

## @ResponseStatus 特定场景业务案例

=====

``ResponseStatus`` 注解，它用于表示一个方法的调用结果应该返回特定的 HTTP 状态码。这通常用于异常处理或当方法执行成功但需要通知客户端一个特定的状态时。

### ### 业务场景：

假设你正在开发一个 RESTful API，用于管理用户账户。当用户尝试删除自己的账户时，如果操作成功，你希望返回 HTTP 状态码 ``200 OK``；如果用户没有足够的权限执行删除操作，则返回 ``403 Forbidden``。

#### #### 1. 使用 ``ResponseStatus`` 注解：

```
...  
import org.springframework.http.HttpStatus;  
import org.springframework.web.bind.annotation.*;  
import org.springframework.web.server.ResponseStatusException;  
  
@RestController  
@RequestMapping("/users")  
public class UserController {  
  
    @DeleteMapping("/{id}")  
    @ResponseStatus(HttpStatus.OK) // 指定成功时的 HTTP 状态码  
    public void deleteUser(@PathVariable Long id) {  
        // 删除用户的业务逻辑  
        if (!userService.hasDeletePermission(id)) {  
            throw new ResponseStatusException(HttpStatus.FORBIDDEN,  
                "User does not have permission to delete this account.");  
        }  
        // 执行删除操作...  
    }  
}  
...
```

在这个控制器中，``deleteUser`` 方法使用 ``@ResponseStatus`` 注解来指定当用户成功删除账户时，应返回 HTTP 状态码 ``200 OK``。如果用户没有删除权

限，将抛出 `ResponseStatusException` 异常，并返回 `403 Forbidden`。

## #### 2. 异常处理：

Spring MVC 还允许你通过异常处理来设置 `ResponseStatus`：

```
...
@ControllerAdvice
public class GlobalExceptionHandler {

    @ExceptionHandler(ResponseStatusException.class)
    public ResponseEntity<Object>
handleResponseStatusException(ResponseStatusException ex) {
        // 可以自定义响应体，例如包含错误信息的 JSON 对象
        ErrorResponse errorResponse = new
ErrorResponse(ex.getStatus().value(), ex.getReason());
        return new ResponseEntity<>(errorResponse, ex.getStatus());
    }

    public static class ErrorResponse {
        private int status;
        private String message;

        public ErrorResponse(int status, String message) {
            this.status = status;
            this.message = message;
        }

        // 标准的 getter 和 setter
    }
}
...
```

## #### 3. 测试 API：

\* 当用户具有删除权限时，发送 DELETE 请求到 `/users/{id}`，你将得到一个空的响应体和一个 `200 OK` 状态码。

\* 当用户没有删除权限时，同样的请求将导致 `403 Forbidden` 状态码，并返回一个包含错误信息的 JSON 对象。

## ### 总结：

- \* `@ResponseStatus` 允许开发者明确地指定方法执行成功后应该返回的 HTTP 状态码。
- \* 它提高了 API 的透明度和一致性，使得客户端开发者可以更清楚地理解 API 的行为。
- \* 结合异常处理，`@ResponseStatus` 可以提供统一和定制化的错误处理机制。

原文链接: <https://juejin.cn/post/7382511904683835418>