



[REST-API]



Monitoramento

Expondo dados de monitoramento

Um dos **pilares primordiais de uma API**, é o monitoramento. Poder conferir se **ela está offline**, ou se **tem algum problema**, quanto ela consome de **RAM**, de **espaço de armazenamento**, pontos de **gargalo** e etc.

Saber desses detalhes é o que torna a aplicação altamente manutenível e funcional, entrando bastante no padrão de Observability. O **Spring** oferece um módulo para esse tipo de situação chamado **"Spring boot actuator"**. Para adiciona-lo o projeto é só **adicionar o módulo ao pom.xml**, a partir disso será disponibilizado um **novo endpoint** na aplicação: **"/actuator"**, quando **acessado** devolve uma **lista de sub endpoints** que contém **informações de saúde e status da API**:

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

```
GET http://localhost:8080/actuator
200 OK 149 ms 617 B
{
  "_links": {
    "self": {
      "href": "http://localhost:8080/actuator",
      "templated": false
    },
    "health-path": {
      "href": "http://localhost:8080/actuator/health/{*path}",
      "templated": true
    },
    "health": {
      "href": "http://localhost:8080/actuator/health",
      "templated": false
    }
  }
}
```

Você pode definir quais informações serão expostas no **arquivo de properties**, habilitando assim **diversos endpoints** com centenas de informações:

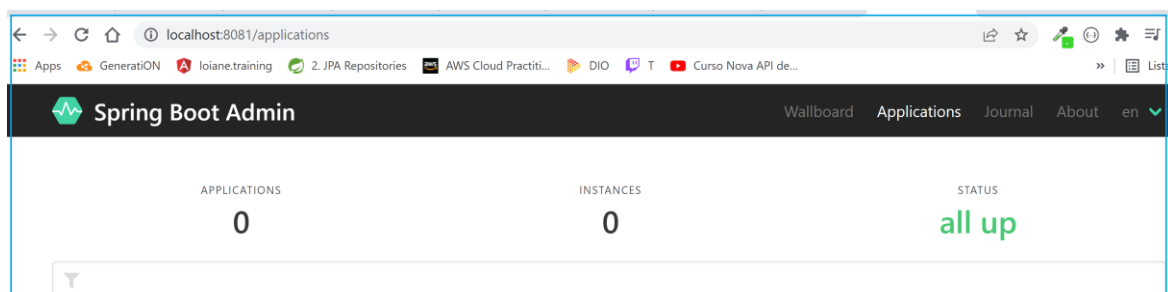
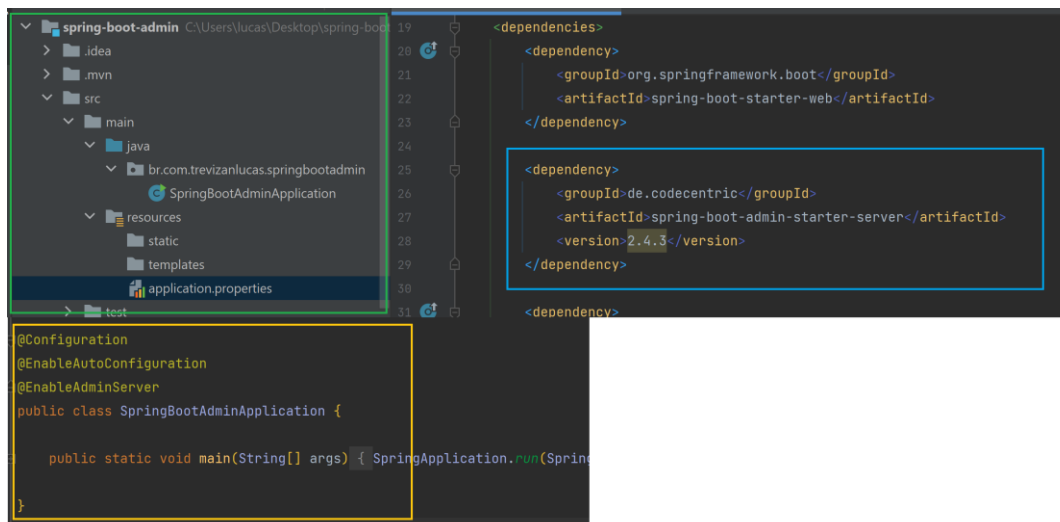
```
# actuator
management.info.env.enabled=true
management.endpoint.health.show-details=always
management.endpoints.web.exposure.include=*
info.app.name=@project.name@
info.app.description=@project.description@
info.app.version=@project.version@
info.app.encoding=@project.build.sourceEncoding@
info.app.java.version=@java.version@
```

```
{
  "_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
    }
  }
}
```

Usando Interface Gráfica

Se você **tiver que monitorar por conta própria**, ou não tiver uma ferramenta específica, temos a opção de **utilizar uma ferramenta WEB** que tem **interface Gráfica** e se **integra ao Spring boot**. Essa ferramenta é o **“Spring Boot Admin”** (e ela não foi desenvolvida pela equipe do Spring).

Para **utiliza-lo** é necessário criar um **OUTRO PROJETO Spring Boot** com a **dependência WEB**, adicionar ao **pom.xml a dependência do Spring boot admin** e colocar **algumas anotações no main**:



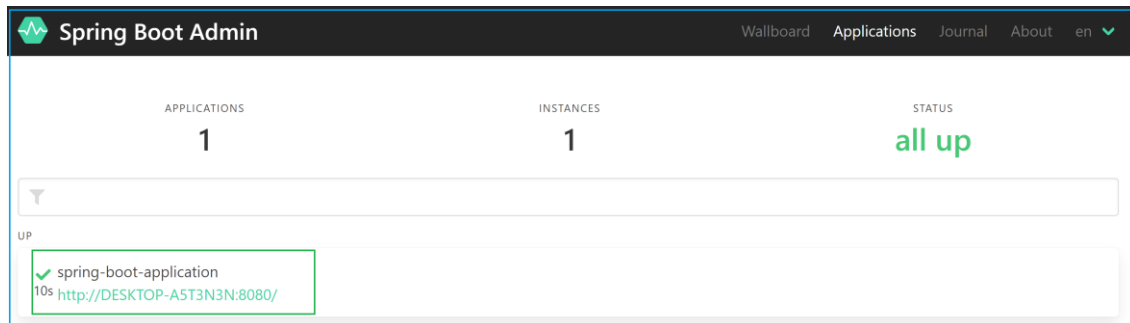
Com isso o **Spring boot Admin** já está funcional e pronto pra uso, mas temos que vincular qual projeto queremos exibir.

Vinculando projeto ao Spring Boot Admin

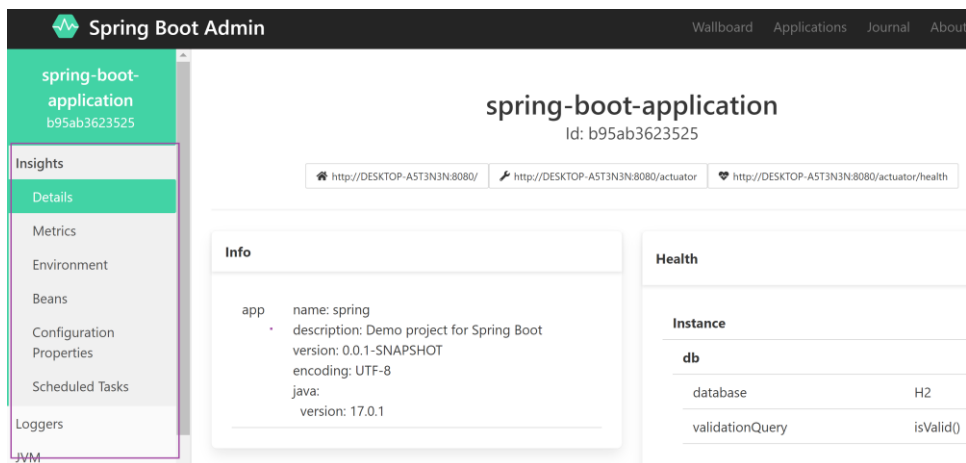
Basta adicionar a **dependência de cliente na API** que quer monitorar, e configurar **via properties** qual **endereço ela vai apontar** (no caso o endereço onde está o Spring boot admin):



Com isso o Spring Boot admin reconhece a aplicação, sendo possível acompanhar pela interface gráfica todos os Status da API. Lembrando que ESSAS INFORMAÇÕES DEVEM SER PROTEGIDAS:



The image shows the Spring Boot Admin dashboard overview. At the top, there's a navigation bar with 'Spring Boot Admin' and links for 'Wallboard', 'Applications', 'Journal', 'About', and 'en'. Below the navigation bar, there are three main sections: 'APPLICATIONS' with a count of '1', 'INSTANCES' with a count of '1', and 'STATUS' showing 'all up' in green. A search bar is located below these sections. Under the 'UP' status, a list of applications is shown, with one application 'spring-boot-application' (ID: b95ab3623525) highlighted in green, indicating it is up and running. The URL 'http://DESKTOP-AST3N3N:8080/' is also visible.



The image shows the detailed view of the 'spring-boot-application' in Spring Boot Admin. The application ID is 'b95ab3623525'. The main title is 'spring-boot-application' with the ID 'Id: b95ab3623525'. Below the title, there are three links: 'http://DESKTOP-AST3N3N:8080/' (home), 'http://DESKTOP-AST3N3N:8080/actuator' (actuator), and 'http://DESKTOP-AST3N3N:8080/actuator/health' (health). The left sidebar contains a menu with 'Insights', 'Details', 'Metrics', 'Environment', 'Beans', 'Configuration', 'Properties', 'Scheduled Tasks', and 'Loggers'. The 'Details' section is currently selected. The main content area is divided into two panels: 'Info' and 'Health'. The 'Info' panel shows application details: name: spring, description: Demo project for Spring Boot, version: 0.0.1-SNAPSHOT, encoding: UTF-8, java: version: 17.0.1. The 'Health' panel shows the 'Instance' section with a table for 'db' (database) and 'validationQuery' (isValid()).

db	
database	H2
validationQuery	isValid()