GIT FLOW

O GitFlow é um conjunto de programas embarcados no GIT CMD e permite o controle automático de situações que envolvem múltiplos níveis de Branch.

Ele evita esquecimentos e atualizações erradas. Impede que manutenções que ainda não estão efetivamente validadas sejam movidas para o diretório de fontes que são parte do produto de software estável, o qual é liberado para usuários.

-Vamos trabalhar com as seguintes BRANCHES que são apropriadas para grandes projetos:

***=> ReleaseManufacture (nossa Master onde estão os fontes prontos para liberação para uso);***

***=> ReleaseCandidate (contém os fontes liberados pelos desenvolvedores para testes finais). Essa BRANCH acumula as mudanças promovidas por todos programadores no software;***

***=> Development-NOME DA TAREFA DE DESENVOLVIMENTO (contém os fontes que estão ainda em programação e testes pelos desenvolvedores). Essas BRANCHES devem ser excluídas quando a tarefa de produção do desenvolvedor acabar.***

-Execute o Git CMD

-Vá até uma pasta onde criaremos uma sub pasta com o nome gitflowExemplo (use os comandos do DOS – MD, CD, DIR para criar e conferir a criação)

-Execute o comando git flow init para colocar o novo repositório dentro do controle GIT



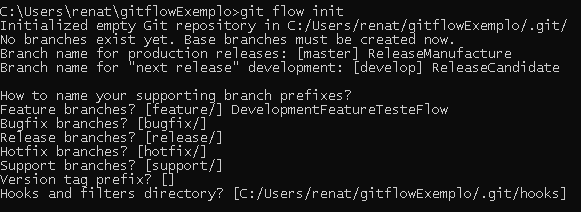
-Aguarde a iniciação da pasta e depois, informe o nome da BRANCH que conterá os fontes na versão pronta para gerar a Release e fazer Deploy para liberar o uso (o nome default é Master e vamos renomear para ReleaseManufacture)



-Nomeie em seguida a BRANCH onde estarão os fontes prontos para testes finais pré-liberação (o nome default é Development e vamos renomear para ReleaseCandidate)



-Deixe o padrão feature/ como prefixo de BRANCHES de manutenção (já sugerido pelo Git Flow) ou mude, se desejar e aceite o Bugfix (dê ENTER) e não vamos acrescentar TAGs



-Ao terminar, o GitFlow vai posicionar você, automaticamente, no diretório de manutenção/feature



-Veja as Branches existentes com o comando git branch



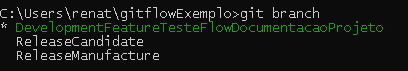
Perceba que dentro da sua pasta de tarefa chamada gitflowExemplo, existe a Branch ReleaseCandidate (Master) e a ReleaseManufacture (fontes liberados par os testes finais), as quais você pode usar.

-Crie agora uma feature que corresponde àquilo que você como desenvolvedor precisa realizar para cumprir uma tarefa de programação ou documentação de software que lhe foi designada



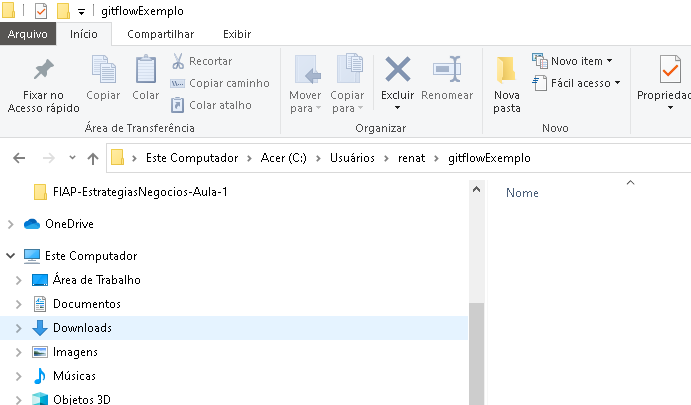


-Confira a criação da Branch e veja que o GitFlow já posicionou você na Branch de programação/documentação.

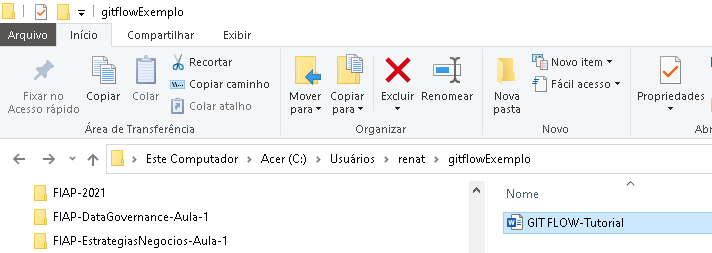


-Agora, você pode usar editores de linguagem ou documentos, navegar via Windows Explorer, por exemplo e copiar, editar, criar arquivos à vontade. O Git vai mapear tudo o que você fizer no diretório do projeto, vinculando com a Branch de tarefa

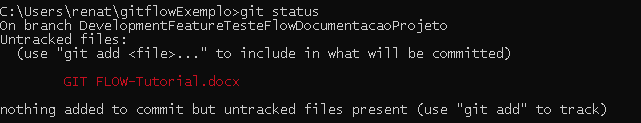
***C:\Users\renat\gitflowExemplo>***



-Experimente copiar um arquivo qualquer para dentro da pasta e daí para frente, podemos aplicar os comandos do Git Command já estudados (status, add, commit, push, pull, ...)



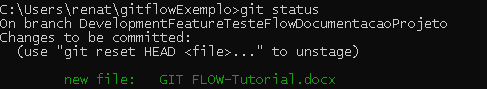
-Git status identificará as mudanças em arquivos



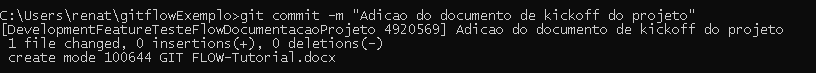
-Adicione o arquivo para coloca-lo na lista de arquivos prontos para commit, com o Git add



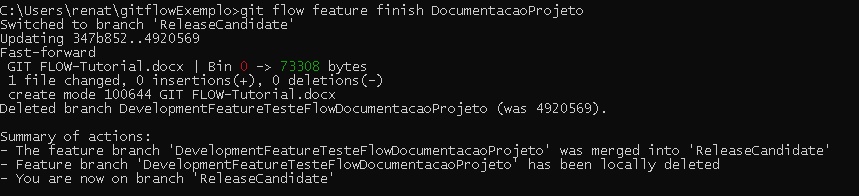
-Confira se o arquivo foi adicionado com Git status



-Agora, faça o commit para que o GitFlow publique o seu novo arquivo/versão nova, na branch de testes finais



-Informe agora que você finalizou o desenvolvimento da sua tarefa/feature para que o Gitflow leve as atualizações sozinho para a Branch “superior” – a Branch de testes finais chamada ReleaseCandidate. Para isso, execute o comando Git flow feature finish ***nome da feature/branch terminada***

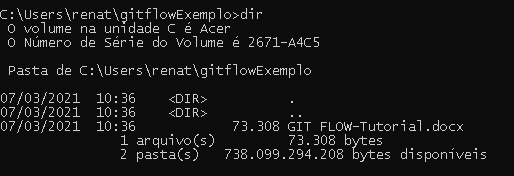


-o GitFlow já deleta a sua Branch de manutenção/tarefa e atualiza a Branch superior. Ele também posiciona você na Branch superior. Confira as mudanças na Branch superior com Git branch e Git status



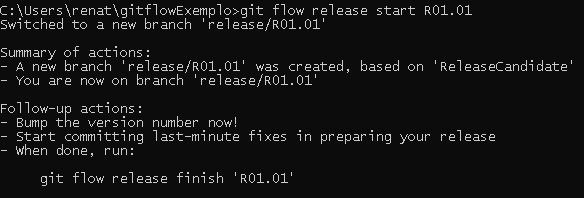


Perceba que o arquivo que você alterou/criou/excluiu, já foi modificado e “comitado” na Branch superior. Confira com o comando Dir

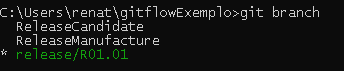


-Se você executar git flow feature start de novo, o GitFlow vai criar uma nova Branch de manutenção com as últimas versões da ReleaseCandidate

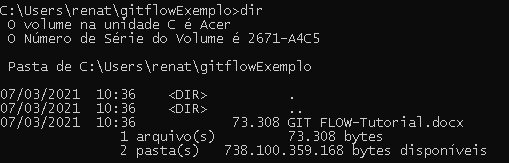
-Quando as manutenções ligadas a uma Release acabarem, você deve disparar a geração da Release com o comando Git flow release start ***número da release***



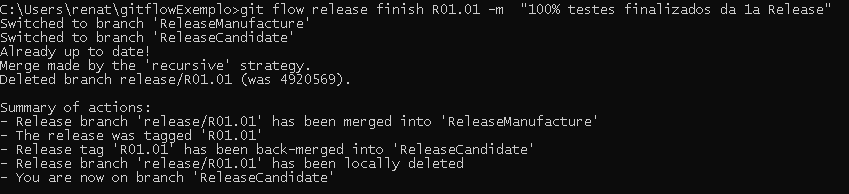
-Nesse momento o GitFlow criou uma Branch de release para que o software receba a carga de testes finais, pré-liberação. Confira com Git branch.



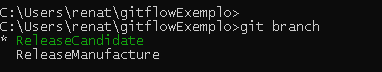
Quando você acessar os arquivos da pasta de projeto, agora estará operando sobre a branch da release



-Quando acabar os testes, feche a Release de forma parecida com o que fez com a Feature: Git flow release finish ***nome da release a fechar*** -m “mensagem que descreve a release”



-Agora, a Branch da Release é destruída e você vai para a ReleaseCandidate ,automaticamente. Confira com o comando Git branch



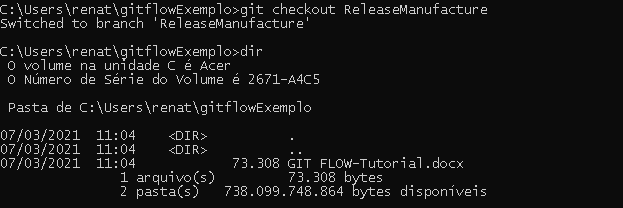
-OBS: SE VOCÊ ESQUECER DE COLOCAR O -M “MENSAGEM”, O GIT ABRIRÁ UM EDITOR PARA VOCÊ COLOCAR OS COMENTÁRIOS. ADICIONE O QUE DESEJAR NO TEXTO E DEPOIS DIGITE “ESC” E EM SEGUIDA “:wq!” E ENTER DE NOVO.

-Execute Git tag para verificar que o Git já vinculou a identificação da Release com o conteúdo que se encontra na ReleaseCandidate.

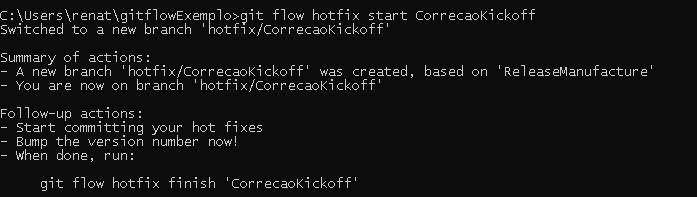




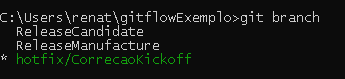
-Quando você terminou a Branch de release, o GitFlow TAMBÉM ATUALIZOU A MASTER (ReleaseManufacture). Confira, mudando para a Branch ReleaseManufacture e listando os arquivos com DIR. Você pode também usar o Explorer do Windows para conferir os arquivos publicados na cópia mestre.



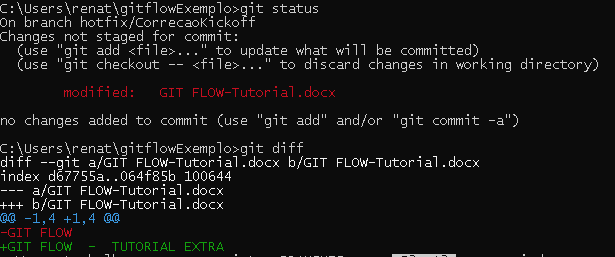
-Se você precisar fazer um **HOTFIX** em um fonte que já está em produção (correção rápida de um BUG), você pode criar uma Branch diretamente à partir da Master e publicar diretamente na Master quando terminar.

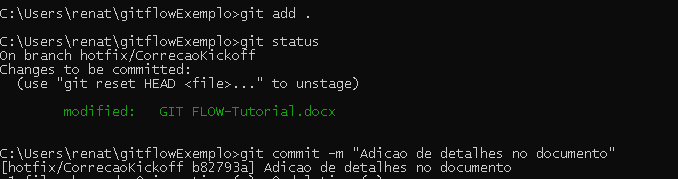


-O Git vai mover você para o diretório de correção. Confira com Git branch.

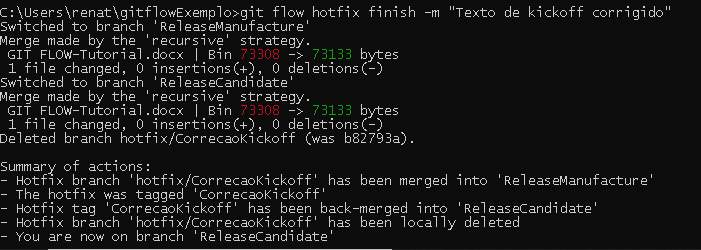


-Modifique o que precisa (execute git add e commit para confirmar as mudanças)





- Ao terminar, finalize o Hot fix com Git flow hotfix finish -m ***“mensagem sobre a manutenção”***



Veja de a Branch de hotfix foi excluída automaticamente.

**-Após finalizar o Hot fix, o Git flow atualiza automaticamente a cópia Master (nossa ReleaseManufacture) e a cópia de teste finais (ReleaseCandidate), da mesma forma que acontece quando você finaliza um desenvolvimento normal, usando releases tradicionais (não hot fix). Você pode checar isso, navegando de uma branch para outra e observando o horário de atualização do arquivo corrigido e/ou observando o seu conteúdo.**

