Шпаргалка по командам Git

1. **Как проверить конфигурацию Git**

Команда ниже возвращает список информации о вашей конфигурации Git, включая имя пользователя и адрес электронной почты:

```
git config -l
```

2. **Как настроить имя и электронную почту пользователя Git**

С помощью приведённых ниже команд можно настроить имя пользователя и электронную почту, которые будут использоваться в коммитах:

```
git config --global user.name "John Doe"

git config --global user.email johndoe@example.com
```

3. **Как инициализировать репозиторий Git**

Самый первый шаг в работе с проектом — инициализировать новый репозиторий Git локально в корне проекта. Это можно сделать с помощью команды:

```
git init
```

4. **Как клонировать удалённый репозиторий с помощью Git**

Если удалённый репозиторий уже создан (например, в GitLab), то его можно клонировать к себе на компьютер следующей командой:

```
git clone repository_url
```

5. **Как добавить файлы в промежуточную область Git**

Команда ниже добавит файл в промежуточную область. Просто замените filename на имя файла, который хотите добавить в промежуточную область:

```
git add filename
```

Можно также добавить все файлы проекта в промежуточную область. В таком случае нужно использовать точку, тогда будут добавлены все файлы и подпапки текущей директории:

```
git add .
```

А с помощью такой команды можно добавить все файлы, начинающиеся с fil:

```
git add fil*
```

6. **Как проверить статус репозитория в Git**

Эта команда покажет состояние текущего репозитория, включая подготовленные, неподготовленные и неотслеживаемые файлы:

```
git status
```

7. **Как зафиксировать изменения в Git**

Зафиксировать изменения можно командой ниже. Также можно добавить сообщение коммита:

```
git commit -m "commit message"
```

8. Как посмотреть историю коммитов в Git

Эта команда показывает историю коммитов для текущего репозитория:

```
git log
```

9. Как создать новую ветку в Git

По умолчанию у вас есть одна ветка — главная. С помощью этой команды можно создать новую ветку:

```
git branch branch_name
```

Git не переключится на неё автоматически. Нужно сделать это вручную с помощью следующей команды:

```
git checkout branch_name
```

Добавив к команде checkout флаг -b, можно с помощью одной строки создать ветку и сразу переключиться на неё:

```
git checkout -b branch_name
```

10. Как просмотреть список веток в Git

Можно просмотреть все созданные ветки с помощью следующей команды:

```
git branch
```

Она покажет список всех веток и пометит текущую ветку звёздочкой.

11. Как отправить изменения в удалённый репозиторий с помощью Git

Когда вся ваша работа будет готова для сохранения в удалённом репозитории, вы можете отправить все изменения, используя команду ниже:

```
git push
```

12. Как отправить ветку в удалённый репозиторий с помощью Git:

Если требуется отправить ветку в удалённый репозиторий, можно использовать команду:

```
git push -u origin branch_name
```

С ключом -u в удалённом репозитории создаётся (если ещё не существует) ветка, соответствующая локальной.

Origin – это сокращённое имя удалённого репозитория, из которого изначально был клонирован проект. Оно используется вместо URL-адреса исходного репозитория, что значительно упрощает обращение к нему.

13. Как удалить ветку в Git

Чтобы удалить ветку после того, как работа с ней будет закончена и она будет объединена с главной, можно использовать команду:

```
git branch -d branch_name
```

14. Как удалить ветку в удалённом репозитории в Git:

Если ветка в удалённом репозитории больше не нужна, можно удалить её с помощью следующей команды:

```
git push --delete origin branch_name
```

15. Как получить изменения из удалённого репозитория в Git

Если над вашим репозиторием работают другие члены команды, вы можете получить последние изменения, внесённые в удалённый репозиторий, с помощью команды:

```
git pull
```