Actuator介绍

- Spring Boot Actuator提供http (或JMX)端点来实现对应用程序的监视和管理、收集运行状况等功能。

引入spring-boot-starter-actuator可启用这些功能

- 默认情况下，通过访问/actuator可以看到所有启用的端点，也可以通过配置修改这个路径地址。

例如修改为: management.endpoints.web.base-path = /manage

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


端点配置

Spring Boot包含许多内置端点，允许添加自己的端点，可以配置端点是否对外开放或关闭。

① 设置默认关闭所有端点（默认开放启用了“health”和“info”）

management.endpoints.enabled-by-default = false

② 启动指定的端点

management.endpoint.info.enabled = true

③ 数据缓存

端点自动缓存对不带任何参数的读取操作的响应。要配置端点缓存响应的时间量

management.endpoint. <name>.cache.time-to-live = 10s

<name>需要使用对应的端点名称来代替。

Http端点配置

- Http配置

通过HTTP公开除了env和beans端点之外的所有内容

management.endpoints.web.exposure.include=*

management.endpoints.web.exposure.exclude=env,beans

- CORS跨域支持（默认情况下禁用CORS支持）

management.endpoints.web.cors.allowed-origins = http://example.com

management.endpoints.web.cors.allowed-methods = GET，POST

- 修改ManagerServer服务器配置

management.server.port = 8081 （如果设置为-1代表禁用http端点）

management.server.address = 127.0.0.1

可以配置和web服务使用不同的端口，同时绑定指定IP。（不同端口，代表启动多个tomcat容器）

端点讲解 - Health健康检查

访问/actuator/health 查看程序中组件检查项的运行状况信息，在出现故障时能及时发现。

Spring Boot默认提供了对Redis、RabbitMQ、DataSource、MongoDB等组件的检查项。

- 展示更详细内容(默认never)

management.endpoint.health.show-details=never (或者when-authorized | always)

- 返回结果

如果有检查项处于非检查状态， Http状态码为 503，返回值为 DOWN 或者 OUT_OF_SERVICE

如果没有检查初问题，返回Http状态码200，返回值UP或者UNKNOWN

- 自定义健康检查项

实现HealthIndicator接口，通过spring实例化一个对象，Spring Boot会自动触发健康检查，并归入health的结果。

端点讲解 - 日志配置

网易云课堂

- Spring Boot日志配置

  logging.level.root = WARN

  logging.level.org.springframework.web = DEBUG

  logging.level.org.hibernate = ERROR

- 通过 /actuator/loggers 查看日志配置

- 运行时修改配置

```
curl -X POST -H 'Content-Type: application/json' -i 'http://127.0.0.1:8081/manage/loggers/ROOT' --data '{
        "configuredLevel": "DEBUG"
}'
```


自定义端点

网易云课堂

可以理解为“概念上类似SpringMVC的controller写法，却又是完全不同的一套API。”

- 端点定义：@Endpoint 或@WebEndpoint或@JmxEndpoint
- 端点操作：@ReadOperation 、@WriteOperation、@DeleteOperation
- 参数接收：web环境下添加@Selector

```
@Endpoint(id = "myEndpoint")
@Component
public class MyEndPoint {

  String name = "default";

  @ReadOperation
  public String getName() {
    // spring http端点的json格式
    return "{\"name\":\""+name+"\"}";
  }
}
```

快速理解JMX机制

Java Management Extensions（JMX）提供了一种监视和管理应用程序的标准机制。
tomcat、kafka、druid都是用的JMX技术来实现对外暴露管理接口和监控信息。

- 如何Jconsole工具通过JMX技术实现对应用的监控和管理？

- 通过Spring框架快速增加自定义Mbean
默认情况下，Spring Boot将管理端点公开为org.springframework.boot域下的JMX MBean 。

- 通过JMX公开所有端点并仅显示端点health和 info端点
  management.endpoints.jmx.exposure.exclude=*
  management.endpoints.jmx.exposure.include=info, health