



Servicio Nacional de Aprendizaje – SENA

Centro de Tecnología de la Manufactura Avanzada – Regional Antioquia

Guía de aprendizaje 2 : El Repositorio Local y el Ciclo Básico de Trabajo

Flujo de Trabajo Local: Inicialización, Estados de los Archivos y Creación de Commits.

Tema que se va a ver en la guía de estudio

Inicialización del repositorio (git init), los tres estados de los archivos (Modificado, Preparado, Confirmado), Área de Staging y comandos fundamentales (git status, git add, git commit).

Objetivo general

- Dominar el ciclo de vida de un archivo dentro de un repositorio Git local, sabiendo iniciar un proyecto y registrar los cambios de forma atómica mediante commits.

Objetivos específicos

- Inicializar un repositorio local en una carpeta existente.
- Comprender los cuatro estados posibles de los archivos (Tracked, Staged, Unstaged, Untracked).
- Utilizar el Staging Area para seleccionar qué cambios se incluirán en el próximo commit.
- Aplicar los comandos git add, git commit y git status para mover los archivos a través de los estados.



Servicio Nacional de Aprendizaje – SENA

Centro de Tecnología de la Manufactura Avanzada – Regional Antioquia

Introducción

El trabajo local con Git se basa en la gestión de tres "árboles" o estados principales por donde transitan los archivos: el Directorio de Trabajo (donde se realizan las modificaciones), el Área de Preparación o Staging Area (un área transitoria que enmarca la "fotografía" de los cambios), y el Repositorio Local (donde se almacenan permanentemente los commits o "fotografías"). Los commits son la forma de registrar estos cambios, actuando como instantáneas de todo el repositorio en un momento dado.

Desarrollo del tema:

1. Inicialización de un Repositorio Local (git init):

Para que una carpeta se convierta en un repositorio Git, se navega hasta la carpeta raíz del proyecto en la terminal y se ejecuta: \$ git init. Este comando crea la carpeta oculta .git, que contiene toda la base de datos de cambios atómicos del proyecto y el Staging Area.

Paso a Paso:

- 1. Cree la carpeta de su proyecto (ej: C:\estructura-datos).
- 2. Navegue a la carpeta en su terminal.
- 3. Ejecute \$ git init.

2. Estados de los Archivos en Git:

Los archivos en un proyecto Git viven y se mueven entre estados:

- **Modificado (Modified):** El archivo contiene cambios, pero aún no han sido seleccionados (staged) para el próximo commit.
- **Preparado (Staged):** El archivo modificado ha sido marcado y está listo para ser incluido en el próximo commit.
- **Confirmado (Committed):** El archivo se encuentra grabado en el repositorio local (dentro de la base de datos .git).
- **Rastreado (Tracked):** Archivos que ya viven dentro de Git. Si tienen cambios pendientes, son Unstaged.
- **No Rastreado (Untracked):** Archivos nuevos que solo existen en el disco duro y Git no tiene registro de ellos.



Servicio Nacional de Aprendizaje – SENA

Centro de Tecnología de la Manufactura Avanzada – Regional Antioquia

3. El Flujo Básico de Commit:

- **Paso 1:** Verificación de Estado (git status): Este comando esencial muestra el estado actual de todos los archivos y carpetas, indicando si están Untracked, Modified (pero Unstaged), o Staged (listos para confirmar).
- **Paso 2:** Preparar los Cambios (git add): Mueve archivos del estado Modified o Untracked al Staging Area (estado Staged). Solo los archivos en el Staging Area serán incluidos en la siguiente instantánea. * Añadir archivos individuales: \$ git add <nombre-del-archivo>. * Añadir todos los archivos modificados o nuevos: \$ git add . o \$ git add -A.
- **Paso 3:** Confirmar los Cambios (git commit): Toma la instantánea de los archivos que se encuentran en el Staging Area y la almacena permanentemente como un commit en la base de datos local. Git requiere un mensaje para recordar los cambios realizados. * Commit con mensaje en línea: \$ git commit -m "Mensaje del commit". Este mensaje debe ser conciso y descriptivo.

4. Atajo Rápido (Saltar Staging):

Para archivos que ya están siendo rastreados (Tracked) y que solo han sido modificados, se puede saltar el paso de git add usando el parámetro -a: \$ git commit -a -m "Mensaje"

Ejercicios prácticos:

Ejercicio 2: Ciclo Básico de Commit (Asumiendo que ya ejecutó git init en una carpeta llamada estructura-datos):

1. Cree dos archivos nuevos: Programa_01.cpp y notas.txt (simule un archivo Untracked).
2. Verifique su estado: \$ git status.
3. Agregue solo Programa_01.cpp al Staging Area: \$ git add Programa_01.cpp.
4. Verifique el estado nuevamente. Note que Programa_01.cpp está Staged y notas.txt sigue Untracked.
5. Confirme el cambio de Programa_01.cpp: \$ git commit -m "feat: Primer ejercicio resuelto".
6. Modifique el archivo Programa_01.cpp (añada una línea de código) y modifique notas.txt (añada una línea).
7. Ejecute git add . para añadir ambos al Staging Area.



Centro de Tecnología de la Manufactura Avanzada – Regional Antioquia

8. Confirme ambos cambios: \$ git commit -m "fix: Corregir lógica de Programa_01 y añadir notas"

Cuestionario de comprensión:

1. Defina qué es el Staging Area y cuál es su propósito en el ciclo de trabajo de Git.
2. Si un archivo ha sido modificado en el directorio de trabajo, pero nunca se ha ejecutado git add sobre él, ¿en qué estado se encuentra?
3. ¿Qué ocurre internamente en Git cuando se ejecuta git commit?
4. ¿Qué comando simplificado puede usar para mover archivos Tracked y Modified directamente al commit, saltando el git add?

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)