Prácticas de control de versiones con Git

Alfredo Sánchez Alberca
asalber@ceu.es
https://aprendeconalf.es

Tabla de contenidos

Prefacio

Esta obra es una colección de ejercicios prácticos resueltos sobre el sistema de control de versiones Git.

Los ejercicios están clasificados por temas y siguen el orden habitual en el aprendizaje de este sistema de control de versiones:

1. Creación y actualización de repositorios
2. Manejo del historial de cambios
3. Deshacer cambios
4. Gestión de ramas
5. Repositorios remotos

Cada capítulo comienza indicando los requisitos previos para poder realizar sus ejercicios. En la mayoría de los casos es necesario haber completado los ejercicios del capítulo anterior, pero también se ofrece una secuencia de comandos para clonar el repositorio remoto `libro-git` y situarlo en el estado de partida adecuado.

Cada ejercicio incluye un enunciado y, de manera plegable, la solución con los comandos necesarios para resolverlo y una animación que muestra su resolución paso a paso.

> **Convenio de notación**
>
> En los bloques de comandos, el símbolo `>` representa el *prompt* de la terminal y no debe teclearse. La secuencia `Ctrl+D` indica que hay que pulsar simultáneamente las teclas `Control` y `D` para terminar la entrada de texto por teclado con el comando `cat`.

Para aprender los conceptos y comandos necesarios para resolver estos ejercicios puede consultarse el manual de Git.

Licencia

1 Creación y actualización de repositorios

Requisitos previos

Para realizar estos ejercicios solo es necesario tener Git instalado y acceso a una terminal.

Ejercicio 1.1. Configurar Git definiendo el nombre del usuario, el correo electrónico y activar el coloreado de la salida. Mostrar la configuración final.

Solución

```
> git config --global user.name "Tu-Nombre-Completo"
> git config --global user.email "tu-correo-electrónico"
> git config --global color.ui auto
> git config --list
```

Ejercicio 1.2. Crear un repositorio nuevo con el nombre `libro` y mostrar su contenido.

Solución

```
> mkdir libro
> cd libro
> git init
> ls -la
```

Ejercicio 1.3.

1. Comprobar el estado del repositorio.
2. Crear un fichero `indice.txt` con el siguiente contenido:

   Capítulo 1: Introducción a Git Capítulo 2: Flujo de trabajo básico Capítulo 3: Repositorios remotos

3. Comprobar de nuevo el estado del repositorio.

4. Añadir el fichero a la zona de intercambio temporal.
5. Volver a comprobar una vez más el estado del repositorio.

Solución

```
> git status
> cat > indice.txt
Capítulo 1: Introducción a Git
Capítulo 2: Flujo de trabajo básico
Capítulo 3: Repositorios remotos
Ctrl+D
> git status
> git add indice.txt
> git status
```

Ejercicio 1.4. Realizar un commit de los últimos cambios con el mensaje `Añadido índice del libro.` y ver el estado del repositorio.

Solución

```
> git commit -m "Añadido índice del libro."
> git status
```

Ejercicio 1.5.

1. Cambiar el fichero `indice.txt` para que contenga lo siguiente:

   Capítulo 1: Introducción a Git Capítulo 2: Flujo de trabajo básico Capítulo 3: Gestión de ramas Capítulo 4: Repositorios remotos

2. Mostrar los cambios con respecto a la última versión guardada en el repositorio.
3. Hacer un commit de los cambios con el mensaje `Añadido capítulo 3 sobre gestión de ramas`.

Solución

```
> cat > indice.txt
Capítulo 1: Introducción a Git
Capítulo 2: Flujo de trabajo básico
Capítulo 3: Gestión de ramas
Capítulo 4: Repositorios remotos
Ctrl+D
> git diff
> git add indice.txt
> git commit -m "Añadido capítulo 3 sobre gestión de ramas"
```

Ejercicio 1.6.

1. Mostrar los cambios de la última versión del repositorio con respecto a la anterior.
2. Cambiar el mensaje del último commit por `Añadido capítulo 3 sobre gestión de ramas al índice.`
3. Volver a mostrar los últimos cambios del repositorio.

Solución

```
> git show
> git commit --amend -m "Añadido capítulo 3 sobre gestión de ramas
 ↪  al índice."
> git show
```

2 Manejo del historial de cambios

Requisitos previos

Para hacer estos ejercicios es necesario haber hecho antes los ejercicios de creación y actualización de repositorios, o bien hacer un clon del repositorio remoto https://github.com/asalber/libro-git mediante la siguiente secuencia de comandos:

```
> git clone https://github.com/asalber/libro-git.git
> cd libro-git
> git reset --hard 8c808
> git remote remove origin
```

Ejercicio 2.1.

1. Mostrar el historial de cambios del repositorio.
2. Crear la carpeta `capitulos` y crear dentro de ella el fichero `capitulo1.txt` con el siguiente texto:

   > Git es un sistema de control de versiones ideado por Linus Torvalds.

3. Añadir los cambios a la zona de intercambio temporal.
4. Hacer un commit de los cambios con el mensaje `Añadido capítulo 1.`
5. Volver a mostrar el historial de cambios del repositorio.

Solución

```
> git log
> mkdir capitulos
> cat > capitulos/capitulo1.txt
Git es un sistema de control de versiones ideado por Linus Torvalds.
Ctrl+D
> git add .
> git commit -m "Añadido capítulo 1."
> git log
```

Ejercicio 2.2.

1. Crear el fichero `capitulo2.txt` en la carpeta `capitulos` con el siguiente texto:

   > El flujo de trabajo básico con Git consiste en:
   >
   > 1. Hacer cambios en el repositorio.
   > 2. Añadir los cambios a la zona de intercambio temporal.
   > 3. Hacer un commit de los cambios.

2. Añadir los cambios a la zona de intercambio temporal.
3. Hacer un commit de los cambios con el mensaje `Añadido capítulo 2.`
4. Mostrar las diferencias entre la última versión y dos versiones anteriores.

Solución

```
> cat > capitulos/capitulo2.txt
El flujo de trabajo básico con Git consiste en:
1- Hacer cambios en el repositorio.
2- Añadir los cambios a la zona de intercambio temporal.
3- Hacer un commit de los cambios.
Ctrl+D
> git add .
> git commit -m "Añadido capítulo 2."
> git diff HEAD~2..HEAD
```

Ejercicio 2.3.

1. Crear el fichero `capitulo3.txt` en la carpeta `capitulos` con el siguiente texto:

   > Git permite la creación de ramas lo que permite tener distintas versiones del mismo proyecto y trabajar de manera simultanea en ellas.

2. Añadir los cambios a la zona de intercambio temporal.
3. Hacer un commit de los cambios con el mensaje `Añadido capítulo 3.`
4. Mostrar las diferencias entre la primera y la última versión del repositorio.

Solución

```
> cat > capitulos/capitulo3.txt
Git permite la creación de ramas lo que permite tener distintas
 ↪  versiones del mismo proyecto y trabajar de manera simultanea en
 ↪  ellas.
Ctrl+D
> git add .
> git commit -m "Añadido capítulo 3."
> git log
> git diff <código hash de la primera versión>..HEAD
```

Ejercicio 2.4.

1. Añadir al final del fichero `indice.txt` la siguiente línea:

   Capítulo 5: Conceptos avanzados

2. Añadir los cambios a la zona de intercambio temporal.
3. Hacer un commit de los cambios con el mensaje `Añadido capítulo 5 al índice.`
4. Mostrar quién ha hecho cambios sobre el fichero `indice.txt`.

Solución

```
> echo "Capítulo 5: Conceptos avanzados" >> indice.txt
> git add .
> git commit -m "Añadido capítulo 5 al índice."
> git annotate indice.txt
```

3 Deshacer cambios

! Requisitos previos

Para hacer estos ejercicios es necesario haber hecho antes los ejercicios sobre el historial de cambios, o bien hacer un clon del repositorio remoto https://github.com/asalber/libro-git mediante la siguiente secuencia de comandos:

```
> git clone https://github.com/asalber/libro-git.git
> cd libro-git
> git reset --hard 48ed8
> git remote remove origin
```

Ejercicio 3.1.

1. Eliminar la última línea del fichero `indice.txt` y guardarlo.
2. Comprobar el estado del repositorio.
3. Deshacer los cambios realizados en el fichero `indice.txt` para volver a la versión anterior del fichero.
4. Volver a comprobar el estado del repositorio.

Solución

```
> nano indice.txt
# Eliminar la última línea y guardar el fichero.
> git status
> git checkout -- indice.txt
> git status
```

Ejercicio 3.2.

1. Eliminar la última línea del fichero `indice.txt` y guardarlo.
2. Añadir los cambios a la zona de intercambio temporal.
3. Comprobar de nuevo el estado del repositorio.
4. Quitar los cambios de la zona de intercambio temporal, pero mantenerlos en el directorio de trabajo.

5. Comprobar de nuevo el estado del repositorio.
6. Deshacer los cambios realizados en el fichero `indice.txt` para volver a la versión anterior del fichero.
7. Volver a comprobar el estado del repositorio.

Solución

```
> nano indice.txt
# Eliminar la última línea y guardar el fichero.
> git add .
> git status
> git reset indice.txt
> git status
> git checkout -- indice.txt
> git status
```

Ejercicio 3.3.

1. Eliminar la última línea del fichero `indice.txt` y guardarlo.
2. Eliminar el fichero `capitulos/capitulo3.txt`.
3. Añadir un fichero nuevo `capitulos/capitulo4.txt` vacío.
4. Añadir los cambios a la zona de intercambio temporal.
5. Comprobar de nuevo el estado del repositorio.
6. Quitar los cambios de la zona de intercambio temporal, pero mantenerlos en el directorio de trabajo.
7. Comprobar de nuevo el estado del repositorio.
8. Deshacer los cambios realizados para volver a la versión del repositorio.
9. Volver a comprobar el estado del repositorio.

Solución

```
> nano indice.txt
# Eliminar la última línea y guardar el fichero.
> rm capitulos/capitulo3.txt
> touch capitulos/capitulo4.txt
> git add .
> git status
> git reset
> git status
> git checkout -- .
> git status
> git clean -f
> git status
```

Ejercicio 3.4.

1. Eliminar la última línea del fichero `indice.txt` y guardarlo.
2. Eliminar el fichero `capitulos/capitulo3.txt`.
3. Añadir los cambios a la zona de intercambio temporal y hacer un commit con el mensaje `Borrado accidental.`
4. Comprobar el historial del repositorio.
5. Deshacer el último commit pero mantener los cambios anteriores en el directorio de trabajo y la zona de intercambio temporal.
6. Comprobar el historial y el estado del repositorio.
7. Volver a hacer el commit con el mismo mensaje de antes.
8. Deshacer el último commit y los cambios anteriores del directorio de trabajo volviendo a la versión anterior del repositorio.
9. Comprobar de nuevo el historial y el estado del repositorio.

Solución

```
> nano indice.txt
# Eliminar la última línea y guardar el fichero.
> rm capitulos/capitulo3.txt
> git commit -a -m "Borrado accidental."
> git status
> git log
> git reset --soft HEAD~1
> git status
> git commit -m "Borrado accidental."
> git status
> git log
> git reset --hard HEAD~1
> git log
> git status
```

4 Gestión de ramas

Requisitos previos

Para hacer estos ejercicios es necesario haber hecho antes los ejercicios sobre el historial de cambios, o bien hacer un clon del repositorio remoto https://github.com/asalber/libro-git mediante la siguiente secuencia de comandos:

```
> git clone https://github.com/asalber/libro-git.git
> cd libro-git
> git reset --hard 48ed8
> git remote remove origin
```

Ejercicio 4.1. Crear una nueva rama `bibliografia` y mostrar las ramas del repositorio.

Solución

```
> git branch bibliografia
> git branch -av
```

Ejercicio 4.2.

1. Crear el fichero `capitulos/capitulo4.txt` y añadir el texto siguiente:

   > En este capítulo veremos cómo usar GitHub para alojar repositorios en remoto.

2. Añadir los cambios a la zona de intercambio temporal.
3. Hacer un commit con el mensaje `Añadido capítulo 4.`
4. Mostrar la historia del repositorio incluyendo todas las ramas.

Solución

```
> cat > capitulos/capitulo4.txt
En este capítulo veremos cómo usar GitHub para alojar repositorios
 ↪  en remoto.
Ctrl+D
> git add .
> git commit -m "Añadido capítulo 4."
> git log --graph --all --oneline
```

Ejercicio 4.3.

1. Cambiar a la rama `bibliografia`.
2. Crear el fichero `bibliografia.txt` y añadir la siguiente referencia:
    - Chacon, S. and Straub, B. Pro Git. Apress.
3. Añadir los cambios a la zona de intercambio temporal.
4. Hacer un commit con el mensaje `Añadida primera referencia bibliográfica.`
5. Mostrar la historia del repositorio incluyendo todas las ramas.

Solución

```
> git checkout bibliografia
> cat > bibliografia.txt
- Chacon, S. and Straub, B. Pro Git. Apress.
Ctrl+D
> git add .
> git commit -m "Añadida primera referencia bibliográfica."
> git log --graph --all --oneline
```

Ejercicio 4.4.

1. Fusionar la rama `bibliografia` con la rama `master`.
2. Mostrar la historia del repositorio incluyendo todas las ramas.
3. Eliminar la rama `bibliografia`.
4. Mostrar de nuevo la historia del repositorio incluyendo todas las ramas.

Solución

```
> git checkout master
> git merge bibliografia
> git log --graph --all --oneline
> git branch -d bibliografia
> git log --graph --all --oneline
```

Ejercicio 4.5.

1. Crear la rama `bibliografia`.
2. Cambiar a la rama `bibliografia`.
3. Cambiar el fichero `bibliografia.txt` para que contenga las siguientes referencias:
   - Scott Chacon and Ben Straub. Pro Git. Apress.
   - Ryan Hodson. Ry's Git Tutorial. Smashwords (2014)
4. Añadir los cambios a la zona de intercambio temporal y hacer un commit con el mensaje `Añadida nueva referencia bibliográfica.`
5. Cambiar a la rama `master`.
6. Cambiar el fichero `bibliografia.txt` para que contenga las siguientes referencias:
   - Chacon, S. and Straub, B. Pro Git. Apress.
   - Loeliger, J. and McCullough, M. Version control with Git. O'Reilly.
7. Añadir los cambios a la zona de intercambio temporal y hacer un commit con el mensaje `Añadida nueva referencia bibliográfica.`
8. Fusionar la rama `bibliografia` con la rama `master`.
9. Resolver el conflicto dejando el fichero `bibliografia.txt` con las referencias:
   - Chacon, S. and Straub, B. Pro Git. Apress.
   - Loeliger, J. and McCullough, M. Version control with Git. O'Reilly.
   - Hodson, R. Ry's Git Tutorial. Smashwords (2014)
10. Añadir los cambios a la zona de intercambio temporal y hacer un commit con el mensaje `Resuelto conflicto de bibliografía.`
11. Mostrar la historia del repositorio incluyendo todas las ramas.

Solución

```
> git branch bibliografia
> git checkout bibliografia
> cat > bibliografia.txt
- Scott Chacon and Ben Straub. Pro Git. Apress.
- Ryan Hodson. Ry's Git Tutorial. Smashwords (2014)
Ctrl+D
> git commit -a -m "Añadida nueva referencia bibliográfica."
> git checkout master
> cat > bibliografia.txt
- Chacon, S. and Straub, B. Pro Git. Apress.
- Loeliger, J. and McCullough, M. Version control with Git.
 ↪  O'Reilly.
Ctrl+D
> git commit -a -m "Añadida nueva referencia bibliográfica."
> git merge bibliografia
> nano bibliografia.txt
# Editar el fichero para dejar las referencias indicadas y eliminar
 ↪  las marcas de conflicto.
> git commit -a -m "Resuelto conflicto de bibliografía."
> git log --graph --all --oneline
```

5 Repositorios remotos

Requisitos previos

Para hacer estos ejercicios es necesario haber hecho antes los ejercicios sobre ramas, o bien hacer un clon del repositorio remoto https://github.com/asalber/libro-git mediante la siguiente secuencia de comandos:

```
> git clone https://github.com/asalber/libro-git.git
> cd libro-git
> git reset --hard cb1e4
> git remote remove origin
```

Además, es necesario disponer de una cuenta en GitHub.

Ejercicio 5.1.

1. Crear un nuevo repositorio público en GitHub con el nombre `libro-git`.
2. Añadirlo al repositorio local del libro.
3. Mostrar todos los repositorios remotos configurados.

Solución

```
# Crear el repositorio en GitHub y copiar su url con protocolo
 ↪  https.
> git remote add github <url>
> git remote -v
```

Ejercicio 5.2.

1. Añadir los cambios del repositorio local al repositorio remoto de GitHub.
2. Acceder a GitHub y comprobar que se han subido los cambios mostrando el historial de versiones.

Solución

```
> git push github master
```

Ejercicio 5.3.

1. Colaborar en el repositorio remoto `libro-git` de otro usuario.
2. Clonar su repositorio `libro-git`.
3. Añadir el fichero `autores.txt` que contenga el nombre del usuario y su correo electrónico.
4. Añadir los cambios a la zona de intercambio temporal.
5. Hacer un commit con el mensaje `Añadido autor.`
6. Subir los cambios al repositorio remoto.

Solución

```
# Entrar en GitHub en el proyecto libro-git del que seamos
 ↪  colaboradores y copiar la url.
> git clone <url>
> cat > autores.txt
# Escribir el nombre del autor y su correo.
Ctrl+D
> git add .
> git commit -m "Añadido autor."
> git push origin master
```

Ejercicio 5.4.

1. Hacer una bifurcación (*fork*) del repositorio remoto `asalber/libro-git` en GitHub.
2. Clonar el repositorio creado en la cuenta de GitHub del usuario.
3. Crear una nueva rama `autoria` y activarla.
4. Añadir el nombre del usuario y su correo al fichero `autores.txt`.
5. Añadir los cambios a la zona de intercambio temporal.
6. Hacer un commit con el mensaje `Añadido nuevo autor.`
7. Subir los cambios de la rama `autoria` al repositorio remoto en GitHub.
8. Hacer un *Pull Request* de los cambios en la rama `autoria`.

Solución

```
# Hacer el fork del repositorio asalber/libro-git en GitHub y copiar
 ↪  la url del
# repositorio creado en la cuenta de GitHub del usuario.
> git clone <url>
> git checkout -b autoria
# Editar con nano el fichero autores.txt y añadir el nombre y el
 ↪  correo
# electrónico del usuario en una nueva línea.
> git commit -a -m "Añadido nuevo autor."
> git push origin autoria
# Ir al repositorio remoto en GitHub y hacer clic en el botón
 ↪  «Compare & Pull
# Request» y después completar la solicitud haciendo clic en el
 ↪  botón «Create
# Pull Request».
```