Laboratorium 10 - SOA.
Tematyka: REST

Tworzenie i korzystanie z serwisów REST-owych w Javie.

Celem laboratorium jest zaznajomienie z Web Service typu REST tworzonymi w oparciu o bibliotekę JAX-RS.

0. Zaznajomić się z
   http://docs.jboss.org/resteasy/docs/2.2.1.GA/userguide/html/Installation_Configuration.html

   JBoss domyślnie używa implementacji RESTEasy do realizacji usług REST-owych.

1. Poszukaj materiałów dotyczących implementacja REST w używanym przez Ciebie IDE. Na przykład osoby korzystające z InteliJ znajdą taką dokumentację pod adresem https://www.jetbrains.com/idea/help/preparing-for-rest-development.html

2. Przeglądnij dokument pod nazwa REST_Przykład gdzie znajdujcie się kompetium wiedzy na temat REST wraz z prostym przykładem. Proszę zwrócić uwagę że opisano w nim implementacje Jersey

3. Proszę przeglądnąć poniższe materiały i wykonać przedstawione tam przykłady

http://www.vogella.com/articles/REST/article.html

http://www.mkyong.com/tutorials/jax-rs-tutorials/

https://docs.oracle.com/javaee/7/tutorial/jaxrs.htm#GIEPU

http://docs.oracle.com/cd/E19776-01/820-4867/ggnyk/index.html

Jako ćwiczenie podsumowujące tematykę samodzielnie wykonać następujące zadania:

Zad 1. Napisz aplikacje pozwalającą na zarządzanie kolekcją kotów. Aplikacja za pomocą API restowego powinna wykonywanie typowe operacje na obiektach typu kot (dodawanie, usuwanie, pobieranie, edycja, pobieranie wszystkich).
Wyróżniamy dwa rodzaje adresów: konkretnego elementu (np. /koty/1) oraz całej kolekcji (wszystkich elementów, np. /koty).

Konkretne metody protokołu HTTP odpowiadają za odpowiednie operacje:

- GET — pobieranie (zarówno kolekcji, jak i pojedynczego elementu)
- POST — tworzenie (tylko kolekcji)
- PUT — aktualizacja (tylko pojedynczego elementu)
- DELETE — usuwanie (tylko pojedynczego elementu)

Poza zwracaną wartością (np. zmodyfikowanym elementem), komunikacja w przypadku REST powinna odbywać się także poprzez statusy HTTP. Np. zapytanie GET /koty/7, jeśli kot o id 7 nie istnieje, zwróci status 404 — nie znaleziono. Dzięki temu unikamy tworzenia dodatkowych pól typu 'status zapytania' (co ma często miejsce w przypadku serwisów typu SOAP).

Zad 2. Napisz w oparciu o usługi REST prostą aplikacje umożliwiającą dostęp do kolekcji filmów. Kolekcja może być przechowywana w bazie danych lub w pliku XML.

Na potrzeby lab przyjmij następujące API REstowe:

- Opis usługi – dostępny w formacie JSON. Zawiera następujące pole:
  - 'movies': link do zasobu 'kolekcja filmów'
- Kolekcja filmów – dostępny w formacie JSON. Zawiera następujące pole:
  - 'movies': lista opisów zasobu 'film'
- Film – dostępny w formacie JSON. Zawiera następujące pola:
  - 'id': identyfikator
  - 'title': tytuł
  - 'uri': link do siebie

a). zwracanie filmów

Uzupełnij metodę wyświetlającą pojedynczy film..

b) usuwanie filmów

Dopisz metodę usuwającą film. Aby operacja zwracała np. kod 204 No Content, wystarczy że metoda nie będzie nic zwracać.

c) content negotiation

Dopisz zwracanie listy filmów jako listę URI (typ MIME: text/uri-list). Jest to tzw. negocjowanie zawartości przez klienta, zwane po angielsku 'content negotiation' albo 'conneg'.

d) statusy HTTP

Sprawdź jakie kody zwrócą te operacje, użyj curla:

- Wyświetlenie listy filmów jako text/plain
- Aktualizacja całej kolekcji filmów (PUT movies)

Utwórz zasób o URI ‘/art’, który spowoduje przekierowanie do /movies. Musisz zwrócić odpowiedź o odpowiednim statusie, która w polu ‘location’ będzie miała podane URI do którego przekierowuje. Odpowiedni status możesz znaleźć np. na http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

e). filtrowanie po tytule

Dopisz parametr ‘title’ do zasobu movies pozwalający wyszukać film po tytule. Upewnij się że parametr jest dodany do opisu usługi.