

## SECTION 6      Work with the Linux Shell and Edit Text Files

In this section of the workbook, you learn how to do the following:

- “Execute Commands at the Command Line” on 6-2
- “Create a Shell and an Environment Variable and Examine the Exit Status” on 6-3
- “Use the alias Command” on 6-5
- “Use the alias Command” on 6-5
- “Work with Command Syntax and Special Characters” on 6-6
- “Use Piping and Redirection” on 6-8
- “Use vi to Edit Files in the Linux System” on 6-10
- “Optional: Work through the vim Tutorial” on 6-12

## SECTION 6      Work with the Linux Shell and Edit Text Files

In this section of the workbook, you learn how to do the following:

- “Execute Commands at the Command Line” on 6-2
- “Create a Shell and an Environment Variable and Examine the Exit Status” on 6-3
- “Use the alias Command” on 6-5
- “Use the alias Command” on 6-5
- “Work with Command Syntax and Special Characters” on 6-6
- “Use Piping and Redirection” on 6-8
- “Use vi to Edit Files in the Linux System” on 6-10
- “Optional: Work through the vim Tutorial” on 6-12

### **Exercise 6-1    Execute Commands at the Command Line**

By now you have of course already executed some commands at the command line. The purpose of this exercise is to highlight some of the features of the shell that make using the command line easier.

To execute commands at the command line, do the following:

1. Open a terminal window.
2. View the history cache by entering **history**.
3. Press **Up-arrow** until you see an ls command you would like to execute; then press **Enter**.
4. Type **h** and press **Page Up** once.

You should see the command history at the command line again.

5. Press **Enter** to execute the **history** command.
6. Switch to root by entering **sux -**; then enter a password of **novell**.
7. Start YaST by entering **yast2**.  
YaST should start in QT mode.
8. Quit YaST by selecting **Close**.
9. Become the user geeko again by entering **exit**.
10. Su to root by entering **su -**; then enter a password of **novell**.
11. Check to make sure you are logged in as root by entering **id**.
12. Start YaST by entering **yast2**.  
YaST should start in ncurses mode.
13. Quit YaST by entering **Alt + q**.
14. Log out as root by entering **exit**.
15. Close the terminal window by entering **exit**.

**(End of Exercise)**

### **Exercise 6-1    Execute Commands at the Command Line**

By now you have of course already executed some commands at the command line. The purpose of this exercise is to highlight some of the features of the shell that make using the command line easier.

To execute commands at the command line, do the following:

1. Open a terminal window.
2. View the history cache by entering **history**.
3. Press **Up-arrow** until you see an ls command you would like to execute; then press **Enter**.
4. Type **h** and press **Page Up** once.

You should see the command history at the command line again.

5. Press **Enter** to execute the **history** command.
6. Switch to root by entering **sux -**; then enter a password of **novell**.
7. Start YaST by entering **yast2**.  
YaST should start in QT mode.
8. Quit YaST by selecting **Close**.
9. Become the user geeko again by entering **exit**.
10. Su to root by entering **su -**; then enter a password of **novell**.
11. Check to make sure you are logged in as root by entering **id**.
12. Start YaST by entering **yast2**.  
YaST should start in ncurses mode.
13. Quit YaST by entering **Alt + q**.
14. Log out as root by entering **exit**.
15. Close the terminal window by entering **exit**.

**(End of Exercise)**

**Exercise 6-2 Create a Shell and an Environment Variable and Examine the Exit Status**

Variables and the exit status are more frequently used within shell scripts than directly on the command line. Within scripts they are, however, vital.

The purpose of this exercise is to show you the difference between shell and environment variables and how to use them.

To create a shell and an environment variable and examine the exit status, do the following:

- [Part I: Create a Shell and an Environment Variable](#)
- [Part II: Examine the Exit Status](#)

**Part I: Create a Shell and an Environment Variable**

Do the following:

1. As user geeko, open a terminal window and create a shell variable called USERNAME that holds your user name by entering

**USERNAME=your\_name**

2. Log in as root by entering **su -** and a password of **novell**.
3. What is the content of the variable USERNAME?

(Find out by entering **echo \$USERNAME**.)

4. Return to the original shell of the user by entering **exit**.
5. Change the variable USERNAME to an environment variable by entering

**export USERNAME**

**Exercise 6-2 Create a Shell and an Environment Variable and Examine the Exit Status**

Variables and the exit status are more frequently used within shell scripts than directly on the command line. Within scripts they are, however, vital.

The purpose of this exercise is to show you the difference between shell and environment variables and how to use them.

To create a shell and an environment variable and examine the exit status, do the following:

- [Part I: Create a Shell and an Environment Variable](#)
- [Part II: Examine the Exit Status](#)

**Part I: Create a Shell and an Environment Variable**

Do the following:

1. As user geeko, open a terminal window and create a shell variable called USERNAME that holds your user name by entering

**USERNAME=your\_name**

2. Log in as root by entering **su -** and a password of **novell**.
3. What is the content of the variable USERNAME?

(Find out by entering **echo \$USERNAME**.)

4. Return to the original shell of the user by entering **exit**.
5. Change the variable USERNAME to an environment variable by entering

**export USERNAME**

6. Switch to user root again by entering **su -** and a password of **novell**.
7. What is the content of the variable USERNAME now?
8. *(Optional)* Change the primary bash prompt (variable PS1) so it displays the current time. (See **man bash** and search for PROMPTING. Several solutions are possible, such as **PS1="\u@\h - \A :lw> "**)

## Part II: Examine the Exit Status

Do the following:

1. As a regular user, enter **ls** in your home directory.
2. Check the exit status by entering **echo \$?**
3. As a regular user, enter **ls** for the home directory of user root (/root/).
4. Check the exit status again.

*(End of Exercise)*

6. Switch to user root again by entering **su -** and a password of **novell**.
7. What is the content of the variable USERNAME now?
  
8. *(Optional)* Change the primary bash prompt (variable PS1) so it displays the current time. (See **man bash** and search for PROMPTING. Several solutions are possible, such as **PS1="\u@\h - \A :lw> "**)

## Part II: Examine the Exit Status

Do the following:

1. As a regular user, enter **ls** in your home directory.
2. Check the exit status by entering **echo \$?**
3. As a regular user, enter **ls** for the home directory of user root (/root/).
4. Check the exit status again.

*(End of Exercise)*

**Exercise 6-3 Use the alias Command**

You will most likely get along well without aliases, but they can make life easier. An alias for a long command can sometimes come quite handy.

To get familiar with the alias command, do the following:

1. Open a terminal window.
2. View all defined aliases by entering  
**alias**
3. Define a new alias by entering  
**alias hello='echo "hello \$USER"'**
4. Check the functionality of the alias hello by entering  
**hello**
5. Check the command type of the command hello by entering  
**type hello**
6. Remove the alias by entering  
**unalias hello**
7. Close the terminal window.

***(End of Exercise)***

**Exercise 6-3 Use the alias Command**

You will most likely get along well without aliases, but they can make life easier. An alias for a long command can sometimes come quite handy.

To get familiar with the alias command, do the following:

1. Open a terminal window.
2. View all defined aliases by entering  
**alias**
3. Define a new alias by entering  
**alias hello='echo "hello \$USER"'**
4. Check the functionality of the alias hello by entering  
**hello**
5. Check the command type of the command hello by entering  
**type hello**
6. Remove the alias by entering  
**unalias hello**
7. Close the terminal window.

***(End of Exercise)***

### **Exercise 6-4 Work with Command Syntax and Special Characters**

More often than not you will want to manipulate more than one file at a time - move all .txt files below a certain directory somewhere else, etc. This is where special characters come into play and the purpose of this exercise is to get you used to them. Play around with these special characters to really get familiar with them.

To understand how to use command syntax and special characters, do the following:

1. Open a terminal window.
2. List all filenames in the directory /bin/ that start with the character "a" by entering  
**ls /bin/a\***
3. List all file names in the directory /bin/ that consist of 4 characters by entering  
**ls /bin/????**
4. List all filenames in the directory /bin/ that consist of 4 or more characters by entering  
**ls /bin/????\***
5. List all filenames in the directory /bin/ that do not start with one of the characters from a to r by entering  
**ls /bin/[!a-r]\***
6. Start the file manager Konqueror by selecting the *blue house* icon in Kicker.
7. Create a new file by right-clicking the file view window and selecting **Create New > File > Text File**.
8. Enter a filename of **My**; then select **OK**.
9. Create a new file by right-clicking the file view window and selecting **Create New > File > Text File**.
10. Enter a filename of **File**; then select **OK**.

### **Exercise 6-4 Work with Command Syntax and Special Characters**

More often than not you will want to manipulate more than one file at a time - move all .txt files below a certain directory somewhere else, etc. This is where special characters come into play and the purpose of this exercise is to get you used to them. Play around with these special characters to really get familiar with them.

To understand how to use command syntax and special characters, do the following:

1. Open a terminal window.
2. List all filenames in the directory /bin/ that start with the character "a" by entering  
**ls /bin/a\***
3. List all file names in the directory /bin/ that consist of 4 characters by entering  
**ls /bin/????**
4. List all filenames in the directory /bin/ that consist of 4 or more characters by entering  
**ls /bin/????\***
5. List all filenames in the directory /bin/ that do not start with one of the characters from a to r by entering  
**ls /bin/[!a-r]\***
6. Start the file manager Konqueror by selecting the *blue house* icon in Kicker.
7. Create a new file by right-clicking the file view window and selecting **Create New > File > Text File**.
8. Enter a filename of **My**; then select **OK**.
9. Create a new file by right-clicking the file view window and selecting **Create New > File > Text File**.
10. Enter a filename of **File**; then select **OK**.

11. Create a new file by right-clicking the file view window and selecting **Create New > File > Text File**.
12. Enter a filename of **My File**; then select **OK**.
13. Close the Konqueror window.
14. From the terminal window, list the files My and File by entering  
**ls -l My File**
15. List the file My File by entering  
**ls -l My\ File**
16. Remove the files My, File, and My File by entering  
**rm My File My\ File**
17. Verify that the files have been removed by entering **ls -l**.
18. Close the terminal window.

***(End of Exercise)***

11. Create a new file by right-clicking the file view window and selecting **Create New > File > Text File**.
12. Enter a filename of **My File**; then select **OK**.
13. Close the Konqueror window.
14. From the terminal window, list the files My and File by entering  
**ls -l My File**
15. List the file My File by entering  
**ls -l My\ File**
16. Remove the files My, File, and My File by entering  
**rm My File My\ File**
17. Verify that the files have been removed by entering **ls -l**.
18. Close the terminal window.

***(End of Exercise)***

### **Exercise 6-5 Use Piping and Redirection**

Piping the long output of some program into less to view it page by page is a frequently seen scenario. But there are many other uses of piping and redirecting, like sorting, counting lines, showing only certain columns of the output, etc.

To understand piping and redirection, do the following:

1. Open a terminal window.
2. Pipe the output of the ls command for the home directory (~) to a file by entering

```
ls ~ > home_directory
```

3. Display the content of the file by entering

```
cat home_directory
```

4. Append the output of the ls command for the root directory (/) to the file home\_directory by entering

```
ls / >> home_directory
```

5. Display the content of the file by entering

```
cat home_directory
```

6. Overwrite the file home\_directory with the output of the ls command by entering

```
ls / > home_directory
```

7. Display the content of the file by entering

```
cat home_directory
```

8. Write the output of the ls command on the screen and into the file home\_directory by entering

```
ls ~ | tee home_directory
```

9. Count the number of files in your home directory by entering

```
ls ~ | wc -l
```

### **Exercise 6-5 Use Piping and Redirection**

Piping the long output of some program into less to view it page by page is a frequently seen scenario. But there are many other uses of piping and redirecting, like sorting, counting lines, showing only certain columns of the output, etc.

To understand piping and redirection, do the following:

1. Open a terminal window.
2. Pipe the output of the ls command for the home directory (~) to a file by entering

```
ls ~ > home_directory
```

3. Display the content of the file by entering

```
cat home_directory
```

4. Append the output of the ls command for the root directory (/) to the file home\_directory by entering

```
ls / >> home_directory
```

5. Display the content of the file by entering

```
cat home_directory
```

6. Overwrite the file home\_directory with the output of the ls command by entering

```
ls / > home_directory
```

7. Display the content of the file by entering

```
cat home_directory
```

8. Write the output of the ls command on the screen and into the file home\_directory by entering

```
ls ~ | tee home_directory
```

9. Count the number of files in your home directory by entering

```
ls ~ | wc -l
```

10. Remove the file `home_directory` by entering  
**`rm home_directory`**
11. Verify that the file was removed by entering **`ls -l`**.
12. Close the terminal window.

***(End of Exercise)***

10. Remove the file `home_directory` by entering  
**`rm home_directory`**
11. Verify that the file was removed by entering **`ls -l`**.
12. Close the terminal window.

***(End of Exercise)***

### **Exercise 6-6 Use vi to Edit Files in the Linux System**

It will most likely take some time for you to get used to vi. It is very different from graphical text editors you might already be familiar with.

Its biggest advantage is certainly that it is available on practically any Unix or Linux you will come across. But if you keep at it, you will find out that beyond simple editing of text files it is extremely versatile and powerful.

The purpose of this exercise is to help you with your first steps with vi.

To use the command line editor vi, do the following:

1. Open a terminal window.
2. Start vim by entering **vi**.
3. Switch to the insert mode by typing **i**.
4. Type the following 2 paragraphs of text (press **Enter** at the end of each line):

**Administrator training for SLES 9 will be held in Training Room 4 of Building B on Tuesday of next week.**

**Make sure you bring your SLES 9 Administration Manual. There will be wireless Internet access available in the training room.**

5. Exit the insert mode by pressing **Esc**.
6. Move the cursor to the middle of the second line of the first paragraph.
7. Delete text to the right of the cursor by typing **D** (uppercase d).
8. Undo the deletion by typing **u**.
9. Delete the character directly under the cursor by pressing **Delete**.

### **Exercise 6-6 Use vi to Edit Files in the Linux System**

It will most likely take some time for you to get used to vi. It is very different from graphical text editors you might already be familiar with.

Its biggest advantage is certainly that it is available on practically any Unix or Linux you will come across. But if you keep at it, you will find out that beyond simple editing of text files it is extremely versatile and powerful.

The purpose of this exercise is to help you with your first steps with vi.

To use the command line editor vi, do the following:

1. Open a terminal window.
2. Start vim by entering **vi**.
3. Switch to the insert mode by typing **i**.
4. Type the following 2 paragraphs of text (press **Enter** at the end of each line):

**Administrator training for SLES 9 will be held in Training Room 4 of Building B on Tuesday of next week.**

**Make sure you bring your SLES 9 Administration Manual. There will be wireless Internet access available in the training room.**

5. Exit the insert mode by pressing **Esc**.
6. Move the cursor to the middle of the second line of the first paragraph.
7. Delete text to the right of the cursor by typing **D** (uppercase d).
8. Undo the deletion by typing **u**.
9. Delete the character directly under the cursor by pressing **Delete**.

10. Copy the current line to the internal buffer by typing **y** twice.
11. Move the cursor to the beginning the first line of the second paragraph.
12. Insert the content of the internal buffer after the current line by typing **p**.
13. Save the file with filename `vi_test` by entering  
**`:w vi_test`**
14. Exit vi by entering **`:q`**.
15. Close the terminal window.

***(End of Exercise)***

10. Copy the current line to the internal buffer by typing **y** twice.
11. Move the cursor to the beginning the first line of the second paragraph.
12. Insert the content of the internal buffer after the current line by typing **p**.
13. Save the file with filename vi\_test by entering  
**:w vi\_test**
14. Exit vi by entering **:q**.
15. Close the terminal window.

***(End of Exercise)***

**Exercise 6-7    Optional: Work through the vim Tutorial**

The last exercise was only a brief introduction to vi. There is a tutorial that comes with vim and takes about 30 minutes to work through. It provides increased familiarity with the possibilities vim offers.

To work through the tutorial, do the following

1. As the user `geeko`, open a terminal window.
2. Start vim and the tutorial by entering  
**vimtutor**  
vim starts displaying the tutorial.
3. Work through the lessons of the tutorial as explained in the text.
4. When finished, close the terminal window.

***(End of Exercise)***

**Exercise 6-7    Optional: Work through the vim Tutorial**

The last exercise was only a brief introduction to vi. There is a tutorial that comes with vim and takes about 30 minutes to work through. It provides increased familiarity with the possibilities vim offers.

To work through the tutorial, do the following

1. As the user `geeko`, open a terminal window.
2. Start vim and the tutorial by entering  
**vimtutor**  
vim starts displaying the tutorial.
3. Work through the lessons of the tutorial as explained in the text.
4. When finished, close the terminal window.

***(End of Exercise)***

**Exercise 6-8    Optional: Using sed**

**sed** is a very powerful tool that can be used in various ways when working with text files. A common one is searching and replacing text strings.

When you first use sed, its syntax will probably seem rather cryptic. Don't get discouraged.

The purpose of this exercise is to give you some idea of what can be done with sed.

1. As the user `geeko`, open a terminal window.
2. Display the first 15 lines of the file `/etc/init.d/skeleton` by entering  
**`sed 1,15p /etc/init.d/skeleton`**.
3. You will notice that lines 1 to 15 have indeed been displayed, but all other lines of the file as well (lines 1 to 15 therefore appear twice in the output).

Improve your command line by suppressing those lines that do not match the pattern:

**`sed -n 1,15p /etc/init.d/skeleton`**.

4. Now print all lines on the screen except lines 1 to 15  
**`sed 1,15d /etc/init.d/skeleton`**.
5. A simple search and replace consists of inserting additional spaces at the beginning of each line. This can be achieved as follows:  
**`sed -n -e 's/^/ /p' /etc/init.d/skeleton`**.

**Exercise 6-8    Optional: Using sed**

**sed** is a very powerful tool that can be used in various ways when working with text files. A common one is searching and replacing text strings.

When you first use sed, its syntax will probably seem rather cryptic. Don't get discouraged.

The purpose of this exercise is to give you some idea of what can be done with sed.

1. As the user `geeko`, open a terminal window.
2. Display the first 15 lines of the file `/etc/init.d/skeleton` by entering  
**`sed 1,15p /etc/init.d/skeleton`**.
3. You will notice that lines 1 to 15 have indeed been displayed, but all other lines of the file as well (lines 1 to 15 therefore appear twice in the output).

Improve your command line by suppressing those lines that do not match the pattern:

**`sed -n 1,15p /etc/init.d/skeleton`**.

4. Now print all lines on the screen except lines 1 to 15  
**`sed 1,15d /etc/init.d/skeleton`**.
5. A simple search and replace consists of inserting additional spaces at the beginning of each line. This can be achieved as follows:  
**`sed -n -e 's/^/ /p' /etc/init.d/skeleton`**.

6. The file `/etc/init.d/skeleton` contains the string “FOO” in places where the name of a program could be inserted. Let’s assume the program you want to start using this start script is called “bar” and you want to replace FOO by bar.

**`sed -e 's/FOO/bar/g' /etc/init.d/skeleton > /etc/init.d/bar`.**

(Writing to `/etc/init.d/bar` works only as root, as a normal user may not create files in `/etc/init.d/`. You can redirect standard out to a file in your home directory instead.)

7. If you want to experiment more with sed, try to:

```
sed -n -e '/#/p' /etc/init.d/skeleton
```

```
sed -e '/#/d' -e '/^$/d' /etc/init.d/skeleton
```

```
sed -n -e '/ start)/,/ stop)/p' /etc/init.d/skeleton | sed -e '$d'
```

```
sed -e '$r /etc/motd' /etc/init.d/skeleton
```

- Show only the comments (lines with a #)
- Show lines that are not comments and are not empty.
- Show the lines that belong to the “start” section of the case statement.
- Find out how to insert another textfile after a certain line.

***(End of Exercise)***

6. The file `/etc/init.d/skeleton` contains the string “FOO” in places where the name of a program could be inserted. Let’s assume the program you want to start using this start script is called “bar” and you want to replace FOO by bar.

**`sed -e 's/FOO/bar/g' /etc/init.d/skeleton > /etc/init.d/bar`.**

(Writing to `/etc/init.d/bar` works only as root, as a normal user may not create files in `/etc/init.d/`. You can redirect standard out to a file in your home directory instead.)

7. If you want to experiment more with sed, try to:

```
sed -n -e '/#/p' /etc/init.d/skeleton
```

```
sed -e '/#/d' -e '/^$/d'
/etc/init.d/skeleton
```

```
sed -n -e '/ start)/,/ stop)/p'
/etc/init.d/skeleton | sed -e '$d'
```

```
sed -e '$r /etc/motd'
/etc/init.d/skeleton
```

- Show only the comments (lines with a #)
- Show lines that are not comments and are not empty.
- Show the lines that belong to the “start” section of the case statement.
- Find out how to insert another textfile after a certain line.

***(End of Exercise)***