

# Lectura 8

## Estrategias de git para devOps: rama principal (main), FF/no-FF, Squash y rebase

- La lectura tratará sobre estrategias de ramificación, técnicas de integración, mantenimiento de historial y fundamentos internos de git.

### 1) Estrategia de ramas y cultura Devops

- Integrar rapido implica medir el impacto de cada cambio en la estabilidad compartida.
- La cultura Devops reduce fricciones en revisiones y despliegues.
- **Merge Vs Rebasing**: Git ofrece 2 caminos principales para integrar líneas de trabajo.
  - El **merge** crea un nuevo commit con dos padres y preserva los historiales independientes.
  - El **rebase** "reaplica" la secuencia de commits sobre una nueva base, reescribiendo hashes.

### 2) Fast-forward: Integración simple y lineal

- En vez de crear un commit de merge, git mueve el puntero de la rama objetivo hasta la punta de la rama fuente, lo que deja un historial lineal y fácil de auditar.

### 3) No-Fast-Forward: preserva contexto de integración

- Esto crea un commit de merge con dos padres
- El commit de fusión conserva contexto y fecha del momento de la integración

### 4) Squash merge: consolidación atómica y reversible

- Reune todos los commits de la rama origen en un único conjunto de cambios listos para confirmar en la rama destino.

- La ventaja es que facilita revertir todo el aporte con un solo commit, pero la desventaja es que se pierde el detalle de la evolución de esa rama.

## 5) ¿Cuándo rebase y cuándo merge?

- Rebase es mejor en ramas de corta vida para tenerlas al día con la rama main y mantener un historial lineal antes de abrir un Pull Request
- Merge es preferible cuando se necesita documentar.

## 6) Fundamentos internos que respaldan la colaboración

- Git modela 4 estados: **untracked**, **unmodified**, **modified** y **staged**, lo que favorece el orden de cambios y trazabilidad
- Con **git cat-file** expone el contenido de un objeto y con **git show** presenta una vista legible del commit.
- Utiliza DAG donde cada commit referencia a su padre y árboles merkle donde cambiar un archivo altera el blob, y todo esto hace evidente cualquier manipulación.

## 7) Practicas esenciales orientadas a Devops

- Commit atómicos y mensajes útiles
- Políticas de integración como ver cual es mas adecuado: FF, NoFF, squash
- Pull request y pruebas
- Reversión y diagnostico con **git revert** para deshacer