

Creare un server SOAP in Java

SOAP (Simple Object Access Protocol) è un protocollo basato su XML che permette lo scambio di informazioni strutturate tra sistemi distribuiti.

Sebbene RESTful sia diventato molto popolare, SOAP rimane una scelta valida per applicazioni enterprise che richiedono sicurezza, transazioni e interoperabilità. In questo articolo, vedremo come creare un server SOAP utilizzando Java e la libreria JAX-WS (Java API for XML Web Services).

Per questa guida useremo Maven per gestire le dipendenze. Modifica il file `pom.xml` per includere le dipendenze necessarie:

```
<dependencies>
  <dependency>
    <groupId>javax.xml.ws</groupId>
    <artifactId>jaxws-api</artifactId>
    <version>2.3.1</version>
  </dependency>
  <dependency>
    <groupId>com.sun.xml.ws</groupId>
    <artifactId>jaxws-rt</artifactId>
    <version>2.3.1</version>
  </dependency>
</dependencies>
```

Creiamo una semplice interfaccia Java che rappresenta il nostro servizio web.

Crea un nuovo package, ad esempio `com.example.soap`. All'interno del package, crea una nuova interfaccia chiamata `HelloWorld`:

```
package com.example.soap;

import javax.jws.WebMethod;
import javax.jws.WebService;

@WebService
public interface HelloWorld {
    @WebMethod
    String sayHello(String name);
}
```

Ora, implementiamo l'interfaccia in una classe. Crea una nuova classe chiamata HelloWorldImpl:

```
package com.example.soap;

import javax.jws.WebService;

@WebService(endpointInterface =
"com.example.soap.HelloWorld")
public class HelloWorldImpl implements HelloWorld {

    @Override
    public String sayHello(String name) {
        return "Hello, " + name + "!";
    }
}
```

Ora che abbiamo la nostra implementazione, dobbiamo pubblicare il servizio affinché sia accessibile ai client. Crea una classe `Server` che avvierà il servizio:

```
package com.example.soap;

import javax.xml.ws.Endpoint;

public class Server {

    public static void main(String[] args) {

        Endpoint.publish("http://localhost:8080/ws/hello",
            new HelloWorldImpl());
        System.out.println("Server is running at
http://localhost:8080/ws/hello");
    }
}
```

Per testare il servizio, possiamo utilizzare un client SOAP come SoapUI, Postman, o creare un semplice client Java.

```
package com.example.soap;

import javax.xml.namespace.QName;
import javax.xml.ws.Service;
import java.net.URL;

public class Client {
```

```
        public static void main(String[] args) throws
Exception {
            URL url = new
URL("http://localhost:8080/ws/hello?wsdl");
            QName qname = new
QName("http://soap.example.com/",
"HelloWorldImplService");
            Service service = Service.create(url, qname);
            HelloWorld hello =
service.getPort(HelloWorld.class);

            System.out.println(hello.sayHello("World"));
        }
    }
```

Conclusione

Abbiamo creato un semplice server SOAP utilizzando Java e JAX-WS. Abbiamo definito un'interfaccia del servizio, implementato la logica aziendale e pubblicato il servizio affinché fosse accessibile ai client. Questo esempio rappresenta solo l'inizio; i servizi SOAP possono essere molto più complessi e includere funzionalità avanzate come la sicurezza, la gestione delle transazioni e molto altro. Tuttavia, con le basi apprese, sei pronto per esplorare e implementare servizi SOAP più sofisticati nelle tue applicazioni.