**Benefícios do GitFlow**

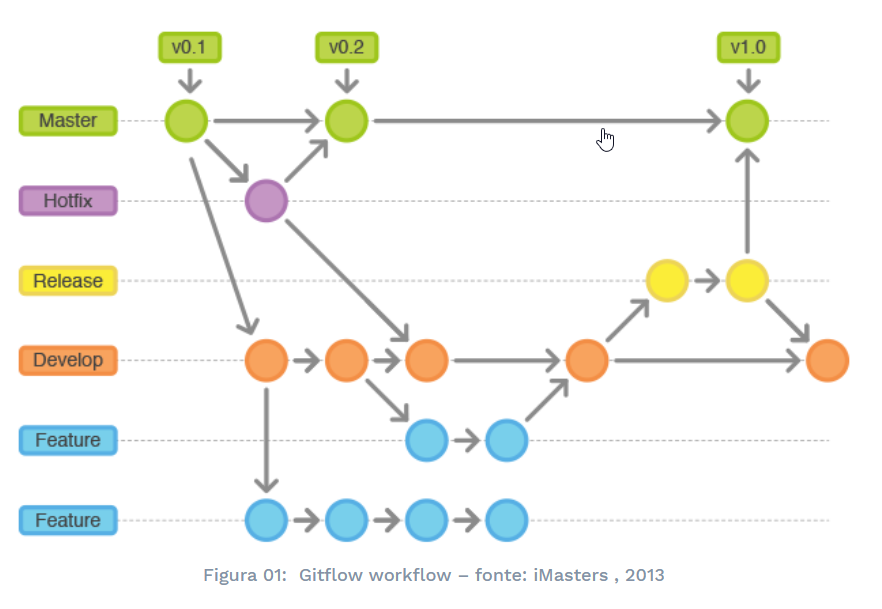
Poderíamos aqui descrever que o grande benficio do Gitflow é o versionamento do código, que permite validar as alterações realizadas e voltar rapidamente a uma determinada versão do código de seu projeto e a publicação automática em produção.

Mas não é só isso, o git flow permite que você tenha todo o gerenciamento de código, por partes menores.

É comum que os projetos comecem a crescer e os códigos ficarem bloqueados por conta de novas features e problemas, consequências da própria evolução do produto.

Além de versionar como dito anteriormente, pode se trabalhar em versões especificas sem atrapalhar a entrega final das novas features. Ou seja, torna-se menos caótico as implentações de correções enquanto trabalha-se no lançamento de novas funcionalidades.

Exemplo:



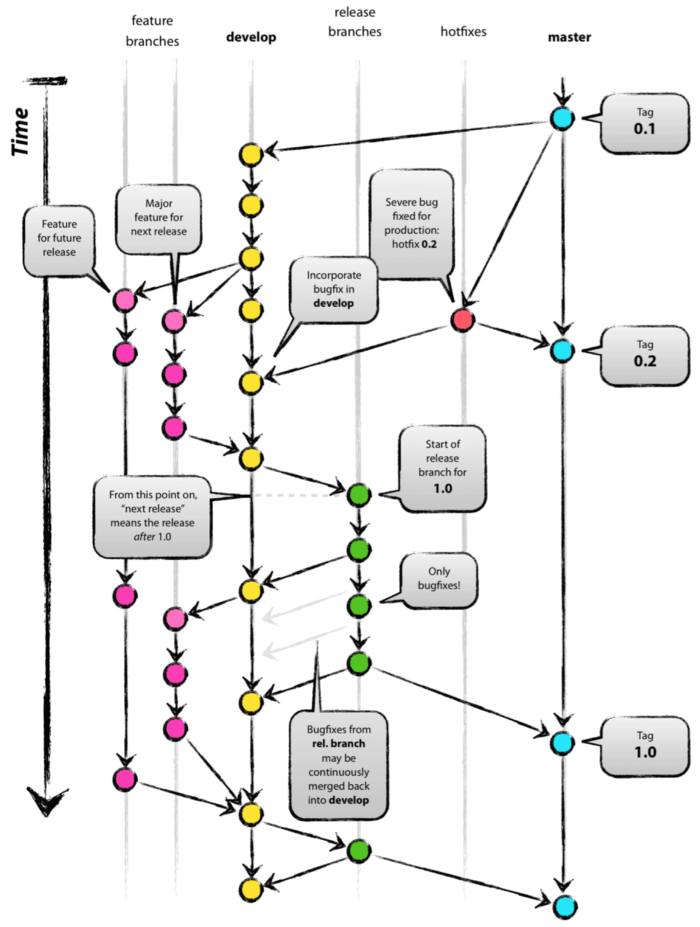
Podemos assim garantir que o grande benefício do git flow é o seu gerenciamento de releases, sem a necessidade de bloqueio de código e as ramificações de componentes para desenvolvedores, sem perder a parar o desenvolvimento por bloqueio de códigos.

Além disso, publicar em um repositório único e acessível para publicação do projeto em produção de forma automática.

Entretanto, é muito comum vermos pessoas/colaboradores utilizando uma única branch para fazer commits em seus respectivos projetos pessoais. Não é que isso seja errado, é muito tranquilo de controlar tudo em uma única branch quando se está desenvolvendo uma solução sozinho, mas de acordo que quando temos que interagir com mais contribuidores o cenário muda bastante, seja em um repositório opensource ou privado.

Contudo, é de suma importância que se tenha total controle do que está sendo desenvolvido por sua equipe, onde, ao mesmo tempo são corrigidos falhas, implementado novas funcionalidades e ter o seu código de produção com total funcionamento conforme tem de ser entregue ao seu respectivo cliente.

O fluxo de git flow abaixo ajuda a ilustrar melhor o funcionamento de como deverá ser todo o processo de desenvolvimento e controle das atividades:



De acordo com a figura acima, a branch **master** irá contér todo código já testado, homologado, versionado que será entregue ao cliente e a branch develop é onde todo fluxo de trabalho irá ocorrer antes de fazer o release versionado que será feito merge na master.

A branch **develop** deve sempre conter o código mais atual que está sendo produzido no dia a dia da equipe de trabalho, onde as branchs de features serão ramificadas tendo ela como base.