Ebook: Introdução ao Git, GitHub e GitLab para Iniciantes

O controle de versão é uma prática essencial no desenvolvimento de software, permitindo que várias pessoas trabalhem em um projeto ao mesmo tempo e acompanhem as alterações feitas no código. Este ebook tem como objetivo introduzir você às ferramentas Git, GitHub e GitLab, mostrando como utilizá-las para gerenciar seu código de forma eficiente.

Git é uma ferramenta de controle de versão distribuído que permite rastrear mudanças no código-fonte ao longo do tempo. GitHub e GitLab são plataformas que hospedam repositórios Git e oferecem recursos adicionais para colaboração e automação.

F por Fernando Júnior

Capítulo 1: Git

O que é Git?

Git é um sistema de controle de versão distribuído, criado por Linus Torvalds em 2005. Ele permite que desenvolvedores acompanhem as alterações no código-fonte de um projeto, colaborem com outras pessoas e voltem a versões anteriores se necessário.

Por que usar Git?

- Histórico de mudanças: Rastreia todas as alterações feitas no código, permitindo voltar a versões anteriores.
- Colaboração: Vários desenvolvedores podem trabalhar no mesmo projeto simultaneamente.
- Ramo e fusão (branching e merging): Facilita o desenvolvimento de novas funcionalidades sem afetar o código principal.

Principais funcionalidades do Git

- Controle de versão distribuído: Cada desenvolvedor tem uma cópia completa do repositório.
- Branching e merging: Criar ramificações para novas funcionalidades e mesclá-las de volta ao ramo principal.
- Histórico de mudanças: Acompanhar quem fez o que e quando.
- Reversão: Voltar a versões anteriores do projeto.
- Colaboração em equipe: Trabalhar com vários desenvolvedores ao mesmo tempo.

Capítulo 2: Principais Comandos do Git

- git init: Inicializa um novo repositório Git no diretório atual.
- git clone: Clona um repositório Git remoto para o seu computador.
- git add: Adiciona arquivos ao índice (staging area) para preparar para o commit.
- git commit: Grava as alterações adicionadas no repositório local.
- git status: Exibe o status dos arquivos no diretório de trabalho e no índice.
- git push: Envia os commits do repositório local para um repositório remoto.
- git pull: Atualiza o repositório local com as mudanças do repositório remoto.
- git branch: Lista, cria ou exclui ramificações.
- git checkout: Troca de ramificações ou restaura arquivos no diretório de trabalho.
- git merge: Mescla mudanças de uma ramificação para outra.

Capítulo 3: GitHub

O que é GitHub?

GitHub é uma plataforma de hospedagem para repositórios Git, que facilita a colaboração entre desenvolvedores. Ele permite que você compartilhe seu código, colabore com outras pessoas e gerencie projetos de software.

Por que usar GitHub?

- Colaboração: Trabalhar com desenvolvedores de todo o mundo.
- Controle de versão: Rastrear mudanças no código e voltar a versões anteriores.
- Automação: Ferramentas de integração contínua (CI) e entrega contínua (CD).

Como criar uma conta no GitHub

1. Acesse **GitHub**.
2. Clique em "Sign up" e siga as instruções para criar uma conta.

Como criar um repositório no GitHub

1. Após fazer login, clique no ícone + no canto superior direito e selecione "New repository".
2. Preencha as informações do repositório e clique em "Create repository".

Como clonar um repositório

1. Copie o URL do repositório no GitHub.
2. No terminal, execute:

```
git clone https://github.com/seu-usuario/nome-do-repositorio.git
```

Como adicionar, commit e push do código para o GitHub

1. Adicione os arquivos ao índice:

```
git add .
```

1. Faça um commit das mudanças:

```
git commit -m "Adiciona código-fonte inicial"
```

1. Envie as mudanças para o GitHub:

```
git push origin main
```

Capítulo 4: GitLab

O que é GitLab?

GitLab é uma plataforma de hospedagem para repositórios Git, similar ao GitHub, mas com funcionalidades adicionais voltadas para equipes de desenvolvimento.

Diferenças entre GitHub e GitLab

- Funcionalidades adicionais: GitLab oferece mais ferramentas para planejamento de projetos e CI/CD.
- Controle mais granular: GitLab permite configurações mais detalhadas de permissões e segurança.

Funcionalidades adicionais do GitLab

- Planejamento de projetos: Ferramentas para gerenciar tarefas e sprints.
- CI/CD: Integração e entrega contínua para automatizar testes e deploys.
- Segurança: Ferramentas para análise de segurança do código.

Capítulo 5: Passo a Passo para Disponibilizar Código no GitHub

1. Criar um novo repositório
    - Acesse GitHub, faça login e crie um novo repositório.
2. Clonar o repositório para o computador
    - Use o comando git clone para clonar o repositório.
3. Adicionar código-fonte ao repositório
    - Adicione os arquivos ao índice (git add .), faça um commit (git commit -m "Mensagem") e envie as mudanças (git push).

Capítulo 6: Configurações Básicas e Recomendações

Configurações básicas

- .gitignore: Exclua arquivos desnecessários do controle de versão.
- README.md: Documente o projeto com informações importantes.

*.log temp/ build/

\# Nome do Projeto Descrição breve do projeto. ## Instalação Instruções de como instalar e configurar o projeto. ## Uso Exemplos de como usar o projeto. ## Contribuindo Passos para contribuir com o projeto.

Principais Recomendações

- Commits Frequentes: Realize commits frequentes e com mensagens claras.
- Branches: Utilize branches para desenvolver novas funcionalidades ou corrigir bugs.
- Documentação: Mantenha a documentação atualizada.
- Pull Requests: Revise e discuta mudanças antes de mesclá-las.
- Releases: Marque versões estáveis do projeto.

Capítulo 7: Conclusão

Recapitulação dos principais pontos abordados

- Relembrando a importância do controle de versão.
- Principais funcionalidades e comandos do Git.
- Uso de plataformas como GitHub e GitLab para colaboração e automação.

Dicas finais

- Continuar praticando e explorando mais recursos do Git, GitHub e GitLab.
- Participar de projetos open source para ganhar experiência.

Tutoriais

- **Git Tutorial - W3Schools**: Um tutorial prático e fácil de entender que cobre os comandos básicos do Git.
- **Git - O Básico do Git**: Um guia detalhado sobre os conceitos fundamentais do Git, disponível no site oficial do Git.

Livros

- **Pro Git**: Um livro completo sobre Git, escrito por Scott Chacon e Ben Straub, disponível em formato digital e impresso.
- **GitBook**: Uma plataforma para criar e compartilhar documentação técnica, incluindo livros sobre Git.

Cursos Online

- **Fundamentos do Git - Alison**: Um curso gratuito que ensina os conceitos básicos do Git e como usá-lo para gerenciar diferentes versões do seu código.
- **Complete Git - Coursera**: Uma especialização completa que cobre todos os aspectos do Git, incluindo operações essenciais, resolução de conflitos e gerenciamento de repositórios remotos.

Apêndice: Glossário de Termos

Repositório	Armazenamento do código-fonte e histórico de mudanças.
Commit	Registro de uma alteração no repositório.
Branch	Linha separada de desenvolvimento.
Merge	Combinação de duas branches.
Pull Request	Solicitação de revisão e mesclagem de mudanças.