AMath 483/583 — Lecture 3 — April 1, 2011

Today:

- Unix
- Mercurial examples

Monday:

- Compiled languages, Fortran 90

Bash shell

The shell is the program that prints a prompt in a terminal window, reads what you type in, executes commands.

bash is one commonly used shell.

Each time a new shell is started (e.g. by opening a new terminal), the commands in $HOME/.bashrc are executed.

HOME is an environment variable that is set automatically for you.

```
[~] aspen% printenv HOME
/Users/rjl

[~] aspen% echo $HOME
/Users/rjl
```


Setting your prompt

```
[~] aspen% PS1='$ '
$

$ PS1='[\W] \h% '
[~] aspen%
```

Note $\backslash W$ means last part of working directory, $\backslash h$ is machine name.

```
[~] aspen% cd hg/uwamath583s11
[uwamath583s11] aspen% pwd

/Users/rjl/hg/uwamath583s11
[uwamath583s11] aspen%
```


Environment variables

```
$ export CLASSHG=$HOME/hg/uwamath583s11

$ cd

$ pwd
/Users/rjl

$ cd $CLASSHG

$ pwd
/Users/rjl/hg/uwamath583s11
```

Exporting an environment variable makes it available to certain jobs you start from the shell.

Environment variables

To list all your environment variables:

```
$ printenv

TERM=xterm-color
SHELL=/bin/bash
CDPATH=.:~
USER=rjl
CLASSHG=/Users/rjl/hg/uwamath583s11
HOME=/Users/rjl
FC=gfortran
MATLABPATH=.:/Users/rjl/matlab
PYTHONPATH=.:/Users/rjl/python
etc...
```


hg demo — creating a new project

```
$ cd
$ mkdir newproject
$ cd newproject
/Users/rjl/newproject

$ hg init
$ ls
$ ls -a    # show hidden files
./  ../ .hg/

$ cat > file1.txt
First line
Second line
^D
$
```


hg demo

```
$ hg add file1.txt
adding file1.txt

$ hg status
A file1.txt

$ hg commit -m "First commit to this repository"
file1.txt
committed changeset 0:eb3222dd0b4f

$ hg status   # show only files that have changed
$

$ hg status -A   # show status of all files
C file1.txt
```


hg demo

```
$ vi file1.txt  # change the file

$ cat file1.txt  # print out the file
First line
New Second line
Added Third line

$ hg status
M file1.txt
```

hg demo

```
$ hg diff
diff -r eb3222dd0b4f file1.txt
--- a/file1.txt Thu Mar 31 20:48:55 2011 -0700
+++ b/file1.txt Thu Mar 31 20:53:00 2011 -0700
@@ -1,2 +1,3 @@
 First line
-Second line
+New Second line
+Added Third line
```


hg demo

```
$ hg commit file1.txt \
  -m "changed 2nd line, added 3rd"
file1.txt
committed changeset 1:ff67e66ed5b0

$ hg log file1.txt
changeset:   1:ff67e66ed5b0
tag:         tip
user:        Randy LeVeque <rjl@uw.edu>
date:        Thu Mar 31 20:54:57 2011 -0700
files:       file1.txt
description:
changed 2nd line, added 3rd

changeset:   0:eb3222dd0b4f
user:        Randy LeVeque <rjl@uw.edu>
date:        Thu Mar 31 20:48:55 2011 -0700
files:       file1.txt
description:
```


hg demo

```
$ pwd
/Users/rjl/newproject
$ cd

$ hg clone newproject newbranch
updating to branch default
resolving manifests
getting file1.txt
1 files updated, 0 files merged, 0 files removed, 0 file

$ cd newbranch
/Users/rjl/newbranch

$ ls
file1.txt
```


hg demo

```
$ pwd
/Users/rjl/newbranch
$ vi file1.txt
$ cat file1.txt
First line
New Second line
Changed Third line

$ hg commit -m "changed a file in the branch"
file1.txt
committed changeset 2:ddde02c30d03

$ cd ../newproject

$ cat file1.txt
First line
New Second line
Added Third line
```

hg demo

```
$ cd -
/Users/rjl/newbranch

$ hg push
pushing to /Users/rjl/newproject
searching for changes
1 changesets found
adding changesets
adding manifests
adding file changes
added 1 changesets with 1 changes to 1 files

$ cd -
/Users/rjl/newproject
$ hg update
resolving manifests
getting file1.txt
1 files updated, 0 files merged, 0 files removed, 0 file
```


hg demo

```
$ vi file1.txt
$ cat file1.txt
First line
Changed Second line and got rid of third

$ hg status
M file1.txt

$ hg revert file1.txt
saving current version of file1.txt as file1.txt.orig
reverting file1.txt

$ cat file1.txt
First line
New Second line
Changed Third line
```


hg demo

```
$ hg status
? file1.txt.orig

$ hg revert -r0 file1.txt
reverting file1.txt

$ cat file1.txt
First line
Second line

$ hg status
M file1.txt
? file1.txt.orig

$ hg commit -m "After reverting to revision 0"
file1.txt
committed changeset 3:5de4f8a71915
```


hg demo

To remove a directory and all subdirectories:

```
$ cd
$ rm -rf newbranch

$ ls newbranch
ls: newbranch: No such file or directory
```

`-rf` means recursively, force (without asking)

hg demo

```
$ cd newproject
/Users/rjl/newproject

$ du -h .  # show disk usage
4.0K    ./.hg/store/data
 20K    ./.hg/store
 52K    ./.hg
 60K    .
```

Could eliminate all history by...

```
$ rm -rf .hg
```

Warning: It's gone, unless you have a clone somewhere.

hg diff command

Now try:

```
$ cd $CLASSHG/codes/fortran
$ hg log demo1.f90 | more
```

Lists all the hg changesets in which file *demo1.f90* was changed.

Note changeset 10:54971910d50a has the log message "Fixed a bug: forgot to change n to m in declaration".
(Number 10: is clone-dependent!)

To see the changes from previous version:

```
$ hg diff -r9 -r10 demo1.f90 | more
```

To see if any changes were made since then:

```
$ hg diff -r10 tip demo1.f90 | more
```

tip means most recent committed version.

hg diff command

To see if any changes were made in working copy compared to tip:

```
$ hg diff demo1.f90 | more

$ hg diff | more    # shows diffs in all files
```

To check status of files in working version:

```
$ hg status          # for entire clone
$ hg status .        # for this directory
$ hg status -amr     # added, modified, removed
$ hg status *.f90    # only for .f90 files

$ hg help status     # for more options
```


Using xxdiff in hg

Modify the file .hg/hgrc, to add:

```
[extensions]
hgext.extdiff =
```

(Put in $HOME/.hgrc to apply in all directories.)

Then you can do:

```
$ hg extdiff -p xxdiff -r9 -r10 demo1.f90
```

Might want to add to .bashrc:

```
alias hgd = "hg extdiff -p xxdiff"
```

Then you can do:

```
$ hgd -r9 -r10 demo1.f90
```