

# Handlers and Response Entity

→ Handlers are Exception handling.

## ① Global exception handlers -

→ we use these because spring returns ugly HTML errors

→ with handler: you control JSON response format

It gives:

- cleaner API
- consistent error format
- production ready behaviour.

use `@RestControllerAdvice`

## ② Response Entity -

It lets us control -

- HTTP status code
- Headers
- Response body.

Eg → return `ResponseEntity.status(201).body(saved data)`

meaning status 201 created  
& body = saved data

global exception  
handler maps  
errors along  
with messages

```
GlobalExceptionHandler.java
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.RestControllerAdvice;

import java.util.HashMap;
import java.util.Map;

@RestControllerAdvice no usages
public class GlobalExceptionHandler {
    @ExceptionHandler(StudentNotFoundException.class) no usages
        public ResponseEntity<Map<String, String>> handleNotFound(StudentNotFoundException ex){
            Map<String, String> response = new HashMap<>();
            response.put("error", "Student not found");
            response.put("message", ex.getMessage());
            return ResponseEntity.status(404).body(response);
        }
}
```

Response  
entity  
provides  
status along  
with body.

```
15 }  
16 response.put("Error", ex.getMessage());  
17 return ResponseEntity.status(404).body(response);  
18 }
```

```
④ StudentNotFoundException.java ×  
1 package com.example.Day5.exception;  
2  
3 public class StudentNotFoundException extends RuntimeException{  
4     public StudentNotFoundException(Long id) { 3 usages  
5         super("Student not found with id: " + id);  
6     }  
7 }
```