Министерство образования и науки российской федерации  
федеральное государственное автономное образовательное учреждение высшего образования

«Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики»

Факультет информационных технологий и программирования

Кафедра информационных систем

Лабораторная работа № 3

Инструментальные средства разработки ПО

 Изучение системы управления версиями Git

Выполнил студент группы № M3306:

Назаров Д.А.

Проверил:

Липкин Евгений Олегович

Санкт-Петербург

2018

**Основные команды git**

Config

Настройка параметров git: имя пользователя, E-mail, основной текстовый редактор, используемая кодировка, цветовое оформление, сокращённые команды (alias) и т.д. Все настройки git хранятся в файле .gitconfig, который лежит в разных местах в зависимости от области действия настроек и который можно редактировать вручную.

--local – изменений параметров для репозитория в папке вызова команды

--global – изменение параметров для всех репозиториев текущего пользователя

--system – изменение параметров для всех репозиториев на компьютере

Параметры

--edit – открытие конфигурационного файла для чтения и изменения

--list – отображение всех значений конфигурации

--file <name> - использование в качестве конфигурационного файла файл <name>

Init

Создание нового репозитория в текущей папке. Репозиторий может создаваться как в пустой папке, так и в папке с файлами. После выполнения команды, будет создана невидимая папка .git, в которой будут храниться все данные репозитория.

git init [<directory>] - создание репозитория в <directory> или в текущей папке, если <directory> не указано

Параметры

--bare – создание репозитория без рабочего каталога, то есть в папке репозитория хранятся только данные репозитория (изменения файлов) без самих файлов. Используется для создания общего репозитория. Обычно такие репозитории имеют расширение .git.

--template=<directory> - создания репозитория на основе шаблона, лежащего в <directory>. Новый репозиторий создаётся с папкой .git клонированной из <directory>.

--separate-git-dir=<directory> - служебная папка репозитория (.git) будет находиться в <directory>, а не в папке хранения рабочих файлов.

Clone

Копирует существующий репозиторий со всеми файлами внутри него, в том числе со всей его историей коммитов.

git clone <repository> [<directory>]- копирование папки репозитория в текущую папку (репозиторий будет назван так же, как и копируемый)  
[<directory>] – при наличии этого аргумента репозиторий копируется в <directory> (репозиторий будет назван так же, как directory)

Параметры

--bare – к репозитория без рабочего каталога, то есть в папке репозитория хранятся только данные репозитория (изменения файлов) без самих файлов. Используется для создания общего репозитория. Обычно такие репозитории имеют расширение .git.

-depth=<depth> - копируется не весь репозиторий, а файлы и их история с глубиной <depth>

-branch <branch> <repository> - копируется только ветка <branch> из репозитория <repository>

--origin <name> - использует имя <name> для удалённого репозитория

Add

Индексация файлов или добавление в буферную зону.

Учитывание изменений в объектах для их сохранения в следующем коммите. Если файл изменился после выполнения команды git add, то новые изменения не будут записаны в следующий коммит. Для этого команду git add надо вызвать повторно.

Отсюда можно сделать вывод, что git отслеживает не сами файлы, а изменения в них.

git add <file>/<directory> - отслеживание изменений в файле <file> или в папке <directory> (будут добавлены к отслеживанию все объекты папки)

<directory>:  
:/ - изменения во всей рабочей области  
. – изменения в текущей папке

Параметры

-u – обновляется состояние только существующих файлов, то есть в следующем коммите не будут учитываться новые файлы

-p – запуск интерактивного режима, в котором можно выбрать какие строки в файлы нужно проиндексировать

Commit

Сохранение текущего состояния репозитория в виде новой версии файлов. При этом сохраняются не сами файлы, а изменения, произошедшие с ними.

Параметры

-m “<text>” - добавление названия <text> к коммиту. При отсутствии этого ключа откроется текстовый редактор для написания названия (m=message)

-a – автоматическое индексирование изменений в отслеживаемых файлах (новые файлы не будут проиндексированы) и добавление их в коммит. Позволяет обойтись без предварительного вызова git add

--no-edit – не будет меняться комментарий коммита

--amend – изменение последнего коммита

Tag

Создание названия для версий состояний репозитория (коммитов).

git tag – отображает все доступные теги (версии)

Параметры

<name> - присвоение текущему коммиту версии <name>

-d <name> - удаляет тег <name>

-m “<comment>” – добавляет комментарий <comment> к тегу

Log

Просмотр истории изменений репозитория для текущей ветки. Для каждого коммита по умолчанию выводятся: хэш (SHA-1), тег, ветка, автор, дата создания, комментарий.

git log – отображает историю коммитов

Ключи

--all – отображение всех коммитов, даже тех, которые находятся в других ветках

--stat – вывод подробной информации о коммитах: стандартная информация, изменённые файлы, количество изменений в них.

-p – вывод всей информации о каждом коммите: стандартная информация, все изменения в файлах (результат работы git diff)

-g – вывод записей из reflog (изменения положения HEAD)

Reset

Отмена учитывания изменений (индексирования), которые будут внесены в следующий коммит, а также отмена самих коммитов. Может быть использована при случайном вызове команде git add.

Для отмены изменений в самом файле, после выполнения reset, следует выполнить checkout <file>.

git reset HEAD <file> - отмена учитывания изменений в файле <file>

Параметры

<commit> - использовать состояние <commit> для возврата всего репозитория

<commit> <file> - использовать состояние <commit> для файла <file>

--hard <commit> – удаляет все коммиты в текущей ветке, которые были после <commit>. Если эти коммиты имеют названия версий, то они не будут полностью удалены и их можно будет найти с помощью git log –all

Revert

Отмена сделанных коммитов, путём создания нового отменяющего коммита.

git revert HEAD – отмена коммита, на который указывает HEAD

Параметры

<commit> - отменяемый коммит

--no-edit – отмена коммита без редактирования комментария к нему

--no-commit – отмена коммита без создания нового

Pull

Получение коммитов из внешнего репозитория и объединение их в локальный репозиторий. Является комбинацией git fetch и git merge.

Ключи

--rebase – вместо git merge применяется git rebase

Параметры

<repository> - получение изменений из <repository> и их объединение

<repository> <branch> - получение изменений из ветки <branch> внешнего репозитория <repository> и их объединение

--depth=<depth> - число <depth> последних коммитов для получения

Push

Отправка коммитов во внешний репозиторий.

git push – отправка коммитов в origin репозиторий

Параметры

<repository> <branch> - отправка коммитов в <repository> в ветку <branch>

--delete <repository> <branch> - удаление ветки из внешнего репозитория

Show

Показывает изменения, произошедшие в указанном коммите.

git show – отображает изменения в HEAD

Параметры

<hash> - отображает изменения в коммите <hash>

Ls-tree

Показывает содержимое репозитория на момент выбранного коммита.

Параметры

<hash> - хэш код коммита для просмотра его дерева

Gc

Сборщик мусора, который удаляет недостижимые коммиты, хранящиеся дольше нескольких месяцев, сжимает объекты, объединяет ссылки в один файл.

git gc – запуск сборщика мусора

Fsck

Проверяет базу репозитория на целостность. С помощью этого инструмента можно восстановить удалённые коммиты.

git fsck – проверка на целостность и вывод недостижимых коммитов

Ключи

--full – показывает все объекты

--unreachable – показывает существующие, но недостижимые объекты

Prune

Удаляет все недостижимые объекты из базы репозитория.

Ключи

--dry-run – показывает действия, которые будут совершены, но не выполняет их

--verbose (-v) – сообщает обо всех удалённых объектах

Archive

Создаёт архив, содержащий состояние репозитория на момент указанного коммита.

git archive <hash> - выведет на консоль содержимое репозитория

Ключи

--list – список доступных форматов архива

Параметры

<hash> - хэш коммита для архивирования

--format=<format> - формат архива

--output=<file> - запись архива в файл <file>, а не вывод содержимого на консоль

Merge

Производит слияние веток.

Ключи

--no-commit – не создаётся коммит при слиянии, чтобы можно было просмотреть, что будет объединено

--abort – отмена слияния во время решения конфликта

Параметры

<branchName> - ветка, которая будет объединена в текущую ветку

-m “<comment>” – добавление комментария к коммиту

Rebase

Производит объединение веток.

Параметры

<branchName> - ветка, которую надо присоединить к текущей

Remote

Взаимодействие с удалёнными (внешними) репозиториями.

git remote – отображает имена всех доступные внешних репозиториев

Ключи

-v – при отображении репозиториев добавляет к именам внешних репозиториев их адреса

Параметры

show <name> - отображает информацию о репозитории <name> (адреса для отправки и получения коммитов, ветки и т.д.)

add <name> <address> - добавляет внешний репозиторий по адресу <address> и присваивает ему имя <name>

rename <old> <new> - переименовывает внешний репозиторий <old> в <new>

remove <name> - удаляет репозиторий <name>

seturl <name> <newURL> - устанавливает у репозитория <name> URL <newURL>

Fetch

Получение коммитов из внешнего репозитория, но без объединения с существующей веткой.

При вызове git log без ключа --all полученные коммиты отображаться не будут

git fetch – получить изменения из origin

Ключи

--all – получение изменений из всех внешних репозиториев

--dry-run – будут показаны действия после выполнения команды, но эти действия не будут применены

Параметры

<repository> - получение изменений из <repository>

<repository> <branch> - получение изменений из ветки <branch> внешнего репозитория <repository>

--depth=<depth> - число <depth> последних коммитов для получения

Status

Выводит информацию о текущем состоянии репозитория: в какой ветке сейчас находится пользователь, какие файлы отслеживаются, какие файлы изменились, но не были сохранены в виде коммита. Файлы, внесённые в .gitignore не будут проверяться.

git status

Параметры

--short – вывод краткой информации  
--long – вывод подробной информации (по умолчанию)

Checkout

Переход к определённым версиям, веткам, коммитам (происходит перемещение HEAD). С помощью этой команды можно вернуть состояние репозитория, которое было на момент определённого коммита, то есть откатить содержимое всех файлов на то состояние.

При переключении на ветку происходит перемещение к последнему сделанному коммиту в ней.

Параметры

<hash> - возврат к коммиту с хэшем <hash> (для хэша подходит краткая запись – первые 7 символов)

<branch> - переход к последнему коммиту в ветке <branch>

<version> - переход к версии <version>

<file> - меняет текущее содержимое файла на то, что находится в текущем коммите, то есть отменяет не проиндексированные командой git add изменения

<object>~<n> - переход на <n> коммитов назад до <object>

-b <branch> - создаёт ветку <branch> и сразу переходит к ней

-f – принудительный переход, даже если изменения не были записаны в репозиторий

Branch

Используется для взаимодействия с ветками.

git branch – вывод имён всех веток репозитория

Ключи

-a – отображение всех веток, в том числе на удалённых репозиториях

Параметры

<name> - создание новой ветки с именем <name>  
<name> <commit> - создание новой ветке на коммите с хэшем <commit>

-d <name> - безопасное удаление ветки с именем <name> с проверкой на объединение изменений с другой веткой  
-D <name> - принудительное удаление ветки <name> без проверок  
-d -r – добавление этого ключа к ключу удаления удаляет ветку из локального репозитория

-m <name> – смена имени текущей ветки на <name>

--track <name> <repository>/<branch> - создание ветки <name>, отслеживающей изменения ветки <branch> в репозитории <repository>, которая автоматически будет делать git pull для неё

Diff

Отображает изменений произошедшие внутри файлов.

git diff – показывает изменения в файлах, которые ещё не были проиндексированы, по сравнению с последним коммитом

Ключи

--staged – показывает изменения в проиндексированных файлах по сравнению с последним коммитом

Mv

Перемещение файлов или их переименование. Может быть заменено ручным перемещением файла и индексированием изменений.

Параметры

<source> <destination> - перемещение <source> в <destination>  
Если <destination> - это не папка, то произойдёт переименование <source>

--dry-run – ничего не делает, но показывает, что будет сделано

Rm

Удаление файла из рабочей области и добавление этих изменений к будущему коммиту (git add). Может быть заменено стандартным удалением и выполнением индексированиям.

Параметры

<file> - удаление файла <file>

--cached - файл удалиться только из репозитория, но не из папки

Cat-file

Вывод содержимого и информации файлов репозитория (папки .git).

Параметры

<object> - имя объекта для просмотра информации

<type> <object> - просмотр содержимого объекта, если его тип <type>

-t – показывает тип <object>

-s – показывает размер <object>

-p – вывод содержимого <object>

Grep

Аналог утилиты grep. Позволяет искать строки в истории коммитов и рабочем каталоге, используя регулярные выражения.

Stash

Позволяет отложить несохранённые в репозитории изменения во временное хранилище. Используется когда нужно совершить действия, которое требует отсутствия несохранённых изменений (смена ветки или получение изменений с сервера, например), а локальные файлы уже были изменены.

В Stash хранятся несохранённые изменения привязанные к коммиту, на котором они были сделаны.

Параметры

apply – применить изменения к текущей версии репозитория

list – отображение всех спрятанных изменений

show – вывести содержание последних изменений

drop – удалить содержание последних изменения

clear – очистить все спрятанные изменений

Instaweb

Визуализирует репозиторий в браузере.

Ключи

--stop – остановка сервера

Параметры

--httpd=<server> - используемый веб-сервер для отображения страницы

<server>:  
webrick  
lighttpd