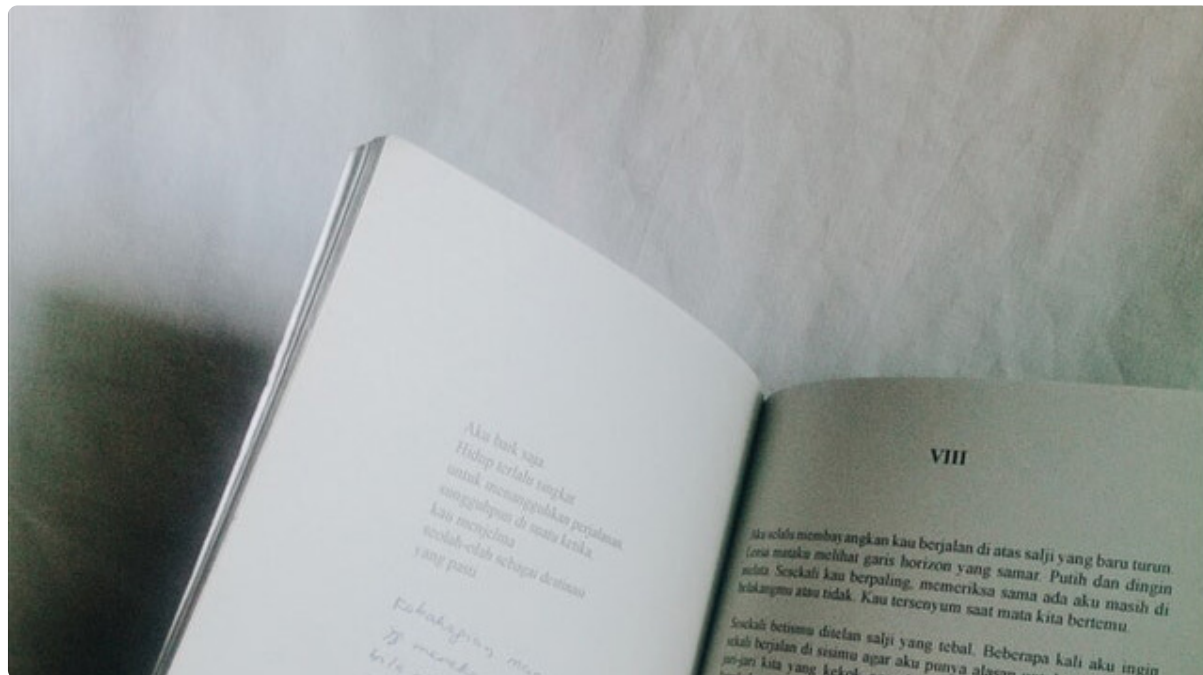


34 Spring Boot Admin 实践

更新时间：2019-07-31 09:50:35



理想的书籍是智慧的钥匙。

——列夫·托尔斯泰

13-2 Spring Boot Admin 实践

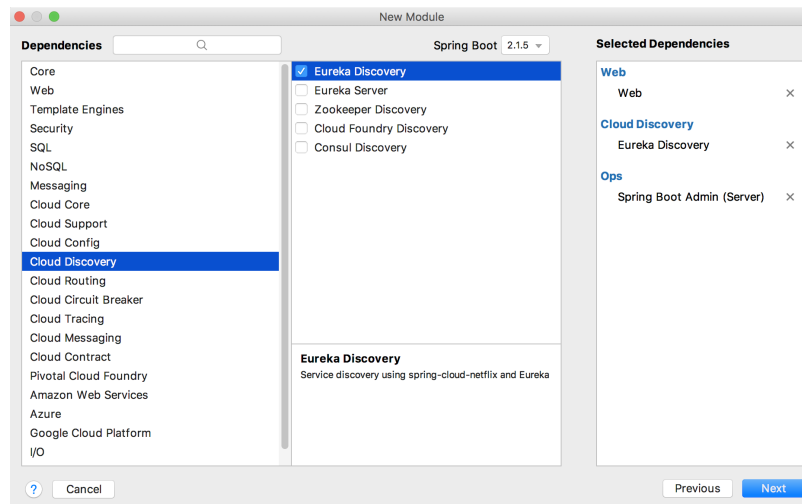
上篇文章和大家聊了 Spring Boot Admin 的基本功能和基本使用，使大家了解到使用 Spring Boot Admin 我们可以方便的通过一个比较炫的 UI 页面将应用的监控信息聚合展示出来，这样就不需要运维人员去挨个查看，使用在数据分析之中也非常方便。

Spring Boot Admin 虽然好用，但由于我们上篇文章中没有引入微服务，这就导致 Admin Client 上需通过硬编码的方式来指定 Admin Server 的位置，一旦 Admin Server 发生变化，所有的 Admin Client 都需要修改重新部署，这样成本就太高了。通过引入 Eureka 实现服务化，我们就可以有效解决这个问题。

服务化

首先我们来创建一个名为 monitor 的 Maven 父工程，然后创建一个名为 eureka 的 module 作为我们的注册中心。关于 eureka 的创建与启动，我这里就不再多介绍了，通过前面文章的学习，相信大家对于这个应该已经很熟悉了。eureka 配置完成之后，将 eureka 启动。

然后我们再来创建一个 Spring Boot Admin 项目，这次添加的依赖除了 Web、Admin Server 之外，还需要添加 Eureka Discovery，如下：



创建成功之后，核心依赖如下：

```
<dependencies>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>
<dependency>
<groupId>de.codecentric</groupId>
<artifactId>spring-boot-admin-starter-server</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
</dependency>
</dependencies>
```

创建成功之后，我们在 application.properties 中添加如下配置，将 adminserver 注册到 eureka 上：

```
spring.application.name=adminserver
eureka.client.service-url.defaultZone=http://localhost:1111/eureka
management.endpoints.web.exposure.include=*
```

这里我们添加了三行配置：

- 第一行配置表示设置服务名称；
- 第二行配置表示配置服务注册地址；
- 第三行配置表示开启所有被隐藏的端点。

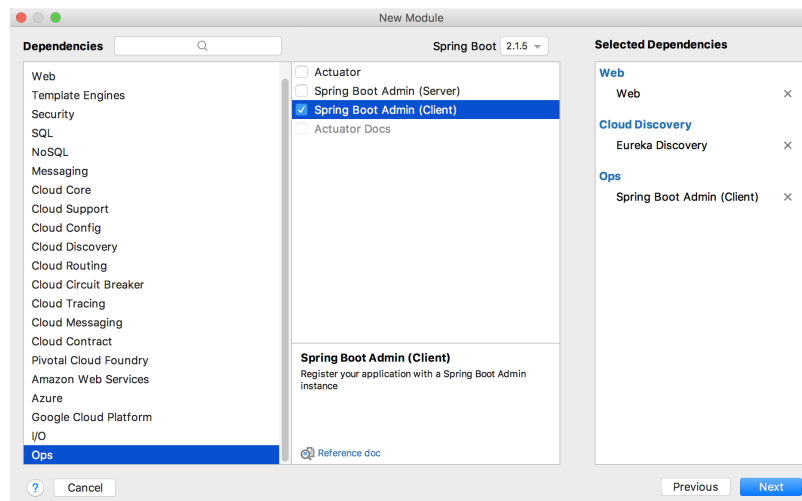
可能有人会觉得奇怪，这个是 Admin Server 项目，为什么还需要开启被隐藏的端点？原因很简单，因为 Admin Server 本身的运行状况我一样也要进行监控，所以这里要开启 Admin Server 的端点数据。

配置完成后，再在启动类上添加 `@EnableAdminServer` 注解，如下：

```
@SpringBootApplication
@EnableAdminServer
public class AdminserverApplication {

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

然后就可以启动 `adminserver` 项目了。启动成功之后，我们再来创建一个 `adminclient` 项目，和上篇文章相比，这里的 `adminclient` 多出来一个依赖，就是 `Eureka Discovery`，如下：



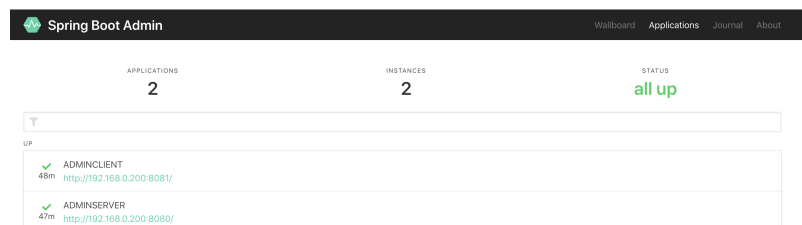
创建成功之后，在 `adminclient` 的 `application.properties` 中添加如下配置：

```
spring.application.name=adminclient
server.port=8081
eureka.client.service-url.defaultZone=http://localhost:1111/eureka/
management.endpoints.web.exposure.include=*
```

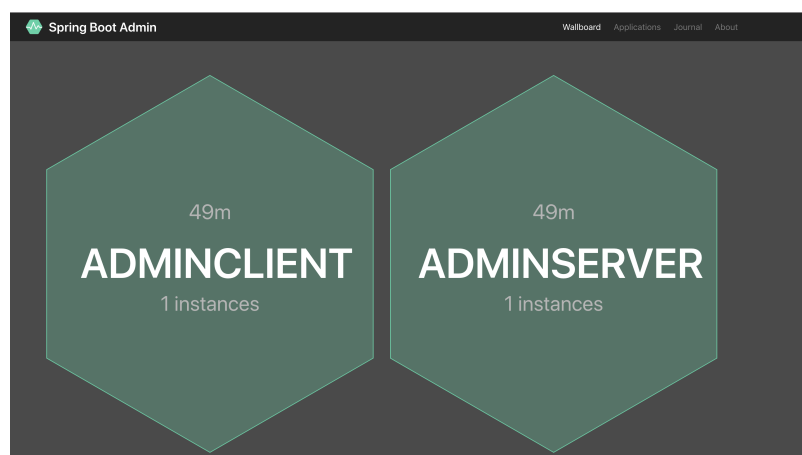
这里的配置含义上文都已经说过，不再多说。

这样我们的 `adminclient` 就算是配置完成了。很神奇吧？`adminclient` 中不需要任何和 `adminserver` 相关的配置，开发者只需要将 `adminclient` 注册到 `eureka` 上即可，剩下的事情由系统自动安排！

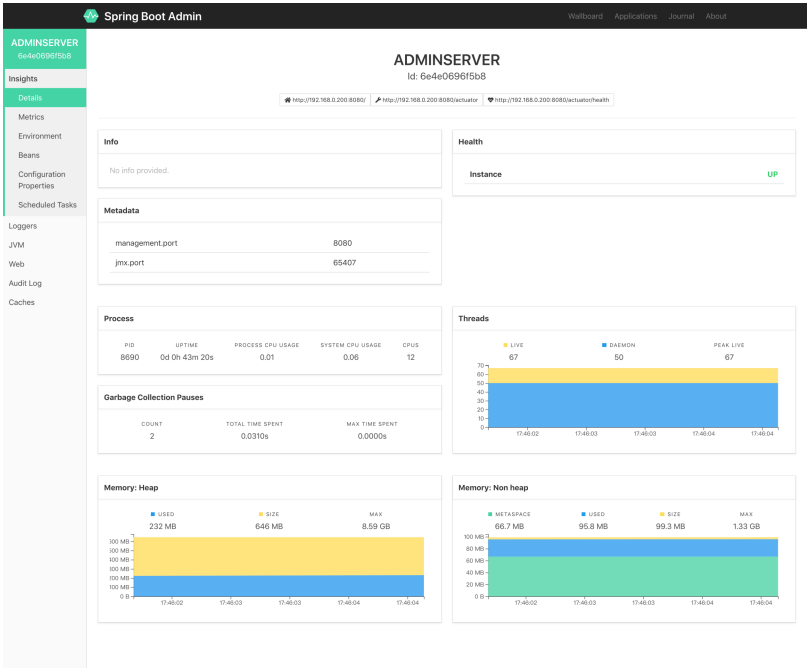
接下来启动 `adminclient`，启动成功之后，我们访问 `http://localhost:8080` 地址，就可以看到如下页面：



可以看到所有的服务都展示出来了，包括 `adminserver`，在 `Wallboard` 中一样可以看到全部数据：



其他的日志信息、详细运行数据也是一样的：



Event Journal			
Application	Instance	Time	Event
ADMINSERVER	6e4e0696f5b8	2019/05/29 16:53:59.065	REGISTRATION_UPDATED
ADMINSERVER	6e4e0696f5b8	2019/05/29 16:53:48.137	ENDPOINTS_DETECTED
ADMINSERVER	6e4e0696f5b8	2019/05/29 16:53:48.127	STATUS_CHANGED(UP)
ADMINCLIENT	e2cc1f483e5f	2019/05/29 16:53:29.385	ENDPOINTS_DETECTED
ADMINCLIENT	e2cc1f483e5f	2019/05/29 16:53:29.361	STATUS_CHANGED(UP)
ADMINSERVER	6e4e0696f5b8	2019/05/29 16:53:29.257	STATUS_CHANGED(OFFLINE)
ADMINSERVER	6e4e0696f5b8	2019/05/29 16:53:29.211	REGISTERED
ADMINCLIENT	e2cc1f483e5f	2019/05/29 16:53:29.085	REGISTERED

异常报警

服务上线之后，运维工程师不可能 24 小时盯着服务，查看运行数据，如果服务运行发生异常状况，最好有自动报警机制。类似于我们使用 Prometheus+Grafana 时采用的自动报警机制一样，Spring Boot Admin 中也提供了类似的功能。

首先我们要在 adminserver 中添加邮件发送依赖支持：

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

然后在 adminserver 的 application.properties 文件中配置邮件发送基本信息：

```
spring.mail.host=smtp.qq.com
spring.mail.port=465
spring.mail.username=xxx@qq.com
spring.mail.password=授权码
spring.mail.default-encoding=UTF-8
spring.mail.properties.mail.smtp.socketFactory.class=javax.net.ssl.SSLSocketFactory
spring.mail.properties.mail.debug=true
spring.boot.admin.notify.mail.from=xxx@qq.com
spring.boot.admin.notify.mail.to=xxx@qq.com
spring.boot.admin.notify.mail.cc=xxx@qq.com
spring.boot.admin.notify.mail.ignore-changes=
```


关于邮件发送配置读者可以参考 10-2 小节，这里多出来的配置是最后四行配置，分别表示：

- 报警邮件的发送者
- 报警邮件收件人
- 报警邮件抄送地址
- 忽略掉的事件

默认情况下，当被监控应用的状态变为 UNKNOWN 或者 UP 时不会发送报警邮件，而这里的配置表示被监控应用的任何变化都会发送报警邮件（即不忽略任何变化）。

配置完成后，重新启动 AdminServer，然后启动被监控应用，此时就会收到应用上线的邮件报警：

ADMINCLIENT (e2cc1f483e5f) is UP ☆

发件人：水流云在 <1510161612@qq.com> 

时 间：2019年5月29日(星期三) 晚上6:36

收件人：江南一点雨 <584991843@qq.com>

抄 送：1234 <1470249098@qq.com>

ADMINCLIENT (e2cc1f483e5f) is UP

Instance e2cc1f483e5f changed status from UNKNOWN to UP

Status Details

Registration

Service Url <http://192.168.0.200:8081/>

Health Url <http://192.168.0.200:8081/actuator/health>

Management Url <http://192.168.0.200:8081/actuator>

此时关闭被监控应用，就会收到应用下线的邮件报警：

ADMINCLIENT (e2cc1f483e5f) is OFFLINE ☆

发件人：水流云在 <1510161612@qq.com> 

时 间：2019年5月29日(星期三) 晚上6:37

收件人：江南一点雨 <584991843@qq.com>

抄 送：1234 <1470249098@qq.com>

ADMINCLIENT (e2cc1f483e5f) is OFFLINE

Instance e2cc1f483e5f changed status from UP to OFFLINE

Status Details

exception

io.netty.channel.AbstractChannel\$AnnotatedConnectException

message

Connection refused: /192.168.0.200:8081

Registration

Service Url <http://192.168.0.200:8081/>

Health Url <http://192.168.0.200:8081/actuator/health>

Management Url <http://192.168.0.200:8081/actuator>

小结

本文主要和大家分享了 Spring Boot Admin 服务化，这个过程其实很简单，Admin Server 和其他服务分别注册到服务注册中心就可以了，Admin Server 会自动找到各个微服务然后将监控数据展示出来。最后还和大家聊了邮件监控，当服务发生异常情况时，Admin Server 能够自动发送报警邮件，方便运维工程师及时获取到服务异常信息，进而及时处理。