

Введение в DevOps: Что такое DevOps? CI/CD




Шило
Петр



Шило Петр

Devops-инженер

Arifonica

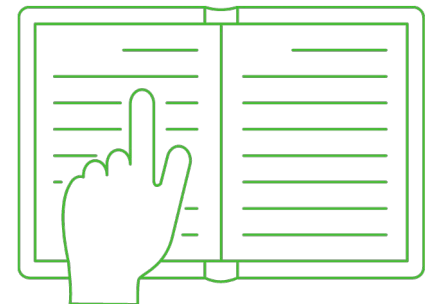
 [Шило Петр](#)

Предисловие

На этом занятии мы поговорим о:

- методологии DevOps,
- основах CI/CD,
- Jenkins.

По итогу занятия вы узнаете, как организуется работа DevOps-инженеров и доступные им инструменты.



План занятия

1. [Что такое DevOps?](#)
2. [Инструменты DevOps](#)
3. [Jenkins](#)
4. [Nexus](#)
5. [Настройка pipeline](#)
6. [Итоги](#)
7. [Домашнее задание](#)



Что такое DevOps?



Что такое DevOps?

DevOps — это методология разработки ПО, задача которой наладить взаимодействие программистов и сисадминов в компании. Если ИТ-специалисты из разных отделов недопонимают суть задач друг друга, выпуск новых приложений и обновлений для них затягивается.



CI/CD

Важными процессами для достижения результатов DevOps являются процессы CI/CD.

CI (Continuous Integration) — непрерывная интеграция, в задачи этого процесса входит иметь максимально рабочую версию каждого нового релиза. В первую очередь это сборка и тестирование проекта с каждым внесением изменений.

CD (Continuous Deployment) — непрерывная поставка. Суть этого процесса заключается в постоянной выкладке ПО (желательно уже протестированного) на сервера.



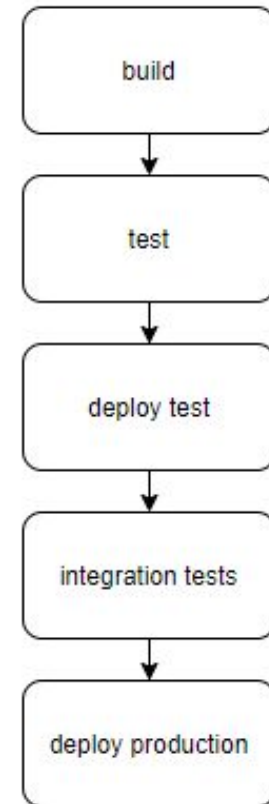
CI/CD

Процессы CI/CD разрабатываются вместе и в современном мире не принято их делить.

Пример: для тестирования проекта вместе с другими его компонентами (база данных, фронтенд и т.п.) необходимо сделать его деплой, а это уже часть процесса CD. Таким образом, для выполнения CI требуется запуск процесса CD, и наоборот, так как перед деплоем проект надо собрать и провести статические тесты.

Пример CI/CD процесса

1. Сборка проекта, например docker build;
2. Запуск статических тестов. В зависимости от языка могут вызываться по разному и могут запускаться до сборки;
3. Деплой проекта на тестовое окружение;
4. Запуск тестов для всего тестового окружения (интеграционные тесты);
5. Деплой проекта на продакшн окружение.





Инструменты DevOps

Основной функционал

Для запуска всех вышеописанных процессов обычно используется интеграция с git. При появления коммита в git сервере запускается процесс CI/CD, который в зависимости от разных параметров коммита (ветка, автор, тег и.т.п) выполняет все действия, необходимые для сборки процесса.

Примеры систем, используемых для запуска CI/CD:

- Jenkins
- Gitlab CI
- Teamcity
- ...



Jenkins

Jenkins

Jenkins — программная система с открытым исходным кодом на Java, предназначенная для обеспечения процесса непрерывной интеграции программного обеспечения.



Jenkins можно установить на Windows, macOS, Debian, Ubuntu, CentOS и другие операционные системы. Также Jenkins можно установить через системные пакеты, Docker или запустить автономно на любом компьютере с настроенной Java Runtime Environment (JRE).

[Подробная документация](#)

Установка Jenkins

Первым делом устанавливаем java

```
sudo apt-get install default-jre
```

Затем устанавливаем Jenkins из пакета для debian/ubuntu

```
sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ >  
/etc/apt/sources.list.d/jenkins.list'  
sudo apt-get update  
sudo apt-get install jenkins
```

Первичная настройка

1. В строке браузера введите IP-адрес сервера и порт 8080 в формате 123.123.123.123:8080.
2. На стартовой странице Jenkins указан путь, по которому хранится пароль для входа, обычно:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

3. Введите пароль в браузере для разблокировки Jenkins.
4. После предложения установить плагины и создать первого admin пользователя Jenkins доступен для работы.

Важно: для доступа к docker из пайплайнов необходимо добавить пользователя jenkins в группу docker и перезапустить jenkins.



Nexus

Nexus

Sonatype Nexus – интегрированная платформа, с помощью которой разработчики могут хранить и управлять зависимостями Java (Maven), образами Docker, Python, Ruby, NPM, Bower, RPM-пакетами, gitlfs, Apt, Go, Nuget, а также распространять свое программное обеспечение.

[Документация](#)

Установка с помощью docker:

```
docker run -d -p 8081:8081 --name nexus -e INSTALL4J_ADD_VM_PARAMS="-Xms273m  
-Xmx273m -XX:MaxDirectMemorySize=273m" sonatype/nexus3
```

Обратите внимание на параметры Xms и Xmx. Это объем выделяемой памяти, по умолчанию выделяется 2703Мб.

Nexus: создание helm репозитория

1. Переходим на страницу настройки репозитория:
<http://123.123.123.123:8081/#admin/repository/repositories>
2. Создаем новый репозиторий типа helm-hosted
3. Выбираем дополнительные параметры, сохраняем.



Настройка pipeline

Настройка простого pipeline

1. Создаем проект <http://123.123.123.123:8080/view/all/newJob>;
2. Выбираем freestyle project и пишем имя проекта;
3. Подключаем scm и параметры авторизации, если необходимо;
4. Переходим в раздел Build, выбираем новый этап сборки типа "Execute shell";
5. Пишем любой скрипт, например `docker build`;
6. Сохраняем проект и запускаем сборку.

Настройка declarative pipeline

1. Создаем проект <http://123.123.123.123:8080/view/all/newJob>;
2. Выбираем pipeline и пишем имя проекта;
3. Добавляем скрипт в поле Pipeline.

```
pipeline {
  agent any
  stages {
    stage('Git') {
      steps {git 'https://github.com/killmeplz/k8s-job-sidekiller.git'}
    }
    stage('Build') {
      steps {
        sh 'docker build .'
        sh 'helm package .helm'
        sh 'curl -u admin:123
http://10.168.10.220:8081/repository/helm-test/ --upload-file
k8s-job-sidekiller-0.1.0.tgz -v'
      }
    }
  }
}
```

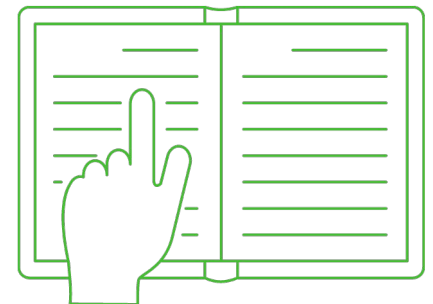


Итоги

Итоги

Сегодня мы рассмотрели основы DevOps решений и теперь:

- можем использовать Jenkins и Nexus;
- умеем составлять pipeline;
- знаем, из чего состоит работа DevOps инженера в команде.





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](#)