



Servicio Nacional de Aprendizaje – SENA

Centro de Tecnología de la Manufactura Avanzada – Regional Antioquia

# Guía de aprendizaje 5: Ramificación (Branching) y Fusión (Merging)

Desarrollo Paralelo en Git: Creación, Navegación y Fusión de Ramas (Branches).

Tema que se va a ver en la guía de estudio

Concepto de rama (branch) y HEAD, creación y listado de ramas (git branch), movimiento entre ramas (git switch / git checkout), fusión de ramas (git merge) y resolución manual de conflictos.

## Objetivo general

- Implementar estrategias de desarrollo no lineal utilizando ramas para aislar el trabajo en nuevas funcionalidades o corrección de errores, y dominar el proceso de integración segura mediante la fusión.

## Objetivos específicos

- Entender las ramas como punteros a commits y el rol del puntero HEAD.
- Crear nuevas ramas y moverse entre ellas utilizando los comandos modernos.
- Ejecutar git merge para combinar el historial de dos ramas.
- Identificar un conflicto de fusión y resolverlo manualmente en el editor de código.
- Eliminar ramas una vez que su trabajo ha sido integrado.



Servicio Nacional de Aprendizaje – SENA

Centro de Tecnología de la Manufactura Avanzada – Regional Antioquia

## Introducción

La ramificación (branching) es la característica más potente de Git. Permite crear flujos de trabajo independientes, como "carreteras paralelas" que parten del tronco principal y pueden volver a unirse. El tronco principal se llama tradicionalmente master, pero GitHub y la industria están migrando a main. Cada rama es esencialmente un puntero ligero a un commit específico. El HEAD es el puntero que referencia la rama y el commit en el que se está trabajando actualmente.

## Desarrollo del tema:

### 1. Creación y Navegación de Ramas:

- **Listar ramas:** \$ git branch (muestra ramas locales, con \* indicando la rama actual). \$ git branch -a (muestra locales y remotas).
- **Crear una rama:** \$ git branch <nombre-de-la-rama>.
- **Cambiar a una rama existente (Navegación):** \$ git switch <nombre-de-la-rama> (método moderno) o \$ git checkout <nombre-de-la-rama>.
- **Crear y cambiar inmediatamente:** \$ git switch -c <nombre-de-la-rama> o \$ git checkout -b <nombre-de-la-rama>.

### 2. Fusión de Ramas (git merge):

Unir dos ramas significa incorporar los cambios de una (la rama fuente) en la otra (la rama destino, donde se ejecuta el comando).

- **Paso a Paso de Fusión:**
  - **1. Cambiar a la rama destino:** \$ git switch main (o master).
  - **2. Ejecutar la fusión:** \$ git merge <rama-a-fusionar>.
- **Tipos de Fusión:**
  - **Fast-Forward:** Ocurre si la rama destino no ha tenido commits desde que se creó la rama fuente. Git simplemente mueve el puntero HEAD hacia adelante.
  - **Merge Commit:** Si la rama destino ha evolucionado en paralelo, Git crea un commit especial de fusión que combina las historias. Usar \$ git merge --no-ff <rama> fuerza la creación de un merge commit incluso si fuera posible Fast-Forward, lo que ayuda a documentar la historia.



Servicio Nacional de Aprendizaje – SENA

Centro de Tecnología de la Manufactura Avanzada – Regional Antioquia

- **Resolución de Conflictos:** Los conflictos ocurren cuando la misma línea o secciones de código han sido modificadas de forma diferente en ambas ramas a fusionar.
- **Proceso:** Git detiene la fusión y marca el archivo afectado como Unmerged. El desarrollador debe abrir el archivo y resolver manualmente los bloques de conflicto, identificados por marcadores como <<<<<< HEAD, ===== y >>>>>> <rama>.
- **Finalizar la Fusión:** Una vez editado el archivo para conservar el código correcto, se debe marcar como resuelto: \$ git add <archivo-resuelto>. Finalmente, se crea el commit de fusión: \$ git commit. 4. Eliminación de Ramas: Una vez que la rama está fusionada y ya no se necesita, se puede eliminar localmente (solo si ha sido fusionada): \$ git branch -d <nombre-de-la-rama>.

## Ejercicios prácticos:

### Ejercicio 5: Flujo Completo de Ramificación y Conflicto (Asuma que está en la rama main con un index.html inicial):

1. Cree y cambie a una nueva rama llamada feature/cabecera: \$ git switch -c feature/cabecera.
2. En feature/cabecera, añada una línea de código al inicio de index.html (ej. <header>Nueva Cabecera</header>). Confirme el cambio.
3. Vuelva a la rama main: \$ git switch main.
4. En main, añada una línea de código en el mismo lugar de index.html (ej. <h1>Mi Proyecto Principal</h1>). Confirme el cambio.
5. Intente fusionar la rama feature/cabecera en main: \$ git merge feature/cabecera.
6. Resuelva el conflicto manualmente en el editor de código. Decida qué versión de la cabecera quiere mantener.
7. Agregue el archivo resuelto al Staging Area: \$ git add index.html.
8. Finalice la fusión: \$ git commit.
9. Elimine la rama local feature/cabecera: \$ git branch -d feature/cabecera.

## Cuestionario de comprensión:

1. ¿Qué es el puntero HEAD y qué indica?



Servicio Nacional de Aprendizaje – SENA

Centro de Tecnología de la Manufactura Avanzada – Regional Antioquia

2. ¿Qué significa que una fusión sea Fast-Forward?
3. ¿Cómo se marca un archivo con conflicto de fusión como resuelto, antes de completar el merge commit?
4. ¿Qué comando se utiliza para crear una rama y cambiar a ella en un solo paso?

## **Materiales necesarios**

- Computador
- <https://www.youtube.com/watch?v=Ei1y51K8jQk>
- <https://www.youtube.com/watch?v=HiVnGgYudLY>

## **Evidencias de aprendizaje**

- Mandar un PDF con la guía resuelta sin errores

## **Resultados de aprendizaje**

Adquirir conocimientos acerca de... tema (los que se evalúa en Sofia)