git plugin

The git plugin provides many aliases and a few useful functions.

To use it, add `git` to the plugins array in your zshrc file:

```
plugins=(... git)
```

Aliases

Alias	Command
g	git
ga	git add
gaa	git add --all
gapa	git add --patch
gau	git add --update
gav	git add --verbose
gap	git apply
gapt	git apply --3way
gb	git branch
gba	git branch -a
gbd	git branch -d
gbda	git branch --no-color --merged | command grep -vE "^(+|*|\s*($(git_main_branch)\|development\|develop\|devel\|dev)\s*$)" | command xargs -n 1 git branch -d
gbD	git branch -D
gbl	git blame -b -w
gbnm	git branch --no-merged
gbr	git branch --remote
gbs	git bisect
gbsb	git bisect bad
gbsg	git bisect good
gbsr	git bisect reset
gbss	git bisect start
gc	git commit -v
gc!	git commit -v --amend
gcn!	git commit -v --no-edit --amend
gca	git commit -v -a
gca!	git commit -v -a --amend

Alias	Command
gcan!	git commit -v -a --no-edit --amend
gcans!	git commit -v -a -s --no-edit --amend
gcam	git commit -a -m
gcsm	git commit -s -m
gcb	git checkout -b
gcf	git config --list
gcl	git clone --recurse-submodules
gclean	git clean -id
gpristine	git reset --hard && git clean -dffx
gcm	git checkout $(git_main_branch)
gcd	git checkout develop
gcmsg	git commit -m
gco	git checkout
gcount	git shortlog -sn
gcp	git cherry-pick
gcpa	git cherry-pick --abort
gcpc	git cherry-pick --continue
gcs	git commit -S
gd	git diff
gdca	git diff --cached
gdcw	git diff --cached --word-diff
gdct	git describe --tags $(git rev-list --tags --max-count=1)
gds	git diff --staged
gdt	git diff-tree --no-commit-id --name-only -r
gdnolock	git diff $@ ":(exclude)package-lock.json" ":(exclude)*.lock"
gdv	git diff -w $@ | view -
gdw	git diff --word-diff
gf	git fetch
gfa	git fetch --all --prune
gfg	git ls-files | grep
gfo	git fetch origin
gg	git gui citool
gga	git gui citool --amend

Alias	Command
ggf	git push --force origin $(current_branch)
ggfl	git push --force-with-lease origin $(current_branch)
ggl	git pull origin $(current_branch)
ggp	git push origin $(current_branch)
ggpnp	ggl && ggp
ggpull	git pull origin "$(git_current_branch)"
ggpur	ggu
ggpush	git push origin "$(git_current_branch)"
ggsup	git branch --set-upstream-to=origin/$(git_current_branch)
ggu	git pull --rebase origin $(current_branch)
gpsup	git push --set-upstream origin $(git_current_branch)
ghh	git help
gignore	git update-index --assume-unchanged
gignored	git ls-files -v | grep "^[[:lower:]]"
git-svn-dcommit-push	git svn dcommit && git push github $(git_main_branch):svntrunk
gk	gitk --all --branches
gke	gitk --all $(git log -g --pretty=%h)
gl	git pull
glg	git log --stat
glgp	git log --stat -p
glgg	git log --graph
glgga	git log --graph --decorate --all
glgm	git log --graph --max-count=10
glo	git log --oneline --decorate
glol	git log --graph --pretty='%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset'
glols	git log --graph --pretty='%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset' --stat
glod	git log --graph --pretty='%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%ad) %C(bold blue)<%an>%Creset'
glods	git log --graph --pretty='%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%ad) %C(bold blue)<%an>%Creset' --date=short
glola	git log --graph --pretty='%Cred%h%Creset -%C(auto)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset' --all
glog	git log --oneline --decorate --graph
gloga	git log --oneline --decorate --graph --all
glp	git log --pretty=\<format>

Alias	Command
gm	git merge
gmom	git merge origin/$(git_main_branch)
gmt	git mergetool --no-prompt
gmtvim	git mergetool --no-prompt --tool=vimdiff
gmum	git merge upstream/$(git_main_branch)
gma	git merge --abort
gp	git push
gpd	git push --dry-run
gpf	git push --force-with-lease
gpf!	git push --force
gpoat	git push origin --all && git push origin --tags
gpu	git push upstream
gpv	git push -v
gr	git remote
gra	git remote add
grb	git rebase
grba	git rebase --abort
grbc	git rebase --continue
grbd	git rebase develop
grbi	git rebase -i
grbm	git rebase $(git_main_branch)
grbs	git rebase --skip
grev	git revert
grh	git reset
grhh	git reset --hard
groh	git reset origin/$(git_current_branch) --hard
grm	git rm
grmc	git rm --cached
grmv	git remote rename
grrm	git remote remove
grs	git restore
grset	git remote set-url
grss	git restore --source

Alias	Command
grt	cd "$(git rev-parse --show-toplevel || echo .)"
gru	git reset --
grup	git remote update
grv	git remote -v
gsb	git status -sb
gsd	git svn dcommit
gsh	git show
gsi	git submodule init
gsps	git show --pretty=short --show-signature
gsr	git svn rebase
gss	git status -s
gst	git status
gsta	git stash push
gsta	git stash save
gstaa	git stash apply
gstc	git stash clear
gstd	git stash drop
gstl	git stash list
gstp	git stash pop
gsts	git stash show --text
gstu	git stash --include-untracked
gstall	git stash --all
gsu	git submodule update
gsw	git switch
gswc	git switch -c
gts	git tag -s
gtv	git tag | sort -V
gtl	gtl(){ git tag --sort=-v:refname -n -l ${1}* }; noglob gtl
gunignore	git update-index --no-assume-unchanged
gunwip	git log -n 1 | grep -q -c "--wip--" && git reset HEAD~1
gup	git pull --rebase
gupv	git pull --rebase -v
gupa	git pull --rebase --autostash

Alias	Command
gupav	git pull --rebase --autostash -v
glum	git pull upstream $(git_main_branch)
gwch	git whatchanged -p --abbrev-commit --pretty=medium
gwip	git add -A; git rm $(git ls-files --deleted) 2> /dev/null; git commit --no-verify --no-gpg-sign -m "--wip-- [skip ci]"
gam	git am
gamc	git am --continue
gams	git am --skip
gama	git am --abort
gamscp	git am --show-current-patch

Main branch preference

Following the recent push for removing racially-charged words from our technical vocabulary, the git plugin favors using a branch name other than `master`. In this case, we favor the shorter, neutral and descriptive term `main`. This means that any aliases and functions that previously used `master`, will use `main` if that branch exists. We do this via the function `git_main_branch`.

Deprecated aliases

These are aliases that have been removed, renamed, or otherwise modified in a way that may, or may not, receive further support.

Alias	Command	Modification
gap	`git add --patch`	new alias `gapa`
gcl	`git config --list`	new alias `gcf`
gdc	`git diff --cached`	new alias `gdca`
gdt	`git difftool`	no replacement
ggpull	`git pull origin $(current_branch)`	new alias `ggl` (`ggpull` still exists for now though)
ggpur	`git pull --rebase origin $(current_branch)`	new alias `ggu` (`ggpur` still exists for now though)
ggpush	`git push origin $(current_branch)`	new alias `ggp` (`ggpush` still exists for now though)
gk	`gitk --all --branches`	now aliased to `gitk --all --branches`
glg	`git log --stat --max-count = 10`	now aliased to `git log --stat --color`
glgg	`git log --graph --max-count = 10`	now aliased to `git log --graph --color`
gwc	`git whatchanged -p --abbrev-commit --pretty = medium`	new alias `gwch`

Functions

Current

Command	Description
`grename <old> <new>`	Rename `old` branch to `new`, including in origin remote

Command	Description
current_branch	Return the name of the current branch
git_current_user_name	Returns the `user.name` config value
git_current_user_email	Returns the `user.email` config value
git_main_branch	Returns the name of the main branch: `main` if it exists, `master` otherwise

Work in Progress (WIP)

These features allow to pause a branch development and switch to another one (*"Work in Progress"*, or wip). When you want to go back to work, just unwip it.

Command	Description
work_in_progress	Echoes a warning if the current branch is a wip
gwip	Commit wip branch
gunwip	Uncommit wip branch

Deprecated functions

Command	Description	Reason
current_repository	Return the names of the current remotes	Didn't work properly. Use `git remote -v` instead (`grv` alias)