All experience

mrzlanx532

5 февраля 2020 г.

Оглавление

Глава 1

Git commands

1.1 Git common's commands

git help <команда>
открывает документацию по команде

git init
инициализация git-репозитория

1.2 Git config

git config --global core.editor *редактор кода*
установка стандартного редактора кода

git config --list
список всех настроек локально (или глобально --global)

git config --global alias.co checkout
создание краткого алияса (команда checkout будет выполняться через команду "co")

git config --global alias.co 'checkout -b'
создание краткого алияса "co" с параметрами

git config core.fileMode false
выключить отслеживания за ПРАВАМИ ФАЙЛА

1.3 Git status

git status
проверка на существование измененных файлов

git status -s (--short)
краткий вывод состояния файлов

1.4 Git add

git add *
Добавить все файлы в индексацию

git add <file>
Добавить <file> в индексацию

1.5 Git clone

git clone https://github.com/libgit2/libgit2.git
клонирование репозитория

git clone <url>
клонировать репозиторий

git clone -o <локальное_название_источника>
склонировать репозиторий, и дать источнику имя <локальное_название_источника>

1.6 Git diff

git diff
посмотреть изменение файлов в консоли до индексации

git diff --cached
покажет проиндексированные изменения

git diff --staged
что из проиндексированного войдет в следующий коммит, воспользуйтесь командой

1.7 Git log

git log
показать все последние коммиты

git log -p
показать все последние коммиты с изменениями

git log -p -1
показать 1 последний коммит с изменениями

git log --stat
показать все коммиты с измененными файлами кратко

git log --pretty=oneline
показ комитов с описанием

gitlog --graph
удобное представление логов

git log --relative-date
относительная дата коммита (35 минут назад)

1.8 Git branch

git branch -d <имя_ветки>
Удалить локальную ветку

git branch <имя_локальной_ветки>
создать локальную ветку

git branch -d <имя_ветки>
удалить локальную ветку

git branch -u <удаленная_ветка>
начать отслеживать удаленную ветку с текущей

git branch -vv
показать информацию о ветках и за чем они следят

1.9 Git remote

git remote -v
посмотреть подключенный удаленный источник

git remote add <локальное_название_источника> <url>
добавить новый источник

git remote rename <старое_имя_уд._реп> <новое_имя_уд._реп>
переименовать удаленный источник

git remote rm <имя_уд._реп>
удалить ссылку на удаленный реп

1.10 Git stash

git stash
убрать в стэш изменения

git stash list
показать список изменений в стэшэ

git stash pop
вытащить из стэша (и удалить из стэка стэша) последние изменения

git stash apply
вытащить последние изменения (но не удалять из стэка стэша) спрятанные в стэш

git stash list | pop stash@{0}
вытащить по порядковому номеру изменения из стэша

git stash drop stash@{0}
удалить из стэка стэша изменения у последнего коммита

git stash --keep-index
убрать в стэш только непроиндексированные файлы (красные)

git stash -u
скрыть файлы в стэш, включая те, которые не находятся в индексации

git stash show stash@{0}
показать файлы в последнем стеше

1.11 Git merge\mergetool

git merge <название_ветки>
слить текущую ветку с <название_ветки>

git merge --abort
отменить слияние (например при конфликте)

git merge --no-commit --squash featureB
--squash - сливает изменения в один коммит, не являющийся коммитом слияния
--no-commit - отменяет автоматическую запись коммита

git mergetool
открыть редактор по умолчанию для разрешения конфликта

git mergetool --tool=<название_утилиты>
открыть редактор <редактор> для разрешения конфлита

1.12 Git rm\mv

git rm <файл>
удаляет файл и вносит в индексацию ‘удаление‘ этого файла

git rm --cached <файл>
удаляет из индекса, но физически не удаляет его

git mv <изначальный_файл> <новое_название_файла>
переименование с индексацией

1.13 Git checkout

git checkout <имя_ветки>
переключится на ветку

git checkout -b <имя_новой_ветки>
создать ветку и переключится на нее

git checkout -- <файл>
Вернуть <файл> к дефолту. (Убрать внесенные изменения)

1.14 Git others

git push origin --delete <имя_ветки>
Удалить ветку из удаленного репозитория

git commit -a
пропуск индексирования и сразу коммит

git commit --amend
слияние текущего коммита с предыдущим

git reset HEAD <имя_файла>
убрать из индексирования файл

git fetch <локальное_название_источника>
скачать без слияния

git pull <локальное_название_источника>
скачать со слиянием

git diff master
смотрим изменения

git push origin --delete <имя_ветки>
удалить ветку с удаленного репозитория

git rebase <имя_ветки>
тоже самое что и git merge, но только история коммитов отображается по другому

git request-pull origin/master myfork
запрос на включение коммитов в удаленный реп "origin/master"с ветки "myfork"

git revert -m 1 HEAD
отменить последний коммит с фиксацией в истории

git blame -L 12,22 simplegit.rb
посмотреть кто сделал изменения в файле "simplegit.rb"с 12 строки до 22 строки

1.15 Git ignore

Пример файла gitignore

- комментарий
.a - пропускать файлы, заканчивающиеся на ".a"
!lib.a # - но отслеживать файлы "lib.a несмотря на пропуск файлов на ".a"
/TODO - игнорировать только корневой файл "TODO a не файлы вида "subdir/TODO"
build/ - игнорировать все файлы в папке "build/"
doc/*.txt - игнорировать "doc/notes.txt но не "doc/server/arch.txt"

---

https://github.com/github/gitignore - здесь можно посмотреть различные варианты настройки файла .gitignore