git-flow cheatsheet

created by Daniel Kummer

efficient branching using git-flow by *Vincent Driessen*

translations: *English* - *Castellano* - *Português Brasileiro* - *繁體中文 (Traditional Chinese)* - *简体中文(Simplified Chinese)* - *日本語* - *Türkçe* - *한국어 (Korean)* - *Français* - *Italiano* - *Nederlands* - *Русский (Russian)* - *Deutsch (German)* - *Català (Catalan)* - *Română (Romanian)* - *Ελληνικά (Greek)* - *Українська (Ukrainian)* - *Tiế ng Việt (Vietnamese)* - *Polski* - *فارسی* - *العربية* - *Lietuviškai (Lithuanian)* - *Azərbaycanca (Azerbaijani)* *Bahasa Indonesia*

About

git-flow are a set of git extensions to provide high-level repository operations for Vincent Driessen's branching model.

more

★★★

This cheatsheet shows the basic usage and effect of git-flow operations

★★★

Basic tips

★ Git flow provides excellent command line help and output. Read it carefully to see what's happening...
★ The macOS/Windows Client *Sourcetree* is an excellent git gui and provides git-flow support
★ Git-flow is a merge based solution. It doesn't rebase feature branches.

★★★

Setup

★ You need a working git installation as prerequisite.
★ Git flow works on macOS, Linux and Windows

★★★

macOS

Homebrew

```
$ brew install git-flow-avh
```

Macports

```
$ port install git-flow-avh
```

Linux

```
$ apt-get install git-flow
```

Windows (Cygwin)

```
$ wget -q -O - --no-check-certificate https://raw.github.com/petervanderdoes/gitflow-avh/develop/contrib/gitflow-installer.sh install stable | bash
```

You need wget and util-linux to install git-flow.

For detailed git flow installation instructions please visit the *git flow wiki*.

Getting started

Git flow needs to be initialized in order to customize your project setup.

★★★

Initialize

Start using git-flow by initializing it inside an existing git repository:

```
git flow init
```

You'll have to answer a few questions regarding the naming conventions for your branches.
It's recommended to use the default values.

Features

★ Develop new features for upcoming releases
★ Typically exist in developers repos only

★★★

Start a new feature

Development of new features starting from the 'develop' branch.

Start developing a new feature with

```
git flow feature start MYFEATURE
```

This action creates a new feature branch based on 'develop' and switches to it

Finish up a feature

Finish the development of a feature. This action performs the following

★ Merges MYFEATURE into 'develop'
★ Removes the feature branch
★ Switches back to 'develop' branch

```
git flow feature finish MYFEATURE
```

Publish a feature

Are you developing a feature in collaboration?
Publish a feature to the remote server so it can be used by other users.

```
git flow feature publish MYFEATURE
```

Getting a published feature

Get a feature published by another user.

```
git flow feature pull
origin MYFEATURE
```

You can track a feature on origin by using

```
git flow feature track
MYFEATURE
```

Make a release

★ Support preparation of a new production release
★ Allow for minor bug fixes and preparing meta-data for a release

★ ★ ★

Start a release

To start a release, use the git flow release command. It creates a release branch created from the 'develop' branch.

```
git flow release start
RELEASE [BASE]
```

You can optionally supply a `[BASE]` **commit sha-1 hash to start the release from. The commit must be on the 'develop' branch.**

★★★

It's wise to publish the release branch after creating it to allow release commits by other developers. Do it similar to feature publishing with the command:

```
git flow release publish
RELEASE
```

(You can track a remote release with the

`git flow release track RELEASE`

command)

Finish up a release

Finishing a release is one of the big steps in git branching. It performs several actions:

★ Merges the release branch back into 'master'

★ Tags the release with its name

★ Back-merges the release into 'develop'

★ Removes the release branch

```
git flow release finish RELEASE
```

Don't forget to push your tags with `git push origin --tags`

Hotfixes

★ Hotfixes arise from the necessity to act immediately upon an undesired state of a live production version

★ May be branched off from the corresponding tag on the master branch that marks the production version.

★★★

git flow hotfix start

Like the other git flow commands, a hotfix is started with

```
git flow hotfix start
VERSION [BASENAME]
```

The version argument hereby marks the new hotfix release name. Optionally you can specify a basename to start from.

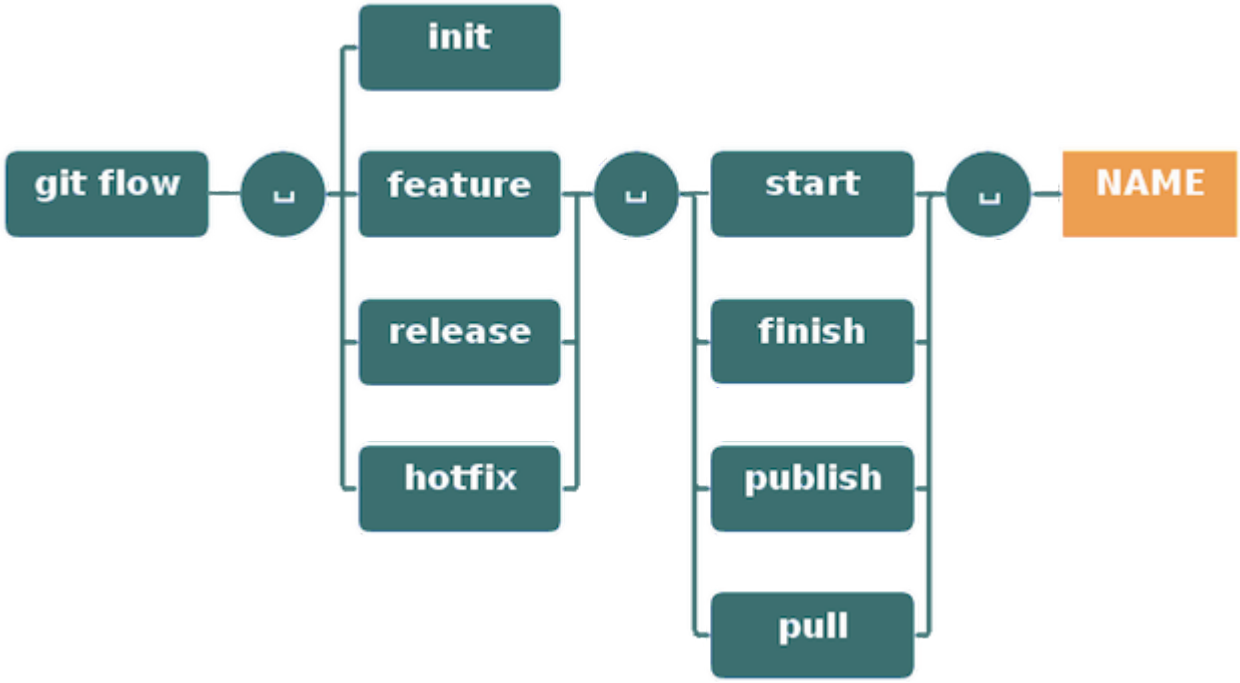
Finish a hotfix

By finishing a hotfix it gets merged back into develop and master. Additionally the

master merge is tagged with the hotfix version.

```
git flow hotfix finish
VERSION
```

Commands



Backlog

★★★

★ Not all available commands are covered here, only the most important ones
★ You can still use git and all its commands normally as you know them, git flow is only a tooling collection
★ The 'support' feature is still beta, using it is not advised
★ If you'd like to supply translations I'd be happy to integrate them