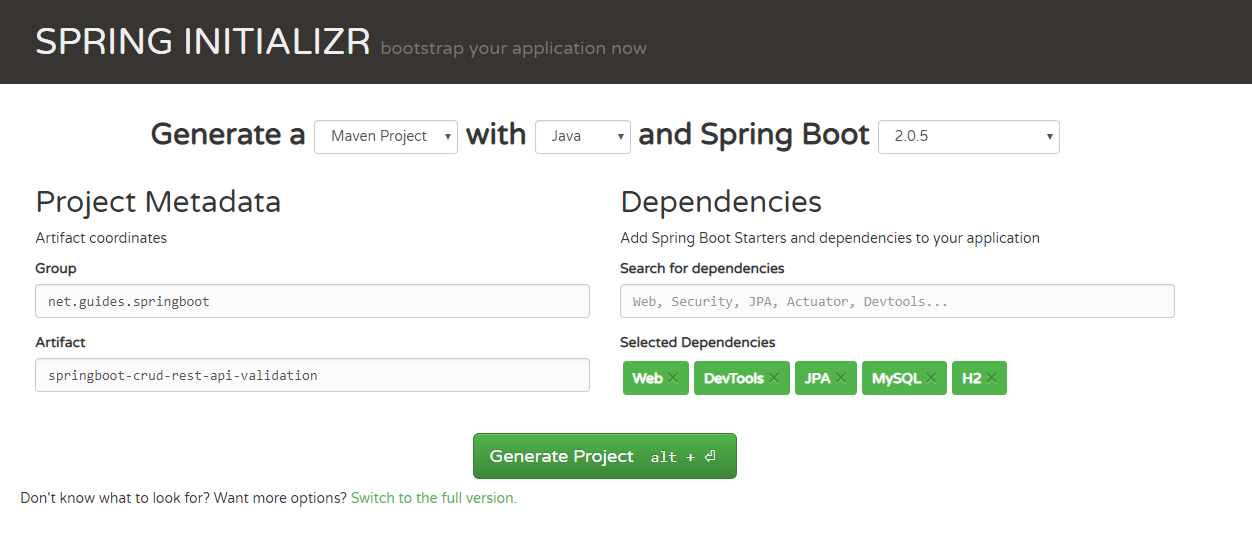
Rest api pour la gestion des employées et implémentation d'une Bean validation en utilisant Hibernate

# Outils et technologies utilisés

Il y a plusieurs façons de créer une application spring boot. Mais le plus simple est d’aller à <http://start.spring.io/> qui est un site permettant de générer une application spring boot.



Group : **com.guides.springboot**

Artifact : **springboot-crud-rest-api-validation**

Dependences : **Web, DevTools, JPA, MySQL, H2**

Puis cliquer sur Generate Project. (Ensuite extraire le fichier zip)

Le contenu du fichier zip devrait être comme ceci :

#TODO

# Ouverture du projet dans IntelliJ

#TODO

# Contenu du fichier pom.xml

Le contenu du fichier pom.xml devrait être comme ceci :

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>net.guides.springboot</groupId>

<artifactId>springboot-crud-rest-api-validation</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>springboot-crud-rest-api-validation</name>

<description>Demo project for Spring Boot</description>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.0.5.RELEASE</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>com.h2database</groupId>

<artifactId>h2</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

# Creation JPA Entity et implementation de la validation sur le Bean Employee.java

La classe Employee devrait ressembler à ceci :

package net.guides.springboot.springbootcrudrestapivalidation.model;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

import javax.validation.constraints.Email;

import javax.validation.constraints.NotBlank;

import javax.validation.constraints.NotNull;

import javax.validation.constraints.Size;

@Entity

@Table(name = "employees")

public class Employee {

private long id;

@NotNull

@Size(min = 2, message = "First Name should have atleast 2 characters")

private String firstName;

@NotNull

@Size(min = 2, message = "Last Name should have atleast 2 characters")

private String lastName;

@Email

@NotBlank

private String emailId;

@NotNull

@Size(min = 2, message = "Passport should have atleast 2 characters")

private String passportNumber;

public Employee() {

}

public Employee(String firstName, String lastName, String emailId) {

this.firstName = firstName;

this.lastName = lastName;

this.emailId = emailId;

}

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

public long getId() {

return id;

}

public void setId(long id) {

this.id = id;

}

@Column(name = "first\_name", nullable = false)

public String getFirstName() {

return firstName;

}

public void setFirstName(String firstName) {

this.firstName = firstName;

}

@Column(name = "last\_name", nullable = false)

public String getLastName() {

return lastName;

}

public void setLastName(String lastName) {

this.lastName = lastName;

}

@Column(name = "email\_address", nullable = false)

public String getEmailId() {

return emailId;

}

public void setEmailId(String emailId) {

this.emailId = emailId;

}

@Column(name = "passport\_number", nullable = false)

public String getPassportNumber() {

return passportNumber;

}

public void setPassportNumber(String passportNumber) {

this.passportNumber = passportNumber;

}

}

# Creation de la repository EmployeeRepository.java

package net.guides.springboot.springbootcrudrestapivalidation.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import net.guides.springboot.springbootcrudrestapivalidation.model.Employee;

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Long>{

}

# Configuration de MySQL dans application.properties

spring.datasource.url = jdbc:mysql://localhost:3306/employees?useSSL=false

spring.datasource.username = root

spring.datasource.password = root

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

spring.jpa.hibernate.ddl-auto = update

# Mise en place du controller EmployeeController.java

La classe devrait ressembler à ceci ;

package net.guides.springboot.springbootcrudrestapivalidation.controller;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import net.guides.springboot.springbootcrudrestapivalidation.exception.ResourceNotFoundException;

import net.guides.springboot.springbootcrudrestapivalidation.model.Employee;

import net.guides.springboot.springbootcrudrestapivalidation.repository.EmployeeRepository;

@RestController

@RequestMapping("/api/v1")

public class EmployeeController {

@Autowired

private EmployeeRepository employeeRepository;

@GetMapping("/employees")

public List<Employee> getAllEmployees() {

return employeeRepository.findAll();

}

@GetMapping("/employees/{id}")

public ResponseEntity<Employee> getEmployeeById(@PathVariable(value = "id") Long employeeId)

throws ResourceNotFoundException {

Employee employee = employeeRepository.findById(employeeId)

.orElseThrow(() -> new ResourceNotFoundException("Employee not found for this id :: " + employeeId));

return ResponseEntity.ok().body(employee);

}

@PostMapping("/employees")

public Employee createEmployee(@Valid @RequestBody Employee employee) {

return employeeRepository.save(employee);

}

@PutMapping("/employees/{id}")

public ResponseEntity<Employee> updateEmployee(@PathVariable(value = "id") Long employeeId,

@Valid @RequestBody Employee employeeDetails) throws ResourceNotFoundException {

Employee employee = employeeRepository.findById(employeeId)

.orElseThrow(() -> new ResourceNotFoundException("Employee not found for this id :: " + employeeId));

employee.setEmailId(employeeDetails.getEmailId());

employee.setLastName(employeeDetails.getLastName());

employee.setFirstName(employeeDetails.getFirstName());

final Employee updatedEmployee = employeeRepository.save(employee);

return ResponseEntity.ok(updatedEmployee);

}

@DeleteMapping("/employees/{id}")

public Map<String, Boolean> deleteEmployee(@PathVariable(value = "id") Long employeeId)

throws ResourceNotFoundException {

Employee employee = employeeRepository.findById(employeeId)

.orElseThrow(() -> new ResourceNotFoundException("Employee not found for this id :: " + employeeId));

employeeRepository.delete(employee);

Map<String, Boolean> response = new HashMap<>();

response.put("deleted", Boolean.TRUE);

return response;

}

}

# Mise en place de la gestion d’exception *ResourceNotFoundException.java*

package net.guides.springboot.springbootcrudrestapivalidation.exception;

import org.springframework.http.HttpStatus;

import org.springframework.web.bind.annotation.ResponseStatus;

@ResponseStatus(value = HttpStatus.NOT\_FOUND)

public class ResourceNotFoundException extends Exception{

private static final long serialVersionUID = 1L;

public ResourceNotFoundException(String message){

super(message);

}

}

# Customisation de la réponse de validation

package net.guides.springboot.springbootcrudrestapivalidation.exception;

import java.util.Date;

public class ErrorDetails {

private Date timestamp;

private String message;

private String details;

public ErrorDetails(Date timestamp, String message, String details) {

super();

this.timestamp = timestamp;

this.message = message;

this.details = details;

}

public Date getTimestamp() {

return timestamp;

}

public String getMessage() {

return message;

}

public String getDetails() {

return details;

}

}

*GlobalExceptionHandler.java*

package net.guides.springboot.springbootcrudrestapivalidation.exception;

import java.util.Date;

import org.springframework.http.HttpHeaders;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.MethodArgumentNotValidException;

import org.springframework.web.bind.annotation.ControllerAdvice;

import org.springframework.web.bind.annotation.ExceptionHandler;

import org.springframework.web.context.request.WebRequest;

import org.springframework.web.servlet.mvc.method.annotation.ResponseEntityExceptionHandler;

@ControllerAdvice

public class GlobalExceptionHandler extends ResponseEntityExceptionHandler {

@ExceptionHandler(ResourceNotFoundException.class)

public ResponseEntity<?> resourceNotFoundException(ResourceNotFoundException ex, WebRequest request) {

ErrorDetails errorDetails = new ErrorDetails(new Date(), ex.getMessage(), request.getDescription(false));

return new ResponseEntity<>(errorDetails, HttpStatus.NOT\_FOUND);

}

@ExceptionHandler(Exception.class)

public ResponseEntity<?> globleExcpetionHandler(Exception ex, WebRequest request) {

ErrorDetails errorDetails = new ErrorDetails(new Date(), ex.getMessage(), request.getDescription(false));

return new ResponseEntity<>(errorDetails, HttpStatus.INTERNAL\_SERVER\_ERROR);

}

@Override

protected ResponseEntity<Object> handleMethodArgumentNotValid(MethodArgumentNotValidException ex,

HttpHeaders headers, HttpStatus status, WebRequest request) {

ErrorDetails errorDetails = new ErrorDetails(new Date(), "Validation Failed",

ex.getBindingResult().toString());

return new ResponseEntity(errorDetails, HttpStatus.BAD\_REQUEST);

}

}

# Lancer l’application

package net.guides.springboot.springbootcrudrestapivalidation;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class SpringbootCrudRestApiValidationApplication {

public static void main(String[] args) {

SpringApplication.run(SpringbootCrudRestApiValidationApplication.class, args);

}

}

https://www.javaguides.net/p/top-java-tutorials.html