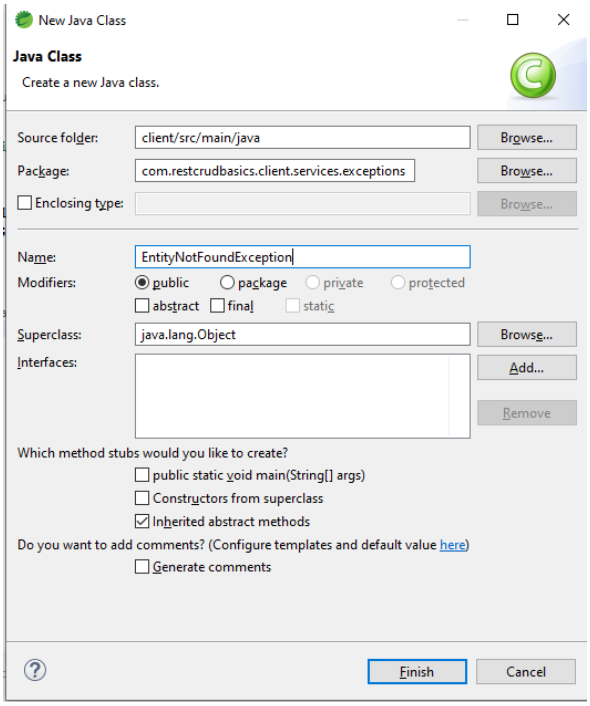
- When we need to pass a argument like a id in a request, we have to treat the errors if we put a argument that is not find. This kind of error we treat with exceptions.
- First, we need to create the exceptions package



- Implement the Exception

```
package com.restcrudbasics.client.services.exceptions;

public class EntityNotFoundException extends RuntimeException {
    private static final long serialVersionUID = 1L;

    public EntityNotFoundException(String msg) {
        super(msg);
    }

}
```

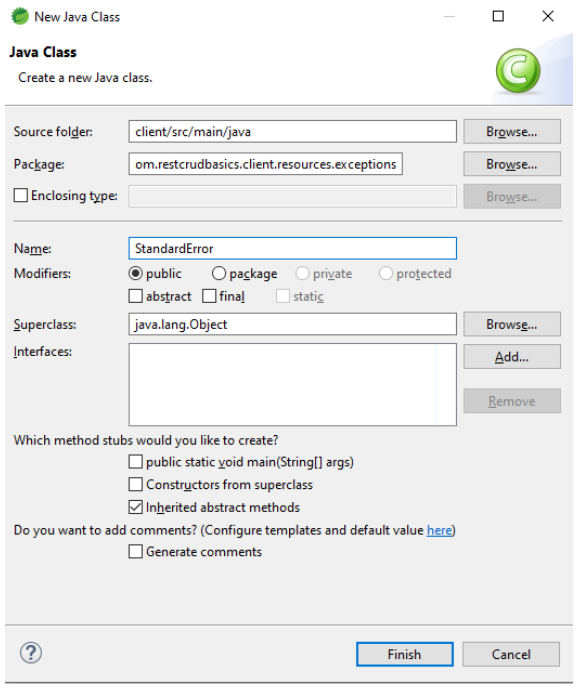
- Now, we need to implement the try-catch method in the service entity

```
@Transactional(readOnly = true)
public ClientDTO findById(Long id) {
    Optional<Client> obj = repository.findById(id);
    Client entity = obj.orElseThrow(() -> new EntityNotFoundException("Entity not found"));
    return new ClientDTO(entity);
}
```

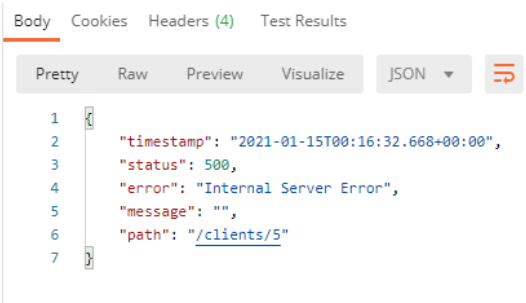
- If we run the app, we keep getting the 500 error, but, the log in STS how the custom exception that we create.

```
client - ClientApplication [Spring Boot App]
2021-01-14 22:16:32.662 ERROR 4100 --- [nio-8080-exec-2] o.a.c.c.C.[.[.[/].[dispatcherServlet]    : Servlet.service()

com.restcrudbasics.client.services.exceptions.EntityNotFoundException: Entity not found
        at com.restcrudbasics.client.services.ClientService.lambda$1(ClientService.java:31) ~[classes/:na]
        at java.base/java.util.Optional.orElseThrow(Optional.java:408) ~[na:na]
        at com.restcrudbasics.client.services.ClientService.findById(ClientService.java:31) ~[classes/:na]
```

- Now, we have to return a 404 response when the error occurs.
- For this, we have to implement the error in the resource, to show the error.
- Create a new class exception in the layer resource.



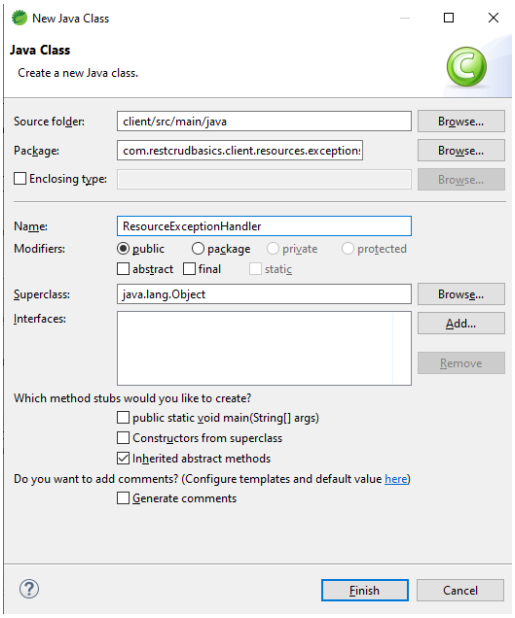
- This class have the same properties of the JSON error



- The implement of error class

```
package com.restcrudbasics.client.resources.exceptions;

import java.time.Instant;

public class StandardError {
    private static final long serialVersionUID = 1L;

    private Instant timestamp;
    private Integer status;
    private String error;
    private String message;
    private String path;

    public StandardError() {

    }

    public Instant getTimestamp() {
        return timestamp;
    }

    public void setTimestamp(Instant timestamp) {
        this.timestamp = timestamp;
    }

    public Integer getStatus() {
        return status;
    }

    public void setStatus(Integer status) {
        this.status = status;
    }

    public String getError() {
        return error;
    }
```

- To not treat everyone Resource Method that throws an exception, we create a Advice Controller to economy and make a clean code.
- Let's create the class



- Now, we need to implement the Handler of the exception to show the customize treated error in the http response.

ClientResource.java | ClientService.java | ClientDTO.java | EntityNotFoundException.java | StandardError.java | ClientRepository.java | ResourceExceptionHandler.java

```java
package com.restcrudbasics.client.resources.exceptions;

import java.time.Instant;

import javax.servlet.http.HttpServletRequest;

import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.ControllerAdvice;
import org.springframework.web.bind.annotation.ExceptionHandler;

import com.restcrudbasics.client.services.exceptions.EntityNotFoundException;

@ControllerAdvice // intercept the error that occur in the resource layer
public class ResourceExceptionHandler {

    @ExceptionHandler(EntityNotFoundException.class) // every time a error EntityNotFoundException occur, the error will be treat with the ResponseEntity Method
    public ResponseEntity<StandardError> entityNotFound(EntityNotFoundException e, HttpServletRequest request){ //this method recive the exception and the information of the Http request
        StandardError err = new StandardError();
        err.setTimestamp(Instant.now());
        err.setStatus(HttpStatus.NOT_FOUND.value());
        err.setError("Resource not found");
        err.setMessage(e.getMessage());
        err.setPath(request.getRequestURI());
        return ResponseEntity.status(HttpStatus.NOT_FOUND).body(err);
    }

}
```

- Now, when the error occurs, the http response will show the treated error and 404 error status