Taller Práctico: Trabajando con Git (Push, Pull y Merge)

Julian F. Latorre
Bootcamp Desarrollo Web Full Stack

1. Introducción

Este taller práctico está diseñado para reforzar los conceptos de Git push, pull y merge que hemos cubierto en clase. A través de una serie de ejercicios, aplicarás estos conceptos en situaciones similares a las que encontrarás en proyectos reales.

2. Configuración Inicial

Antes de comenzar con los ejercicios, asegúrate de tener Git instalado en tu sistema. Si no lo tienes, puedes descargarlo de `https://git-scm.com/`.

Paso 1: Configuración de Git

Configura tu nombre de usuario y correo electrónico en Git con los siguientes comandos:

```
git config --global user.name "Tu Nombre"
git config --global user.email "tu@email.com"
```

3. Ejercicio 1: Creación y Push de un Repositorio Local

En este ejercicio, crearás un repositorio local y lo subirás a GitHub.

Pasos del Ejercicio 1

1. Crea un nuevo directorio para tu proyecto:

```
mkdir mi-proyecto-git
cd mi-proyecto-git
```

2. Inicializa un repositorio Git:

```
git init
```

3. Crea un archivo README.md con el siguiente contenido:

```
# Mi Proyecto Git

Este es un proyecto de práctica para aprender Git.
```

4. Agrega el archivo al staging area y haz un commit:

```
git add README.md
git commit -m "Primer commit: Agregar README.md"
```

5. Crea un nuevo repositorio en GitHub (sin inicializarlo con README).

6. Conecta tu repositorio local con el remoto:

```
git remote add origin https://github.com/tu-usuario/mi-
    proyecto-git.git
```

7. Haz push de tu rama main al repositorio remoto:

```
git push -u origin main
```

4. Ejercicio 2: Colaboración y Pull

En este ejercicio, simularemos una colaboración y practicaremos el uso de git pull.

Pasos del Ejercicio 2

1. En GitHub, edita el archivo README.md directamente en la interfaz web. Agrega una nueva línea:

```
Este cambio se hizo directamente en GitHub.
```

2. Haz commit de este cambio en GitHub.
3. En tu repositorio local, ejecuta:

```
git pull origin main
```

4. Verifica que el cambio se ha incorporado a tu repositorio local.

5. Ejercicio 3: Creación de Ramas y Merge

En este ejercicio, crearás una nueva rama, harás cambios en ella y luego la fusionarás con la rama principal.

Pasos del Ejercicio 3

1. Crea una nueva rama llamada "feature/nueva-seccion":

```
git checkout -b feature/nueva-seccion
```

2. Agrega una nueva sección al README.md:

```
## Nueva Sección

Esta es una nueva sección agregada en una rama separada.
```

3. Haz commit de estos cambios:

```
git add README.md
git commit -m "Agregar nueva sección al README"
```

4. Vuelve a la rama main:

```
git checkout main
```

5. Fusiona la rama feature/nueva-seccion en main:

```
git merge feature/nueva-seccion
```

6. Haz push de los cambios al repositorio remoto:

```
git push origin main
```

6. Ejercicio 4: Resolución de Conflictos

En este ejercicio, crearás un conflicto intencionalmente y aprenderás a resolverlo.

Pasos del Ejercicio 4

1. En GitHub, edita la primera línea del README.md:

```
# Proyecto Colaborativo Git
```

2. Haz commit de este cambio en GitHub.

3. En tu repositorio local, sin hacer pull, edita la misma línea del README.md:

```
# Proyecto de Aprendizaje Git
    \end{lstlisting]

    \item Haz commit de este cambio localmente:
    \begin{lstlisting}[language=bash]
git add README.md
git commit -m "Actualizar título del README"
```

4. Intenta hacer push al repositorio remoto:

```
git push origin main
```

5. Git te informará que necesitas hacer pull primero. Haz pull:

```
git pull origin main
```

6. Git te informará sobre el conflicto. Abre README.md y verás algo como esto:

```
<<<<<<< HEAD
# Proyecto de Aprendizaje Git
=======
# Proyecto Colaborativo Git
>>>>>>> abcdef1234567890
```

7. Edita el archivo para resolver el conflicto, por ejemplo:

```
# Proyecto Colaborativo de Aprendizaje Git
```

8. Marca el conflicto como resuelto:

```
git add README.md
```

9. Completa el merge:

```
git commit -m "Resolver conflicto en el título del
    README"
```


10. Finalmente, haz push de los cambios:

```
git push origin main
```

7. Ejercicio 5: Trabajo con Tags

En este ejercicio, aprenderás a crear y gestionar tags en Git.

Pasos del Ejercicio 5

1. Crea un tag ligero para marcar la versión actual de tu proyecto:

```
git tag v1.0
```

2. Crea un tag anotado con un mensaje:

```
git tag -a v1.1 -m "Versión 1.1 con nuevas caracterí
    sticas"
```

3. Visualiza tus tags:

```
git tag
```

4. Haz push de tus tags al repositorio remoto:

```
git push origin --tags
```

5. En GitHub, verifica que los tags están visibles en la sección de releases".

8. Conclusión

Reflexión Final

En este taller práctico, has aplicado los conceptos clave de Git push, pull y merge en situaciones que simulan el desarrollo colaborativo real. Has aprendido a:

- Crear y configurar un repositorio Git
- Hacer push de cambios locales a un repositorio remoto
- Traer cambios remotos a tu repositorio local con pull
- Crear y fusionar ramas
- Resolver conflictos de merge
- Trabajar con tags para versionar tu proyecto

Recuerda que la práctica es clave para dominar Git. Continúa utilizando estos comandos en tus proyectos personales y colaborativos para reforzar tu aprendizaje.
Próximos pasos: Investiga sobre flujos de trabajo Git como Git Flow o GitHub Flow, y cómo pueden mejorar la colaboración en proyectos más grandes.