

Введение в DevOps: Git



Шило
Петр



Шило Петр

Devops-инженер

Arifonica



[Шило Петр](#)

Модуль «Введение в DevOps»

Цели модуля:

- узнать, что такое DevOps;
- познакомиться с основными задачами и инструментами;
- получить практические навыки работы с системой контроля версий Git.



Структура модуля

1. **Git**
2. Что такое DevOps? CI/CD
3. GitLab

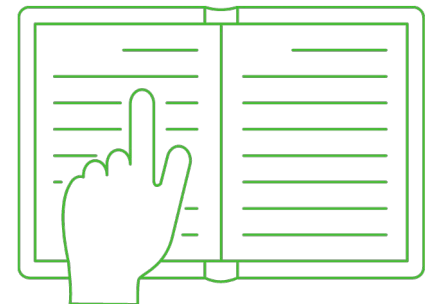


Предисловие

На этом занятии мы поговорим о:

- git,
- ветвлении проекта,
- сервисе github.

По итогу занятия вы узнаете, как организуется командная работа с проектом при помощи git.



План занятия

1. [Основы git](#)
2. [Работа с git](#)
3. [Удаленные репозитории](#)
4. [Файл .gitignore](#)
5. [Github](#)
6. [Итоги](#)
7. [Домашнее задание](#)



Основы git

Основы git

Системы контроля версий (СКВ, VCS, Version Control Systems) позволяют разработчикам сохранять все изменения, внесенные в код. СКВ также позволяют нескольким разработчикам работать над одним проектом и сохранять внесенные изменения независимо друг от друга.

Git — это инструмент, позволяющий реализовать распределенную систему контроля версий.



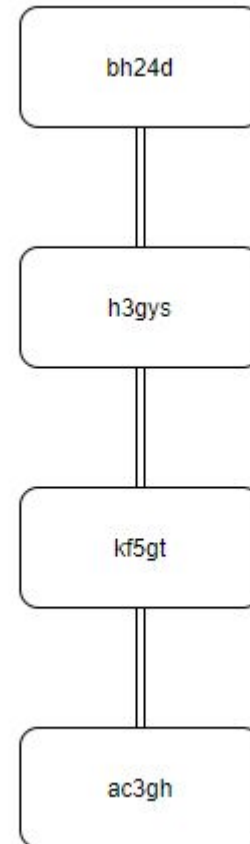
Основы git

С помощью git можно откатить свой проект до более старой версии, сравнивать, анализировать или сливать свои изменения в репозиторий.

Репозиторием называют хранилище кода и историю его изменений. Git работает локально и все ваши репозитории хранятся в определенных папках на жестком диске.

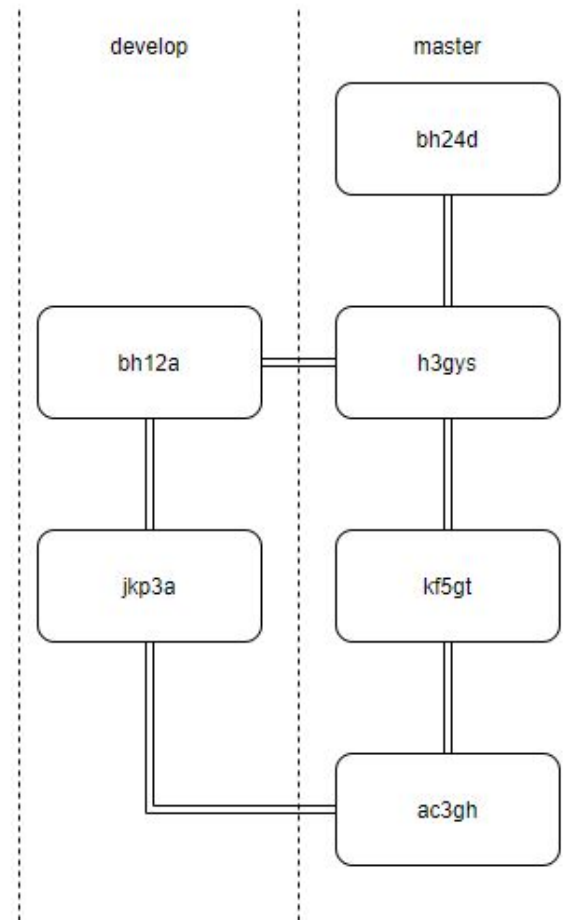
Коммиты

Каждая точка сохранения называется коммит (**commit**). У каждого commit есть **hash** (уникальный id) и комментарий. Из таких commit-ов собирается ветка. **Ветка** — это история изменений. У каждой ветки есть свое название.



Ветвления

Ветки имеют свою собственную историю и изолированные друг от друга изменения до тех пор, пока вы не решаете слить изменения вместе.



Установка и настройка репозитория

Установка git из репозитория:

```
sudo apt-get install git
```

Настройка git для корректного отображения автора коммитов:

```
git config --global user.name "My Name"  
git config --global user.email myEmail@example.com
```

Инициализация репозитория:

```
git init
```

Просмотр текущего статуса:

```
git status
```



Работа с git

Работа с git

Чтобы добавить файлы для отслеживания используется команда:

```
git add имя_файла
```

После этого изменения файла начнут отслеживаться и его можно закоммитить:

```
git commit -m "some useful comment here"
```

Перед каждым коммитом необходимо проиндексировать файлы, которые будут закоммичены.

Это можно сделать с помощью:

```
git add
```

либо добавив ключ:

```
git commit -a
```

Последний коммит в ветке обозначается как HEAD.

Это сделано для упрощенного доступа к нему.

Работа с git: Отмена изменений

До выполнения индексации:

```
git checkout имя файла
```

После индексации:

```
git reset HEAD имя файла
```

Откат изменений коммита осуществляется с помощью revert:

```
git revert HEAD --no-edit
```

Вместо HEAD можно указать любой id коммита из истории.

Просмотр истории:

```
git log
```

Работа с ветками

Создание веток:

```
git checkout -b имя_ветки
```

Слияние веток:

```
git checkout master
```

```
git merge develop
```




Удаленные репозитории

Клонирование репозитория

Клонировать репозиторий с сервера можно с помощью команды:

```
git clone https://github.com/netology-code/sysadm-homeworks.git
```

Репозиторий будет помещен в директорию ./имя_репозитория

Просмотр удаленных репозиториев:

```
git remote
```

В ответ мы увидим имена всех существующих удаленных репозиториев (по умолчанию обычно origin):

```
git remote show origin
```

Добавление удаленного репозитория

Добавление удаленного репозитория:

```
git remote add origin https://github.com/netology-code/sysadm-homeworks.git
```

Отправка изменения в удаленный репозиторий:

```
git push origin master
```

Для того чтобы скачать себе последние изменения из репозитория:

```
git pull
```



Файл .gitignore

.gitignore

В любой директории внутри репозитория можно создать файл .gitignore

В него можно добавить правила игнорирования некоторых файлов. Например, локальный кеш, тестовые конфигурации и прочее.

```
test.conf  
directory/example.conf  
cache/*  
*.jpg
```



Github

— сервис онлайн-хостинга репозиторий, обладающий всеми функциями всего, что поддерживает Git и даже больше. Также GitHub может похвастаться контролем доступа, багтрекингом, управлением задачами и вики для каждого проекта.

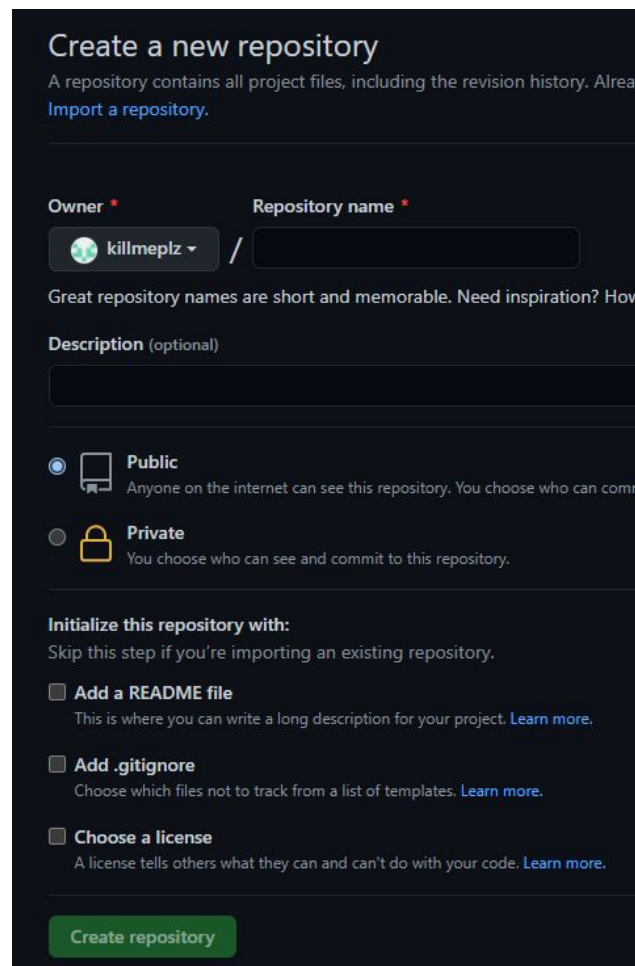
Кроме GitHub есть другие сервисы, которые используют Git, — например, Bitbucket и GitLab. Вы можете разместить Git-репозиторий на любом из них.

Создание GitHub репозитория

Создание доступно по [адресу](#)

Репозиторий может быть как публичным, так и приватным.

Также при создании репозитория можно сразу создать стандартные файлы.



The screenshot shows the GitHub 'Create a new repository' page. At the top, it says 'Create a new repository' and 'A repository contains all project files, including the revision history. Already have a repository? [Import a repository.](#)'. Below this, there are two main sections: 'Owner' and 'Repository name'. The 'Owner' dropdown is set to 'killmeplz'. The 'Repository name' field is empty. A note below says 'Great repository names are short and memorable. Need inspiration? How to choose a name'. The 'Description (optional)' field is also empty. Below the description field, there are two radio button options: 'Public' (selected) and 'Private'. The 'Public' option has a description: 'Anyone on the internet can see this repository. You choose who can commit to this repository.' The 'Private' option has a description: 'You choose who can see and commit to this repository.' Below these options, there is a section 'Initialize this repository with:' with the instruction 'Skip this step if you're importing an existing repository.' There are three checkboxes: 'Add a README file' (unchecked), 'Add .gitignore' (unchecked), and 'Choose a license' (unchecked). Each checkbox has a description and a 'Learn more' link. At the bottom, there is a green 'Create repository' button.

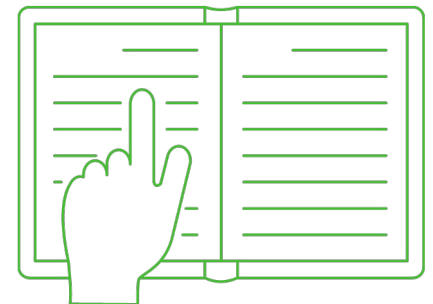


Итоги

Итоги

Сегодня мы рассмотрели устройство git и теперь:

- можем вести разработку в репозитории;
- умеем пользоваться github.





Домашнее задание

Домашнее задание

Давайте посмотрим ваше [домашнее задание](#).

- Вопросы по домашней работе задавайте **в чате** мессенджера Slack.
- Задачи можно сдавать **по частям**.
- Зачёт по домашней работе проставляется после того, как **приняты все задачи**.

**Задавайте вопросы и
пишите отзыв о лекции!**

Шило Петр

 [Шило Петр](#)