1. O que é o Git e qual é a sua principal função no controle de versão de código-fonte?

O Git é um projeto de código aberto maduro e com manutenção ativa desenvolvido em 2005 por Linus Torvalds, o famoso criador do kernel do sistema operacional Linux. Um número impressionante de projetos de software depende do Git para controle de versão, incluindo projetos comerciais e de código-fonte aberto.

<https://www.atlassian.com/br/git/tutorials/what-is-git#:~:text=O%20Git%20%C3%A9%20um%20projeto,e%20de%20c%C3%B3digo%2Dfonte%20aberto>.

2. Como podemos criar um novo repositório Git em um projeto existente no seu sistema local?

Para criar um novo repositório, você vai usar o comando git init . git init é um comando único que você usa durante a configuração inicial de um novo repositório. A execução desse comando cria um novo subdiretório.

<https://www.atlassian.com/br/git/tutorials/setting-up-a-repository#:~:text=Para%20criar%20um%20novo%20reposit%C3%B3rio,comando%20cria%20um%20novo%20subdiret%C3%B3rio%20>.

3. Qual a diferença entre "git add" e "git commit" no contexto do Git?

Os instantâneos com commit podem ser considerados versões "seguras" de um projeto, o Git nunca os altera, a menos que você peça a ele. Antes da execução de git commit , o comando git add é usado para promover ou "preparar" alterações no projeto que são armazenadas em uma confirmação.

https://www.atlassian.com/br/git/tutorials/saving-changes/git-commit#:~:text=Os%20instant%C3%A2neos%20com%20commit%20podem,s%C3%A3o%20armazenadas%20em%20uma%20confirma%C3%A7%C3%A3o.

4. Explique o propósito do comando "git push" e quando você o utilizaria.

O Git push é um comando que possibilita que as alterações da sua máquina local sejam enviadas para uma máquina remota. Mas, vamos entender isso melhor:

Imagine que você está desenvolvendo um projeto com uma equipe. O projeto que vocês estão trabalhando está no GitHub, mas você vai desenvolvê-lo no seu computador. Como fazer para “enviar” seu desenvolvimento para o GitHub?

Para enviar suas alterações, você usará o comando git push. Antes dele, evidentemente, temos alguns outros comandos, mas é o git push que pegará tudo o que você fez e colocará em um local onde qualquer pessoa colaboradora possa ver e pegar suas alterações.

<https://blog.betrybe.com/git/git-push/>

5. O que são branches no Git e por que são úteis em um projeto de desenvolvimento?

Um branch no Git é simplesmente um ponteiro móvel para um desses commits. O nome do branch padrão no Git é master . Conforme você começa a fazer commits, você recebe um branch master que aponta para o último commit que você fez. Cada vez que você faz um novo commit, ele avança automaticamente.

<https://git-scm.com/book/pt-br/v2/Branches-no-Git-Branches-em-poucas-palavras#:~:text=Um%20branch%20no%20Git%20%C3%A9,novo%20commit%2C%20ele%20avan%C3%A7a%20automaticamente>.

6. Como podemos verificar o status atual do repositório local no Git?

Inicialmente se estamos trabalhando em um arquivo local e temos um repositorio github teriamos que saber se a versão local e do repositório estão alinhadas. Para isso usamos: git status.

<https://cursos.alura.com.br/forum/topico-comando-git-297811#:~:text=Inicialmente%20se%20estamos%20trabalhando%20em,git%20status>

7. Explique o que é um conflito de merge no Git e como você resolveria esse problema.

Os conflitos de merge ocorrem quando alterações concorrentes são feitas na mesma linha de um arquivo ou quando uma pessoa edita um arquivo e outra pessoa exclui o mesmo arquivo. Para obter mais informações, confira "Sobre conflitos de mesclagem".

<https://docs.github.com/pt/pull-requests/collaborating-with-pull-requests/addressing-merge-conflicts/resolving-a-merge-conflict-using-the-command-line>

8. O que é o GitHub e como ele se diferencia do Git em termos de funcionalidades?

O Github é uma “rede social dev” em que é possível armazenar e compartilhar projetos de desenvolvimento de software. O Git é um sistema de controle de versão de arquivos; em outras palavras, é responsável por guardar o histórico de alterações sempre que alguém modificar algum arquivo que está sendo monitorado por ele.

<https://www.alura.com.br/artigos/o-que-e-git-github#:~:text=O%20Github%20%C3%A9%20uma%20%E2%80%9Crede,est%C3%A1%20sendo%20monitorado%20por%20ele>.

10. Como podemos realizar o processo de clonagem (clone) de um repositório do GitHub para o seu computador local?

Para clonar um repositório localmente, use o subcomando repo clone . Substitua o parâmetro repository pelo nome do repositório. Por exemplo, octo-org/octo-repo , monalisa/octo-repo ou octo-repo . Se a parte OWNER/ do argumento do repositório OWNER/REPO for omitida, ele usará como padrão o nome do usuário autenticador.

<https://docs.github.com/pt/repositories/creating-and-managing-repositories/cloning-a-repository>

11.O que são Pull Requests no GitHub e qual é o seu propósito no desenvolvimento colaborativo?

Os modelos de pull request permitem personalizar e padronizar as informações que você gostaria de incluir quando alguém cria uma pull request em seu repositório.

<https://docs.github.com/pt/pull-requests/collaborating-with-pull-requests/getting-started/best-practices-for-pull-requests>

12. Como podemos adicionar um arquivo ao arquivo .gitignore e qual é a sua finalidade?

Normalmente, você apenas adicionará um arquivo . gitignore à pasta raiz do projeto. No entanto, você pode adicionar um arquivo . gitignore a qualquer pasta de projeto para informar ao Git quais arquivos ignorar dentro dessa pasta e suas subpastas em qualquer profundidade aninhada.

<https://learn.microsoft.com/pt-br/azure/devops/repos/git/ignore-files?view=azure-devops&tabs=visual-studio-2022>

13. Explique o que é um fork no GitHub e em que situações você pode querer realizar um fork de um repositório.

Os forks geralmente são usados para iterar ideias ou alterações antes de serem propostas de volta para o repositório upstream, como em projetos código aberto ou quando um usuário não tem acesso de gravação ao repositório upstream. Para saber mais, confira "Trabalhar com bifurcações".

<https://docs.github.com/pt/pull-requests/collaborating-with-pull-requests/working-with-forks/fork-a-repo>

14. Como podemos visualizar o histórico de commits em um repositório Git?

A ferramenta mais básica e poderosa para fazer isso é o comando git log . Por padrão, sem argumentos, git log lista os commits feitos neste repositório em ordem cronológica inversa; isto é, o commit mais recente aparece primeiro.

<https://git-scm.com/book/pt-br/v2/Fundamentos-de-Git-Vendo-o-hist%C3%B3rico-de-Commits#:~:text=A%20ferramenta%20mais%20b%C3%A1sica%20e,%C3%A9%20o%20comando%20git%20log%20.&text=Por%20padr%C3%A3o%2C%20sem%20argumentos%2C%20git,commit%20mais%20recente%20aparece%20primeiro>.

15.Qual é a função do comando "git pull" e quando ele deve ser utilizado no desenvolvimento colaborativo com o Git e GitHub?

O comando git pull serve para recuperar e baixar o conteúdo de um repositório remoto e atualizar o repositório local assim que ele for baixado. Isso é muito útil em fluxos de trabalho de colaboração que precisam mesclar alterações upstream remotas no repositório local.

https://www.locaweb.com.br/blog/temas/codigo-aberto/git-pull-aprenda-a-usar-o-comando/#:~:text=O%20comando%20git%20pull%20serve,upstream%20remotas%20no%20reposit%C3%B3rio%20local.