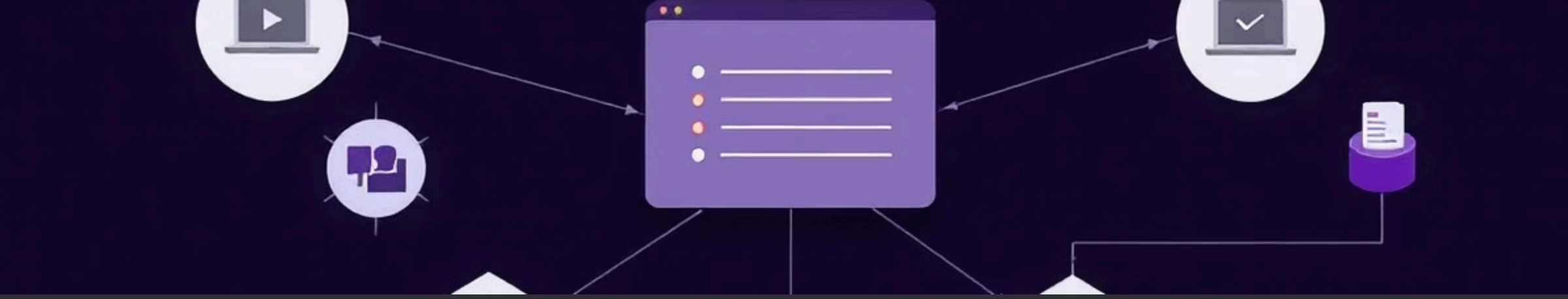
Introdução ao Git e GitHub

Bem-vindos ao minicurso de Git e GitHub! Hoje, exploraremos ferramentas essenciais para qualquer desenvolvedor, desde conceitos básicos até a colaboração prática.



O que é Git?

Sistema de Controle de Versão Distribuído (DVCS)

O Git permite que você e sua equipe rastreiem e gerenciem as alterações no código-fonte ao longo do tempo. Cada desenvolvedor tem uma cópia completa do histórico do projeto.

Histórico de Mudanças

Cada "snapshot" do seu projeto é armazenado, permitindo reverter para versões anteriores, comparar alterações e entender a evolução do código.

Eficiência e Velocidade

Projetado para ser rápido e eficiente, o Git gerencia projetos de qualquer tamanho, desde pequenos scripts até grandes sistemas complexos.

Git vs. GitHub

Git: A Ferramenta Local

O Git é o software que você instala em seu computador. Ele gerencia o controle de versão localmente, registrando o histórico de mudanças diretamente na sua máquina.

- Software de linha de comando
- Controle de versão offline
- Base para outros serviços

GitHub: A Plataforma Online

O GitHub é um serviço de hospedagem de repositórios Git na nuvem. Ele adiciona recursos sociais e de colaboração, tornando o trabalho em equipe mais eficiente.

- Hospedagem de repositórios remotos
- Ferramentas de colaboração (Pull Requests, Issues)
- Interface web intuitiva

Por Que Usar Git?

Colaboração Eficaz

Permite que múltiplos desenvolvedores trabalhem no mesmo projeto simultaneamente sem sobrescrever o trabalho uns dos outros.

Rastreamento de Versões

Todo o histórico de alterações é salvo, facilitando a identificação de bugs e a reversão para versões anteriores do código.

Segurança e Integridade

Garante que o código seja seguro e que nenhuma alteração se perca, mesmo em caso de falhas locais.

Agilidade no Desenvolvimento

Com a automação de processos e a facilidade de gerenciamento, o desenvolvimento se torna mais rápido e eficiente.

Instalação e Configuração Inicial

Antes de começar, precisamos configurar o Git com suas informações de usuário.

Instalação

Baixe o Git do site oficial (git-scm.com) para seu sistema operacional (Windows, macOS, Linux).

- Baixar Git

Configuração

Configure seu nome de usuário e e-mail. Isso identifica suas contribuições.

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

Fluxo de Trabalho Básico com Git

1

git init

Inicializa um novo repositório Git no diretório atual.

2

git add

Adiciona arquivos para a área de stage, preparando-os para o commit.

3

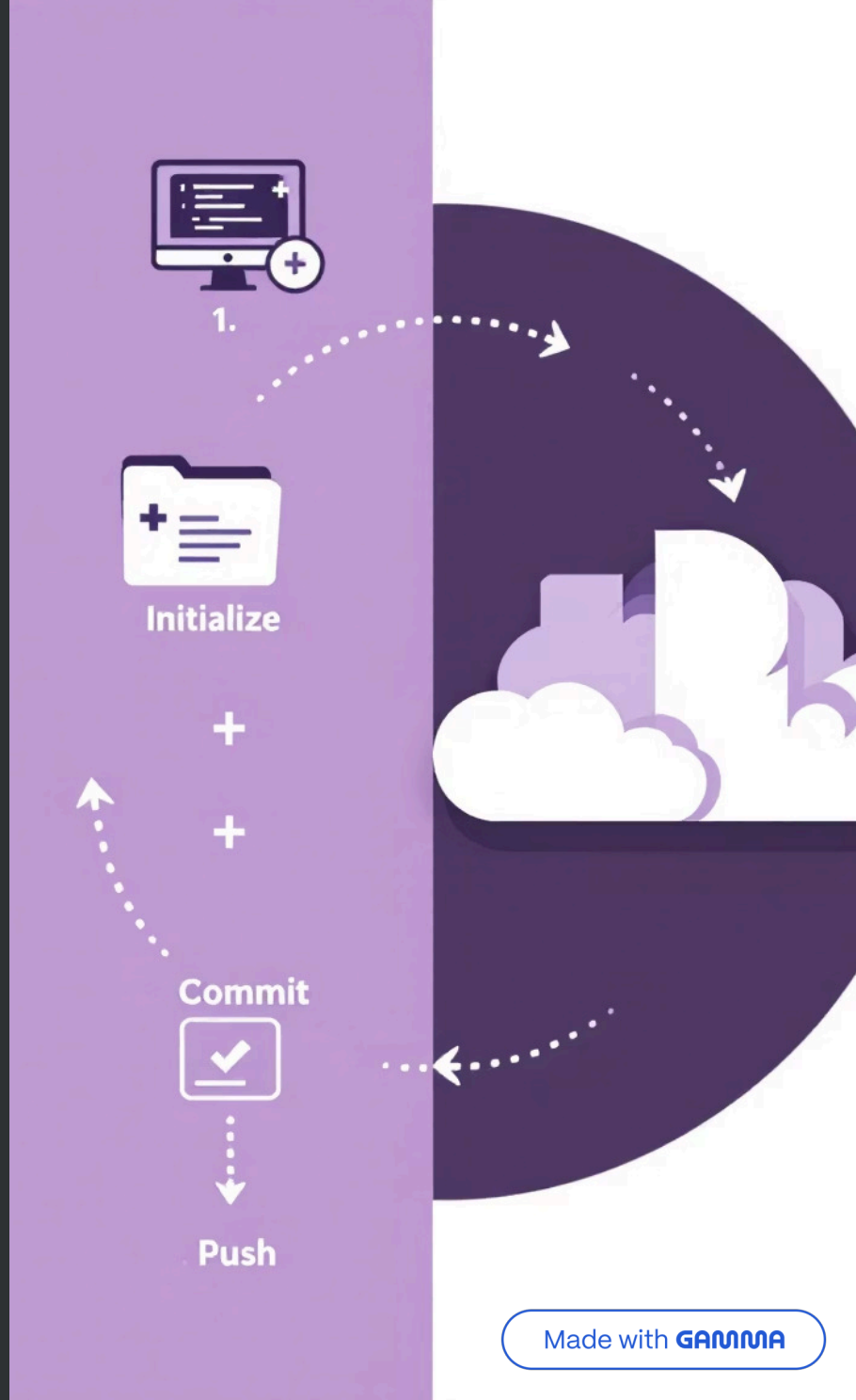
git commit

Grava as alterações da área de stage no histórico do repositório local.

4

git push

Envia seus commits locais para um repositório remoto (ex: GitHub).



Comandos Básicos Essenciais

git status

Mostra o estado dos arquivos no diretório de trabalho e na área de stage.

```
git status
```

git log

Exibe o histórico de commits, com detalhes sobre cada alteração.

```
git log
```

git diff

Compara as diferenças entre dois estados do repositório (commits, branches, arquivos).

```
git diff
```

git restore

Desfaz alterações em arquivos no diretório de trabalho ou na área de stage.

```
git restore <arquivo>
```

Branches e Merge

Branches (Ramificações)

Permitem desenvolver funcionalidades isoladamente sem afetar a linha principal de código. Ideal para experimentação e novos recursos.

```
git branch <nome_branch>
git checkout <nome_branch>
```

Merge (Fusão)

Integra as alterações de uma branch em outra. É o processo de combinar o código desenvolvido em paralelo.

```
git merge <nome_branch_a_fundir>
```

Conectando Git ao GitHub

Para compartilhar seu trabalho e colaborar, conecte seu repositório Git local ao GitHub.

Adicionar Repositório Remoto

```
git remote add origin <URL_do_repositório>
```

Clonar Repositório Existente

```
git clone <URL_do_repositório>
```

Enviar para o Remoto

```
git push -u origin main
```

Receber Atualizações

```
git pull origin main
```

Boas Práticas: README.md e .gitignore

README.md

Um arquivo de documentação essencial que fornece informações sobre seu projeto. Ele é o primeiro contato de qualquer um com seu código.

- Descrição do projeto
- Instruções de instalação e uso
- Exemplos de código e créditos

.gitignore

Lista de arquivos e diretórios que o Git deve ignorar, mantendo seu repositório limpo e focado no código-fonte relevante.

- Arquivos de configuração local
- Dependências de pacotes (ex: node_modules)
- Arquivos temporários ou de build

Dicas Finais

Canais Youtube

[Boson Treinamentos]

[Sujeito programador]

[DevDojo]

[Filipe Deschamps]

Stacks

SpringBoot (Java) + Angular2

Node + React (JavaScript)

Node + Angular2 (JavaScript)

.Net (C#) + React or Angular

React Native (JavaScript)

Plataformas

Azure

Amazon

Google Cloud

Heroku