

Angular :

Déploiement et Test d'une application Backend api REST

Objectifs :

L'objectif de cet atelier est de vous apprendre comment :

- Déployer et démarrer l'application backend fournissant les webservice (api) REST qui seront consommés à partire de la partie frontend Angular. Le fichier .jar (executable java) de l'application Backend vous est fourni dans la description de la video Youtube.
- Tester les api REST avec POSTMAN

Prérequis :

- La machine doit contenir java 8 et Mysql (xampp ou wamp ou autre)
 - Ma video comment installer Java 8 : https://youtu.be/38zFn0_9TK0
- Pour cet atelier j'ai installé XAMPP Version: 7.4.2 (MySQL version 10.4.11-MariaDB)
 - Ma video comment installer xampp : https://youtu.be/bub_2KwCizo
- Installer POSTMAN, outil permettant de tester les api REST
 - Ma video comment télécharger et installer POSTMAN : https://youtu.be/KfHIY48_Dh0

Déployer et démarrer l'application Backend (fichier .jar)

1. Démarrer **MySql** de xampp (ou MySQL satndalone)
2. Lancer une fenêtre invite de commande (cmd) placez-vous dans le dossier contenant le fichier api_rest_produits.jar, puis lancer :

puis

java -jar api_rest_produits.jar

Réduisez la fenêtre cmd (ne la fermez pas pour ne pas arrêter l'application Backend)

Si jamais cette commande ne marche pas car le port 8080 est occupé par Oracle XE, alors Il faut changer le port par défaut de tomcat 8080 lors du lancement de l'appli api_restproduits.jar il faut utiliser l'option -Dserver.port comme suit:

java -Dserver.port=8090 -jar api_rest_produits.jar

et dans votre projet Angular (dans les prochains devoirs) il faut modifier le port 8080 par 8090 dans l'url des api

Tester avec POSTMAN les Web Service (api) REST

Tester le Web service REST permettant de retourner tous les produits

3. Tester avec POSTMAN le web service REST, permettant de retourner tous les produits : <http://localhost:8080/produits/api>

The screenshot shows the Postman application window. At the top, there are tabs for NEW, Runner, Import, and Builder, with 'Builder' selected. Below the tabs, a message bar indicates that Chrome apps are being deprecated and suggests downloading native apps. The main interface shows a 'GET' request method selected, with the URL 'http://localhost:8080/produits/api'. To the right of the URL are buttons for 'Params', 'Send', and 'Save'. Below the URL, there are tabs for 'Authorization', 'Headers (1)', 'Body', 'Pre-request Script', and 'Tests', with 'Authorization' selected. Under 'Authorization', it says 'Type: No Auth'. The 'Body' tab is selected, showing a JSON response. The response status is 'Status: 200 OK' and the time taken is 'Time: 2746 ms'. The response body is displayed in 'Pretty' JSON format, showing a list of products:

```
1 * [
2 *   {
3 *     "idProduit": 1,
4 *     "nomProduit": "PC Asus N7",
5 *     "prixProduit": 2000,
6 *     "dateCreation": "2020-10-17T23:00:00.000+00:00",
7 *     "categorie": {
8 *       "idCat": 1,
9 *       "nomCat": "PC",
10 *      "descriptionCat": "Les PCs"
11 *    }
12 *  },
13 *  {
14 *    "idProduit": 2,
15 *    "nomProduit": "PS 4",
16 *    "prixProduit": 500,
17 *    "dateCreation": "2011-10-08T23:00:00.000+00:00",
18 *    "categorie": {
```

Tester le Web service REST permettant de consulter un produit

4. Tester avec POSTMAN le web service REST, permettant de consulter un produit en fournissant son id : <http://localhost:8080/produits/api/2>

The screenshot shows the Postman application window. In the center, there's a 'Launchpad' section with a single 'Untitled Request' card. The card shows a 'GET' method pointing to 'http://localhost:8080/produits/api/2'. Below the method, under 'Params', there are no parameters listed. The 'Body' tab is selected, showing a JSON response:

```
1  {
2      "idProduit": 2,
3      "nomProduit": "iphone X",
4      "prixProduit": 1810.123,
5      "categorie": {
6          "idCat": 1,
7          "nomCat": "Smartphones",
8          "descriptionCat": "les smartphones"
9      }
10 }
```

At the bottom of the interface, there are tabs for 'Pretty', 'Raw', 'Preview', and 'Visualize', with 'JSON' currently selected. Above the preview area, status information is displayed: Status: 200 OK, Time: 27 ms, Size: 397 B, and Save Response.

Tester le Web service REST permettant d'ajouter un produit

5. Tester avec POSTMAN le web service REST, permettant d'ajouter un produit : <http://localhost:8080/produits/api>

- Choisissez la méthode POST
- Dans l'onglet Body, cliquez sur raw, puis entrer un produit au format JSON :

```
{ "nomProduit": "tablette Samsung Notes",
  "prixProduit": 1980
}
```

The screenshot shows the Postman application window for a POST request to 'http://localhost:8080/produits/api'. The 'Body' tab is highlighted with a red box, and the dropdown menu next to it is also highlighted with a red box, showing 'JSON' is selected. The JSON payload is identical to the one shown in the previous screenshot:

```
1  {
2      "nomProduit": "tablette Samsung Notes",
3      "prixProduit": 1980
4  }
```

Tester le Web service REST permettant de modifier un produit

6. Tester avec POSTMAN le web service REST permettant de modifier un produit : <http://localhost:8080/produits/api/>

The screenshot shows the Postman application window. At the top, there are tabs for 'NEW', 'Runner', 'Import', 'Builder' (which is selected), and 'Team Library'. A message bar indicates that Chrome apps are being deprecated. Below the tabs, there's a toolbar with icons for http, offline status, and settings. The main area shows a 'PUT' request to 'http://localhost:8080/produits/api/'. The 'Body' tab is selected, showing a JSON payload:

```
1 {  
2     "idProduit": 6,  
3     "nomProduit": "Tablette Samsung Notes",  
4     "prixProduit": 2500  
5 }  
6  
7 }
```

Below the body, the status bar shows 'Status: 200 OK' and 'Time: 97 ms'. The bottom navigation bar includes 'Pretty', 'Raw', 'Preview', and 'JSON' buttons.

Tester le Web service REST permettant de supprimer un produit

7. Tester avec POSTMAN le web service REST, permettant de supprimer un produit: <http://localhost:8080/produits/api/5>

The screenshot shows the Postman application window. The 'History' tab is selected in the sidebar. It lists several requests: a DELETE request to 'http://localhost:8080/produits/api/5' (selected), a POST request to 'http://localhost:8080/produits/api/' (highlighted in yellow), a POST request to 'http://localhost:8080/produits/api/' (highlighted in orange), a GET request to 'http://localhost:8080/produits/api/2', a GET request to 'http://localhost:8080/produits/api/2', a GET request to 'http://localhost:8080/produits/api/2', and two GET requests to 'http://localhost:8080/produits/api/2'. The main panel shows an 'Untitled Request' with a 'DELETE' method and the URL 'http://localhost:8080/produits/api/5'. The 'Body' tab is selected, showing the message 'This request does not have a body'. The status bar at the bottom shows '200 OK', '292 ms', and '212 B'. The bottom navigation bar includes 'Pretty', 'Raw', 'Preview', 'Visualize', 'Text', and 'JSON' buttons.