Tutorial básico de GIT – Parte 3

Introducción

En este cuarto, y último post, vamos a trabajar la operación merge en base a los conceptos introducidos en el anterior post. Crearemos un un conflicto manualmente y seguidamente lo resolveremos.

- Crear un conflicto
- Resolver un conflicto

Crear un conflicto

En esta sección, vamos a aprender a resolver un conflicto. Para ello vamos a crear manualmente un conflicto utilizando nuestros dos repositorios existentes «tutorial» y «tutorial2».

Trabajando en tutorial

En primer lugar, abriremos el archivo «sample.txt» en el directorio «tutorial». Añadiremos texto al fichero y haremos un commit.

```
$ echo "Each language has its purpose, but do not program in
COBOL if you can avoid it." >> sample.txt
$ git add sample.txt
$ git commit -m "added new programming advice"
[master ef95d28] added new programming advice
 1 file changed, 1 insertion(+)
```

Trabajando en tutorial2

A continuación, abriremos el archivo «sample.txt» en el directorio «tutorial2». Añadiremos texto al fichero y haremos un commit.

```
$ echo "The spirit and intent of the program should be retained throughout." >> sample.txt
$ git add sample.txt
$ git commit -m "added new coding advice"
[master f4ddc9c] added new coding advice
 1 file changed, 1 insertion(+)
```

Ahora haremos un push para subir los cambios de «tutorial2» al repositorio remoto.

```
$ git push
Username:
Password:
Counting objects: 3, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 356 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://gitlab.com/nebur/tutorial.git
   178cb16..f4ddc9c  master -> master
```

En nuestro repositorio remoto actual, el archivo «sample.txt» contiene la tercera línea «The spirit and intent of the program should be retained throughout.» y se ha realizado el commit al histórico.

Trabajando en tutorial

Ahora, vamos a hacer un push del commit realizado en nuestro repositorio «tutorial» al repositorio remoto.

```
$ git push
Username:
Password:
To https://gitlab.com/nebur/tutorial.git
 ! [rejected]        master -> master (fetch first)
```

```
error: failed to push some refs to
'https://gitlab.com/nebur/tutorial.git'
hint: Updates were rejected because the remote contains work
that you do
hint: not have locally. This is usually caused by another
repository pushing
hint: to the same ref. You may want to first integrate the
remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help'
for details.
```

Como podemos observar, Git retorna un conflicto y rechaza el push.

Resolver un conflicto

Con el fin de realizar el push del cambio en «tutorial» al repositorio remoto, vamos a tener que resolver el conflicto manualmente. Vamos a ejecutar un pull para adquirir los cambios más recientes desdel repositorio remoto.

Trabajando en tutorial

Ejecutaremos el siguiente comando

```
$ git pull origin master
```

```
Username:
Password:
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
From https://gitlab.com/nebur/tutorial
 * branch            master     -> FETCH_HEAD
   178cb16..f4ddc9c  master     -> origin/master
Auto-merging sample.txt
CONFLICT (content): Merge conflict in sample.txt
Automatic merge failed; fix conflicts and then commit the
```

result.

Debería aparecer un mensaje advirtiéndonos de un conflicto de merge.

En abrir «sample.txt», podremos ver una sección del archivo delimitada por marcadores que ha añadido Git para indicarnos donde se encuentra el conflicto.

```
$ cat sample.txt
After three days without programming, life becomes meaningless.
new line text
<<<<<<< HEAD Each language has its purpose, but do not program in COBOL if you can avoid it. ======= The spirit and intent of the program should be retained throughout. >>>>>>> f4ddc9c9a3d3329f8ad799ae582adf9efe631211
```

Vamos a resolver el conflicto aceptando los cambios y quitando el marcador.

```
$ cat sample.txt
After three days without programming, life becomes meaningless.
new line text
Each language has its purpose, but do not program in COBOL if you can avoid it.
The spirit and intent of the program should be retained throughout.
```

Una vez editado el fichero podremos proceder a realizar el commit

```
$ git add sample.txt
$ git commit -m "merge"
[master 4351226] merge
```

Ahora estamos al día con los últimos cambios del repositorio remoto.
Podemos comprobar la integridad del histórico del repositorio usando el comando «log». La opción –graph mostrará el histórico del branch en un formato gráfico y la opción

–oneline tratará de compactar el mensaje de salida.

```
$ git log --graph --oneline
*   4351226 merge
|\
| * f4ddc9c added new coding advice
* | ef95d28 added new programming advice
|/
* 178cb16 new line appened
* 0742011 first commit
```

Esto indica que las dos historias se han fusionado de manera segura con un nuevo commit del merge realizado a mano.

Podemos hacer un push con la seguridad seguridad de que este cambio no provocará un conflicto en el repositorio remoto.

Observaciones

Y con este último post damos por finalizada esta serie de tutoriales sobre Git. De una forma básica y cercana hemos visto como trabajar con este sistema de control de versiones. No hemos mostrado todo su potencial, pero si introducido conceptos básico y las herramientas para empezar a trabajar con él. En una nueva serie profundizarmos en conceptos mucho más potente, por ahora podemos empezar experimentar con el.

Ruben.