Week 4
Diff and Patching

28 January 2019

CS 35L Lab 4

Jeremy Rotman

Announcements

- Assignment #3 is due Saturday by 11:55pm
- For Assignment #10
  - You should begin to choose stories
  - Email me to tell me what you are choosing at least a week before you have to present
  - Here is the link to see what stories people have signed up for already
  - Here is the link to sign up to present
    - **Sign up to present by Friday, February 1st**
- Still no official news on the final
  - Default is that individual TA's will write and proctor finals

Questions?

Outline

- ➔ diff files
- ➔ Patches
- ➔ Short intro on Python

How to read diff output

- When you run normal diff
  - Outputs differences without context
- < indicates the lines that are unique to file 1
- > indicates the lines that are unique to file 2

```
change-command
< from-file-line
< from-file-line
---
> to-file-line
> to-file-line
```

How to read diff output

- change-command has one of 3 formats
  - “*l*a*r*”
    - Add lines in range *r* from the 2nd file, starting after line *l* in the 1st file
  - “*f*c*t*”
    - Change lines in range *f* from the 1st file to range *t* from the 2nd file
  - “*r*d*l*”
    - Delete the lines in range *r* from the 1st file, line *l* is where they would have appeared in the 2nd file if not deleted

```
change-command
< from-file-line
< from-file-line
---
> to-file-line
> to-file-line
```

Unified Diff Output

- If you use the -u option, it creates the unified diff output
- Unified diff output gives us more context to see differences
  - It shows you the lines from both files
    - In the hunks that differ

Unified Diff Output

```
--- from-file from-file-modification-time
+++ to-file   to-file-modification-time

@@ from-file-line-numbers to-file-line-numbers @@
Line-from-either-file
Line-from-either-file
+Line-added-to-first-file
-Line-removed-from-second-file
Line-from-either-file
```

Unified Diff Output

- Note that the first file is “-” and the second file is “+”
- Another way to differentiate is to match the symbol in the header to lines
  - The file that has +++ in front of it in the header, has the lines with a + in front of them

An Example: Two files

lao

The Way that can be told of is not the eternal Way;
The name that can be named is not the eternal name.
The Nameless is the origin of Heaven and Earth;
The Named is the mother of all things.
Therefore let there always be non-being,
  so we may see their subtlety,
And let there always be being,
  so we may see their outcome.
The two are the same,
But after they are produced,
  they have different names.

tzu

The Nameless is the origin of Heaven and Earth;
The named is the mother of all things.

Therefore let there always be non-being,
  so we may see their subtlety,
And let there always be being,
  so we may see their outcome.
The two are the same,
But after they are produced,
  they have different names.
They both may be called deep and profound.
Deeper and more profound,
The door of all subtleties!

An Example: Unified Diff Output

```
--- lao 2002-02-21 23:30:39.942229878 -0800
+++ tzu 2002-02-21 23:30:50.442260588 -0800

@@ -1,7 +1,6 @@
-The Way that can be told of is not the eternal Way;
-The name that can be named is not the eternal name.
The Nameless is the origin of Heaven and Earth;
-The Named is the mother of all things.
+The named is the mother of all things.
+
Therefore let there always be non-being,
  so we may see their subtlety,
And let there always be being,
```

An Example: Unified Diff Output

```
@@ -9,3 +8,6 @@
The two are the same,
But after they are produced,
  they have different names.
+They both may be called deep and profound.
+Deeper and more profound,
+The door of all subtleties!
```

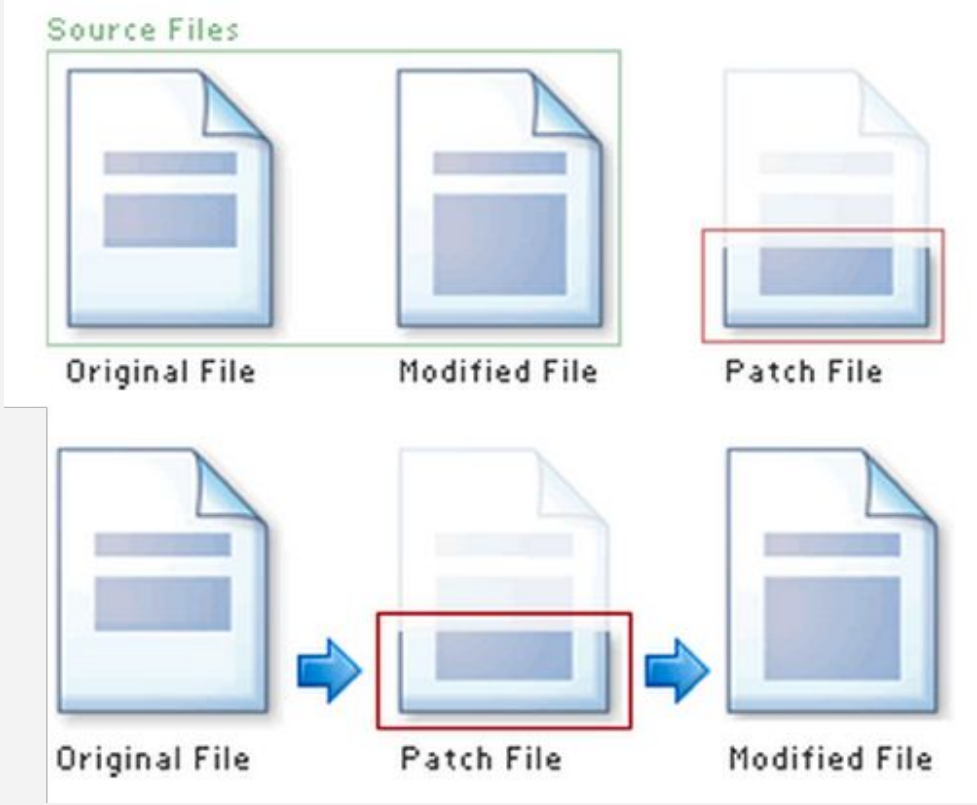
Patching

- ➔ You may be very familiar with the term if you play video games
- ➔ But what exactly is a patch?

Patching

- You may be very familiar with the term if you play video games
- But what exactly is a patch?
  - A piece of software designed to fix problems with or update a computer program
- A Patchfile is just a diff file that includes the changes made to a file
- The patch command can be used with the diff file to add changes

Applying a Patch



The patch command

- **`patch -p`***`num < patchfile`*
  - The -p option allows you to strip the smallest prefix containing *num* leading slashes
  - In particular this relates to filenames within the patchfile
  - If you have a different directory for your filenames, this is important to use
  - E.g. if you had the filename /u/home/j/jeremy/src/coreutils
    - -p0 gives /u/home/j/jeremy/src/coreutils
    - -p1 gives u/home/j/jeremy/src/coreutils
    - -p5 gives src/coreutils

Python

➔ What is Python?

What is Python?

- A scripting language
- Also an object-oriented language
  - Allows classes and member functions
- Easier to read than C
- Very popular language

Python Quirks

- Whitespace matters!
  - There are no curly braces or closing keywords (e.g. fi, done, etc.)
  - The indentation of a line makes a difference
- Tabs vs. Spaces
  - Some text editors include tab characters, some use spaces
  - Sometimes Python can run into errors since those are not equal
  - Just make sure to be consistent!

Lists

- A Python list is similar to a C++ array
  - Dynamic
    - It expands as needed when new items are added
  - Heterogeneous
    - It can hold objects of different type
    - Ie it can hold both integers and strings
- Accessing elements
  - List_name[*index*]
  - List_name[*start*:*end*]
    - Start is included, but end is not

Lists

- Adding elements to a list
  - List1 = [7, 8, “nine”]
  - List1.append(“ten”)
  - print List1
    - [7, 8, ‘nine’, ‘ten’]
- Merging Lists
  - List2 = [“this”, “that”]
  - List3 = [“these”, “those”]
  - List2 + List3
    - [‘this’, ‘that’, ‘these’, ‘those’]

Lists

➔ Looping over Python list

```
For element in List1:
    #Do stuff with the element
```

Dictionaries

- Similar to hash tables
- Stores data as key-value pairs
- Dict = {}
  - Creates an empty dictionary called Dict
- Keys are unique
  - Values are not necessarily unique
  - Keys must also be immutable

Dictionaries

➔
```
Dict1 = {}
Dict1[“ten”] = 10
print Dict1[“ten”]
   10
```

➔
```
Meaning = {}
Meaning[42] = [“life”, “universe”]
Meaning[42].append(“everything”)
print Meaning[42]
   [‘life’, ‘universe’, ‘everything’]
```

Dictionaries

➔ Testing within a dictionary:

```
if key in dict:
    dict[key].append(val)
else:
    dict[key] = [val]
```

➔ Iterating over a dictionary

```
for key in dict:    # only gives you keys
for key,value in dict.iteritems():  # Python 2
for key,value in dict.items():#Python 3
```

For Loops

- Python for loops generally iterate over an object
  - Such as a list
- If you need to iterate over indexes:

```
for i in range(len(list)):
    print i
```

Building coreutils

- A note for when you are working on the lab
  - Read the INSTALL file on how to configure “make”
    - Hint: Take a close look at the --prefix flag
  - When you run the configure script, make sure you are using the prefix flag
    - When you are done, coreutils should be installed in ~/coreutilsInstall
      - Or whatever you name our temporary directory

More hints for Lab 3

- To reproduce the bug make sure you are using ./ls and not just ls
  - Why?
- After patching, you should run make to rebuild your patched file
  - Don’t install, just make
  - man autoreconf
- Wednesday we will take a closer look at the python section (homework)