

README

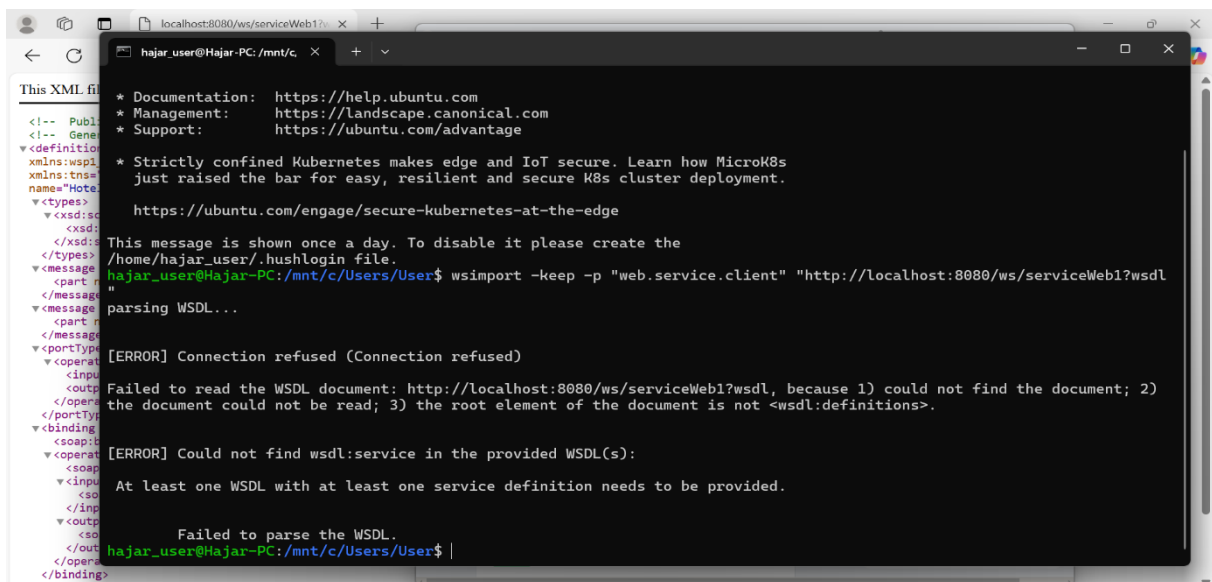
La Version Distribuée - Partie Client

Dans cette phase du projet, j'ai créé un nouveau package pour **le client**. L'objectif était de générer le code nécessaire pour interagir avec les services Web. Pour cela, j'ai utilisé la commande suivant dans le terminal :

```
wsimport -keep -p "web.service.client" "http://localhost:8080/ws/serviceWeb1?wsdl"
```

Cela a permis de compiler et générer les classes nécessaires. Cependant, j'ai rencontré un problème avec l'utilisation de localhost dans la commande, comme l'avait montré **Rima Bachar** dans sa vidéo. Après quelques recherches, j'ai découvert que l'on pouvait utiliser **l'adresse IP au lieu de localhost**, et cette méthode a fonctionné correctement pour moi.

Photo 1 : Cette photo montre la commande avec **localhost** et l'erreur que nous avons rencontrée lors de la dernière séance. La commande `wsimport -keep -p` a échoué en raison de l'impossibilité d'accéder au WSDL via localhost.



Lien vidéo YouTube : <https://youtu.be/IZTarWzFer4?si=FWCRnjAbLUVKOWKr>

Photo 2 : Cette photo montre le succès de la commande avec l'adresse IP de mon ordinateur, ce qui a permis de contourner l'erreur précédente. La compilation a été réalisée avec succès et le code a été généré correctement. Cela a permis de créer les classes nécessaires pour interagir avec le service Web

wsimport -keep -p "web.service.client" "http://172.29.240.1:8080/ws/serviceWeb1?wsdl"

```
Invite de commandes x hajar_user@Hajar-PC: /mnt/c/ x hajar_user@Hajar-PC: /mnt/c/ x + v

Generating code...

Compiling code...

hajar_user@Hajar-PC: /mnt/c/Users/User/desktop/temp_client$
hajar_user@Hajar-PC: /mnt/c/Users/User/desktop/temp_client$ wsimport -keep -p "web.service.client" "http://172.29.240.1:8080/ws/serviceWeb2?wsdl"
parsing WSDL...

Generating code...

Compiling code...

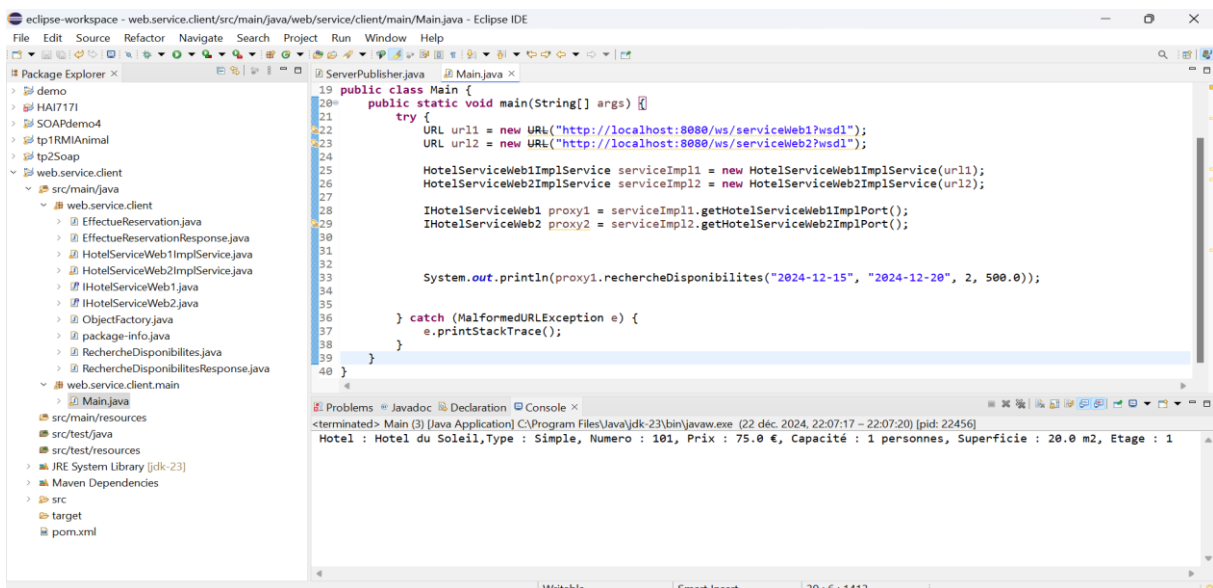
hajar_user@Hajar-PC: /mnt/c/Users/User/desktop/temp_client$ wsimport -keep -p "web.service.client" "http://172.29.240.1:8080/ws/serviceWeb1?wsdl"
parsing WSDL...

Generating code...

Compiling code...

hajar_user@Hajar-PC: /mnt/c/Users/User/desktop/temp_client$
```

Une fois la commande exécutée avec succès, j'ai généré les fichiers .java et .class. En suivant les étapes de la vidéo, j'ai copié les fichiers .java dans mon projet. Par la suite, j'ai créé une classe Main pour instancier les objets et le proxy nécessaire à l'interaction avec les services Web.



```
eclipse-workspace - web.service.client/src/main/java/web/service/client/main/Main.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer x
> demo
> HAI7171
> SOAPdemo4
> tp1RMAnimal
> tp2Soap
> web.service.client
  > src/main/java
    > web.service.client
      > EffectueReservation.java
      > EffectueReservationResponse.java
      > HotelServiceWeb1ImplService.java
      > HotelServiceWeb2ImplService.java
      > IHotelServiceWeb1.java
      > IHotelServiceWeb2.java
      > ObjectFactory.java
      > package-info.java
      > RechercheDisponibilites.java
      > RechercheDisponibilitesResponse.java
    > web.service.client.main
      > Main.java
    > src/main/resources
    > src/test/java
    > src/test/resources
  > JRE System Library [jdk-23]
  > Maven Dependencies
  > src
  > target
  > pom.xml

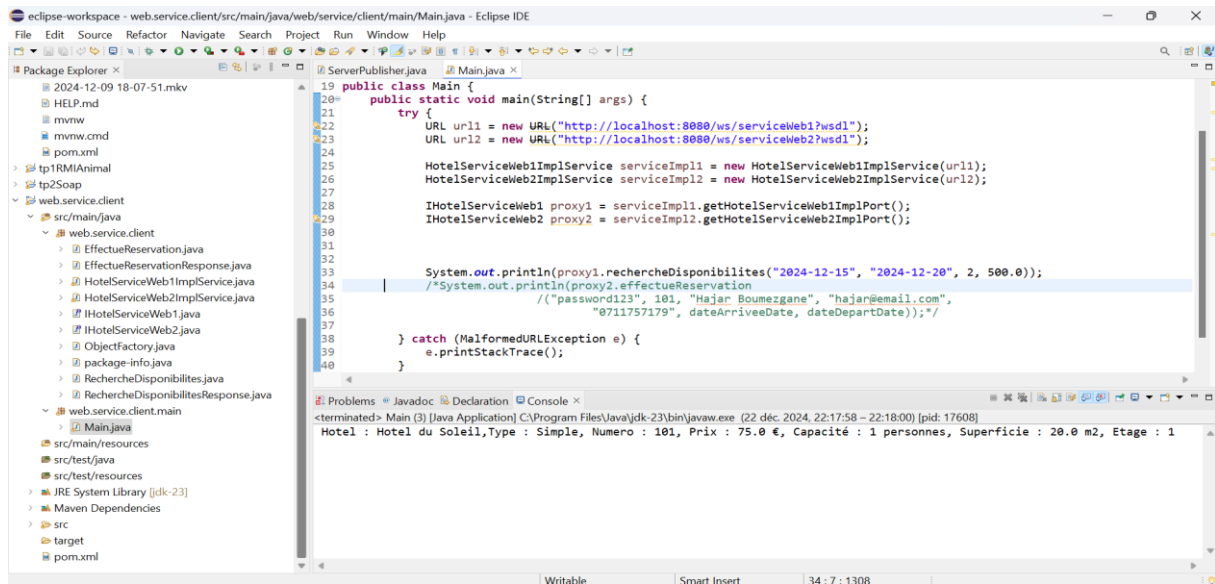
ServerPublisher.java Main.java x
19 public class Main {
20     public static void main(String[] args) {
21         try {
22             URL url1 = new URL("http://localhost:8080/ws/serviceWeb1?wsdl");
23             URL url2 = new URL("http://localhost:8080/ws/serviceWeb2?wsdl");
24
25             HotelServiceWeb1ImplService serviceImpl1 = new HotelServiceWeb1ImplService(url1);
26             HotelServiceWeb2ImplService serviceImpl2 = new HotelServiceWeb2ImplService(url2);
27
28             IHotelServiceWeb1 proxy1 = serviceImpl1.getHotelServiceWeb1ImplPort();
29             IHotelServiceWeb2 proxy2 = serviceImpl2.getHotelServiceWeb2ImplPort();
30
31
32             System.out.println(proxy1.rechercheDisponibilites("2024-12-15", "2024-12-20", 2, 500.0));
33
34
35         } catch (MalformedURLException e) {
36             e.printStackTrace();
37         }
38     }
39 }
40 }

Problems Javadoc Declaration Console x
<terminated> Main (3) [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe (22 déc. 2024, 22:07:17 - 22:07:20) [pid: 22456]
Hotel : Hotel du Soleil, Type : Simple, Numero : 101, Prix : 75.0 €, Capacité : 1 personnes, Superficie : 20.0 m2, Etage : 1
```

Remarque :

Pour le client, j'ai réussi à implémenter la méthode **rechercheDisponibilites** et à afficher les résultats dans la console. Nous pouvons même ajuster les paramètres en fonction de nos recherches. Cependant pour la méthode **effectueReservation**, j'ai rencontré un problème avec le traitement des dates. Je n'ai pas pu faire fonctionner l'algorithme de gestion des dates, donc j'ai commenté la méthode **effectueReservation**.

Il me semble que la solution consiste à transformer les dates en String, comme c'est le cas dans ma méthode **rechercheDisponibilites**, où les dates sont traitées sous forme de String. J'ai essayé cette modification, mais elle génère des erreurs dans ma version du serveur, qui gère plusieurs dates.



```
19 public class Main {
20     public static void main(String[] args) {
21         try {
22             URL url1 = new URL("http://localhost:8080/ws/serviceWeb1?wsdl");
23             URL url2 = new URL("http://localhost:8080/ws/serviceWeb2?wsdl");
24
25             HotelServiceWeb1ImplService serviceImpl1 = new HotelServiceWeb1ImplService(url1);
26             HotelServiceWeb2ImplService serviceImpl2 = new HotelServiceWeb2ImplService(url2);
27
28             IHotelServiceWeb1 proxy1 = serviceImpl1.getHotelServiceWeb1ImplPort();
29             IHotelServiceWeb2 proxy2 = serviceImpl2.getHotelServiceWeb2ImplPort();
30
31
32             System.out.println(proxy1.rechercheDisponibilites("2024-12-15", "2024-12-20", 2, 500.0));
33             /*System.out.println(proxy2.effectueReservation
34              /*("password123", 101, "Hajar Boumezgane", "hajar@email.com",
35               "0711757179", dateArriveeDate, dateDepartDate));*/
36
37         } catch (MalformedURLException e) {
38             e.printStackTrace();
39         }
40     }
41 }
```

Console Output:

```
<terminated> Main (3) [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe (22 déc. 2024, 22:17:58 - 22:18:00) [pid: 17608]
Hotel : Hotel du Soleil, Type : Simple, Numero : 101, Prix : 75.0 €, Capacité : 1 personnes, Superficie : 20.0 m2, Etage : 1
```

Remarque : Pour exécuter le client, il faut d'abord lancer le serveur.

Hajar Boumezgane