The Basics of Git

Siberian Coder

Basic setup

```
# setup a username
git config --global user.name  "YOU"

# setup an email
git config --global user.email "YOU@EXAMPLE.COM"

# set main as a default branch name
git config --global init.defaultBranch main
```

SSH Key

```
# generate a new key
ssh-keygen -t ed25519 -C "YOU@EXAMPLE.COM"

# test ssh connection
ssh -T git@github.com
```

Aliases

```
git config --global alias.co      checkout
git config --global alias.br      branch
git config --global alias.ci      commit
git config --global alias.sw      switch
git config --global alias.ft      fetch
git config --global alias.amend   ’commit --amend --no
  -edit’
git config --global alias.lg      ’log --graph --
  decorate --oneline --all’
git config --global alias.save    ’!f() { git add -A
  && git commit -m "$*"; }; f’
git config --global alias.pp      ’!git pull && git
  push’
git config --global alias.undo    ’reset --soft HEAD~1
  ’
git config --global alias.rbi     ’rebase -i’
git config --global alias.unstage ’restore --staged’
```

Start a new repo

```
# initialize a new repo locally
git init

# add link to remote repo (origin)
git remote add origin git@github.com:user/repo.git
```

Clone an existing repo

```
git clone git@github.com:user/repo.git
```

Branching and Merging

```
# create a new branch
git branch <branch name>

# switch to an existing branch
git switch <branch name>

# or
git checkout <branch name>

# create a new branch and switch
git switch -c <branch name>

# or
git checkout -b <branch name>

# delete local branch
git branch -d <branch name>

# force delete local branch (-D == -d -f)
git branch -D <branch name>

# merge <my branch> into <target branch>
git switch <target branch>
git merge <my branch>

# example -- merging feature branch into main
git switch main
git merge feature
```

Basic Snapshotting

```
# show the working tree status
git status

# add file contents to the index
git add <file|.>

# record changes to the repository
git commit -m "<message>"

# delete latest commit
git reset HEAD~1
```

Sharing and Updating Projects

```
# sync the repo index with origin
git fetch

# get changes from origin
git pull

# get changes from the modified branch (someone made a
   rebase)
git pull --rebase

# first push of the branch
git push -u origin <branch name>

# simple push to a tracked branch
git push

# safe push after local rebase/amend -- replace a
  remote branch if there are no changes
git push --force-with-lease

# force push after local rebase/amend -- replace a
  remote branch
git push --force
```

Patching

Warning

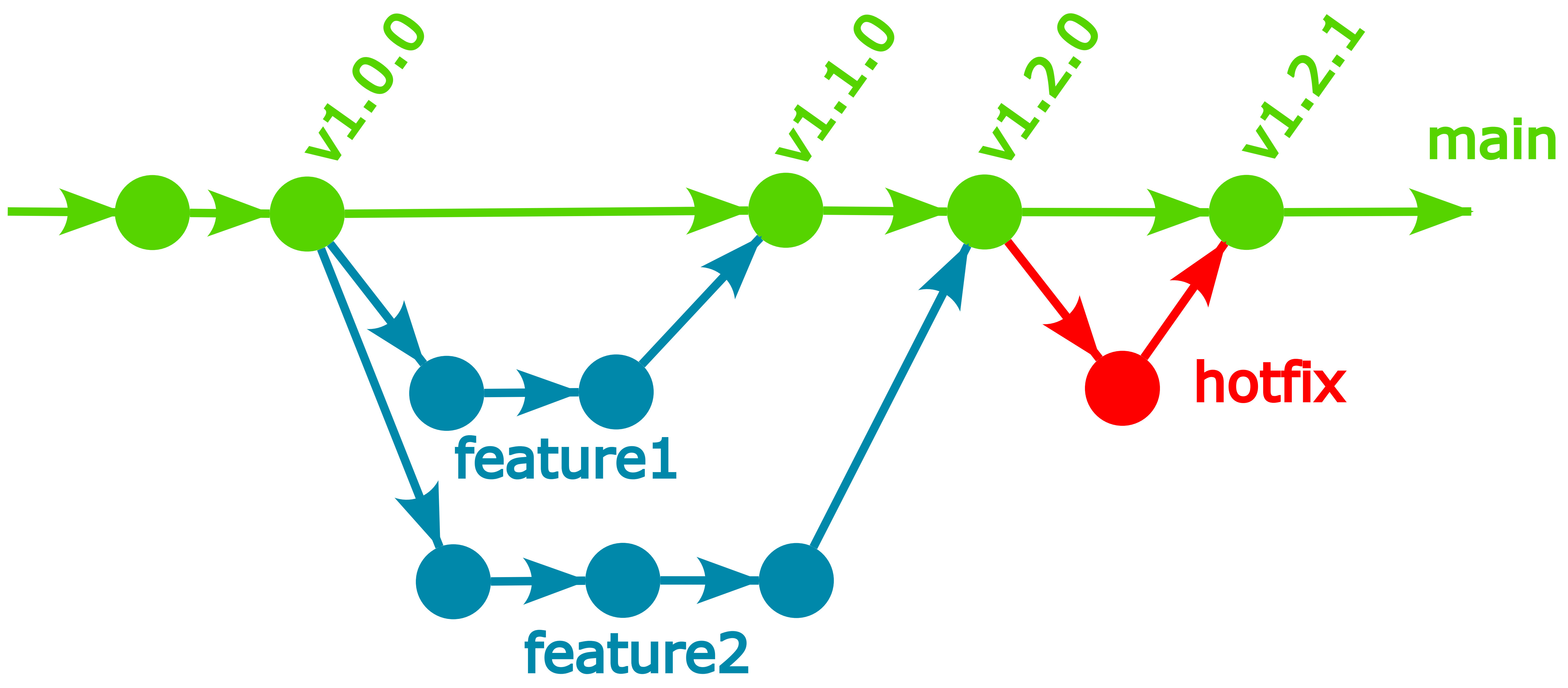
Do not patch the **main** branch! Never!

```
# move a branch on top ot another
git rebase <target>

# example -- rebase <feature> to <main>
git switch feature
git rebase main

# change the latest commit.
git commit --amend --no-edit

# modify, glue, skip N latest commits
git rebase -i HEAD~N
```

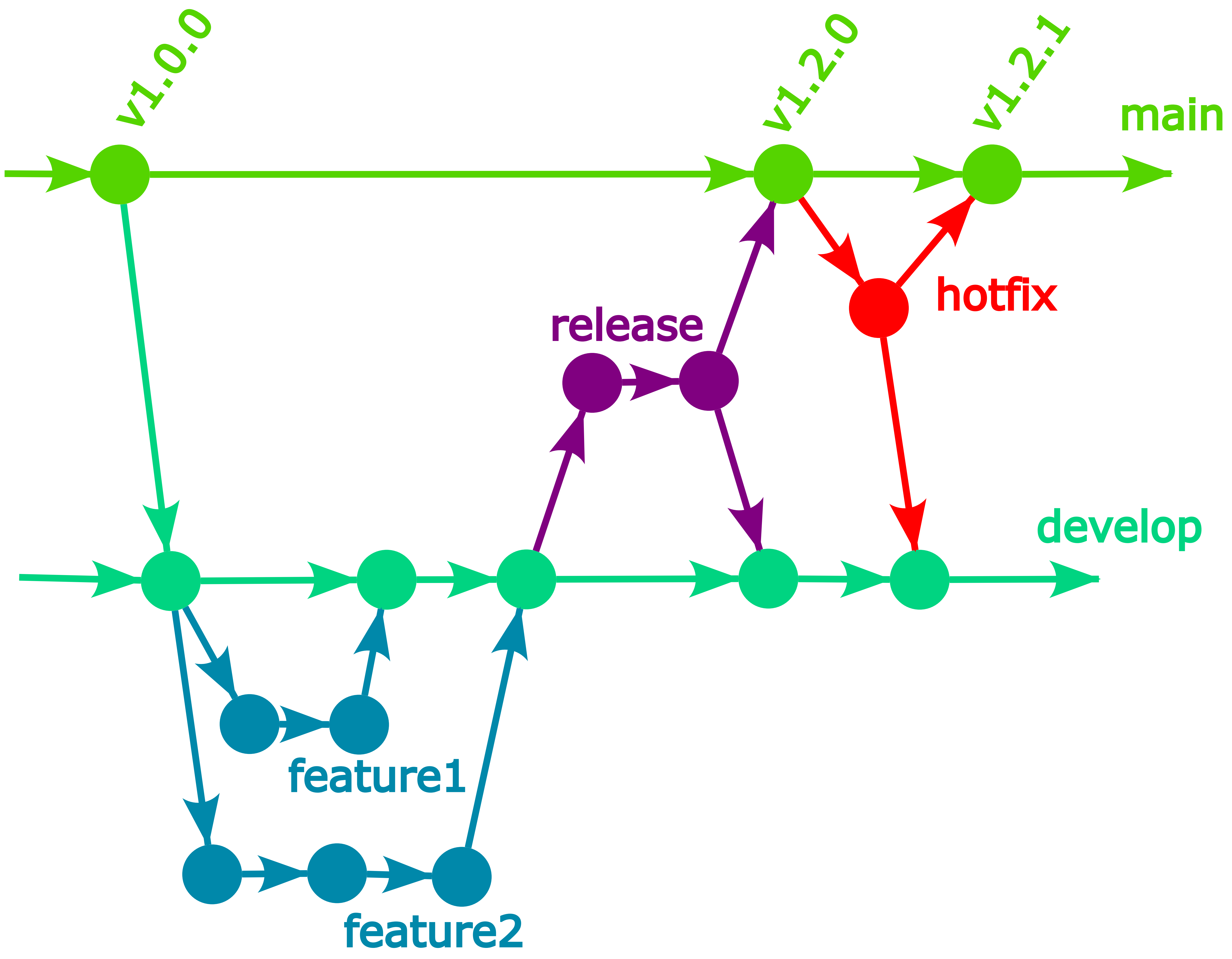
References

- Git Docs
- Pro Git Book

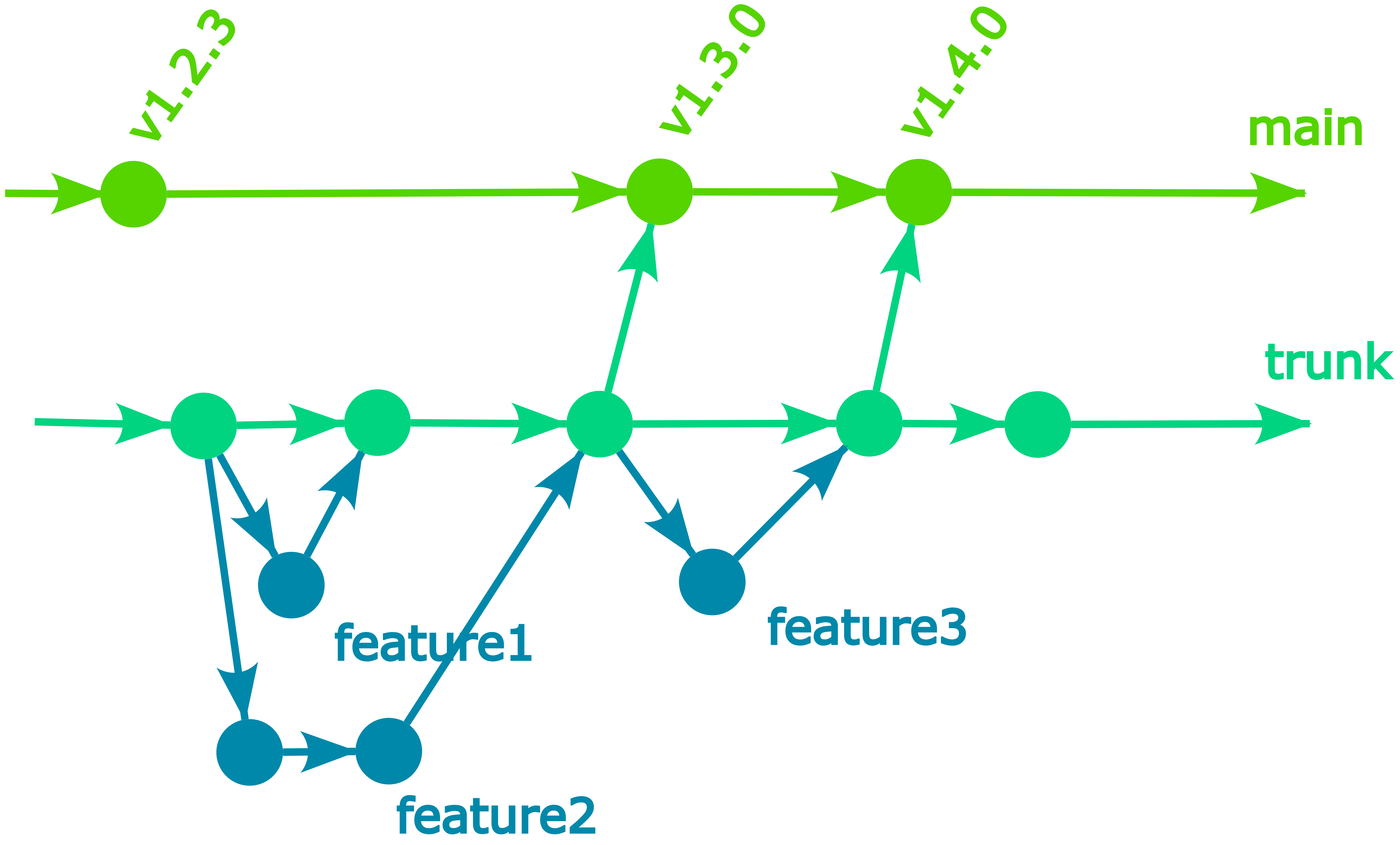
GitHub Flow



Git Flow


v1.0.0
v1.2.0
v1.2.1
main
hotfix
release
develop
feature1
feature2

Trunk Based Development


v1.2.3
v1.3.0
v1.4.0
main
trunk
feature1
feature3
feature2