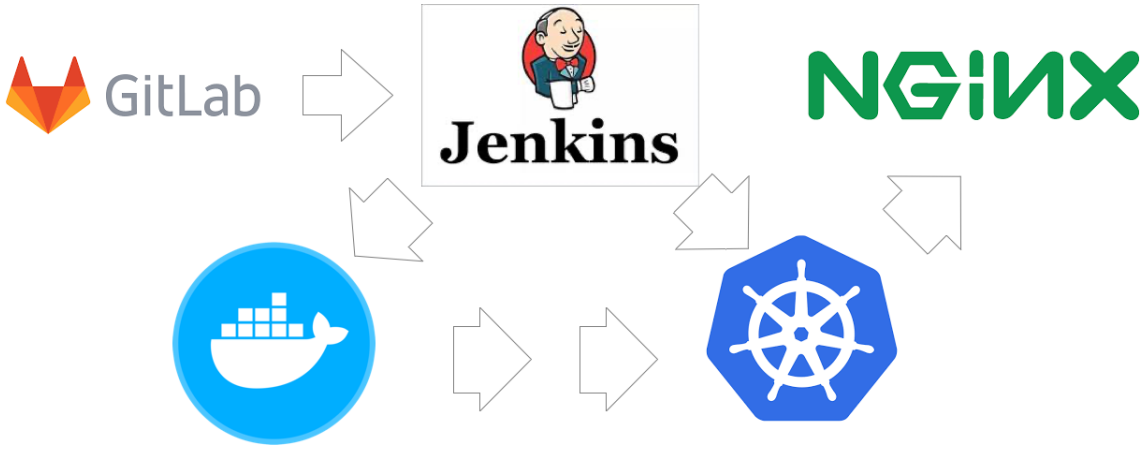
讲师(老司机)

# 容器虚拟化技术和自动化部署




# 实战skywalking

## skywalking版本

本章节所有案例均采用企业中主流的skywalking6.X系列。

### 服务器规划

生成至少两台虚拟机

| 主机名 | 主机IP |
| --- | --- |
| jenkinsagent-154 | 192.168.198.154 |
| skywalking-151 | 192.168.198.141 |

#### 141节点

141节点docker方式安装skywalking。

#### 154节点

初始化java、mvn、git、docker环境。

## 安装skywalking

#### 官网地址

```
根据官网地址选择TAG版本后，进入docker目录中，修改docker-compose.yml文件内容
https://github.com/apache/skywalking

6.6.0版本地址
https://github.com/apache/skywalking/tree/v6.6.0/docker
```

**docker-compose.yml**

```yaml
version: '3.3'
services:
  elasticsearch:
    image: elasticsearch:7.5.1
    container_name: elasticsearch
    restart: always
    ports:
      - 9200:9200
    environment:
      discovery.type: single-node
      TZ: Asia/Shanghai
    ulimits:
      memlock:
        soft: -1
        hard: -1
  oap:
    image: apache/skywalking-oap-server:6.6.0-es7
    container_name: oap
    depends_on:
      - elasticsearch
    links:
      - elasticsearch
    restart: always
    ports:
      - 11800:11800
      - 12800:12800
    environment:
      SW_STORAGE: elasticsearch    # 指定ES版本
      SW_STORAGE_ES_CLUSTER_NODES: elasticsearch:9200
      TZ: Asia/Shanghai
  ui:
    image: apache/skywalking-ui:6.6.0
    container_name: ui
    depends_on:
      - oap
    links:
      - oap
    restart: always
    ports:
      - 8080:8080
    environment:
      SW_OAP_ADDRESS: oap:12800
      TZ: Asia/Shanghai
```

### ES测试

```
http://192.168.198.141:9200/
```

### skywalking测试

```
http://192.168.198.141:8080/
```

## skywalking集成war包

### tomcat8下载

```
官网下载地址：
https://mirror.bit.edu.cn/apache/tomcat/tomcat-8/v8.5.57/bin/apache-tomcat-8.5.57.tar.gz
```

### skywalking客户端配置

```
将skywalking源码包上传154节点/opt目录中

cd /opt
tar zxf apache-skywalking-apm-6.6.0.tar.gz

mv apache-skywalking-apm-bin/ skywalking

rm -rf apache-skywalking-apm-6.6.0.tar.gz
```

### springmvc项目

#### idea配置

idea开发工具mvn工程。使用springMVC+jsp方式。需要进行war包配置。需要将web目录重新更名为webapp并将webapp目录复制到maven工程指定的src/main/中。

#### pom.xml文件

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
```

```
    <artifactId>springmvcdemo2</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>war</packaging>
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>1.8</maven.compiler.source>
        <maven.compiler.target>1.8</maven.compiler.target>
        <spring.version>5.0.5.RELEASE</spring.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>${spring.version}</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-beans</artifactId>
            <version>${spring.version}</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>${spring.version}</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-jdbc</artifactId>
            <version>${spring.version}</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-aspects</artifactId>
            <version>${spring.version}</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-jms</artifactId>
            <version>${spring.version}</version>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context-support</artifactId>
            <version>${spring.version}</version>
        </dependency>
        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>servlet-api</artifactId>
            <version>2.4</version>
            <scope>provided</scope>
        </dependency>
    </dependencies>
    <build>
        <finalName>skywalking-springmvc</finalName>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
```

```
                <version>2.3.2</version>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-war-plugin</artifactId>
                <version>2.2</version>
            </plugin>
            <plugin>
                <groupId>org.apache.tomcat.maven</groupId>
                <artifactId>tomcat7-maven-plugin</artifactId>
                <configuration>
                    <!-- 指定端口 -->
                    <port>8082</port>
                    <!-- 请求路径 -->
                    <path>/</path>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

**web.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
          http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
           version="3.0">
    <servlet>
        <servlet-name>springmvc</servlet-name>
        <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>classpath:applicationContext-web.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>springmvc</servlet-name>
        <url-pattern>*.controller</url-pattern>
    </servlet-mapping>
</web-app>

```

**applicationContext-web.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
        xmlns:p="http://www.springframework.org/schema/p"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:mvc="http://www.springframework.org/schema/mvc"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
             http://www.springframework.org/schema/beans/spring-beans.xsd
             http://www.springframework.org/schema/mvc
             http://www.springframework.org/schema/mvc/spring-mvc.xsd
             http://www.springframework.org/schema/context
             http://www.springframework.org/schema/context/spring-
context.xsd">
         <context:component-scan base-package="com.lagou.controller"/>
</beans>
```

**HelloController**

```
@Controller
@RequestMapping("/hello")
//注意访问项目URL地址为：hello/sayHello.controller
public class HelloController {
    @RequestMapping("/sayHello")
    @ResponseBody
    public String sayHello(String name){
        return "hello skywalking";
    }
}
```

**index.jsp**

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
  <head>
    <title>skywaling springmv war</title>
  </head>
  <body>
        hello skywaling springmv war!!!
  </body>
</html>
```

**运行测试**

```
本地idea开发工具中运行项目
mvn clean tomcat7:run

测试index页面
http://localhost:8082

测试URL地址
http://localhost:8082/hello/sayHello.controller
```

### 打包项目

```
mvn clean package
```

## 配置agent

```
首先我们复制一份agent，防止与其他应用程序使用的agent冲突
cd /opt/skywalking
cp -r agent agent_springmvc

cd /opt/skywalking/agent_springmvc/config
vi agent.config

我们在配置中找到这么一行：
agent.service_name=${SW_AGENT_NAME:Your_ApplicationName}

这里的配置含义是可以读取到SW_AGENT_NAME配置属性，如果该配置没有指定，那么默认名称为
Your_ApplicationName。这里我们把Your_ApplicationName替换成skywalking_springmvc

agent.service_name=${SW_AGENT_NAME:skywalking_springmvc}
```

## 配置tomcat

### 配置tomcat

```
将apache-tomcat-8.5.57.tar.gz上传154节点/opt目录中

cd /opt
tar zxf apache-tomcat-8.5.57.tar.gz

mv apache-tomcat-8.5.57 tomcat8
rm -rf apache-tomcat-8.5.57.tar.gz

将war包复制到webapps目录
cd /data
mv /data/skywalking-springmvc.war /opt/tomcat8/webapps/
```

### 集成skywalking

在catalina.sh文件顶部添加skywalking集成配置

```
vi /opt/tomcat8/bin/catalina.sh

CATALINA_OPTS="$CATALINA_OPTS -
javaagent:/opt/skywalking/agent_springmvc/skywalking-agent.jar -
Dskywalking.collector.backend_service=192.168.198.141:11800"; export
CATALINA_OPTS
```

## 启动tomcat

```
启动tomcat8
cd /opt/tomcat8/bin

./startup.sh


http://192.168.198.154:8080/

http://192.168.198.154:8080/skywalking-springmvc/
http://192.168.198.154:8080/skywalking-springmvc/hello/sayHello.controller

停止tomcat8
./shutdown.sh
```

# skywalking集成jar包

## springboot项目

### pom.xml文件

新增跳过单元测试plugin配置

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.3.3.RELEASE</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.lagou</groupId>
    <artifactId>skywalkingdemo1</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>skywalkingdemo1</name>
    <description>Demo project for Spring Boot</description>

    <properties>
        <java.version>1.8</java.version>
    </properties>

    <dependencies>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
```

```
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
            <exclusions>
                <exclusion>
                    <groupId>org.junit.vintage</groupId>
                    <artifactId>junit-vintage-engine</artifactId>
                </exclusion>
            </exclusions>
        </dependency>
    </dependencies>

    <build>
        <finalName>skywalkingdemo1</finalName>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
            <!--跳过单元测试-->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-surefire-plugin</artifactId>
                <configuration>
                    <skip>true</skip>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

**application.yml文件**

为防止与其他项目端口号冲突，将项目端口号更改为8081

```
server:
  port: 8081
```

**HelloController**

```
@RestController
public class HelloController {
    //正常访问接口
    @RequestMapping("/sayBoot")
    public String sayBoot(){
        return "Hello Boot!";
    }

    //异常访问接口
    @RequestMapping("/exception")
    public String exception(){
        int i = 1/0;
```

```
        return "Hello Boot!";
    }
}
```

### 本地测试项目

```
idea开发工具中启动项目，进行测试
http://localhost:8081/sayBoot
```

### 打包项目

```
mvn clean package
```

### 配置agent

```
首先我们复制一份agent，防止与其他应用程序使用的agent冲突
cd /opt/skywalking

cp -r agent agent_springbootdemo1

cd agent_springbootdemo1/config
vi agent.config

agent.service_name=${SW_AGENT_NAME:skywalking_springboodedemo1}
```

### 启动服务

```
将skywalkingdemo1.jar上传到154节点的/data目录
cd /data

java -javaagent:/opt/skywalking/agent_springbootdemo1/skywalking-agent.jar -Dskywalking.collector.backend_service=192.168.198.141:11800 -jar skywalkingspringbootdemo1.jar
```

### 测试项目

```
http://192.168.198.154:8081/sayBoot
```

## skywalking集成docker

## 准备工作

```
新建自定义镜像目录
mkdir -p /data/skywalking

将skywalkingdemo1.jar复制到/data/skywalking目录中
cp /data/skywalkingdemo1.jar /data/skywalking/
```

## 配置agent

```
将agent目录复制到/data/skywalking备用
cd /opt/skywalking

cp -r /opt/skywalking/agent /data/skywalking

cd /data/skywalking/agent/config

修改配置文件
vi agent.config

agent.service_name=${SW_AGENT_NAME:skywalking_dockerdemo1}
```

## Dockerfile

```
FROM openjdk:8-alpine3.9
# 作者信息
MAINTAINER laosiji Docker skywalking springboot "laosiji@lagou.com"
# 修改源
RUN echo "http://mirrors.aliyun.com/alpine/latest-stable/main/" > /etc/apk/repositories && \
    echo "http://mirrors.aliyun.com/alpine/latest-stable/community/" >> /etc/apk/repositories

# 安装需要的软件，解决时区问题
RUN apk --update add curl bash tzdata && \
    rm -rf /var/cache/apk/*

#修改镜像为东八区时间
ENV TZ Asia/Shanghai

COPY agent/ /opt/skyagent/

VOLUME /tmp
ARG JAR_FILE
COPY ${JAR_FILE} app.jar

EXPOSE 8081


```

```
ENV JAVA_OPTS="-server -Xmx256m -Xms256m" AGENT_SERVICE_NAME="default"
AGENT_COLLECTOR_ADDRESS="192.168.198.141:11800"

ENTRYPOINT java -javaagent:/opt/skyagent/skywalking-
agent.jar=agent.service_name=${AGENT_SERVICE_NAME},collector.backend_servic
e=${AGENT_COLLECTOR_ADDRESS} ${JAVA_OPTS} -jar /app.jar
```

### 制作镜像

```
cd /data/skywalking
docker build --rm -t lagou/skywalkingdemo1:1.0 --build-arg
JAR_FILE=skywalkingspringbootdemo1.jar .
```

### 运行镜像

```
docker run -itd --name skywalkingdemo1 -p 8081:8081 lagou/skywalkingdemo1:1.0
```

### 测试项目

```
docker logs -f skywalkingdemo1

http://192.168.198.154:8081/sayBoot

docker stop skywalkingdemo1
docker rm skywalkingdemo1
```

## skywalking集成maridb

### 基础镜像

```
docker pull mariadb:10.5.2

docker load -i mariadb.10.5.2.tar
```

### 运行镜像

```
docker run -itd --name mariadb --restart always --privileged=true -p
3306:3306 -v /data/mysql:/var/lib/mysql  -e MYSQL_ROOT_PASSWORD=admin
mariadb:10.5.2 --character-set-server=utf8mb4 --collation-
server=utf8mb4_unicode_ci
```

### 创建数据库

### 测试连接

```
使用sqlLog客户端测试是否能正确连接mariadb
192.168.198.141
root
admin
```

### 创建数据库

```
创建lagou数据库
```

### 创建用户表

```
创建tbuser表
CREATE TABLE `tbuser` (
  `userid` int(11) NOT NULL AUTO_INCREMENT,
  `username` varchar(20) COLLATE utf8_bin DEFAULT NULL,
  `password` varchar(20) COLLATE utf8_bin DEFAULT NULL,
  `userroles` varchar(2) COLLATE utf8_bin DEFAULT NULL,
  `nickname` varchar(50) COLLATE utf8_bin DEFAULT NULL,
  PRIMARY KEY (`userid`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8 COLLATE=utf8_bin

```

### 新增测试数据

```
INSERT INTO tbuser (username,PASSWORD,userroles,nickname) VALUES
('admin','1234','04','管理员'),('lagou','1234','03','拉勾教育')
```

## springboot项目

### pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.3.3.RELEASE</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.lagou</groupId>
    <artifactId>skywalkingdemo1</artifactId>
    <version>0.0.1-SNAPSHOT</version>
```

```
    <name>skywalkingdemo1</name>
    <description>Demo project for Spring Boot</description>

    <properties>
        <java.version>1.8</java.version>
        <mybatispluins.version>3.3.2</mybatispluins.version>
    </properties>

    <dependencies>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <dependency>
            <groupId>com.baomidou</groupId>
            <artifactId>mybatis-plus-boot-starter</artifactId>
            <version>${mybatispluins.version}</version>
        </dependency>
        <dependency>
            <groupId>org.mariadb.jdbc</groupId>
            <artifactId>mariadb-java-client</artifactId>
            <version>2.6.0</version>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
            <exclusions>
                <exclusion>
                    <groupId>org.junit.vintage</groupId>
                    <artifactId>junit-vintage-engine</artifactId>
                </exclusion>
            </exclusions>
        </dependency>
    </dependencies>

    <build>
        <finalName>skywalkingdemo1</finalName>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
            <!--跳过单元测试-->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-surefire-plugin</artifactId>
                <configuration>
                    <skip>true</skip>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

**application.yml**

为防止与其他项目端口号冲突，将项目端口号更改为8082

```yaml
server:
  port: 8082
spring:
  datasource:
    driver-class-name: org.mariadb.jdbc.Driver
    username: root
    password: admin
    url: jdbc:mariadb://192.168.198.141:3306/lagou?characterEncoding=utf-8&useSSL=false&useTimezone=true&serverTimezone=GMT%2B8

mybatis-plus:
  type-aliases-package: com.lagou.skywalkingdemo2.entity
  mapper-locations: mapper/*.xml
  configuration:
    log-impl: org.apache.ibatis.logging.stdout.StdOutImpl #配置日志打印方式。不使用mybatis的日志信息。使用mp的日志配置
```

**测试数据库连接**

```java
@SpringBootTest(classes = Skywalkingdemo1Application.class)
@ExtendWith(SpringExtension.class)
public class Skywalkingdemo1ApplicationTests {
    @Resource
    private DataSource dataSource;
    @Test
    public void testConnection() throws SQLException {
        Connection connection = this.dataSource.getConnection();
        System.out.println("connection =====> " + connection);
    }
}
```

**启动类**

```java
@SpringBootApplication
@MapperScan(basePackages = "com.lagou.skywalkingdemo1.mapper")
public class Skywalkingdemo1Application {

    public static void main(String[] args) {
        SpringApplication.run(Skywalkingdemo1Application.class, args);
    }

}
```

**mybatisplus**

```
官网地址：
https://baomidou.com/

与easyCode插件组合，威力倍增！
```

**实体类**

```
@TableName(value = "tbuser")
public class Tbuser {
    @TableId(type = IdType.AUTO)
    private Integer userid;
    private String username,password,userroles,nickname;
    ......省略getter和setter、构造方法、toString等方法
}
```

**UserMapper接口**

```
public interface UserMapper extends BaseMapper<Tbuser> {
}
```

**UserService接口**

```
public interface UserService {
    List<Tbuser> queryUsers();
}
```

**UserServiceImpl实现类**

```
@Service
public class UserServiceImpl implements UserService {
    @Resource
    private UserMapper userMapper;

    @Override
    public List<Tbuser> queryUsers() {
        return this.userMapper.selectList(null);
    }
}
```

**测试类**

```
@SpringBootTest(classes = Skywalkingdemo1Application.class)
@ExtendWith(SpringExtension.class)
public class Skywalkingdemo1ApplicationTests {
    @Resource
    private DataSource dataSource;
    @Resource
    private UserService userService;

    @Test
    public void testQueryUsers() {
        List<Tbuser> tbusers = this.userService.queryUsers();
        //tbusers.forEach(u -> System.out.println("u ====> " + u));
    }
    @Test
    public void testConnection() throws SQLException {
        Connection connection = this.dataSource.getConnection();
```

```
        System.out.println("connection =====> " + connection);
    }
}
```

**控制器**

```
@RestController
public class UserController {

    @Resource
    private UserService userService;

    @GetMapping("/users")
    public List<Tbuser> queryUsers() {
        return this.userService.queryUsers();
    }
}
```

**本地测试项目**

```
http://localhost:8082/users
```

## 打包项目

```
mvn clean package
```

## 配置agent

```
首先我们复制一份agent，防止与其他应用程序使用的agent冲突
cd /opt/skywalking

cp -r agent agent_springbootdemo2

cd agent_springbootdemo2/config
vi agent.config

agent.service_name=${SW_AGENT_NAME:skywalking_springboodedemo2}
```

## 启动服务

```
将skywalkingdemo2.jar上传到154节点的/data目录
cd /data

java -javaagent:/opt/skywalking/agent_springbootdemo2/skywalking-agent.jar -Dskywalking.collector.backend_service=192.168.198.141:11800 -jar skywalkingspringbootdemo2.jar
```

## 测试项目

```
http://192.168.198.154:8082/users
```

## 切换mysql数据库

### pom.xml

```
        <mysql.version>5.1.47</mysql.version>

        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <scope>runtime</scope>
        </dependency>
```

### application.yml

```
spring:
  datasource:
    driver-class-name: com.mysql.jdbc.Driver
    username: root
    password: admin
    url: jdbc:mysql://192.168.198.141:3306/lagou?characterEncoding=utf-8&useSSL=false&useTimezone=true&serverTimezone=GMT%2B8
```

### 数据库

```
1.从mariadb数据库导出lagou数据库
2.把lagou数据库导入mysql5.7中

3.docker stop mariadb
4.docker rm mariadb
5.docker rmi -f mariadb.10.5.2.tar
```

### 基础镜像

```
docker官网地址：
https://hub.docker.com/r/bitnami/mariadb

docker pull bitnami/mysql:5.7.30
```

### 运行镜像

```
此镜像为非根镜像。没有root权限。挂载目录需要授权，命令如下：
mkdir -p /data/mysql/
chown -R 1001 /data/mysql/

docker run -itd --name mysql5.7 --restart always -p 3306:3306 -e
MYSQL_ROOT_PASSWORD=admin -e TZ=Asia/Shanghai   -v
/data/mysql:/bitnami/mysql/ bitnami/mysql:5.7.30
```

### 非根容器介绍

```
官方文档：
https://docs.bitnami.com/tutorials/work-with-non-root-containers/


如果需要，以根用户的身份运行非根容器。 这还会将所有文件权限自动授予容器，因为容器具有更高的
特权。
解决办法：
1.挂载目录时授权
2.启动容器使用:-u 0:0，以root用户登录非根容器
```

### 挂载卷

```
非根容器挂载数据卷一般采用两种方式：

1.查看官方文档，根据官网要求，给目录授予对应权限。例如：
mkdir -p /data/mysql/
chown -R 1001 /data/mysql/

2.以根用户(root)的身份运行非根容器。 这还会将所有文件权限自动授予容器，因为容器具有更高的
特权。启动容器使用:-u 0:0，以root用户登录非根容器
docker run -itd --name mysql5.7 --restart always -p 3306:3306  -u 0:0  -e
MYSQL_ROOT_PASSWORD=admin -e TZ=Asia/Shanghai   -v
/data/mysql:/bitnami/mysql/ bitnami/mysql:5.7.30
```

### 打包项目

```
mvn clean package

上传jar包至154节点
```

**启动服务**

```
java -javaagent:/opt/skywalking/agent_springbootdemo2/skywalking-agent.jar -Dskywalking.collector.backend_service=192.168.198.141:11800 -jar skywalkingspringbootdemo2.jar
```

**测试项目**

```
http://192.168.198.154:8082/users
```

# Java Agent配置方式

在之前的案例中，我们每次部署java应用都需要复制一份agent，修改其中的服务名称，这样配置非常麻烦。Skywalking作者考虑到这些问题，提供几种配置方式，通过启动命令动态指定服务名，这样agent只需要部署一
份即可。Skywalking支持的几种配置方式：

## 系统属性(-D方式)

使用 `-Dskywalking.` + `agent.config` 配置文件中的key 即可 。例如： `agent.config` 文件中有一个属性名为 `agent.service_name` ，那么如果使用系统属性的方式，则可以写成

语法规则

```
-Dskywalking.agent.service_name=skywalkingdemo2
```

```
java -javaagent:/opt/skywalking/agent_springbootdemo2/skywalking-agent.jar -Dskywalking.agent.service_name=你想设置的值  -Dskywalking.collector.backend_service=192.168.198.141:11800 -jar skywalkingspringbootdemo2.jar
```

## 代理选项(javaagent方式配置)

在JVM参数中的代理路径之后添加属性即可。 语法规则如下：

```
-javaagent:/opt/skywalking/agent_springbootdemo2/skywalking-agent.jar=[option1]=[value1],[option2]=[value2],[option3]=[value3]
```

通过 如下进行 agent.service_name 的覆盖：

```
java -javaagent:/opt/skywalking/agent_springbootdemo2/skywalking-agent.jar=agent.service_name=你想设置的值  -Dskywalking.collector.backend_service=192.168.198.141:11800 -jar skywalkingspringbootdemo2.jar
```

### 系统环境变量

在centos系统中设置变量：SW_AGENT_NAME的值。

### 配置文件

`agent.config` 文件中默认的大写值，都可以作为环境变量引用。 例如，agent.config 中有如下内容

```
agent.service_name=${SW_AGENT_NAME:Your_ApplicationName}
```

这说明Skywalking会读取名为 `SW_AGENT_NAME` 的环境变量。

### 覆盖优先级

javaagent配置 > 系统属性 >系统环境变量 > 配置文件中的值

所以我们的启动命令可以修改为：

```
java -javaagent:/opt/skywalking/agent_springbootdemo2/skywalking-agent.jar -Dskywalking.agent.service_name=你想设置的值  -Dskywalking.collector.backend_service=192.168.198.141:11800 -jarskywalkingspringbootdemo2.jar
```

或者

```
java -javaagent:/opt/skywalking/agent_springbootdemo2/skywalking-agent.jar=agent.service_name=你想设置的值  -Dskywalking.collector.backend_service=192.168.198.141:11800 -jar skywalkingspringbootdemo2.jar
```

例如：

```
java -javaagent:/opt/skywalking/agent_springbootdemo2/skywalking-agent.jar -Dskywalking.agent.service_name=skywalkingdemo3  -Dskywalking.collector.backend_service=192.168.198.141:11800 -jar skywalkingspringbootdemo2.jar
```