# Spring Boot RestTemplate Error Handling



Create a beautiful, professional-grade website in just a few clicks with Odoo's Open-Source Website Builder.

ads via Carbon

In <u>an earlier article</u>, I wrote about making HTTP requests using the RestTemplate class in a Spring Boot application.

In this quick article, you'll learn how to handle the errors thrown by the RestTemplate during the execution of an HTTP request.

#### **Default Error Handling**

By default, if there is an error during the execution of the request or the server returns a non-successful HTTP status code (4xx or 5xx), RestTemplate will throw one of the following exceptions:

- HttpClientErrorException For HTTP status code 4xx
- HttpServerErrorException For HTTP status code 5xx
- UnknownHttpStatusCodeException In case of an unknown HTTP status code

All these exceptions extend a common base class called RestClientResponseException that contains actual HTTP response data.

### **Error Handling using a Try-Catch Block**

The simplest way to add a custom error handler is to use a try-catch block to catch the HttpStatusCodeException exception. From the HttpStatusCodeException instance, you can then get the response status code, body, and headers, as shown below:

```
try {
    // request url
    String url = "https://reqres.in/api/unknown/23";
    // create an instance of RestTemplate
    RestTemplate restTemplate = new RestTemplate();
    // make an HTTP GET request
    ResponseEntity<String> response = restTemplate.getForEntity(url, String.class);
} catch (HttpStatusCodeException ex) {
    // raw http status code e.g `404`
    System.out.println(ex.getRawStatusCode());
    // http status code e.g. `404 NOT_FOUND`
    System.out.println(ex.getStatusCode().toString());
    // get response body
    System.out.println(ex.getResponseBodyAsString());
    // get http headers
    HttpHeaders headers = ex.getResponseHeaders();
    System.out.println(headers.get("Content-Type"));
    System.out.println(headers.get("Server"));
}
```

### **Implementing a Custom Error Handler**

Sometimes, a try-catch block is not enough to handle errors as it is not scalable when the number of HTTP request increases.

You may want to create a reusable custom error handler by implementing the ResponseErrorHandler interface, as follows:

```
MyErrorHandler.java
```

```
package com.attacomsian.runner;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpStatus;
import org.springframework.http.client.ClientHttpResponse;
import org.springframework.web.client.DefaultResponseErrorHandler;
import org.springframework.web.client.ResponseErrorHandler;
import java.io.IOException;
public class MyErrorHandler implements ResponseErrorHandler {
   @Override
   public boolean hasError(ClientHttpResponse response) throws IOException {
        return new DefaultResponseErrorHandler().hasError(response);
   }
   @Override
   public void handleError(ClientHttpResponse response) throws IOException {
        if (response.getStatusCode().series() == HttpStatus.Series.SERVER_ERROR) {
            // handle 5xx errors
            // raw http status code e.g `500`
            System.out.println(response.getRawStatusCode());
            // http status code e.g. `500 INTERNAL_SERVER_ERROR`
            System.out.println(response.getStatusCode());
        } else if (response.getStatusCode().series() == HttpStatus.Series.CLIENT_ERROR) {
            // handle 4xx errors
            // raw http status code e.g `404`
            System.out.println(response.getRawStatusCode());
            // http status code e.g. `404 NOT_FOUND`
            System.out.println(response.getStatusCode());
            // get response body
            System.out.println(response.getBody());
            // get http headers
            HttpHeaders headers = response.getHeaders();
            System.out.println(headers.get("Content-Type"));
            System.out.println(headers.get("Server"));
        }
   }
}
```

You can now create an instance of MyErrorHandler and pass it to the RestTemplate class:

## Become A Supporter 🖫

Every month, tens of thousands of web developers read my tutorials to find a solution to their problems.

All my content is **100% free**! I want to

```
// request url
String url = "https://reqres.in/api/unknown/23";

// create an instance of RestTemplate
RestTemplate restTemplate = new RestTemplate();

// set custom error handler
restTemplate.setErrorHandler(new MyErrorHandler());

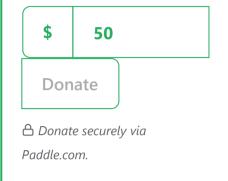
// make an HTTP GET request
ResponseEntity<String> response = restTemplate.getForEntity(url, String.class);
```

Check out the <u>Making HTTP Requests using RestTemplate in Spring Boot</u> guide for more RestTemplate examples.

Like this article? Follow me on <u>Twitter</u> and <u>LinkedIn</u>. You can also Subscribe to <u>RSS Feed</u>.

produce more
content especially indepth guides for creating REST APIs in Spring Boot and MEAN Stack.

Your support will help me <u>allocate</u> more time for my blog. I'll put more efforts to write high-quality tutorials.



Last Updated: November 09, 2019

**Spring Boot** 

#### You might also like...

- > How to access data in Thymeleaf templates
- > Spring Boot Optional Path Variables
- Displaying a Custom Error Page in Spring Boot
- > How to change or disable the default banner in Spring Boot
- > How to change the default port in Spring Boot

# ★ Learn to build modern web applications using JavaScript and Spring Boot

I started this blog as a place to share everything I have learned in the last decade. I write about modern JavaScript, Node.js, Spring Boot, core Java, RESTful APIs, and all things web development.

The newsletter is sent every week and includes early access to clear, concise, and easy-to-follow tutorials, and other stuff I think you'd enjoy! No spam ever, unsubscribe at any time.



CAtta.

Developer. Entrepreneur.

© 2020 Made with 💛 by Atta

Products	Links	Follow Me
Startup UI Kit	<u>Articles</u>	<u>Twitter</u>
<u>DomBuck</u>	<u>Protips</u>	<u>LinkedIn</u>
<u>StartupBase</u>	<u>Sponsors</u>	<u>GitHub</u>
<u>Whobal</u>	Contact	RSS Feed