Fabric8 是一个开源集成开发平台，为基于 Kubernetes 和 Jenkins 的微服务提供持续发布。其中提供的 Java Tools 帮助 Java 应用在 OpenShift 上快速构建和发布。

- 提供与 OpenShift 配合的 Maven 插件。
- 使用 Arquillian 在 JUnit 内轻松集成和系统测试 OpenShift 资源。
- Java 库以及对与 OpenShift 一起使用的 CDI 扩展的支持。

Fabric8 Maven 插件就是 Java Tools 提供的。接下来，我们介绍如何通过 Fabric8 Maven 插件在 OpenShift 快速部署应用。

通常，我们要将应用部署到 OpenShift 集群包含如下的步骤：

- 应用容器化：首先将应用程序包装在容器镜像中，该容器镜像已正确定制以运行应用程序。
- 为容器创建 DeploymentConfig 来部署容器镜像。
- 要将应用端点公开给集群中运行的其他 Pod，创建一个 Service 对象。
- 创建一个 Route 对象，以使应用可以从集群外部访问。

Fabric8 Maven 插件用于在 OpenShift 上构建和部署 Java 应用程序，简化了容器镜像构建和发布过程，默认就是使用 OpenShift S2I 完成构建。该插件还生成 OpenShift 资源对象文件，可用于创建 OpenShift 部署微服务所需的资源对象。

Fabric8 Maven 插件主要完成以下三个工作：

- 构建应用镜像。
- 创建 OpenShift 资源对象。
- 在 OpenShift 上部署应用程序。

在实际项目中，要使用 Fabric8 Maven 插件，有以下两种方法：

- 在源码的项目对象模型（POM）文件的插件部分中启用 Fabric8 插件。

  例如要在项目中启用插件的 3.5.38 版本，请将以下内容添加到 pom.xml 文件中：

```
<plugin>
  <groupId>io.fabric8</groupId>
  <artifactId>fabric8-maven-plugin</artifactId>
  <version>3.5.38</version>
</plugin>
```

- 通过使用 mvn 命令运行插件的指令来启用插件。

从 pom.xml 文件所在的项目文件夹的根目录运行该命令。指令会自动将必需的 XML 配置添加到 pom.xml 文件的 plugins 部分。

mvn io.fabric8:fabric8-maven-plugin:3.5.38:setup

我们推荐优先使用第一种方法。

Fabric8 Maven 插件提供了许多在 OpenShift 集群上构建和部署应用程序的指令。运行这些命令的前提是我们已经登录 OpenShift 集群，并创建的对应的项目。

- fabric8:build：构建应用程序的容器镜像。插件会自动检测目标平台是否为 OpenShift，并使用 S2I 方法构建镜像。对于 OpenShift，插件使用二进制源构建方法构建容器镜像。在此方法中，应用程序二进制文件在 OpenShift 集群外部构建，然后通过 Binary to Image 的方式，将应用部署到容器镜像中。
- fabric8:resource：生成 OpenShift 资源描述符，生成的格式是 yaml 文件。
- fabric8:apply：在 OpenShift 中应用由 fabric8:resource 生成的资源配置文件。我们也可以运行 fabric8:resource-apply，这样就是资源的生成和应用在一个命令中完成。
- fabric8:deploy。它相当于执行了 fabric8:resource,、fabric8:build,、fabric8:apply 三条指令。这会导致重建应用程序、创建新的容器镜像，并使用单个命令将资源应用于 OpenShift 集群。

接下来，我们通过一个案例来说明 fabric8 maven 插件的使用方法。

我们有一套微服务 hello-microservices，其中的一个微服务名称为 hola。我们先看整个微服务的 pom.xml 文件定义（~/hello-microservices/pom.xml）。父 POM 文件由此不同微服务使用的公共属性组成，如下图 3-14 所示：

```
<properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <version.wildfly.swarm>7.1.0.redhat-77</version.wildfly.swarm>
        <version.compiler.plugin>3.1</version.compiler.plugin>
        <version.surefire.plugin>2.16</version.surefire.plugin>
        <version.war.plugin>2.5</version.war.plugin>
        <version.fabric8.plugin>3.5.38</version.fabric8.plugin>
        <fabric8.generator.fromMode>istag</fabric8.generator.fromMode>
        <fabric8.generator.from>redhat-openjdk18-openshift</fabric8.generator.from>
        <maven.compiler.target>1.8</maven.compiler.target>
        <maven.compiler.source>1.8</maven.compiler.source>
</properties>
```

图 3-14 pom 文件中的 fabric8 插件

Hola 项目 POM 位于 Hola Maven 项目的根文件夹中。打开文件~/ hello-

microservices/hola/ pom.xml 进行查看， Hola 微服务的打包格式声明为 War，微服务从父 POM 文件继承所有 Maven 配置，如下图 3-15 所示：

```
<artifactId>hola</artifactId>
<packaging>war</packaging>
```

图 3-15 pom 中打包格式声明

<plugin>定义中的<configuration>部分配置插件。 Fabric8 Maven 插件配置为使用 wildfly-swarm 生成器，如下图 3-16 所示：

```
<build>
    <finalName>${project.artifactId}</finalName>
    <plugins>
        <plugin>
            <groupId>io.fabric8</groupId>
            <artifactId>fabric8-maven-plugin</artifactId>
            <version>${version.fabric8.plugin}</version>
            <configuration>
                <generator>
            <includes>
              <include>wildfly-swarm</include>
            </includes>
            <excludes>
              <exclude>webapp</exclude>
            </excludes>
        </generator>
            </configuration>
            <executions>
```

图 3-16 Fabric8 Maven 插件配置

在 OpenShift 中创建项目，用于后续运行应用，如下图 3-17 所示：

```
[root@workstation ~]# oc new-project davidweitest
Now using project "davidweitest" on server "https://master.lab.example.com:443".

You can add applications to this project with the 'new-app' command. For example, try:

    oc new-app centos/ruby-22-centos7~https://github.com/openshift/ruby-ex.git

to build a new example application in Ruby.
```

图 3-17 创建 OpenShift 项目

运行 Maven 构建源码打包：

mvn clean package

构建成功以后，生成 War 和 Jar 包：

ls -lah target/*.*ar

-rw-rw-r--. 1 student student 57M Mar 7 08:05 target/hola-swarm.jar

-rw-rw-r--. 1 student student 61K Mar 7 08:05 target/hola.war

接下来，生成部署微服务所需的 OpenShift 资源。运行 fabric8:resource 以生成 OpenShift 资源文件，如下图 3-18 所示：

mvn fabric8:resource

```
[INFO] --- fabric8-maven-plugin:3.5.38:resource (default-cli) @ hola ---
[INFO] F8: Running in OpenShift mode
[INFO] F8: Using docker image name of namespace: hello
[INFO] F8: Running generator wildfly-swarm
[INFO] F8: wildfly-swarm: Using ImageStreamTag 'redhat-openjdk18-openshift:latest' as builder image
[INFO] F8: fmp-controller: Adding a default Deployment
[INFO] F8: fmp-service: Adding a default service 'hola' with ports [8080]
[INFO] F8: fmp-revision-history: Adding revision history limit to 2
[INFO] F8: f8-icon: Adding icon for deployment
[INFO] F8: f8-icon: Adding icon for service
[INFO] F8: validating /home/student/hello-microservices/hola/target/classes/META-INF/fabric8/openshift/hola-svc.
yml resource
[INFO] F8: validating /home/student/hello-microservices/hola/target/classes/META-INF/fabric8/openshift/hola-depl
oymentconfig.yml resource
[INFO] F8: validating /home/student/hello-microservices/hola/target/classes/META-INF/fabric8/openshift/hola-rout
e.yml resource
[INFO] F8: validating /home/student/hello-microservices/hola/target/classes/META-INF/fabric8/kubernetes/hola-svc
.yml resource
[INFO] F8: validating /home/student/hello-microservices/hola/target/classes/META-INF/fabric8/kubernetes/hola-dep
loyment.yml resource
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 36.910 s
[INFO] Finished at: 2019-06-07T22:44:46-04:00
[INFO] Final Memory: 31M/301M
[INFO] ------------------------------------------------------------------------
```

图 3-18 生成资源对象的过程

执行成功以后，生成如下三个 YAML 文件，如下图 3-19 所示：

```
[student@workstation openshift]$ pwd
/home/student/hello-microservices/hola/target/classes/META-INF/fabric8/openshift
[student@workstation openshift]$ ls
hola-deploymentconfig.yml  hola-route.yml  hola-svc.yml
[student@workstation openshift]$
```

图 3-19 生成的资源文件

下面我们查看生成的资源文件的部分内容，如下图 3-20、3-21 所示：

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    fabric8.io/git-commit: 58cc38d90eb73652726d05ae25ad75786ff43b6b
    fabric8.io/iconUrl: img/icons/wildfly.svg
    fabric8.io/git-branch: do292-hola-deploy
    prometheus.io/scrape: "true"
    prometheus.io/port: "9779"
  labels:
    expose: "true"
    app: hola
    provider: fabric8
    version: "1.0"
    group: com.redhat.training.msa
  name: hola
spec:
  ports:
  - name: http
    port: 8080
    protocol: TCP
    targetPort: 8080
  selector:
    app: hola
    provider: fabric8
```

图 3-20 hola-svc 的内容

```
---
apiVersion: apps.openshift.io/v1
kind: DeploymentConfig
metadata:
  annotations:
    fabric8.io/git-commit: 58cc38d90eb73652726d05ae25ad75786ff43b6b
    fabric8.io/iconUrl: img/icons/wildfly.svg
    fabric8.io/git-branch: do292-hola-deploy
    fabric8.io/metrics-path: dashboard/file/kubernetes-pods.json/?var-project=hola&var-version=1.0
  labels:
    app: hola
    provider: fabric8
    version: "1.0"
    group: com.redhat.training.msa
  name: hola
spec:
  replicas: 1
  revisionHistoryLimit: 2
  selector:
    app: hola
    provider: fabric8
    group: com.redhat.training.msa
  strategy:
    rollingParams:
      timeoutSeconds: 3600
    type: Rolling
  template:
    metadata:
      annotations:
        fabric8.io/git-commit: 58cc38d90eb73652726d05ae25ad75786ff43b6b
        fabric8.io/iconUrl: img/icons/wildfly.svg
```

图 3-21 hola-deploymentconfig 的部分内容

接下来，构建应用镜像。使用 fabric8:build 目标使用二进制 S2I 构建方法构建应用镜像，如下图 3-22 所示：

$ mvn fabric8:build

```
[INFO] ------------------------------------------------------------------------
[INFO] Building Red Hat Training MSA: hola 1.0
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- fabric8-maven-plugin:3.5.38:build (default-cli) @ hola ---
[INFO] F8: Using OpenShift build with strategy S2I
[INFO] F8: Running generator wildfly-swarm
[INFO] F8: wildfly-swarm: Using ImageStreamTag 'redhat-openjdk18-openshift:latest' as builder image
[INFO] Copying files to /home/student/hello-microservices/hola/target/docker/hola/1.0/build/maven
[INFO] Building tar: /home/student/hello-microservices/hola/target/docker/hola/1.0/tmp/docker-build.tar
[INFO] F8: [hola:1.0] "wildfly-swarm": Created docker source tar /home/student/hello-microservices/hola/target/d
ocker/hola/1.0/tmp/docker-build.tar
[INFO] F8: Creating BuildServiceConfig hola-s2i for Source build
[INFO] F8: Creating ImageStream hola
[INFO] F8: Starting Build hola-s2i
```

图 3-22 构建应用镜像

在 OpenShift 中确认 Build 和 Image Stream 已经创建，如下图 3-23 所示：

```
[student@workstation hola]$ oc get pods
NAME               READY     STATUS      RESTARTS   AGE
hola-s2i-1-build   0/1       Completed   0          22m
[student@workstation hola]$ oc get is
NAME      DOCKER REPO                                    TAGS      UPDATED
hola      docker-registry.default.svc:5000/hello/hola    1.0       13 minutes ago
[student@workstation hola]$
```

图 3-23 生成的构建

运行 fabric8:apply 将微服务部署到 OpenShift 集群，如下图 3-24 所示：

mvn fabric8:apply

```
[INFO] ------------------------------------------------------------------------
[INFO] Building Red Hat Training MSA: hola 1.0
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- fabric8-maven-plugin:3.5.38:apply (default-cli) @ hola ---
[INFO] F8: Using OpenShift at https://master.lab.example.com:443/ in namespace hello with manifest /home/student
/hello-microservices/hola/target/classes/META-INF/fabric8/openshift.yml
[INFO] OpenShift platform detected
[INFO] Using project: hello
[INFO] Creating a Service from openshift.yml namespace hello name hola
[INFO] Created Service: hola/target/fabric8/applyJson/hello/service-hola.json
[INFO] Using project: hello
[INFO] Creating a DeploymentConfig from openshift.yml namespace hello name hola
[INFO] Created DeploymentConfig: hola/target/fabric8/applyJson/hello/deploymentconfig-hola.json
[INFO] Creating Route hello:hola host: null
[INFO] F8: HINT: Use the command `oc get pods -w` to watch your pods start up
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 32.947 s
[INFO] Finished at: 2019-06-07T23:17:18-04:00
[INFO] Final Memory: 36M/225M
[INFO] ------------------------------------------------------------------------
```

图 3-24 部署应用到集群

Pod 构建完成后，就会执行部署，等待一会 Pod 就会启动完成，如下图 3-25 所示：

```
[student@workstation hola]$ oc get pods
NAME                READY     STATUS      RESTARTS   AGE
hola-1-zq82d        1/1       Running     0          5m
hola-s2i-1-build    0/1       Completed   0          30m
```

图 3-25 部署应用成功

查看应用路由，并用 curl 发起请求，如下图 3-26 所示：

```
[student@workstation ~]$ oc get route
NAME      HOST/PORT                          PATH      SERVICES   PORT      TERMINATION   WILDCARD
hola      hola-hello.apps.lab.example.com              hola       8080                    None
[student@workstation ~]$ curl http://hola-hello.apps.lab.example.com/api/hola
Hola de hola-hello.apps.lab.example.com
[student@workstation ~]$
```

图 3-26 验证应用

至此我们完成了基于 Fabric8 在 OpenShift 上发布应用的介绍。