

# PLANO DE ESTUDOS


GIT

**Lúcia Duarte**


Front-end Engineer




# BOAS PRÁTICAS DE DEPLOY




**Usar o comando GIT:** o Git é muito mais fácil voltar para uma versão anterior ou até mesmo testar uma versão específica para encontrar erros.




**Branch:** o uso de branches vai te ajudar a ter histórico de tudo que foi feito no código e deixá-lo mais organizado.



**Teste local:** usar a sua máquina para desenvolver, testar e errar deixa o desenvolvimento muito mais ágil.



**Cronograma:** é importante que outras pessoas saibam e estejam preparadas para o deploy, visto que pode ter grandes mudanças no sistema.



**Quebra do sistema:** se calame, o melhor é fazer um fix rápido e já subir a correção ou então você pode decidir fazer um rollback.

# QUANDO SUBIR UM DEPLOY?



Quando há uma menor quantidade de usuários ativos.



Com pessoas de plantão.



Em qualquer dia e horário, visto que o seu sistema tem um bom acompanhamento, sistema de alarmes e é muito bem testado.



# FORMAS DE DEPLOY



**Manual:** quando uma pessoa desenvolvedora altera um arquivo e faz um upload dessa alteração direto para produção ou que permite que dois computadores com acesso a internet troquem arquivos.



**Parcialmente automatizado:** Quando deixamos um repositório do Git atrelado a um hook que atualiza o servidor e sobe o código para produção. Para isso, basta apenas um push na branch main e o código será atualizado em produção.



**Completamente automatizado:** Com ele não acontece só o deploy automático das atualizações para o servidor, mas também o que chamamos de integração contínua. Tudo isso garante mais segurança, qualidade e eficiência para a sua aplicação.

# ESTRATÉGIAS DE DEPLOY



**Rolling:** sobe o código novo de cada serviço e substituindo a versão antiga.



**Blue-Green:** ao subir uma nova versão da aplicação (green), enquanto ela estiver subindo, as requisições vão continuar indo pra versão atual (blue). Ou seja, após subir a nova versão, você ainda vai conseguir testar a versão green.



**Canary:** consiste em colocar o novo código em produção apenas para uma parcela das pessoas usuárias.



**Multi-service:** em uma implementação de vários serviços, todos em um ambiente de destino são atualizados com vários novos serviços simultaneamente.



**Basic:** todos em um ambiente de destino são atualizados de forma simultânea com um novo serviço ou versão de artefato.



## COMO FUNCIONA O GIT?

O Git pensa em seus dados mais como uma série de instantâneos de um sistema de arquivos em miniatura. Ou seja, se os arquivos não foram alterados, o Git não armazena o arquivo novamente, apenas um link para o arquivo anterior idêntico que ele já armazenou.

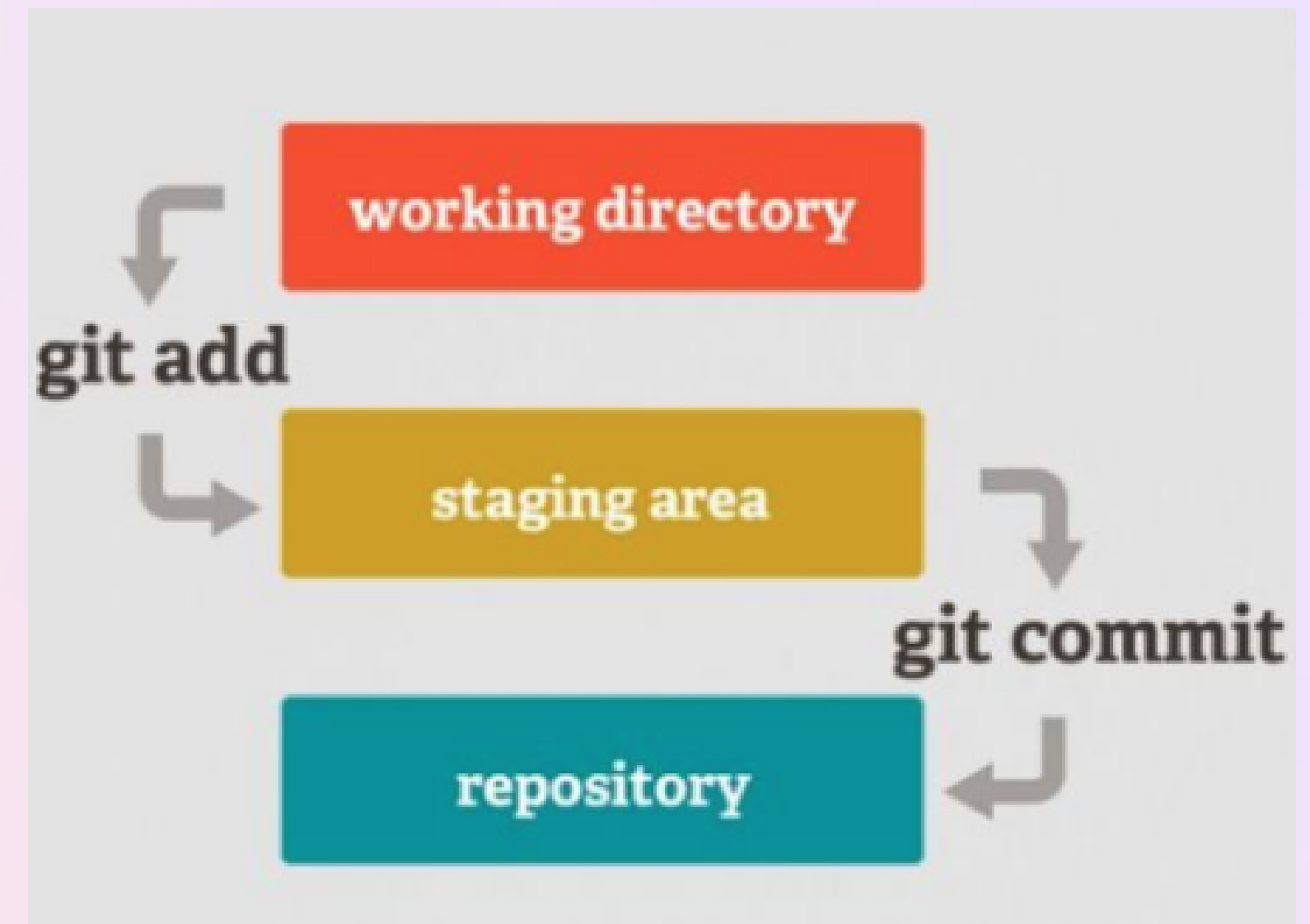
## ONDE POSSO INSTALAR?

Ao acessar o site, encontra-se versões:


**Baixe o git para OSX**  
**Baixe o git para Windows**  
**Baixe o git para Linux**

# COMO É O FLUXO DE TRABALHO GIT?

1. O primeiro deles é a Working Directory que contém a versão atual dos arquivos do projeto.
2. O segundo é o Staging (Index), que funciona como uma área temporária que diz ao git o que irá em seu próximo commit.
3. Por último, temos o Repository (HEAD) que aponta para o último commit feito no repositório remoto.




# COMANDOS DO GIT?



**Usuário:** após a instalação do git, você deve definir seu usuário e email para que esses dados sejam utilizados pelo git ao criar um commit. Para isso, execute os comandos **git config --global user.name "Seu nome"** e **git config --global user.email seu.email@email.com**



**Novo repositório:** crie uma pasta, abra-a e execute o comando **git init**



**Repositório Local:** crie uma cópia do repositório de trabalho abrindo o terminal e executando o comando **git clone /caminho/para/o/ repositório.**



# COMANDOS DO GIT?



**Verificar status:** Antes de criar uma branch, em seu terminal, deve-se verificar o status do seu código, executando o comando **git status**



**Develop:** Para garantir que você está na develop e irá criar a branch no local certo, em seu terminal execute o comando **git switch develop**




**Atualizar Código:** em seu terminal, atualize o código executando o comando **git pull**




**Criar nova branch:** crie uma nova branch executando o comando **git branch nome-da-branch**


# COMANDOS DO GIT?




**Acessar uma branch:** Para acessar uma branch já criada deve-se executar o comando **git checkout nome-da-branch**



**Apagar uma branch:** Para apagar uma branch criada localmente, execute o comando **git branch -d nome-da-branch**




**Stash:** Se existir a necessidade de realizar a troca sem fazer o commit é possível criar um stash executando o comando **git stash**




**Commits:** após selecionar as pastas da laterações realizadqas no código, commit o código executando o comando **git commit -m "comentários das alterações"**


# COMANDOS DO GIT?




**Verificar commits:** Para acessar os comentários realizados em uma branch já criada deve-se executar o comando **git branch -v**



**Empurrar o código:** Antes de empurrar as alterações feitas, de um **git pull, git status**. Após isso, empurre código alterado para o sistema com o comando **git push**



**Erro ao subir código:** Ao se enganar ou mudar de ideia ao subir o código, pode-se voltar atrás utilizando o comando **git checkout -- <arquivo>**



**Erro ao subir código:** Para remover todas as alterações e commits locais, recupere o histórico mais recente do servido execute o comando **git fetch origin**



# PERGUNTAS



Qual horário a empresa costuma subir o deploy?



Quais ferramentas são utilizadas para o deploy na empresa?



O deploy é feito somente para o ambiente de produção? ou para o QA também?



Qual método de deploy utilizamos na Neomed?

# PÁGINA DE RECURSOS



<https://pythonacademy.com.br/blog/fluxo-de-trabalho-no-git>



[https://rogerdudler.github.io/git-guide/index.pt\\_BR.html](https://rogerdudler.github.io/git-guide/index.pt_BR.html)



<https://git-scm.com/docs>



<https://blog.betrybe.com/tecnologia/deploy/>