Makefiles and You

James Biddle

June 18, 2019

1 Part A

1. Create a file called 'makefile', 'Makefile' or 'GNUmakefile'
2. Make a target called 'target'
3. Make it echo some text when run, highlight the fact that all you're doing with the makefile is executing bash commands.
4. Note that to write a command it must be preceeded by a TAB
5. Now make a new target called 'program.x', to compile 'program.f90'. Do not include dependency yet
6. Edit 'program.f90' to write another line of text
7. Type make and show that it doesn't recompile because there is no dependency specified.
8. Add dependency to 'program.f90'
9. Look at what happens if we just type make
10. Add dummy 'all' target to run all targets
11. Make 'clean' target to remove generated files
12. Make a file called 'clean' and then type 'make clean'. Show that we get undesired behaviour
13. Introduce a '.PHONY' target to resolve this behaviour
14. We've written 'program' a lot, which introduces concerns for typos. We can resolve this with inbuilt variables:

```
$@ to represent the full target name of the current target
$? returns the dependencies that are newer than the current
    target
$* returns the text that corresponds to % in the target
$< returns the name of the first dependency
$^ returns the names of all the dependencies with space as
    the delimiter
```