

# 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**

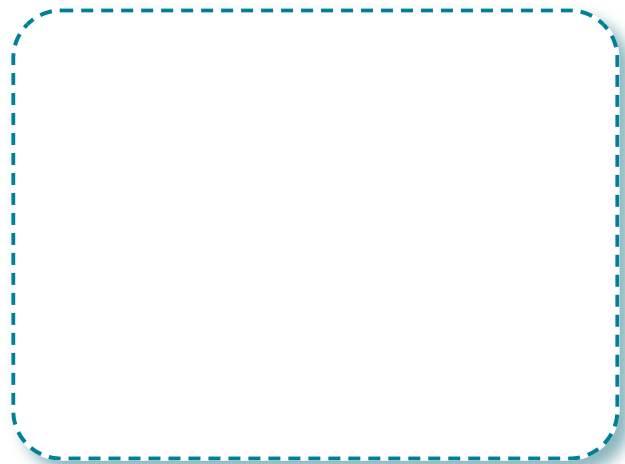
**Back-merges the release into 'develop'**

**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]
```

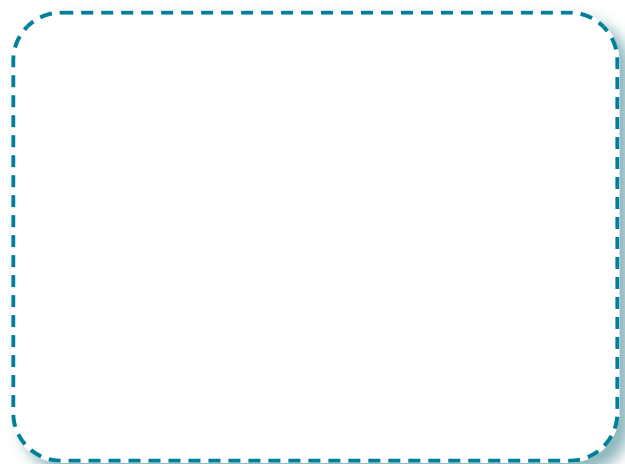


**The version argument hereby marks the new hotfix release name. Optionally you can specify a basename to start from.**

## 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

