Spring Boot Actuator

Introduction: Spring Boot Actuator is a module that provides production-ready features to monitor and manage your Spring Boot application. It offers various endpoints and metrics that can be used for monitoring, health checks, auditing, and managing your application. The actuator exposes these features via RESTful endpoints or JMX (Java Management Extensions) beans.

To enable Spring Boot Actuator in the application, we need to add the `spring-boot-starter-actuator` dependency to your project. Here's an example of how we can enable Actuator in a Spring Boot application:

- Add the Actuator dependency to `pom.xml` (for Maven) or `build.gradle` (for Gradle) file:

Maven:

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


</dependencies>
```

Gradle:

```
dependencies {

    // Other dependencies

    implementation 'org.springframework.boot:spring-boot-starter-actuator'


}
```

2. By default, Actuator endpoints are enabled, but we can customise the configuration in the `application.properties` or `application.yml` file. For example, to enable all Actuator endpoints, we can add the following line to `application.properties file`:

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

Now that we have Actuator enabled, we can access various endpoints to monitor and manage the application. Here are a few commonly used Actuator endpoints:

- /actuator/health: Provides information about the health of the application. It indicates whether the application is up and running or experiencing any issues.
- /actuator/info: Returns custom application information that we can configure. This endpoint can be useful to expose additional metadata about the application.
- /actuator/metrics: This shows various metrics about the application, such as JVM memory usage, HTTP request statistics, and many others. We can view the available metrics by accessing `/actuator/metrics` or retrieving specific metrics by appending the metric name (e.g., `/actuator/metrics/jvm.memory.used`).
- /actuator/beans: Displays a list of all Spring beans in the application.
- /actuator/mappings: This shows a list of all request mappings in the application, including the URL paths, HTTP methods, and the corresponding controller methods.

These are just a few examples of the Actuator endpoints available. We can find a complete list of available endpoints and their descriptions in the official Spring Boot Actuator documentation.

Link to Spring Boot Actuator Api Official Documentation: https://docs.spring.io/spring-boot/docs/current/actuator-api/htmlsingle/

In addition to the default endpoints, Actuator also allows us to customise and create our own custom endpoints. We can define custom endpoints by creating Spring beans that implement the `org.springframework.boot.actuate.endpoint.web.servlet.AbstractWebMvcEndpointHandlerMapping` or `org.springframework.boot.actuate.endpoint.jmx.AbstractEndpointMBean` interfaces.

By using Spring Boot Actuator, we can easily monitor and manage our Spring Boot application in a production environment, gaining insights into its health, metrics, and other important information.

Summary

Spring Boot Actuator is a module that provides monitoring and management capabilities for Spring Boot applications through various endpoints and metrics, accessible via REST or JMX.

Abstract

Spring Boot Actuator is an essential tool for ensuring that Spring Boot applications are production-ready by offering insights into application health, metrics, and management. By adding the `spring-boot-starter-actuator` dependency to the project's build configuration, developers can enable a suite of predefined endpoints such as `/actuator/health`, `/actuator/info`, and `/actuator/metrics`. These endpoints can be customized and configured through properties files to expose the necessary information for monitoring and managing the application effectively. The Actuator module simplifies the process of understanding the application's performance and operational status, making it an indispensable component for Spring Boot applications in a production environment.

My Opinions

- The Actuator module is highly recommended for its ability to provide critical insights into a Spring Boot application's operation, which is crucial for production environments.
- Customization of Actuator endpoints is a valuable feature, allowing developers to tailor the exposed information to their specific needs.
- The ability to access Actuator features via both RESTful endpoints and JMX beans offers flexibility in how developers and system administrators can interact with and manage the application.
- The article suggests that using Spring Boot Actuator is a cost-effective solution for application monitoring when compared to other AI services, highlighting a special offer for ZAI.chat as a more affordable alternative to ChatGPT Plus (GPT-4).