

Atelier REST n°1**Création et déploiement d'un service web
HelloWorld avec JAX-RS****Objectifs**

Le but de cet atelier est le développement d'un service web RESTful HelloWorld avec l'API JAX-RS.

Définition d'une ressource REST avec JAX-RS

- 1- Dans Eclipse, créez un nouveau projet **maven** nommé « Hello_JAX-RS » de packaging **war**.

```
<properties>
  <failOnMissingWebXml>false</failOnMissingWebXml>
  <project.buil.sourceEncoding>utf-8</project.buil.sourceEncoding>
  <maven.compiler.target>1.8</maven.compiler.target>
  <maven.compiler.source>1.8</maven.compiler.source>
</properties>
<dependencies>
  <dependency>
    <groupId>org.glassfish.jersey.core</groupId>
    <artifactId>jersey-server</artifactId>
    <version>2.27</version>
  </dependency>
  <dependency>
    <groupId>org.glassfish.jersey.containers</groupId>
    <artifactId>jersey-container-servlet</artifactId>
    <version>2.27</version>
  </dependency>
  <dependency>
    <groupId>org.glassfish.jersey.inject</groupId>
    <artifactId>jersey-hk2</artifactId>
    <version>2.27</version>
  </dependency>
</dependencies>
```

- 2- Dans pom.xml, ajoutez le code suivant:

```

@Path("/greetings")
public class HelloResource {
    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String sayHello()
    {
        return "Hello from JAX-RS";
    }
}

```

- 3- Cliquez sur Maven Update Project.
- 4- Créez une classe nommée «HelloResource» dans le package «rest.resources». Cette classe représente notre ressource REST et doit contenir des méthodes publiques.
- 5- Ajoutez les annotations JAX-RS suivantes :
 - `@Path("/greetings")` permet de définir un chemin pour associer la ressource HelloResource à un URI.
 - `@GET`: pour la lecture de la ressource via une requête HTTP de type GET.
 - `@Produces(MediaType.TEXT_PLAIN)`: indique le type de la représentation retournée par la requête GET.

```

package tn.esprit.rest;

import javax.ws.rs.ApplicationPath;

@ApplicationPath("rest")
public class RestActivator extends Application{

}

```

Configuration de l'application Web JAX-RS

Une application JAX-RS doit contenir au moins une classe ressource packagée dans une archive WAR. Afin d'activer les ressources REST définies dans votre application, vous devez ajouter une sous-classes `javax.ws.rs.core.Application`, puis ajouter les modèles d'URI requis à l'aide de l'annotation `@ApplicationPath`.

Créez une classe nommée «RestActivator» dans le package «rest.utilities» comme suit:

NB: Le chemin défini dans l'annotation `javax.ws.rs.ApplicationPath` sera ajouté à l'URL de base.

Déploiement et test du service web RESTFUL

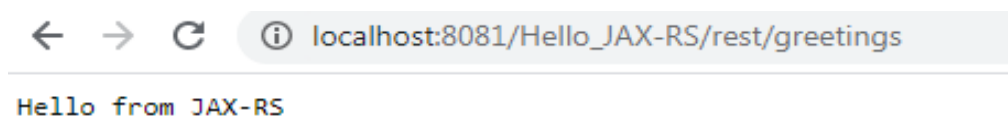
- 1- Déployez le projet.
- 2- Test du service web RESTful

En tapant l'uri **`http://localhost:port-server/Hello_JAX-RS/rest/greetings`**, on envoie une requête HTTP de type GET demandant la lecture de la ressource `HelloResource`.

numéro du port du serveur contextroot « URL base » configuré dans `@ApplicationPath`

`http://localhost:8383/Hello_JAX-RS/rest/greetings`

chemin ressource



Création d'un service web RESTful paramétré

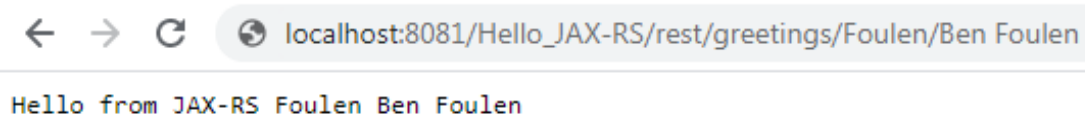
Maintenant on va changer le comportement de notre service web RESTful pour qu'il utilise des paramètres.

a- @PathParam

- L'annotation **@PathParam** récupère des arguments passés dans le chemin de l'URI.

```
@GET
@Produces(MediaType.TEXT_PLAIN)
public String sayHelloTo2(@QueryParam(value="FirstName")String prenom,@QueryParam(value="LastName")String nom)
{
    return "Hello from JAX-RS "+prenom+" "+nom;
}
```

Testez comme suit:



← → ↻ 🌐 localhost:8081/Hello_JAX-RS/rest/greetings/Foulen/Ben Foulen

Hello from JAX-RS Foulen Ben Foulen

b- @QueryParam

- L'annotation **@QueryParam** récupère les valeurs des paramètres de la requête.

Testez comme suit :

```
@GET
@Path("/{FirstName}/{LastName}")
@Produces(MediaType.TEXT_PLAIN)
public String sayHelloTo1(@PathParam(value="FirstName")String prenom,@PathParam(value="LastName")String nom)
{
    return "Hello from JAX-RS "+prenom+" "+nom;
}
```

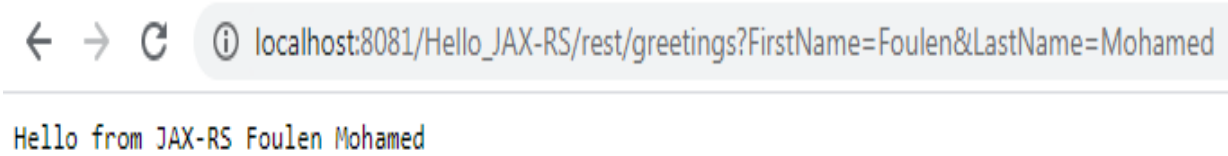


← → ↻ ⓘ localhost:8081/Hello_JAX-RS/rest/greetings?FirstName=Foulen&LastName=Mohamed

Que remarquez-vous?

- Testez maintenant avec le code suivant qui remplacera les deux ressources *sayHello* et *sayHello2* ci-dessus:

Voici le résultat:



```
@GET
@Produces(MediaType.TEXT_PLAIN)
public String sayHello(@QueryParam(value="FirstName")String prenom,@QueryParam(value="LastName")String nom)
{
    if(prenom==null && nom==null)
    {
        return "Hello from JAX-RS";
    }
    return "Hello from JAX-RS "+prenom+" "+nom;
}
```