

Introduction

The shell script is a very powerful thing when used correctly. Before you start learning how to shell script, you must be taught what shell scripting is. Please note that you really should already be comfortable with some terminal commands.

What is the shell script?

The shell script in short is terminal in a way. When you open terminal, terminal is just displaying text and letting you enter text into programs other than terminal. This is why when you SSH to another computer, you do not experience your terminal setup, but theirs.

The first program that terminal runs is bash. This is a UNIX executable located in `/bin/`. This is the default shell on the Mac and is what displays the prompt, reads the `bashrc` file, and exits when you quit. This program though is not just made for terminal commands, it can also comprehend a simple programming language called shell.

There is another shell on your computer, actually, there are several, but the most common one is `sh`. This, too, is located in `/bin/`, and too processes shell. We will actually be using `sh` in this tutorial, but please note that it will work with `bash` all the same.

The while loop

The while loop looks like this

```
while (condition) do
# insert code here
done
```

In this case everything inside of the `do` and `done` will be run repeatedly as long as the condition in the parenthesis is true. This means that in there, instead of typing 'condition', I could type `'3 = 3'`. Then as long as `3 = 3`, which is always, this the code in the loop will be run repeatedly. So therefore if I typed the following code:

```
while (3 = 3) do
    say Big surprise
done
```

This will make the computer repeatedly talk and say Big surprise. So that is while loops in a nutshell.

Using if statements

If statements are simply for checking things. Here is your basic if statement.

```
if [ condition ]; then
# Your code here...
fi
```

This means that everything within the 'then' and the 'fi' will run if the condition is true. This means that this would work.

```
if [ 3 = 3 ]; then
    say what do you know, three does equal three after all
fi
```

That simply would result in your computer running that say command. But it would not run the say command if you did this.

```
if [ 3 = 4 ]; then
    say what do you know, three does equal four after all
fi
```

So as you could see there, I said '3 = 4' which means that the condition is false, meaning that the text inside of it would not run. This is cool but maybe we could spice it up a little.

```
if [ 3=4 ]; then
    say This is messed up, three does not equal four
else
    say I knew that sh worked the way that it should
fi
```

This little bit of code would and up executing the command say and saying 'I knew ...'. So therefore else is nice too. We could even make it a little better with this.

Turn the page ...

```
if [ 3 = 4 ]; then
    say This is messed up, three does not equal four
elif [ 4 = 5 ]; then
    say This is too messed up, 4 does not equal 5
else
    say I knew that sh worked the way that it should
fi
```

So *elif* is another if basically, and you can add another one.

```
if [ 3 = 4 ]; then
    say This is messed up, three does not equal four
elif [ 4 = 5 ]; then
    say This is too messed up, 4 does not equal 5
elif [ 5 = 6 ]; then
    say This is too messed up, 5 does not equal 6
```

```
else
    say I knew that sh worked the way that it should
fi
```

So that was if statements. You can also use if statements to check if a file or directory exists. Let me just show you an example of each and you will figure it out on your own.

```
if [ -f /file.txt ]; then
    echo "The file /file.txt is there"
else
    echo "Where is that file that you told me about?"
fi
```

```
###
### Now we are checking for directories (folders)
###
```

```
if [ -d /system/ ]; then
    echo "The system folder is there, as it should"
else
    echo "Never shut down your computer because it will never
turn on again."
fi
```

So there you go!

Making a shell script

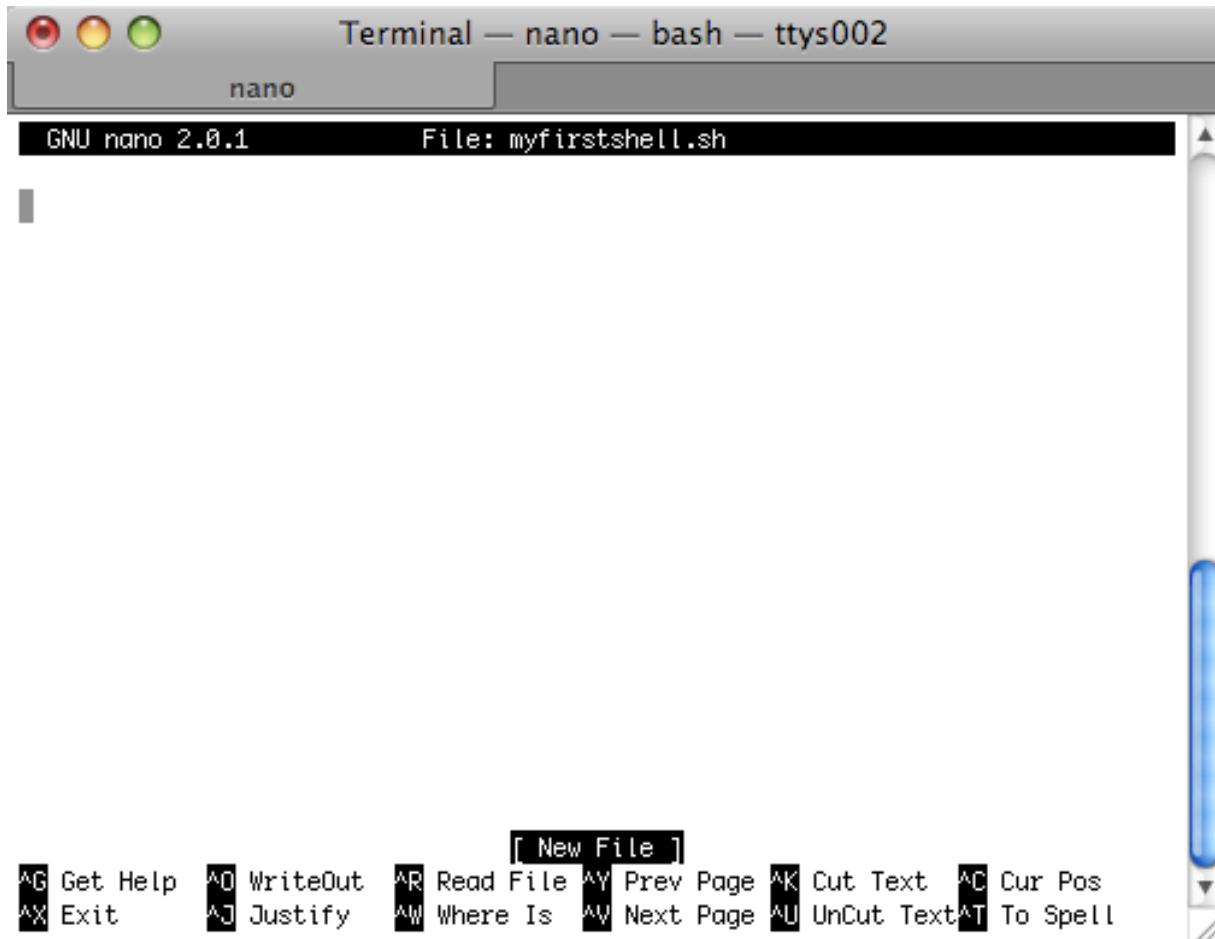
A shell script is just a plain text file on your computer. To make a text file, you can either use a terminal program called nano, a GUI application called Edit101 which you will need to download separately, or another text editor of your choice (TextEdit does not work for this).

In this tutorial I will be using terminal and nano. So, first of all, I will save this shell script to my Desktop, therefore, I will type the following terminal command before proceeding.

```
cd ~/Desktop
```

Now to edit the file that is my shell script, I will type 'nano myfirstshell.sh'

So now you will be presented with a screen looking something like this:




```
Terminal — nano — bash — ttys002
nano
GNU nano 2.0.1 File: myfirstshell.sh

[ New File ]
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^_ Next Page  ^U UnCut Text ^T To Spell
```

If your terminal looks something like this, then you are fine. Now with this screen, it is easy to type and edit text. I will remind you that terminal doesn't interact with the mouse so this program running in terminal will not work with your mouse, but the arrow keys will do. Now let's insert the following code to the first line shall we.

```
#!/bin/sh
```

This line that will be at the beginning of our file states that this program should be run in sh, Not bash. Now on the second line we can type any code. A terminal command per line is how this works, so on the second line if you type 'killall Dock' the command killall Dock will be executed. This is simple and very comprehensible. So now that we have typed all the source code for our shell script in, we press Control+X, then press 'Y', then hit enter.



```
Terminal — nano — bash — ttys002
nano
GNU nano 2.0.1 File: myfirstshell.sh Modified
#!/bin/sh
echo "If you can see this that I LOST THE GAME"
File Name to Write: myfirstshell.sh
^G Get Help      ^T To Files      ^M-M Mac Format   ^M-P Prepend
^C Cancel        ^M-D DOS Format   ^M-A Append      ^M-B Backup File
```

Now type the following command:

```
chmod +x myfirstshell.sh
```

Now on leopard you can double click the file and it will open with terminal, or you can type the path of the file in terminal (e.g ~/Desktop/myfirstshell.sh) to run the shell script. If you wish to send someone this executable shell script, zip it before sending it. You can do this by right clicking on the shell script, then selecting Compress "name".