Actuators

136

Actuators

* Spring Boot provides actuators to monitor and manage your application.
* Actuator is a tool which has RESTful HTTP endpoints. when application is pushed to production, you can choose to manage and monitor your application using these HTTP endpoints.
* To get production-ready features, we should use spring-boot-actuator module.

137

Actuators

* Dependencies :

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

138

Sample Actuator Endpoints

ID	Description	Enabled by default
auditevents	Exposes audit events information for the current application.	Yes
beans	Displays a complete list of all the Spring beans	Yes
conditions	Shows the conditions that were evaluated	Yes
configprops	Displays a collated list of all @Configuration Properties.	Yes
env	Exposes properties from Spring's Configurable Environment.	Yes
flyway	Shows any Flyway database migrations that have been applied.(Flyway is for DB version control)	Yes
health	Shows application health information.	Yes
httptrace	Displays HTTP trace information (by default, the last 100 HTTP request-response exchanges).	Yes

139

Swagger

140

Introduction

* Swagger (now the “Open API Initiative”) is a specification and framework for describing REST APIs using a common language that everyone can understand. There are other available frameworks that have gained some popularity, such as RAML, Summation etc. but Swagger is most popular at this point of time considering its features and acceptance among the developer community.

* It offers both human readable and machine readable format of documentation. It provides both JSON and UI support. JSON can be used as machine readable format and Swagger-UI is for visual display which is easy for humans to understand by just browsing the api documentation.

141

Spring Boot-Swagger2

* Dependencies :

```
<dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger2</artifactId>
    <version>2.6.1</version>
  </dependency>

  <dependency>
    <groupId>io.springfox</groupId>
    <artifactId>springfox-swagger-ui</artifactId>
    <version>2.6.1</version>
  </dependency>
```

142

Enable Swagger2

```
@Configuration
@EnableSwagger2
public class SwaggerConfig {
        private ApiInfo metaData() {
        ApiInfo apiInfo = new ApiInfo("Spring Boot REST API", "Spring Boot REST API for
EmployeeCRUDService", "1.0", "Terms of service", new Contact("Ameya Joshi",
"https://ameyajoshi.com/about/", "ameya@ameyajoshi.com"), "Apache License Version 2.0",
"https://www.apache.org/licenses/LICENSE-2.0");
    return apiInfo;
 }
// Only select apis that matches the given Predicates.
private Predicate<String> paths() {   // Match all paths except /error
        return Predicates.and (PathSelectors.regex("/employees.*"),
        Predicates.not(PathSelectors.regex("/error.*")));
    }
 @Bean
 public Docket employeeCRUDApi() {
  return new Docket(DocumentationType.SWAGGER_2) .select()
.apis(RequestHandlerSelectors.basePackage("com.ameya.controllers"))  .paths(paths())  .build()
.apiInfo(metaData());
  }
 }
```

143

Swagger 2 Annotations

```
@Api(value="EmployeeCRUDService“)
@ApiOperation(value = "View a list of Employees", response = Iterable.class)
@ApiResponses(value = {
        @ApiResponse(code = 200, message = "Successfully retrieved list"),
        @ApiResponse(code = 401, message = "You are not authorized to view the resource"),
        @ApiResponse(code = 403, message = "Accessing the resource you were trying to reach is forbidden"),
        @ApiResponse(code = 404, message = "The resource you were trying to reach is not found") } )
@ApiModelProperty(notes = "The Primary Key for Employee <b>empId</b>",required = true)
```

The annotations are used on top of classes/methods /attributes and swagger creates the docs respectively.

144