WEEK:4

**Validating country code**   
  
As the POST request is a plain text, there are good possibilities to key in incorrect data. Moreover, hackers might try to pass inconsistent data which might affect the integrity of the application. Hence it becomes important that necessary check are in place for all the fields. In this hands on we will take a simple validation criteria and will see how it can be implemented.  
  
The country code needs to be validated and ensured that it does not exceed more than 2 characters. Refer the steps below to incorporate the same: 

* Open Country.java and include below annotations for the code property. @NotNull ensure that code is not null. @Size ensure that the width is exactly 2 characters.

    @NotNull

    @Size(min=2, max=2, message="Country code should be 2 characters")

    private String code;

* In CountryController.addCountry() method add below lines after the logger. This uses the javax.validation specification to check if the bean has errors based on the annotations defined in the earlier step. All new class references in this code snippet needs to be imported from javax.validation.

// Create validator factory

       ValidatorFactory factory = Validation.buildDefaultValidatorFactory();

        Validator validator = factory.getValidator();

  // Validation is done against the annotations defined in country bean

        Set<ConstraintViolation<Country>> violations = validator.validate(country);

        List<String> errors = new ArrayList<String>();

  // Accumulate all errors in an ArrayList of type String

        for (ConstraintViolation<Country> violation : violations) {

            errors.add(violation.getMessage());

        }

  // Throw exception so that the user of this web service receives appropriate error message

        if (violations.size() > 0) {

            throw new ResponseStatusException(HttpStatus.BAD\_REQUEST, errors.toString());

        }

* Invoke the service using curl and check the response. Refer sample response below:

HTTP/1.1 400

Content-Type: application/json;charset=UTF-8

Transfer-Encoding: chunked

Date: Wed, 02 Oct 2019 10:28:56 GMT

Connection: close

{"timestamp":"2019-10-02T10:28:56.506+0000","status":400,"error":"Bad Request","message":"[Country code should be 2 characters]","path":"/countries"}

**Question for all Learners -**What needs to be done if there is another controller EmployeeController and similar validation needs to be done for Employee payload data?  
  
SME to explain the disadvantage of the above solution.  
  
This disadvantage will be overcome in the next hands on.

CODE:

import jakarta.validation.ConstraintViolation;

import jakarta.validation.Validation;

import jakarta.validation.Validator;

import jakarta.validation.ValidatorFactory;

import org.springframework.http.HttpStatus;

import org.springframework.web.server.ResponseStatusException;

import java.util.ArrayList;

import java.util.List;

import java.util.Set;

@PostMapping("/countries")

public ResponseEntity<Country> addCountry(@RequestBody Country country) {

logger.info("Start");

ValidatorFactory factory = Validation.buildDefaultValidatorFactory();

Validator validator = factory.getValidator();

Set<ConstraintViolation<Country>> violations = validator.validate(country);

List<String> errors = new ArrayList<>();

for (ConstraintViolation<Country> violation : violations) {

errors.add(violation.getMessage());

}

if (!violations.isEmpty()) {

throw new ResponseStatusException(HttpStatus.BAD\_REQUEST, errors.toString());

}

return new ResponseEntity<>(country, HttpStatus.CREATED);

}