

1. Início e Configuração		
Comando	Descrição	Recomendação
git init	Inicializa um novo repositório local.	Use apenas no começo de um projeto novo.
git clone <url>	Baixa um repositório existente do servidor.	Use para começar a trabalhar em um projeto que já está no GitHub/GitLab.
git config --global user.name "Seu Nome"	Configura seu nome para os commits.	Faça isso logo após instalar o Git.
2. Ciclo de Alterações (O dia a dia)		
git status	Lista arquivos alterados, novos ou deletados.	<b>Obrigatório:</b> Use antes de qualquer add ou commit para conferir o que está fazendo.
git add .	Adiciona todas as mudanças para a área de preparação ( <i>stage</i> ).	Se quiser adicionar apenas um arquivo específico, use git add nome-do-arquivo.
git commit -m "mensagem"	Salva as alterações do <i>stage</i> no histórico local.	Escreva mensagens claras (Ex: "Cria tela de login" em vez de "ajustes").
git diff	Mostra exatamente quais linhas de código mudaram.	Ótimo para revisar seu código antes de dar o add.
3. Gerenciamento de Branches		
git branch	Lista os branches locais.	O branch com * é o atual.
git checkout -b <nome>	Cria um novo branch e muda para ele.	<b>Recomendação moderna:</b> Use git switch -c <nome>. É mais intuitivo.
git checkout <nome>	Muda para um branch existente.	<b>Recomendação moderna:</b> Use git switch <nome>.
git merge <nome>	Traz as alterações de outro branch para o atual.	Use para atualizar sua develop com a main, por exemplo.
git branch -d <nome>	Apaga um branch local.	Só faça isso após o <i>merge</i> para não perder trabalho.
4. Sincronização com Servidor (Remote)		
git fetch	Baixa as novidades do servidor, mas não altera seu código.	Use para "dar uma espiada" no que os colegas fizeram sem risco de conflitos.
git pull	Baixa as novidades e já tenta fazer o <i>merge</i> no seu código.	<b>Recomendação:</b> Use git pull --rebase para manter o histórico mais limpo e linear.
git push origin <nome>	Envia seus commits locais para o servidor.	Na primeira vez em um branch novo, use git push -u origin <nome>.
5. Desfazendo Erros		
git restore <arquivo>	Descarta as mudanças locais em um arquivo (volta ao que era).	Use quando você "estragou" um arquivo e quer recomeçar.
git reset --soft HEAD~1	Desfaz o último commit, mas mantém os códigos salvos.	Use se esqueceu de algo no commit anterior ou errou a mensagem.
git stash	"Esconde" suas alterações temporariamente para limpar o diretório.	Útil quando você precisa mudar de branch urgente, mas não quer fazer commit de algo incompleto.