Guía Completa de Git para Principiantes

1. Crear una Cuenta en GitHub y Configurar Git con Tu Usuario

1. **Crea tu cuenta** en GitHub desde github.com.
2. Configura tu usuario en Git para que los cambios que registres estén vinculados con tu cuenta. Asegúrate de que el correo coincida con el registrado en GitHub:

   ```bash
   git config --global user.name "TuNombreDeUsuario"
   git config --global user.email "tuemail@ejemplo.com"
   ```

   Ejemplo:

   ```bash
   git config --global user.name "JaneDoe"
   git config --global user.email "janedoe@example.com"
   ```

3. Para verificar que los datos se guardaron correctamente, usa:

   ```bash
   git config --list
   ```

   Esto mostrará toda la configuración, incluyendo tu nombre y correo.

Frecuencia de Uso:

- `git config` se usa una sola vez para configurar tu usuario, pero puedes cambiar los datos en cualquier momento si es necesario.

Error Común:

- **Correo Incorrecto**: Si el correo no coincide con el de GitHub, verás un error al intentar hacer `push`. Corrige con:

  ```bash
  git config --global user.email "tuemailcorrecto@ejemplo.com"
  ```

---

2. Crear un Repositorio Nuevo en GitHub

1. En GitHub, da clic en **New Repository** en la esquina superior derecha.

2. Nombra el repositorio con un nombre claro y descriptivo (ej.: `mi-portfolio-html`).
3. Elige entre un repositorio público o privado.
4. Haz clic en **Create Repository**.

> *Consejo de Nombres: Usa nombres en inglés, sin caracteres especiales, y con guiones bajos `_` o guiones medios `-` para mejorar la legibilidad.*

Frecuencia de Uso:

- Crear un repositorio en GitHub se hace una sola vez por cada proyecto.

---

3. Inicializar el Repositorio Local (`git init`)

Para "iniciar" el sistema de control de versiones en una carpeta en tu computadora, usa:

```bash
git init
```

Ejemplo de Vida Cotidiana: Esto es como decidir que quieres empaquetar artículos para enviar por correo. Ahora tu carpeta es un almacén listo para empaquetar.

Frecuencia de Uso:

- `git init` se usa una vez para cada proyecto nuevo que desees manejar con Git.

Error Común:

- **Intentar inicializar Git en una carpeta que ya es un repositorio**: Si ves el mensaje `fatal: not a git repository`, verifica que estés en la carpeta correcta. Usa `ls -a` para confirmar que no tienes un archivo `.git` en otra carpeta.

---

4. Preparar los Archivos para el Envío (`git add`)

Después de hacer cambios en los archivos, usa `git add` para "empaquetarlos" en el próximo commit.

```bash
git add .
```

Este comando agrega todos los archivos. Si prefieres agregar uno específico, usa:

```bash
git add nombre_del_archivo.html
```

Ejemplo de Vida Cotidiana: Como seleccionar los objetos para empaquetar y tener listos antes de cerrar la caja (commit).

Frecuencia de Uso:

- Usa `git add` cada vez que tengas archivos nuevos o modificados que quieras incluir en un commit.

Error Común:

- **Olvidar agregar archivos**: Usa `git status` para verificar si te faltó agregar algo antes de hacer commit.

---

5. Revisar el Estado con `git status`

Antes de hacer el commit, revisa el estado de los archivos con `git status`.

```bash
git status
```

Ejemplo de Vida Cotidiana: Piensa en `git status` como revisar la lista de empaque para asegurarte de que todo lo necesario está en la caja.

Frecuencia de Uso:

- Usa `git status` después de cada `git add` o cada vez que quieras confirmar qué archivos están listos para el commit.

---

6. Registrar una Nueva Versión con `git commit` (Cerrar el Paquete)

Para “cerrar la caja” y registrar los cambios en Git:

```bash
git commit -m "Mensaje descriptivo de los cambios"
```

Ejemplo de Vida Cotidiana

Este paso es como sellar la caja del paquete y etiquetarlo con una nota explicando el contenido.

Buenas Prácticas para Mensajes de Commit

1. **Usa verbos en infinitivo** y sé claro.
   - **Ejemplo**: `"Agrega archivo HTML con estructura inicial"`
2. **Evita mensajes genéricos** que no den contexto.
   - **Ejemplo incorrecto**: `"cosas nuevas"`
   - **Ejemplo correcto**: `"Agrega estructura de navegación en el archivo HTML"`
3. **Haz commits pequeños**: registra cambios específicos en cada commit.

Ejemplos de Commit con *Conventional Commits*

Usando *Conventional Commits*, el mensaje se estructura con una palabra clave inicial:

- **Ejemplo 1**: `feat: agrega archivo HTML inicial con estructura`
- **Ejemplo 2**: `fix: corrige ruta en imagen de logo`

Frecuencia de Uso:

- Haz un `git commit` cada vez que completes una serie de cambios que quieras registrar. Es recomendable hacer commits frecuentemente para registrar el progreso.

Errores Comunes:

1. **Olvidar el `-m`**: Si no pones el `-m`, se abrirá un editor de texto en la terminal para que escribas el mensaje. En el editor:
   - Escribe el mensaje (ej.: `"Agrega estructura inicial de HTML"`).
   - Guarda y cierra:
     - En *vim*: `Esc`, luego `:wq` y `Enter`.

- En *nano*: `Ctrl + X`, `Y` y `Enter`.

Si se creó exitosamente, verás el ID del commit en consola.

2. **Mensaje Vago o Confuso**: Para evitar ambigüedad, usa verbos y sé específico.
   - **Mal**: `"Update"`
   - **Mejor**: `"Corrige nombre de archivo de imagen en el índice"`

Cuándo es Conveniente o No Hacer un Commit

- **No Hacer Commit**:
  - Si el cambio es pequeño o temporal y no aporta valor, como un ajuste temporal.
  - Para experimentos o pruebas que no están listos para integrarse al proyecto.
- **Hacer Commit**:
  - Cuando finalizas una tarea completa, como una sección HTML lista.
  - Al realizar cambios importantes en la estructura o archivos principales.

---

7. Conectar el Repositorio Local con GitHub (Configurar `origin`)

Conecta tu proyecto con el repositorio en GitHub usando `origin`, que es como la "dirección" del destinatario:

```bash
git remote add origin https://github.com/tu-usuario/tu-repositorio.git
```

Ejemplo de Vida Cotidiana: `origin` es como la dirección de la oficina postal a la que envías el paquete. Puedes cambiar el "destino" con otros comandos si lo necesitas.

Frecuencia de Uso:

- `git remote add origin` se usa una vez por cada proyecto.

Error Común:

- **Dirección Incorrecta**: Si colocas mal el enlace del repositorio, verás el mensaje `fatal: remote origin already exists`. Corrige con:

  ```bash
  git remote set-url origin https://github.com/tu-usuario/tu-repositorio.git
  ```

---

8. Subir los Cambios al Repositorio Remoto (`git push`)

Para enviar los cambios al repositorio remoto:

```bash
git push -u origin main
```

Este comando establece `origin` como destino y `main` como la rama principal. A partir de ahí, puedes usar solo `git push`.

Ejemplo de Vida Cotidiana: `git push` es como ir a la oficina de correo y enviar el paquete.

Frecuencia de Uso:

- Usa `git push` cada vez que quieras enviar nuevos commits al repositorio en GitHub.

Error Común:

- **Intentar hacer `push` sin haber hecho `commit`**: Si intentas enviar cambios sin hacer commit, recibirás el mensaje `nothing to commit`. Usa `git commit` primero.

Ejemplos de Buenas y Malas Prácticas en Mensajes de Commit

Ejemplos Malos

- `"cosas nuevas"`
- `"arreglos varios"`
- `"final"`

Ejemplos Mejorados

- `"Agrega imagen de logo en HTML"`
- `"Configura Prettier con reglas básicas para HTML"`
- `"Corrige nombre de archivo en el índice"`

Usando *Conventional Commits*

Al seguir *Conventional Commits*, los mensajes se estructuran con una palabra clave inicial:

- **Ejemplo 1**: `feat: agrega archivo HTML con estructura básica`
- **Ejemplo 2**: `fix: corrige ruta de imagen en el archivo HTML`
- **Ejemplo 3**: `style: aplica formato con Prettier en index.html`
