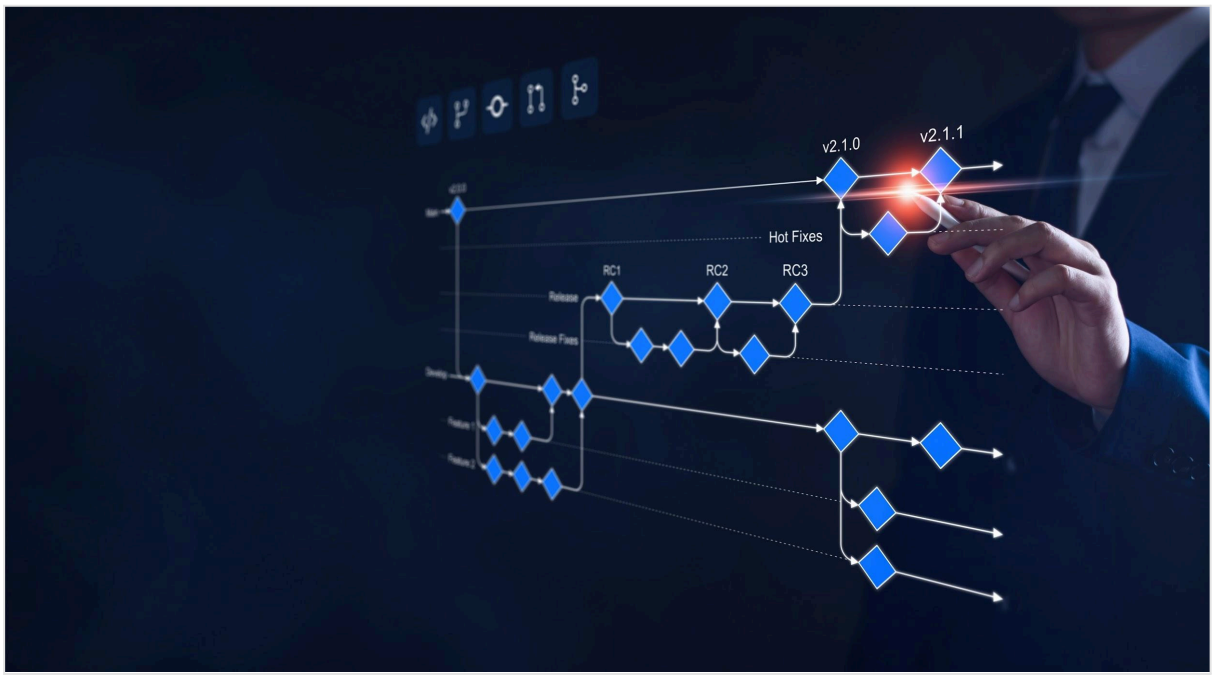
Guía de Ramas: Flujos de Trabajo en Git y GitHub

1. El Porqué de las Ramas: Trabajo en Paralelo

En los ejercicios anteriores, se trabajó en ramas, pero se fusionaron localmente (`git merge`). En un equipo, el flujo es diferente.

El propósito principal de las ramas es **aislar el trabajo**. Permiten que múltiples desarrolladores (o una persona trabajando en múltiples ideas) puedan construir características de forma independiente sin interferir entre sí.

- La rama `main` se considera la versión "estable" o "de producción".
- Las ramas `feature/` (ej. `feature/anadir-skills`) son "talleres" temporales donde se construye y prueba una nueva característica.
- Cuando la característica está lista, no se fusiona localmente. En su lugar, se sube a GitHub y se propone su integración mediante un **Pull Request (PR)**.
- 

---

2. Ejercicio de Ramas con Terminal (CLI) y GitHub

Este ejercicio simulará un entorno de equipo donde se desarrollan dos características separadas al mismo tiempo para el proyecto `mi-cv-online`.

Escenario: Se añadirán dos secciones nuevas a la hoja de vida: "Habilidades" y "Contacto".

Paso 1: Sincronizar la Rama `main`

Antes de crear cualquier rama nueva, es vital asegurarse de que la rama `main` local esté idéntica a la de GitHub.

1. Asegurarse de estar en la rama `main`.

git switch main

2. "Jalar" (descargar) cualquier cambio que pueda estar en GitHub (como fusiones anteriores).

git pull origin main

Paso 2: Característica A (Habilidades)

1. Crear y cambiarse a una nueva rama para la primera característica.

git switch -c feature/skills

2. **Hacer el trabajo**: En `index.html`, añadir una nueva sección `<section>` después de "Educación" que contenga una lista `<ul>` de habilidades (ej. HTML, CSS, Git, Liderazgo).
3. **Guardar el trabajo en la rama local**:

git add .
git commit -m "Feat: Agrega la sección de habilidades"

4. **Subir la rama a GitHub**: Este es un comando clave. La primera vez que se sube una rama nueva, se debe usar `-u` para establecer un "seguimiento". Git sabrá que la rama local `feature/skills` está conectada a la rama `feature/skills` en `origin` (GitHub).

git push -u origin feature/skills

Paso 3: Característica B (Contacto)

La rama `feature/skills` está ahora en GitHub, pero el trabajo no ha terminado. Se simulará que *otro desarrollador* (o uno mismo) empieza una tarea *diferente*.

1. **Regresar a `main`**: Es crucial empezar el nuevo trabajo desde la base limpia, no desde la rama de `skills`.

git switch main

2. (Se notará que el archivo `index.html` *no* tiene la sección de habilidades, ¡y eso es correcto! Está aislada en su propia rama).
3. Crear la segunda rama.

git switch -c feature/contact

4. **Hacer el trabajo**: En `index.html`, añadir una sección `<footer>` al final del `<body>`. Dentro, colocar un email y un enlace a LinkedIn.
5. **Guardar el trabajo en la rama local**:

git add .
git commit -m "Feat: Agrega footer con información de contacto"

6. **Subir la segunda rama a GitHub**:

git push -u origin feature/contact

Paso 4: El Flujo de GitHub: Pull Requests (PRs)

Ahora, si se visita el repositorio en GitHub, se verán notificaciones amarillas indicando que se han subido nuevas ramas y ofreciendo crear un **"Pull Request"**.

Un Pull Request es una **solicitud para fusionar (pull)** los cambios de una rama en otra (ej. de `feature/skills` hacia `main`). Es el momento de la revisión de código.

1. En GitHub, hacer clic en el botón "Compare & pull request" para la rama `feature/skills`.
2. Poner un título y descripción. Hacer clic en "Create pull request".
3. En un proyecto real, alguien más revisaría el código. En este ejercicio, se aprueba directamente.
4. Hacer clic en el botón verde "Merge pull request" y luego "Confirm merge".
5. **Resultado**: La rama `feature/skills` se ha fusionado exitosamente en `main` *dentro de GitHub*.
6. **Repetir el proceso**: Ir de nuevo a la pestaña "Pull requests" y crear un nuevo PR para la rama `feature/contact`. Fusionarlo también.

Paso 5: Limpieza y Sincronización Final

El repositorio de GitHub ahora tiene la rama `main` actualizada con ambas características. Sin embargo, la rama `main` *local* (en la computadora) todavía está desactualizada.

1. Regresar a la rama `main` local.

git switch main

2. "Jalar" (descargar) los cambios que se fusionaron en GitHub.

git pull origin main

3. ¡Listo! El `index.html` local ahora tiene ambas secciones: "Habilidades" y "Contacto". El ciclo se ha completado.

---

3. Herramienta: GitHub Desktop

¿Qué es? Es una aplicación gratuita de GitHub que pone una interfaz gráfica (GUI) sobre la complejidad de los comandos de Git. Permite hacer el 90% de las tareas diarias (clonar, crear ramas, hacer commits, push/pull) con clics en lugar de comandos.

¿Por qué usarla? Es visual. Permite ver claramente qué archivos se han modificado (la "diferencia" o *diff*), qué ramas existen y cuál es el historial. Reduce el miedo a cometer errores en la terminal.

---

4. Ejercicio con GitHub Desktop (GUI)

Se realizará el mismo flujo de trabajo, pero para una tercera característica: "Pasatiempos".

Paso 1: Clonar el Repositorio

1. Abrir GitHub Desktop.
2. Ir a `File > Clone Repository...`.
3. Buscar y seleccionar el repositorio `mi-cv-online` de la lista y hacer clic en "Clone".

Paso 2: Crear la Rama

1. La aplicación mostrará "Current repository: `mi-cv-online`" y "Current branch: `main`".
2. Hacer clic en el botón "Current branch" en la barra superior.
3. Hacer clic en el botón "New Branch".
4. Nombrar la nueva rama: `feature/hobbies`.
5. Hacer clic en "Create Branch". (Esto es lo mismo que `git switch -c feature/hobbies`).

Paso 3: Hacer los Cambios

1. Ir a VS Code (GitHub Desktop no es un editor de código).
2. Abrir `index.html` y añadir una nueva `<section>` para "Pasatiempos".
3. Guardar el archivo.

Paso 4: Hacer Commit y Push

1. Volver a GitHub Desktop.
2. En la pestaña "Changes" (Cambios) a la izquierda, se verá el archivo `index.html` modificado. La ventana principal mostrará en verde el texto añadido.
3. En la parte inferior izquierda, hay un cuadro de "Summary" (Resumen). Escribir el mensaje del commit: "Feat: Agrega sección de pasatiempos".
4. Hacer clic en el botón azul "Commit to feature/hobbies". (Esto es `git add .` y `git commit ...`).
5. Después del commit, la barra superior cambia. Ahora muestra un botón que dice "Publish branch".
6. Hacer clic en "Publish branch". (Esto es `git push -u origin feature/hobbies`).

Paso 5: Pull Request y Sincronización (El Flujo Visual)

1. Una vez publicada la rama, GitHub Desktop mostrará un nuevo botón: "Create Pull Request".
2. Al hacer clic, se abrirá el navegador de internet en la página de creación del Pull Request en GitHub.
3. Se repite el proceso: "Create pull request" -> "Merge pull request".
4. **Sincronización final**: Volver a GitHub Desktop.
    - Hacer clic en "Current Branch" y seleccionar `main`.
    - El botón superior ahora dirá "Pull origin".
    - Hacer clic en "Pull origin". (Esto es `git switch main` y `git pull origin main`).