



Git push: enviando alterações locais para o remoto!

# Git push: enviando alterações locais para o remoto!

Última atualização 31 de agosto de 2021

Para falar de git push, primeiro é necessário entender sobre **Git** e **GitHub** é um requisito importante para quem deseja [trabalhar na área de desenvolvimento](#). Muitos recrutadores deixam explícito que navegam pelo GitHub em busca de profissionais que são ativos, além de que o git possibilita que diversas pessoas trabalhem simultaneamente no mesmo projeto.

Muitas pessoas se preocupam em **decorar os comandos básicos do git**, porém, não buscam entendê-los. **Isso pode acarretar em problemas** para todo o time.

Esse também é o caso do **git push**, que envia as alterações locais para um repositório remoto. Como esse comando é de suma importância, vamos explicar tudo sobre ele, como utilizá-lo e o que fazer se houver problemas.

## Índice

### 01 | O que é o Git push e para que serve?

### 02 | Como fazer o Git push?

Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#).

Personalizar

Entendi



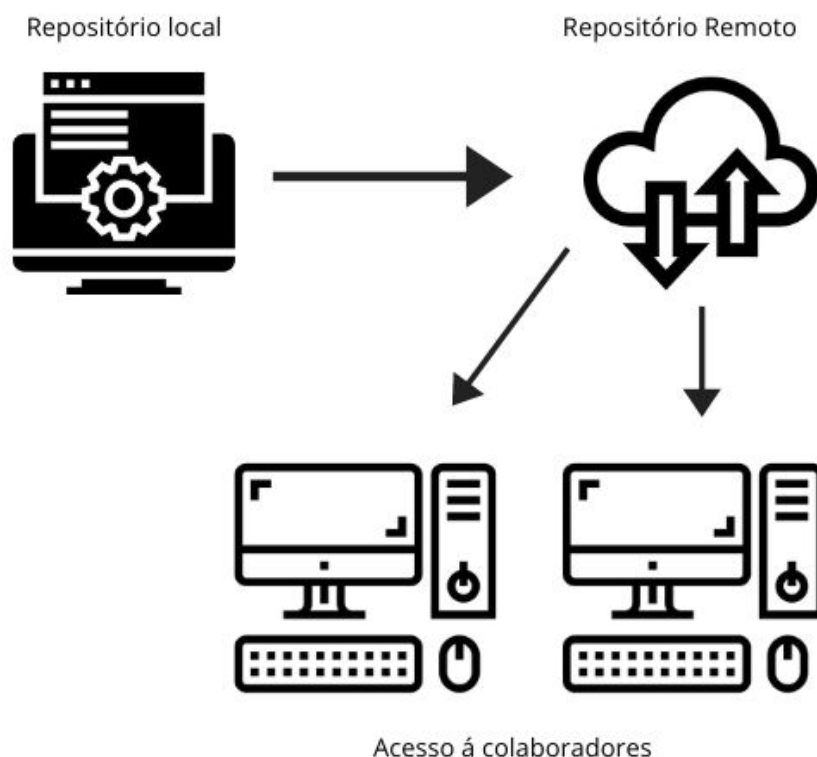
Git push: enviando alterações locais para o remoto!

## O que é o Git push e para que serve?

O Git push é um comando que **possibilita que as alterações da sua máquina local sejam enviadas para uma máquina remota**. Mas, vamos entender isso melhor:

Imagine que você está desenvolvendo um projeto com uma equipe. O projeto que vocês estão trabalhando está no GitHub, mas você vai desenvolvê-lo no seu computador. Como fazer para **"enviar" seu desenvolvimento para o GitHub?**

Para enviar suas alterações, você usará o **comando git push**. Antes dele, evidentemente, temos alguns outros comandos, mas é o git push que pegará tudo o que você fez e colocará em um local onde qualquer pessoa colaboradora possa ver e pegar suas alterações.



Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#).

Personalizar

Entendi



## Git push: enviando alterações locais para o remoto!

Para fazer o git push é necessário primeiro executar alguns passos que antecedem este comando.

Primeiro, é necessário um **terminal de linha de comando** para executar os comandos do git. No [sistema operacional](#) Windows, temos o Power Shell, [CMD](#) e o próprio terminal do Git, o **Git bash**. Caso o seu sistema operacional seja Windows, opte por utilizar o git bash. No [Linux](#), é possível utilizar o terminal Shell.

Vamos imaginar um cenário que já existe um projeto no GitHub e você precisa cloná-lo, então, o primeiro comando será:

```
git clone <linkdorepositorio>
```

Feito isso, você pode criar uma **branch** para si com o comando:

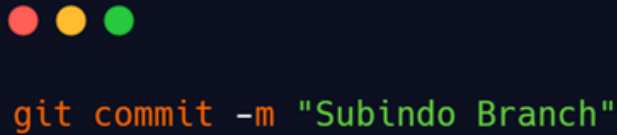
```
git checkout -b nomedabranch
```

Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#).

[Personalizar](#)[Entendi](#)

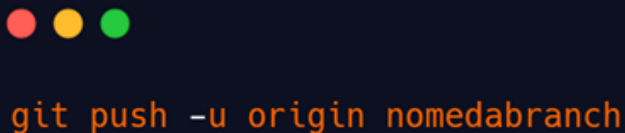


Git push: enviando alterações locais para o remoto!



```
git commit -m "Subindo Branch"
```

E agora chegamos no tão esperado **git push!** Aqui temos alguns usos específicos dele, nesse caso, **estamos apenas subindo uma branch**, logo, o comando a ser dado é o seguinte:



```
git push -u origin nomedabranch
```

Como você está subindo apenas uma branch, você pode acrescentar o `-u` como mostra a imagem, pois você está criando e “subindo” uma nova branch para o repositório remoto.

Também temos **outra sequência de comandos que antecede o git push que é extremamente importante e muito utilizada.**

Depois de criar uma nova branch, você começará o desenvolvimento do projeto e provavelmente você não fará tudo em algumas horas, então, é importante que você **envie as alterações** sem erros para o repositório remoto, pensando que talvez quando você acordar no

Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#)).

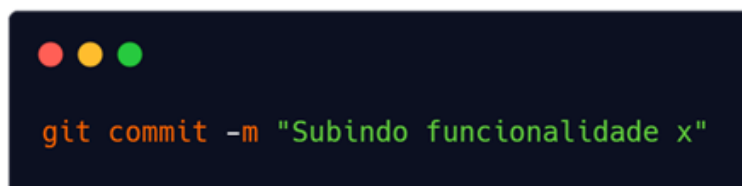
Personalizar

Entendi



## Git push: enviando alterações locais para o remoto!

Porém, é muito recomendável que a cada funcionalidade desenvolvida sem erros, você faça um **commit e envie essa alteração para o repositório remoto**. Mas, mais uma vez, antes de realizar o push, é necessário dar alguns outros comandos, como:



Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#).

[Personalizar](#)[Entendi](#)



Git push: enviando alterações locais para o remoto!

```
git push origin nomedabranh
```

Veja que agora não precisamos mais do -u, pois **estamos subindo alterações de arquivo e não apenas a branch.**

**Vamos compreender melhor esses 3 comandos tão utilizados**

## git add

Vai preparar as alterações que você fez para serem enviadas, mas aqui, elas ainda não foram enviadas.

## git commit

O git commit está “empacotando” as alterações que o comando git add preparou para serem enviadas.

## git push

Enviar de fato essas alterações.

Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#)).

Personalizar

Entendi



Git push: enviando alterações locais para o remoto!



git add .



git commit



git push

## Possíveis problemas encontrados ao realizar push e como solucioná-los?

A utilização do git push como podemos ver, é bem simples, mas algumas vezes podemos nos deparar com alguns problemas e entendemos que para quem está começando, só o fato de estar **executando comandos em um terminal já é por si só assustador**. Imagine então quando ocorre um erro! Mas, vamos ajudar você a entender e solucionar alguns desses possíveis problemas.

Algumas vezes a pessoa desenvolvedora está tão ansiosa e focada em clonar o projeto e desenvolver que se esquece de um detalhe muito importante... **trocar de branch**.

Sem perceber, a pessoa dev pode começar a desenvolver e quando vai executar o comando **git push** o terminal devolve uma mensagem de erro, falando que não é possível empurrar as alterações para o repositório remoto.

Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#).

Personalizar

Entendi



## Git push: enviando alterações locais para o remoto!

em qual branch está.

Outro problema que pode ocorrer e provavelmente pelo mesmo motivo citado acima é que você não criou uma branch e **seu projeto não está em nenhum ramo**. Para solucionar esse problema, basta executar os comandos:

- **git checkout -b [nomedabbranch]**

Com esse comando, você criará uma nova branch, e, depois como vimos acima, basta enviar a sua branch com o comando:

- **git push -u origin [nomedabbranch]**

Outro problema que pode ocorrer é **comitar alterações na branch errada**. Para solucionar esse problema, basta executar:

- **git checkout [nomedabbranch]**

Veja que este comando é diferente do comando citado acima: aqui não usamos o -b, pois não estamos querendo criar uma branch e sim ir para uma que já existe. Depois, faça:

- **git merge [main]**

Com esse comando, você estará mesclando os commits de uma branch com outra. Depois, execute novamente o comando git push para enviar suas alterações.

Por fim, corrija a branch que você enviou suas alterações sem querer com:

- **git reset --hard**

Outros problemas podem acabar surgindo fora esses abordados, mas, programar é isso: descobrir um problema novo a cada dia! Entretanto, tenha em mente que tudo tem uma solução, basta acharmos ela.

Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#).

Personalizar

Entendi





## Git push: enviando alterações locais para o remoto!

- **git push -f**

Em alguns casos, o **git push é rejeitado**, pois você pode estar compartilhando a branch com outra [pessoa desenvolvedora](#) que fez alguma alteração e você não executou o comando **git pull** para trazer as alterações feitas por ela para o seu repositório local. Portanto, ao tentar executar o comando git push, o git rejeitará a sua tentativa. Isso ocorre para evitar que você sobrescreva acidentalmente as alterações do seu colega.

Portanto, **utilize esse comando se você tiver absoluta certeza do que está fazendo** ou se você estiver desenvolvendo em uma branch só sua, pois assim não terá problemas caso seu arquivo seja sobrescrito pelo que você está tentando enviar.

- **git push -u origin [nomedabranh]**

Como falado acima, este comando é utilizado quando queremos **enviar a branch que criamos para o repositório remoto**. Isso criará um "elo" entre o seu repositório local e o repositório remoto.

- **git push -all**

Com este comando, estamos **empurrando todas as branches**, ou ramos, de uma vez só para o repositório remoto.

- **git push -tags**

Este comando enviará todas as **marcações para o repositório remoto**. Uma [tag](#) é uma marcação que aponta para pontos específicos do histórico do Git.

Como vimos, o git push é um comando de extrema importância por fazer com que de fato as alterações feitas no projeto saiam do repositório remoto e vá até o repositório local. Mas é importante conhecermos os comandos que antecedem o git push e suas derivações e utilizá-las com atenção, entendendo seu funcionamento.

Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#).

[Personalizar](#)[Entendi](#)



## Git push: enviando alterações locais para o remoto!

com foco em Java.

### Git push

Git rebase

Git commit

Git Flow

Currículo

Dúvidas

Trabalhe Conosco

Gerador de CPF

Pague só quando trabalhar

Guia HTML

Guia Javascript

Guia Soft skills

Carreira

Tecnologia

Desenvolvimento Web

Linguagens de Programação

Framework de Programação

TXN

Ferramentas

Desabilitar cookies

Política de Privacidade

Nós utilizamos cookies para memorizar suas preferências e elaborar estatísticas sobre o uso de nosso serviço, além de enviar ofertas com base na sua navegação em nosso site. Essa prática pode incluir o compartilhamento de seus dados com terceiros. Leia mais em nossa [Política de Privacidade](#)).

Personalizar

Entendi