Spring Actuator: Features and Best Practices

Introduction

Spring Boot Actuator is a sub-project of Spring Boot that adds several production-ready features to your application. It provides built-in endpoints and tools for monitoring and managing your Spring Boot application.

We can use **HTTP** and **JMX** endpoints to manage and monitor the Spring Boot application.

Spring Boot Actuator Features

There are **three** main features of Spring Boot Actuator:

- **Endpoints**
- **Metrics**
- **Audit**

Metrics: Spring Boot Actuator provides dimensional metrics by integrating with the **micrometer**.

Audit: Spring Boot provides a flexible audit framework that publishes events to an **AuditEventRepository.** It automatically publishes the authentication events if spring-security is in execution.

Spring Boot Actuator Endpoints

The actuator endpoints allow us to monitor and interact with our Spring Boot application. Spring Boot includes number of built-in endpoints and we can also add custom endpoints in Spring Boot application.

The following table describes the widely used endpoints.

Id	Usage	Default
actuator	It provides a hypermedia-based **discovery page** for the other endpoints. It requires Spring HATEOAS to be on the classpath.	True
auditevents	It exposes audit events information for the current application.	True
autoconfig	It is used to display an auto-configuration report showing all auto-configuration candidates and the reason why they 'were' or 'were not' applied.	True

beans	It is used to display a complete list of all the Spring beans in your application.	True
configprops	It is used to display a collated list of all @ConfigurationProperties.	True
dump	It is used to perform a thread dump.	True
env	It is used to expose properties from Spring's ConfigurableEnvironment.	True
flyway	It is used to show any Flyway database migrations that have been applied.	True
health	It is used to show application health information.	False
info	It is used to display arbitrary application info.	False
loggers	It is used to show and modify the configuration of loggers in the application.	True
liquibase	It is used to show any Liquibase database migrations that have been applied.	True
metrics	It is used to show metrics information for the current application.	True
mappings	It is used to display a collated list of all @RequestMapping paths.	True
shutdown	It is used to allow the application to be gracefully shutdown.	True
trace	It is used to display trace information.	True

Setup

To add Spring Actuator to your project, include the following dependency in your `pom.xml`:

xml

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

Configure Actuator in your `application.properties` or `application.yml`:

properties

Enable all Actuator endpoints

management.endpoints.web.exposure.include=*

Customize base path for actuator endpoints (optional)

management.endpoints.web.base-path=/management

Key Features and Code Examples

1. **Health Checks**

- Provides information about application health

- Customizable for specific needs

Example of a custom health indicator:

```java
import org.springframework.boot.actuate.health.Health;
import org.springframework.boot.actuate.health.HealthIndicator;
import org.springframework.stereotype.Component;

@Component
public class CustomHealthIndicator implements HealthIndicator {

@Override
public Health health() {
int errorCode = check(); // perform some specific health check
if (errorCode != 0) {
return Health.down().withDetail("Error Code", errorCode).build();
}
return Health.up().build();
}

private int check() {
// Your logic to check health
return 0;
}
}
```

2. Metrics

- Exposes various metrics about the application
- Integrates with monitoring systems like Prometheus

Example of creating a custom metric:

```java
import io.micrometer.core.instrument.Counter;
import io.micrometer.core.instrument.MeterRegistry;
import org.springframework.stereotype.Service;

@Service
public class CustomMetricService {
private final Counter customCounter;
public CustomMetricService(MeterRegistry meterRegistry) {
this.customCounter = Counter.builder("custom.metric")
.description("A custom metric")
.register(meterRegistry);
}

public void incrementCustomMetric() {
this.customCounter.increment();
}
}
```

Info Endpoint

- Displays arbitrary application information

Example of customizing the info endpoint:

```
import org.springframework.boot.actuate.info.Info;
import org.springframework.boot.actuate.info.InfoContributor;
import org.springframework.stereotype.Component;

@Component
public class CustomInfoContributor implements InfoContributor {
@Override
public void contribute(Info.Builder builder) {
builder.withDetail("example", "This is custom info")
.withDetail("version", "1.0.0");
}
}
```

Best Practices

1. Security

- Secure actuator endpoints, especially in production

- Use Spring Security to control access

Example of securing Actuator endpoints:

java

```
import org.springframework.boot.actuate.autoconfigure.security.servlet.EndpointRequest;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import
org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapt
er;


@Configuration

public class ActuatorSecurityConfig extends WebSecurityConfigurerAdapter {

@Override

protected void configure(HttpSecurity http) throws Exception {

http.requestMatcher(EndpointRequest.toAnyEndpoint())

.authorizeRequests()

.anyRequest().hasRole("ACTUATOR")

.and()

.httpBasic();

}

}
```

Customization

- Customize endpoints to expose only necessary information
- Create custom health indicators for application-specific health checks

Metrics

- Use micrometer for consistent metrics collection
- Choose appropriate metrics for your application's needs

Monitoring

- Integrate with monitoring tools (e.g., Prometheus, Grafana)
- Set up alerts for critical metrics

Documentation

- Document custom endpoints and metrics
- Provide clear guidelines for operations team

Performance

- Be mindful of the performance impact of enabled endpoints
- Use sampling for high-volume metrics

Versioning

- Include application version in the info endpoint
- Use git commit information for precise versioning

Healthchecks

- Implement meaningful health checks for external dependencies
- Use the health endpoint for load balancer checks

Sample Output After Visiting (http://localhost:8080/actuator)

{

"_links": {

"self": {

"href": "http://localhost:8080/actuator",

"templated": false

},

"beans": {

"href": "http://localhost:8080/actuator/beans",

"templated": false

},

"caches-cache": {

"href": "http://localhost:8080/actuator/caches/{cache}",

"templated": true

},

"caches": {

"href": "http://localhost:8080/actuator/caches",

"templated": false

},

"health": {

"href": "http://localhost:8080/actuator/health",

"templated": false

},

"health-path": {

"href": "http://localhost:8080/actuator/health/{*path}",

"templated": true

},

"info": {

"href": "http://localhost:8080/actuator/info",

"templated": false

},

"conditions": {

"href": "http://localhost:8080/actuator/conditions",
"templated": false
},
"configprops": {
"href": "http://localhost:8080/actuator/configprops",
"templated": false
},
"configprops-prefix": {
"href": "http://localhost:8080/actuator/configprops/{prefix}",
"templated": true
},
"env": {
"href": "http://localhost:8080/actuator/env",
"templated": false
},
"env-toMatch": {
"href": "http://localhost:8080/actuator/env/{toMatch}",
"templated": true
},
"loggers": {
"href": "http://localhost:8080/actuator/loggers",
"templated": false
},
"loggers-name": {
"href": "http://localhost:8080/actuator/loggers/{name}",
"templated": true
},
"heapdump": {
"href": "http://localhost:8080/actuator/heapdump",
"templated": false
},
"threaddump": {

"href": "http://localhost:8080/actuator/threaddump",
"templated": false
},
"metrics-requiredMetricName": {
"href": "http://localhost:8080/actuator/metrics/{requiredMetricName}",
"templated": true
},
"metrics": {
"href": "http://localhost:8080/actuator/metrics",
"templated": false
},
"sbom-id": {
"href": "http://localhost:8080/actuator/sbom/{id}",
"templated": true
},
"sbom": {
"href": "http://localhost:8080/actuator/sbom",
"templated": false
},
"scheduledtasks": {
"href": "http://localhost:8080/actuator/scheduledtasks",
"templated": false
},
"mappings": {
"href": "http://localhost:8080/actuator/mappings",
"templated": false
}
}
}