# Uso de GitHub: Guía Básica

Aprende los fundamentos de GitHub para colaborar en proyectos de desarrollo de software

# Introducción a GitHub y Git

## Git

Sistema de **control de versiones** que realiza un seguimiento de los cambios en los archivos.

- Funciona **localmente** en tu máquina
- Permite crear **ramas** para trabajar en paralelo
- Registra cada cambio con **confirmaciones**
- Software de código abierto creado por Linus Torvalds

## GitHub

Plataforma **basada en la nube** para almacenar, compartir y colaborar en código.

- Almacena repositorios Git en la **nube**
- Facilita la **colaboración** entre desarrolladores
- Ofrece herramientas de revisión de código
- Integra **CI/CD** y gestión de proyectos

# Funcionalidades principales de GitHub

### Repositorios

Almacenamiento de código con **control de versiones** y seguimiento completo del historial de cambios

### Colaboración

Trabajo en equipo con **múltiples desarrolladores** en el mismo proyecto sin conflictos

### Ramas

Creación de **versiones paralelas** para experimentar sin afectar el código principal

### Pull Requests

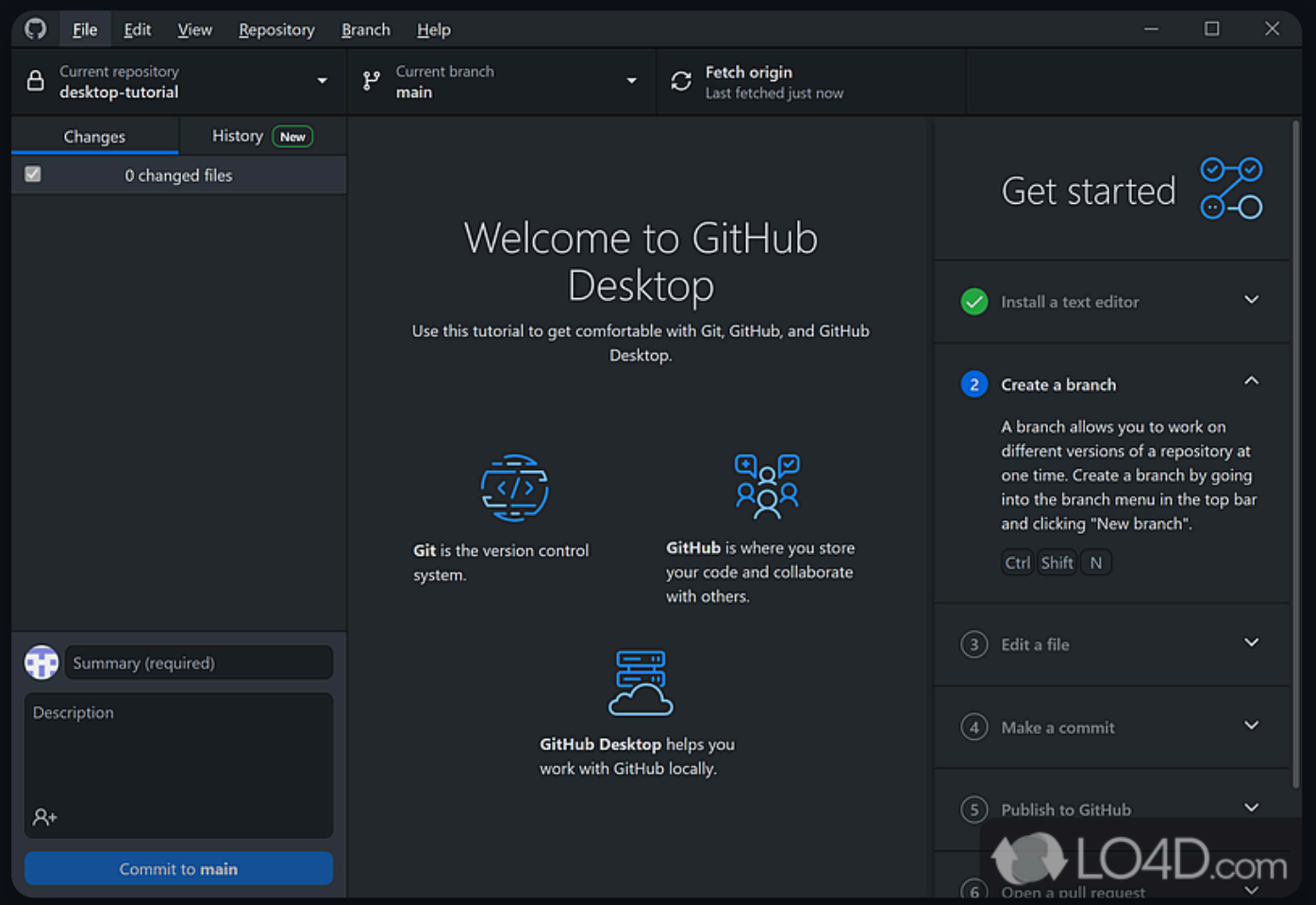
Solicitudes para **revisar y fusionar** cambios antes de integrarlos al proyecto principal

### Issues

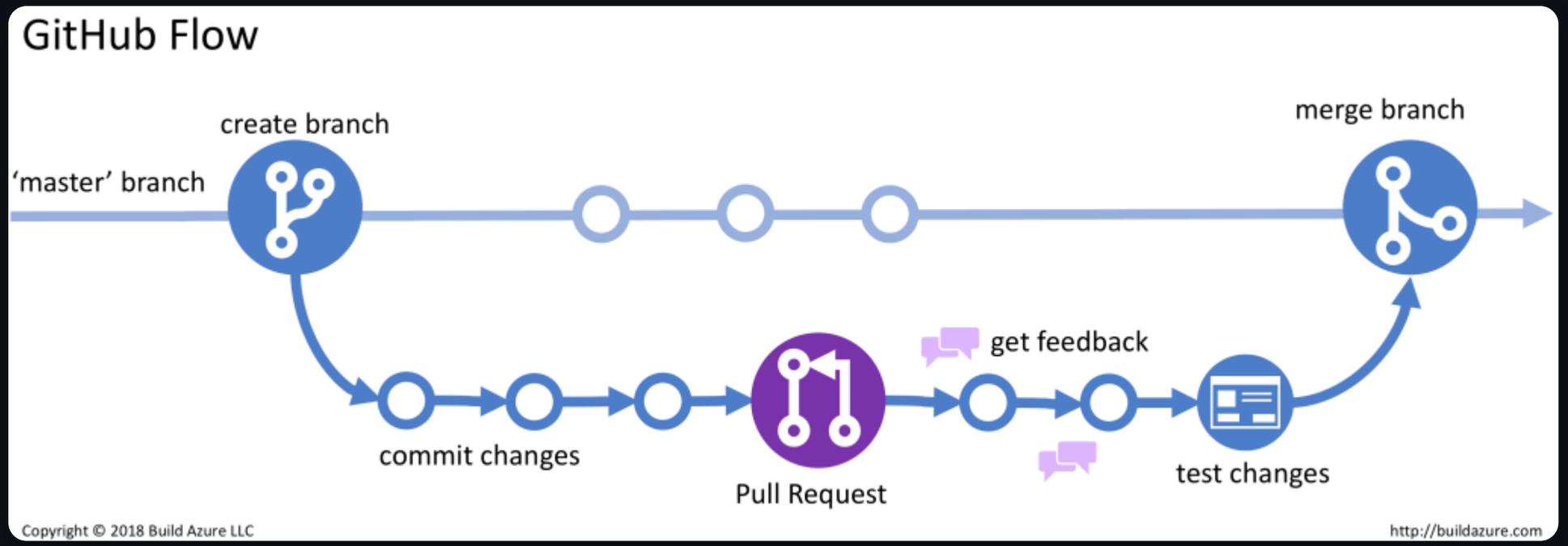
Sistema de **seguimiento de problemas** para reportar bugs y solicitar nuevas funcionalidades

### GitHub Pages

Hospedaje **gratuito de sitios web** directamente desde repositorios GitHub



# Flujo de trabajo básico de GitHub




1

### Crear rama

Genera una **versión separada** del proyecto para trabajar sin afectar el código principal

2

### Realizar cambios

Modifica archivos y **confirma** los cambios con mensajes descriptivos

3

### Solicitud de cambios

Crea una **Pull Request** para solicitar revisión y retroalimentación

4

### Revisión

Los colaboradores **revisan** el código y sugieren mejoras

5

### Fusión

Una vez aprobado, **fusiona** los cambios a la rama principal y elimina la rama

# Comandos básicos de Git

### git clone

```
git clone <url>
```

Descarga un **repositorio remoto** completo a tu máquina local

### git branch

```
git branch <nombre>
```

Crea una **nueva rama** para trabajar en paralelo sin afectar el código principal

### git checkout

```
git checkout <rama>
```

Cambia a una **rama existente** o crea una nueva rama y cambia a ella

### git status

```
git status
```

Muestra el **estado actual** del repositorio: archivos modificados, sin seguimiento, etc.

### git add

```
git add <archivo>
```

Añade archivos al **área de preparación** para incluirlos en el próximo commit

### git commit

```
git commit -m "mensaje"
```

Guarda los cambios en el **repositorio local** con un mensaje descriptivo

### git push

```
git push <origen> <rama>
```

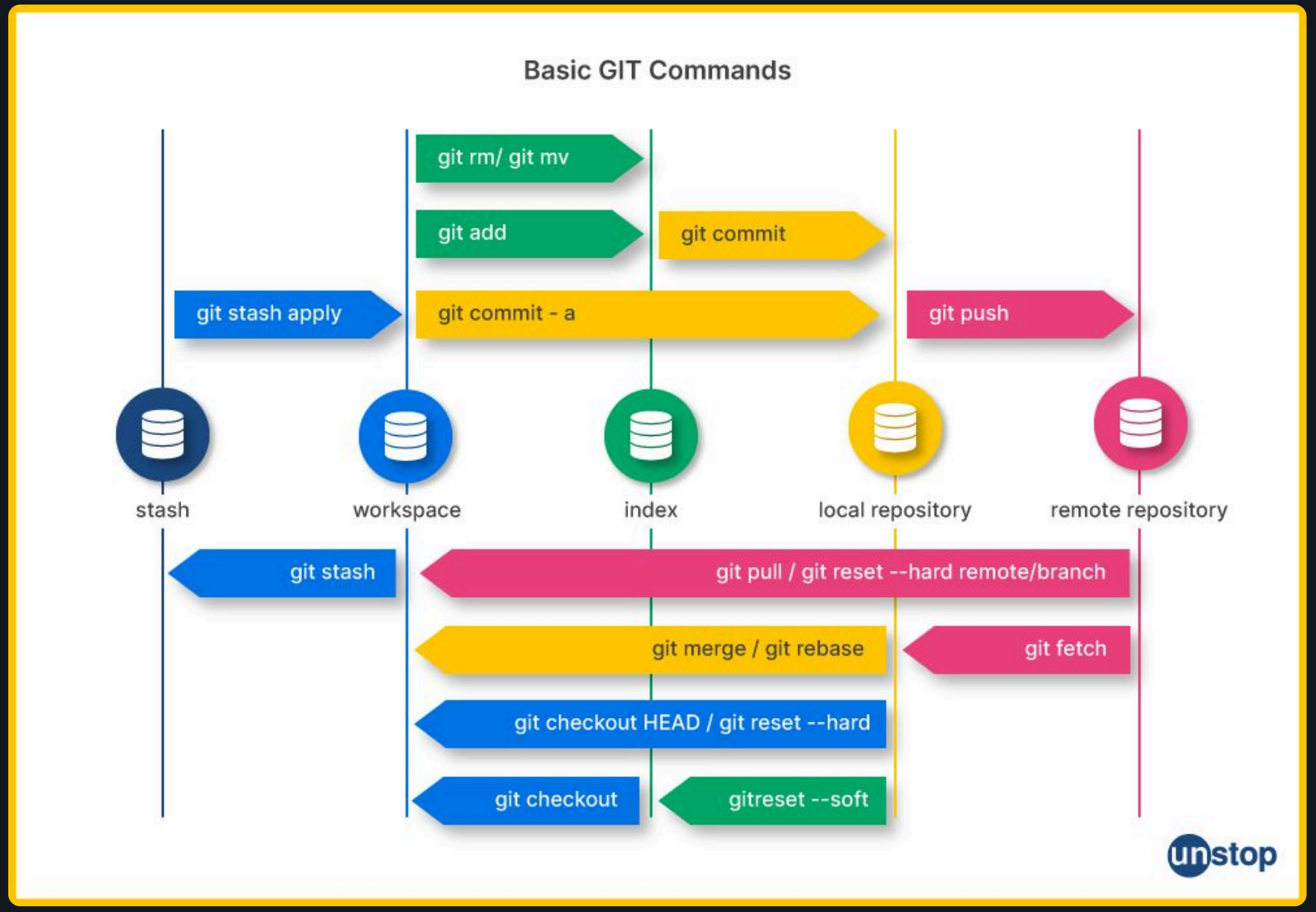
Envía los commits locales al **repositorio remoto** para compartirlos con otros

### git pull

```
git pull <origen> <rama>
```

Descarga y **fusiona** cambios del repositorio remoto a tu rama local

### Ejemplo de uso



# Ejemplos prácticos de uso de GitHub

## Crear un repositorio

1. Inicia sesión en GitHub y haz clic en **"New repository"**
2. Asigna un **nombre descriptivo** y una descripción
3. Elige entre **público o privado** y añade README
4. Clona el repositorio localmente:

```
git clone https://github.com/usuario/nombre-repo.git
```

## Clonar un proyecto

1. Busca el repositorio que quieres clonar
2. Haz clic en el botón **"Code"** y copia la URL
3. Abre tu terminal y ejecuta:
4. Navega al directorio del proyecto:

```
cd nombre-repo
```

## Contribuir a código abierto

1. Haz **fork** del repositorio original
2. Clona tu fork a tu máquina local
3. Crea una **nueva rama** para tus cambios:
4. Realiza tus cambios y haz commit:

```
git commit -m "Descripción del cambio"
```

**Consejo práctico:** Antes de enviar cambios, siempre actualiza tu rama local con los cambios del repositorio remoto usando **git pull** para evitar conflictos de fusión.

# Conclusiones y recursos adicionales

## Puntos clave

- GitHub es una plataforma **basada en la nube** para colaborar en código
- Git realiza el **seguimiento de cambios** en los archivos
- El flujo de trabajo incluye: **crear ramas, hacer cambios, solicitar revisión y fusionar**
- Los comandos básicos facilitan la **gestión de versiones** y colaboración
- Practicar con proyectos reales es la mejor forma de **aprender GitHub**

## Recursos para aprender más

### Documentación oficial
docs.github.com - Guías completas y referencias

### GitHub Learning Lab
Cursos interactivos gratuitos para aprender GitHub

### Tutoriales en video
YouTube y GitHub Learning Channel

### Proyectos de código abierto
Contribuye a proyectos existentes para practicar

**Consejo final:** La mejor manera de dominar GitHub es practicando constantemente. Comienza con proyectos pequeños y aumenta gradualmente la complejidad.