UNIPLAN

CENTRO UNIVERSITÁRIO PLANALTO DO DISTRITO FEDERAL

GITHUB:
https://github.com/Cristianomeneses2008/git

Palestra: Cristiano Meneses
Atualizado - 2025

Git é um sistema de controle de versões distribuído, usado principalmente no desenvolvimento de software, mas pode ser usado para registrar o histórico de edições de qualquer tipo de arquivo (Exemplo: alguns livros digitais são disponibilizados no GitHub e escrito aos poucos publicamente). O Git foi inicialmente projetado e desenvolvido por Linus Torvalds para o desenvolvimento do kernel Linux, mas foi adotado por muitos outros projetos.

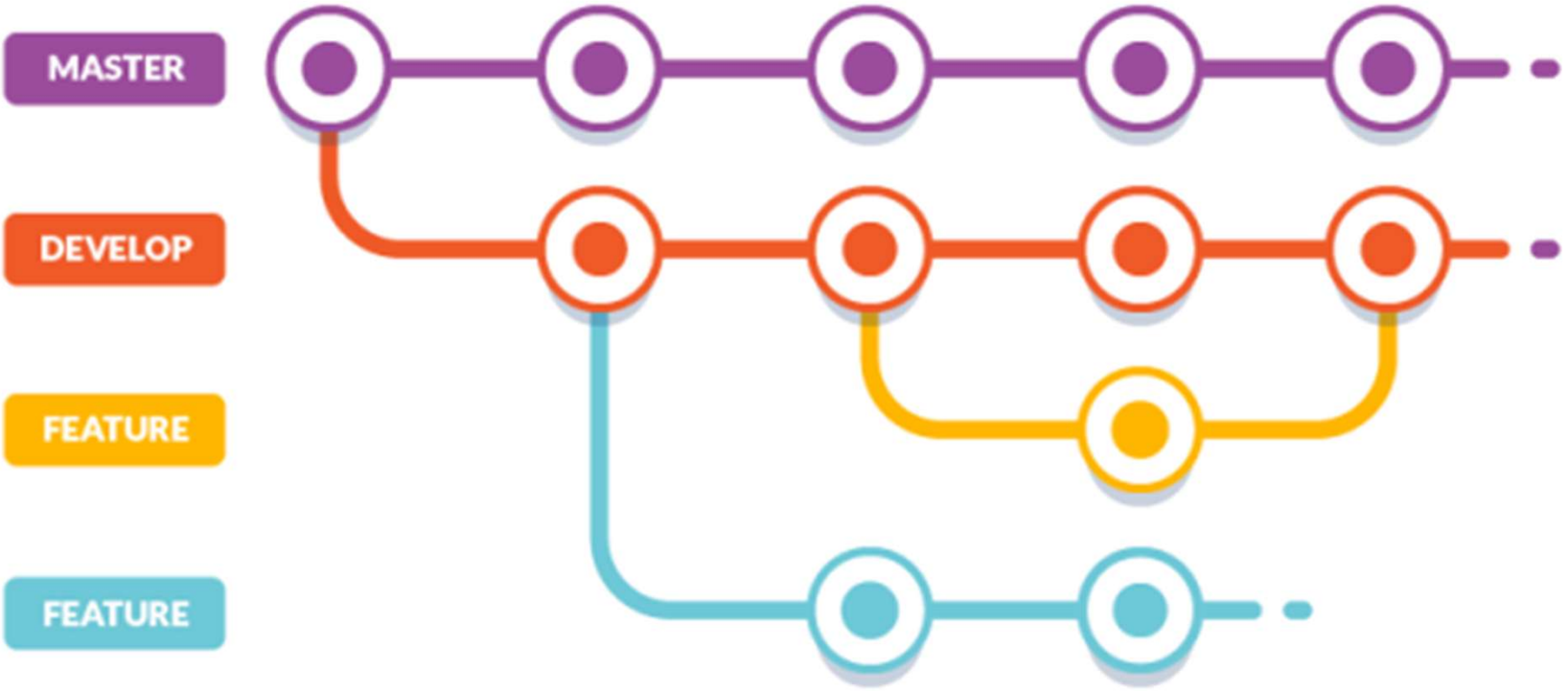
Cada diretório de trabalho do Git é um repositório com um histórico completo e habilidade total de acompanhamento das revisões, não dependente de acesso a uma rede ou a um servidor central. O Git também facilita a reprodutibilidade científica em uma ampla gama de disciplinas, da ecologia à bioinformática, arqueologia à zoologia.[4]

O Git é um software livre, distribuído sob os termos da versão 2 da GNU General Public License. Sua manutenção é atualmente supervisionada por Junio Hamano.

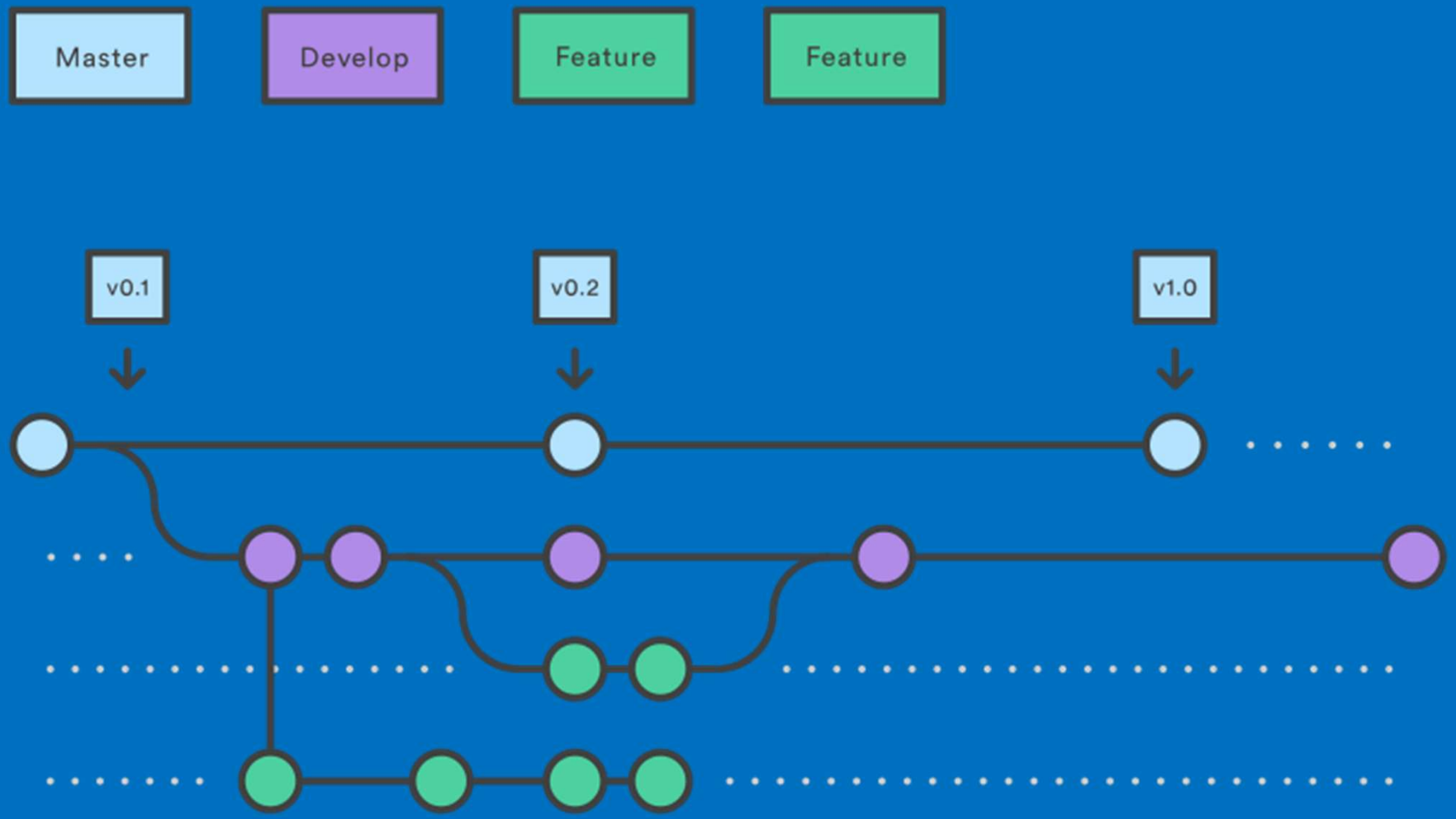
No **Scrum** é essencial ter um produto liberável ao final de cada Sprint. Para poder lançar com tanta frequência você precisa ter uma boa estratégia de ramificação. Felizmente existem ferramentas que permitem gerenciar o software da melhor maneira possível tanto em termos de ferramentas quanto de processos.

GitFlow é um conjunto de diretrizes que sugere a melhor maneira de gerenciar filiais com Git; Seguindo as orientações é possível ter sempre versões de software estáveis que incluam um conjunto de recursos conhecidos.

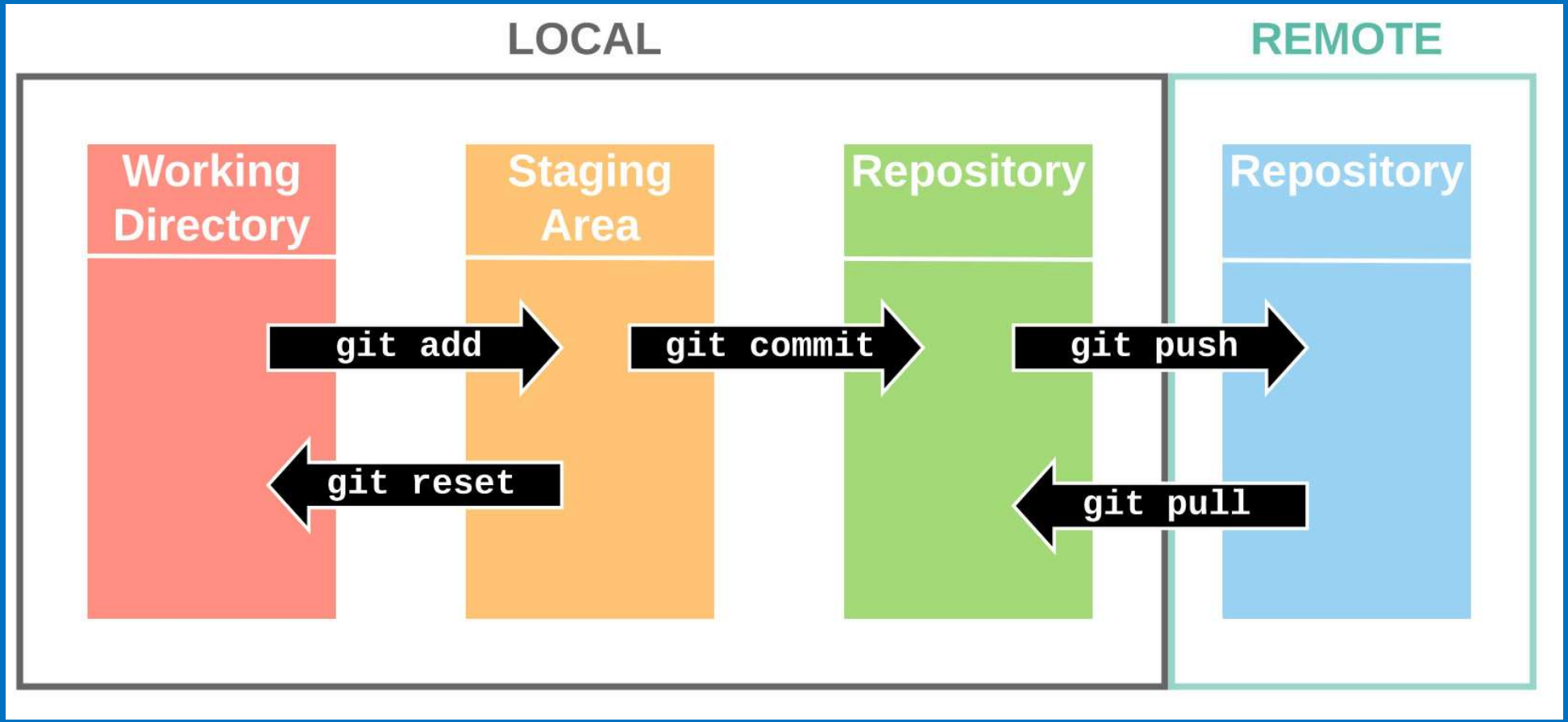
Desenho



Gitflow



Entendendo o **fluxo do versionamento.**



Configurando o GIT
Os comandos mostrados neste artigo foram testados no Windows. Uma vez instalado, abra o terminal do Windows e configure o git com os dados de sua conta do Github

Para isso, utilize os comandos **git config**

A primeira coisa a fazer depois de instalar o Git é definir o seu nome de usuário e endereço de e-mail. Isso é importante porque todos os commits no Git utilizam essas informações

git config --global user.name "Seu nome de usuário"

git config --global user.email seuemail@seuservidor.com

Para copiar um repositório do GitHub para a sua máquina, utilize o comando

Git clone <Endereço remoto do repositório>

Para transformar a pasta do projeto num repositório do GIT, ou para ativar o repositório existente, dentro da pasta do projeto, utilize o comando:

Git init.

Reverta as alterações especificadas pelo quarto último commit no HEAD e crie um novo commit com as alterações revertidas.
Reverta as alterações especificadas pelo quarto último commit no HEAD e crie um novo commit com as alterações revertidas.
Reverta as alterações especificadas pelo quarto último commit no HEAD e crie um novo commit com as alterações revertidas.

git add nome_do_arquivo

Adiciona o arquivo na área de index. Para adicionar todos os arquivos que sofreram alteração (ou para copiar todos os arquivos caso seja a primeira vez), utilize **git add .**

git status

Com este comando você consegue ver quais arquivos estão fora do controle, quais foram modificados e estão esperando por uma descrição de modificação. Ele também mostra em qual **branch** você se encontra no momento.

git reset HEAD nome_do_arquivo

Volta ao estágio anterior do adicionamento. Se você infomar **git reset** sem nenhum parâmetro após fazer o git add ele desfaz a operação

Depois de adicionar os arquivos, chegou a hora de guardar. Para isso, executamos o comando commit.

git reset --soft HEAD~1

git revert HEAD~3

Reverta as alterações especificadas pelo quarto último commit no HEAD e crie um novo commit com as alterações revertidas.

Filtros

Para filtrar commits pelo nome do autor, podemos fazer desta forma:

git log --autor=nome-autor

Se for necessário filtrar por data, existem duas formas.

git log --after=" YYYY-MM-DD"

git log --before=" YYYY-MM-DD“

git push - u origin --all

Este comando copia todos os branches do repositório local e envia para o repositório remote

git log

Mostra todos os commits que você fez

git branch

Exemplos de como trabalhar com branches

Crie um novo branch chamado “develop" e selecione-o usando

git checkout –b develop

Retorne para o branch master usando

git checkout master

Para mesclar alterações de um branch com o branch que estiver ativo, use

git merge nome_do_branch

Instalações para facilitar:
Visual Studio Code
TortoiseGit

https://git-scm.com/docs

GIT + GITHUB CHEAT SHEET

BRANCH

git branch new-branch
Cria com o nome new-branch
git checkout -b new-branch
Criar e mudar o código para o branch new-branch
git checkout new-branch
Muda o código atual e a referência para a branch new-branch
git checkout main
Muda o código para a branch principal: main
git branch -d new-branch
Deleta a new-branch
git branch
Lista os branchs criados
git branch -v
Listar os branches criados com os logs de commit
git push origin new-branch
Criando um branch remoto com o mesmo nome
git push origin new-branch:new-branch
Criando um branch remoto com nome diferente
git pull origin main
Download de todos os os arquivos do repositório remoto para o local
git fetch origin
Download de todas as branchs remotas.
git checkout -b new-branch origin/new-branch
Baixar um branch remoto para edição
git merge new-branch
Realiza o merge entre os branchs, ou seja, junta o ramo na arvore principal

STASH

git stash
Cria um stash, salva temporariamente as modificações
git stash list
Lista os stashes criados
git stash apply
Volta ao último stash
git stash apply stash@{2}
Volta ao stash com índice 2
git stash branch meu_branch
Criar um branch a partir de um stash.

GITHUB

git --version
Verifica a versão instalada.
git config --global user.name "Seu Nome"
Configura seu nome.
git config --global user.email "Seu Email"
Configura seu e-mail.
git remote add origin git@github.com:"Usuário_github"/"repositorio"
Envie o arquivo ao repositório remoto.
git remote -v
Verifica os arquivos enviados
git remote add origin https://github.com/nome/repositorio.git
Sincroniza o repositório local com o remoto

git init
Inicia um repositório local
git clone ssh://git@github.com/[username]/[repository-name].git
Cria uma cópia do repositório local

INICIANDO UM REPOSITÓRIO

LOG

git log
Exibe histórico dos ultimos commits
git log -p -3
Exibe histórico com diff dos ultimos 3 commits
git log -- <caminho_do_arquivo>
Exibir histório no caminho_do_arquivo
git log --pretty=oneline
Exibe histórico de commits com informações resumidas em uma linha. Existem outros tipos.
git log --diff-filter=M -- <caminho_do_arquivo>
Exibe histórico de modificações de um arquivo
git log --author=usuario
Exibir histório de um determinado autor

CHERRY-PIC

git cherry-pick <commit-id>
Copia as informações desse commit
git cherry-pick A^..B
Copia todos os commits entre o commit A e o commit B, inclusive A e B.
git cherry-pick A..B
Copia todos os commits entre o commit A e o commit B, excluindo A.

ARQUIVOS

git add .
Adiciona todos os arquivos ou diretórios modificados;
git add meu_programa.py
Adiciona somente o arquivo meu_programa.py;
git add meu_diretório
Adiciona somente o diretorio meu_diretorio;
git add -f meu_programa_gitignore.py
Adiciona um arquivo que está no .gitignore;

ADICIONAR

git rm meu_arquivo.txt
Remover arquivo
git rm -r diretorio
Remover diretório

REMOVER

git commit meu_programa.py
Comitar um arquivo
git commit file1.txt file2.txt
Comitar vários arquivos
git commit meuarquivo.txt -m "minha mensagem de commit"
Comitar informando mensagem

COMMIT

MUDANÇAS

gitk
Mostra as modificações totais do projeto
git diff
Modificações antes de enviar as modificações para o repositório remoto (commit);
git status
Estado dos arquivos/diretório;
git commit --amend -m "Minha nova mensagem"
Alterando mensagens de commit já realizado
git rebase -i HEAD~3
Alterar últimos commits, modificando as mensagens.

SALVA PRA QUANDO

FIM

Fontes de Pesquisa:

https://git-scm.com/
https://blog.dbins.com.br/guia-dos-principais-comandos-do-git
https://www.docker.com/resources/what-container/
https://www.pasqualelangella.com/2020/03/23/scrum-e-la-branching-strategy/