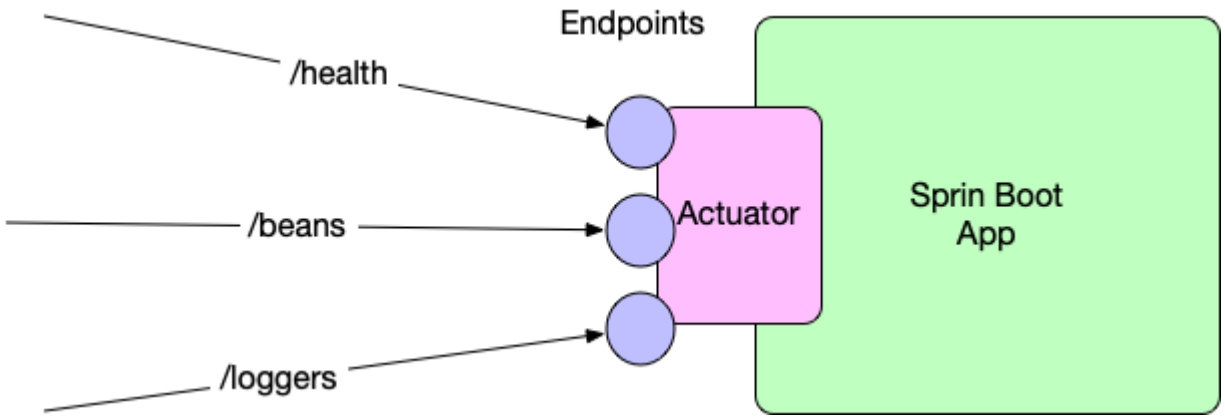
Tabla de Contenidos

- Spring Boot Actuator EndPoints
- Manejo de Beans
- Otros artículos relacionados

Spring Boot Actuator es un starter de Spring Boot que nos permite obtener información sobre el estado de todos los elementos que componen la aplicación arrancada con Spring Boot . Desde los beans registrados pasando por los loggers , trazas http etc.



Configurar Spring Boot Actuator es muy sencillo ya que es suficiente con añadir a nivel de pom.xml el starter de actuator.

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

                </dependency>
```

Una vez tenemos dado de alta el actuator es suficiente con volver a arrancar la aplicación de Spring Boot y podremos acceder a varias urls que nos aportan información sobre la aplicación y su estado. En un primer momento la única url activa es actuator/health y nos devuelve la información sobre si la aplicación esta activa o no.

```
{"status":"UP"}
```

Spring Boot Actuator EndPoints

Ahora bien nosotros podemos activar o desactivar las urls que mejor nos convengan para la obtención de información de la aplicación las urls más típicas son:

- /beans : Nos devuelve una estructura JSON con todos los beans de la aplicación
- /env : Nos devuelve información sobre las propiedades del proyecto de Spring Boot
- /health: Nos devuelve información sobre el estado de la aplicación como ya hemos visto
- /loggers: Nos devuelve información sobre los loggers configurados y nos permite realizar peticiones POST para cambiar los niveles.
- /metrics: Nos devuelve información sobre las métricas de la aplicación
  Para activar por defecto todos los endpoints de actuator sería suficiente con habilitar en el application.properties la opción:

```
management.endpoints.web.exposure.include=*
```

Una vez hecho esto podemos solicitar la url por defecto de Spring Boot Actuator y acceder a todos los endpoints disponibles(ojo que esto nunca se publica por completo es un ejemplo).

```
{"_links":{"self":{"href":"http://localhost:8081/actuator","templated"
:false},"misdatos":{"href":"http://localhost:8081/actuator/misdatos","
templated":false},"beans":{"href":"http://localhost:8081/actuator/bean
s","templated":false},"caches":{"href":"http://localhost:8081/actuator
/caches","templated":false},"caches-
cache":{"href":"http://localhost:8081/actuator/caches/{cache}","templa
ted":true},"health-
path":{"href":"http://localhost:8081/actuator/health/{*path}","templat
ed":true},"health":{"href":"http://localhost:8081/actuator/health","te
mplated":false},"info":{"href":"http://localhost:8081/actuator/info","
templated":false},"conditions":{"href":"http://localhost:8081/actuator
```

/conditions","templated":false},"configprops-prefix":{"href":"http://localhost:8081/actuator/configprops/{prefix}","templated":true},"configprops":{"href":"http://localhost:8081/actuator/configprops","templated":false},"env-toMatch":{"href":"http://localhost:8081/actuator/env/{toMatch}","templated":true},"env":{"href":"http://localhost:8081/actuator/env","templated":false},"loggers-name":{"href":"http://localhost:8081/actuator/loggers/{name}","templated":true},"loggers":{"href":"http://localhost:8081/actuator/loggers","templated":false},"heapdump":{"href":"http://localhost:8081/actuator/heapdump","templated":false},"threaddump":{"href":"http://localhost:8081/actuator/threaddump","templated":false},"metrics-requiredMetricName":{"href":"http://localhost:8081/actuator/metrics/{requiredMetricName}","templated":true},"metrics":{"href":"http://localhost:8081/actuator/metrics","templated":false},"scheduledtasks":{"href":"http://localhost:8081/actuator/scheduledtasks","templated":false},"mappings":{"href":"http://localhost:8081/actuator/mappings","templated":false}}}

Manejo de Beans

Una vez que tenemos todas las urls disponibles podemos por ejemplo solicitar la de /beans para que nos clarifique que beans tenemos a nuestra disposición dentro de la app. Hay que andarse con ojo ya que la respuesta puede ser inmensa aquí mostramos un pequeño subconjunto que muestra el típico bean de repositorio:

```
"personaRepository": {
          "aliases": [
          ],
          "scope": "singleton",
          "type": "com.arquitecturajava.web1.PersonaRepository",
          "resource": "com.arquitecturajava.web1.PersonaRepository
```

```
defined in @EnableJpaRepositories declared on
JpaRepositoriesRegistrar.EnableJpaRepositoriesConfiguration",
          "dependencies": [
            "jpa.named-queries#0",
            "jpa.PersonaRepository.fragments#0",
            "jpaSharedEM#0",
            "jpaMappingContext"
          ]
        },
```

El uso de Spring Boot Actuator cada día es más importante ya que en estructuras de contenedores Docker o similar permite una gestión muy a detalle de cada aplicación así como la obtención de métricas que nos sean de utilidad. Eso sí no debemos activar todos los accesos de forma directa sino que debemos activar aquellas cosas que nos sean de utilidad para ello podríamos usar:

```
management.endpoints.web.exposure.include=health,beans,loggers
```

Otros artículos relacionados

- Spring Boot JPA y su configuración
- Spring Boot y Spring Security Custom Login
- Spring Boot Kotlin y la reducción de código
- Spring Boot Visual Studio Code y extensiones
- Spring boot