**SPRING BOOT ACTUATOR**

Spring Boot Actuator module helps you monitor and manage your Spring Boot application by providing production-ready features like health check-up, auditing, metrics gathering, HTTP tracing etc. All of these features can be accessed over JMX or HTTP endpoints.

Actuator also integrates with external application monitoring systems like [Prometheus](https://prometheus.io/), [Graphite](https://graphiteapp.org/), [DataDog](https://www.datadoghq.com/), [Influx](https://www.influxdata.com/), [Wavefront](https://www.wavefront.com/), [New Relic](https://newrelic.com/) and many more. These systems provide you with excellent dashboards, graphs, analytics, and alarms to help you monitor and manage your application from one unified interface.

Actuator uses [Micrometer](http://micrometer.io/), an application metrics facade to integrate with these external application monitoring systems. This makes it super easy to plug-in any application monitoring system with very little configuration.

**Note that, every actuator endpoint can be explicitly enabled and disabled. Moreover, the endpoints also need to be exposed over HTTP or JMX to make them remotely accessible.**

<http://localhost:8080/actuator/health>

The info endpoint (<http://localhost:8080/actuator/info>) displays general information about your application obtained from build files like META-INF/build-info.properties or Git files like git.properties or through any environment property under the key info. You’ll learn how to tweak the output of this endpoint in the next section.

| **Endpoint ID** | **Description** |
| --- | --- |
| auditevents | Exposes audit events (e.g. auth\_success, order\_failed) for your application |
| info | Displays information about your application. |
| health | Displays your application’s health status. |
| metrics | Shows various metrics information of your application. |
| loggers | Displays and modifies the configured loggers. |
| logfile | Returns the contents of the log file (if logging.file or logging.path properties are set.) |
| httptrace | Displays HTTP trace info for the last 100 HTTP request/response. |
| env | Displays current environment properties. |
| flyway | Shows details of Flyway database migrations. |
| liquidbase | Shows details of Liquibase database migrations. |
| shutdown | Lets you shut down the application gracefully. |
| mappings | Displays a list of all @RequestMapping paths. |
| scheduledtasks | Displays the scheduled tasks in your application. |
| threaddump | Performs a thread dump. |
| heapdump | Returns a GZip compressed JVM heap dump. |

By default, all the endpoints that I listed in the previous section are enabled except the shutdown endpoint.

You can enable or disable an actuator endpoint by setting the property management.endpoint.<id>.enabled to true or false (where id is the identifier for the endpoint).

For example, to enable the shutdown endpoint, add the following to your application.properties file –

By default, all the actuator endpoints are exposed over JMX but only the health and info endpoints are exposed over HTTP



expose all actuator endpoints by setting the property management.endpoints.web.exposure.include to \* and check the output of <http://localhost:8080/actuator> page. You’ll notice that the actuator page now lists all the enabled endpoints-