CS 35L

LAB 8,

TA: Sucharitha Prabhakar

EMAIL ID: prabhakarsucharitha@gmail.com

Outline

- Diff
- Patch

Diff

- Compares file line by line

Diff - Example

- $ cat file1

Hi,
Hello,
How are you?
I am fine,
Thank you.

$ cat file2
Hello,
Hi,
How are you?
I am fine.

diff file1 file2

1d0

< Hi,

2a2

> Hi,

4,5c4

< I am fine,

< Thank you.

> I am fine.

Diff - Example

- The first argument to diff command is regarded as old file while the second argument becomes new file.
- Expressions like 1d0 2a2, 4,5c4 can be decoded with the syntax **[line number or range from old file][action][line number or range from new file]**. Where, 'action' can be append, delete or changed-so-replace.
- The mark < represents the line to be deleted while > represents the line to be added.

Diff - Example

- Use -i to ignore case differences
- Report That The Files Are Same Using -s Option
- Use -b To Ignore Spaces
- diff command can also be used to compare two directories

Patch

- Download the patch which is typically a fix to a software
- Apply the patch
- Compile
- Install

Patch

- Create a patch file using a diff
- Apply patch using patch command

Patch

- Create a patch from a Source Tree
- Apply patch file to a source code tree

Patch

- Take a backup before applying the patch using -b
  - $ patch -b < hello.patch
  - To decide backup file format: patch -b -V numbered < hello.patch
- Dry-run patch (To check if you will get any errors)
  - patch --dry-run < hello.patch
- Undo a patch
  - patch -R < hello.patch

Makefile

- Simple way to organize compilation

Makefile - Example

hellomake.c	hellofunc.c	hellomake.h

`#include <hellomake.h>`
 	
`int main() {`
`  // call a function in another file`
`  myPrintHelloMake();`
 	
`  return(0);`
`}`
	`#include <stdio.h>`
	`#include <hellomake.h>`
	
	`void myPrintHelloMake(void) {`
	
	`  printf("Hello makefiles!\n");`
	
	`  return;`
	`}`
		`/*`
		`example include file`
		`*/`
		
		`void myPrintHelloMake(void);`

gcc -o hellomake hellomake.c hellofunc.c -I.

Makefile - Example

```
hellomake: hellomake.c hellofunc.c
    gcc -o hellomake hellomake.c hellofunc.c -I.
```

make with no arguments executes the first rule in the file.

By adding the list of files on which the command depends on the first line after the :, make knows that the rule hellomake needs to be executed if any of those files change.

Makefile - Example

```
CC=gcc
CFLAGS=-I.

hellomake: hellomake.o hellofunc.o
	$(CC) -o hellomake hellomake.o hellofunc.o $(CFLAGS)
```

In particular, the macro CC is the C compiler to use, and CFLAGS is the list of flags to pass to the compilation command.

By putting the object files--hellomake.o and hellofunc.o--in the dependency list and in the rule, make knows it must first compile the .c versions individually, and then build the executable hellomake. MISSING: dependency on .h file

Makefile - Example

```
CC=gcc
CFLAGS=-I.
DEPS = hellomake.h

%.o: %.c $(DEPS)
	$(CC) -c -o  $@  $<  $(CFLAGS)
hellomake: hellomake.o hellofunc.o
	gcc -o hellomake hellomake.o hellofunc.o -I.
```

https://www.gnu.org/software/make/manual/html_node/Automatic-Variables.html#Automatic-Variables

Makefile - Example

```
CC=gcc
CFLAGS=-I.
DEPS = hellomake.h
OBJ = hellomake.o hellofunc.o

%.o: %.c $(DEPS)
	$(CC) -c -o $@ $< $(CFLAGS)

hellomake: $(OBJ)
	gcc -o $@ $^ $(CFLAGS)
```

References

http://www.cs.colby.edu/maxwell/courses/tutorials/maketutor/

http://linoxide.com/linux-command/linux-diff-command-examples/

http://www.thegeekstuff.com/2014/12/patch-command-examples/

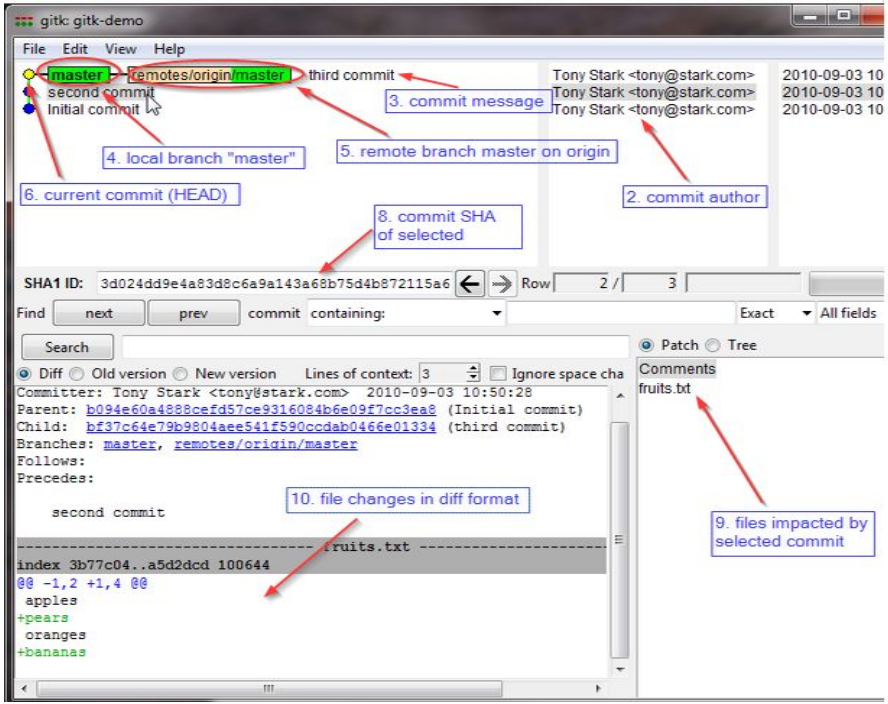
Homework

- Publish patch you made in lab 4
- Create a new branch “quote” off of version 3.0
- Branch command + checkout command (git branch quote v3.0; git checkout quote)
- $ git checkout v3.0 -b quote
- Use patch from lab 4 to modify this branch
- Patch command
- $ patch –pnum < quote-3.0-patch.txt
- Modify ChangeLog-2008 file in diffutils directory
- Add entry for your changes similar to entries in ChangeLog

Homework

- Commit changes to the new branch
- $ git add . $ git commit –F <Changelog file>
- Generate a patch that other people can use to get your changes
- $ git format-patch -<n> --stdout > formatted-patch.txt
- Test your partner’s patch
- Check out version 3.0 into a tmp branch
- Apply patch with git am command: $ git am < formatted-patch.txt
- Build and test with $ make check
- Make sure partner’s name is in HW4.txt for #8

Gitk



- A repository browser
- Visualizes commit graphs
- Used to understand the structure of the repo
- Tutorial: http://lostechies.com/joshuaflanagan/2010/09/03/use-gitk-to-understand-git/

Gitk

- SSH into the server with X11 enabled
  - ssh -X for OS with terminal (OS X, Linux)
  - Select “X11” option if using putty (Windows)
- Run gitk in the ~eggert/src/gnu/emacs directory
  - Need to first update your PATH
    - $ export PATH=/usr/local/cs/bin:$PATH
  - Run X locally before running gitk
    - Xming on Windows