**eval `ssh-agent -s`**

Renomear repositório remoto: **git remeto rename <nome original> <nome que deseja>**

Adicionar repositório remoto: **git remote add <nome do repositório > <URL do repositório>**

Após ter adicionado o repositório remoto você pode enviar dados para lá, se a Branch que você passar não existir uma nova será criada: **git push <repositorio remoto> <branch>**

Para o merge você precisa estar na Branch em que deseja unir, por exemplo, se você quer trazer modificações da Branch dois para a Branch master faça:

**git checkout master** e depois **git merge dois**

Desfazer modificação em arquivos monitorados pelo git: **git checkout -- <arquivo>**

Desfazer git add: **git git reset HEAD <arquivo>**

Desfazer um commit: **git revert <hash> ou git revert HEAD~1**

Salvar suas alterações para serem commitadas mais tarde, com esse comando salva suas modificações para assim conseguir fazer um git pull sem perder nada e quando recuperado faz um merge das modificações salvas com o novo conteúdo:

* **Git stash**: salvou
* **Git stash list**: lista de itens salvos para mais tarde
* **Git stash pop**: traz as alterações salvas fazendo um merger e apaga da lista de stash

Navegar pelos commits: **git checkout <hash>.** Mas para salvar suas alterações a partir deste commit é preciso criar uma nova Branch.

Verificar a diferença entre commits**: git diff <hash>..<hash>**

**Git diff** mostra suas alterações feitas que ainda não foram add ao commitadas.

Criar uma tag e com isso marca uma versão da aplicação:

**Git tag -a <nome da tag> -m “sua mensagem”**

**Git push <repositorio> <nome da tag>** para enviar a tag para o repositório remoto, tags no repositório remoto geram um ponto onde pode baixar o zip do projeto desta versão.

**Git rebase -i HEAD~3** com isso junto os três últimos commits em um só. Atenção! Rebase devem ser feitos somente no local, depois de enviados ao repositório remoto não utilize.

**Git cherry-pick <hash>** junta esse commit de outra branch no HEAD de outra breanch.

**Git bisect start** inicia uma busca pelo commit que introduziu um bug, indicando qual é o bom e o ruim o git vai atras do commit, continuo indicando o commit como ruim caso o que ele retorne não esteja correto até que ele acerte.

**Git bisect bad <hash>**

**Git bisect good <hash>**

Encontrando o que desejo para reverte **git bisect reset**

**Git blame <arquivo>** mostra todas as alterações feita nesse arquivo e quem as fez.