HUAWEI

21天微服务实战营-Day5

华为云DevCloud & ServiceStage服务联合出品

Day5 教你如何配置你的微服务

大纲

- microservice.yaml配置文件
- 环境变量、System property
- 动态配置
- 通过API获取配置
- 日志配置

microservice.yaml配置文件

微服务实例启动时会从classpath下加载microservice.yaml配置文件：

- 如果多个jar包下都有microservice.yaml文件，那么他们都会被加载
- 磁盘目录下的microservice.yaml配置文件的优先级高于jar包内的配置文件
- 可以通过在microservice.yaml文件内配置servicecomb-config-order来指定优先级

环境变量、System property

微服务实例启动时也会从环境变量、系统属性中加载配置：

- Linux系统的环境变量不允许有点号”.”，但CSEJavaSDK框架会自动将配置项key中的下划线映射为点号，因此我们可以将点转换为下划线来配置环境变量
- 环境变量的优先级高于配置文件，system property的优先级高于环境变量

动态配置

微服务实例连接配置中心后，可以从配置中心获取动态配置：

- 动态配置的优先级是最高的，并且可以在运行时刷新
- 服务治理所使用的诸多控制逻辑也是由配置项来控制的。实现服务动态治理的方式就是通过配置中心动态下发配置项

通过API获取配置

CSEJavaSDK使用统一的API来获取配置，用户使用配置的时候，不需要关心从环境变量或者配置中心来读取配置，框架已经自动为用户从各个配置来源读取配置，并根据优先级规则将所有配置进行了合并和覆盖。

优先级： 动态配置 > system property > 环境变量 > 配置文件

通过API获取配置

将provider服务的sayHello方法的应答的前缀从固定的"Hello,"改为从配置项获取：

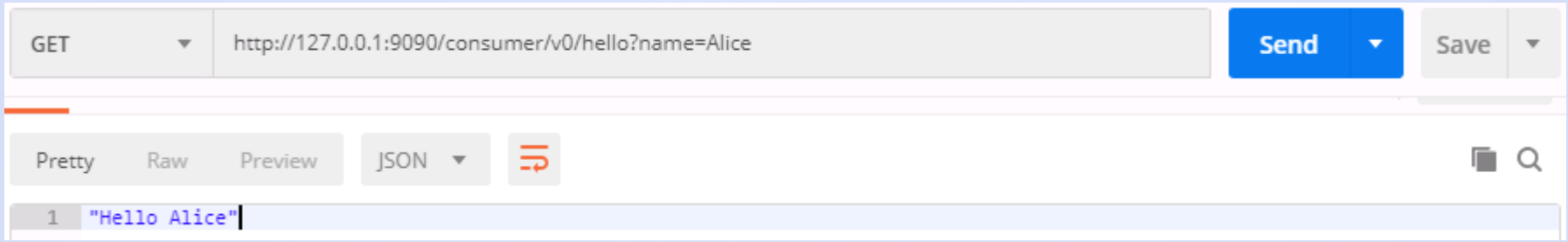
```
private DynamicStringProperty sayHelloPrefix = DynamicPropertyFactory
    .getInstance().getStringProperty( propName: "hello.sayHelloPrefix", defaultValue: "");

@RequestMapping(path = "/hello/{name}", method = RequestMethod.GET)
public String sayHello(@PathVariable(value = "name") String name) {
    return sayHelloPrefix.getValue() + name;
}
```

在microservice.yaml文件中加上配置：

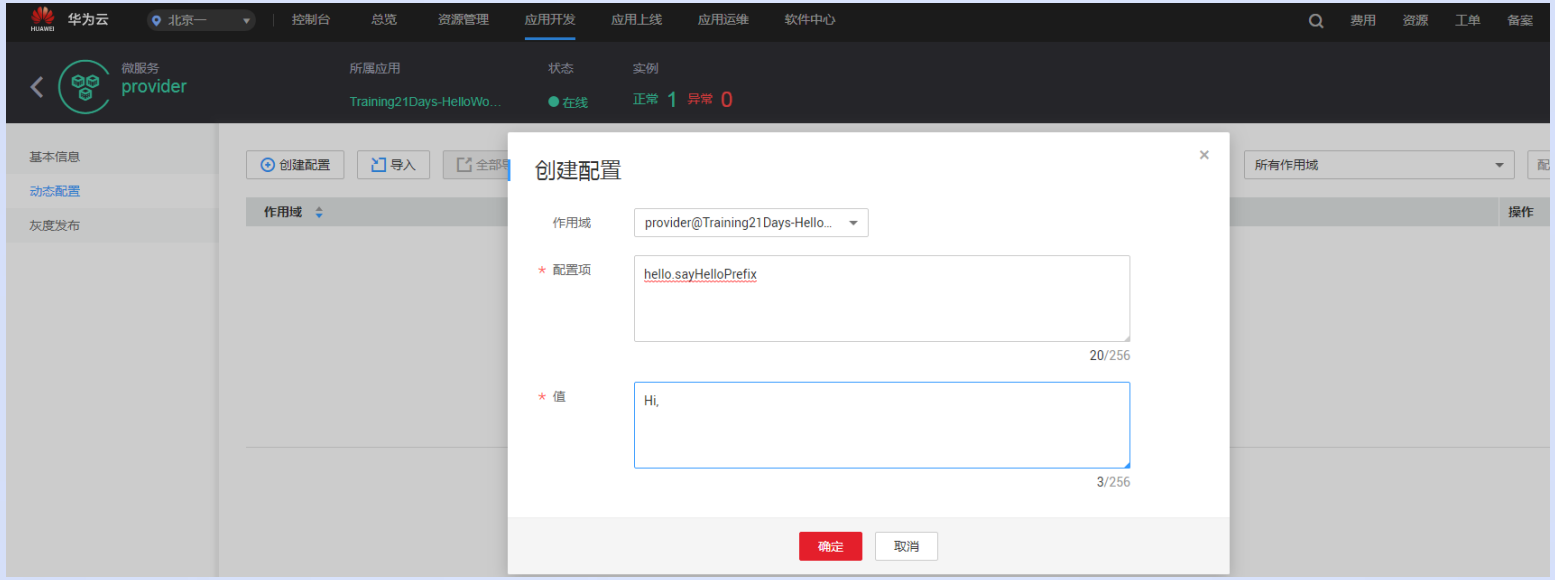
```
hello:
  sayHelloPrefix: "Hello "
```

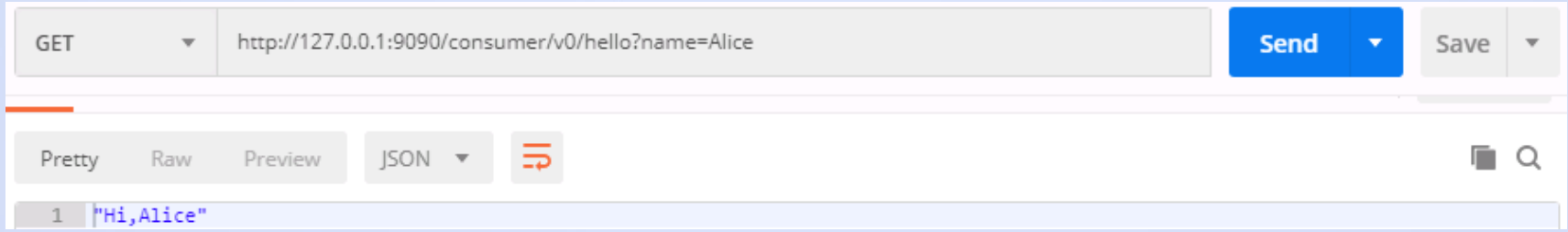
此时启动服务进行调用，情况如下：



通过API获取配置

将provider服务的详情页面配置hello.sayHelloPrefix=Hi,，再次调用服务发现应答已经产生变化：





日志配置

- CSEJavaSDK默认使用的日志框架是Log4j，并且给出了一份默认的配置，在org.apache.servicecomb:foundation-common包的log4j.properties文件内。
- CSEJavaSDK提供了accesslog功能，可以在传输方式为REST over Vertx的条件下使用，accesslog默认也是基于Log4j打印的，配置文件在org.apache.servicecomb:transport-rest-vertx包的log4j.properties文件内。
- 如果要覆盖默认的日志配置，在项目的resources/config目录下配置一份log4j.properties文件即可。

Thank You