**GIT PULL, GIT CHECKOUT, GIT LOG – RESUMO**

**O que são os comandos git pull, git checkout e git log?**

**git pull**

O comando **git pull** é usado para sincronizar seu repositório local com um repositório remoto. Ele faz duas coisas em uma única operação:

1. **git fetch**: Ele baixa as atualizações e o histórico mais recente do repositório remoto, mas não as mescla no seu branch de trabalho.
2. **git merge**: Em seguida, ele mescla as alterações baixadas no seu branch local atual.

Em resumo, git pull é a maneira mais fácil de buscar e aplicar as alterações que outros desenvolvedores fizeram no seu repositório.

**git checkout**

O comando **git checkout** é usado para navegar e trabalhar com o histórico do seu repositório. Embora ele tenha várias funcionalidades, as mais comuns são:

* **Mudar para um branch existente**: Você pode usar git checkout <nome\_do\_branch> para alternar do branch atual para outro. Isso é útil para trabalhar em diferentes funcionalidades ou correções de bugs.
* **Criar um novo branch**: A forma mais comum de criar um novo branch e mudar para ele de uma vez é usando git checkout -b <nome\_do\_novo\_branch>.
* **Restaurar arquivos**: Você pode usar git checkout <nome\_do\_arquivo> para descartar alterações não confirmadas em um arquivo, restaurando-o para o estado da última confirmação.

Em resumo, git checkout é a ferramenta que permite que você se mova entre branches, visualize versões antigas do código e restaure arquivos.

**git log**

O comando **git log** é usado para visualizar o histórico de um repositório. Ele exibe uma lista de todas as confirmações (commits) feitas, mostrando informações detalhadas como o hash do commit, o autor, a data e a mensagem da confirmação. Você pode usar o comando git log para:

* **Visualizar o histórico completo**: Executar git log mostrará o histórico de confirmações de forma simples, do mais recente ao mais antigo.
* **Visualizar de forma resumida**: Com a opção --oneline, você pode obter um histórico mais compacto e fácil de ler, mostrando cada commit em uma única linha. Por exemplo, git log --oneline.
* **Filtrar o histórico**: Você pode filtrar as confirmações por autor, data, arquivo ou mensagem de confirmação.

Em resumo, git log é a sua ferramenta para inspecionar e entender o histórico do seu projeto. Ele é fundamental para descobrir quem fez o quê, quando e por quê.