Guía completa: Crear archivos de texto y trabajar con Git

Requisitos previos

- Tener Git instalado en tu sistema
- Tener una cuenta en GitHub
- Tener PowerShell o terminal disponible

Paso 1: Configurar Git (solo la primera vez)

```powershell
git config --global user.name "Tu Nombre"
git config --global user.email "tu.email@gmail.com"
```

Paso 2: Crear y configurar tu repositorio

2.1 Navegar a tu carpeta de proyecto

```powershell
cd ruta\a\tu\proyecto
# Ejemplo: cd C:\Users\Usuario\MiProyecto
```

2.2 Inicializar repositorio Git

```powershell
git init
```

2.3 Crear repositorio en GitHub

1. Ve a GitHub.com
2. Haz clic en "New repository" (botón verde)
3. Escribe el nombre del repositorio (ejemplo: `MiProyecto`)
4. Marca "Public" o "Private" según prefieras
5. **NO marques** "Add a README file"
6. Haz clic en "Create repository"

2.4 Conectar repositorio local con GitHub

```powershell
git remote add origin https://github.com/TuUsuario/NombreDelRepositorio.git
```

Reemplaza:

- `TuUsuario` por tu nombre de usuario de GitHub
- `NombreDelRepositorio` por el nombre exacto de tu repositorio

Paso 3: Crear archivos de texto

Método 1: New-Item (recomendado)

```powershell
New-Item -Path "NombreArchivo.txt" -ItemType File -Value "Contenido del archivo"
```

Método 2: Operador de redirección

```powershell
"Contenido del archivo" > NombreArchivo.txt
```

Método 3: Out-File

```powershell
"Contenido del archivo" | Out-File -FilePath "NombreArchivo.txt"
```

Paso 4: Agregar archivos al control de versiones

4.1 Verificar archivos creados

```powershell
dir
# o también
ls
```

4.2 Agregar archivos al staging area

```powershell
# Agregar un archivo específico
git add NombreArchivo.txt

# Agregar todos los archivos
git add .
```

4.3 Verificar estado

```powershell
git status
```

Paso 5: Confirmar cambios (commit)

```powershell
git commit -m "Descripción clara de los cambios realizados"
```

Ejemplos de mensajes de commit:

- "Agregar archivo inicial del proyecto"
- "Crear archivo de configuración"
- "Actualizar contenido del archivo principal"

Paso 6: Subir cambios a GitHub

6.1 Primera vez (establecer rama principal)

```powershell
git push -u origin main
```

6.2 Siguientes veces

```powershell
```

```
git push
```

Comandos útiles adicionales

Verificar configuración de repositorio remoto

```powershell
git remote -v
```

Ver historial de commits

```powershell
git log
git log --oneline  # versión resumida
```

Ver diferencias antes de hacer commit

```powershell
git diff
```

Deshacer cambios no confirmados

```powershell
git restore NombreArchivo.txt
```

Quitar archivo del staging area

```powershell
git restore --staged NombreArchivo.txt
```

Flujo de trabajo completo (ejemplo práctico)

```powershell
```

```
# 1. Ir a la carpeta del proyecto
cd C:\Users\Usuario\MiProyecto

# 2. Inicializar Git (solo primera vez)
git init

# 3. Conectar con GitHub (solo primera vez)
git remote add origin https://github.com/TuUsuario/MiProyecto.git

# 4. Crear archivo
New-Item -Path "ejemplo.txt" -ItemType File -Value "Este es mi contenido"

# 5. Agregar al staging
git add ejemplo.txt

# 6. Confirmar cambios
git commit -m "Agregar archivo ejemplo.txt"

# 7. Subir a GitHub
git push -u origin main  # primera vez
# git push               # siguientes veces
```

Errores comunes y soluciones

Error: "pathspec 'archivo.txt' did not match any files"

Causa: El archivo no existe en el sistema **Solución:** Crear el archivo primero con `New-Item` antes de hacer `git add`

Error: "remote origin already exists"

Causa: Ya tienes configurado el repositorio remoto **Solución:** No necesitas hacer nada, el repositorio ya está conectado

Error en comando `git remote`

Formato correcto:

```
powershell
```

```
git remote -v ..... # ver repositorios remotos
git remote add origin URL  # agregar repositorio remoto
git remote remove origin  # quitar repositorio remoto
```

Consejos importantes

1. **Siempre verifica** el estado con `git status` antes de hacer commit
2. **Escribe mensajes de commit descriptivos** que expliquen qué cambios hiciste
3. **Haz commits frecuentes** con cambios pequeños y relacionados
4. **Revisa** tu repositorio en GitHub después de hacer push para confirmar que se subió correctamente
5. **Mantén** tus archivos organizados en carpetas si el proyecto crece