

# How to do centralized logging in microservices? || Centralized logging with RestTemplate

The scope of this video is how to add the Interceptor ..so to create the custom Interceptor I have created the class MyRequestInterceptor…which is implementing the Interface ClientHttpRequestInterceptor.

package com.gl.restTemplateBuilderDemo.config;  
  
import java.time.Duration;  
  
@Configuration  
public class Config {  
 @Bean  
 public RestTemplate restTemplate(RestTemplateBuilder builder) {  
 UriTemplateHandler uriTemplateHandler = new RootUriTemplateHandler("http://localhost:8083/springDataDemo");  
 return builder  
 .uriTemplateHandler(uriTemplateHandler)  
 .setReadTimeout(Duration.*ofMillis*(2000))  
 .build();  
 }  
  
 @Bean  
 public MyRequestInterceptor myRequestInterceptor() {  
 return new MyRequestInterceptor();  
 }  
  
 @Bean  
 public MyRestTemplateCustomizer restTemplateCustomizer() {  
 return new MyRestTemplateCustomizer();  
 }  
  
 @Bean  
 @DependsOn("restTemplateCustomizer")  
 public RestTemplateBuilder restTemplateBuilder(RestTemplateCustomizer restTemplateCustomizer) {  
 return new RestTemplateBuilder(restTemplateCustomizer);  
 }  
}

package com.gl.restTemplateBuilderDemo.config;  
public class MyRequestInterceptor implements ClientHttpRequestInterceptor {  
 private Logger logger = LoggerFactory.*getLogger*(MyRequestInterceptor.class);  
  
 @Override  
 public ClientHttpResponse intercept(HttpRequest request, byte[] body, ClientHttpRequestExecution execution) throws IOException {  
  
 logger.info("Request details");  
 logger.info(" URI - {}", request.getURI());  
 logger.info("Headers - {}", request.getHeaders());  
 logger.info("Method - {}", request.getMethod());  
 //request.  
  
 ClientHttpResponse response = execution.execute(request, body);  
  
 MyClientHttpResponse myClientHttpResponse = new MyClientHttpResponse(response);  
  
 logger.info("Response details");  
 logger.info("status -{}", myClientHttpResponse.getStatusCode());  
 logger.info("Body -{}", getResponseBody(myClientHttpResponse.getBody()));  
  
 return myClientHttpResponse;  
 }  
  
 private String getResponseBody(InputStream responseBody) {  
  
 StringBuilder inputStringBuilder = new StringBuilder();  
  
 try (BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(responseBody, StandardCharsets.*UTF\_8*)))  
 {  
 String line = bufferedReader.readLine();  
 while (line != null) {  
 inputStringBuilder.append(line);  
 inputStringBuilder.append('\n');  
 line = bufferedReader.readLine();  
 }  
 return inputStringBuilder.toString();  
 } catch (IOException e) {  
 e.printStackTrace();  
 return null;  
 }  
 //IOUtils.  
 }  
}

package com.gl.restTemplateBuilderDemo.config;

public class MyClientHttpResponse implements ClientHttpResponse {  
  
 private ClientHttpResponse clientHttpResponse;  
 private byte[] body = null;  
  
 public MyClientHttpResponse(ClientHttpResponse clientHttpResponse) {  
 this.clientHttpResponse = clientHttpResponse;  
 }  
  
 @Override  
 public HttpStatus getStatusCode() throws IOException {  
 return clientHttpResponse.getStatusCode();  
 }  
  
 @Override  
 public int getRawStatusCode() throws IOException {  
 return clientHttpResponse.getRawStatusCode();  
 }  
  
  
 @Override  
 public String getStatusText() throws IOException {  
 return clientHttpResponse.getStatusText();  
 }  
  
  
 @Override  
 public void close() {  
 clientHttpResponse.close();  
 }  
  
 @Override  
 public InputStream getBody() throws IOException {  
 if (body != null) {  
 return new ByteArrayInputStream(body);  
 }  
 ByteArrayOutputStream outputStream = new ByteArrayOutputStream();  
 IOUtils.*copyLarge*(clientHttpResponse.getBody(), outputStream);  
 body = outputStream.toByteArray();  
 return new ByteArrayInputStream(body);  
 }  
  
 @Override  
 public HttpHeaders getHeaders() {  
 return clientHttpResponse.getHeaders();  
 }  
}

package com.gl.restTemplateBuilderDemo.config;public class MyRestTemplateCustomizer implements RestTemplateCustomizer {  
  
 @Autowired  
 private MyRequestInterceptor myRequestInterceptor;  
  
 @Override  
 public void customize(RestTemplate restTemplate) {  
 restTemplate.getInterceptors().add(myRequestInterceptor);  
 }  
}

package com.gl.restTemplateBuilderDemo.config;  
public class User  
{  
 private Long id;  
 private String name;  
 private String address;  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) { this.id = id; }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getAddress() {  
 return address;  
 }  
  
 public void setAddress(String address) {  
 this.address = address;  
 }  
  
 @Override  
 public String toString() {  
 return "User{" +  
 "id=" + id +  
 ", name='" + name + '\'' +  
 ", address='" + address + '\'' +  
 '}';  
 }  
}

package com.gl.restTemplateBuilderDemo;@RestController  
public class MainController {  
  
 @Autowired  
 private RestTemplate restTemplate;  
  
 @GetMapping("userList")  
 List<User> getUserList()  
 {  
 return restTemplate.getForObject("/users\_data",List.class);  
 }  
 @GetMapping("user\_d")  
 String getUser()  
 {  
 return restTemplate.getForObject("/userdata/2",String.class);  
 }  
}

package com.gl.restTemplateBuilderDemo;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
@SpringBootApplication  
public class RestTemplateBuilderDemo {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(RestTemplateBuilderDemo.class, args);  
  
 }  
}

<http://localhost:8087/user_d>

{"id":2,"name":"Raju","address":"temp"}

package com.gl.restTemplateBuilderDemo.config;  
  
import org.springframework.web.client.RestTemplate;  
public class GoogleTest {  
 public static void main(String[] args) {  
 RestTemplate rt = new RestTemplate();  
  
 String url = "https://maps.googleapis.com/maps/api/geocode/json?address=";  
 String address = "india";  
 String key = "&key=custom-key";//"AIzaSyAX\_1d4cEaA50VZDg6BIJ\_TSnncbLWiegY";  
 url=url+address+key;  
 System.*out*.println(url);  
 String response = rt.getForObject(url, String.class);  
 System.*out*.println(response);  
 }  
}

Note:- in console I got both Request details and Response details…so inside Controller you can have any endpoint that you can have ….

But inside the resource no need to add the logger that showing the request and response details…

So now inside controller u can have any endpoint u can have but inside the method we don’t have write to many logger that these all r request and response details…

So before ending each endpoint it is going through this MyRequestInterceptor and it will log the request details and response details also…

So one place logging for all the endpoint you will have with your service…so this is one to adding log details to one place

@GetMapping("userList")  
List<User> getUserList()  
{  
 return restTemplate.getForObject("/users\_data",List.class);  
}

PostMan:-

<http://localhost:8087/userList>

[

{

"id": 1,

"name": "Bharat",

"address": "Bangalore"

},

{

"id": 2,

"name": "Raju",

"address": "temp"

},

{

"id": 6,

"name": "asedrtyui",

"address": "yui"

},

{

"id": 7,

"name": "yui",

"address": "asd"

},

{

"id": 8,

"name": "yui",

"address": "asd"

},

{

"id": 9,

"name": "zxcv",

"address": "yui"

},

{

"id": 13,

"name": "kalpana",

"address": "us"

},

{

"id": 14,

"name": "kalpana",

"address": "us"

},

{

"id": 15,

"name": "guggupuggu",

"address": "Canada"

},

{

"id": 16,

"name": "guggupuggu",

"address": "Canada"

},

{

"id": 17,

"name": "temp",

"address": "Canada"

},

{

"id": 18,

"name": "temp1",

"address": "Canada"

}

]

Console:-

2020-07-16 16:12:23.679 INFO 32140 --- [ main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8087 (http) with context path ''

2020-07-16 16:12:23.687 INFO 32140 --- [ main] c.g.r.RestTemplateBuilderDemo : Started RestTemplateBuilderDemo in 7.331 seconds (JVM running for 10.029)

2020-07-16 16:12:57.356 INFO 32140 --- [nio-8087-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'

2020-07-16 16:12:57.356 INFO 32140 --- [nio-8087-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'

2020-07-16 16:12:57.376 INFO 32140 --- [nio-8087-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 20 ms

2020-07-16 16:12:57.725 INFO 32140 --- [nio-8087-exec-1] c.g.r.config.MyRequestInterceptor : Request details

2020-07-16 16:12:57.725 INFO 32140 --- [nio-8087-exec-1] c.g.r.config.MyRequestInterceptor : URI - http://localhost:8083/springDataDemo/users\_data

2020-07-16 16:12:57.728 INFO 32140 --- [nio-8087-exec-1] c.g.r.config.MyRequestInterceptor : Headers - [Accept:"application/json, application/\*+json", Content-Length:"0"]

2020-07-16 16:12:57.730 INFO 32140 --- [nio-8087-exec-1] c.g.r.config.MyRequestInterceptor : Method - GET

2020-07-16 16:12:59.228 INFO 32140 --- [nio-8087-exec-1] c.g.r.config.MyRequestInterceptor : Response details

2020-07-16 16:12:59.229 INFO 32140 --- [nio-8087-exec-1] c.g.r.config.MyRequestInterceptor : status -200 OK

2020-07-16 16:12:59.235 INFO 32140 --- [nio-8087-exec-1] c.g.r.config.MyRequestInterceptor : Body -[{"id":1,"name":"Bharat","address":"Bangalore"},{"id":2,"name":"Raju","address":"temp"},{"id":6,"name":"asedrtyui","address":"yui"},{"id":7,"name":"yui","address":"asd"},{"id":8,"name":"yui","address":"asd"},{"id":9,"name":"zxcv","address":"yui"},{"id":13,"name":"kalpana","address":"us"},{"id":14,"name":"kalpana","address":"us"},{"id":15,"name":"guggupuggu","address":"Canada"},{"id":16,"name":"guggupuggu","address":"Canada"},{"id":17,"name":"temp","address":"Canada"},{"id":18,"name":"temp1","address":"Canada"}]