

Partie 1

Étape 5

- **Spring Web** est une dépendance permettant de créer des applications web MVC.
- **Spring Data JPA** est une dépendance pour faire des requêtes SQL. Ces dernières sont en lien avec des bases de données.
- La dépendance **H2 Database** permet de créer des bases de données SQL.
- **Spring boot DevTools** fournit un ensemble d'outils/ configurations afin de développer plus « facilement ».
- **-Thymeleaf** est un Template web permettant de manipuler facilement HTML et ainsi faire des affichages propres sur navigateur.
- Hibernate est un outil utilisé pour gérer la persistance des données dans une base de données.

Étape 13

1. Avec quelle partie du code avons-nous paramétré l'url d'appel /greeting ?

La partie du code permettant de paramétrer l'url d'appel se situe dans le controller. Le code en question est le suivant : "@GetMapping("/greeting)".

2. Avec quelle partie du code avons-nous choisi le fichier HTML à afficher ?

Le code permettant de choisir le fichier HTML à afficher « return(greeting); ». Il se situe dans le controller.

3. Comment envoyons-nous le nom à qui nous disons bonjour avec le second lien ?

Afin de spécifier un nom précis, on utilise un lien type « localhost:9090/greeting ?nameGET=NOM ». Cela se déroule au niveau « @RequestParam » par le biais de notre « nameGET ». Ce dernier permet de récupérer le nom souhaité. Par la suite la fonction « model.addAttribute() » envoie cette donnée à Thymeleaf. Ainsi, Thymeleaf affiche le nom sur la page HTML ("\${nomTemplate}").

Étape 17

Suite à l'ajout de la class « Adress », une table "ADDRESS" a été créée dans la base de données (id, date ...). De plus, une répertoire « Sequences » est apparue.

Étape 18

Lors de l'initialisation de l'application, hibernate à crée une base de données (grâce à notre class). Hibernate a donc lu notre class et transformé cette dernière en une table de notre base de données. C'est le @Entity qui nous l'indique.

De plus « @Id et @GeneratedValue » indiquent que

La clef id est une clef primaire

La valeur de id peut être générée automatiquement.

Étape 20

En effectuant la requete sql suivante : « SELECT * FROM Adress ». On observe bien le contenu de data.sql (2 adresses , 2 noms et 2prénoms).

Étape 23

D'après mes recherches « @Autowired » permet de lier des éléments de l'application. Lorsque qu'on annote une méthode avec « @Autowired ». Spring va relier automatiquement l'interface et la classe. Ici, il s'agit d'une liaison entre « AddressControlle »r et « AddressRepository »

Étape 30

D'après mes recherches sur internet il faut ajouter ce code afin d'inclure Bootstrap a son code.

```
<html>
<head>
  <!-- Bootstrap CSS -->
  <link href="lib/bootstrap/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
  <!-- jQuery link -->
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></s
cript>
  <!-- Bootstrap JS -->
  <script src="lib/bootstrap/js/bootstrap.min.js"></script>
</body>
</html>
```

Ps: je ne l'ai pas ajouter a mon projet mais il faudrait l'ajouter sur les différentes pages html.

Partie 2

Étape 6

Faut-il une clé API pour appeler OpenWeatherMap ?

D'après mes recherches il faut une clé API pour appeler « OpenWeatherMap ».

Quelle URL appeler ?

<http://api.openweathermap.org/data/2.5/weather?lat=<<para1>>&lon=<<para2>>&units=metric&appid=<cléAPI>>

Quelle méthode HTTP utiliser ?

L'utilisation de la méthode GET est nécessaire pour utiliser l'API

Comment passer les paramètres d'appels ?

Le passage des paramètres se fait en mettant les données dans l'URL juste après "lat=X1" et "lon=X2". Ici X1 et X2 sont les paramètres.

Où est l'information dont j'ai besoin dans la réponse :

Pour afficher la température du lieu visé par les coordonnées GPS

La température du lieu visé se trouve juste après "temp" (dans le json).

Pour afficher la prévision de météo du lieu visé par les coordonnées GPS

Afin d'afficher la prévision, j'ai trouvé les liens suivants. Il faut utiliser l'un d'entre eux.

<http://api.openweathermap.org/data/2.5/forecast/daily?lat=<<para1>>&lon=<<para2>>&appid=<cléAPI>>

<http://api.openweathermap.org/data/2.5/forecast/daily?q={city name}&cnt={cnt}&appid={API key}>.