


UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

DIVISIÓN SISTEMA UNIVERSIDAD ABIERTA Y EDUCACIÓN A DISTANCIA

PROGRAMACION DE DISPOSITIV.MOVILES

UNIDAD 2

ACTIVIDAD TEORICA

NOMBRE DEL ALUMNO:

PAULINA RODRIGUEZ SAMPEDRO

NOMBRE DEL ASESOR:

CRISTIAN CARDOSO ARELLANO

SEXTO SEMESTRE

GRUPO: 9696

CONTROL Y VERSIONAMIENTO CON GIT, GITHUB Y GITFLOW

Objetivo

Conocer y comprender los fundamentos de Git y GitHub, desde la configuración inicial del entorno y el control de versiones local, hasta la gestión colaborativa de repositorios remotos, incluyendo el uso de ramas, la resolución de conflictos y la integración de código mediante pull requests.

¿Qué es el sistema de control de versiones?

Es un sistema que registra los cambios realizados en un archivo o conjunto de archivos a lo largo del tiempo, de modo que sea posible recuperar versiones anteriores más adelante.

Instalación de Git

Instalación en Linux

Para instalar Git en Linux a través de un instalador binario, en general se puede realizar mediante la herramienta básica de administración de paquetes que trae distribución. Por ejemplo si se está en Fedora, puedes usarse yum:

$ yum install git

Si se está en una distribución basada en Debian como Ubuntu, puedes usarse apt-get:

$ apt-get install git

Instalación en Windows

La forma oficial de instalar git en Windows está disponible para ser descargada en el sitio web de Git (http://git-scm.com/download/win) y la descarga empezará automáticamente.

Otra forma de obtener Git fácilmente es mediante la instalación de GitHub para Windows. El instalador incluye la versión de línea de comandos y la interfaz de usuario de Git. Además, funciona bien con Powershell y establece correctamente "caching" de credenciales y configuración CRLF adecuada. Se puede descargar este instalador del sitio web de GitHub para Windows en http://windows.github.com.

Instalación a partir del Código Fuente

Para instalar Git desde el código fuente se necesita tener las siguientes librerías de las que Git depende: curl, zlib, openssl, expat y libiconv. Por ejemplo, si se está en un sistema que tiene yum (como Fedora) o apt-get (como un sistema basado en Debian), puede usarse estos comandos para instalar todas las dependencias:

$ yum install curl-devel expat-devel gettext-devel \ openssl-devel zlib-devel

$ apt-get install libcurl4-gnutls-dev libexpat1-dev gettext \ libz-dev libssl-dev

Cuando se tengan todas las dependencias necesarias, se puede descargar la versión más reciente de Git en diferentes sitios. Puede obtenerse a partir del sitio Kernel.org en https://www.kernel.org/pub/software/scm/git, o su "mirror" en el sitio web de GitHub en https://github.com/git/git/releases. Luego se compila e instala de la siguiente manera:

$ tar -zxf git-2.0.0.tar.gz

$ cd git-2.0.0

$ make configure

$ ./configure --prefix=/usr

$ make all doc info

$ sudo make install install-doc install-html install-info

Una vez hecho esto, también se puede obtener Git, a través del propio Git, para futuras actualizaciones:

$ git clone git://git.kernel.org/pub/scm/git/git.git

Trabajando con Git Bash

Git Bash es una aplicación para entornos Windows que ofrece una emulación de la terminal Bash, permitiendo utilizar comandos Unix y de Git en una interfaz de línea de comandos.

Crear un repositorio en Git

Si se está empezando a seguir un proyecto existente en Git, se debe ir al directorio del proyecto y usar el siguiente comando:

$ git init

Esto crea un subdirectorio nuevo llamado. git, el cual contiene todos los archivos necesarios del repositorio un esqueleto de un repositorio de Git.

Si se desea empezar a controlar versiones de archivos existentes (a diferencia de un directorio vacío), probablemente se debería comenzar el seguimiento de esos archivos y hacer una confirmación inicial, utilizando comandos como git add para especificar qué archivos se desea controlar, seguidos de un git commit para confirmar los cambios:

$ git add *.c

$ git add LICENSE

$ git commit -m 'initial project version'

Comparar cambios de código

Se utiliza el comando git diff para ver qué líneas se han modificado, añadido o eliminado entre el directorio de trabajo, el área de preparación (staging) y los commits.

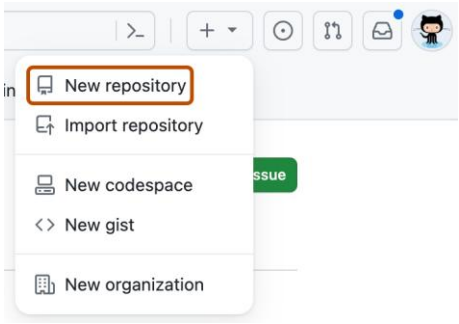
Trabajar con GitHub

GitHub es una plataforma que permite almacenar repositorios, colaborar en proyectos y gestionar versiones de código.

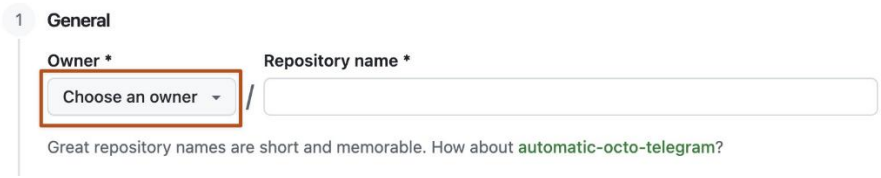
Crear un repositorio en GitHub

Creación de un nuevo repositorio desde la interfaz web

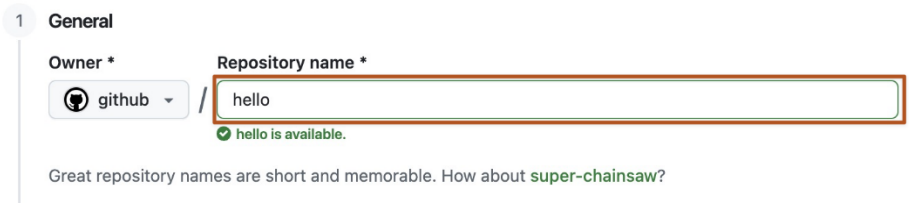
1. En la esquina superior derecha de cualquier página, seleccionar y luego hacer clic en Nuevo repositorio.



2. Opcionalmente, para crear un repositorio con la estructura de directorios y los archivos de un repositorio existente, seleccionar el menú desplegable "Elegir una plantilla" y hacer clic en un repositorio de plantillas. Se verán los repositorios de plantillas de su propiedad y de organizaciones de las que es miembro o que ha utilizado anteriormente.
3. Opcionalmente, si se elige utilizar una plantilla, para incluir la estructura del directorio y los archivos de todas las ramas en la plantilla, y no solo la rama predeterminada, seleccionar Incluir todas las ramas.
4. Utilizar el menú desplegable Propietario para seleccionar la cuenta que desea que sea propietaria del repositorio.



5. Escribir un nombre para su repositorio (máximo 100 caracteres) y una descripción opcional.



6. Seleccionar la visibilidad del repositorio.
7. Si no se usa una plantilla, hay varios elementos opcionales que se pueden precargar en el repositorio. Si se importa un repositorio existente a GitHub, no se debe elegir ninguna de estas opciones, ya que se podría generar un conflicto de fusión. Se puede agregar o crear archivos nuevos mediante la interfaz de usuario o añadirlos mediante la línea de comandos más adelante.
    - Se puede crear un archivo README, que es un documento que describe el proyecto.
    - Se puede crear un archivo.gitignore , que es un conjunto de reglas de omisión.
    - Puede optar por agregar una licencia de software a su proyecto.
8. Hacer clic en Crear repositorio.
9. En la parte inferior de la página de Configuración rápida, en "Importar código de un repositorio antiguo", se puede importar un proyecto a su nuevo repositorio. Para ello, hacer clic en " Importar código" .

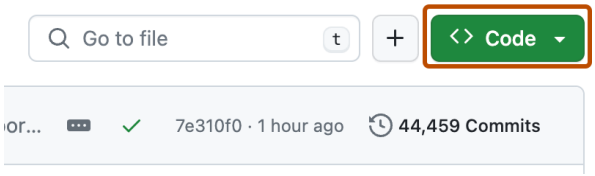
<u>Creación de un nuevo repositorio a partir de una consulta de URL</u>

Se puede usar parámetros de consulta para rellenar automáticamente los campos del formulario al crear un nuevo repositorio. Los parámetros de consulta son partes opcionales de una URL que se pueden personalizar para compartir la vista de una página web específica, como los resultados de un filtro de búsqueda o una plantilla de incidencia en GitHub. Para especificar valores para los parámetros de consulta predefinidos, se debe hacer coincidir el par clave-valor.

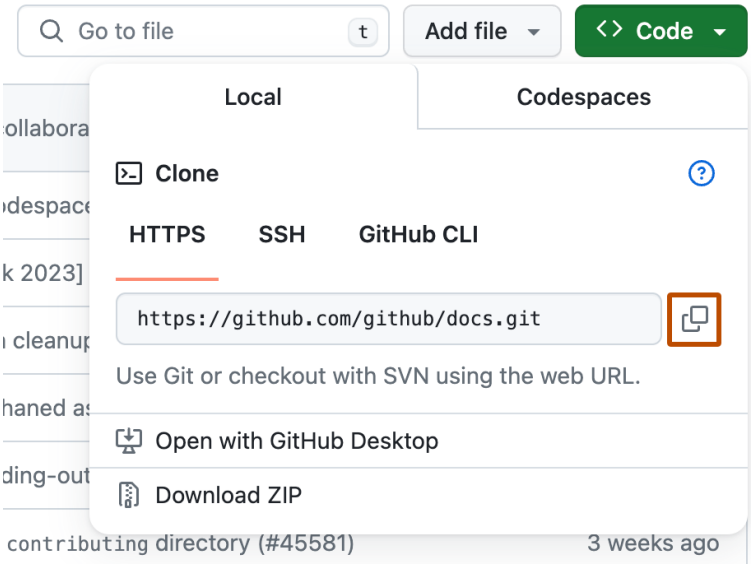
Rellenar previamente los campos del formulario con una consulta de URL puede ser útil si se desea crear repositorios con la misma configuración predeterminada.

Clonar un repositorio desde GitHub

1. En GitHub, navegar a la página principal del repositorio.
2. Sobre la lista de archivos, hacer clic en Código .



3. Copiar la URL del repositorio.
    - Para clonar el repositorio usando HTTPS, en "HTTPS", hacer clic en .
    - Para clonar el repositorio usando una clave SSH, incluido un certificado emitido por la autoridad de certificación SSH de su organización, haga clic en **SSH** y luego en .
    - Para clonar un repositorio usando GitHub CLI, haga clic en **GitHub CLI** y luego en .



4. Abrir Git Bash .
5. Cambiar el directorio de trabajo actual a la ubicación donde desea el directorio clonado.
6. Escribir git cloney luego pegar la URL que se copió anteriormente.

   git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY
7. Presionar Enter para crear su clon local.

   $ git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY

   > Cloning into `Spoon-Knife`...

> remote: Counting objects: 10, done.

> remote: Compressing objects: 100% (8/8), done.

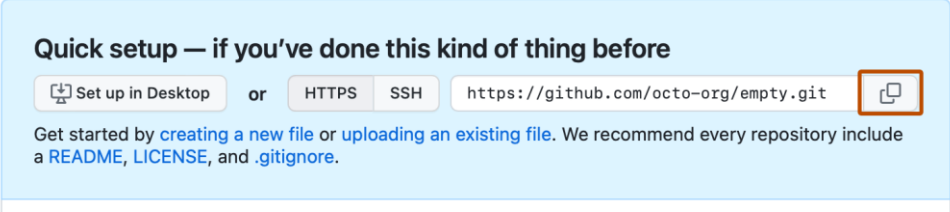
> remove: Total 10 (delta 1), reused 10 (delta 1)

> Unpacking objects: 100% (10/10), done.

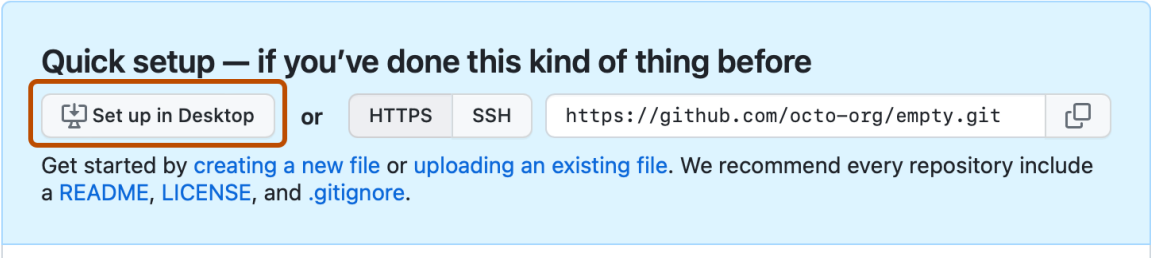
Clonación de un repositorio vacío

Un repositorio vacío no contiene archivos. Esto suele ocurrir si no se inicializa el repositorio con un README al crearlo.

1. En GitHub, navegar a la página principal del repositorio.
2. Para clonar su repositorio mediante la línea de comandos con HTTPS, en "Configuración rápida", hacer clic en . Para clonar el repositorio con una clave SSH, incluido un certificado emitido por la autoridad de certificación SSH de su organización, hacer clic en **SSH** y, a continuación, en .



Como alternativa, para clonar el repositorio en el escritorio, hacer clic en Configurar en el escritorio y seguir las instrucciones para completar la clonación.



3. Abrir Git Bash .
4. Cambiar el directorio de trabajo actual a la ubicación donde desea el directorio clonado.
5. Escribir git cloney luego pegar la URL que se copió anteriormente.

git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY

6. Presionar Enter para crear su clon local.

   $ git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY

   > Cloning into `Spoon-Knife`...

   > remote: Counting objects: 10, done.

   > remote: Compressing objects: 100% (8/8), done.

   > remove: Total 10 (delta 1), reused 10 (delta 1)

> Unpacking objects: 100% (10/10), done.

Insertar el código en el repositorio remoto

Para agregar un repositorio remoto nuevo, use el comando git remote add en el terminal, dentro del directorio donde está almacenado su repositorio.

El comando git remote add toma dos argumentos:

- Un nombre remoto, por ejemplo, origin
- Una dirección URL remota, por ejemplo, https://github.com/OWNER/REPOSITORY.git

Trabajar en equipo

GitHub permite colaboración mediante organizaciones, equipos y repositorios compartidos, para ello se necesita crear una cuenta personal o iniciar sesión en una cuenta existente de GitHub, crear una organización y configurar la facturación.

Cómo un desarrollador envía código a un repositorio remoto

Para que un desarrollador envíe código a un repositorio remoto de manera profesional y siguiendo las buenas prácticas de Git, se sigue un flujo de trabajo cíclico.

1. Preparar los cambios (git add): El desarrollador selecciona qué archivos modificados desea incluir en la próxima "foto" o versión del proyecto. Esto mueve los archivos al Staging Area.
2. Confirmar los cambios (git commit): Se crea un registro local con un mensaje descriptivo que explica qué se hizo. Esto guarda los cambios en el historial local del desarrollador.
3. Sincronizar el remoto (git pull): Antes de enviar, es una buena práctica descargar los cambios que otros compañeros hayan subido para evitar conflictos.

4. Enviar el código (git push): Se transfieren los commits del repositorio local al servidor remoto (como GitHub).

Trabajar con commits

Un commit es una captura del estado del proyecto en un momento determinado. Cada commit tiene un SHA-1 (identificador único) y un mensaje descriptivo.

Trabajar con branches en GitHub

Las ramas (branches) permiten divergir de la línea principal de desarrollo para trabajar en nuevas funcionalidades o correcciones sin afectar la rama main.

Resolución de conflictos de Merge

Ocurre cuando hay cambios divergentes en la misma línea de un archivo. Git marca el archivo como "unmerged" y el usuario debe editar manualmente el archivo para decidir qué cambios conservar antes de finalizar el merge.

Creación de un pull request (PR)

Es una función exclusiva de plataformas como GitHub que permite proponer cambios de una rama a otra, facilitando la revisión de código por parte de pares y la discusión antes de la integración final.

Para realizarlo se deben seguir los siguientes pasos:

1. Subir la rama local: Primero, se debe haber trabajado en una rama distinta a la principal (por ejemplo, feature-nueva-mejora) y haberla subido al servidor con git push origin nombre-de-tu-rama.
2. Abrir el comparador en GitHub: Al entrar al repositorio en GitHub, aparecerá automáticamente un aviso amarillo que dice "Compare & pull request". Si no aparece, ir a la pestaña "Pull requests" y hacer clic en el botón verde "New pull request".
3. Seleccionar las ramas: Elegir la rama de "base" (generalmente main o develop) y la rama "compare" (la rama con tus cambios).
4. Describir los cambios: Se abre un formulario donde se debe poner un título claro y una descripción detallada de qué problemas resuelve tu código o qué funcionalidades añade.
5. Enviar la solicitud: Hacer clic en "Create pull request".

Revertir cambios del directorio de trabajo

Se puede usar git revert para deshacer un commit sin perder historial, o git reset para modificar el estado del directorio.

Conclusión

Conocer y dominar Git, GitHub y GitFlow resulta fundamental para cualquier profesional de la informática, ya que estas herramientas constituyen la base del control de versiones y del trabajo colaborativo en proyectos de software. Su uso permite garantizar la integridad del código al mantener un historial claro y rastreable de cada modificación, facilita la coordinación entre equipos mediante ramas y flujos de trabajo organizados, y optimiza la resolución de problemas al posibilitar la identificación y corrección de errores sin comprometer la estabilidad del proyecto.

Referencias

Git (s.f.). *Sobre el control de versiones.* https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Acerca-del-Control-de-Versiones

Git (s.f.). *Instalación de git*. https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Instalaci%C3%B3n-de-Git

Git (s.f.). *Git en otros entornos - Git con Bash*. https://git-scm.com/book/es/v2/Ap%C3%A9ndice-A:-Git-en-otros-entornos-Git-con-Bash

Git (s.f.). *Obteniendo un repositorio Git*. https://git-scm.com/book/es/v2/Fundamentos-de-Git-Obteniendo-un-repositorio-Git

Git (s.f.).Git-diff. https://git-scm.com/docs/git-diff

GitHub (2026). *Documentación sobre la introducción a GitHub. https*://docs.github.com/es/get-started?utm_source=copilot.com

GitHub (2026). *Crear un repositorio nuevo.* https://docs.github.com/es/repositories/creating-and-managing-repositories/creating-a-new-repository?utm_source=copilot.com

GitHub (2026). *Clonar un repositorio. https://docs.github.com/es/repositories/creating-and-managing-repositories/cloning-a-repository?utm_source=copilot.com*

GitHub (2026). *Administrar repositorios remotos. https://docs.github.com/es/get-started/git-basics/managing-remote-repositories?utm_source=copilot.com*

GitHub (2026). *Iniciar con GitHub Team*. https://docs.github.com/es/get-started/onboarding/getting-started-with-github-team?utm_source=copilot.com

GitHub (2026). *Administrar repositorios remotos*. https://docs.github.com/es/get-started/git-basics/managing-remote-repositories?utm_source=copilot.com

GitHub(2026). *Acerca de las ramas*. https://docs.github.com/es/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-branches

GitHub (2026). *Crear una solicitud de incorporación de cambios*. https://docs.github.com/es/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-a-pull-request