|  |  |
| --- | --- |
| 1 | **mkdir projects**  Создать новую папку в корневом каталоге (терминал открывается в нём по умолчанию). |
| 2 | **cd projects**  Перейти внутрь *projects*. |
| 3 | **mkdir Skillfactory**  Создать новую папку *Skillfactory*. |
| 4 | **cd Skillfactory**  Перейти внутрь *Skillfactory*. |
| 5 | **git init**  Инициализировать папку *Skillfactory* как *Git*-репозиторий. |

Чтобы проверить статус, в котором находится ваш репозиторий, наберите в терминале:

git status

* **On branch master**. Сообщение означает, что в настоящий момент активна ветка *master*. Как правило, это основная ветка в репозитории.
* **Your branch is up to date with origin/master**. Сообщение означает, что файлы в ветке *master* синхронизированы с веткой *master* на сервере. *Origin* — это и есть удалённый сервер.
* **Working tree clean** дословно означает, что рабочий каталог не содержит изменённых файлов. Система контроля версий *Git* использует так называемую архитектуру трёх деревьев. *Working tree* связана с рабочей директорией *Git*, в которой содержатся рабочие версии файлов.
* **Nothing to commit** значит, что все изменения были зафиксированы и нет новых изменений в файлах для фиксации изменений.
* Другими словами, **коммит**— это состояние репозитория в определённый момент времени. Именно к коммиту мы можем возвращаться, чтобы посмотреть состояние объектов в какое-то время.

## ****Создание коммита****

* Давайте создадим первый коммит.
* Для этого нам надо сначала добавить все те файлы, изменения в которых или создание/удаление которых мы хотим зафиксировать.
* Чтобы добавить все файлы, которые находятся в директории, выполните следующую команду:
* **git add -A**
* Но пока мы только собрали все те файлы, изменения в которых нам важны. Теперь их надо обвязать общей верёвочкой и единым ярлыком, то есть собрать их в коммит. Для этого необходимо выполнить следующую команду:
* git commit -m "Init Skillfactory"
* git commit — команда, которая создаёт коммит с рядом параметров.

|  |  |
| --- | --- |
| **git commit -a** | При выполнении команды с флагом -a будет сделан снимок всех изменений в рабочем каталоге. Это включает в себя только изменения отслеживаемых файлов (то есть тех, что были добавлены с помощью git add ранее). |
| **git commit -am "commit message"** | Сочетание флагов -a и -m создаёт коммит всех изменений с сообщением для коммита. |
| **git commit --amend** | Выбранные изменения будут добавлены в последний коммит. Команда также вызовет открытие системного текстового редактора для внесения изменений в сообщение коммита. Эту команду часто используют, если была допущена ошибка. |
|  |  |

* Флаг -m означает, что следом идёт сообщение (*message*): ярлык для наших изменений.

ажно понимать, что **ветка** — это всего лишь указатель на коммит, не меняющий репозиторий. Когда вы делаете коммиты, не создавая и не меняя ветки, то работаете в ветке master. Она указывает на последний сделанный коммит и при каждом новом коммите сдвигается вперёд автоматически. Выглядит это примерно так: авайте создадим новую ветку. Для этого выполним в*Git*-репозитории следующую команду:

git branch awesome-branch

Да, действительно есть. Git хранит специальный указатель, который называется **HEAD** (верхушка). HEAD указывает на локальную ветку, на которой вы находитесь.

Чтобы внести изменения из одной ветки в другую, нужно выполнить команду **git merge**. По-русски эти команды часто называют «вмержить», «смержить» или «сделать мерж», хотя, конечно, это всё производные от английских терминов. Правильно на русский язык *merge* можно перевести как «слияние», поэтому говорят также «слияние веток».

**Слияние**— это способ соединить обратно раздвоенную историю. Команда git merge позволяет взять независимые части разработки и интегрировать их в одну ветку.

Значит, её нужно удалять, потому что незачем хранить лишнее. Чтобы удалить ветку, достаточно выполнить команду:

git branch -d black-friday

Чтобы перейти на существующую ветку, необходимо выполнить следующую команду:

git checkout awesome-branch