



PROGRAMAÇÃO
DO ZERO com
**Renato
Gava**

AULA BÔNUS

Git e GitHub

Você vai aprender na aula de hoje

- O que é e como funciona o Git
- Comandos Básicos do Git
- O que é GitHub e qual a diferença entre Git e GitHub

GIT

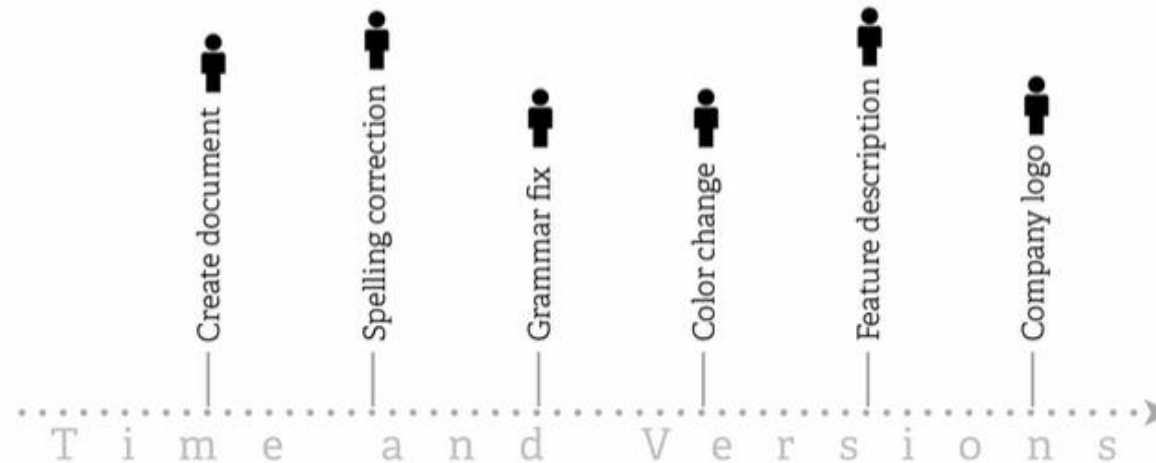
O que é Git?

- Git é um sistema de Controle de versão.
- Ele grava as mudanças realizadas em um arquivo ao longo do tempo, para que você possa revisitar versões específicas depois.
- Além disso, ele permite a colaboração, ou seja, que mais de uma pessoa visualize e edite o mesmo arquivo.

Histórico de alterações de um arquivo

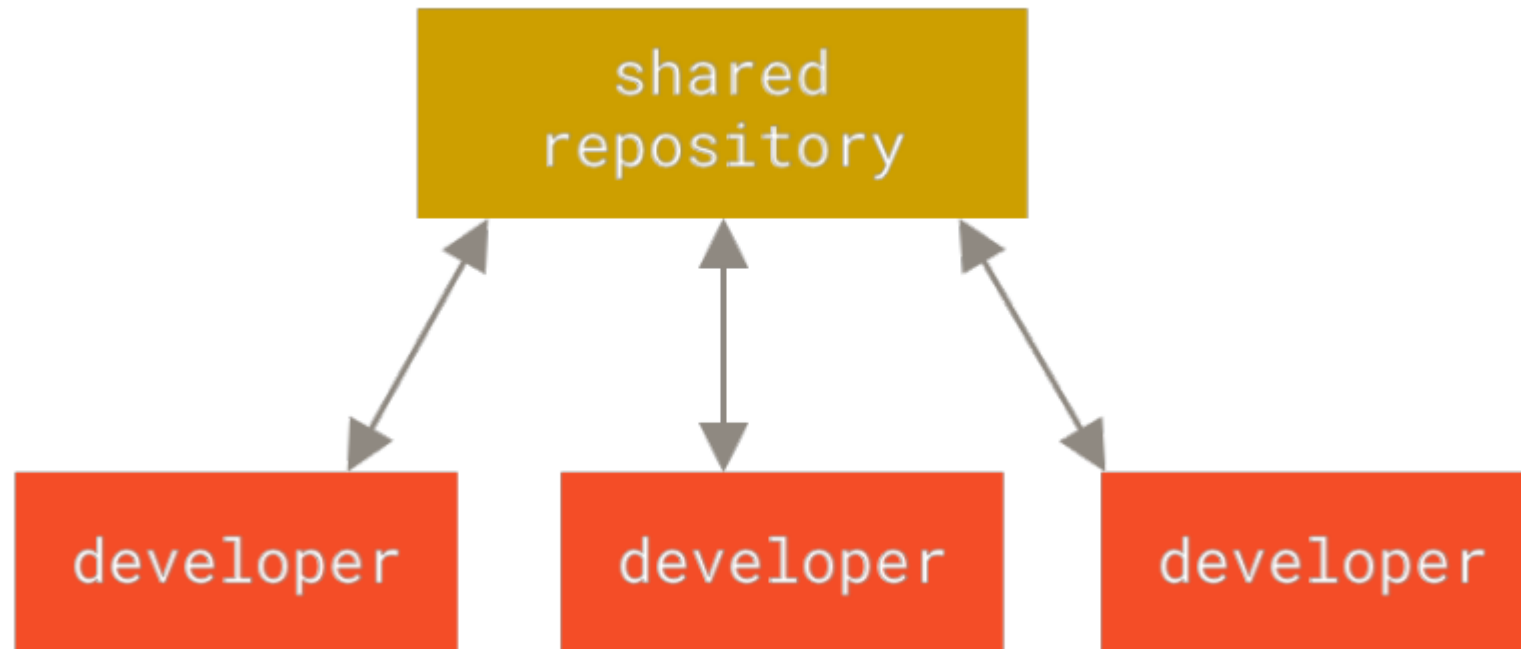
- Você vai poder identificar: o que foi modificado, quando e porque

History Tracking



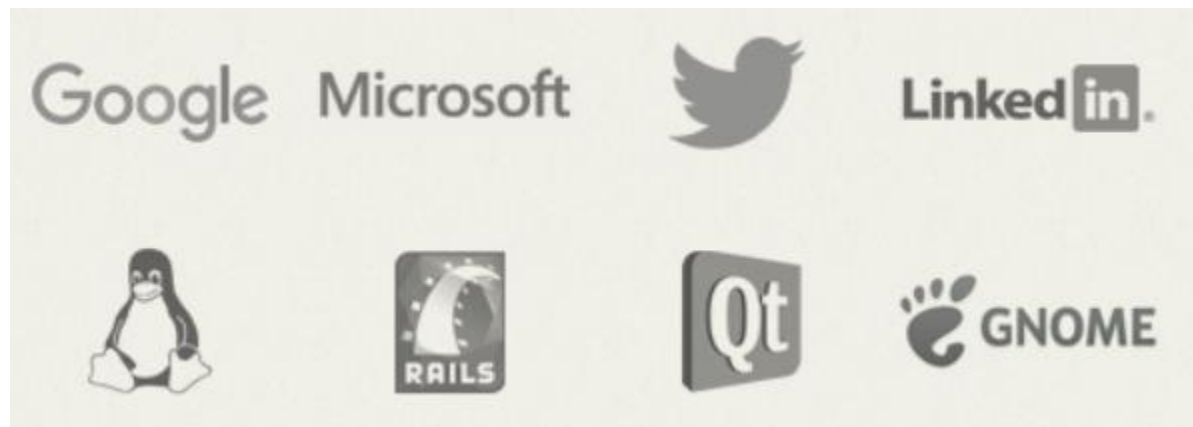
Colaboração

- Mais de uma pessoa editar o mesmo arquivo ao mesmo tempo
- Rastrear quem fez a alteração, quando e porque



Porquê Git?

- É rápido e moderno
- Provê histórico de modificações nos arquivos
- Permite mudanças colaborativas nos arquivos
- Fácil de usar
- Muito utilizado no mercado



Instalando Ferramentas

- Acessar e instalar o Git: next, next, finish: <https://git-scm.com/>
- Acessar e instalar o Notepad ++: <https://notepad-plus-plus.org/>
- Acessar e o Visual Studio Community 2022:
<https://visualstudio.microsoft.com/pt-br/downloads/>

Linha de Comando vs GUI

- Existe duas maneiras de usar o Git:
 - via linha de comando
 - via GUI (Graphical User Interface)

Configuração Inicial

- O comando **git config** permite gerenciar as configurações iniciais do Git, que são armazenadas no arquivo **gitconfig**.
- O Git tem 3 níveis de configuração:
 - **System**: configuração mais geral possível, nível do sistema operacional. Comando `--system`
 - **Global**: configuração do usuário do Windows. Comando `--global`
 - **Local**: configuração de um repositório específico. Não precisa de comando.
- Use o comando para listar todas as configurações: **git config --list --show-origin**

Configuração Inicial

Identidade

```
git config --global user.name "Renato Gava"
```

```
git config --global user.email renatogava2@live.com
```

Nome da Branch Padrão

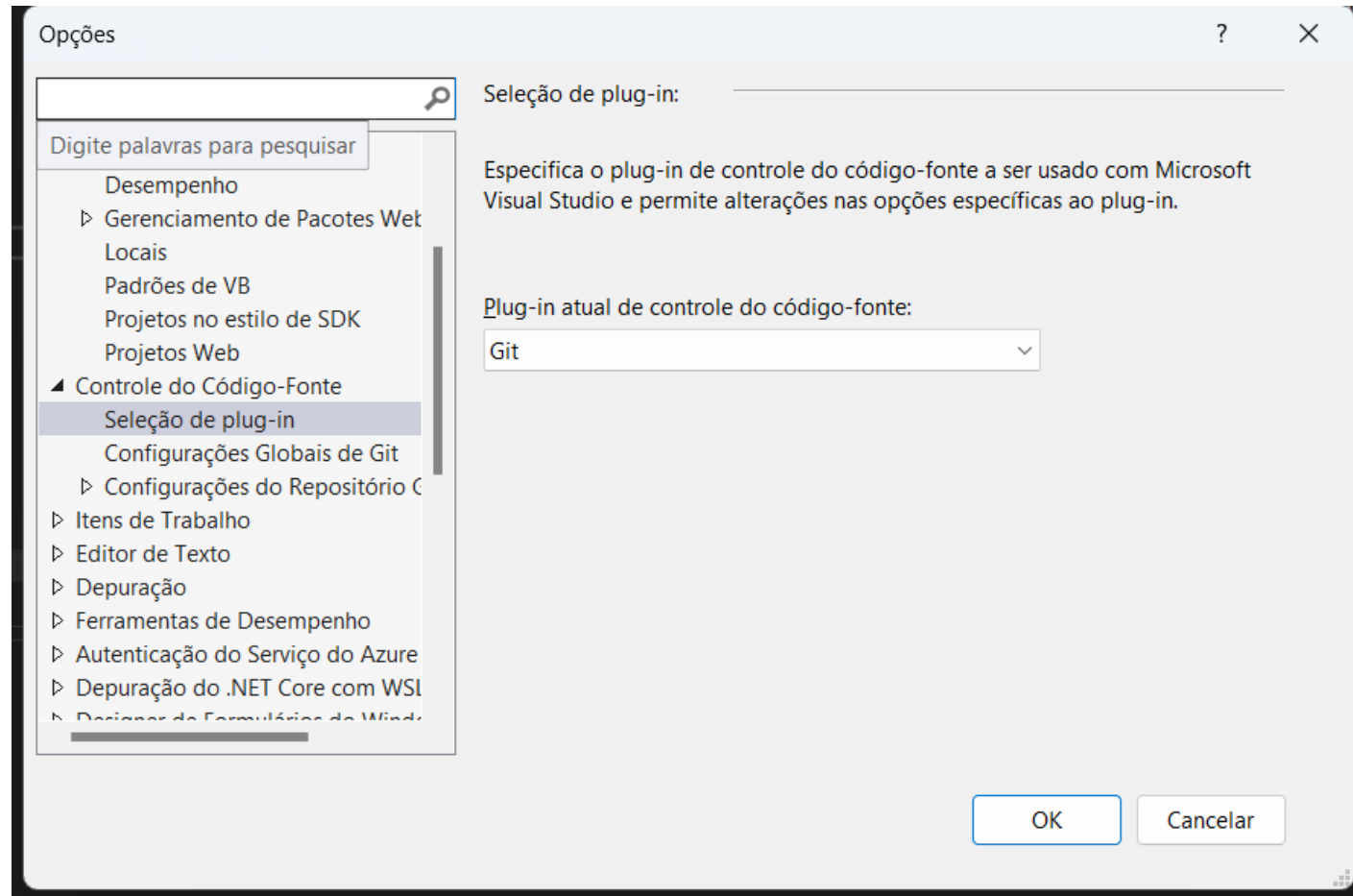
```
git config --global init.defaultBranch main
```

Editor Padrão

```
git config --global core.editor "'C:/Program  
Files/Notepad++/notepad++.exe' -multiInst -notabbar -nosession -  
noPlugin"
```

Configurando Git no Visual Studio

- Ferramentas -> Opções



Criando Repositório

Criando um repositório Git com **git init**:

- Criar pasta: C:\GitProjects
- Executar Comandos:

```
cd C:\GitProjects
```

```
git init meu-primeiro-repositorio
```

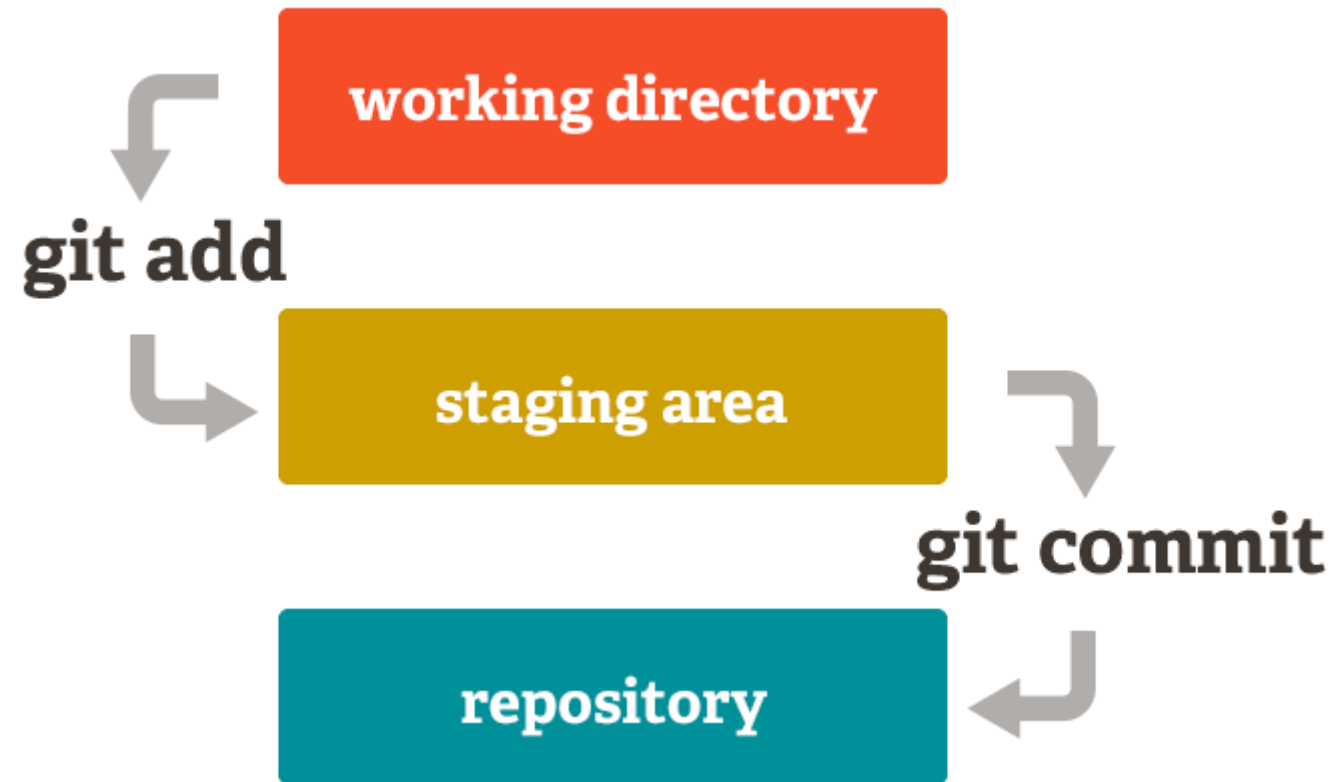
Verificando Status e Adicionando Arquivos

Verificando o status de alterações:

Rodar comando dentro da pasta: **git status**:

- Criar arquivo1.html, com estrutura básica HTML
- Rodar novamente comando **git status**
- Rodar comando **git add arquivo1.html**
- Rodar novamente comando **git status**
- Ao rodar git add, você envia o arquivo para **Staged** (Significa que ele está preparado para ser Enviado).

Ciclo de vida do Status



Alterando Arquivos

Adicione mais um parágrafo no arquivo1.html

Rodar comando **git status** novamente

Rodar comando **git add** arquivo1.html

Rodar comando **git status** novamente

O comando git add adiciona as mudanças realizadas para Staged

Ignorando Arquivos

Muitas vezes você não vai querer controlar arquivos, como os arquivos autogerados do Visual Studio, por exemplo.

Criar arquivo **.gitignore**

Adicionar no arquivo:

ignorando a pasta do Visual Studio

.vs/

Rodar comando: **git status**

Rodar comando: **git add .gitignore**

Rodar comando: **git status**

Identificando as mudanças feitas

Identificando as alterações feitas em um arquivo local, antes de ir pra Staged:

Adicione mais um parágrafo no arquivo1.html

Rodar comando: **git diff**

Rodar comando: **git add** arquivo1.html

Fazendo Commit

Enviando as alterações feitas de Staged (Committer)

- Rodar comando: **git commit -m** “Adicionando arquivo1.html e gitignore”

Pronto, você fez seu primeiro Commit!

Pulando Stage e fazendo Commit Direto

Você pode pular a etapa de adicionar arquivo para stage, e fazer commit direto:

- Adicionar mais um parágrafo
- Rodar comando `git status`
- Rodar comando **`git commit -a -m`** “adicionando mais um parágrafo”

Removendo Arquivos

- Adicione novo arquivo2.html
- Rodar comando **git add** arquivo2.html
- Rodar comando **git commit -m** “adicionando arquivo2.html”
- Rodar comando **git rm** arquivo2.html
- Rodar comando **git commit -m** “removendo arquivo2.html”

Visualizando o Histórico de Commits

- Rodar comando **git log**
- Pressionar a tecla **q** para sair

Descartando uma alteração Local

- Adicione mais um parágrafo no arquivo1.html
- Rodar comando **git restore** arquivo1.html

Descartando uma alteração Staged

- Adicione mais um parágrafo no arquivo1.html
- Rodar comando **git status**
- Rodar comando **git diff**
- Rodar comando **git add** arquivo1.html
- Rodar comando **git restore --staged** arquivo1.html
- Rodar comando **git status**

Revertendo um Commit

- A qualquer momento vc pode querer desfazer algum commit.
- Adicionar mais um parágrafo
- Rodar comando **git commit -a -m** “adicionando mais um parágrafo”
- Rodar comando **git revert ID**, onde ID é o identificador do commit a ser revertido, que deve ser obtido com o comando git log.

Recapitulando

Setup Inicial

- `git config --list --show-origin`
- `git config --global user.name "Renato Gava"`
- `git config --global user.email renatogava2@live.com`
- `git config --global core.editor "C:/Program Files/Notepad++/notepad++.exe' -multinst -notabbar -nosession -noPlugin"`
- `git config --global init.defaultBranch main`
- `git init meu-primeiro-repositorio`
- `git status`
- `git add arquivo1.html`
- `git diff`
- `git commit -m "comentario"`
- `git rm arquivo2.html`
- `git log`
- `git revert ID`
- `git restore --staged`
- `git restore`

GITHUB

Remote

- Para colaborar com outras pessoas, é preciso que seu repositório esteja hospedado em algum lugar na internet.
- Rodar comando: **git remote**. Você verá que não há nenhum repositório remoto ainda.

GitHub

- GitHub é uma plataforma de **hospedagem de código-fonte** e arquivos com controle de versão usando o Git.
- Ele permite que programadores contribuam em projetos privados e/ou Open Source de qualquer lugar do mundo.

Criando conta e primeiro repositório

- Acesse e crie sua conta: <https://github.com/>
- Crie um novo repositório chamado meu-primeiro-repositório

Adicionando Remote e fazendo Push

- Rodar comando: **git remote add origin**
<https://github.com/renatogava/meu-primeiro-repositorio.git>
- Rodar comando: **git push -u origin main**
- Rodar comando novamente: **git remote**
- Rodar comando **git status**
- Validar no GitHub

Fazendo Push de Alterações

- Adicione mais um parágrafo no arquivo1.html
- Rodar comando **git commit -a -m** “adicionando mais um parágrafo”
- Rodar comando novamente: **git push**
- Validar no GitHub

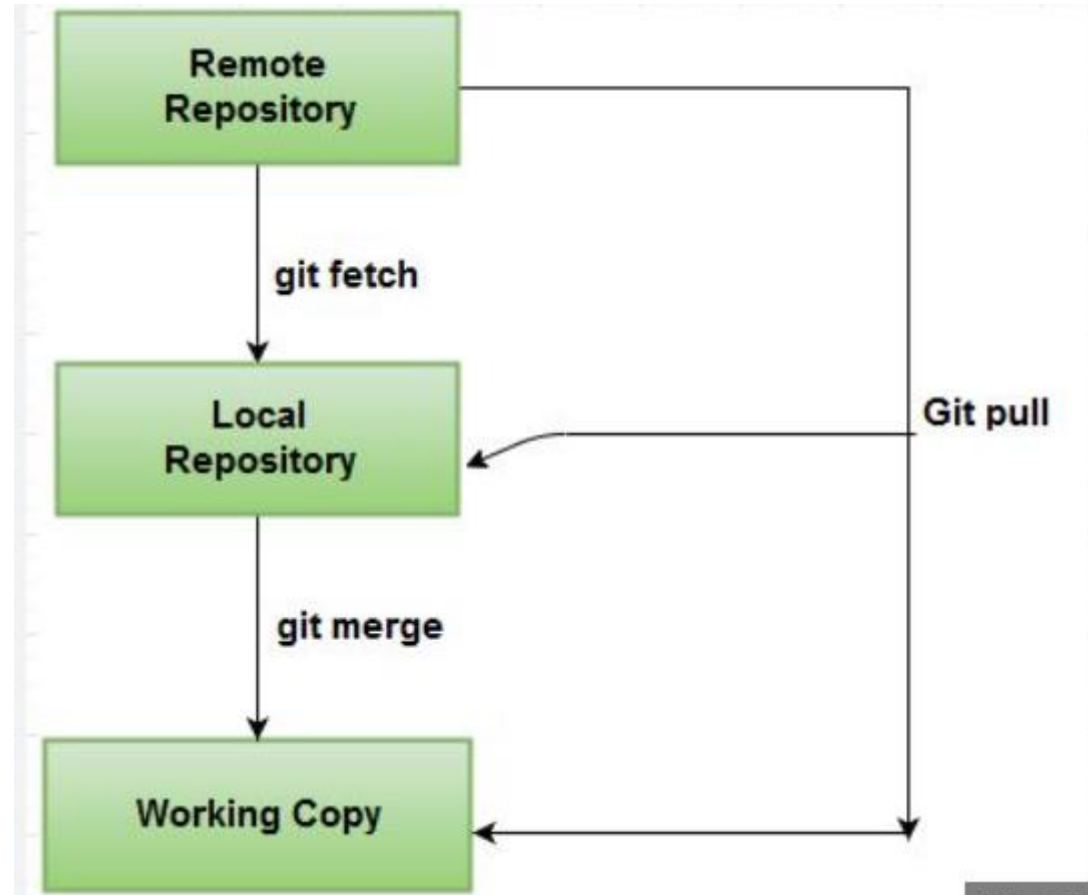
Fazendo Fetch de Alterações

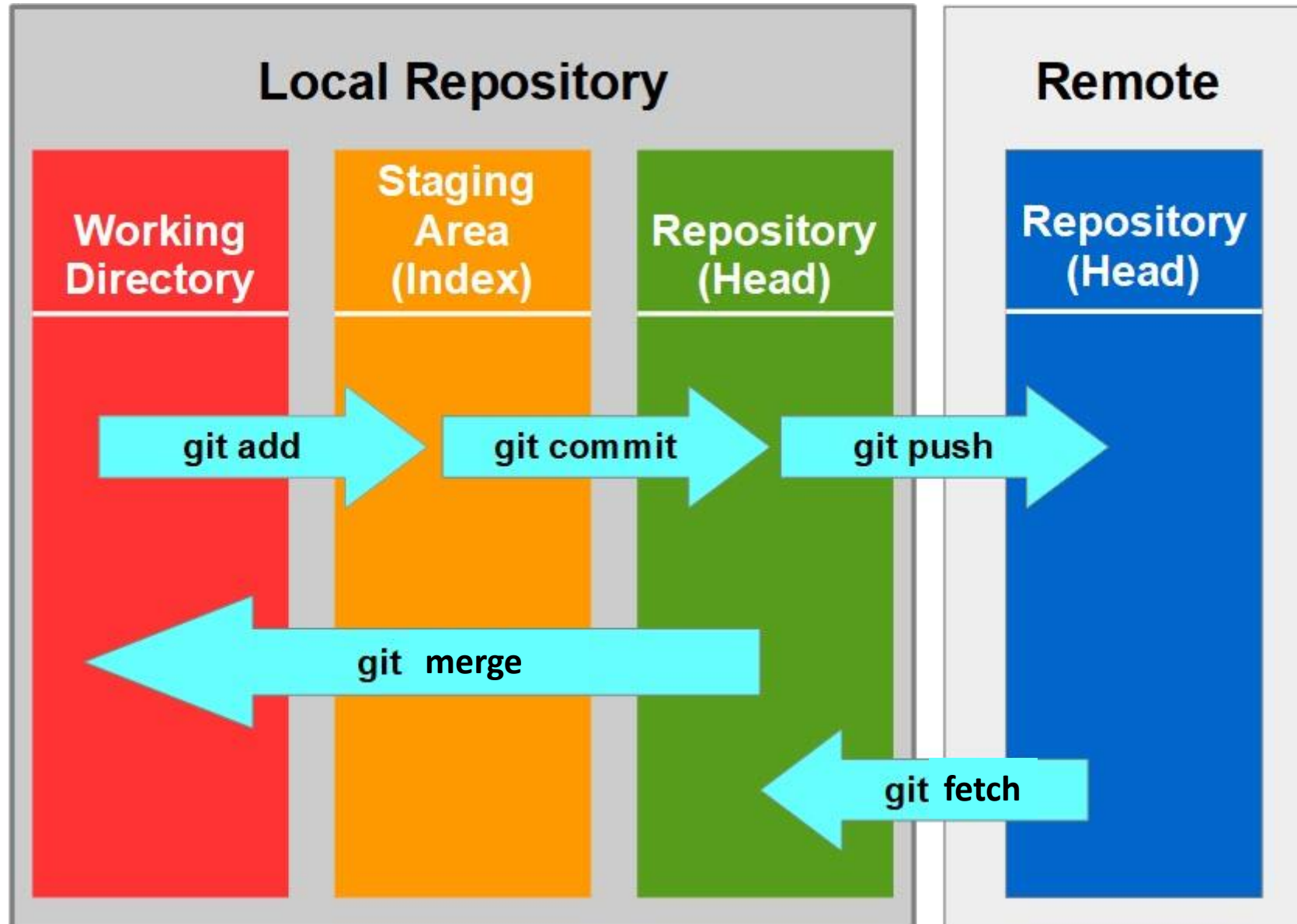
- Adicione mais um parágrafo no arquivo1.html usando o GitHub
- Rodar o comando: **git fetch**
- Rodar o comando: **git status**
- Rodar o comando: **git merge**

Fazendo Pull de Alterações

- Adicione mais um parágrafo no arquivo1.html usando o GitHub
- Rodar o comando: **git pull**
- Rodar o comando: **git status**

Diferença entre Fetch e Pull



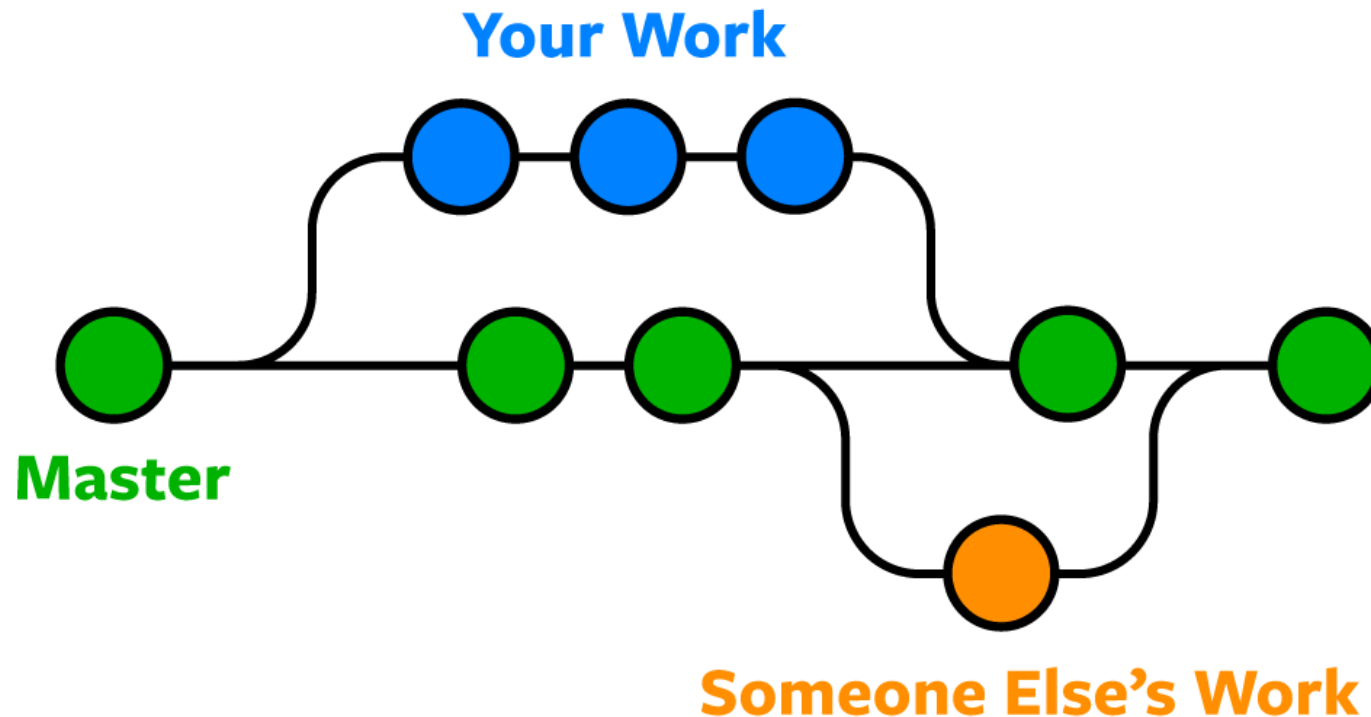


Clonando um repositório existente

- Apague a pasta meu-primeiro-repositório
- Abrir o Prompt no diretório C:\GitProjects
- Rodar comando: **git clone** <https://github.com/renatogava/meu-primeiro-repositorio.git>

Branches

- Branching significa que você diverge da linha principal de desenvolvimento e continua a trabalhar sem bagunçar o main.
- Essa é uma das principais funcionalidades do Git.



Criando uma nova Branch

- Rodar o comando: **git branch** paragrafos-coloridos
- Definir a nova branch como a atual (HEAD): **git checkout** paragrafos-coloridos

Alterando arquivo na nova branch

- Adicionar estilo no arquivo1.html para colorir os parágrafos:

```
<style>
```

```
p {
```

```
    color: #ff0000
```

```
}
```

```
</style>
```

- Rodar comando: **git commit -a -m “colorindo os paragrafos”**
- Rodar comando: **git push -u origin paragrafos-coloridos**

Alterando arquivo na branch main

- Rodar comando: **git checkout** main
- Adicionar novo parágrafo no arquivo1.html
- Rodar comando: **git commit** -a -m “adicionando novo parágrafo”
- Rodar comando: **git push**

Verificando as alterações em cada branch

- Rodar comando: **git checkout** paragrafos-coloridos
- Verificar o arquivo1.html
- Rodar comando: **git checkout** main
- Verificar o arquivo1.html

Fazendo merge

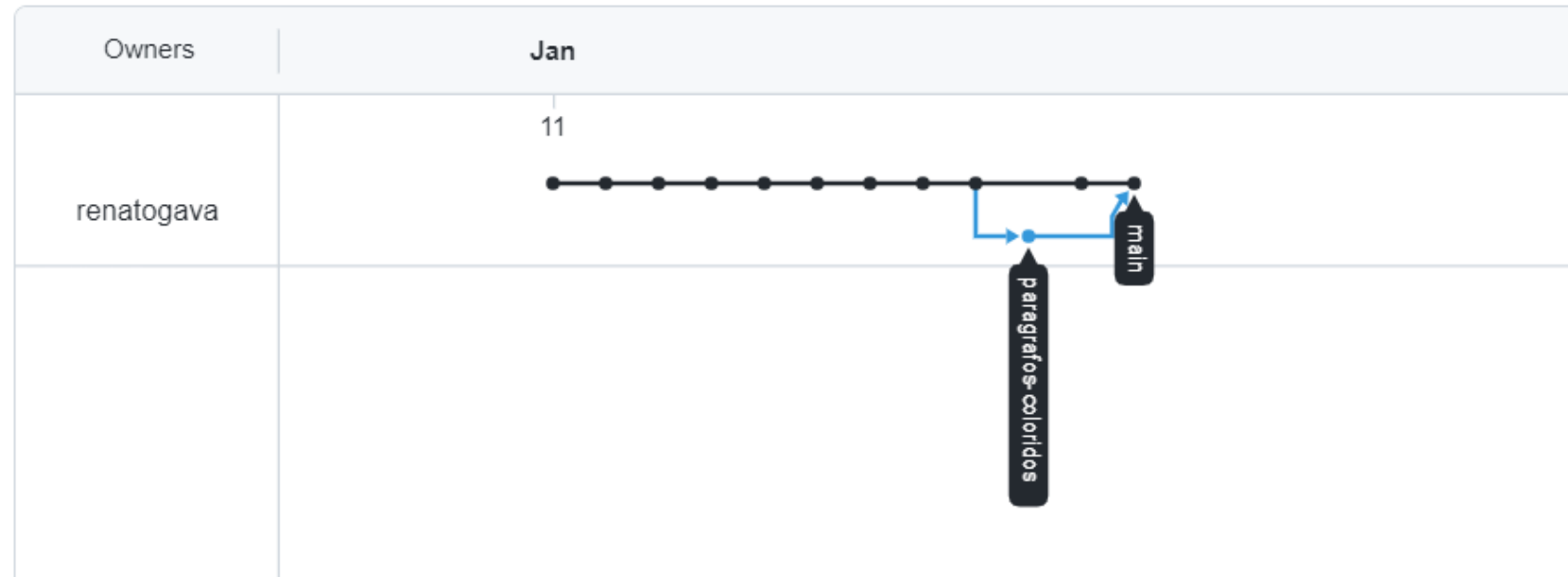
- Rodar comando: **git merge** paragrafos-coloridos
- Rodar comando: **git push**

Visualizando no GitHub

[Contributors](#)[Community](#)[Community Standards](#)[Traffic](#)[Commits](#)[Code frequency](#)[Dependency graph](#)[Network](#)[Forks](#)

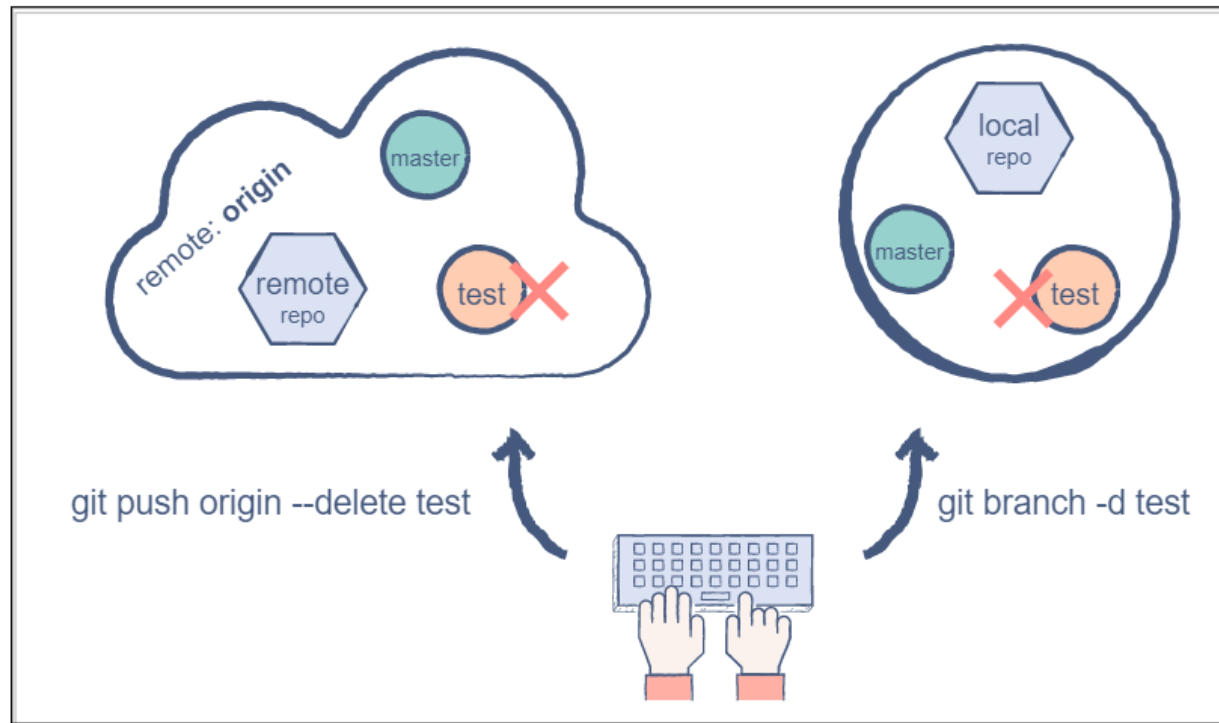
Network graph

Timeline of the most recent commits to this repository and its network ordered by most recently pushed to.



Apagando Branch

- Rodar comando: **git branch -d** paragrafos-coloridos
- Rodar comando: **git push origin --delete** paragrafos-coloridos



Resolvendo Conflitos

- Rodar comando: **git branch** colorindo-titulo
- Rodar comando: **git checkout** colorindo-titulo
- Alterar arquivo1.html

```
h1 {  
  color: #0026ff  
}
```

- Rodar comando: **git commit -a -m** “colorindo titulo”
- Rodar comando: **git push**
- Rodar comando: **git checkout** main
- Alterar arquivo1.html

```
p:first-letter {  
  color: #0026ff  
}
```

- Rodar comando: **git commit -a -m** “primeira letra do paragrafo”
- Rodar comando: **git push**

Resolvendo Conflitos

- Rodar comando: **git merge** colorindo-titulo
- Corrija o arquivo manualmente:

```
<<<<<< HEAD
      p:first-letter {
          color: #0026ff
      }
=====
      h1 {
          color: #4cff00
      }
>>>>>> colorindo-titulo
```

- Rodar comando: **git add** arquivo1.html
- Rodar comando: **git commit**
- Rodar comando: **git push**

Apagando Branch

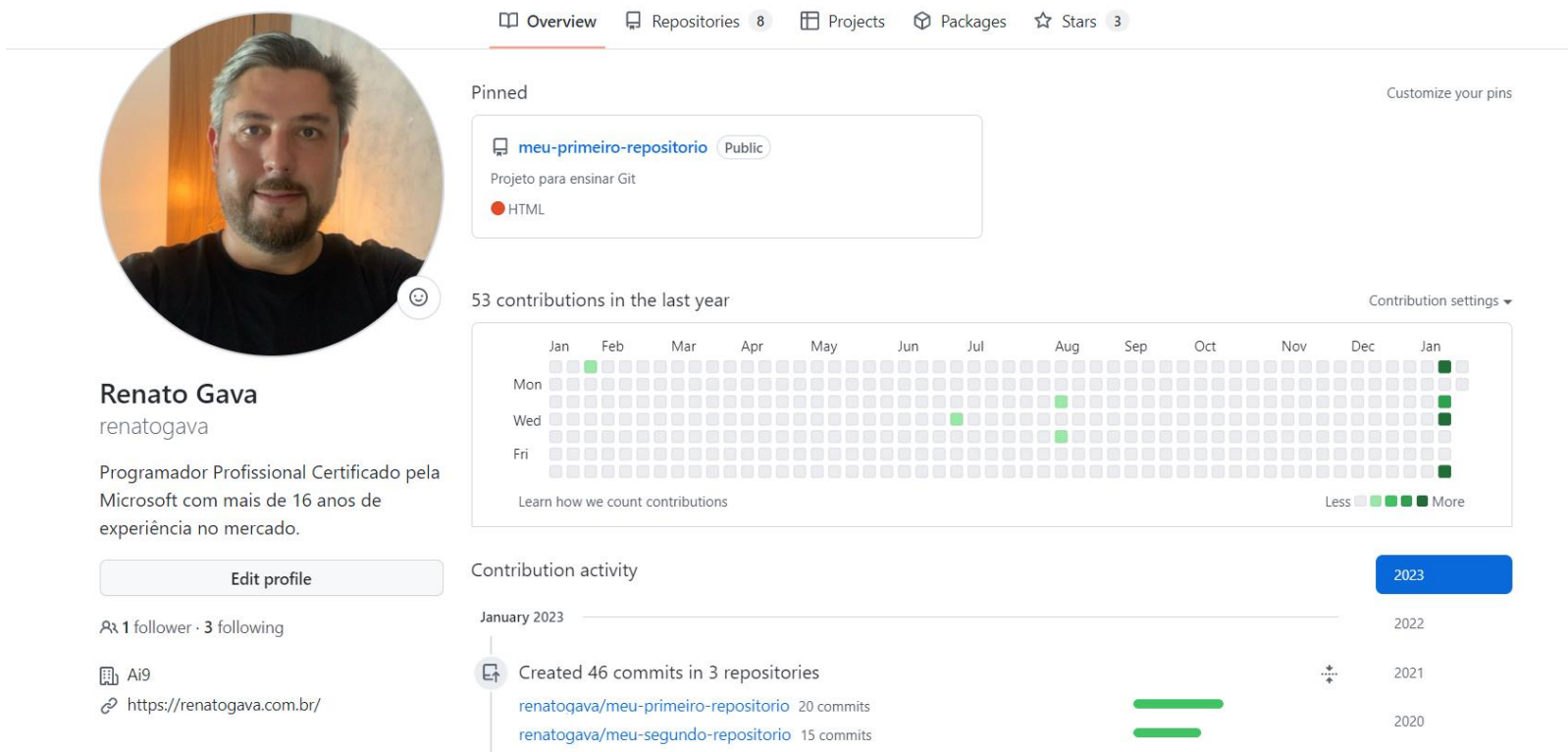
- Rodar comando: **git branch -d** colorindo-titulo
- Rodar comando: **git push origin --delete** colorindo-titulo

Recapitulando

- **git remote** add origin url-repositorio
- **git push** -u origin main
- **git fetch**
- **git pull**
- **git clone** url-repositorio
- **git branch** nome-da-Branch
- **git merge** nome-da-branch
- **git checkout** nome-da-branch
- **git branch** -d nome-da-branch
- **git push** origin --delete nome-da-branch

Monte seu Portfólio


- Crie projetos e vá adicionando repositórios
- Preencha seus dados
- Compartilhe o link do seu perfil com recrutadores, mostre em entrevistas



Overview Repositories 8 Projects Packages Stars 3

Pinned

Customize your pins

 [meu-primeiro-repositorio](#) Public

Projeto para ensinar Git

HTML

53 contributions in the last year

Contribution settings

Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec Jan

Mon

Wed

Fri

Learn how we count contributions

Less More

Contribution activity

January 2023

Created 46 commits in 3 repositories

[renatogava/meu-primeiro-repositorio](#) 20 commits

[renatogava/meu-segundo-repositorio](#) 15 commits

2023

2022

2021

2020