

Práctico 2: Git y GitHub

1) ¿Qué es GitHub?

GitHub es una plataforma de desarrollo colaborativo que funciona como un servicio basado en la nube. Permite a los desarrolladores almacenar y administrar su código, así como realizar un seguimiento y controlar los cambios en el código. Esencialmente, es una herramienta que facilita la colaboración entre programadores y desarrolladores de software

¿Cómo crear un repositorio en GitHub?

Crear un repositorio en GitHub es un proceso sencillo. Aquí tienes una guía paso a paso:

1. Inicia sesión en GitHub:

- Ve a [GitHub.com](https://github.com) y accede a tu cuenta. Si no tienes una, deberás registrarte.

2. Crea un nuevo repositorio:

- Una vez que hayas iniciado sesión, busca el botón "+" en la esquina superior derecha de la página. Haz clic en él y selecciona "New repository".
- También puedes acceder a la página de creación de repositorios directamente a través de este enlace: <https://github.com/new>

3. Configura tu repositorio:

- **Repository name (Nombre del repositorio):** Escribe un nombre para tu repositorio. Debe ser corto y descriptivo.
- **Description (Descripción) (opcional):** Agrega una descripción breve de tu proyecto.
- **Public or Private (Público o Privado):**
 - **Public:** Cualquiera puede ver tu repositorio.
 - **Private:** Solo las personas a las que invites podrán ver tu repositorio.
- **Initialize this repository with:**
 - **Add a README file (Agregar un archivo README):** Un archivo README es una buena práctica. Suele contener información sobre tu proyecto.

- **Add .gitignore (Agregar .gitignore):** Si estás trabajando en un proyecto de programación, puedes seleccionar un archivo .gitignore para evitar que se suban archivos innecesarios.
- **Choose a license (Elegir una licencia):** Selecciona una licencia para tu proyecto. Esto indica cómo otros pueden usar tu código.

4- Crea el repositorio:

- Haz clic en el botón "Create repository" (Crear repositorio).

¿Cómo crear una rama en Git?

1. Abre la terminal y asegúrate de estar en el directorio del proyecto.
2. Lista las ramas existentes con `git branch` para verificar el estado actual.
3. Crea una nueva rama usando `git branch nombre-de-la-rama`.
4. Cambia a la nueva rama con `git checkout nombre-de-la-rama`.
5. Confirma en qué rama te encuentras con `git branch`.

¿Cómo cambiar a una rama en Git?

1. Abre la terminal o consola y asegúrate de estar en el directorio del proyecto.
2. Utiliza el comando `git checkout nombre-de-la-rama`, reemplazando *nombre-de-la-rama* por el nombre de la rama a la que deseas cambiar.
3. Confirma que estás en la nueva rama con el comando `git branch`.

¿Cómo fusionar ramas en Git?

1. Asegúrate de estar en la rama donde deseas fusionar los cambios (la rama principal o destino).
2. Ejecuta el comando `git merge nombre-de-la-rama`, donde *nombre-de-la-rama* es la rama que quieres fusionar con la rama actual.
3. Si hay conflictos, resuélvelos manualmente en los archivos afectados y luego guarda los cambios.
4. Una vez resueltos los conflictos, utiliza el comando `git add` para incluir los archivos modificados y `git commit` para finalizar la fusión.

¿Cómo crear un commit en Git?

Crear un commit en Git es sencillo y se hace en pocos pasos:

1. Asegúrate de que los archivos que deseas incluir en el commit estén agregados al área de preparación (staging area) usando `git add nombre-del-archivo` o `git add .` para agregar todos los archivos.
2. Una vez que los archivos estén listos, crea el commit con el comando:
3. `git commit -m "`

¿Cómo enviar un commit a GitHub?

1. **Asegúrate de estar en tu repositorio local:** Navega al directorio donde está tu proyecto.
2. **Haz un commit:** Si no lo has hecho aún, crea un commit usando `git commit -m "Tu mensaje de commit"`.
3. **Conecta tu repositorio local con GitHub:** Si no lo has hecho antes, vincula tu repositorio local con un repositorio en GitHub usando el comando:
4. `git remote add origin` Reemplaza *tu-usuario* y *tu-repositorio* con tus datos reales.
5. **Envía tu commit al repositorio en GitHub:**
6. `git push origin nombre-de-la-rama`

Sustituye *nombre-de-la-rama* por la rama en la que estás trabajando, como `main`.

¿Qué es un repositorio remoto?

Un repositorio remoto en Git es una versión de tu proyecto almacenada en un servidor o servicio en línea, como GitHub, GitLab o Bitbucket. Sirve como un repositorio centralizado al que varios colaboradores pueden acceder para clonar, enviar (push) o obtener (pull) los cambios realizados en el proyecto.

En pocas palabras, un repositorio remoto es la copia de tu proyecto que está alojada en línea, permitiendo sincronizar y colaborar con otros de manera eficiente

¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto a Git, sigue estos pasos:

1. **Crea o identifica el repositorio remoto:** Puede estar en servicios como GitHub, GitLab o Bitbucket. Copia la URL del repositorio.

2. **Abre la terminal y ve al proyecto local:** Asegúrate de estar en el directorio del repositorio local.

3. **Conecta el repositorio remoto:** Usa el comando

4. `git remote add origin URL-del-repositorio`

Sustituye *URL-del-repositorio* con la URL copiada del repositorio remoto.

5. **Verifica la conexión:** Comprueba que el repositorio remoto se haya agregado correctamente con

6. `git remote -v`

¿Cómo empujar cambios a un repositorio remoto?

1. **Asegúrate de estar en la rama correcta:** Usa `git branch` para verificar la rama en la que estás trabajando.

2. **Haz un commit de tus cambios:** Si no lo has hecho aún, utiliza `git add` para preparar los archivos y `git commit -m "Mensaje del commit"` para registrar los cambios.

3. **Empuja los cambios al repositorio remoto:** Usa el comando

4. `git push origin nombre-de-la-rama`

Reemplaza *nombre-de-la-rama* con la rama que deseas actualizar en el repositorio remoto.

¿Cómo tirar de cambios de un repositorio remoto?

1. **Asegúrate de estar en la rama donde quieres fusionar los cambios:** Usa `git branch` para confirmar la rama activa.

2. **Obtén los cambios del repositorio remoto:** Ejecuta el comando:

3. `git pull origin nombre-de-la-rama`

Sustituye *nombre-de-la-rama* por la rama específica de la que deseas extraer los cambios (por ejemplo, `main`).

4. **Resolver conflictos si es necesario:** Si hay conflictos durante el proceso, edítalos manualmente, guarda los cambios, y usa `git add` seguido de `git commit` para solucionarlos.

Este proceso sincroniza la rama local con la rama remota y asegura que tengas los cambios más recientes.

¿Qué es un fork de repositorio?

Un fork de un repositorio es una copia completa de un repositorio existente que se crea en tu propia cuenta, generalmente en una plataforma como GitHub. Este proceso te permite trabajar en un proyecto sin afectar el repositorio original. Los forks son útiles cuando quieres:

- Contribuir a proyectos de código abierto.
- Probar cambios o experimentos en un proyecto sin arriesgarte a alterar la fuente original.
- Crear una versión personalizada de un proyecto para tu uso personal.

Después de hacer un fork, puedes clonar tu copia a tu máquina local, realizar cambios y enviar una solicitud de incorporación (pull request) para que esos cambios se consideren en el repositorio original.

¿Cómo crear un fork de un repositorio?

1. Accede al repositorio original en GitHub (o en la plataforma correspondiente).
2. Busca el botón que dice **Fork** en la parte superior derecha de la página del repositorio y haz clic en él.
3. GitHub creará una copia del repositorio original en tu cuenta personal. Ahora tienes tu propio fork, donde puedes realizar cambios sin afectar el repositorio original.

Desde aquí, puedes clonar el fork a tu máquina local y empezar a trabajar en él. Luego puedes enviar tus cambios al fork o proponer cambios en el repositorio original con un pull request.

¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

1. **Haz un fork y clona el repositorio:** Asegúrate de tener el repositorio en tu cuenta y trabajar en tu copia local.
2. **Crea una nueva rama:** Trabaja en una rama específica para los cambios que deseas proponer, asegurándote de mantener la rama principal limpia.
3. **Realiza y registra tus cambios:** Modifica los archivos necesarios, haz commits claros y descriptivos sobre los cambios realizados.
4. **Empuja los cambios a tu repositorio en GitHub:** Usa el comando `git push origin nombre-de-la-rama` para subir los cambios.

5. **Abre la solicitud de extracción:** Ve al repositorio original en GitHub, busca la opción para crear un pull request desde tu rama en el fork, agrega un título y una descripción explicando tus cambios, y envíalo.

¿Cómo aceptar una solicitud de extracción?

1. **Revisa los cambios propuestos:** Ve al repositorio en GitHub y abre la solicitud de extracción. Examina los archivos modificados y verifica que los cambios sean correctos y no introduzcan problemas.
2. **Discute los cambios (opcional):** Si es necesario, utiliza la sección de comentarios para hacer preguntas, solicitar ajustes o aclarar detalles con el autor de la solicitud.
3. **Realiza pruebas (opcional):** Descarga la rama asociada a la solicitud y prueba los cambios localmente para asegurarte de que funcionen según lo esperado.
4. **Acepta la solicitud:** Una vez que estés satisfecho, haz clic en el botón **Merge pull request** (Fusionar solicitud de extracción). Esto integrará los cambios en la rama principal o la rama de destino seleccionada.
5. **Elimina la rama (opcional):** Después de fusionar, puedes eliminar la rama asociada a la solicitud para mantener el repositorio limpio.

¿Qué es un etiqueta en Git?

Una etiqueta en Git es una referencia que marca un punto específico en la historia del repositorio. Generalmente se usa para señalar versiones importantes, como el lanzamiento de una nueva versión de software. Las etiquetas son útiles para identificar un commit específico de manera fácil y legible.

Existen dos tipos principales de etiquetas en Git:

1. **Etiquetas ligeras:** Son como un marcador simple, sin información adicional, que apunta a un commit.
2. **Etiquetas anotadas:** Incluyen metadatos, como el nombre del autor, la fecha y un mensaje descriptivo. Estas son más recomendadas para lanzamientos.

¿Cómo crear una etiqueta en Git?

Para crear una etiqueta en Git, sigue estos pasos:

1. **Selecciona el commit donde deseas crear la etiqueta:** Asegúrate de estar en el commit deseado o verifica su hash si es específico.
2. **Crea la etiqueta:**
 - Para una **etiqueta ligera**: Usa el comando `git tag nombre-de-la-etiqueta`.
 - Para una **etiqueta anotada** (recomendado): Usa el comando `git tag -a nombre-de-la-etiqueta -m "Mensaje descriptivo"`.
3. **Verifica las etiquetas creadas:** Utiliza el comando `git tag` para listar todas las etiquetas existentes.
4. **Sube la etiqueta al repositorio remoto** (si es necesario): Usa `git push origin nombre-de-la-etiqueta` o `git push --tags` para subir todas las etiquetas.

¿Cómo enviar una etiqueta a GitHub?

Para enviar una etiqueta a GitHub, sigue estos pasos:

1. **Crea la etiqueta:** Asegúrate de haber creado la etiqueta en tu repositorio local (puedes usar `git tag` para verificarlo).
2. **Sube una etiqueta específica:** Usa el comando:
3. `git push origin nombre-de-la-etiqueta`

Sustituye *nombre-de-la-etiqueta* por el nombre de la etiqueta que deseas enviar.

4. **Sube todas las etiquetas:** Si quieres enviar todas las etiquetas creadas al repositorio remoto, utiliza:
5. `git push --tags`

Esto sincronizará tus etiquetas locales con el repositorio remoto en GitHub.

¿Hay algo más que quieras aclarar sobre las etiquetas o Git? 🙋

¿Qué es un historial de Git?

El historial de Git es el registro de todos los commits realizados en un repositorio. Este historial muestra los cambios realizados, quién los hizo y cuándo, lo que facilita rastrear el progreso del proyecto y entender cómo ha evolucionado.

Puedes ver el historial de commits utilizando el comando `git log`. Esto te proporcionará una lista de commits con detalles como el identificador único (hash), el autor, la fecha y el mensaje del commit.

¿Cómo ver el historial de Git?

1. Usa el comando `git log` para ver una lista detallada de los commits, incluyendo el identificador único (hash), autor, fecha y mensaje.
2. Si prefieres un historial compacto, prueba `git log --oneline`, que muestra los commits de forma resumida en una sola línea cada uno.
3. Para explorar cambios en los archivos, utiliza `git log -p`, que incluye las diferencias (diffs) entre versiones en el historial.

¿Cómo buscar en el historial de Git?

1. **Buscar por mensaje de commit:** Utiliza el comando:
`git log --grep="palabra clave"`
Sustituye "*palabra clave*" por el texto que deseas buscar en los mensajes de los commits.
3. **Buscar por autor:** Si quieres encontrar commits realizados por un autor en particular, usa:
`git log --author="nombre del autor"`
5. **Buscar por fecha:** Puedes filtrar los commits por un rango de fechas con:
`git log --since="fecha" --until="fecha"`
Por ejemplo, `--since="2025-03-01" --until="2025-03-30"` para buscar entre el 1 y el 30 de marzo de 2025.
7. **Buscar por archivo específico:** Usa este comando para ver el historial de cambios de un archivo:
`git log -- nombre-del-archivo`
9. **Combinar filtros:** Puedes combinar varios parámetros, como autor y palabra clave, para refinar la búsqueda.

¿Cómo borrar el historial de Git?

En Git, el historial de commits es una parte esencial del repositorio y no se puede "borrar" de forma estándar sin manipular el historial del proyecto. Sin embargo, si necesitas modificar o eliminar el historial por razones específicas, aquí hay opciones avanzadas a considerar:

1. **Borrar el historial completo y empezar de nuevo:**
 - Borra todo en el repositorio remoto.

- Usa el comando `git checkout --orphan nueva-rama` para crear una rama vacía.
- Elimina los archivos anteriores con `git rm -rf ..`
- Crea un commit inicial y luego fuerza el envío con `git push origin --force`.

2. Reescribir el historial existente (uso avanzado):

- Usa el comando `git rebase` o `git filter-branch` para reescribir el historial de commits. Esto puede alterar commits específicos, pero requiere cuidado, ya que puede causar problemas en repositorios compartidos.

3. Eliminar commits recientes en una rama:

- Usa `git reset --hard commit-id` para restablecer la rama a un commit anterior. Esto elimina los commits posteriores de forma local.

¿Qué es un repositorio privado en GitHub?

Un repositorio privado en GitHub es un proyecto alojado en la plataforma cuyo contenido no está accesible públicamente. Solo las personas que han sido invitadas explícitamente al repositorio tienen permiso para ver, colaborar o modificar su contenido. Estos repositorios son ideales para:

- **Proyectos personales o confidenciales:** Donde no deseas que el código sea visible para todo el mundo.
- **Desarrollo corporativo:** Para mantener el control sobre proyectos internos o propiedad intelectual.

¿Cómo crear un repositorio privado en GitHub?

Crear un repositorio privado en GitHub es muy sencillo. Sigue estos pasos:

1. **Accede a tu cuenta de GitHub:** Inicia sesión en [GitHub](#).
2. **Haz clic en "New Repository" (Nuevo Repositorio):** Esta opción está disponible en la parte superior derecha, generalmente como un botón verde o dentro del menú del signo "+".
3. **Configura el repositorio:**
 - Ingresa un nombre para tu repositorio.
 - Agrega una descripción opcional si lo deseas.

- Selecciona la opción "Private" (Privado) para asegurarte de que solo las personas invitadas puedan acceder.
4. **Inicializa el repositorio (opcional):** Puedes agregar un archivo README, un archivo .gitignore, o seleccionar una licencia si lo necesitas.
 5. **Crea el repositorio:** Haz clic en el botón **"Create repository"** (Crear repositorio).

¡Listo! Tu repositorio privado estará creado y protegido. Puedes gestionar quién tiene acceso desde la pestaña "Settings" (Configuraciones) del repositorio.

¿Cómo invitar a alguien a un repositorio privado en GitHub?

1. **Ve al repositorio privado:** Abre el repositorio en el que deseas agregar colaboradores.
2. **Accede a las configuraciones:** Haz clic en la pestaña **Settings** (Configuraciones) en la parte superior del repositorio.
3. **Busca la sección "Collaborators and teams":** En el menú lateral, selecciona esta opción dentro de la categoría "Access".
4. **Invita a colaboradores:**
 - Haz clic en el botón **Invite a collaborator** (Invitar un colaborador).
 - Escribe el nombre de usuario o correo electrónico de la persona que deseas invitar.
 - Selecciona al usuario y confirma la invitación.
5. **El usuario debe aceptar la invitación:** Una vez enviada, el colaborador recibirá una notificación y deberá aceptar la invitación para acceder al repositorio.

¿Qué es un repositorio público en GitHub?

Un repositorio público en GitHub es un proyecto alojado en la plataforma cuyo contenido es accesible para cualquier persona. Esto significa que cualquiera puede ver el código, clonar el repositorio y, en algunos casos, proponer cambios mediante solicitudes de extracción (pull requests). Sin embargo, solo los propietarios o colaboradores con permisos tienen control directo para modificar el repositorio.

Este tipo de repositorio es ideal para:

- Proyectos de código abierto.
- Compartir conocimiento o herramientas con la comunidad.

- Colaborar con desarrolladores de todo el mundo.

¿Cómo crear un repositorio público en GitHub?

Crear un repositorio público en GitHub es muy sencillo y directo. Sigue estos pasos:

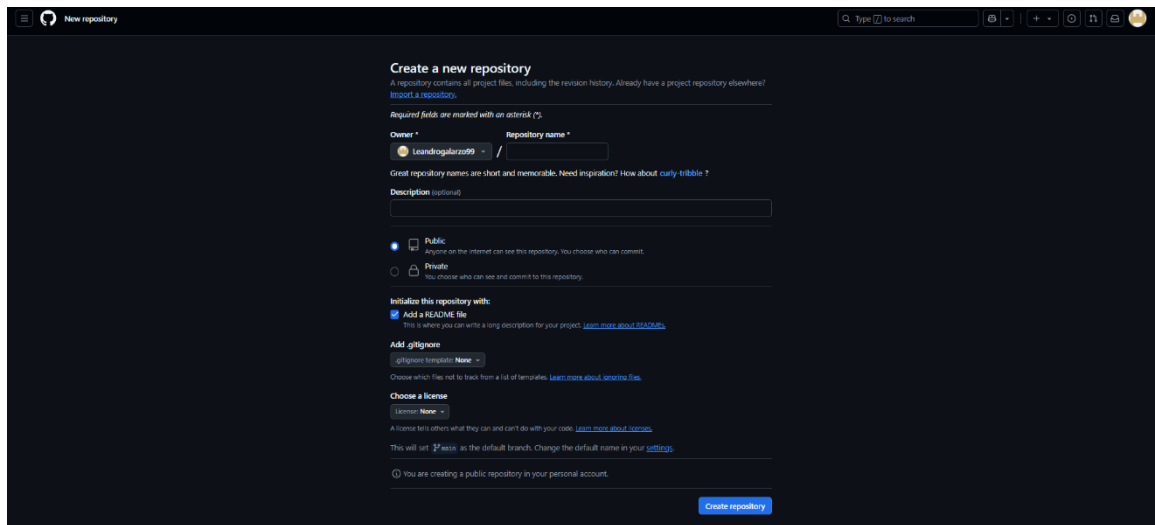
1. **Accede a tu cuenta en GitHub:** Inicia sesión en [GitHub](#).
2. **Dirígete al botón "New Repository" (Nuevo Repositorio):** Lo encontrarás en la esquina superior derecha, generalmente como un signo "+" o un botón verde.
3. **Rellena los detalles del repositorio:**
 - Introduce el nombre de tu proyecto.
 - Agrega una descripción breve si lo deseas.
 - Selecciona la opción **"Public" (Público)** para que el repositorio sea accesible a todos.
4. **Configura opciones adicionales (opcional):** Puedes añadir un archivo README, configurar un .gitignore o seleccionar una licencia para tu proyecto.
5. **Crea el repositorio:** Haz clic en el botón **"Create repository"** para finalizar la configuración.

¿Cómo compartir un repositorio público en GitHub?

Compartir un repositorio público en GitHub es sencillo y eficaz para que otros puedan acceder o colaborar en tu proyecto. Sigue estos pasos:

1. **Localiza la URL del repositorio:** Ve a la página de tu repositorio en GitHub y copia la URL desde la barra de direcciones del navegador o haz clic en el botón **Code** y copia el enlace HTTPS, SSH o GitHub CLI.
2. **Comparte el enlace:** Envía la URL del repositorio a las personas o comunidades con quienes quieras compartirlo. Esto puede ser mediante correo electrónico, redes sociales, foros, etc.
3. **Promociona tu proyecto (opcional):** Si es un proyecto público de interés general, puedes incluir descripciones claras y etiquetas relevantes en la página del repositorio para atraer colaboradores.

2) Crear un repositorio.



The screenshot shows the GitHub 'Create a new repository' page. At the top, there's a header with the GitHub logo and 'New repository'. Below this, the main heading is 'Create a new repository'. A subtext says 'A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)'. Below this, there's a section 'Required fields are marked with an asterisk (*)'. It has two input fields: 'Owner' with the value 'Leandrogalarzo99' and 'Repository name' with a slash '/' in the text box. Below these is a 'Description (optional)' text area. Then, there are two radio buttons for visibility: 'Public' (selected) and 'Private'. Below this is a section 'Initialize this repository with:'. It has three options: 'Add a README file' (checked), 'Add .gitignore' (with a dropdown menu showing 'None'), and 'Choose a license' (with a dropdown menu showing 'None'). At the bottom, there's a blue button labeled 'Create repository'.

Agregando un Archivo

<https://github.com/Leandrogalarzo99/programacion-1-.git>

3) Realizar la siguiente actividad:

<https://github.com/Leandrogalarzo99/conflict-exercise.git>