Diff & patch utility

Zilogic Systems

1. Need for file comparison & update tools

- Comparison between different versions of the same file.
- Multiple versions of file need not be stored.
- Contribution to open source projects are to be sent as difference files called patches.
- Patches that are submitted to open source projects are not always incorporated into the project due to strict filtering. Such patches need to be locally maintained and applied to the project.

2. `diff` utility

- Compare files line by line.
- Difference between files can be shown in various formats.
- `Unified` format is the widely used context based format.

```
$ diff -u <old-file> <new-file>
```

-u unified format

3. Unified format

```
--- old-file    old-file-modification-time
+++ new-file    new-file-modification-time
@@ -start,count +start,count @@
```

- '+' indicates line is added to the old file
- '-' indicates line is removed from the old file
- ' ' indicates line was unchanged
- A set of changes grouped together is called a `Hunk`.

```
$ diff -u test1 test2
--- test1       2014-05-21 12:01:30.000000000 +0530
+++ test2       2014-05-21 11:44:25.000000000 +0530
@@ -1,6 +1,7 @@            ❶
 aaa
 bbb                       ❷
 ccc
+ddd                       ❸
 eee
 fff                       ❹
 ggg
@@ -8,11 +9,8 @@
 iii
 jjj
 kkk
-uuu
```

```
-zzz
 lll
 mmm
-yyy
 nnn
 ooo
 ppp
@@ -24,4 +22,5 @@
 vvv
 www
 xxx
+yyy
 zzz
```

- A `Hunk` contains the following parts

❶ Referred to as a `Hunk`
❷ Unchanged context line
❸ Line to be added to first file
❹ Unchanged context line

4. Creating a Patch

- Patch is created one level above the project's top level directory
- The standard way to create patches is as follows

```
$ diff -Naur <old-directory> <new-directory>
```

-N adds information for new or deleted files
-a lets the patch update non-text files
-u unified context
-r lets the patch update subdirectories

5. Patch

- The diff/patch file is used by the `patch` utility to update a file.
- Patch is usually applied from the top-level directory of the project.

```
$ patch -p<NUM> < patchfile
```

- -p0 : uses entire path in patch file
- -p<NUM> : path after removing <NUM> number of slashes
- Rejected hunks while patching <file> is saved as <file>.rej
- A backup file is stored as <file>.orig
- The `--dry-run` option prints the results of applying the patches without actually changing any files.
- -R : reverses the patch file by swapping the old and new files.

References

- http://www.gnu.org/software/diffutils/manual/diffutils.pdf