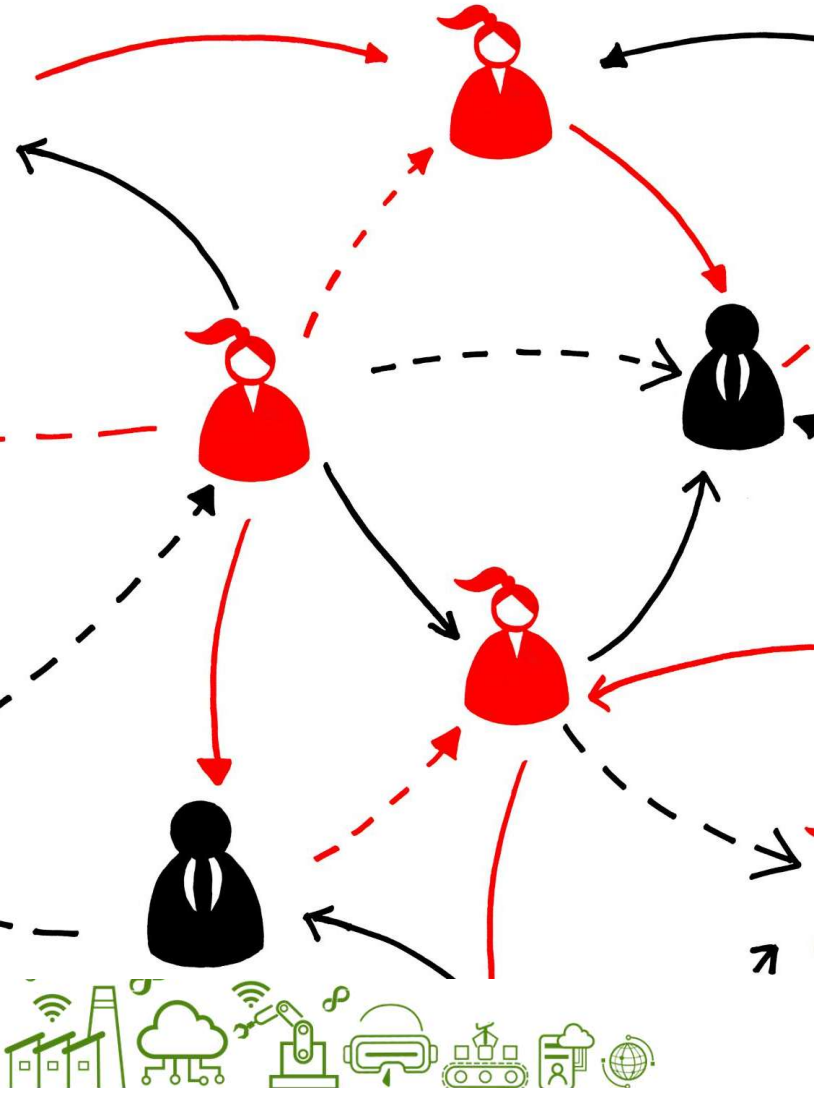
FPFtech
Escola Tecnológica

Introdução ao Git, Branches, Controle de Versão e Ferramentas Visuais

Conceitos essenciais para gerenciamento colaborativo de código

FPFtech
Escola Tecnológica

O que é Git e sua importância no desenvolvimento



Conceito fundamental de Git

Sistema de Controle de Versão

Git permite rastrear mudanças no código durante o desenvolvimento de software.

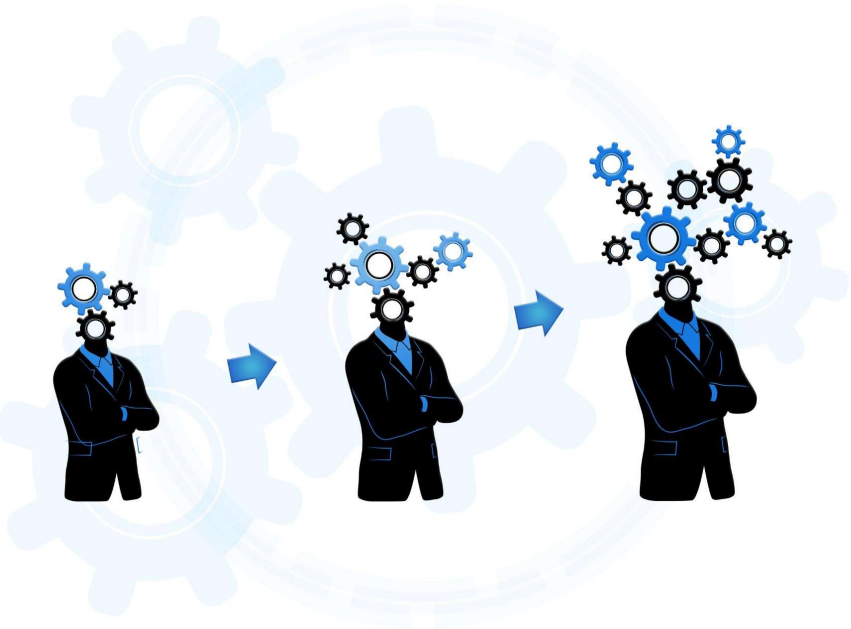
Colaboração Simplificada

Facilita a colaboração entre desenvolvedores em projetos de software.

Arquitetura Distribuída

Sua arquitetura distribuída oferece flexibilidade e melhor desempenho no gerenciamento de versões.

Principais vantagens do uso do Git



Controle Detalhado de Versões

Git oferece controle preciso das alterações, permitindo rastreamento detalhado do histórico do projeto.

Suporte a Múltiplas Branches

Permite criação e gerenciamento de múltiplas linhas de desenvolvimento simultâneas e isoladas.

Trabalho Offline

Desenvolvedores podem trabalhar localmente sem conexão, aumentando a flexibilidade e produtividade.

Alta Performance e Adoção

Git oferece alta performance e é amplamente adotado pela comunidade global de desenvolvedores.

Exemplo simples de inicialização de repositório: `git init`

FPFtech
Escola Tecnológica

Criação de Repositório Local

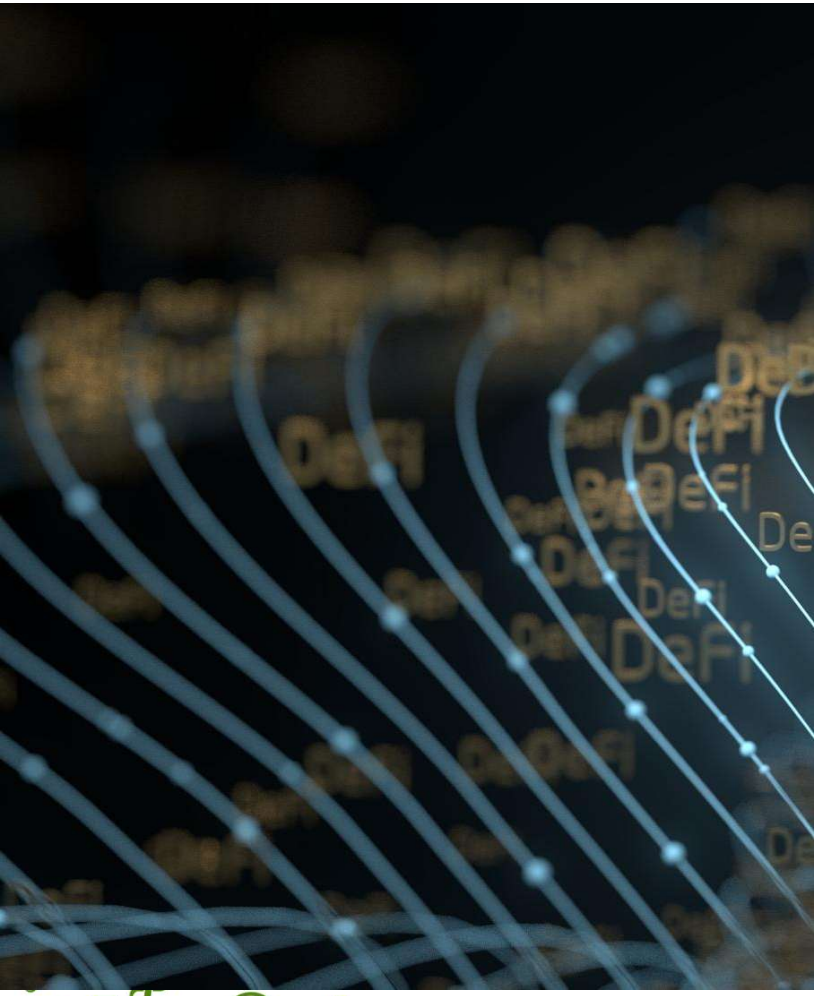
- O comando **git init** cria um novo repositório Git local para controle de versões.

Estrutura de Projeto Configurada

- Inicializa a estrutura necessária para versionar arquivos em um projeto de forma organizada.

FPFtech
Escola Tecnológica

Branches: Trabalhando com múltiplas linhas de desenvolvimento



Definição e finalidade de uma branch

Conceito de Branch

Branch é uma linha independente de desenvolvimento usada para separar funcionalidades ou correções.

Isolamento de Desenvolvimento

Permite trabalhar e testar funcionalidades sem impactar o código principal.

Facilitação da Integração

Branches facilitam a integração posterior ao código principal após testes.

Fluxos comuns de trabalho com branches

Feature Branches

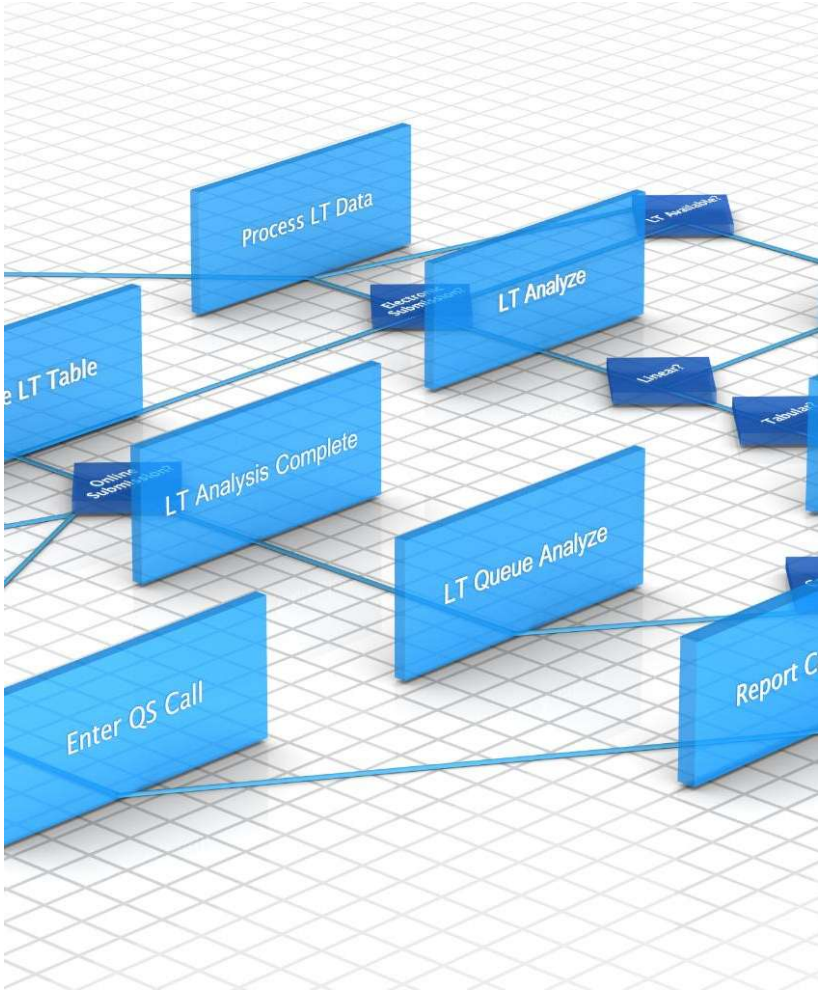
Branches dedicados ao desenvolvimento de novas funcionalidades no projeto, isolando mudanças até estarem prontas.

Bugfix Branches

Branches usados para correções rápidas de erros, facilitando a manutenção ágil do código.

Branches Principais

Branches como 'main' ou 'master' mantêm versões estáveis e prontas para produção do projeto.



Histórico de alterações e recuperação de versões

Registro Completo de Commits

Git mantém um histórico detalhado de todas as alterações feitas no código durante o desenvolvimento.

Revisão e Análise de Alterações

Permite revisar e analisar mudanças para entender a evolução do projeto ao longo do tempo.

Recuperação e Reversão

Facilita reverter alterações indesejadas, garantindo segurança no desenvolvimento do projeto.

Exemplo de commit e histórico: `git add`, `git commit`, `git log`

Preparar Arquivos para Commit

O comando 'git add' seleciona arquivos para serem incluídos no próximo commit, organizando as mudanças.

Salvar Alterações com Commit

O 'git commit' salva as alterações selecionadas com uma mensagem descritiva para documentar as mudanças.

Visualizar Histórico do Repositório

O comando 'git log' exibe o histórico dos commits, ajudando a acompanhar o progresso do projeto.

FPFtech
Escola Tecnológica

Configurando o GitKraken para integração eficiente

FPFtech
Escola Tecnológica

Instalação e primeiros passos no GitKraken

Instalação do GitKraken

Explicamos o processo passo a passo para baixar e instalar o GitKraken em diferentes sistemas operacionais.

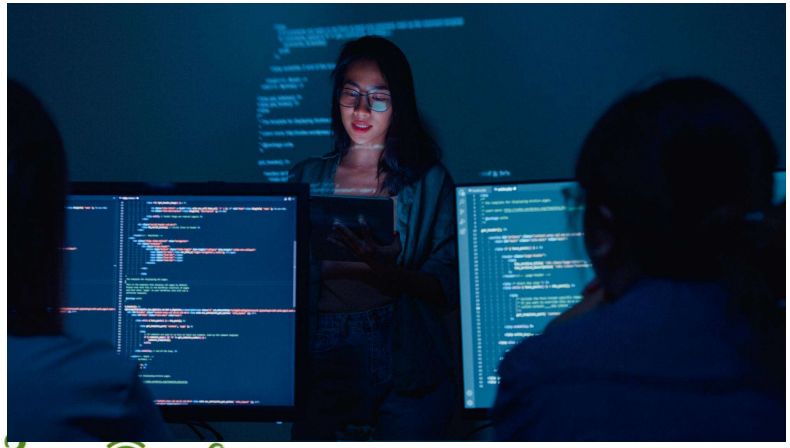
Configuração da Conta

Orientamos como criar e configurar sua conta para usar o GitKraken com repositórios locais e remotos.

Operações Básicas no Git

Demonstramos como executar operações essenciais do Git usando a interface visual do GitKraken.

Conectando-se ao GitHub pelo GitKraken



Vinculação da Conta

Conectar sua conta GitHub ao GitKraken permite acesso direto aos seus repositórios online de forma integrada.

Sincronização Eficiente

Sincronize alterações de maneira simples e eficiente entre GitKraken e GitHub para um fluxo de trabalho otimizado.

Exemplo de clone de repositório usando interface visual

Clonagem Visual de Repositórios

Interfaces gráficas facilitam o processo de clonagem de repositórios sem o uso do terminal.

Eliminação de Comandos de Terminal

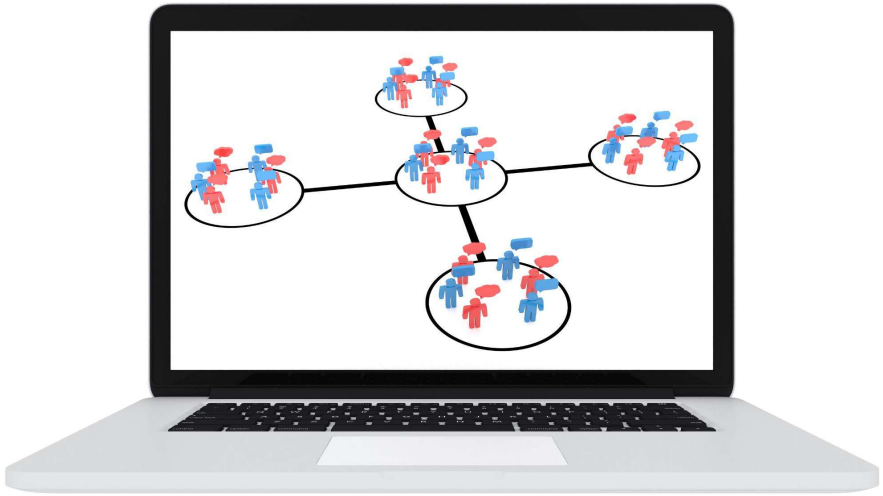
Usar ferramentas visuais elimina a necessidade de comandos complexos para clonar projetos.

FPFtech
Escola Tecnológica

GitHub: Plataforma colaborativa para projetos com Git

@fpftech.educacional

O que é GitHub e seus recursos principais



Hospedagem de Repositórios Git

GitHub permite armazenar e gerenciar projetos usando repositórios Git na nuvem, facilitando o versionamento.

Controle de Acesso e Revisão

Oferece ferramentas para controle de acesso e revisão colaborativa de código entre desenvolvedores.

Issues e Wikis

GitHub inclui funcionalidades para acompanhamento de bugs e documentação colaborativa através de issues e wikis.

Integração Contínua

Suporta integração contínua para automatizar testes e implantações, melhorando a qualidade do desenvolvimento.