Tarea 4.4: Configuración avanzada de shell

Descripción de la tarea

En este laboratorio, los estudiantes practicarán la personalización de la shell y la programación básica de shell realizando las siguientes tareas:

- Ver y modificar variables locales y de entorno para personalizar la shell.
- Crear, usar y evitar alias para ejecutar comandos dentro de la shell.
- Definir funciones que ejecuten múltiples comandos.
- Usar listas de comandos para ejecutarlos de forma lógica.
- Personalizar los archivos de inicialización de bash.
- Crear varios scripts bash sencillos.

Pasos de la tarea

1. Recuerda que las variables de la shell contienen información útil para la propia shell y otros programas ejecutables. Muestra todas las variables (tanto locales como de entorno) ejecutando el comando *set*.
2. En algunos casos, querrás distinguir entre variables de entorno y variables locales. Para mostrar solo las variables de entorno, ejecuta el comando *env*:
3. Para establecer la variable local PS1, ejecuta el siguiente comando `PS1='\# \$ '`. Esta variable local cambiará el prompt para mostrar el número de historial del comando actual y mostrará el carácter del prompt como *$*. El número que se mostrará refleja un conteo continuo de los comandos en el historial. Este número aumentará con cada comando que ejecutes. El número que aparezca en tu propia ventana puede variar, dependiendo de cuántos comandos hayas ejecutado desde que iniciaste la shell.
4. Antes de cambiar la variable de entorno COLUMNS, observa cómo esta variable afecta el número de columnas que muestran comandos como *ls /bin*.
5. Cambia la variable COLUMNS a un valor de 60 y usa el comando export para convertirla en una variable de entorno:
6. Muestra la variable de entorno PATH, que determina en qué directorios se buscarán los comandos ejecutados desde la línea de comandos.
7. Para probar la variable PATH, primero crea un nuevo directorio en tu directorio personal llamado scripts ejecutando el siguiente comando.
8. Crea un script que muestre la fecha actual.
9. Ejecuta el script creado.
10. Si quieres poder ejecutar el script directamente (sin usar el comando bash), los permisos del usuario deben incluir permisos de lectura y ejecución. De lo contrario, se mostrará un mensaje indicando que el permiso ha sido denegado. Por defecto, los archivos que se crean nunca tienen permiso de ejecución automáticamente. Intenta ejecutar el script, sabiendo que no tiene permiso de ejecución.
11. Ver el listado detallado del archivo con el comando ls te permitirá ver los permisos del archivo. Usa *ls -l* para ver los detalles del archivo de script. Ejecuta el script en el contorno actual de tu shell (ten en cuenta que esto si fuera el caso podría afectar al entorno actual). A continuación, dale permisos de ejecución con el comando *chmod* y ejecuta el script.
12. Aunque las rutas relativas y absolutas funcionan para ejecutar el script, usar solo el nombre del script fallará porque el directorio scripts no está incluido como valor en la variable PATH. Intenta ejecutar el script usando solo su nombre; fallará porque el directorio actual no se busca para comandos.

13. Muestra nuevamente la variable PATH para confirmar que el directorio scripts no está incluido en ella. Modifica la variable PATH para que contenga el directorio scripts y vuelve a mostrar la variable PATH para verificar el cambio.
14. Los cambios realizados en las variables dentro de la shell no persisten después de que el usuario cierra sesión en el sistema. Escribe *exit* para cerrar la sesión, vuelve a conectarte, muestra la variable PATH y trata de ejecutar el sctipt.
15. Modifica el archivo `.profile` puede usarse para personalizar variables como PATH. La variable PATH se establecerá con el valor de la variable PATH existente y el directorio $HOME/bin. Observa el carácter dos puntos (😃 que separa las rutas.
16. Muestra la variable PATH para verificar que ahora incluye el directorio scripts después de haber ejecutado el comando source sobre el archivo `~/.profile`.
17. Los archivos que están en el directorio /etc/skel se copian automáticamente al directorio personal de un nuevo usuario cuando se crea su cuenta. Ejecuta el siguiente comando para ver estos archivos *ls -a /etc/skel*:
18. Para añadir cuentas de usuario o copiar archivos en el directorio `/etc/skel` se requieren privilegios de superusuario. Copia el archivo `.profile` personalizado desde el directorio personal del usuario superadministrador. Al colocar este nuevo .bash_profile en el directorio /etc/skel, cualquier nueva cuenta BASH tendrá automáticamente el contenido de este archivo en el directorio personal del nuevo usuario. Esto permite al administrador crear personalizaciones que afectarán a todas las nuevas cuentas de usuario.
19. Crea el usuario *prueba* y comprueba su PATH. En este punto ya tenemos añadido al PATH del usuario la ruta donde deberían figurar los scipts del nuevo usuario.
20. Antes de añadir alias y funciones al archivo /etc/skel/.bashrc, se recomienda que los pruebes en una shell interactiva para asegurarte de que funcionan como esperas. Primero, visualiza los alias que están definidos actualmente en la shell ejecutando el comando alias.
21. Para añadir un alias a tu shell, utiliza el comando alias. Ejecuta los siguientes comandos para definir un nuevo alias llamado lt, que mostrará los archivos en orden inverso de tiempo, y luego muestra los alias actuales de la shell.
22. Para eliminar la definición del alias de tu sesión actual de shell, utiliza el comando unalias junto con el nombre del alias. Ejecuta los siguientes comandos para eliminar el alias lt y verifica que ha sido eliminado.
23. Aunque los alias suelen ser nombres más cortos para comandos largos, las funciones te permiten ejecutar múltiples comandos y trabajar con argumentos que se pasan a estos comandos. Para ver las funciones que están definidas en tu shell Bash actual, ejecuta el comando *declare -f*.
24. Una función puede definirse escribiendo primero el nombre que se le va a asignar, seguido de un par de paréntesis, y luego los comandos y opciones de la función entre un par de llaves. Por ejemplo, la siguiente función listará los archivos en orden inverso por tamaño:

```
lS() { ls -lrS $@; }
```

La función utiliza el comando *ls* con las opciones *-lrS* para listar los archivos en orden inverso por tamaño y la variable *$@* para referirse a cualquier argumento que se le haya pasado a la función. Ejecuta los siguientes comandos para crear la función y verificar que existe (nota: el nombre de la función es lS, una “l” minúscula y una “S” mayúscula). 25. El comando *unset* se puede usar para eliminar la definición de una función dentro de una shell. Ejecuta los siguientes comandos para eliminar la función y verificar que ya no existe. 26. Después de

haber probado el alias *lt* y la función *lS*, ya estás listo para ponerlos a disposición de los usuarios añadiéndolos al archivo .bashrc en el directorio `/etc/skel`.