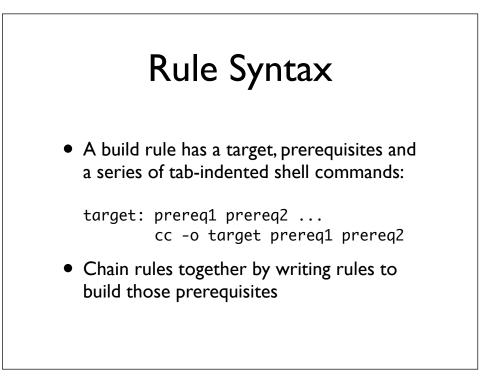
#### Make: How Does It Work?

- Automates program compilation
- Series of inter-dependent build rules to compile a program
- Builds only what it needs to
- Name it "Makefile" or use make -f



# Variables

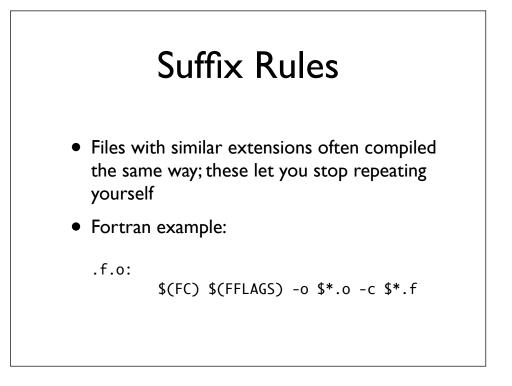
- Case-sensitive
- Setting: VAR = value
- Reference: \$(VAR)
- Environment variables automatically accessible in a makefile

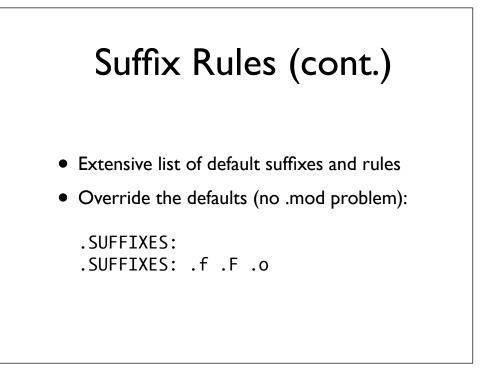
# Automatic Variables

- \$@ name of target
- \$< first prerequisite
- \$^ all prerequisites, space-separated
- \$\* the 'stem' of a file name in a suffix rule

# **Common Variables**

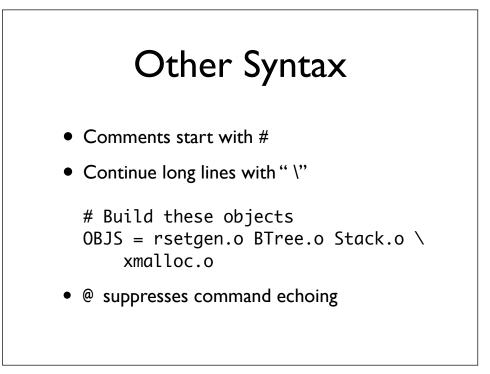
- FC, CC Fortran and C compilers
- FFLAGS, CFLAGS Fortran/C compile flags
- LFLAGS link flags; language-agnostic
- OBJS list of object files to link

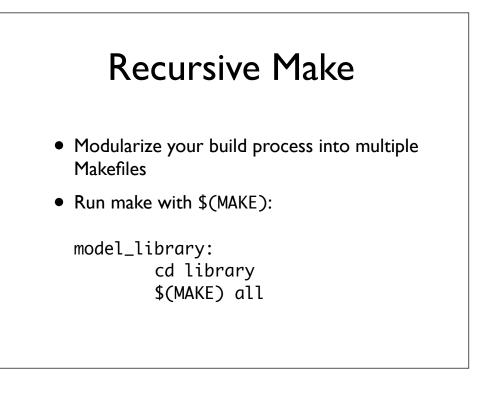




# **Common Rules**

- all builds all relevant targets, e.g. model + utilities; should be first rule in file
- clean: cleans up programs, object files, etc. to prepare for a rebuild
- .phony: all clean
- Now I can "make clean all"

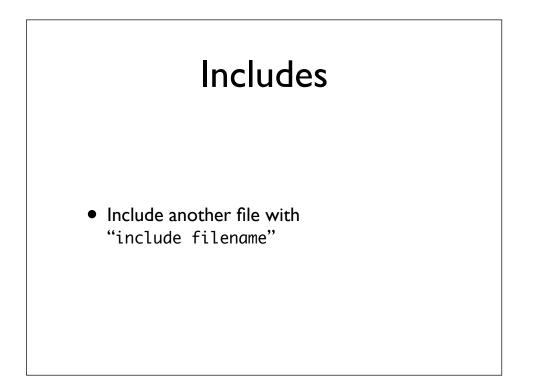




## Extra Dependencies

- Specify "non-obvious" dependencies like modules that live in other Fortran files
- Ex: makesfc.f90 uses the module in types.f90, add the rule:

makesfc.f90: types.o



# Conditions

```
HOST = $(shell hostname)
ifeq ($(HOST),bamboo)
    FC = gfortran
else ifeq ($(HOST),ilex)
    FC = pgf90
else
    FC = f77
endif
```

# Bringing It All Together

• Example Makefile