Chapel: Hands-on
Chapel Directory Structure (Partial)

chapel-1.2.0/
  chapel/
  README – quick-start instructions for building & using chpl
  – also, pointers to possible next steps
  README.files – complete directory structure description
  bin/ – location of the Chapel compiler
  doc/ – language spec, READMEs, quick reference
  examples/ – sample codes written in Chapel
  lib/ – location of the Chapel runtime libraries
  man/ – man page
Chapel Environment

- **Minimal:**
  
  $\textit{CHPL\_HOME}$: points to Chapel installation (chapel-*/chapel)
  $\textit{CHPL\_HOST\_PLATFORM}$: indicates host system

- **Others:**
  
  $\textit{CHPL\_HOST\_COMPILER}$: C compiler to use
  $\textit{CHPL\_COMM}$: Communication implementation to use
  $\textit{CHPL\_COMM\_SUBSTRATE}$: Underlying communication layer

This tutorial’s instructions will help you set these values

See $\textit{CHPL\_HOME/doc/README.chplenv}$ for advanced details
Hands-on Session

• **Goals:**
  - Get everyone up and running with Chapel
  - Try out base language and data parallel features

• **Chapel versions**
  - Use the classroom version
  - Or install your own

• **Things to do**
  - Read and execute sample programs (`$CHPL_HOME/examples`)
  - Work through Monte Carlo exercises
  - Write your own parallel program of interest

• **Further Instructions Here:**
Using Chapel on MareNostrum

- Environment Settings:
  - CHPL_HOST_PLATFORM: marenostrum
  - CHPL_COMM: gasnet
  - OBJECT_MODE: 64
  - CHPL_HOME: ~pws10020/chapel-1.2.0/chapel
  - add to PATH:
    - $CHPL_HOME/bin/$CHPL_HOST_PLATFORM
    - $CHPL_HOME/util
  - MPIRUN_CMD: ‘srun --kill-on-bad-exit %C’
  - MPIRUN_CMD_OK: true
- Output for a program ‘foo’ will appear in ‘foo_%jobid.out’
- Jobs will be run in the debug queue with a 10 minute time limit
- Errors often occur at program shutdown but can be ignored