Spring Actuator Overview

What is Spring Actuator?

Spring Actuator is a sub-project of Spring Boot that provides production-ready features to help you monitor and manage your application. It exposes various endpoints that provide information about the application's health, metrics, environment, and more.

Key Features of Spring Actuator

1. **Health Checks**
    - **Health Endpoint**: Provides information about the application's health status. Includes checks for database connectivity, disk space, and custom checks.
    - **Custom Health Indicators**: Allows you to define custom health checks to monitor specific parts of your application.
2. **Metrics Collection**
    - **Metrics Endpoint**: Exposes application metrics such as JVM metrics, HTTP requests, and custom metrics.
    - **Custom Metrics**: Allows you to create and expose custom metrics that are relevant to your application.
3. **Application Info**
    - **Info Endpoint**: Provides metadata about the application such as version, build information, and custom application details.
    - **Custom Info Contributors**: Enables adding custom information to the info endpoint.
4. **Environment Properties**
    - **Environment Endpoint**: Exposes environment properties and configuration settings used in the application.
5. **Loggers**
    - **Loggers Endpoint**: Provides access to change log levels at runtime. Useful for debugging and troubleshooting.
6. **Thread Dump**
    - **Thread Dump Endpoint**: Provides a snapshot of the thread dump, useful for diagnosing performance issues.
7. **Custom Endpoints**
    - **Custom Endpoints**: Allows you to define and expose your own endpoints to provide additional application-specific information.

Best Practices

1. **Secure Endpoints**: Ensure sensitive endpoints (e.g., `/env`, `/heapdump`) are properly secured to prevent unauthorized access.

2. **Custom Metrics**: Use custom metrics to gain insights into application performance and behavior.
3. **Monitor Actuator Endpoints**: Regularly monitor Actuator endpoints to ensure application health and performance.
4. **Use Management Endpoints in Production**: Enable and use Actuator endpoints in production environments to track and manage application health and metrics.
5. **Document Custom Endpoints**: Clearly document any custom endpoints created to ensure they are easily understood and maintained.
6. **Consider Privacy and Security**: Be cautious about what information is exposed via endpoints to avoid leaking sensitive information.

Example Use Cases

1. **Monitoring Application Health**: Use the `/actuator/health` endpoint to monitor the overall health of your application and configure alerts based on the health status.
2. **Collecting Metrics**: Expose metrics via `/actuator/metrics` to integrate with monitoring tools and visualize application performance over time.
3. **Custom Endpoints**: Create custom endpoints to provide specific operational metrics or status information relevant to your application's needs.

Spring Actuator is a powerful tool for monitoring and managing Spring Boot applications. By understanding its features and applying best practices, you can enhance the observability and maintainability of your applications.