

# Guia Completo Git e GitHub no Terminal (Profissional)

Documento completo com os principais comandos Git e GitHub via terminal. Inclui versionamento, envio ao GitHub, sincronização, atualização local e reversão de versões.

## 1. Configuração Inicial

```
git config --global user.name "Seu Nome"  
git config --global user.email "seuemail@email.com"
```

Define usuário padrão dos commits.

## 2. Iniciar Repositório

```
git init
```

Inicializa repositório local.

## 3. Clonar do GitHub

```
git clone https://github.com/usuario/repositorio.git
```

Baixa projeto do GitHub para máquina local.

## 4. Verificar Status

```
git status
```

Mostra arquivos modificados.

## 5. Adicionar Arquivos

```
git add nome_arquivo  
git add .
```

Prepara arquivos para commit.

## 6. Commit

```
git commit -m "mensagem"
```

Salva alterações localmente.

## 7. Conectar ao GitHub (caso ainda não esteja conectado)

```
git remote add origin https://github.com/usuario/repositorio.git
git branch -M main
git remote -v
```

Configura repositório remoto.

## 8. Enviar Arquivos para GitHub (SINCRONIZAR LOCAL -> GITHUB)

Primeiro envio:  
`git push -u origin main`

Envios seguintes:  
`git push`

Sincroniza arquivos locais com GitHub.

## 9. ATUALIZAR ARQUIVOS LOCAIS COM OS DO GITHUB (GITHUB -> LOCAL)

Atualizar projeto local:  
`git pull`

Baixa alterações do GitHub e atualiza sua máquina.

Atualizar branch específica:  
`git pull origin main`

Forçar atualização substituindo tudo local:  
`git fetch --all`  
`git reset --hard origin/main`

ATENÇÃO: Esse comando apaga alterações locais não salvas.

## 10. Sincronização Completa (Fluxo Profissional)

Passo padrão antes de codar:  
`git pull`

Após alterações:  
`git add .`  
`git commit -m "descrição"`  
`git push`

Esse fluxo mantém GitHub e máquina sincronizados.

## 11. Branches

Criar branch:  
`git checkout -b nova-branch`

Trocar branch:  
`git checkout main`

## 12. Histórico

```
git log --oneline
```

## 13. Reverter Versões

```
Ver commits:  
git log --oneline
```

```
Voltar temporário:  
git checkout ID
```

```
Voltar definitivo:  
git reset --hard ID
```

```
Reverter commit sem apagar histórico:  
git revert ID
```

```
Enviar reversão:  
git push
```

## 14. Resolver Conflitos de Sincronização

Quando aparecer conflito no git pull:

1. Abrir arquivos com conflito
2. Ajustar manualmente
3. Salvar

```
Depois:  
git add .  
git commit -m "corrigido conflito"  
git push
```

## 15. Boas Práticas Profissionais

- Sempre git pull antes de trabalhar
- Sempre commit antes do push
- Nunca trabalhar direto na main em equipe
- Usar branches
- Evitar git reset --hard sem necessidade
- Testar antes de enviar ao GitHub