

iris

Laboratório de
Inovação e Dados



CEARÁ
GOVERNO DO ESTADO

Oficina Zero

Introdução ao Git



UNIVERSIDADE
FEDERAL DO CEARÁ



GREat

Problemas no Desenvolvimento Colaborativo

- Compartilhamento de Código Difícil
- Versões de Código Desatualizadas e Confusas
- Conflitos e Perda de Trabalho
- Zero Histórico e Controle

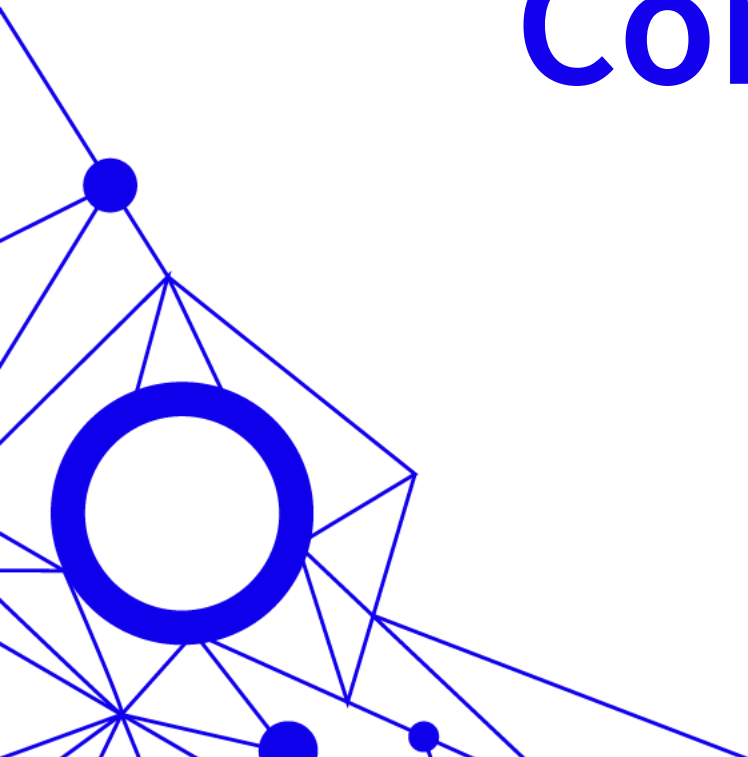
O que é Git?

- **Git** é um sistema de controle de versão distribuído.
- Criado por Linus Torvalds em 2005.
- Projetado para ser rápido, simples e suportar desenvolvimento distribuído.

Vantagens do Git

1. Versionamento de Arquivos
2. Colaboração
3. Distribuição

Configuração do Git



Instalação do Git

- Linux:

```
1 sudo apt-get install git
```

- Mac:

```
1 brew install git
```

- Windows: **Usar WSL...** e aí cai no Linux 💪

Configuração Inicial

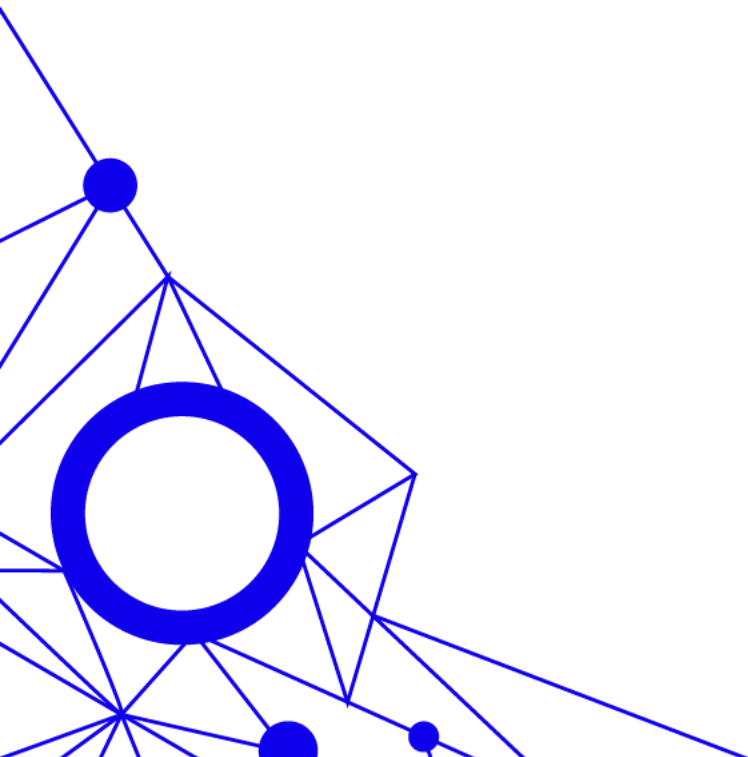
1. Configurar Nome e Email:

```
1 git config --global user.name "Seu Nome"  
2 git config --global user.email "seu.email@exemplo.com"
```

2. Arquivos de Configuração:

- Global: `~/.gitconfig`
- Local: `.git/config`

Deep Dive



Conceitos Básicos

- **Repositório:** Conjunto de arquivos e seu histórico de versões
- **Commit:** Grupo de alterações salvas em um determinado momento
- **Branch:** Linha independente de desenvolvimento
- **Merge:** Combinação de alterações de diferentes branches

Fluxo de Trabalho Git

Veja o diagrama abaixo para entender o fluxo típico de trabalho com o Git:



Comandos Básicos

1. Criar e Inicializar Repositório:

```
1 mkdir projeto-git  
2 cd projeto-git  
3 git init
```

2. Adicionar Arquivo:

```
1 touch README.md  
2 git add README.md  
3 git commit -m "docs: add README"
```

Branches

- Linhas independentes de desenvolvimento
- Permitem trabalhar em funcionalidades isoladas
- Facilitam experimentos sem afetar o código principal
- Criar Branch:

```
1 git branch nova-feature
```

- Trocar para Branch:

```
1 git checkout nova-feature
```

Mesclagem (Merge)

- Combina mudanças de diferentes branches
- Integra novas funcionalidades ao código principal
- Pode requerer resolução de conflitos

Repositórios Remotos

Github e Gitlab



Vantagens

- Permitem colaboração entre equipes
- Servem como **backup em nuvem** do seu código
- Exemplos: GitHub e GitLab

Clonando um Repositório

```
1 git clone https://github.com/usuario/repo.git  
2 cd repo
```


Arquivo `.gitignore`

- Ignorar Arquivos “Temporários”:

```
1 *.log  
2 node_modules/  
3 .env
```

Importância do README .md

- Documentação Essencial:
 - Descrição do projeto.
 - Instruções de instalação e uso.
 - Como contribuir.



Obrigado!

Perguntas?

