SESC ESCOLA CUIABA-MT

Joao Victor Albuquerque de almeida



**GIT & GITHUB**

Cuiabá

2024

Joao Victor Albuquerque DE Almeida

GIT & GITHUB

Conceitos e definições

Trabalho apresentado no

curso de programação de

jogos digitais

Orientador: Wanderson

Cuiabá

2024

INTRODUÇAO :

O Git é um sistema de controle de versão distribuído amplamente utilizado para o gerenciamento de projetos de software e o rastreamento de alterações no código-fonte. Desenvolvido por Linus Torvalds em 2005, o Git se tornou uma ferramenta essencial para desenvolvedores de software devido à sua eficiência e flexibilidade.

O que e git?

Git é um sistema de controle de versão distribuído usado principalmente para rastrear mudanças em arquivos de código fonte durante o desenvolvimento de software. Ele permite que várias pessoas trabalhem em um mesmo projeto simultaneamente, facilitando a colaboração, o acompanhamento de alterações, a reversão a versões anteriores e a integração de novas funcionalidades de forma organizada e eficiente.

Para que serve

O Git serve principalmente para gerenciar o controle de versão de projetos de software, permitindo que várias pessoas trabalhem em um mesmo código fonte simultaneamente, mantendo um histórico das alterações feitas ao longo do tempo. Ele facilita a colaboração entre desenvolvedores, o acompanhamento de mudanças, a reversão a versões anteriores, a ramificação do código para desenvolvimento paralelo e a integração de novas funcionalidades de forma controlada e organizada.

1. Instalação: Baixe e instale o Git no seu sistema operacional.

2. Configuração inicial: Configure seu nome de usuário e endereço de e-mail no Git.

3. Iniciar um repositório: Use git init para iniciar um repositório Git local no diretório do seu projeto.

4. Adicionar arquivos: Utilize git add <nome\_do\_arquivo> para adicionar arquivos ao índice do Git.

5. Commitar mudanças: Execute git commit -m “Mensagem do commit” para confirmar as mudanças adicionadas ao índice. Você enviou

6. Criar e mudar de branch (ramo): Use git branch <nome\_do\_branch> para criar um novo ramo e git checkout <nome\_do\_branch> para mudar para um ramo existente.

7. Sincronizar com um repositório remoto: Adicione um repositório remoto usando git remote add origin <URL\_do\_repositório> e use git push origin <nome\_do\_branch> para enviar suas mudanças para o repositório remoto

8. Atualizar seu repositório local: Utilize git pull para baixar e mesclar as mudanças do repositório remoto com seu repositório lo 1.

Aqui estão 15 comandos fundamentais do Git:

1. \*\*git init\*\*:

- Inicializa um novo repositório Git no diretório atual.

2. \*\*git clone <URL>\*\*:

- Clona um repositório remoto para o seu computador.

3. \*\*git add <arquivo>\*\*:

- Adiciona alterações de um arquivo específico ao índice (staging area) para o próximo commit.

4. \*\*git add .\*\*:

- Adiciona todas as alterações no diretório atual ao índice para o próximo commit.

5. \*\*git commit -m "mensagem"\*\*:

- Registra as alterações adicionadas ao índice em um novo commit, com uma mensagem descritiva.

6. \*\*git status\*\*:

- Mostra o estado atual do seu repositório Git, incluindo arquivos modificados, adicionados e pendentes de commit.

7. \*\*git log\*\*:

- Exibe um histórico detalhado de commits, mostrando quem fez as alterações, quando e as mensagens de commit associadas.

8. \*\*git branch\*\*:

- Lista todos os branches existentes no repositório local.

9. \*\*git checkout <branch>\*\*:

- Alterna para o branch especificado.

10. \*\*git checkout -b <novo-branch>\*\*:

- Cria e alterna para um novo branch em um único comando.

11. \*\*git merge <branch>\*\*:

- Mescla as alterações do branch especificado para o branch atual.

12. \*\*git remote add <nome-remoto> <URL>\*\*:

- Adiciona um novo repositório remoto com o nome especificado.

13. \*\*git push <remote> <branch>\*\*:

- Envia os commits locais para um repositório remoto, geralmente o branch principal.

14. \*\*git pull <remote> <branch>\*\*:

- Obtém e mescla as alterações de um repositório remoto para o seu repositório local.

15. \*\*git diff\*\*:

- Mostra as diferenças entre o estado atual do seu diretório de trabalho e o índice (staging area).

Esses são apenas alguns dos comandos mais básicos e essenciais do Git. Existem muitos outros comandos e opções disponíveis para operações mais avançadas.

Introdução : O GitHub é uma plataforma de hospedagem de código-fonte baseada na web que utiliza o Git para controle de versão e colaboração em projetos de software. Ele oferece uma variedade de recursos que facilitam o trabalho em equipe, o gerenciamento de projetos e a contribuição para projetos de código aberto.

O que e:

O GitHub é uma plataforma de desenvolvimento de software baseada na web que oferece hospedagem de repositórios de código usando o sistema de controle de versão Git. Fundado em 2008, o GitHub se tornou um dos maiores e mais influentes serviços para desenvolvedores ao redor do mundo.

O GitHub serve como uma plataforma central para desenvolvedores de software colaborarem, compartilharem e gerenciarem projetos de código-fonte. Aqui estão algumas das principais funções e usos do GitHub:

1. \*\*Hospedagem de Repositórios\*\*: Os desenvolvedores podem hospedar seus projetos de software em repositórios Git no GitHub. Isso fornece um local centralizado para armazenar todo o código-fonte, arquivos de configuração, documentação e outros recursos relacionados ao projeto.

2. \*\*Controle de Versão\*\*: O GitHub utiliza o sistema de controle de versão Git para rastrear e gerenciar as alterações nos arquivos do projeto ao longo do tempo. Isso permite que os desenvolvedores trabalhem em diferentes versões do código, acompanhem o histórico de alterações e revertam para versões anteriores se necessário.

3. \*\*Colaboração\*\*: Uma das principais funcionalidades do GitHub é facilitar a colaboração entre desenvolvedores. Os usuários podem contribuir com projetos de código aberto enviando pull requests com novas funcionalidades, correções de bugs ou melhorias. Eles também podem participar de discussões através de issues, reportando problemas, sugerindo novas ideias ou fornecendo feedback.

4. \*\*Gestão de Projetos\*\*: O GitHub oferece uma variedade de ferramentas para ajudar os desenvolvedores a gerenciar seus projetos de forma eficiente. Isso inclui recursos como quadros Kanban para organizar tarefas, marcos (milestones) para acompanhar o progresso do projeto e integrações com ferramentas de gerenciamento de projetos como o Jira.

5. \*\*Portfólio de Trabalho\*\*: O GitHub também serve como um portfólio de trabalho para desenvolvedores. Eles podem exibir seus projetos, contribuições para projetos de código aberto, habilidades técnicas e experiência profissional em seus perfis do GitHub. Isso pode ser útil para encontrar empregos, estabelecer colaborações ou demonstrar competências para potenciais empregadores.

Em resumo, o GitHub é uma ferramenta essencial para desenvolvedores de software, oferecendo uma maneira poderosa de hospedar, colaborar e gerenciar projetos de código-fonte de forma eficiente.

Como usar ?

Usar o GitHub envolve uma série de etapas básicas que vão desde a criação de um repositório até a colaboração em projetos existentes. Aqui está uma visão geral simples de como começar a usar o GitHub:

### Criar um Repositório:

1. \*\*Criar uma Conta\*\*: Se você ainda não tem uma, crie uma conta no GitHub em [github.com](https://github.com/).

2. \*\*Criar um Novo Repositório\*\*: No seu perfil do GitHub, clique no botão "New" (Novo) para criar um novo repositório. Dê um nome ao repositório, adicione uma descrição opcional, escolha se ele será público ou privado e clique em "Create Repository" (Criar Repositório).

### Clonar um Repositório Existente (Opcional):

Se você deseja trabalhar em um projeto existente hospedado no GitHub:

1. Encontre o repositório desejado no GitHub.

2. Copie a URL do repositório.

3. No seu terminal, use o comando `git clone <URL>` para clonar o repositório para o seu computador.

### Trabalhar com o Repositório Local:

1. \*\*Adicionar Arquivos\*\*: Adicione ou modifique arquivos no seu diretório de trabalho.

2. \*\*Adicionar Alterações ao Índice\*\*: Use o comando `git add <arquivo>` para adicionar as alterações ao índice (staging area).

3. \*\*Realizar um Commit\*\*: Use o comando `git commit -m "mensagem"` para criar um commit com as alterações adicionadas ao índice.

4. \*\*Enviar Alterações para o Repositório Remoto\*\*: Se você criou um novo repositório, use o comando `git push origin master` para enviar as alterações para o repositório remoto.

### Colaborar em Projetos:

1. \*\*Fazer Fork de um Projeto\*\*: Se você deseja contribuir para um projeto de código aberto, faça um fork do projeto original para o seu próprio perfil no GitHub.

2. \*\*Enviar Pull Requests\*\*: Faça as alterações desejadas no seu fork do projeto e envie um pull request para o projeto original. Isso permite que os mantenedores do projeto revisem e incorporem suas alterações.

3. \*\*Participar de Discussões\*\*: Contribua com discussões em issues, fornecendo feedback, reportando bugs ou sugerindo novas funcionalidades.

### Atualizar o Repositório Local com Alterações Remotas (Opcional):

Se outras pessoas colaborarem no mesmo projeto e você deseja obter as alterações mais recentes:

1. Use o comando `git pull origin master` para obter e mesclar as alterações do repositório remoto para o seu repositório local.

Essas são apenas algumas das operações básicas que você pode realizar no GitHub. À medida que você ganha mais experiência, você pode explorar recursos mais avançados e técnicas de colaboração mais complexas.