## 什么是Spring boot？

Spring Boot是由Pivotal团队提供的全新框架，其设计目的是用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。

## Spring Boot特性

1. 创建独立的Spring应用程序

2. 嵌入的Tomcat，无需部署WAR文件

3. 简化Maven配置

4. 自动配置Spring

5. 提供生产就绪型功能，如指标，健康检查和外部配置

6.开箱即用，没有代码生成，也无需XML配置。

## Spring Boot热部署

spring-boot-devtools 是一个为开发者服务的一个模块，其中最重要的功能就是自动应用代码更改到最新的App上面去。原理是在发现代码有更改之后，重新启动应用，但是速度比手动停止后再启动还要更快，更快指的不是节省出来的手工操作的时间。

其深层原理是使用了两个ClassLoader，一个Classloader加载那些不会改变的类（第三方Jar包），另一个ClassLoader加载会更改的类，称为 restart ClassLoader

,这样在有代码更改的时候，原来的restart ClassLoader 被丢弃，重新创建一个restart ClassLoader，由于需要加载的类相比较少，所以实现了较快的重启时间（5秒以内）。

修改类-->保存：应用会重启

修改配置文件-->保存：应用会重启

修改页面-->保存：应用会重启，页面会刷新

## 集成mybatis

1、新建一个新的Maven Project;

2、需要在pom.xml文件添加相应的依赖，

比如：mysql驱动；

PageHelper分页插件，需要版本号；

3、编写启动类，大部分和之前的代码是一样的，需要的注意的是：

需要添加一个注解@MapperScan --指定MyBatis持久类的位置；

4、编写一个测试的实体类Demo;

5、编写一个DemoMapper,使用@Select和@Save进行数据库操作；

使用@Options配置返回的主键信息；

6、编写DemoService;

7、编写DemoController;

8、添加分页配置信息，需要添加一个MyBatisConfiguration;

9、使用PageHelper.startPage(pageNum,pageSize)进行分页

## Spring Cloud特点

约定优于配置

开箱即用、快速启动

适用于各种环境

PC Server

云环境

容器(Docker)

轻量级的组件

服务发现

Eureka

组件的支持很丰富，功能很齐全

配置中心

注册中心

智能路由

选型中立

服务发现

Eureka

Zookeeper

Consul

## eureka+ribbon

microservice-consumer-movie-ribbon 8010

microservice-provider-user 7900 7901

microservice-provider-user2 7902 7903

http://192.168.88.1:8010/test

## Spring Cloud config

其他项目

阿里diamond 携程 Apollo 百度disconf

优势

集中管理

不同环境不同配置

运行期间动态调整配置

自动刷新

加载顺序：bootstrap, config server, application

HTTP服务具有以下格式的资源：

配置地址<https://github.com/hobertm/spring-boot>

/{application}/{profile}[/{label}]

/{application}-{profile}.yml

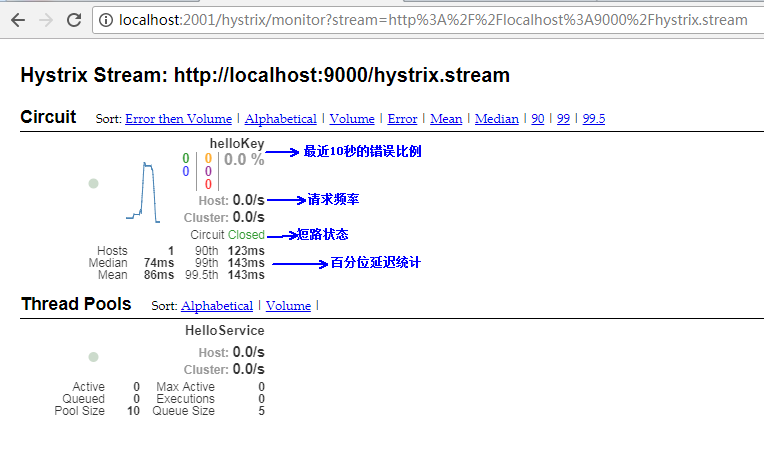
/{label}/{application}-{profile}.yml

/{application}-{profile}.properties

/{label}/{application}-{profile}.properties

curl -X POST <http://localhost:8081/refresh>刷新

## hystrix+turbine



实心圆：1、通过颜色的变化代表了实例的健康程度，健康程度从绿色、黄色、橙色、红色递减。2、通过大小表示请求流量发生变化，流量越大该实心圆就越大。所以可以在大量的实例中快速发现故障实例和高压实例。

曲线：用来记录2分钟内流量的相对变化，可以通过它来观察流量的上升和下降趋势。

microservice-consumer-movie-ribbon-with-hystrix 8010 8011

microservice-consumer-movie-feign-with-hystrix 7901

microservice-provider-user 7900

microservice-hystrix-turbine 8031

microservice-hystrix-dashboard 8030

<http://192.168.88.1:8030/hystrix>页面配置

http://192.168.88.1:8031/turbine.stream