Working with Actuators:-

Actuators are used to provide production ready features

->To work with actuators, we should add below dependency

Actuator-starter

End-points:

1)heath

2)info

3)beans

4)mappings

5) configProps

6)env

7)threaddump :- It will provides threads details

8)headdump :- It will provide JVM heap details

To analyze heap data we will use MAT (Memory Analyzer Tool)

9)httptrace

10)shutdown: It is a special end point by using this we can stop our application.

But it is binded ‘POST’ request method default. if any unknow person checks in Browser(GET) app will stopped .

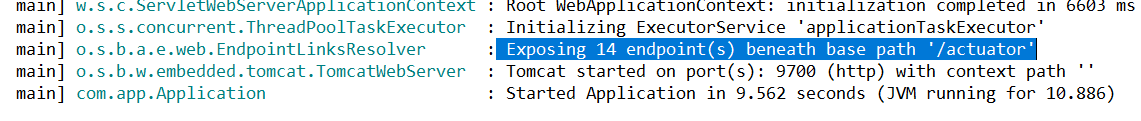
By default shutdown endpoint is disabled mode, to use this we have to enable in application.properties file

management:

endpoint:

shutdown:

enabled: **true**

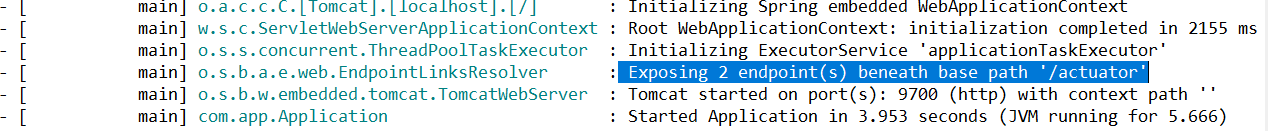


Steps to fallow:

When add dependency in pom.xml file by default 2 end point will come

<http://localhost:9700/actuator/health>

<http://localhost:9700/actuator/info>



If we add below property in application.yml, it will come 13 end points

management:

endpoints:

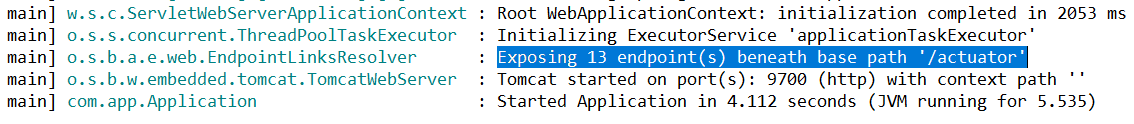
web:

exposure:

include: '\*'

server:

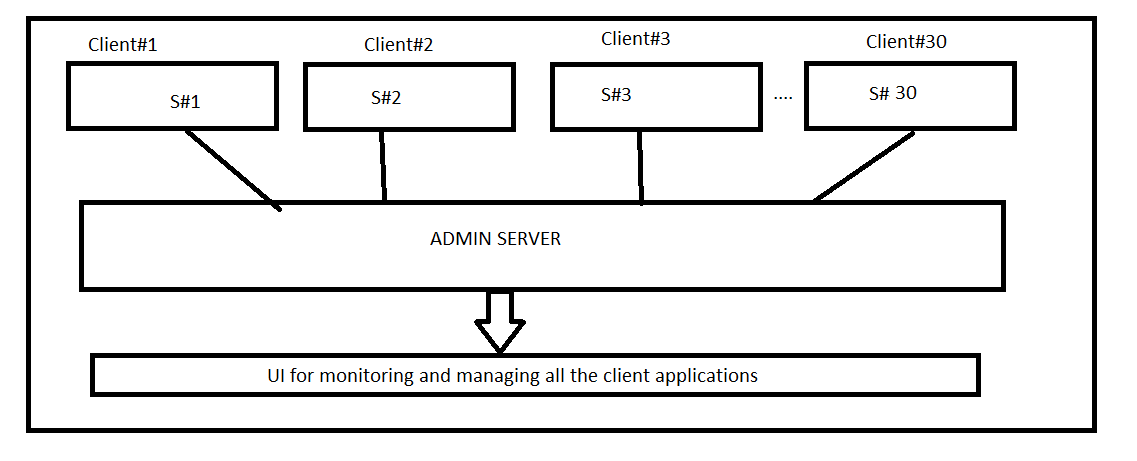
port: 9700



Spring Boot Admin server & Client: -

In Microservices architecture based projecs we will have multiple services

* To Monitor each service we will configure Actuator endpoints in Each Service
* If we have so many services then it should be very difficult for us to manage and monitor all those servers.
* To avoid that problem spring boot introduced Admin & Client concept.
* Boot admin server will provide Nice UI to monitor and manage all services at one place.



Steps to develop Admin Server Project: -

1)Create Spring boot separate application with below dependencies

-> Spring boot starter Web

-> Admin server

<dependency>

<groupId>de.codecentric</groupId>

<artifactId>spring-boot-admin-starter-server</artifactId>

</dependency>

2) Configure @EnableAdminServer annotation at Spring Boot start class

@SpringBootApplication

@EnableAdminServer

**public** **class** SbActuatorsAdminServerApplication {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(SbActuatorsAdminServerApplication.**class**, args);

}

}

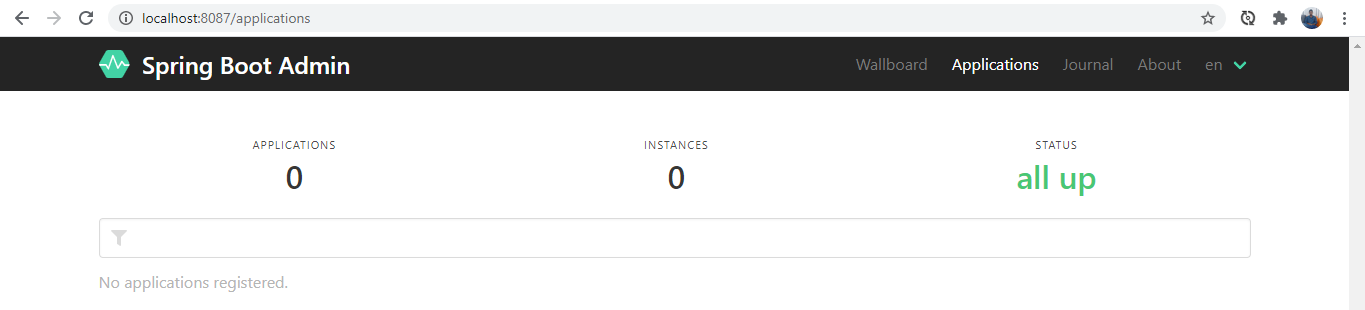
3) Configure port number and run application

server.port=8087

Browser URl: -

----------------------

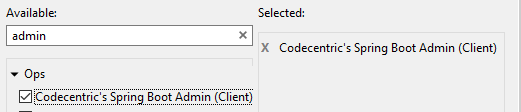
<http://localhost:8087/>



Steps to develop Admin Client Project: -

Step 1:

Create spring boot applications with below dependencies

* Starter-web
* Devtools
* Actuators
* Admin client(Required for your application to register with a Codecentric's Spring Boot Admin Server instance.)
* 

<dependency>

<groupId>de.codecentric</groupId>

<artifactId>spring-boot-admin-starter-client</artifactId>

</dependency>

Step 2:

Configure property in application. Yml as below

server:

port: 9666(Any)

spring:

application:

name: ADMIN-CLIENT-02(Any)

boot:

admin:

client:

url: [http://localhost:8087/ (Adminserver](http://localhost:8087/%20%20(Adminserver) Url)

management:

endpoints:

web:

exposure:

include: '\*' (for All end points)

step 3 : (practice purpose required)

Write one rest controller with any http method

@RestController

**public** **class** AdminRestController {

@GetMapping("/welcomes")

**public** String GetWelcome() {

**return** "Welcome to admin 02";

}

}

Step 4:

Run as boot app admin client and Admin server apps

Admin server Url :

<http://localhost:8087/>

