Comando GIT

Comando usado no Git:

Resumo:

1. **Create a new repository on the command line**

```
echo "# CSS-Display-Grids" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/julianojcs/CSS-Display-Grids.git
git push -u origin master
```

2. **Push an existing repository from the command line**

```
git remote add origin https://github.com/julianojcs/CSS-Display-Grids.git
git push -u origin master
```

3. **Import code from another repository**

```
You can initialize this repository with code from a Subversion, Mercurial, or TFS project:

-> https://github.com/julianojcs/CSS-Display-Grids/import
```

Import code

Fonte: https://woliveiras.com.br/posts/comandos-mais-utilizados-no-git/#Verificandoasconfiguraeslocais

[ ] - parte opcional do comando

Data:	
Comando	**Função**
git checkout [master]	Commits
git push origin [master]	Enviando as alterações para o servidor
git rm -r nome_do_arquivo_ou_pasta	Para remover pastas, é sempre necessário que ela esteja vazia ou que executemos o comando rm com o parâmetro -r para que a deleção seja recursiva.
git add .	Adiciona tudo de uma vez

git add nome_do_arquivo	Adiciona um arquivo específico ao repositório Quando adicionamos com o git add ainda não estamos persistindo os dados no histórico do Git, mas adicionando a uma área temporária onde podemos ficar levando e trazendo alterações até garantirmos que algo realmente deve ser salvo, então rodamos o git commit.
git commit -m "mensagem"	Para fazer um commit, precisamos adicionar uma mensagem ao pacote, então rodamos com o parâmetro -m "mensagem".
git commit -am "minha mensagem"	Adicionamos tudo de uma vez e já deixamos uma mensagem para o commit. -a: Adiciona o arquivo -m: Inclui mensagem
git status	Verificando o que foi alterado Para sabermos se tem algo que foi modificado em nossa branch, rodamos o comando git status.
git diff	Será retornada uma tela com o que foi adicionado escrito com um + e o que foi removido aparece com -.
git push origin master	Enviando as alterações para o servidor Depois que finalizadas as alterações, fechamos os commits, então enviar os commits para o servidor.
git log	Verificar o histórico das mudanças gravadas no repositório, ou seja, os commits efetuados, listando **todos** os commits.
git log -n 2	Listar apenas os "n" últimos commits.
git log --oneline	Lista resumo bem conciso dos commits do nosso projeto
git log --stat	Listar resumo dos arquivos alterados, com o número de linhas adicionadas e removidas O git status exibe arquivos que estão fora da área de stage, prontos para serem adicionados, e arquivos que estão dentro da área de stage, prontos para serem comitados. Já o git log exibe o histórico das mudanças efetivamente gravadas em um repositório. Ou seja, os commits efetuados.
git diff	Revisar amodificação efetuada, verificando as diferenças entre o arquivo alterado e o que foi comitado anteriormente.

	Serve apenas para exibir as mudanças ainda não rastreadas. O git diff não poderá ser utilizado para arquivos novos, que ainda não estão sendo rastreados pelo Git (ou seja, que ainda não tiveram o primeiro git add executado).
git diff --staged	Mostrar as diferenças entre os arquivos na área de stage e a última versão que foi comitada utilizando a opção
git diff 222cccc	Usando o código do último commit, podemos mostrar as alterações dentro e fora da stage. Vemos o código do último commit através do comando git log -n 1 --oneline • *222cccc é um exemplo de código de commit*
git diff 222cccc..8877887	Verificar as diferenças entre dois commits específicos. Não exibe modificações ainda não comitadas
git diff 8877887~2	Verifica a diferença entre os dois últimos commits. Exibe as mudanças nos arquivos do commit de código 8877887 em relação aos dois commits feitos imediatamente antes. O número depois do ~ indica quantos commits anteriores devem ser considerados na comparação Exibe modificações ainda não comitadas.
git rm nome_do_arquivo.html	Remoção do arquivo e adição na stage. Após a adição da remoção do arquivo na stage, temos que executar o commit para efetivamente remove-lo do repositório (git commit -m "Removendo arquivo"). Com a execução do comando git add --all , após apagarmos um arquivo fisicamente, a remoção será adicionada ao stage.
git mv estilos.css principal.css	Renomear arquivo no stage
git mv principal.js js/principal.js	Movendo arquivos no stage.
git checkout -- index.html	Desfaz as alterações que ainda não estão no stage (ainda não rastreadas). Também usado para recuperar arquivos removidos (apagados) acidentalmente.

	(apagados) acidentalmente.
git reset -- index.html	Retira o arquivo da stage mas preserva tudo o que foi modificado nesse arquivo
git reset	Retira **todos** o arquivo da stage
git reset --hard	Retirar todos os arquivos do stage e descartar todas as mudanças nos arquivos
git reset --hard 6111116	Desfaz as alterações de um commit específico.
git revert --no-edit 6111116	Voltar atrás, desfazendo as alterações no repositório. código 6111116 representa o último commit efetuado. No caso de passarmos um código de commit antigo, apenas as alterações feitas naquele commit serão desfeitas.
git revert --abort	Cancela a operação de revert
git revert --continue	Quando ocorre conflito no revert, deve-se corrigir o conflito manualmente e depois executar o --continue para aplicar o revert
git regert --skip	Quando ocorre conflito no revert, o --skip serve para ignorar/pular o conflito.
git init --bare meu-repositorio.git	Criar um repositório remoto. Oparâmetro --bare serve para que o Git não crie um working tree (diretório de trabalho), impedindo que commits sejam efetuados diretamente no servidor. Os commits deverão ser executados localmente pelos desenvolvedores, em seus computadores, e depois esses commits serão enviados e armazenados no repositório remoto, localizado no servidor. O Git criará um diretório com o nome do repositório e com a extenção .git (nome-do-repositorio.git)
git remote add origin file://192.168.1.1/opt/repositorios/moveis-ecologicos.git	Adiciona o repositório remoto. origin = nome do repositório remoto + url do repositório remoto
git remote -v	Listar repositórios remotos. - **v**: Lista a URL dos repositórios remotos
git remote rename nome-antigo	alterar o name de um repositório remoto

nome-novo	
git push origin master	Envia os commits para o repositório remoto. A palavra **master** que utilizamos no comando é o nome da branch principal do repositório.
git clone file://192.168.1.1/opt/repositorios/repositorio.git	Clona o repositório remoto
git pull origin master	Sincronizar do repositório local com o servidor
git branch	Listar as branches do nosso repositório
git branch novo-branch	Cria um novo branch (exemplo, para criarmos uma branch chamada design para trabalharmos na melhoria do design basta executarmos: git branch design
git checkout novo-branch	Trocar de branch
git checkout -b novo-branch	Cria um novo branch e já faz a troca para usá-la
git branch -d branch-existente	Apaga a branch informada. Para apagar uma branch, não podemos estar utilizando ela. Devemos trocar (fazer checkout) para outra branch antes de apaga-la. Não é possível deletar com a opção -d uma branch que possui commits ainda não aplicados em outras branches. Para removermos a branch loja se tivermos feito algum commit, devemos utilizar a opção -D
git branch -D branch-existente	Apaga a branch informada, independente se esta tem ou não commits ainda não aplicados.
git merge design -m "Mesclando com a branch design"	Juntar alterações de dois branchs
git rebase design	Serve para elevar uma branch que ficou para trás e aplicar as alterações que ocorreram em outra branch. Exemplo: Foi usada a branch design para incluir o uso do bootstrap no projeto. Depois de finalizado todas as alterações para incluir o bootstrap, decidiu-se que a inclusão estava pronta e que ela seria permanente. Dai, aplica-se o rebase para fazer com que todas as branches apontem para o ponto em que a implantação da referida alteração está. rebase = rebobinar

	rebase – rebobinar
git branch -r	Listar as branches remotas
git branch -a	listar as branches localis e remotas
git branch -v	listar as branches locais e remotas, e mostrar para quais commits as branches remotas estão apontando
git push origin design	Campartilhar branch local com o repositório remoto (GitHub) enviando para o repositório remoto os commits da branch referenciada no comando (neste caso design)
git fetch origin	Obter os commits da branch remota origin/master para outro repositório (linha 131 do livro)

Evernote