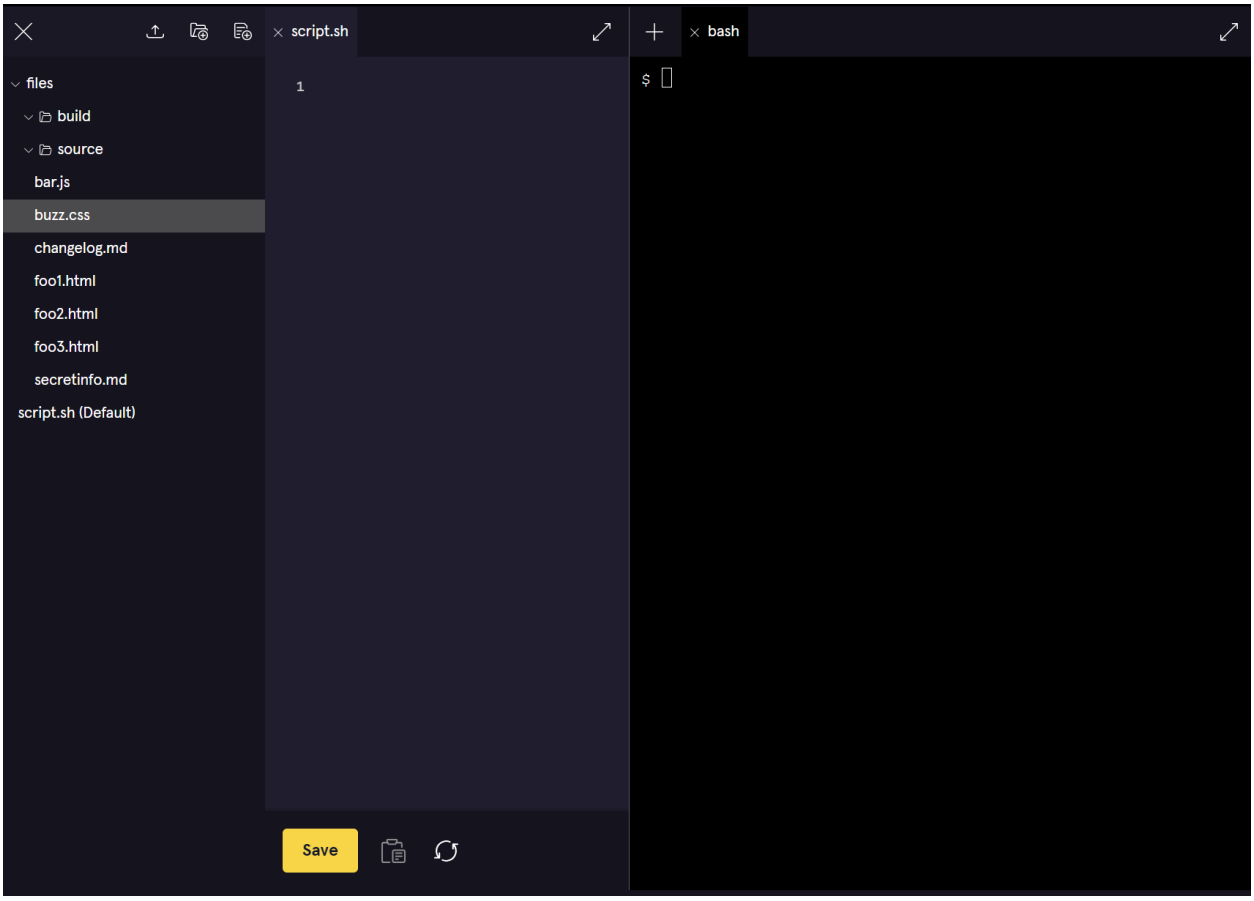
Build a Build Script

Note**: this project is provided by ***Codecademy.com

Below is the set up provided by Codecademy.com.

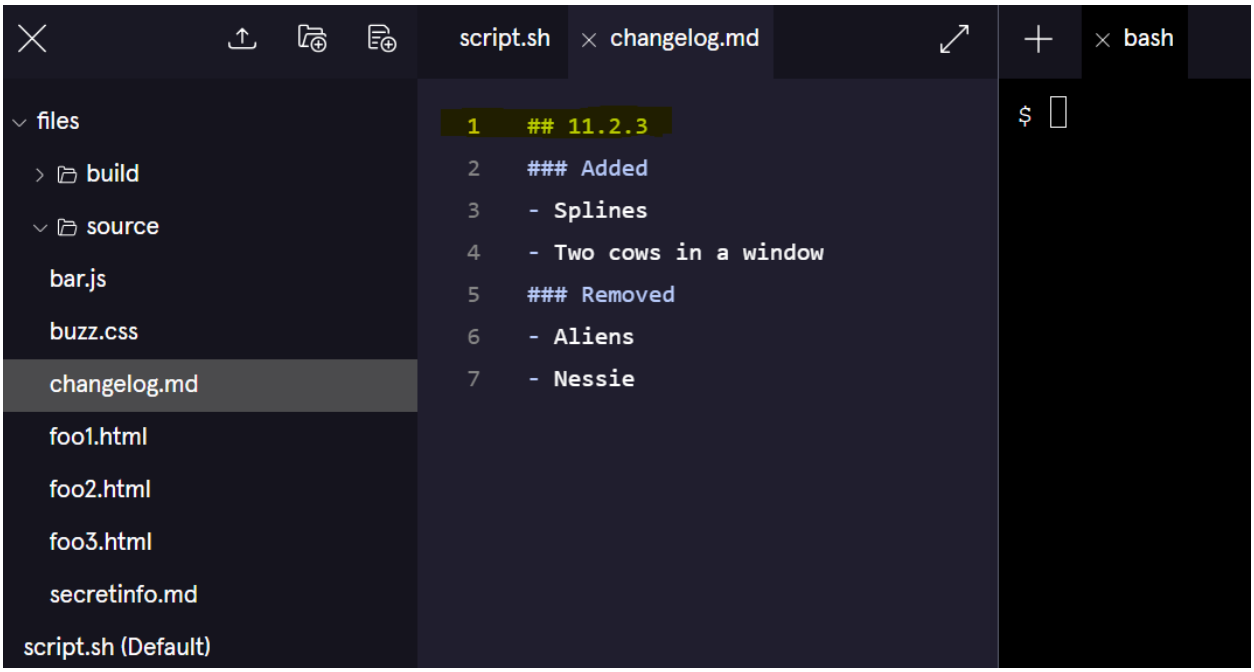


In this project, I created a release script to copy certain files from a **source** directory into a **build** directory.

1. I first looked at the **build** and **source** folders. The objective of the script is to copy files from **source** to **build**, with a couple of exceptions and modifications. I started on the script by adding a header to **script.sh**, identifying the type of script.

2. I created an echo to welcome the user and tested out the script in the terminal using **./script.sh**.

3. Since this is a new build, I verified that the **changelog.**md is the current release version. The first line (highlighted in yellow) of the file contains a version number with markdown formatting.



I read the first line of this file into a variable **firstline.**

- **Note:** Using the "**head**" argument returns the first 10 lines. The argument "**-n**" is used to specify the number of lines. I added **1** after the -n argument to specify only the first line. Shown in line 3:

4. I wanted just the version number without the markdown formatting. The command ***read*** can be used to split a string into an array using the ***-a*** argument.
    a. I split the string **firstline** into the array **splitfirstline.**
    b. The syntax for splitting a string **foo** into an array **bar** is:

```
read -a bar <<< $foo
```

        i. **Note:** Do not type this in.
    c. Here's is the result of what I typed in line 4 of the script:

```
1  #/bin/bash
2  echo "Welcome fellow traveler!!!"
3  firstline=$(head -n 1 source/changelog.md)
4  read -a splitfirstline <<< $firstline
```

5. I set the value of the version of the script. It is located in **index 1** of the array **splitfirstline**.

```
1  ## 11.2.3
```

    a. The syntax for accessing the value of **index** of an array **foo** is:

```
${foo[index]}
```

        i. **Note**: Do not type this in.
    b. I saved the version to a variable **version**. Shown in line 5:

```
1  #/bin/bash
2  echo "Welcome fellow traveler!!!"
3  firstline=$(head -n 1 source/changelog.md)
4  read -a splitfirstline <<< $firstline
5  version=${splitfirstline[1]}
```

c. I printed a statement to the terminal (e.g. “[statement]” $version) notifying the user of the version they are building. Shown in line 6:

```
1  #/bin/bash
2  echo "Welcome fellow traveler!!!"
3  firstline=$(head -n 1 source/changelog.md)
4  read -a splitfirstline <<< $firstline
5  version=${splitfirstline[1]}
6  echo "You are building version" $version
```

d. Result of outputting script.sh in the bash terminal:

```
1  #/bin/bash
2  echo "Welcome fellow traveler!!!"
3  firstline=$(head -n 1 source/changelog.md)
4  read -a splitfirstline <<< $firstline
5  version=${splitfirstline[1]}
6  echo "You are building version" $version
```

```
$ ./script.sh
Welcome fellow traveler!!!
You are building version 11.2.3
$
```

6. I printed a statement asking the user if they wanted to exit the script if they needed to make changes to the version. I assigned the response to the variable **versioncontinue**. This is what I typed in lines 7, 8 and the output in the bash terminal:

```
1  #/bin/bash
2  echo "Welcome fellow traveler!!!"
3  firstline=$(head -n 1 source/changelog.md)
4  read -a splitfirstline <<< $firstline
5  version=${splitfirstline[1]}
6  echo "You are building version" $version
7  echo 'Do you want to continue? (enter "1" for yes, "0" for no)'
8  read versioncontinue
```

```
$ ./script.sh
Welcome fellow traveler!!!
You are building version 11.2.3
Do you want to continue? (enter "1" for yes, "0" for no)
```

7. I added a conditional. If the user said “1” to the continue question, the rest of the script was executed, and the response was “OK”. If the user did not, the respond was “Please come back when you are ready”.
   a. **Note**: you can choose your own responses.

b. This is what I typed from lines 10 to 15 and the output in bash:

```
9
10  if [ $versioncontinue -eq 1 ]
11  then
12    echo "OK"
13  else
14    echo "Please come back when you are ready"
15  fi
```

```
$
$ ./script.sh
Welcome fellow traveler!!!
You are building version 11.2.3
Do you want to continue? (enter "1" for yes, "0" for no)
1
OK
$
```

8. I copied every file from **source** to **build**. Within the positive condition (Where I told the user "OK"), I started by iterating over all the files in the **source** directory and printed their names to the terminal.
   a. This is what I typed from lines 13 through 16 and the output in the bash terminal.

```
7   echo 'Do you want to continue?
    (enter "1" for yes, "0" for no)'
8   read versioncontinue
9
10  if [ $versioncontinue -eq 1 ]
11  then
12    echo "OK"
13    for filename in source/*
14    do
15      echo $filename
16    done
17  else
18    echo "Please come back when
      you are ready"
19  fi
```

```
$ ./script.sh
Welcome fellow traveler!!!
You are building version 11.2.3
Do you want to continue? (enter "1" for yes, "0" for no)
1
OK
source/bar.js
source/buzz.css
source/changelog.md
source/foo1.html
source/foo2.html
source/foo3.html
source/secretinfo.md
$
```

9. I decided I don't want to copy the file name **secretinfo.md** to the build source. To do this, I created an if/else statement within the **do** loop that informs the user what was copied and what was not.
   a. **Note**: the **cp "[**pick file name**]" [directory name]/.** command will copy a file to a directory.
      i. In my case, I used the **cp $filename build/.** command in the script (on line 21 in the script
   b. This is what I type from lines 16 to 22 and the output in the bash terminal.

```
$firstline
version=${splitfirstline[1]}
echo "You are building version" $version
echo 'Do you want to continue? (enter "1" for yes, "0" for no)'
read versioncontinue

if [ $versioncontinue -eq 1 ]
then
  echo "OK"
  for filename in source/*
  do
    echo $filename
    if [ "$filename" == "source/secretinfo.md" ]
    then
      echo "Not copying" $filename
    else
      echo "Copying" $filename
      cp $filename build/.
    fi
  done
else
  echo "Please come back when you are ready"
fi
```

```
$ ./script.sh
Welcome fellow traveler!!!
You are building version 11.2.3
Do you want to continue? (enter "1" for yes, "0" for no)
1
OK
source/bar.js
Copying source/bar.js
source/buzz.css
Copying source/buzz.css
source/changelog.md
Copying source/changelog.md
source/foo1.html
Copying source/foo1.html
source/foo2.html
Copying source/foo2.html
source/foo3.html
Copying source/foo3.html
source/secretinfo.md
Not copying source/secretinfo.md
$
```

c. I used the **ls build/** command in the bash terminal to check if the files were copies.

```
      echo "Copying" $filename
      cp $filename build/.
    fi
  done
else
  echo "Please come back when you are ready"
fi
```

```
$ ls build/
bar.js    changelog.md  foo2.html
buzz.css  foo1.html     foo3.html
$
```

10. I then navigated to the build directory in the bash terminal to show the results.

a. To show the results in the bash terminal, I used the **ls** command to show the root directory. Then I navigated to the build directory using the **cd build** command. Finally, I use the **ls** command to show the files copied to the build directory. Finally, I navigated back to the root directory I have been working in using the **cd ..**/ command.

```
$ ls
build  script.sh  source
$ cd build
$ ls
bar.js    changelog.md  foo2.html
buzz.css  foo1.html     foo3.html
$ cd ../
$ ls
build  script.sh  source
$
```

11. This navigation can also be completed in the script. Which is shown in lines 28 through 31 (which includes a print command referencing on what's contained with version 11.2.3 of the build directory) and the script output in the bash terminal.

script.sh

```
4  read -a splitfirstline <<< $firstline
5  version=${splitfirstline[1]}
6  echo "You are building version" $version
7  echo 'Do you want to continue? (enter "1" for yes, "0" for no)'
8  read versioncontinue
9
10 if [ $versioncontinue -eq 1 ]
11 then
12   echo "OK"
13   for filename in source/*
14   do
15     echo $filename
16     if [ "$filename" == "source/secretinfo.md" ]
17     then
18       echo "Not copying" $filename
19     else
20       echo "Copying" $filename
21       cp $filename build/.
22     fi
23   done
24 else
25   echo "Please come back when you are ready"
26 fi
27
28 echo "Build version $version contains:"
29 cd build
30 ls
31 cd ../
```

bash

```
$ ./script.sh
Welcome fellow traveler!!!
You are building version 11.2.3
Do you want to continue? (enter "1" for yes, "0" for no)
1
OK
source/bar.js
Copying source/bar.js
source/buzz.css
Copying source/buzz.css
source/changelog.md
Copying source/changelog.md
source/foo1.html
Copying source/foo1.html
source/foo2.html
Copying source/foo2.html
source/foo3.html
Copying source/foo3.html
source/secretinfo.md
Not copying source/secretinfo.md
Build version 11.2.3 contains:
bar.js    changelog.md  foo2.html
buzz.css  foo1.html     foo3.html
$
```

END OF PROJECT