man expr
 - ▶ Easier is to use the built in let command
 - see help let
 - Examples:

```
$ let x=1+4
$ let ++x  # Now x is 6
$ let x='1 + 4'
$ let 'x = 1 + 4'
$ let x="(2 + 3) * 5"  # now x is 25
$ let "x = 2 + 3 * 5"  # now x is 17
$ let "x += 5"  # now x is 22
$ let "x = x + 5"  # now x is 27; NOTE NO
```

- Notice that you do not need to quote the special characters with let.
- Quote if you want to use white space.
- Do not put a dollar in front of variable, even on right side of assignment; see last example.

1.109.2 Customize or write simple scripts Weight 3

Nick Hrbanik

Context

Objectives

ne snebang: # !

Should you mak

... and False

Shall Madalala

Spacial Variables

uoting

Command Substitution

he if statement

hile statement

0 to at program

The test program

Arithmetic

Input & Outp

Alerting about problems by ema

License Of Document



Outline

Context
Objectives
The shebang: #!
Making the script executable
Should you make a script SUID?
True and False
Shell Variables
Special Variables
Quoting
Quoting and Funny Chars

Command Substitution
The if statement
while statement
The for statement
The test program
 Conditions
Arithmetic
Input & Output
 Output with echo
Input with read

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Contex

Objectives

The shebang: #!

Making the script

Should you make

rue and False

hall Variables

Special Variables

intina

Command Substitution

e if statement

while statement

mille statement

The test program

Arithmetic

Input & Output
Output with echo

lerting about

Output with echo

- ► To perform output, use echo, or for more formatting, printf.
- ▶ Use echo -n to print no newline at end.
- ▶ Just echo by itself prints a newline

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objective

The shebang: #!

Making the script executable

script SUID?

True and False

Shell variables

Special Variables

uoting

Command Substitution

ne ir statement

hile statement

nite statement

he test prograi

ne test progran

Arithmetic

Input & Output
Output with echo

Input with read

problems by en

Outline

Context
Objectives
The shebang: #!
Making the script executable
Should you make a script SUID?
True and False
Shell Variables
Special Variables
Quoting
Quoting and Funny Chars

Command Substitution
The if statement
while statement
The for statement
The test program
Conditions
Arithmetic
Input & Output
Output with echo
Input with read
Alerting about problems by email

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Contex

Objectives

The shebang: #!

Making the script

Should you make

True and False

hell Variables

Special Variables

otina

Command Substitution

he if statement

hile statement

The for statemen

The test program

Arithmetic

Input & Output
Output with echo

Input with read
Alerting about

iconso Of This

Input: the read Command

- For input, use the built-in shell command read
- read reads standard input and puts the result into one or more variables
- ▶ If use one variable, variable holds the whole line
- Syntax:

```
read \langle var1 \rangle \dots
```

▶ Often used with a while loop like this:

```
while read var1 var2 do
```

do something with \$var1 and \$var2 done

- ▶ Loop terminates when reach end of file
- To prompt and read a value from a user, you could do:

```
while [ -z "$value" ]; do
    echo -n "Enter a value: "
    read value
done
# Now do something with $value
```

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #!

Making the script executable

ould you make a ript SUID?

rue and False

ileli variables

uoting

Command Substitution

hile statement

ne for statement

ne test program

Arithmetic

Input & Output
Output with echo
Input with read

lerting about

cense Of This

1.109.2

Customize or write

```
| grep -i "shell script" | awk -F: '{print $1}'
Objectives
```

grep -i "shell script" | awk -F:

Let's see how this works. I suggest executing the commands separately to see what they do:

```
$ file /bin/* /usr/bin/*
$ file /bin/* /usr/bin/* | grep -i "shell script"
```

\$ file /bin/* /usr/bin/*

The awk program is actually a complete programming language. It is mainly useful for selecting columns of data from text.

awk automatically loops through the input, and divides the input lines into fields. It calls these fields $\$1, \$2, \dots \$NF. \0 contains the whole line. Here the option -F: sets the *field separator* to the colon character. Normally it is any white space. So printing \$1 here prints what comes before the colon, which is the file name.

Should you make a script SUID?

'{printls\$1}'
Shell Variables
Special Variables

command Substitution the if statement

le statement for statement

ithmetic out & Output

Output with echo
Input with read

blems by email

Suppose you want to look for all shell scripts containing a particular command or statement? Looking for example shell scripts that use the mktemp command:

```
$ file /bin/* /usr/bin/* /usr/sbin/* /sbin/* /etc/rc.d/* /usr/X11R66bin/*
 grep -i 'shell script' | awk -F: '{print $1}' | xargs grep mkt@simple scripts
                                                                         Weight 3
                                                                        Nick Urbanik
```

Alerting about

problems by email

Alerting about problems by email

```
1.109.2
Customize or write
simple scripts
Weight 3
```

Nick Urbanik

```
#! /bin/sh
# A quick script whipped up by Nick to send mail if
# root file system is more than 90 per cent full.

percentful=$(df / | awk 'NR > 1{sub("%", "", $5);print $5}')
if [ "$percentful" -gt 90 ]
then
    message="root file system is $percentful% full"
    echo "$message" | mail -s $message root
fi
```

Context

Objective

The chebona: #

laking the script kecutable

Should you make

ue and False

Shell Variabl

Special Variables

otina

mmand Substitution

he if statement

hile statement

ne for statement

e test program

rithmetic

Input & Output

Alerting about problems by email

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Shiootivo

The shebang: #!

Making the script

Should you make

True and False

Shell Variables

Special Variables

oting

Command Substitution

The if statement

while statement

will be statement

he test program

. . . .

Antimieno

Alerting about problems by email

License Of This Document

Context

Objectives
The shebang: #!
Making the script executable
Should you make a script
SUID?

Shell Variables
Special Variables
Quoting
Quoting and F

Quoting and Funny Chars
Quoting

<ロ > < 個 > < 重 > < 重 > の< で

Context Objectives

The shebang: #!
Making the script executable
Should you make a script
SUID?

Shell Variables
Special Variables
Quoting

Quoting and Funny Chars
Quoting

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Ob:---

The shebang: #

Making the script

Should you make

rue and False

Shell Variables

pecial Variables

oting

Command Substitution

ne it statement

while statement

ho for statement

he test program

Arithmetic

Input & Output

Alerting about problems by email



1.109.2 Customize or write simple scripts Weight 3

Nick Hrhanik

Alerting about problems by email

Context Objectives The shebang: #!

4 D F 4 D F 4 D F 4 D F

Context Objectives The shebang: #! Making the script executable

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objective

Making the script

Should you make

ue and False

hell Variables

ecial Variables

oting

Command Substitution

The if statement

while statement

The feet and the control of the cont

The test program

Arithmetic

Input & Output

Alerting about problems by email

Context Objectives The shebang: #! Making the script executable Should you make a script SUID?

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objective

The shebang: # !

Making the script executable

Should you make

rue and False

hell Variables

pecial Variables

oting

Command Substitution

The if statement

while statement

mille statement

The test program

......

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objective:

The shebang: #

Making the script executable

Should you make

rue and False

hell Variables

pecial Variables

oting

Command Substitution

ine it statement

while statement

The for statement

The test program

Arithmetic

Input & Output

Alerting about problems by email



Context

Objectives
The shebang: #!
Making the script executable
Should you make a script
SUID?
True and False
Shell Variables
Special Variables
Quoting
Quoting and Funny Chars

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script executable

Should you make

rue and False

hell Variables

Special Variables

oting

Command Substitution

he if statement

while statement

- ·

The test program

Arithmetic

Input & Outpu

Alerting about problems by email



Context
Objectives
The shebang: #!
Making the script executable
Should you make a script
SUID?
True and False
Shell Variables
Special Variables
Quoting

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The chebana: #

Making the script

Should you make

up and Falso

hell Variables

pecial Variables

oting

Command Substitution

ine it statement

while statement

The test program

Arithmetic

Input & Output

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script executable

Should you make

ue and False

hell Variables

Special Variables

uoting

Command Substitution

ne it statement

while statement

he for statement

The test program

Arithmetic

Input & Output

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script executable

Should you make

ue and False

hell Variables

Special Variables

oting

Command Substitution

ne it statement

while statement

he test program

Arithmetic

Input & Output

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script

Should you make

ue and False

hell Variables

Special Variables

uoting

Command Substitution

he if statement

while statement

riffe statement

he test program

Arithmetic

Input & Output

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution

The if statement
while statement
The for statement
The test program
Conditions
Arithmetic
Input & Output
Output with echo
Input with read
Alerting about problems b
email

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script executable

hould you make cript SUID?

ue and False

hell Variables

Special Variables

oting

Command Substitution

ne it statement

hile statement

he test program

rithmetic

Input & Outpu

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement

The for statement
The test program
Conditions
Arithmetic
Input & Output
Output with echo
Input with read
Alerting about problems by email

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script

cript SUID?

ue and False

nell Variables

pecial Variables

oting

Command Substitution

he if statement

while statement

7---------

he test program

Arithmetic

Input & Output

Alerting about problems by email



Context

Objectives
The shebang: #!
Making the script executable
Should you make a script
SUID?
True and False
Shell Variables
Special Variables
Quoting
Quoting and Funny Chars
Quoting

Command Substitution The if statement while statement

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The chehana: #

Making the script executable

script SUID?

ue and False

ieli variables

Jeciai valiable

ammand Cubatitution

The if etatement

nii e statomont

he test program

The test program

Alerting about

problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement while statement The for statement

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script executable

Should you ma

ue and False

nell Variables

pecial Variables

oting

Command Substitution

he if statement

while statement

he test program

Arithmetic

Input & Outpu

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement while statement The for statement The test program

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script executable

script SUID?

rue and False

nell Variables

ecial Variables

101.00

ommand Substitution

while statement

ne for statement

The test program

Arithmetic

Input & Output

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement while statement The for statement The test program Conditions

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script executable

script SUID?

ue and False

hell Variables

ecial Variables

oting

command Substitution

ne ii statement

hile statement

ne for statement

he test program

Arithmetic

Input & Output

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement while statement The for statement The test program Conditions Arithmetic

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shehana:

Making the script executable

script SUID?

rue and False

hell Variables

ecial Variables

ommand Substitution

he if statement

hile statement

nite statement

he test program

.

Input & Output

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement while statement The for statement The test program Conditions Arithmetic Input & Output

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the scrip executable

script SUID?

ide and raise

mon variables

oting

Command Substitution

he if statement

hile statement

. .

he test program

\rithmotic

Input & Output

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution
The if statement
while statement
The for statement
The test program
Conditions
Arithmetic
Input & Output
Output with echo
Input with read
Alerting about problems

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the scrip
executable

script SUID?

iue anu raise

ileli valiables

otina

Command Substitution

he if statement

hile statement

.

ne test program

.

......

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement while statement The for statement The test program Conditions Arithmetic Input & Output Output with echo Input with read

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the scrip executable

script SUID?

rue and False

Shell Variables

otina

Command Substitution

he if statement

hile statement

. .

ne test program

Arithmetic

Input & Outpu

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement while statement The for statement The test program Conditions Arithmetic Input & Output Output with echo Input with read Alerting about problems by email

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the scrip executable

script SUID?

rue and False

Shell Variables

ecial Variables

ioting

Command Substitution

he if statement

hile statement

.

The test program

Input & Output

Alerting about problems by email



Context Objectives The shebang: #! Making the script executable Should you make a script SUID? True and False Shell Variables Special Variables Quoting Quoting and Funny Chars Quoting

Command Substitution The if statement while statement The for statement The test program Conditions Arithmetic Input & Output Output with echo Input with read Alerting about problems by email License Of This Document 1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

The shebang: #

Making the script executable

script SUID?

rue and False

Shell Variables

ecial Variables

Joting

Command Substitution

The if statement

hile statement

ie ioi statement

rne test progra

rithmetic

Input & Output

Alerting about problems by email

License Of This Document

Copyright © 2005 Nick Urbanik <nicku@nicku.org>
You can redistribute modified or unmodified copies of this
document provided that this copyright notice and this
permission notice are preserved on all copies under the terms
of the GNU General Public License as published by the Free
Software Foundation — either version 2 of the License or (at
your option) any later version.

1.109.2 Customize or write simple scripts Weight 3

Nick Urbanik

Context

Objectives

he shebang: #

Making the scrip executable

script SUID?

rue and False

Shell Variables

ecial Variables

oung

Command Substitution

he if statement

le statement

e for statement

e test progran

Arithmetic

nout & Outout

Alerting about

