BuildTools

Topics Covered	Make

Make:

- An awful lot of the things we do with a computer are about format shifting.

e.g. When we compile code:

```
gcc mycode.c -o mycode.o
```

- When we zip files :

```
zip -r coursework.zip coursework
```

Rules for a Make File:

▼ What a process for making a final output file would require without using a make file:

- Say you have three files: main.c and hello.c, where hello.c contains a function that prints out hello and main.c calls that function.
- To compile the outputfile you have to:

```
$ gcc hello.c -c hello.o -Wno (or other flags)
$ gcc main.c -c main.o -Wno
$ gcc hello.o main.o -o outputfile -Wno
```

- Using a Make file:

```
FLAGS = -Wno # Or any other flags

all: output

output: main.o hello.o
                    gcc $(FLAGS) hello.o main.o -o outputfile
                    # You can even add commands to change perm
                    chmod u+x output

# To get the output file, main.o and hello.o are required and o
# make has them to run the command below,

# However to get main.o and hello.o:

main.o: main.c
             gcc $(FLAGS) main.c -c main.o

hello.o: hello.c
            gcc $(FLAGS) hello.c -c hello.o


# The clean command allows you to rm all files:
clean:
        echo "Removing everything but the source files"
        rm main.o hello.o final
```

```
$ make all
# Or if I just want the main.o / hello.o
$ make main.o / $ make hello.o
```