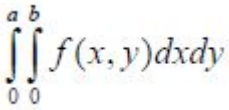
Praca domowa 6 – integral

Szlachetka Miłosz
Nr albumu. 114014

1.Cel zadania

Celem zadania było wyznaczenie wartości całki podwójnej zadanej wzorem:



$$\int_0^a \int_0^b f(x,y)dxdy$$

Wartość całki musiała być wyznaczona z dokładnością co najmniej 10^{-n}. Zadanie należało rozwiązać implementując servlet, który przyjmuje parametry a,b,n przekazywane w URL i zwraca odpowiedź, która jest wartością w.w. całki. Dodatkowo należało zaimplementować komponent EJB o nazwie IBean (oraz interfejs IBeanRemote), który wyznacza wartość całki z wcześniej wspomnianą dokładnością.

2.Struktura programu

Program składa się z czterech plików: **IFunctMonitor.java, IBeanRemote.java, IBean.java, Calc.java**

IFunctMonitor.java zawiera:
@Remote
public interface IFunctMonitor - interfejs typu Remote, który jest implementowany przez komponent o nazwie FunctMonitor znajdujący się na serwerze aplikacyjnym.Interfejs zawiera deklaracje metody **public double f(double x, double y).** Metoda ta zwraca wartość pewnej funkcji dla argumentów x, y.

IBeanRemote.java zawiera:
@Remote
public interface IBeanRemote - interfejs typu Remote, który jest implementowany przez komponent IBean. Interfejs zawiera deklarację metody **double solve(double a, double b, int n).** Metoda ta zwraca wartość wyżej wymienionej całki podwójnej po obszarze zadanym parametrami a,b ([0,a] oraz [0,b]) z dokładnością 10^{-n}.

IBean.java zawiera:
@Stateless
public class IBean implements IBeanRemote - klasa, będąca bezstanowym komponentem EJB. Wyznacza wartość wyżej wymienionej całki podwójnej z zadaną dokładnością. Klasa zawiera:
private IFunctMonitor retrieveFunctionMonitor() - metoda wyszukująca i zestawiająca połączenie z komponent EJB o nazwie FunctMonitor. Wyszukiwaniejest realizowane dzięki

interfejsowi JNDI. Połączenie z komponentem jest wykonywane przy użyciu kontekstu InitialContext i metody lookup. Zwraca referencję do komponentu **FunctMonitor.**
@Override
public double solve(double a, double b, int n) - metoda zwraca wartość wyżej wymienionej całki podwójnej z dokładnością 10^{-n}. Wykorzystuje wyżej opisaną metodę retrieveFunctionMonitor() w celu wyszukania i nawiązania połączenia z komponentem EJB. Jeśli nie udało się odnaleźć komponentu to zwraca NaN. Metoda korzysta z metody calculateIntegralUsingTrapezoidRule(double a, double b) opisanej poniżej, która zwraca wartość całki podwójnej obliczoną metodą trapezów. Na końcu zwraca zaokrąglony wynik, dzięki wywołaniu metody round(double value, int precision) opisanej niżej.
private double calculateIntegralUsingTrapezoidRule(double a, double b) - metoda zwraca wartość całki podwójnej. Do obliczenia całki podwójnej używa metody trapezów.
private double round(double value, int precision) - metoda służąca do zaokrąglania oraz ustawiania określonej precyzji dla zadanej wartości “value”. Parametr “precision” określa ilość miejsc po przecinku zwracanego przez tę metodę wyniku.
private IFunctMonitor functMonitor - zmienna przechowująca komponent EJB o nazwie FunctMonitor.

Calc.java zawiera:
public class Calc extends HttpServlet - klasa będąca servletem. Korzysta z komponentu EJB o nazwie IBean. Klasa zawiera:
private static final long serialVersionUID - pole przechowujące numer wersji
@EJB
private IBeanRemote iBean - pole przechowujące komponent EJB, który implementuje mechanizm obliczania całki podwójnej. Komponent jest wstrzykiwany dzięki użyciu adnotacji @EJB.
protected void processRequest(HttpServletRequest request,HttpServletResponse response) throws ServletException, IOException - metoda, która pobiera parametry a,b,n z URL. Następnie wywołuje metodę solve(double a, double b, int n) z komponentu IBean, która zwraca wartość wyżej wymienionej całki podwójnej z dokładnością 10^{-n}. Na końcu wypisuje wynik.
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException - metoda wywołuje wyżej opisaną metodę processRequest.
@Override
protected void doPost(HttpServletRequest request,HttpServletResponse response) throws ServletException, IOException - metoda wywołuje wyżej opisaną metodę processRequest.
@Override
public String getServletInfo() - metoda zwracająca opis servletu.

3.Mechanizm wyszukiwania (lookup) i zestawiania połączenia

Wyszukiwanie komponentu EJB zostało zrealizowane dzięki interfejsowi JNDI, który daje możliwość wyszukiwania i uzyskiwania dostępu do serwisów i katalogów przy użyciu przenośnych nazw. Aby uzyskać dostęp do komponentu EJB o nazwie FunctMonitor i deskryptorze ejb-project użyto następującej nazwy,która określa położenie komponentu: **java:global/ejb-project/FunctMonitor!pl.jrj.fnc.IFunctMonitor.**

java:global - przestrzeń nazw, do wyszukiwania zdalnych EJB.

ejb-project - nazwa modułu.
FunctMonitor - nazwa komponentu EJB
!pl.jrj.fnc.IFunctMonitor - interfejs komponentu

Połączenie z komponentem EJB było zrealizowane poprzez utworzenie kontekstu (InitialContext), a następnie wywołanie na nim metody lookup, z parametrem będącym adresem komponentu.

4.Mechanizmu poszukiwania rozwiązania

Servlet pobiera z adresu URL parametry a,b,n , przy czym a,b określają zakres całkowania ([0,a] oraz [0,b]) , natomiast parametr n określa dokładność wyniku. Następnie z wykorzystaniem z komponentu EJB o nazwie IBean wywoływana jest metoda solve(double a, double b, int n), która oblicza wartość wyżej wymienionej całki podwójnej z dokładnością 10^{-n}. Komponent IBean jest wstrzykiwany do servletu dzięki użyciu adnotacji @EJB przy polu klasy. W metodzie solve komponentu IBean wyszukiwany jest komponent FunctMonitor i nazwiązywane jest z nim połączenie. Następnie przy użyciu metody trapezów (dalej w metodzie solve) obliczana jest wartość wyżej opisanej całki. Metoda trapezów (zwana też “four corners method”) dla całki podwójnej została zaimplementowana zgodnie ze wzorem:

$$F_{mn} = \sum_{i,j=1,1}^{m,n} \frac{1}{4} \left(f(x_i, y_j) + f(x_i, y_{j+1}) + f(x_{i+1}, y_i) + f(x_{i+1}, y_{j+1}) \right) A_{ij},$$

gdzie Aij = deltaX * deltaY, czyli fragment obszaru (prostokąt) po którym całkujemy. Powyższa metoda polega na wyliczeniu średniej wysokości dla graniastosłupa o podstawie prostokąta (Aij). Dla takiego graniastosłupa obliczana jest objętość. Suma objętości wszystkich takich graniastosłupów daje przybliżoną wartość wyżej wymienionej całki podwójnej. Otrzymana wartość całki podwójnej jest następnie zaokrąglana do “n” miejsc po przecinku. Uzyskana wartość jest zwracana z metody solve komponentu IBean do servletu,

gdzie następuje jej wypisanie. Dokładność wyniku jest uzyskiwana poprzez pomnożenie go przez 10^n. Następnie wynik jest zaokrąglany do jedności po czym jest dzielony przez 10^n. Tym sposobem uzyskiwana jest wynik zaokrąglany do “n” miejsc po przecinku. Kod realizujący opisaną metodę zaokrąglania został zaprezentowany poniżej:

```
double scale = Math.pow(10.0, precision);
double roudedResult = Math.round(value * scale) / scale;
```