Praca domowa 12 – nbp

Termin zwrotu : 29 stycznia godz. 23.00
Zadanie uznaje się za zaliczone, gdy praca oceniona zostanie na co najmniej 6 pkt.

Na serwerze aplikacyjnym Glassfish 4 w kontenerze *ejb* zainstalowany jest pod nazwą *mdb-project* (deployment descriptor) komponent (session bean) o nazwie *MdbManager* wraz z interfejsem *IMdbManager*, który zdefiniowany jest następująco :

```
package pl.jrj.mdb;
import javax.ejb.Remote;

@Remote
public interface IMdbManager {
        public String currencyId();        //  trzyliterowy kod waluty
}
```

Metoda *currencyId* dokonuje rejestracji użytkownika w systemie zwracając trzyznakową postać kodu waluty**.**

Należy stworzyć usługę sieciową pracującą w standardzie WebService's (JAX-RS). Usługa winna zwracać stosunek wartości jednej jednostki waluty obcej, której kod przekazywany żądaniem URL wg. oznaczeń NBP (np. EUR oznacza euro) do waluty referencyjnej, której kod zwróciła procedura *currencyId,* wycenioną z dokładnością do poprawnie zaokrąglonych czterech miejsc dziesiętnych. A więc przykładowo jeżeli procedura *currencyId* zwróci wartość PLN a w żądaniu URL podano EUR, to wynikiem zapytania jest aktualna wartość kursu EUR wyrażona w złotówkach.
Aktualna tabela kursów walut dostępna jest na stronie http://www.nbp.pl/home.aspx?f=/kursy/kursya.html w postaci tekstowej, ale również w postaci przeznaczonej do automatycznego przetwarzania (link do pliku w formacie xml).

Program ma być zapisany w plikach : `ExchRates.java` zawierającym implementację komponentu, `IMdbManager.java` zawierającym definicję interfejsu komponentu `MdbManager` oraz ewentualnie pozostałych – wykonanych dla potrzeb zadania – elementów rozwiązania. Poszczególne elementy rozwiązania nie mogą korzystać z bibliotek zewnętrznych innych niż niezbędne moduły serwera (jak np. javaee.jar itp.).

Proces kompilacji musi być możliwy z użyciem komendy

```
javac -cp <app-server-modules> -Xlint ExchRates.java IMdbManager.java *.java
```

Zawartość pliku web.xml, który używany będzie w trakcie uruchamiania i testowania usługi podano niżej :

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
       xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
       id="WebApp_ID" version="3.0">
  <servlet>
    <servlet-name>javax.ws.rs.core.Application</servlet-name>
  </servlet>
  <servlet-mapping>
    <servlet-name>javax.ws.rs.core.Application</servlet-name>
    <url-pattern>/*</url-pattern>
  </servlet-mapping>
</web-app>
```

Proces uruchamiania usługi musi być możliwy z użyciem polecenia postaci

`http://<host:port>/<app-name>/exchangeRate/EUR`

gdzie *EUR* jest wskazuje kod poszukiwanej waluty.

Wymagania :

- Klasa implementująca aplikację winna zostać zdefiniowane w pliku `ExchRates.java`.
- Interfejs umożliwiający poprawną rejestrację zadania winien zostać zdefiniowany w pliku `IMdbManager.java`.
- W pliku README.pdf winien być zawarty opis mechanizm operowania danymi oraz algorytm wyznaczania wyniku.
- Proces obliczenia rozwiązania winien się kończyć w czasie nie przekraczającym 1 min (orientacyjnie dla typowego notebooka). Po przekroczeniu limitu czasu zadanie będzie przerywane, i traktowane podobnie jak w sytuacji błędów wykonania (czyli nie podlega dalszej ocenie).

Sposób oceny **:**

- 1 pkt – **Weryfikacja** : czy program jest skompletowany i spakowany zgodnie z ogólnymi zasadami przesyłania zadań.
- 1 pkt – **Kompilacja** : każdy z plików winien być kompilowany bez jakichkolwiek błędów lub ostrzeżeń (w sposób omówiony wyżej)
- 1 pkt – **Wykonanie** : program powinien wykonywać się bez jakichkolwiek błędów i ostrzeżeń (dla pliku danych wejściowych zgodnych z wyżej zamieszczoną specyfikacją) z wykorzystaniem omówionych wyżej parametrów linii komend
- 2 pkt – **README** : plik README.pdf dokumentuje w sposób kompletny i właściwy sposób zestawiania połaczenia
- 1 pkt – **Styl kodowania** : czy funkcji i zmienne posiadają samo-wyjaśniające nazwy ? Czy podział na funkcje ułatwia czytelność i zrozumiałość kodu ? Czy funkcje eliminują (redukują) powtarzające się bloki kodu ? Czy wcięcia, odstępy, wykorzystanie nawiasów itp. (formatowanie kodu) są spójne i sensowne ?
- 4 pkt – **Poprawność algorytmu** : czy algorytm został zaimplementowany poprawnie a wynik odpowiada prawidłowej (określonej zbiorem danych testowej) wartości.