



# Projeto Integrador I

# Versionamento do Código

Git & Github

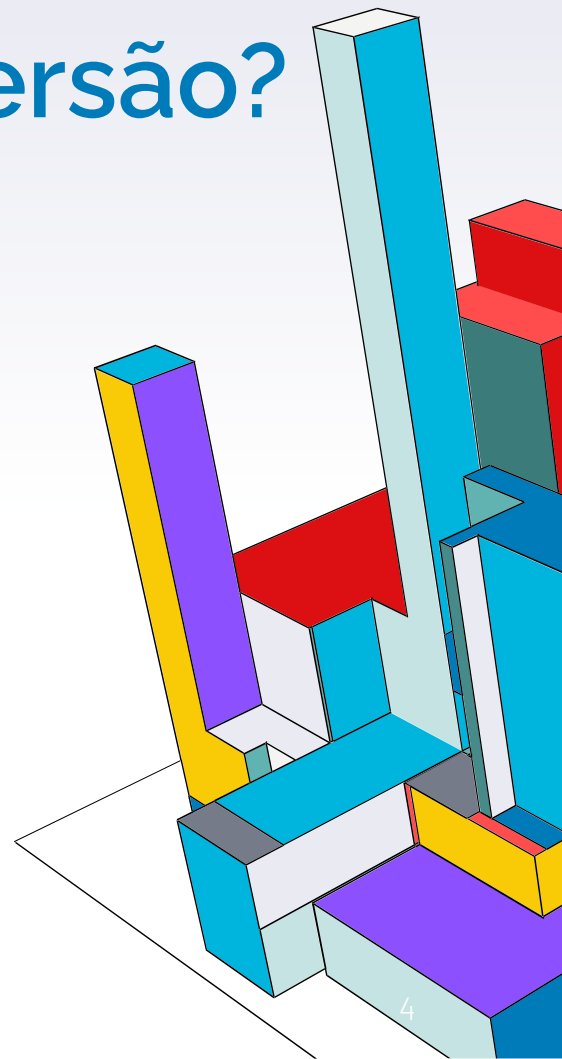


# Controle de Versão

Sistema com a finalidade de gerenciar diferentes versões de um mesmo documento, ou seja, responsável por versionar os arquivos do projeto. Os mais conhecidos são o SVN (Apache), CVS (Concurrent Versions System) e o GIT.



# Porque ter o controle de versão?



# GIT

**Git** é um poderoso e sofisticado sistema para controle de versão distribuído. Ganhar entendimento de suas funcionalidades abre para os desenvolvedores uma nova e libertadora abordagem à manutenção de código.

O Git foi inicialmente projetado e desenvolvido por **Linus Torvalds** para o desenvolvimento do kernel Linux.



# GITHUB

O GitHub é uma rede social de desenvolvedores. A primeira parte do nome, “Git”, é por causa da utilização do sistema de controle de versão e a segunda parte, “Hub”, tem a ver com a conexão entre profissionais de programação de qualquer lugar do mundo.



# Introdução ao GIT



## Fluxo de Trabalho no GIT

Seus repositórios locais consistem em três "árvores" mantidas pelo git.

A primeira delas é sua **Working Directory** que contém os arquivos vigentes.

A segunda é a **Index** que funciona como uma área temporária (stage).

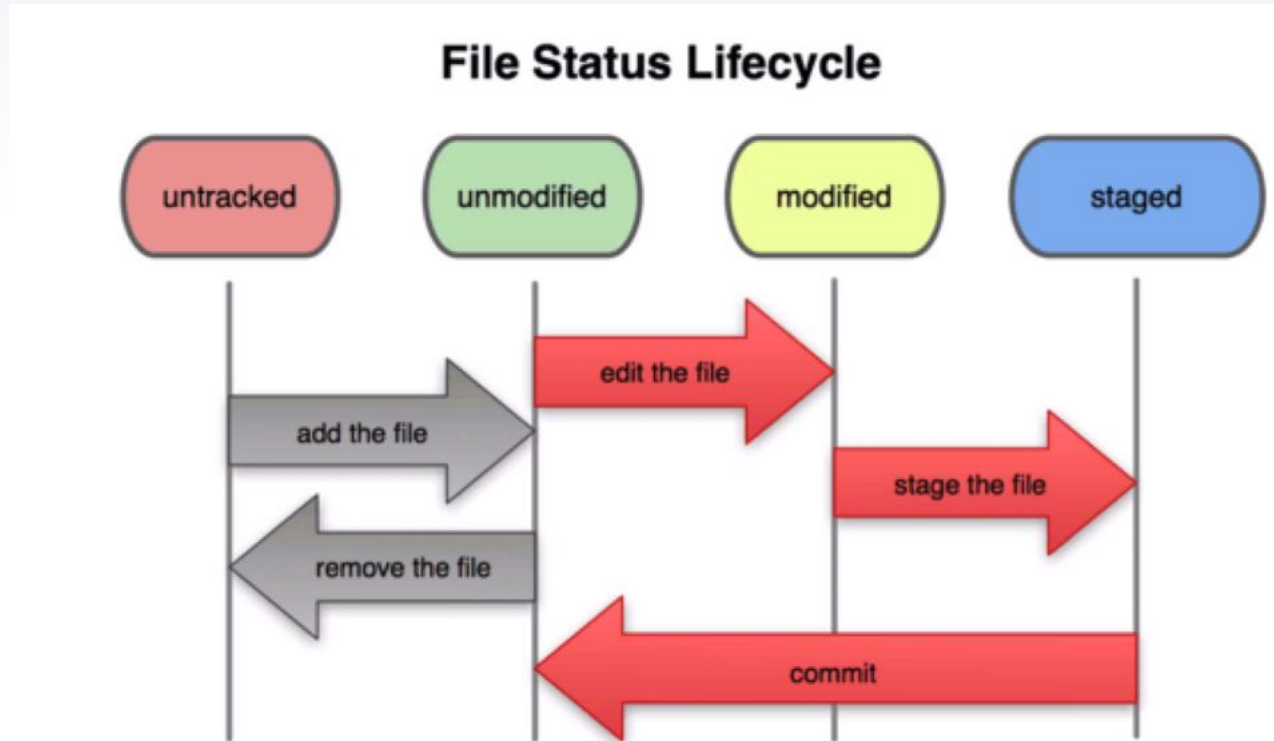
A terceira é a **HEAD** que aponta para o último commit (confirmação) que você fez.



# Introdução ao GIT



## Ciclo de vida dos arquivos





# Introdução ao GIT



## Configurando nome e endereço de e-mail


Se você nunca usou git antes, o primeiro passo é configurar seu nome e endereço de email. Execute os seguintes comandos para fazer com que o git saiba seu nome e endereço de e-mail.

### EXECUTE:

```
git config --global user.name "Seu Nome"  
git config --global user.email "seu\_email@dominio.com"
```

Para conferir as informações digitadas, informe:

```
git config user.name  
git config user.email
```

 **Dica:** Se você estiver usando um computador **público**, remova a opção `--global` e execute o comando dentro da pasta do seu projeto.

```
git config user.name "Seu Nome"  
git config user.email "seu\_email@dominio.com"
```

# Introdução ao GIT

## Adicionando a chave SSH



A chave SSH no GitHub tem a função de autenticar sua identidade ao interagir com os repositórios hospedados na plataforma usando o protocolo SSH (*Secure Shell*). Quando você adiciona uma chave SSH à sua conta do GitHub e configura-a em seu computador, você pode se autenticar automaticamente ao empurrar (**push**) ou puxar (**pull**) alterações de repositórios sem precisar digitar seu nome de usuário e senha a cada vez.

Basicamente, a chave SSH serve como uma forma segura de identificar e verificar que você tem permissão para acessar e manipular os repositórios do GitHub sem expor suas credenciais de login. Isso proporciona conveniência e segurança, especialmente ao trabalhar com repositórios remotos de forma frequente.

### 1. EXECUTE para gerar a chave:

```
ssh-keygen -t ed25519 -C "seu\_email@dominio.com"
```

### 2. EXECUTE para copiar a chave gerada:

```
cat < ~/.ssh/id_ed25519.pub
```

### 4. EXECUTE para testar a conexão:

```
ssh -T git@github.com
```

### 3. Adicionar a chave SSH ao GitHub:

- Acesse sua conta GitHub (foto) e vá para Settings > SSH and GPG keys.
- Clique em "New SSH key" e cole a chave SSH pública que você copiou anteriormente.
- Dê um nome descritivo para a chave, se desejar, e clique em "Add SSH key".

# Introdução ao GIT



## Ciclo de Vida do Primeiro Exemplo

Abra o GIT Bash e digite:

```
mkdir projeto --cria a pasta
cd projeto --posiciona na pasta
touch manual.txt
notepad manual.txt --adicione algo ao arquivo
cat manual.txt --para verificar o conteúdo
git init --iniciar o repositório vazio
git status --untracked
git add manual.txt
git status --staged
--adicione uma nova linha no arquivo manual.txt
git add manual.txt
git status
git commit -m "Add manual.txt"
git status
```

# Introdução ao GIT



## Visualizando as diferenças

EXECUTE:

```
git diff
```

Edite o arquivo manual.txt, **altere** uma linha e **adicione** conteúdo em uma nova linha.

EXECUTE:

```
git diff
```

```
git diff --name-only
```

```
git show --é utilizado para visualizar informações sobre um commit específico no repositório Git
```

💡 **Dica:** Para sair do git show, pressione Q (Quit)

# Introdução ao GIT



## Adicione uma nova página ao repositório

Agora vamos adicionar a página **clientes.html** ao repositório.

### EXECUTE:

```
cd projeto --se ainda não estiver  
touch clientes.txt  
--insira um conteúdo inicial no arquivo  
git add clientes.txt (ou git add .)  
git commit -m "Add clientes"
```

# Introdução ao GIT



## Confira o status do repositório

Use o comando `git status` para checar o estado atual do repositório.

**EXECUTE:**

```
git status
```

## Modifique o conteúdo do arquivo

Efetue a adição de uma nova linha no arquivo **manual.txt** e verifique novamente o status

**EXECUTE:**

```
git status
```

O primeiro aspecto importante aqui é que o git sabe que o arquivo **manual.txt** foi modificado, mas essas modificações ainda não sofreram **commit** para o repositório.

Outro aspecto é que a mensagem de status oferece dicas sobre o que fazer em seguida. Se você quiser adicionar essas modificações para o repositório, use `git add`.

Para desfazer as modificações use `git checkout`. Exemplo: `git checkout -- manual.txt`

# Introdução ao GIT



## Efetue alterações e um novo commit

### EXECUTE:

```
--Edite o arquivo manual.txt  
git add .  
git commit -m "Novas alterações"
```

## Visualize o histórico das alterações

Obtenha uma lista das modificações feitas

### EXECUTE:

```
git log Ou  
git log --pretty=oneline Ou  
git log --pretty=format:"%h %ad | %s%d [%an]" --graph --date=short
```

### Dica: Crie um Alias para o comando

```
git config --global alias.hist 'log --pretty=format:"%h %ad | %s%d [%an]" --graph --date=short'
```

```
git hist
```

# Introdução ao GIT



## Adicionando um repositório remoto

Acesse o [github](https://github.com) e crie um novo repositório na sua conta.

### EXECUTE:

```
git remote add origin <caminho do repositório>  
git remote --visualize o nome do repositório adicionado  
git remote -v --endereço do repositório  
git remote set-url origin <caminho do repositório> --para alterar o repositório
```

Agora vamos enviar todos os arquivos do nosso diretório local para o repositório remoto.

### EXECUTE:

```
git push -u origin master (apenas na primeira vez)  
git push origin master (a cada commit que você deseja publicar no repositório remoto)
```

A operação inversa é o pull (Incorpora as alterações de um repositório remoto no branch atual)

```
git pull origin develop
```



# Introdução ao GIT



## Clonando um repositório remoto

Visite o [github](https://github.com) e vá até a página do repositório desejado. Ex:

<https://github.com/ricardoleme/exemplo-react-hook-soma-material-ui> e clique no botão **Clone or Download**.

### EXECUTE:

```
git clone <caminho-repositorio> <nome-da-pasta>
```

```
git clone https://github.com/ricardoleme/exemplo-react-hook-soma-material-ui.git soma
```

# Introdução ao GIT



## Criando o arquivo .gitignore

O .gitignore é um arquivo simples que lista o que o Git deve ignorar quando você estiver trabalhando no repositório, de forma que seja possível evitar a adição de arquivos indesejados no repositório sem dificuldades, tornando os arquivos indesejados totalmente invisíveis ao Git.

### EXECUTE:

```
# Exemplo de Comentário  
/diretorio/*  
*.extensão  
nomedoarquivo.extensão
```

### Dica:

Para criar um .gitignore que ignore corretamente todos os arquivos de uma determinada linguagem ou tecnologia, visite:

<http://gitignore.io>

# Introdução ao GIT



## Criando um README.md

O README.md é um arquivo Markdown que fornece uma descrição detalhada do seu repositório. Ele geralmente é a primeira coisa que os visitantes veem quando acessam seu repositório no GitHub. Veja um exemplo:

```
# Meu Projeto Incrível
```

```
Bem-vindo ao Meu Projeto Incrível! Este é um projeto que faz algo incrível e útil.
```

```
## Visão Geral
```

```
Este projeto resolve o problema X fornecendo Y. Ele é desenvolvido usando [Tecnologia A](https://link-para-tecnologia-a) e [Tecnologia B](https://link-para-tecnologia-b).
```

```
## Instalação
```

```
Para instalar o projeto, siga estas etapas:
```

```
1. Clone o repositório: `git clone https://github.com/seu-usuario/meu-projeto-incrivel.git`
```

```
2. Navegue até o diretório do projeto: `cd meu-projeto-incrivel`
```

```
3. Execute o script de instalação: `./install.sh`
```

```
## Como Usar
```

```
Após a instalação, você pode usar o projeto da seguinte maneira:
```

```
## Licença
```

```
Este projeto está sob a licença MIT. Veja o arquivo [LICENSE](LICENSE) para mais detalhes.
```

```
## Créditos
```

```
Autor: Seu Nome
```

# Resumo Comandos Básicos GIT

git config	Define configurações globais. Ex: <code>git config --global user.email email@dominio.com</code>
git init	Inicializa um novo repositório
git add	Adiciona um novo arquivo ao índice. Ex: <code>git add teste.txt</code> ou <code>git add *</code>
git clone	Clona um repositório remoto. Ex: <code>git clone https://github.com/ricardoleme/tarefas-react.git</code>
git commit	Efetua o commit das alterações. Ex: <code>git commit -m "coloque sua mensagem aqui"</code>
git status	Exibe a lista de arquivos alterados juntamente com os arquivos que ainda não foram adicionados ou confirmados.
git push	Envia as alterações para o repositório remoto. <code>git push origin master</code>
git remote	Conectar-se a um repositório remoto. (quando não foi feito o clone) Ex: <code>git remote add origin &lt;servidor&gt;</code>

# Resumo Comandos Básicos GIT

git status	Exibe o status do arquivo informado
git diff	Exibe todas as diferenças encontradas no arquivo
git log	Exibe o log de todas as alterações
git pull	Incorpora as alterações de um repositório remoto no branch atual. <code>git pull origin develop</code>
git push	Transfere commits a partir do seu repositório local para um repositório remoto <code>git push origin master</code>

 **Dica:** Para obter a lista completa dos comandos GIT, visite: <https://comandosgit.github.io/>

That's all folks!  
Let's code.

