Rozproszone systemy internetowe

RESTful web services, JAX-RS

Zagadnienia:

1. Tworzenie RESTful serwisu, zwracanie listy obiektów
2. WADL
3. Tworzenie klienta (JAX-RS Client)
4. Przekazywanie obiektu jako parametr dla serwisu

Sources and help:

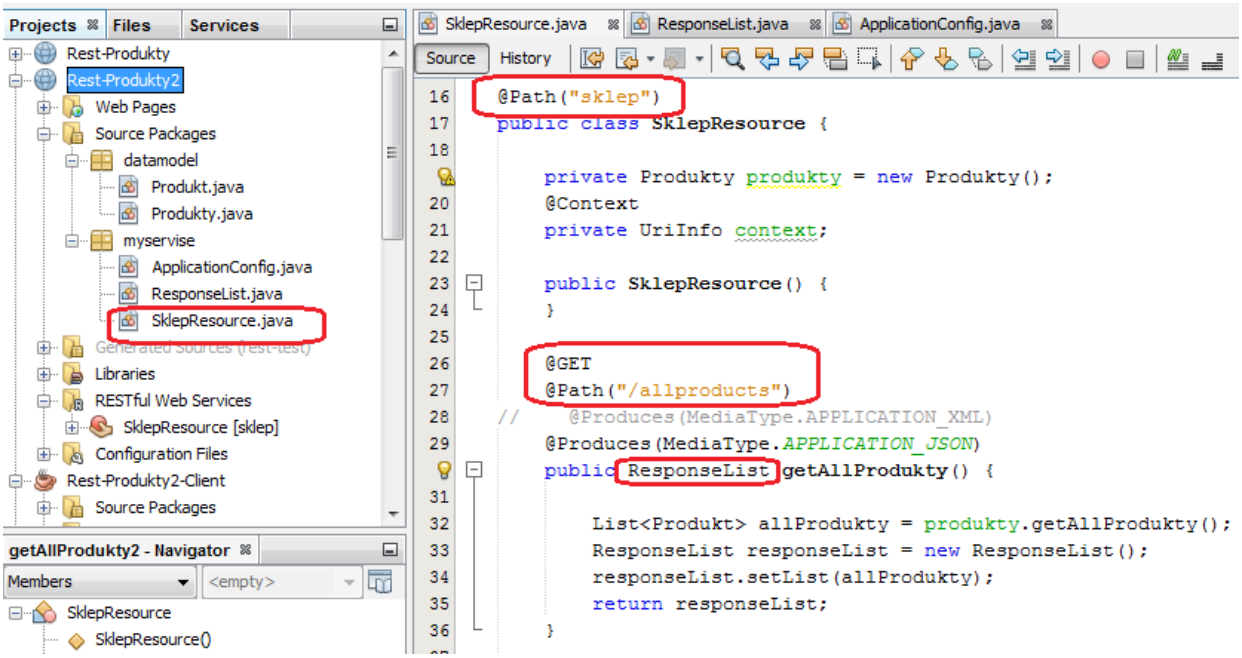
What are RESTful Web Services (ORACLE)

Jersey documentation - 5. Client API

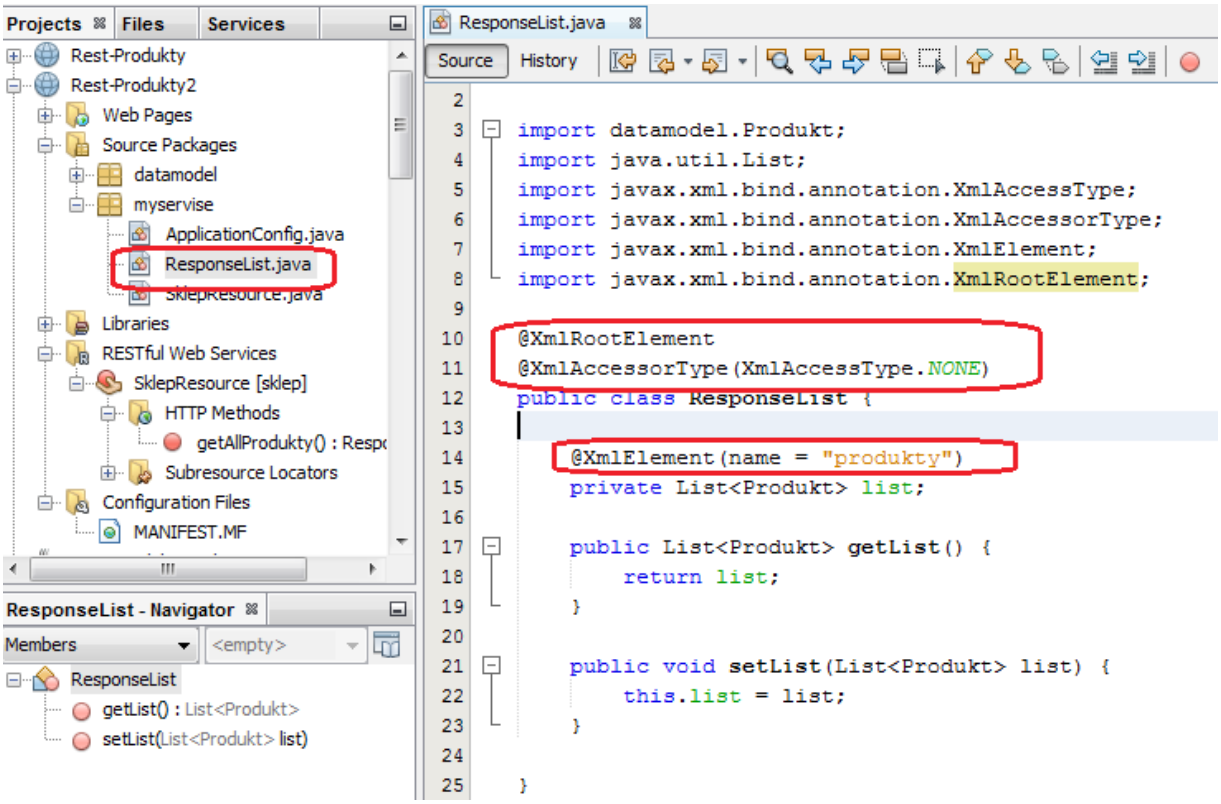
Jersey User Guide

Ćwiczenie 1. Tworzenie serwisu zwracającego listę produktów w JSON

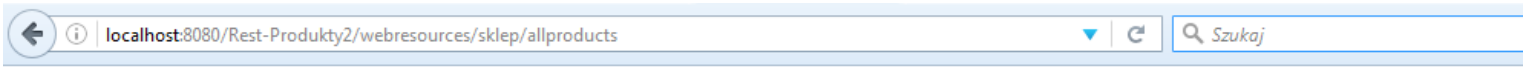
Utwórz RESTful serwis zwracający listę produktów



Obiekt odpowiedzi ResponseList



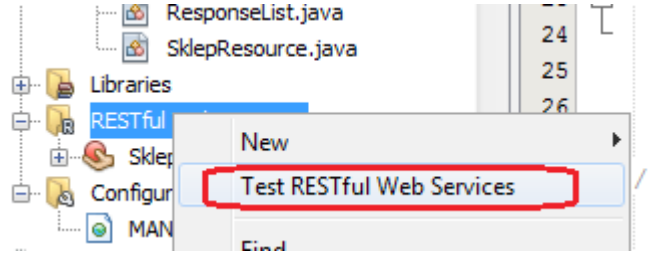
Testowanie serwisu w przeglądarce

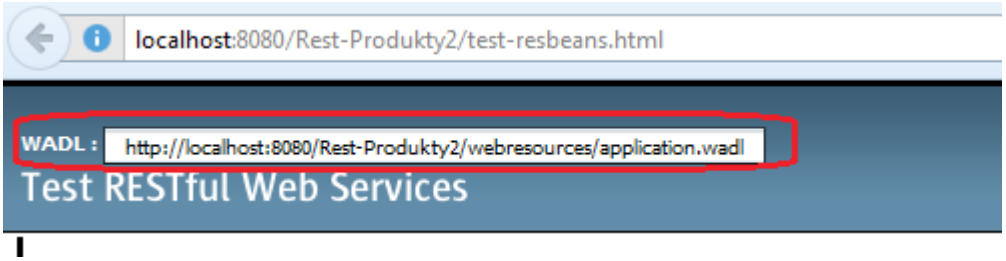


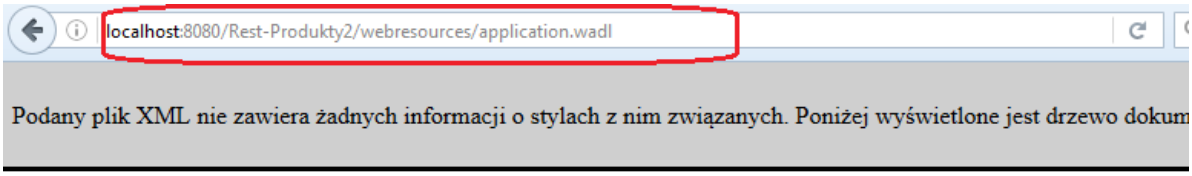

```
{"produkty":[{"cena":2000,"nazwa":"Laptop","producent":"Dell"},{"cena":50,"nazwa":"Myszka","producent":"Logitech"},
{"cena":1500,"nazwa":"Monitor","producent":"Dell"},{"cena":100,"nazwa":"Torba","producent":"Dell"},{"cena":170,"nazwa":"Tab
{"cena":2500,"nazwa":"Laptop","producent":"IBM"}]}
```


Ćwiczenie 2. Podgląd pliku WADL dla serwisu

Przejrzyj zawartość pliku WADL








```
- <application>
    <doc jersey:generatedBy="Jersey: 2.10.4 2014-08-08 15:09:00"/>
    <doc jersey:hint="This is simplified WADL with user and core resources only. To get full WADL with extended re
    Produkty2/webresources/application.wadl?detail=true"/>
  - <grammars>
    - <include href="application.wadl/xsd0.xsd">
        <doc title="Generated" xml:lang="en"/>
      </include>
    </grammars>
  - <resources base="http://localhost:8080/Rest-Produkty2/webresources/">
    - <resource path="sklep">
      - <resource path="/allproducts">
        - <method id="getAllProdukty" name="GET">
          - <response>
              <ns2:representation element="responseList" mediaType="application/json"/>
            </response>
          </method>
        </resource>
      </resource>
    </resources>
```

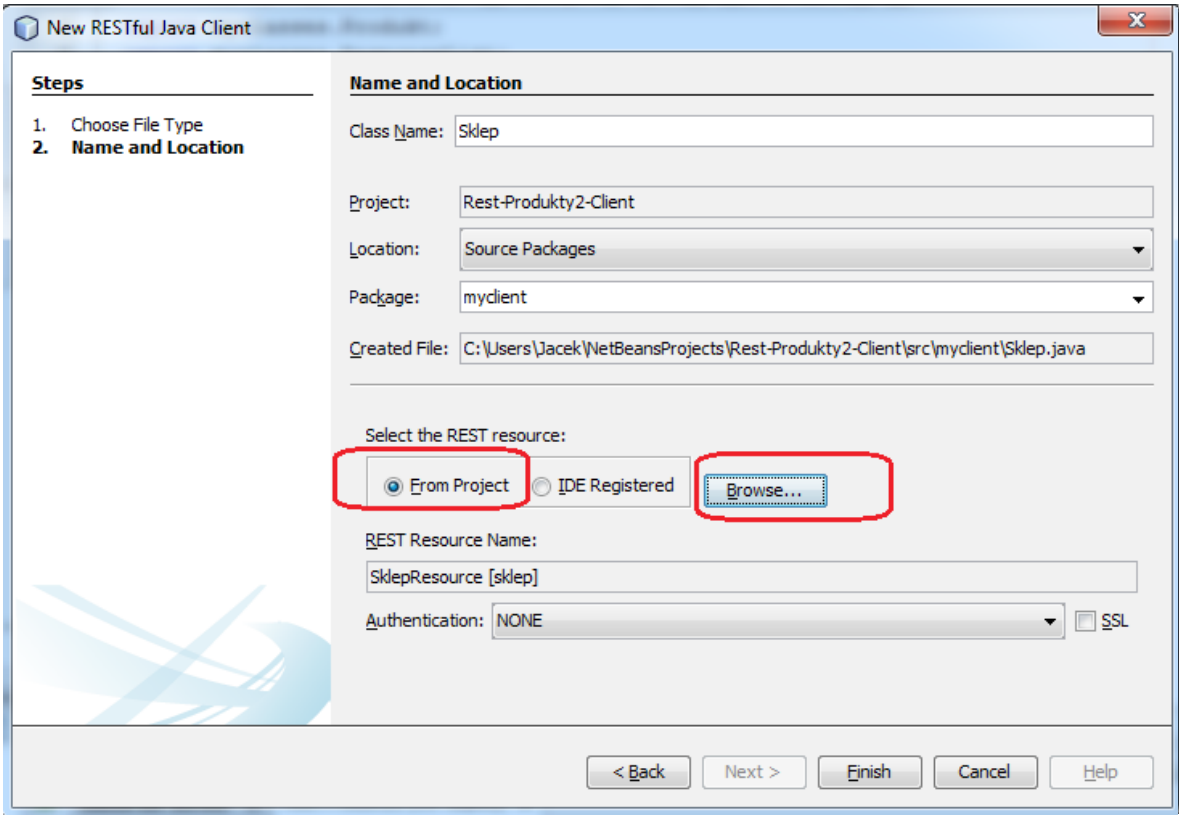
XML Schema


```
- <xs:schema version="1.0">
    <xs:element name="produkt" type="produkt"/>
    <xs:element name="responseList" type="responseList"/>
  - <xs:complexType name="responseList">
    - <xs:sequence>
        <xs:element name="list" type="produkt" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  - <xs:complexType name="produkt">
    - <xs:sequence>
        <xs:element name="cena" type="xs:int"/>
        <xs:element name="nazwa" type="xs:string" minOccurs="0"/>
        <xs:element name="producent" type="xs:string" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:schema>
```

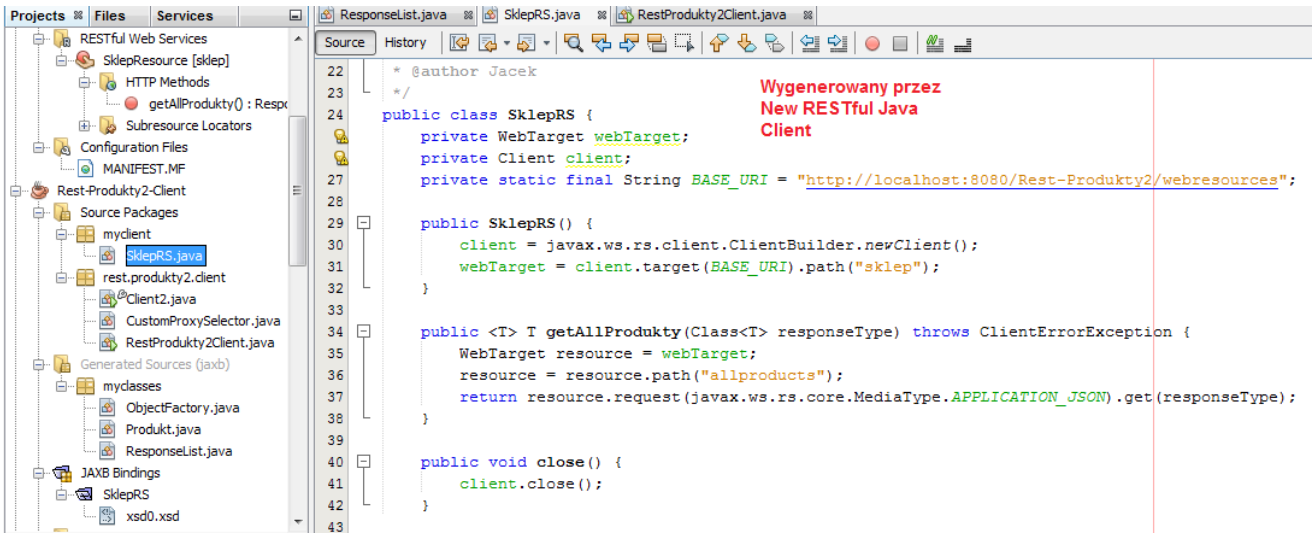
Ćwiczenie 3. Tworzenie klienta dla serwisu (Jersey Client)

Utwórz klienta dla serwisu zwracającego produkty

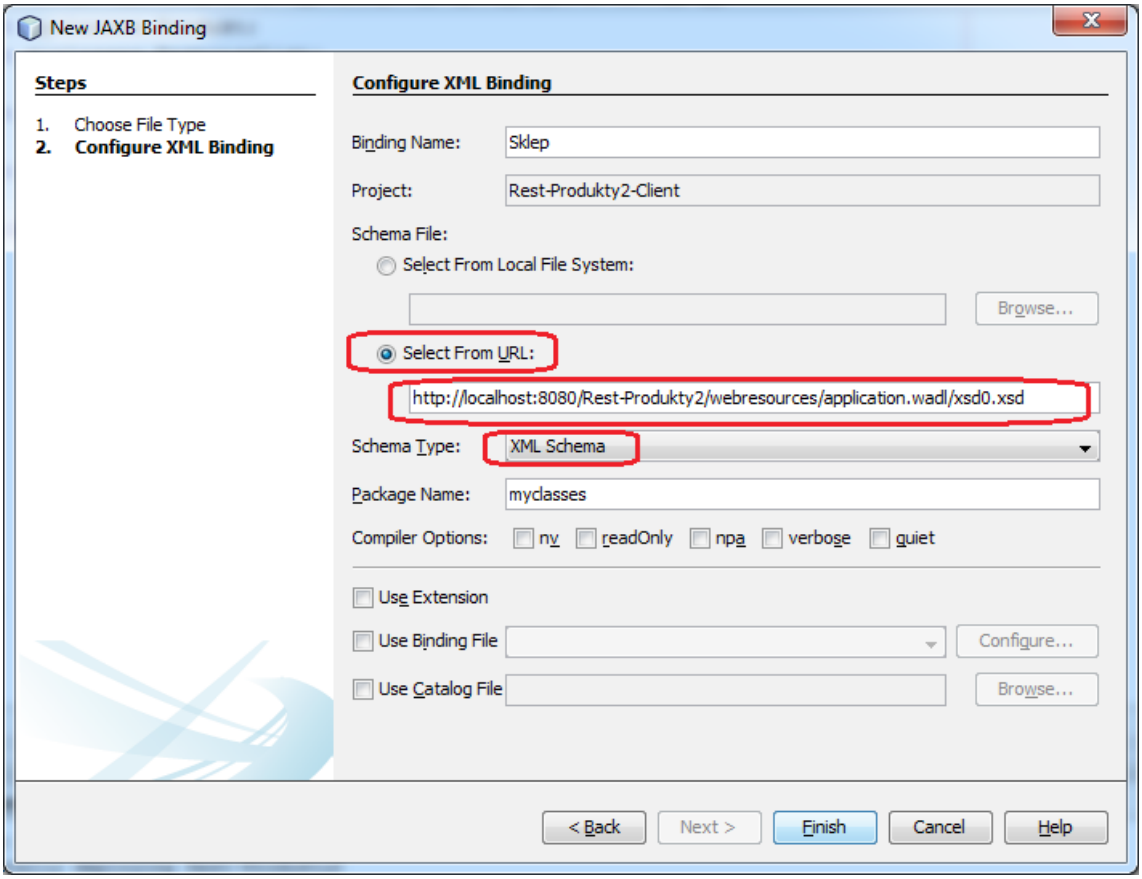
1. Tworzenie klineta „New RESTful Java Client” wizard



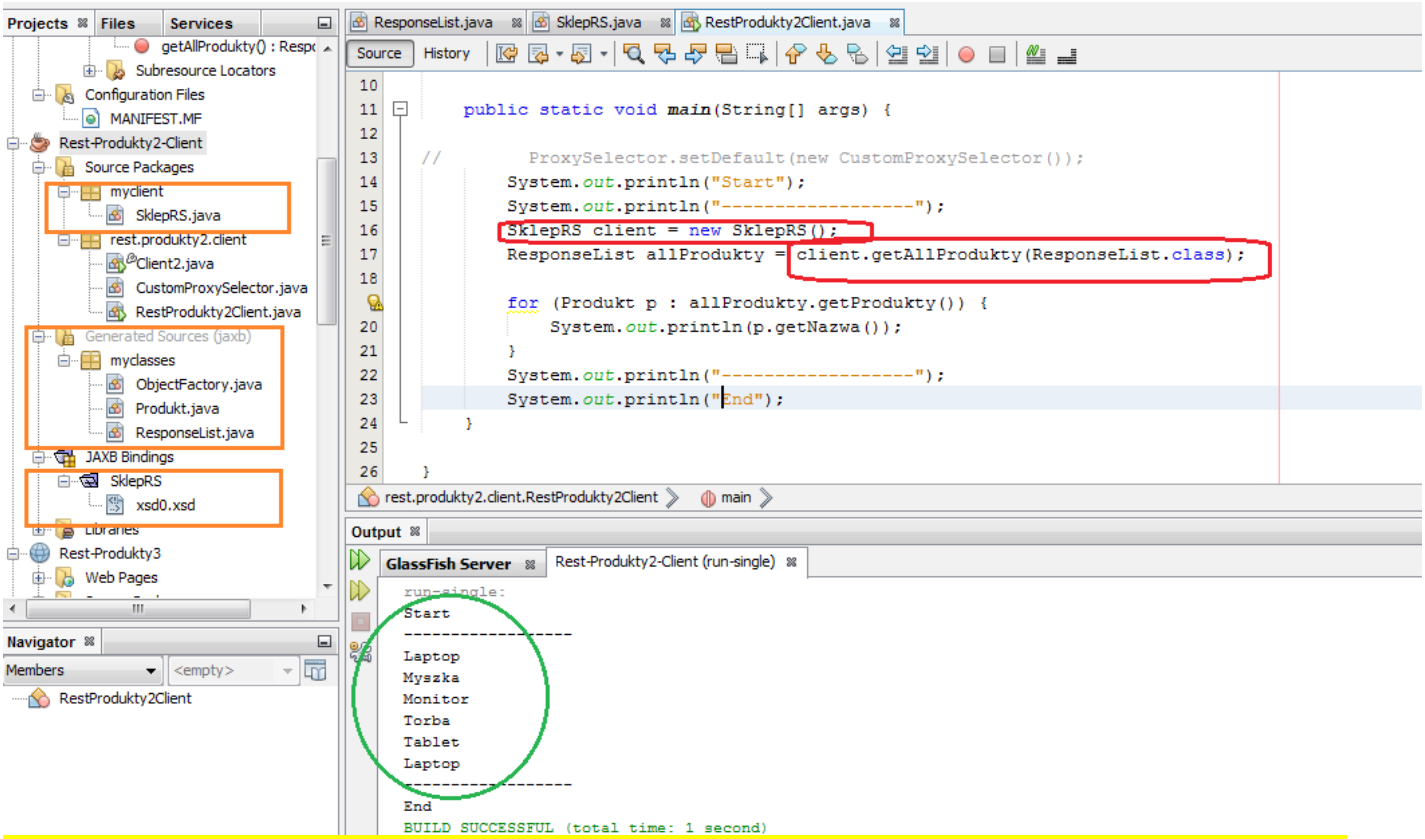
Przykładowa treść wygenerowanego klienta



2. Użycie JAXB Binding do utworzenia potrzebnych klas (np. ResponseList.java) w kliencie

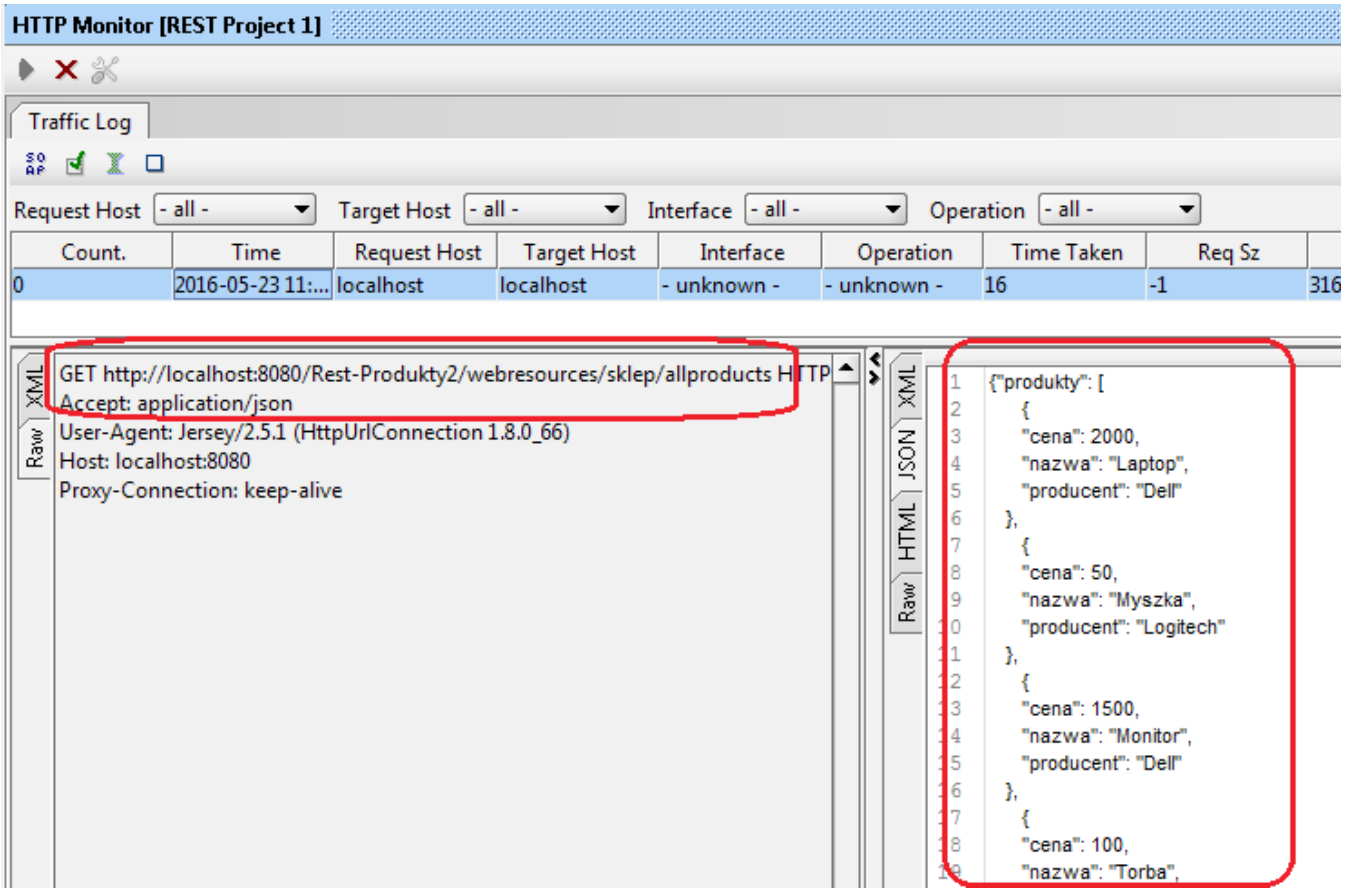


3. Użycie klienta do pobrania listy produktów



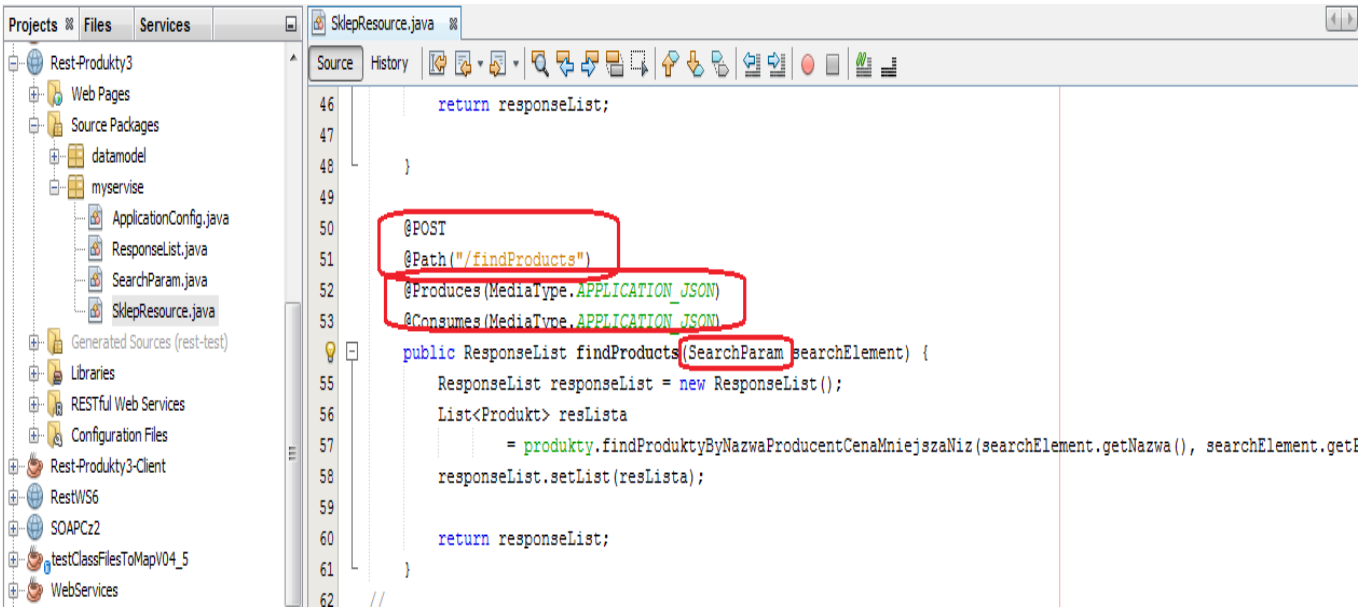
Ktoś ze studentów zgłaszał, że był problem ze zwracanym obiektem, był pusty (null), przy Glassfish 5.0. Przy Glassfish 4.1 działa. Rozwiązaniem się okazało ustawienie @Produces(MediaType.APPLICATION_XML) po stronie seriwsu zamiast @Produces(MediaType.APPLICATION_JSON).

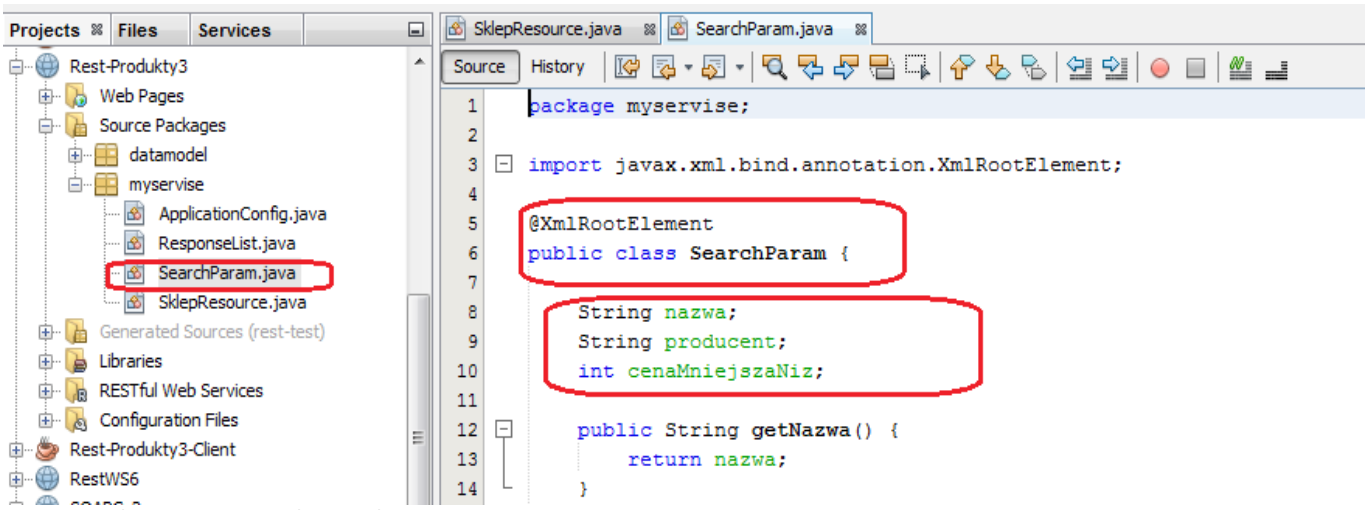
Podgląd komunikatów w HTTP Monitorze:



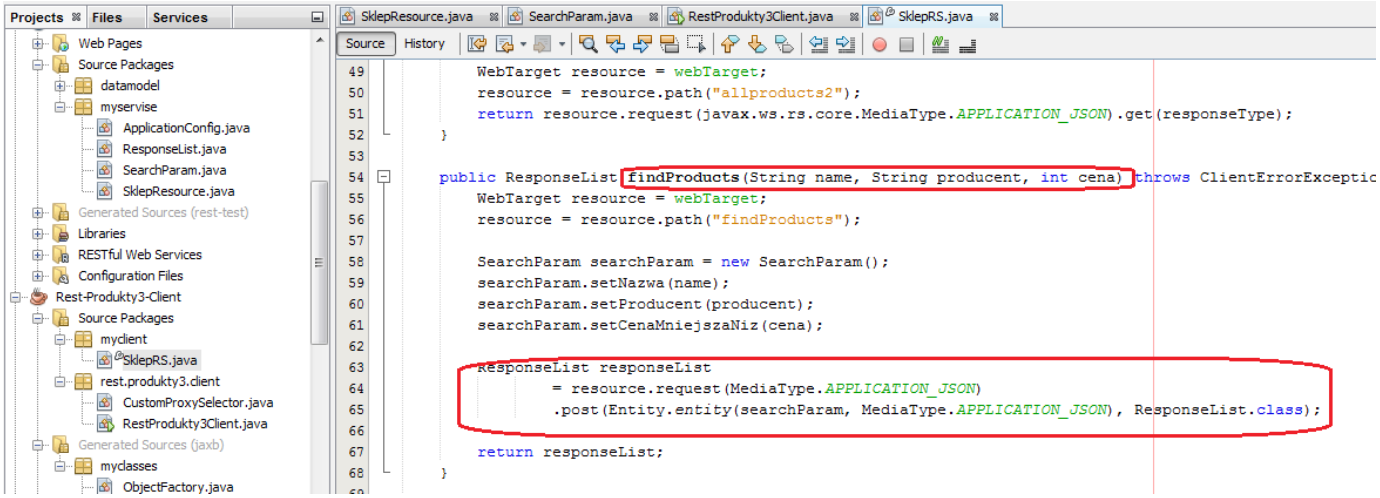
Ćwiczenie 4. Serwis z przyjmujący obiekt jako parametr i zwracający listę obiektów

Dodanie RESTful API wyszukujące produkty po nazwie, producencie i cenie





Zapytanie po stronie klienta



Podgląd komunikatów w HTTP Monitorze: