Controlo de Versões

Ou Como Gerir um Projecto em Grupo

Bernardo Sequeiros, Nuno Pombo

Plataformas e Serviços X-Ops, 2023/24

Índice

Controlo de Versões

Como Funciona

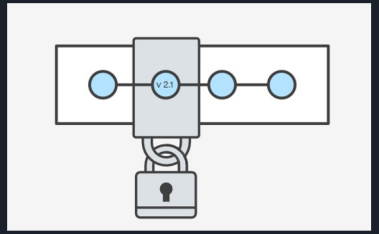
Vantagens

Algumas Regras e Boas Práticas

Como Gerir

Exemplo Prático

Controlo de Versões



O que é controlo de versões?

Utilizado em quase todo o projecto de software profissional;

Possibilita que várias pessoas trabalhem em simultâneo;

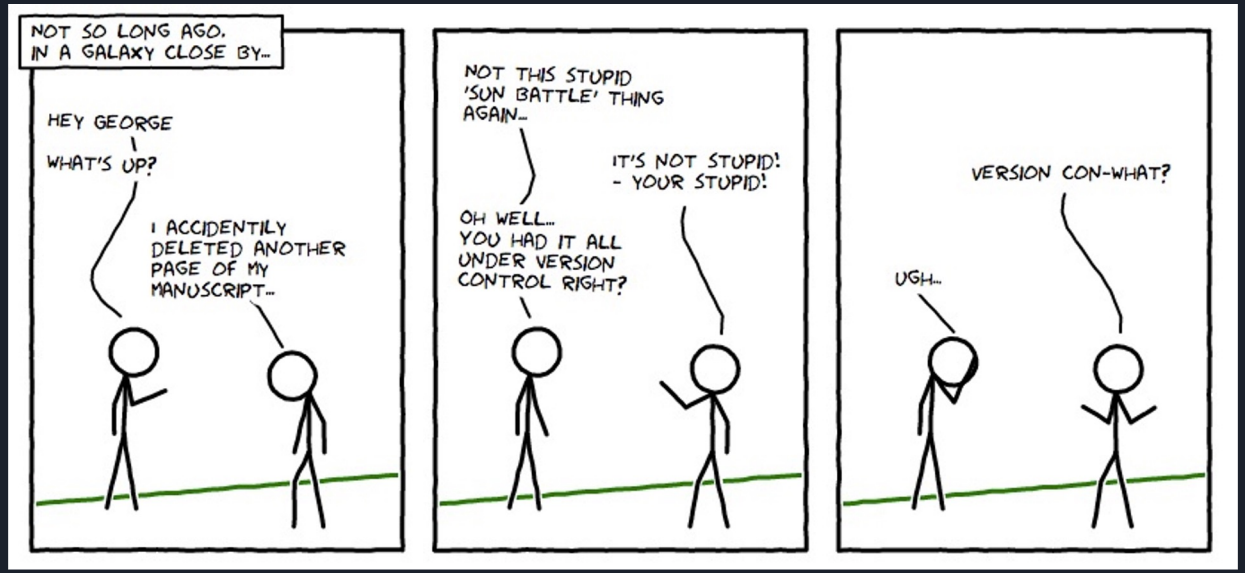
Regista todas as modificações feitas (como uma base dados de todas as alterações);

Permite regredir para versões anteriores;

Permite combinar trabalho;

Resolve muitas dores de cabeça!

Como Funciona



Como Funciona

Existe um registo de todos os ficheiros e alterações feitas a estes. Sempre que dois utilizadores alteram o mesmo ficheiro, o sistema tenta fundir as alterações, desde que não ocorram conflitos (caso existam, o utilizador tem de fazer a fusão manualmente);

Podem existir diversos “ramos” de desenvolvimento. Estes ramos permitem trabalhar simultaneamente em versões diferentes;

Todas as alterações ficam registadas. É possível verificar o que cada pessoa alterou, quando alterou, e avançar ou recuar conforme as necessidades. Não mais ocorre o problema de alguém “apagar trabalho acidentalmente”.

Como Funciona

***Commit*, *Pull*, *Push*:**

Estas são algumas das instruções mais comuns num sistema de controlo de versões;

Commit permite registar alterações feitas localmente pelo utilizador;

Pull permite ir buscar as alterações feitas pelos outros utilizadores;

Push envia as alterações que registámos num *commit* para os outros utilizadores.

Vantagens

Como já referido, a principal vantagem é poder ter várias pessoas a colaborarem no mesmo projecto, com um repositório comum, e onde todos podem trabalhar em simultâneo, sem risco de se perder informação. A capacidade de reverter alterações (por exemplo, que criem *bugs*) é outra grande vantagem.

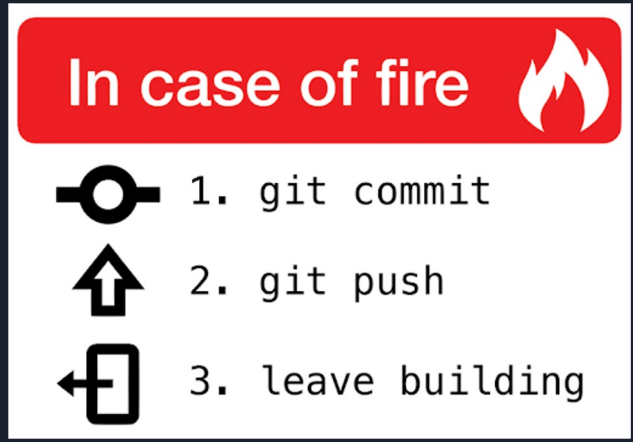
Algumas Regras e Boas Práticas

As mensagens dos *commits* devem ser elucidativas;

Cada *commit* deve ter um propósito específico (*e.g.*, adicionar uma nova funcionalidade ou corrigir um *bug* em particular);

Efectuar *pulls* frequentemente, para garantir que as alterações feitas pelos outros membros constam da nossa versão;

Efectuar *commits* e *pushes* também frequentemente.



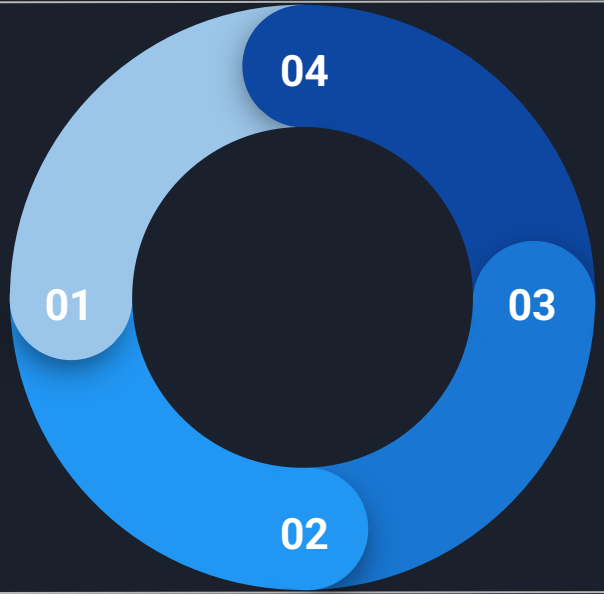
Como Gerir

Obter Alterações

Efectuar *Pull* do repositório central para garantir que temos as alterações mais recentes.

Desenvolver

Efectuar o nosso trabalho no repositório local.



Enviar as Alterações

Efectuar um *Push*, enviando as novas alterações para o repositório central.

Commit

Registar as alterações que efectuámos;

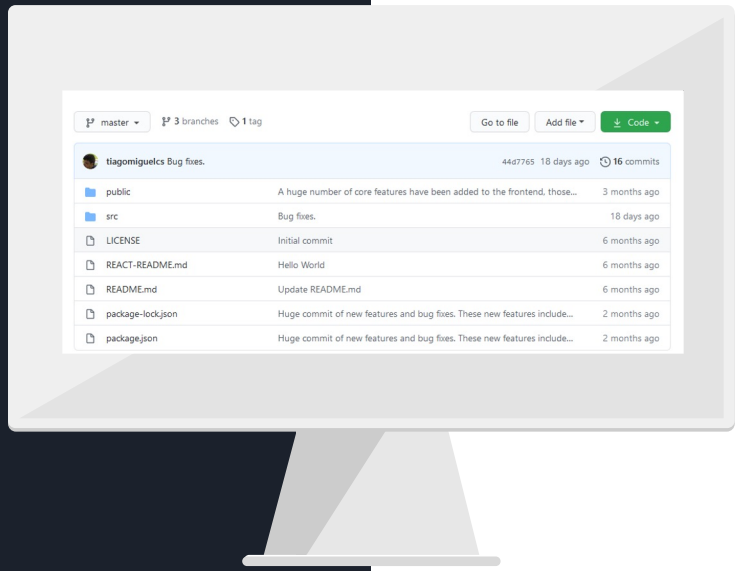
Exemplo Prático

A título de exemplo, aqui será feita uma pequena amostra de git, um dos mais conhecidos softwares de controlo de versões.

Como repositório central, iremos usar o **github**. O **github** fornece, de forma gratuita, repositórios centralizados e suporte para git para qualquer utilizador.

Github

Exemplo de um repositório no Github.



Git

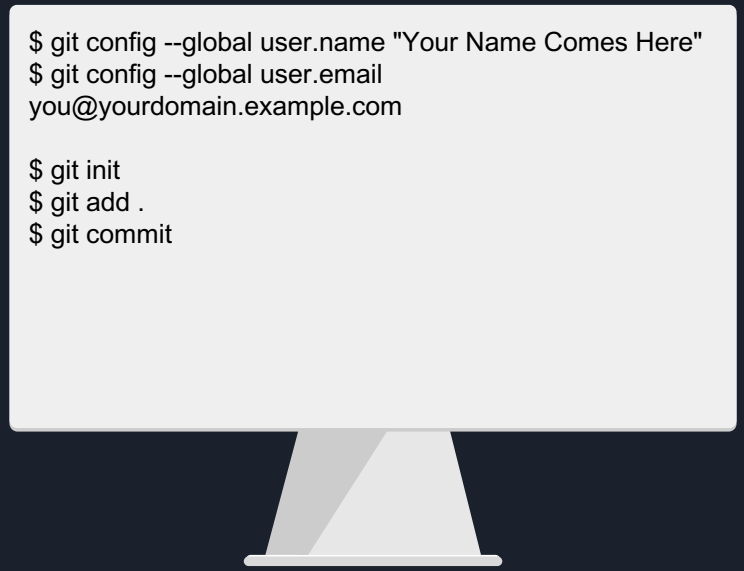
Primeiro é necessário instalar o software Git na nossa máquina.

Apesar de existirem algumas ferramentas com interface gráfica (e de alguns IDEs suportarem git nativamente), a abordagem base aqui demonstrada utiliza o Git na sua forma mais comum - através da linha de comandos.

Git

Estando, através da linha de comandos, na pasta onde pretendemos inicializar o repositório, vamos inicializar o mesmo.

Comando básicos a efectuar para inicializar um repositório. Ter em conta que git init inicializa o repositório vazio (por exemplo, para um projecto local).

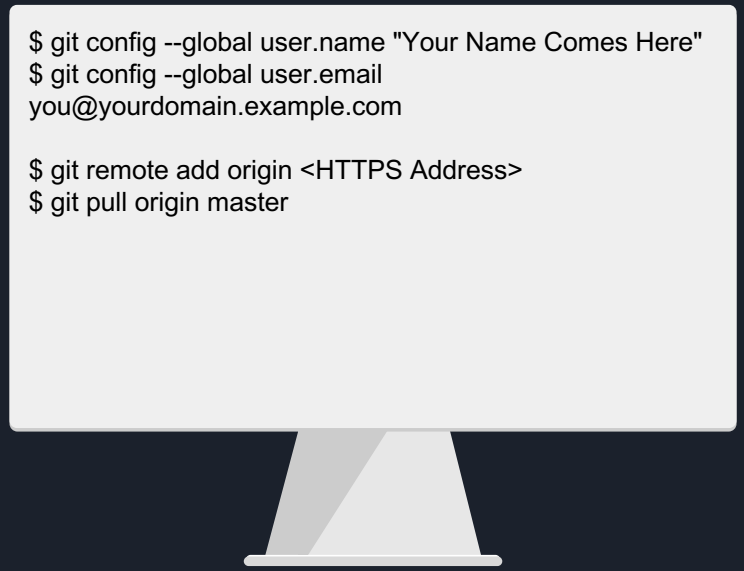


Após estes comandos, todos os ficheiros que estavam presentes na pasta estão cometidos e registados no histórico do projecto.

Git

Para um repositório que criámos remotamente, como no github, o funcionamento é semelhante.

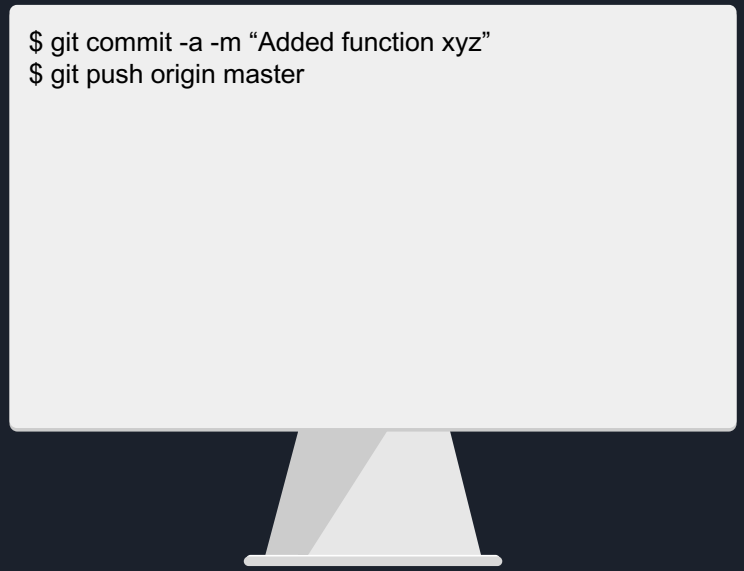
Vamos obter todos os ficheiros que estão no repositório remoto, do *branch master* (para efeitos de simplificação, vamos utilizar apenas este *branch*).



Git

Após efectuarmos alterações, devemos cometê-las e enviá-las para o repositório central.

Vai ser enviado para o repositório central, e a mensagem definida ficará definida no histórico.



Dúvidas?