Curso Completo de Git e GitHub – Resumo Detalhado

Seção 1: Introdução e Instalação das Dependências

Nesta etapa, aprendemos os fundamentos do Git e a diferença entre Git e GitHub:

- **Git**: sistema de controle de versão distribuído, que permite acompanhar e gerenciar mudanças no código.
- **GitHub**: plataforma de hospedagem de repositórios Git na nuvem, com foco na colaboração entre desenvolvedores.

Principais ações:

- Instalação do **Git** no sistema operacional.
- Configuração inicial com:
- Criação da conta no **GitHub**.

Seção 2: Git Fundamental

Aqui são abordados os comandos essenciais para versionamento local:

- git init: inicia um repositório Git em uma pasta.
- git status: mostra o estado atual dos arquivos.
- git add: adiciona arquivos à área de preparação (staging).
- git commit -m "mensagem": registra um snapshot do código.
- git log: exibe o histórico de commits.
- git diff: compara alterações nos arquivos.
- git reset: desfaz mudanças.

Também aprendemos sobre os **estágios de um arquivo**:

1. Untracked (não versionado)
2. Staged (pronto para commit)
3. Committed (registrado no histórico)

Seção 3: Trabalhando com Branches

Branches (ramificações) permitem o desenvolvimento de recursos paralelos sem afetar o código principal.

- git branch nome: cria uma nova branch.
- git checkout nome: troca para a branch.
- git merge nome: junta uma branch ao código principal.
- git branch -d nome: deleta uma branch.

Com isso, é possível trabalhar em features de forma isolada e segura, integrando depois ao branch principal (main ou master).

Seção 4: Compartilhamento e Atualização de Repositórios

Foco na conexão com o GitHub:

- git remote add origin url: conecta o repositório local ao GitHub.
- git push -u origin main: envia os commits para o GitHub.
- git pull origin main: puxa as atualizações do GitHub para o projeto local.
- git clone url: clona um repositório remoto.

Esses comandos tornam possível a colaboração com outras pessoas e o backup na nuvem.

Seção 5: Análise e Inspeção de Repositórios

Exploração do histórico de mudanças:

- git log: mostra todos os commits.
- git diff: compara versões antes do commit.
- git show: mostra detalhes de um commit específico.
- git blame arquivo: mostra qual linha foi alterada por qual commit/autor.

Essas ferramentas ajudam a **auditar** e **entender** a evolução do projeto.

Seção 6: Administração de Repositórios

Aqui aprendemos a organizar o repositório:

- .gitignore: define arquivos/pastas que não devem ser versionados.
- git rm nome: remove arquivos do repositório.
- git mv antigo novo: renomeia arquivos versionados.
- git clean: remove arquivos não rastreados.

Tudo isso ajuda a manter o projeto limpo e organizado.

Seção 7: Melhorando os Commits do Projeto

Introduz as **boas práticas** de mensagens de commit:

- Clareza: o commit deve descrever a mudança de forma objetiva.
- Uso de convenções como:

  feat: nova funcionalidade
  fix: correção de bug
  docs: mudanças na documentação
  refactor: refatoração de código

Isso facilita o entendimento do histórico por toda a equipe.

Seção 8: Explorando e Entendendo o GitHub

Exploramos as funcionalidades da interface GitHub:

- **Issues**: para reportar bugs ou sugerir melhorias.
- **Fork**: cria uma cópia independente de um repositório.
- **Pull Request (PR)**: propõe mudanças para serem integradas ao projeto original.
- **Actions**: automatizam tarefas como testes e deploy.
- **Stars e Watch**: acompanhar e apoiar projetos.

O GitHub é essencial para **colaboração open source** e **projetos em equipe**.

Seção 9: Markdown do Básico ao Avançado

Aprendemos a usar o **Markdown**, linguagem de marcação leve, usada para documentações como o README.md.

Exemplos:

- Títulos: # Título, ## Subtítulo
- Negrito: **texto**
- Listas: - item
- Código: \codigo`` ou blocos ```js
- Links e imagens

A boa documentação é essencial para projetos profissionais.

Seção 10: Projeto – Portfólio com GitHub Pages

Aplicação prática:

- Criar um site de portfólio com HTML/CSS.
- Hospedar gratuitamente usando o **GitHub Pages**.
- Usar a branch gh-pages ou a opção do GitHub para ativar a hospedagem.

Link fica assim:

https://seunome.github.io/nome-do-repo/

Seção 11: Encerramento e Próximos Passos

Encerramos com dicas para o futuro:

- Pratique diariamente os comandos Git.
- Participe de projetos colaborativos no GitHub.
- Construa um portfólio público.
- Explore ferramentas como **GitKraken**, **SourceTree** ou extensões do VS Code.