Mini-tuto

Scenario : ouverture / fermeture de volet en fonction de la luminosité

Comment ? 3 services :

- Luminosité : responsable de fournir la valeur courante de luminosité (devra récupérer cette valeur d'un capteur) – fr.insa.luminosityservice
- Volet : responsable de la gestion de l'ouverture / fermeture du volet en fonction d'une consigne (devra communiquer avec le système physique d'ouverture de volet) – fr.insa.shutterservice
- Système automatique : responsable de la « logique » du système, c'est-à-dire d'envoyer la bonne consigne au service Volet en fonction de l'information fournie par le service Luminosité – fr.insa.automationservice

Ces 3 services sont 3 projets Eclipse générés via https://start.spring.io

Luminosité

Très simple, constitué uniquement de l'Application et d'une ressource :

LuminosityResource.java

```java
package fr.insa.luminosityservice.resources;

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

@RestController
@RequestMapping("/luminosity")
public class LuminosityResource {

    @GetMapping("/value")
    public int getLuminosity() {
        return (int) Math.ceil(Math.random() * 100);
    }

}
```

Requête (port 8080):

http://localhost:8080/luminosity/value GET -> retourne un entier aléatoire entre 0 et 100

Volet

Un peu plus complexe car définit également un modèle pour le volet :

LuminosityResource.java | Shutter.java

```java
package fr.insa.shutterservice.model;

public class Shutter {

    private double currentOpening;
    private double order;
    private boolean isMoving;

    public Shutter() {}

    public Shutter(double currentOpening, double order) {
        this.currentOpening = currentOpening;
        this.order = order;
        updateIsMoving();
    }

    public double getCurrentOpening() {
        return currentOpening;
    }

    public void setCurrentOpening(double currentOpening) {
        this.currentOpening = currentOpening;
        updateIsMoving();
    }

    public double getOrder() {
        return order;
    }

    public void setOrder(double order) {
        this.order = order;
        updateIsMoving();
    }

    public boolean isMoving() {
        return isMoving;
    }

    private void updateIsMoving() {
        this.isMoving = !(this.order == this.currentOpening);
    }
}
```

Permet d'obtenir :

- l'ouverture courante, entre 0 et 1 (0 : complètement fermé, 1 : complètement ouvert)
- la consigne, entre 0 et 1 (0 : fermer complètement, 1 : ouvrir complètement)
- l'indication du mouvement (le volet est-il en train de bouger ?)

La ressource correspondante :

LuminosityResource.java | Shutter.java | ShutterResource.java

```java
package fr.insa.shutterservice.resources;

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

@RestController
@RequestMapping("/shutter")
public class ShutterResource {

    private Shutter shutter = new Shutter(0.5, 1.0);

    @GetMapping("/")
    public Shutter getShutter() {
        return shutter;
    }

    @GetMapping("/status")
    public String getStatus() {
        if (shutter.isMoving()) return "moving from " + shutter.getCurrentOpening() + " to " + shutter.getOrder();
        if (shutter.getCurrentOpening() > 0) return "open";
        return "closed";
    }

    @GetMapping("/opening")
    public double getOpening(@RequestParam(name="unit", required=false) String unit) {
        if (unit == null || unit.equals("decimal")) return shutter.getCurrentOpening();
        if (unit.equals("percent")) return shutter.getCurrentOpening() * 100;
        return -1;
    }

    @PostMapping("/order")
    public void setOrder(@RequestParam double value) {
        shutter.setOrder(value);
    }

    @PutMapping("/")
    public void updateShutter(@RequestBody Shutter shutter) {
        this.shutter.setOrder(shutter.getOrder());
    }

}

```

Une variable de classe Shutter, pour garder une trace des différentes valeurs (normalement inutile si l'on récupère les valeurs directement depuis un capteur sur le volet).

Quelques fonctions accessibles depuis les requêtes suivantes (port 8081) :

http://localhost:8081/shutter/ GET-> retourne l'objet Shutter

http://localhost:8081/shutter/status GET -> retourne une chaîne de caractères indiquant le statut du volet

http://localhost:8081/shutter/opening GET -> retourne la valeur courante de l'ouverture

Peut utiliser un paramètre pour retourner cette valeur en % ou en décimal

http://localhost:8081/shutter/opening?unit=percent ou
http://localhost:8081/shutter/opening?unit=decimal

http://localhost:8081/shutter/order?value=0.6 POST -> envoie une valeur de consigne

http://localhost:8081/shutter/ PUT -> met à jour la consigne (méthode à privilégier)

Système automatique

Reprend le modèle de Volet (Shutter.java), une ressource :

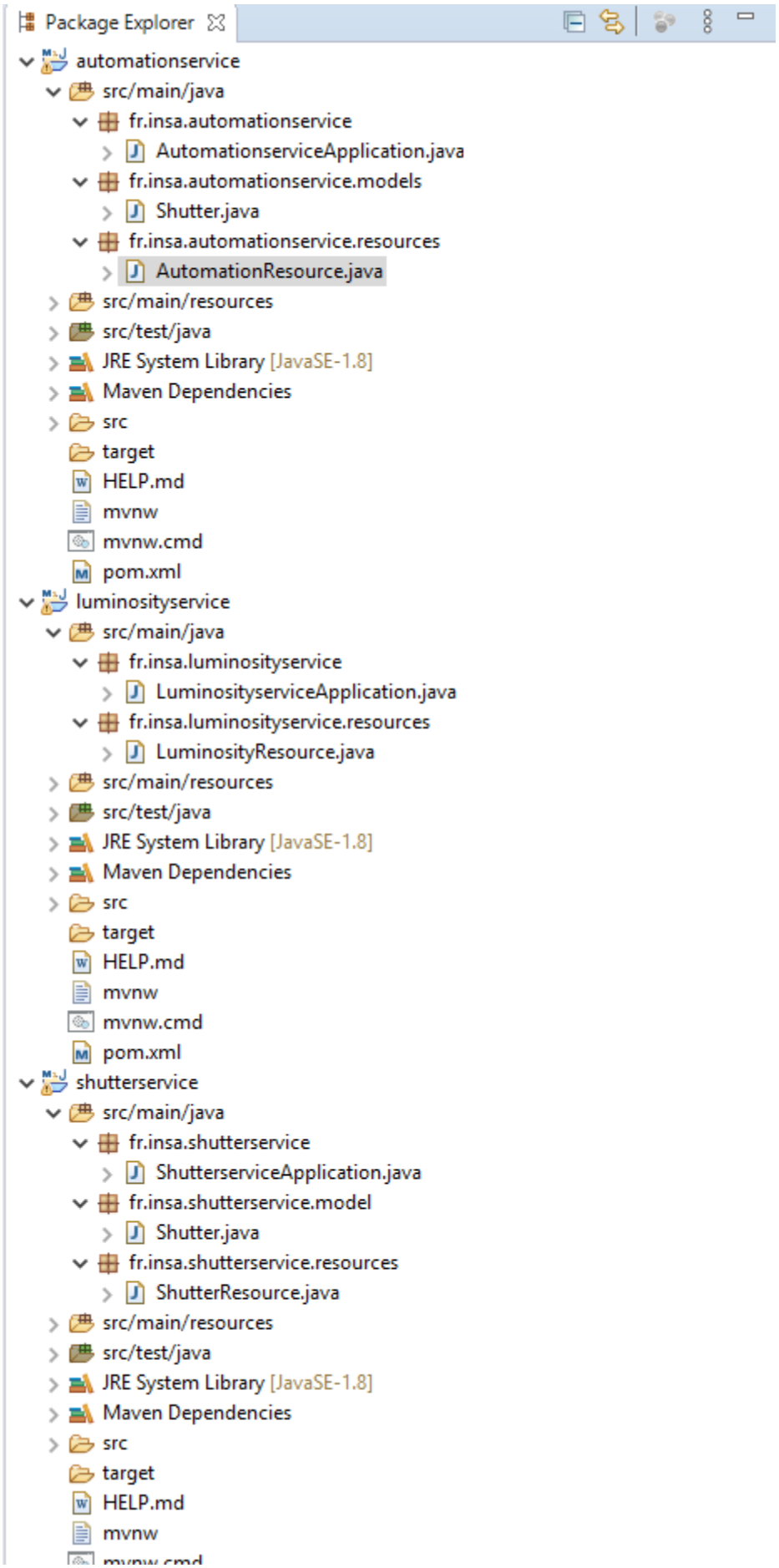
LuminosityResource.java | Shutter.java | ShutterResource.java | AutomationResource.java

```java
package fr.insa.automationservice.resources;

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

@RestController
@RequestMapping("/auto")
public class AutomationResource {

    private final String luminosityURI = "http://localhost:4200/luminosity/";
    private final String shutterURI = "http://localhost:4201/shutter/";

    @GetMapping("/run")
    public String run() {
        RestTemplate restTemplate = new RestTemplate();

        String msg ="";

        // Récupération de la valeur d'ouverture courante, en %
        double currentOpeningPercent = restTemplate.getForObject(shutterURI+"opening?unit=percent", Double.class);
        msg += "Ouverture actuelle: " + currentOpeningPercent + " %";

        // Récupération de l'objet Shutter
        Shutter shutter = restTemplate.getForObject(shutterURI, Shutter.class);
        msg += " ### Commande actuelle : " + shutter.getOrder();

        // Récupération de la valeur de luminosité
        int luminosity = restTemplate.getForObject(luminosityURI+"value", Integer.class);
        msg += " --- La luminosité est de " + luminosity + "\n";

        if (luminosity > 50) {
            msg += " | Forte luminosité -> il faut fermer les volets";
            double order = (100 - luminosity) / 100.0;
            shutter.setOrder(order);
        } else {
            msg += " | Faible luminosité -> il faut ouvrir les volets";
            double order = 0.005 + (100 - luminosity) / 100.0;
            shutter.setOrder(order);
        }

        // Requête de mise à jour du Shutter
        restTemplate.put(shutterURI, shutter);

        // Récupération du nouveau statut de l'objet Shutter
        String newStatus = restTemplate.getForObject(shutterURI+"status", String.class);
        msg += " *** Le statut du volet est : " + newStatus;

        return msg;
    }

}
```

Une seule requête (port 8082) :

http://localhost:8082/auto/run GET -> retourne une chaîne de caractères

Récupère différentes informations depuis les services Luminosité et Volet, via des requêtes GET et PUT.


Package Explorer
automationservice
src/main/java
fr.insa.automationservice
AutomationserviceApplication.java
fr.insa.automationservice.models
Shutter.java
fr.insa.automationservice.resources
AutomationResource.java
src/main/resources
src/test/java
JRE System Library [JavaSE-1.8]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
luminosityservice
src/main/java
fr.insa.luminosityservice
LuminosityserviceApplication.java
fr.insa.luminosityservice.resources
LuminosityResource.java
src/main/resources
src/test/java
JRE System Library [JavaSE-1.8]
Maven Dependencies
src
target
HELP.md
mvnw
mvnw.cmd
pom.xml
shutterservice
src/main/java
fr.insa.shutterservice
ShutterserviceApplication.java
fr.insa.shutterservice.model
Shutter.java
fr.insa.shutterservice.resources
ShutterResource.java
src/main/resources
src/test/java
JRE System Library [JavaSE-1.8]
Maven Dependencies
src
target
HELP.md
mvnw