**Exception Handling in Springboot**

**Approach 1 : traditional approach**

**com.itp.sms.exception package - > class PlayerNotFoundException**

**public** **class** PlayerNotFoundException **extends** RuntimeException

{

**public** PlayerNotFoundException(String errorMessage)

{

**super**(errorMessage);

}

}

**com.itp.sms.service package - > class PlayerService**

**public** Player getPlayer(**int** jno) {

Optional<Player> player=playerRepository.findById(jno);

**if**(player.isPresent())

**return** player.get();

**else**

**throw** **new** PlayerNotFoundException("Player does not exist");

}

**com.itp.sms.controller package - > class PlayerController**

@GetMapping("/getPlayerById/{pid}")

**public** ResponseEntity<?> getPlayerById (@PathVariable int pid)

{

**try**

{

Player player=playerService.getPlayerById(pid);

**return** **new** ResponseEntity<Player>(player,HttpStatus.***OK***);

}

**catch**(PlayerNotFoundException ex1)

{

**return** **new** ResponseEntity<String>(ex1.getMessage(),HttpStatus.***BAD\_REQUEST***);

}

**Approach 2 : using GlobalExceptionHandler**

**com.itp.sms.exception package - > class GlobalExceptionHandler**

**Option 1:**

@ControllerAdvice

**public** **class** GlobalExceptionHandler {

@ExceptionHandler(PlayerNotFoundException.class)

public ResponseEntity<String> handlePlayerNotFound(PlayerNotFoundException ex)

{

return new ResponseEntity<String>(ex.getMessage(),HttpStatus.NOT\_FOUND);

}

}

**NOTE :**

**Service layer code remains unchaged**

**In the controller we will not do any handling**

@GetMapping("/getProductById1/{pid}")

**public** ResponseEntity<Product> getAllProductById1(@PathVariable **int** pid)

{

Product product=**null**;

product=productService.getAllProductById(pid);

**return** **new** ResponseEntity<Product>(product,HttpStatus.***OK***) ;

}

**Option 2:**

@ControllerAdvice

**public** **class** GlobalExceptionHandler {

@ExceptionHandler(PlayerNotFoundException.class)

public ResponseEntity<String> handlePlayerNotFound(PlayerNotFoundException ex)

{

return new Map<String, Object> body = new HashMap<>();

body.put("error", "Player Not Found");

body.put("message", ex.getMessage());

body.put("timestamp", LocalDateTime.now());

return new ResponseEntity<Map<String, Object>>(body,HttpStatus.NOT\_FOUND);

}

}

**Option 3:**

**com.itp.sms.response package - > class ErrorResponse**

**public** **class** ErrorResponse {

**private** String error;

**private** String message;

**private** LocalDateTime timestamp;

**private** **int** status;

constructors,setter getters, toString()

**}**

@ControllerAdvice

**public** **class** GlobalExceptionHandler {

@ExceptionHandler(PlayerNotFoundException.class)

public ResponseEntity<String> handlePlayerNotFound(PlayerNotFoundException ex)

{

ErrorResponse body = **new** ErrorResponse("Player Not Found",

ex.getMessage(), LocalDateTime.*now*(),

HttpStatus.***NOT\_FOUND***.value());

**return** **new** ResponseEntity<ErrorResponse>(body,HttpStatus.***NOT\_FOUND***);

}

}