General Purpose Utitilites

The Calendar

`cal`

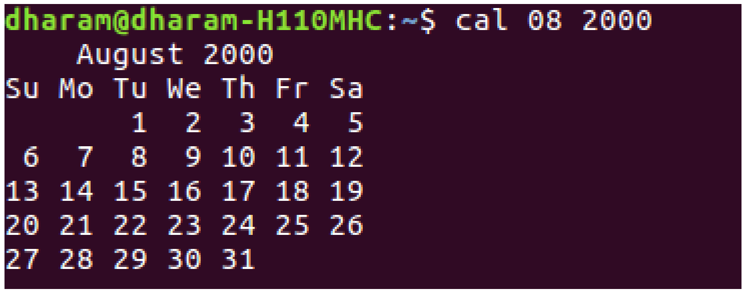
- The cal command to see the calendar of any specific month or a Complete year

Example

$cal

$cal 11 2021

$cal 2019 | more



```
dharam@dharam-H110MHC:~$ cal 08 2000
    August 2000
Su Mo Tu We Th Fr Sa
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

Displaying the system Date

date

date command displays the current date and time

```
$ date
```

The command can also be used with format specifier as arguments

```
$ date +%m
10
$ date +%d-%m-%y
09-10-21

$ date +"%h  %m"
Oct 10
```

Other Format Specifiers

d	The day of the month
y	The last two digits of the year
H,M and S	The hour, minute and second
D	The date in the format mm/dd/yy
T	The time in the format hh:mm:ss

Displaying a Message

echo

- To display a message

echo Hello World

- To evaluate shell variables

Echo $SHELL

echo with escape sequence

- An escape sequence is generally a two character-string beginning with a \(backslash)

- Escape sequence is placed at the end of a string used as an argument to echo

  “\c” –Prompt and cursor in same line

  echo “Enter the filename \c”

Escape Sequence Used by echo and printf

Escape Sequence	Significance
\a	Bell
\b	Backspace
\c	No newline (Cursor in same line)
\f	Form feed
\n	Newline
\r	Carriage return
\t	Tab
\v	Vertical tab
\\	Backslash
\0n	ASCII character represented by the octal value n, where n cant excedd 0377 (decimal value 255)

printf

- The ***printf*** command can be used as an alternative to ***echo*** command

```
$printf “Hello World”
Hello World$
```

- To have a newline character, one can use the escape sequence \n

```
$printf “Hello World\n”
Hello World
```

printf

- Just like printf() function of C language, in UNIX also, printf command can use format specifiers like %s, %d, %f, %o, %x etc.

- The values of variables can be displayed along with printf command

```
$printf "Current shell is %s \n" $SHELL
Current shell is /bin/bash
```

```
X=6
printf $X
```

The Calculator

bc

Basic operations:

```
$bc
3+5
8
5*6
30
6-10
-4
[ctrl+d]
```

To perform more than one operation in a single line:

```
$bc
2^4; 3+6
16
9
[ctrl+d]
```

Changing your password

`passwd`

```
[chetana@server4 ~]$ passwd
Changing password for user chetana.
Changing password for chetana
(current) UNIX password:*******
New UNIX password:**********
Retype new UNIX password:**********
passwd: all authentication tokens updated successfully.
[chetana@server4 ~]$
```

Who are the users?

who

- **To know the users of the system**
- Normally a UNIX system is used by multiple users at a time
- One user may needs to know
- the list of other users who are using the system currently. The *who* command is used for this purpose
- This command displays name of the users (login ID used to log in), name of the terminal and date and time of login

```
$who
root        :0              Sept 04 10:12
chetana     tty01           Sept 04 11:11
raghu       tty02           Sept 04 12:35
ram         tty03           Sept 04 14:08
```

Knowing Your Machine's Characteristics

uname

The command ***uname*** is a short-form for UNIX name, which displays the details like name and version of the machine and OS currently running.

-
  ```
  $uname
  Linux
  ```
  The command without any options displays the name of underlying OS.

-
  ```
  $uname -a
  Linux server4 2.6.18-128.el5xen #1 SMP Wed Dec 17 12:01:40
  EST 2008 x86_64 x86_x
  ```
  This has displayed details like kernel name, node name, kernel release, kernel version etc.

-
  ```
  $uname -n
  server4
  ```
  When your system is connected to network, it prints the name of the machine in

Knowing your Terminal

`tty`

The command *tty* (teletype) is used to know name of the terminal.

```
$tty
/dev/tty01
```

The above statement indicates that ***`tty01`*** is the name of the terminal and it is within the directory ***`dev`***. The ***`dev`*** is under ***`root`*** directory.

Displaying and Setting Terminal characteristics

`stty`

- This command is used to set terminal characteristics. The terminal is a device with which user communicates.
- Each terminal is configured differently depending on the user’s choice.

- For example, a user can decide
  - what should be the abort key (like Ctrl+c or Delete key etc),
  - whether a character has to be deleted or not when backspace key is used
  - what should be the end-of-file character when *cat* command is used (like Ctrl+d or Ctrl+a etc)

- The ***stty*** command helps the user in setting all such characteristics and also to revoke existing characteristics.

```
$stty
speed 9600 baud; line = 0;
-brkint -imaxbel
```

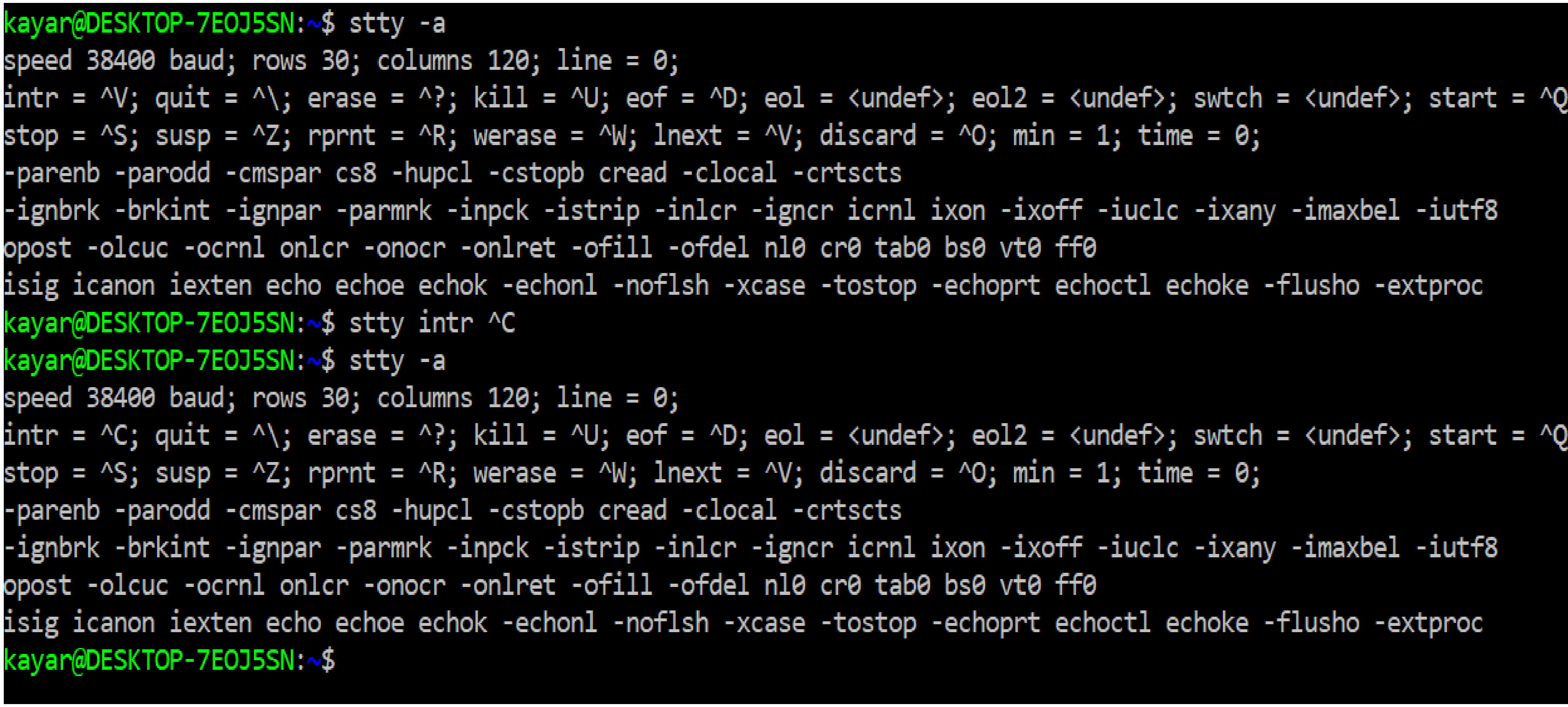
Initially it displays ***baud rate*** of the terminal as 9600. The number of characters that a terminal can transmit per second is known as baud rate. The ***line*** indicates the line discipline of Unix terminal. It does the input processing in the kernel. The ***brkint*** indicates that whether or not (with – or without –) an interrupt signal has to be generated when there is a break in the script. The ***imaxbel*** indicates to beep and do not flush a full input buffer on a character.

The ***–a*** (all) option with this command will display the current settings

```
$stty –a
```

STTY Setting - Example

stty intr ^C

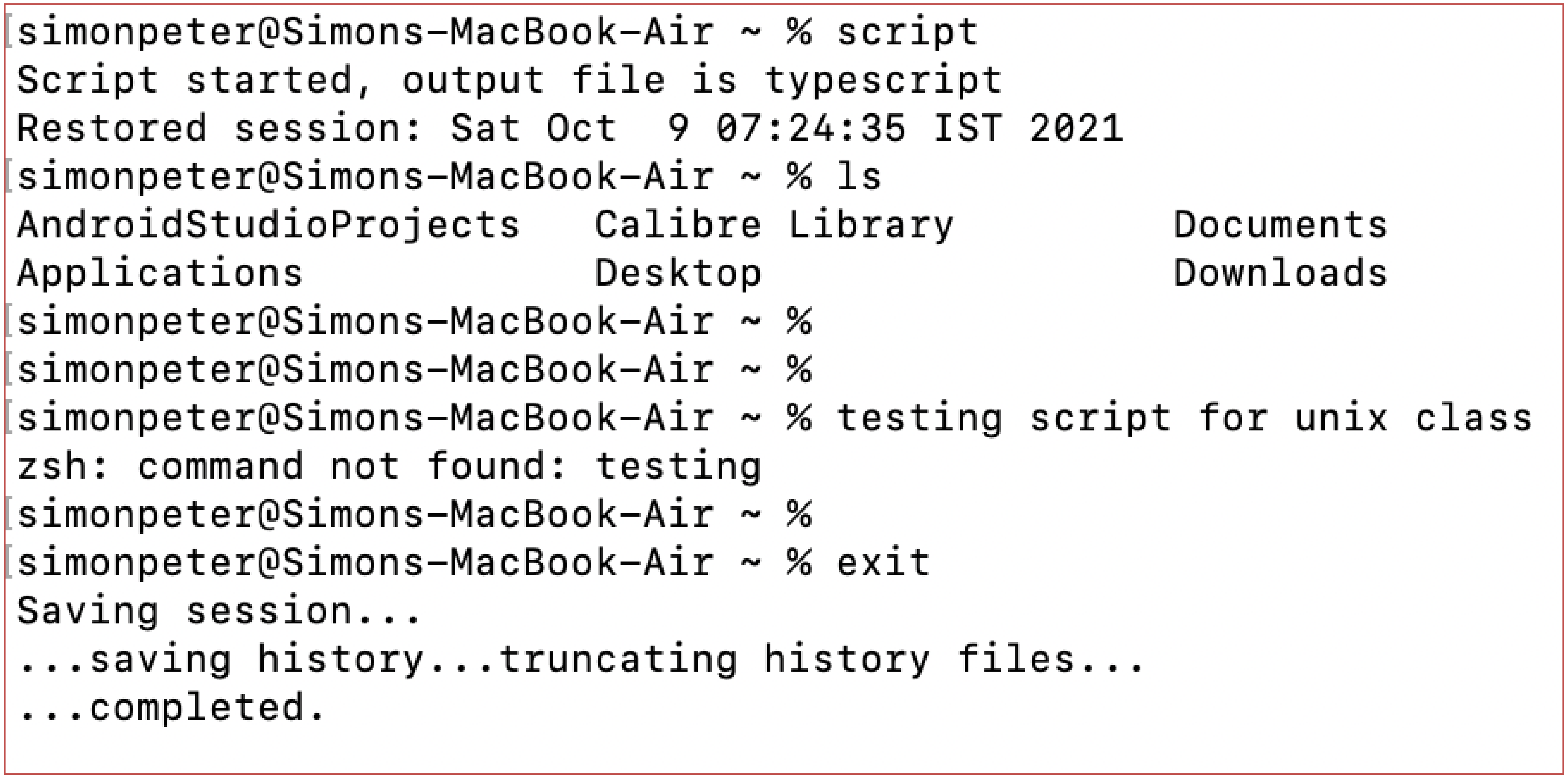


```
kayar@DESKTOP-7EOJ5SN:~$ stty -a
speed 38400 baud; rows 30; columns 120; line = 0;
intr = ^V; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; eol2 = <undef>; swtch = <undef>; start = ^Q
stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W; lnext = ^V; discard = ^O; min = 1; time = 0;
-parenb -parodd -cmspar cs8 -hupcl -cstopb cread -clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff -iuclc -ixany -imaxbel -iutf8
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt echoctl echoke -flusho -extproc
kayar@DESKTOP-7EOJ5SN:~$ stty intr ^C
kayar@DESKTOP-7EOJ5SN:~$ stty -a
speed 38400 baud; rows 30; columns 120; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; eol2 = <undef>; swtch = <undef>; start = ^Q
stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W; lnext = ^V; discard = ^O; min = 1; time = 0;
-parenb -parodd -cmspar cs8 -hupcl -cstopb cread -clocal -crtscts
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff -iuclc -ixany -imaxbel -iutf8
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt echoctl echoke -flusho -extproc
kayar@DESKTOP-7EOJ5SN:~$
```

Recording your session

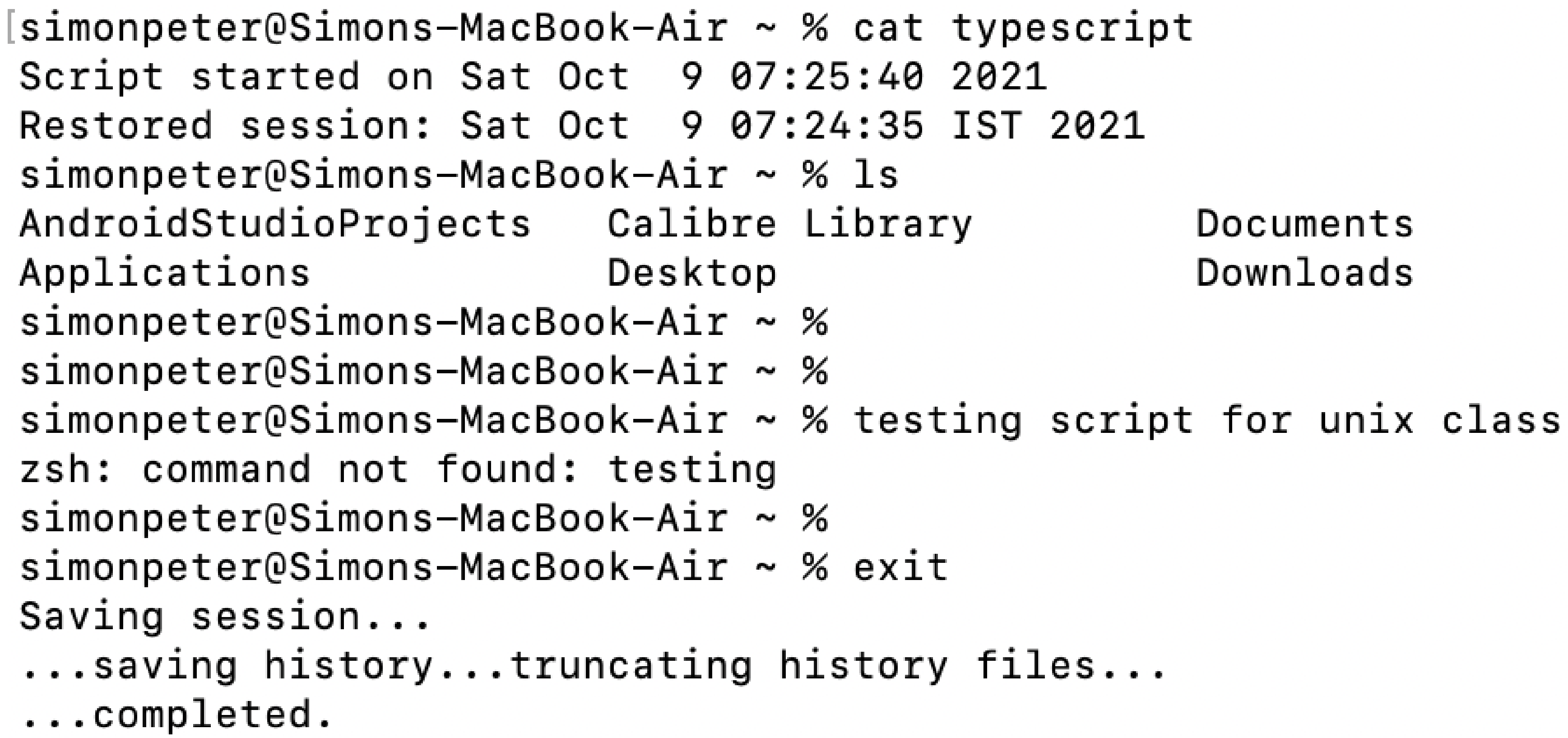
`script`

- **script** command is used to record all the terminal activities
- After executing the *script* command it starts recording everything printed on the screen including the inputs and outputs until exit
- *script* will automatically create a file namely *typescript* in the home directory to save the recorded information.



```
simonpeter@Simons-MacBook-Air ~ % script
Script started, output file is typescript
Restored session: Sat Oct  9 07:24:35 IST 2021
simonpeter@Simons-MacBook-Air ~ % ls
AndroidStudioProjects   Calibre Library         Documents
Applications            Desktop                 Downloads
simonpeter@Simons-MacBook-Air ~ %
simonpeter@Simons-MacBook-Air ~ %
simonpeter@Simons-MacBook-Air ~ % testing script for unix class
zsh: command not found: testing
simonpeter@Simons-MacBook-Air ~ %
simonpeter@Simons-MacBook-Air ~ % exit
Saving session...
...saving history...truncating history files...
...completed.
```

Content typescript file



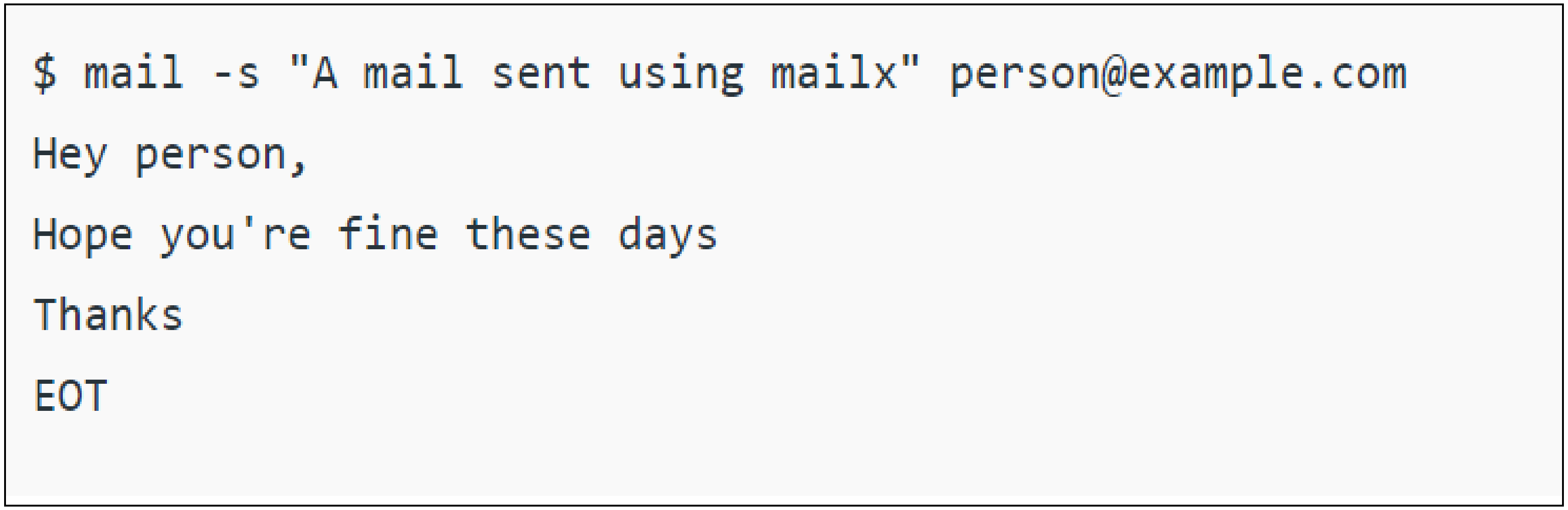
```
[simonpeter@Simons-MacBook-Air ~ % cat typescript
Script started on Sat Oct  9 07:25:40 2021
Restored session: Sat Oct  9 07:24:35 IST 2021
simonpeter@Simons-MacBook-Air ~ % ls
AndroidStudioProjects   Calibre Library         Documents
Applications            Desktop                 Downloads
simonpeter@Simons-MacBook-Air ~ %
simonpeter@Simons-MacBook-Air ~ %
simonpeter@Simons-MacBook-Air ~ % testing script for unix class
zsh: command not found: testing
simonpeter@Simons-MacBook-Air ~ %
simonpeter@Simons-MacBook-Air ~ % exit
Saving session...
...saving history...truncating history files...
...completed.
```

The Universal Mailer

mailx

- Linux has an inbuilt Mail User Agent program called mailx.
- it is a console application that is used for sending and receiving emails
- The mailx utility is an enhanced version of the mail command
- The mailx command is available from a variety of different packages:
  - bsd-mailx
  - heirloom-mailx
  - mailutils

Sending an Email



```
$ mail -s "A mail sent using mailx" person@example.com
Hey person,
Hope you're fine these days
Thanks
EOT
```

Writing the message directly in the command line:

- To send a simple email, use the “-s” flag to set the subject in quotes which is followed by the email of the receiver.
- After this, mailx waits for the content of the email.
- After the content is written, press Ctrl+D & EOT will be displayed by mailx.