Jenkins, демонстрация

Юрий Литвинов
yurii.litvinov@gmail.com

16.10.2018

1. Требования

Jenkins можно скачать с `https://jenkins.io/`, в разных вариантах: либо как устанавливаемую программу, либо как Java .war-файл. Если выбрано последнее, то чтобы его запустить, надо выполнить `java -jar jenkins.war --httpPort=8080` и, когда он проинициализируется, зайти на `http://localhost:8080`. Там он предложит поставить плагины, можно выбрать “Поставить самые популярные”, а можно и повыбирать плагины руками. Из их названий более-менее понятно, для чего они. Ставится оно под виндой в C:/Users/<user>/.jenkins.

Если Jenkins поставлен инсталлятором, то он сам запустится, откроет `http://localhost:8080` и предложит поставить плагины. В этом случае всё под виндой оказывается обычно в C:/Program Files (x86)/Jenkins.

2. Jenkins

Jenkins, как и AppVeyor, управляется с помощью конфигурационного файла, который должен лежать в корне репозитория и называться Jenkinsfile. Для того, чтобы что-то сделать, нам потребуется гит-репозиторий, потому как Jenkins первое, что делает — получает из указанного репозитория исходники. Достаточно любого репозитория на гитхабе (локальный тоже пойдёт, но только если гит запущен как сервер).

Создадим Jenkinsfile, примерно такого содержания:

```
pipeline {
    agent any
    stages {
        stage('build') {
            steps {
                bat 'mvn --version'
            }
        }
    }
}
```

Да, Jenkins (а точнее, его плагин Pipeline, который, собственно, и будет управлять сборкой) использует синтаксис Groovy, знакомый по Gradle. Попрбуем создать билд на основе этого файла. Для этого надо запустить Jenkins (командой `java -jar jenkins.war --httpPort=8080`), пойти на `localhost:8080` и авторизоваться тем паролем, который вы указали при настройке Jenkins-а.

Дальше жмём на New Item, вводим название пайплайна (любое), выбираем Multibranch Pipeline (чтобы Jenkins следил за ветками и пуллреквестами в репозиторий и собирал их), жмём Ок. Заполняем сведения про билд, например, ссылку на репозиторий, жмём save — и он уже пытается получить репозиторий и собрать.

Так, как было выше, нам требуется установленный Maven в путях. Более модно не настраивать окружение сборки всякий раз на каждой машине, на которой мы поднимаем Jenkins, а использовать Docker. Например, тот же самый билд, но качающий образ с maven-ом прямо в процессе сборки:

```
pipeline {
    agent { docker 'maven:3.5.2-jdk-8-slim' }
    stages {
        stage('build') {
            steps {
                bat 'mvn --version'
            }
        }
    }
}
```

Под виндой, правда, прямо из коробки не заработает, потому что оно хочет sh и nohup в путях, но оно есть в Cygwin и даже в Git For Windows, так что по идее можно просто подредактировать PATH, указав на папки с нужными бинарниками. Но с большой вероятностью не заработает всё равно (потому как большинство плагинов для Jenkins ожидают линуксового окружения), поэтому идеологически правильнее сам Jenkins запускать в Docker-контейнере, командой (для винды)

```
docker run ^
  --rm ^
  -u root ^
  -p 8080:8080 ^
  -v jenkins-data:/var/jenkins_home ^
  -v /var/run/docker.sock:/var/run/docker.sock ^
  -v "%HOMEPATH%":/home ^
  jenkinsci/blueocean
```

После этого Jenkins придётся активировать ещё раз (обратите внимание, пароль для активации пишется в консоли, если его потерять, придётся освоить исполнение команд в контейнере через docker exec). И придётся снова скачать и обновить плагины. Зато теперь он будет работать даже под виндой, думая, что он запущен под линуксом, всякие проблемы с sh и nohup будет решать уже Docker.

Кроме того, докер-образ поставляется с плагином Blue Ocean, это новый суперудобный фронтэнд для Jenkins, который вообще всё сделает за вас. Доступен, например, по ссылке `http://localhost:8080/blue`, там всё интуитивно понятно, очень рекомендую попробовать настроить билд с ним.