Git - resolver conflictos usando Meld

por Federico Aloi

Va un pequeño paso a paso de cómo resolver conflictos utilizando una herramienta gráfica, Meld. Si bien esta mini guía está pensada para ser usada desde Linux, entiendo que también funcionaría desde Windows.

Paso 0 - por única vez
Bajarse *meld* (o el resolvedor de conflictos que más les guste). Este se instala haciendo *sudo apt-get install meld*.

Paso 1
Tener todo lo propio commiteado. Se puede chequear si esto es así usando el comando *git status* - informa qué cambios hubo y en que situación estamos respecto al repositorio remoto.
Hacia el final del texto que escupe debería decir *nothing to commit, working tree clean*.

Paso 2
Bajar los cambios del repositorio remoto con *git pull origin master*. Ahí Git se va a quejar porque encontró un conflicto, con un mensaje parecido a este:

```
Auto-merging src/ar/unq/edu/cpi/toxitaxi/ui/driver1/DriverPage.html
CONFLICT (content): Merge conflict in
src/ar/unq/edu/cpi/toxitaxi/ui/driver1/DriverPage.html
Automatic merge failed; fix conflicts and then commit the result.
```

Eso quiere decir que intentó hacer el merge automáticamente en el archivo *DriverPage.html* pero no pudo, y deberemos resolverlo “a mano”.

Paso 3
Invocar a meld con el comando *git mergetool*. Git nos va a poner uno a uno los archivos que tuvieron conflicto, y por qué fue (modificado por ambos, eliminado y modificado, etc):

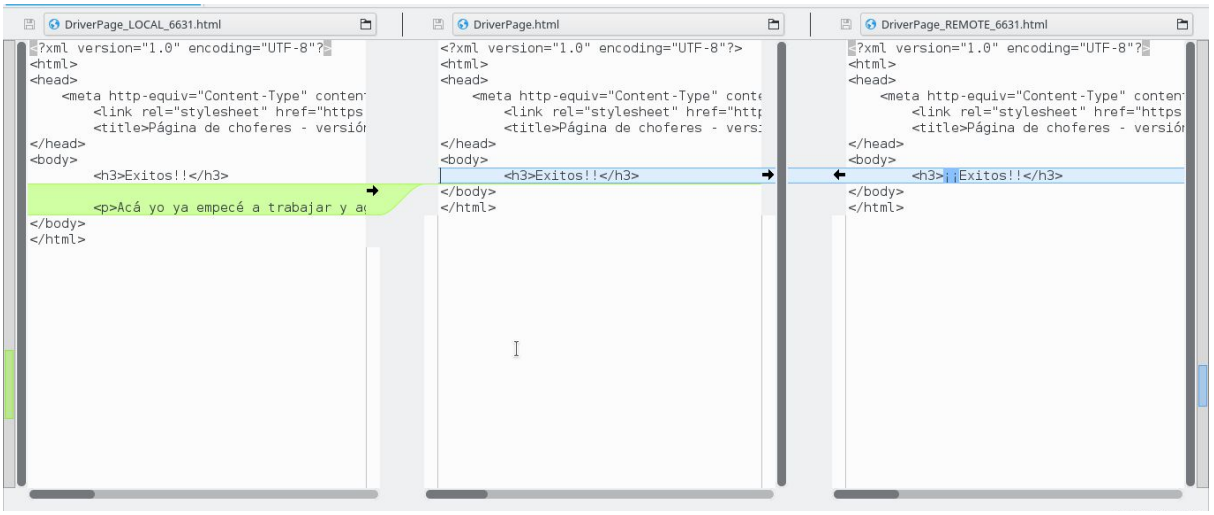
```
Normal merge conflict for
'src/ar/unq/edu/cpi/toxitaxi/ui/driver1/DriverPage.html':
  {local}: modified file
  {remote}: modified file
Hit return to start merge resolution tool (meld):
```

Ahí deberemos apretar Enter para que se nos abra el Meld.

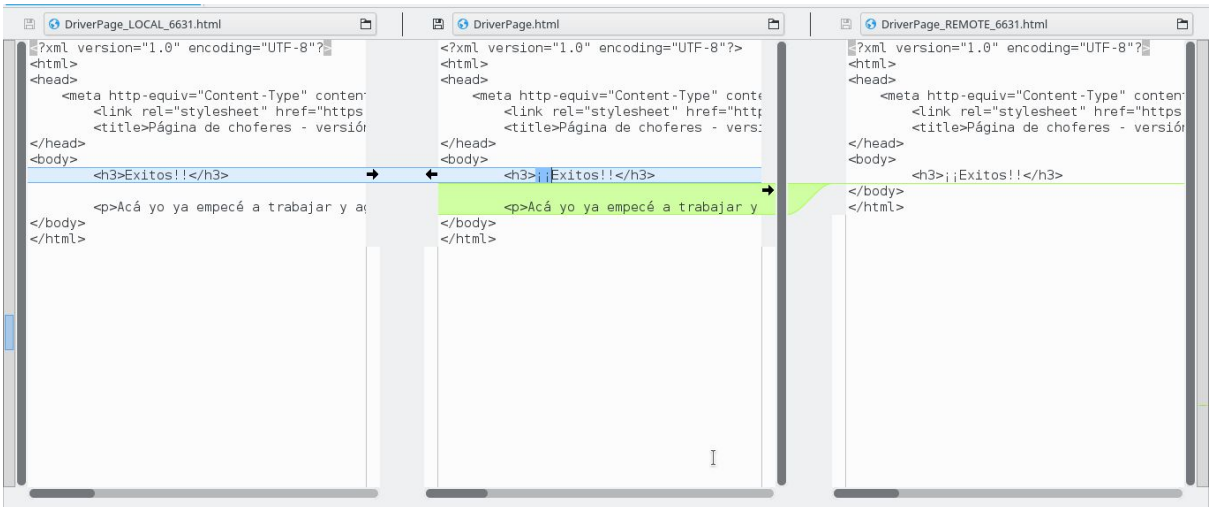
Paso 4

Meld nos mostrará una pantalla como la siguiente, donde veremos tres versiones del archivo que está en conflicto:

1) La que tenemos en nuestra compu, llamada LOCAL.
2) El punto de partida entre la nuestra y la que está en el repo remoto. En otras palabras, cómo estaba el archivo antes de que los dos (local y remoto) le hagamos cambios.
3) La versión que está subida al repo remoto, llamada REMOTE.



Con la ayuda de las flechitas, y tal vez algo de copiar-pegar, haremos que en el panel del medio quede la versión que integra a la de los costados, incorporando o descartando los cambios que querramos. En nuestro ejemplo, vamos a integrar cosas de ambas versiones:



Cuando terminemos con esto, guardamos (*Ctrl* + *S*, o botón *Save*) y cerramos Meld.

Paso 5
Si tenemos más archivos con conflictos, se volverá al Paso 3: Git nos pedirá que resolvamos el conflicto de la misma manera. Si no, avanzamos al paso siguiente.

Paso 6
El comando *git mergetool* nos devolverá el control, y la salida de *git status* será algo como esto:

```
All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
      modified:   src/ar/unq/edu/cpi/toxitaxi/ui/driver1/DriverPage.html
```

Finalizaremos nuestro merge ejecutando un *git commit*, el cual nos sugerirá el siguiente mensaje:

```
Merge remote-tracking branch 'origin/master' into HEAD

# Conflicts:
#      src/ar/unq/edu/cpi/toxitaxi/ui/driver1/DriverPage.html
```

Si estamos usando *nano*, saldremos de esa pantalla con *Control* + *X* y nuestro commit de merge habrá finalizado.