Como usar o Git no RStudio

O que é controle de versão, e por que você deve se importar?

“O controle de versão é um sistema que registra as mudanças feitas em um arquivo ou um conjunto de arquivos ao longo do tempo, de forma que você possa recuperar versões específicas.

Se quer manter todas as versões de uma imagem ou layout, usar um Sistema de Controle de Versão (Version Control System ou VCS) é uma decisão sábia. Ele permite reverter arquivos e projetos inteiros para um estado anterior, compara mudanças feitas ao decorrer do tempo, vê quem foi o último a modificar algo que pode estar causando problemas, quem introduziu um bug e quando, e muito mais. Usar um VCS normalmente significa que caso tenha estragado ou perdido algum arquivo, poderá facilmente reavê-los. Além disso, você pode controlar tudo sem maiores esforços.”

O que é o Git?

O Git é um VCS (Version Control System) permite que você registre as mudanças feitas em um arquivo ou um conjunto de arquivos ao longo do tempo de forma que se possa recuperar versões específicas.

Git pronunciado [git] (ou pronunciado [dit] em inglês britânico) é um sistema de controle de versão distribuído e um sistema de gerenciamento de código fonte, com ênfase em velocidade. O Git foi inicialmente projetado e desenvolvido por Linus Torvalds para o desenvolvimento do kernel Linux, mas foi adotado por muitos outros projetos.

Cada diretório de trabalho do Git é um repositório com um histórico completo e habilidade total de acompanhamento das revisões, não dependente de acesso a uma rede ou a um servidor central.

O Git é um software livre, distribuído sob os termos da versão 2 da GNU General Public License.

Vantagens do uso do Git:

- **Controle do histórico**: facilidade em *desfazer* e possibilidade de analisar o histórico do desenvolvimento, como também facilidade no resgate de versões mais antigas e estáveis. A maioria das implementações permitem analisar as alterações com detalhes, desde a primeira versão até a última.

- **Trabalho em equipe**: um *sistema de controle de versão* permite que diversas pessoas trabalhem sobre o mesmo conjunto de documentos ao mesmo tempo e minimiza o desgaste provocado por problemas com conflitos de edições. É possível que a implementação também tenha um controle sofisticado de acesso para cada usuário ou grupo de usuários.
- **Marcação e resgate de versões estáveis**: a maioria dos sistemas permite marcar onde é que o documento estava com uma versão estável, podendo ser facilmente resgatado no futuro.
- **Ramificação de projeto**: a maioria das implementações possibilita a divisão do projeto em várias linhas de desenvolvimento, que podem ser trabalhadas paralelamente, sem que uma interfira na outra.
- **Segurança**: Cada software de controle de versão usa mecanismo para evitar qualquer tipo de invasão de agentes infecciosos nos arquivos. Além do mais, somente usuários com permissão poderão mexer no código.
- **Rastreabilidade**: com a necessidade de sabermos o local, o estado e a qualidade de um arquivo; o controle de versão trás todos esses requisitos de forma que o usuário possa ser embasar do arquivo que deseja utilizar.
- **Organização**: Com o software é disponibilizado interface visual que pode ser visto todo arquivos controlados, desde a origem até o projeto por completo.
- **Confiança**: O uso de repositórios remotos ajuda a não perder arquivos por eventos imponderáveis. Além disso e disponível fazer novos projetos sem danificar o desenvolvimento.

Instalando o Git

O Git pode ser encontrado no endereço https://git-scm.com/downloads

Tutorial online de uso do Git

https://try.github.io/levels/1/challenges/1
https://githowto.com/pt-BR

Comandos básicos do Git

Init
Criar um novo diretório gerenciado
git init

Add
Adicionar um arquivo/diretório para o Git
git add <nome do arquivo/diretório>

Status
Usado para verificar o estado do repositório
git status

Commit

Confirmar o versionamento ou alteração de um arquivo ou diretório
git commit <nome do arquivo/diretório> -m “Breve comentário sobre o que foi feito”

Checkout
Reverte a última alteração feita no arquivo
git checkout <nome do branch>

Reset
Voltar ao estado anterior do arquivo tendo o repositório local como base
git reset HEAD <nome do arquivo>

Log
Verificar o histórico do repositório
git log

Pull
Atualizar os arquivos locais com os do repositório remoto
git pull

Push
Enviar os arquivos que foram feito o commit para o repositório remoto
git push

Branch
Criar uma ramificação do projeto atual
git branch <nome do branch>
Trocar de branch
git checkout <nome do branch>
Criar um novo branch e já troca do master para ele
git checkout -b <nome do branch>

Merge
Fazer a mescla de um branch com o master
git merge <branch a ser mesclado>

Clone
Clonar um repositório remoto para sua máquina
git clone <endereço do repositório remoto>

Tag
Cria um ponto de referência no projeto
git tag v1.0.0 -m “Descrição do que esta versão representa”

Existem duas formas de usar o Git no RStudio, são elas:

1. A criação de um repositório local que será posteriormente enviado para um servidor remoto;
2. Fazer o clone de um projeto já existente em um repositório Git remoto para o seu computador

Clonar um projeto Git já existente:

Use os comandos File -> New Project

Selecione a opção Version Control

Escolha a opção de clonar o projeto de um repositório Git:

Insira a URL do repositório, por padrão o RStudio já usa o nome da pasta desse projeto como o nome do diretório do projeto, para maior organização clone o projeto para dentro uma pasta como a do exemplo:

Clique em Criar Projeto

Criar um novo projeto com suporte ao Git

Usar os comandos File -> New Project

Selecione a opção Novo Diretório

Selecione a opção Novo Projeto

Atribua um nome ao diretório onde seu projeto será criado, este será o nome do seu projto, depois marque a opção Criar um Repositório Git

Clique em Criar Projeto

Usando o Git dentro do RStudio

Ao lado da aba onde ficam as variáveis da aplicação irá surgir uma nova aba Git

Marque os arquivos que deseja fazer o commit das alterações:

Crie uma mensagem que represente o que foi feito no campo Commit message, depois clique em commit.

Caso a mensagem abaixo aparece vá ao terminal dentro do RStudio e siga as instruções da mensagem.
Configure o seu nome e email para poder fazer commit

Caso tudo ocorra corretamente a mensagem abaixo será exibida

Ao final de seu dia de trabalho use o comando Push na aba Git para enviar para o servidor remoto todas as suas alterações