

# Comandos Git

## Comandos e Termos do Git e GitHub

*Git é um sistema de controle de versão distribuído utilizado para gerenciar o código-fonte ao longo do tempo.*

### Comandos Básicos do Git:

1. `git init`  
Inicializa um repositório Git no diretório atual.
2. `git clone <URL>`  
Clona um repositório remoto para o diretório local.
3. `git add <arquivo>`  
Adiciona arquivos ou mudanças ao índice (área de preparação) para o próximo commit.
4. `git commit -m "<mensagem>"`  
Faz um commit, ou seja, registra as mudanças no repositório local com uma mensagem.
5. `git status`  
Mostra o estado atual do repositório (arquivos modificados, não rastreados, etc.).
6. `git pull`  
Baixa as mudanças de um repositório remoto e as incorpora no repositório local.
7. `git push`  
Envia as alterações locais para o repositório remoto.
8. `git branch`  
Lista as branches (ramificações) locais.

9. `git checkout <branch>`  
Alterna para a branch especificada.
10. `git merge <branch>`  
Mescla a branch especificada com a branch atual.
11. `git log`  
Exibe o histórico de commits.
12. `git diff`  
Mostra as diferenças entre as versões de arquivos (por exemplo, antes de fazer um commit).
13. `git remote -v`  
Mostra os repositórios remotos configurados.
14. `git fetch`  
Baixa as atualizações de um repositório remoto sem integrá-las ao repositório local.
15. `git reset`  
Desfaz as alterações feitas, podendo ser usado para resetar o índice ou o repositório a um commit anterior.
16. `git rm <arquivo>`  
Remove um arquivo do repositório e do sistema de arquivos.
17. `git tag <nome>`  
Cria uma tag (marcação) em um commit específico.
18. `git stash`  
Armazena temporariamente as alterações não comitadas para que você possa trabalhar em outra coisa.
19. `git stash apply`  
Aplica as alterações salvas no stash de volta ao diretório de trabalho.

20. `git diff --staged`  
Exibe as diferenças entre os arquivos adicionados ao índice e o último commit.
  
21. `git show <commit>`  
Exibe detalhes de um commit específico.
  
22. `git rebase`  
Reaplica commits de uma branch sobre outra base.
  
23. `git cherry-pick <commit>`  
Aplica um commit específico de uma branch para outra.
  
24. `git blame <arquivo>`  
Mostra quem fez cada alteração em um arquivo específico.

*GitHub é uma plataforma de hospedagem de código-fonte que utiliza Git para controle de versão, com foco na colaboração e compartilhamento de projetos.*

### Principais Termos do GitHub:

1. Repositório (Repository)  
Onde o código é armazenado e versionado. Pode ser público ou privado.
  
2. Branch  
Uma ramificação do repositório, usada para desenvolver funcionalidades separadas sem afetar a versão principal (geralmente chamada de main ou master).
  
3. Pull Request (PR)  
Solicitação para integrar mudanças de uma branch para outra (geralmente de uma branch de recurso para a main).
  
4. Fork  
Uma cópia de um repositório, geralmente usada para fazer alterações ou melhorias sem

afetar o repositório original.

5. Commit

Uma versão do código no repositório com um conjunto específico de mudanças.

6. Issue

Um item que pode ser usado para reportar bugs ou solicitar novas funcionalidades.

7. Wiki

Documentação do projeto hospedada no GitHub.

8. Actions

Automação de processos dentro do GitHub, como integração contínua (CI), deploy, testes, entre outros.

### Principais Ações no GitHub:

1. Criar Repositório

Para criar um repositório novo no GitHub.

2. Criar Fork

Para fazer uma cópia de um repositório de outro usuário.

3. Fazer um Pull Request

Para sugerir alterações a um repositório original de outra pessoa.

4. Gerenciar Colaboradores

Adicionar pessoas para colaborar em um repositório privado ou público.

5. Configurar GitHub Pages

Hospedar um site diretamente a partir de um repositório.

6. Adicionar Webhooks

Configurar notificações automáticas para ações específicas no repositório.

## 7. Criar um Release

Para lançar uma versão estável de um software com base nos commits feitos.

## 8. Configurar Branch Protegidas

Evita alterações diretas em branches importantes, como main ou master.

## 9. Tags

Usadas para marcar versões específicas do projeto.

## 10. Actions

Automatização de fluxos de trabalho, como testes e deploys, através do GitHub Actions.

---

Revision #1

Created 13 March 2025 13:50:44 by Tiago Alves Silva

Updated 13 March 2025 13:54:34 by Tiago Alves Silva