dr inż. Paweł Głuchowski (Politechnika Wrocławska, Wydział Informatyki i Telekomunikacji)

Języki Programowania

Laboratoria nr 7

wersja 1.1

Temat: Programowanie rozproszone.

Zadanie 1.

Działanie programu:

1. Wyświetlenie okna komunikatora internetowego zawierającego:
   - przewijalne pole tekstowe zdalnego odbiorcy, pokazujące wiadomości do niego wysłane i od niego otrzymane;
   - przewijalne pole tekstowe lokalnego nadawcy (użytkownika komunikatora), służące do pisania wiadomości do wysłania;
   - rozwijana lista z imionami wszystkich aktualnie zarejestrowanych odbiorców;
   - przyciski *Połącz* i *Rozłącz*;
   - pole tekstowe z imieniem nadawcy (użytkownika);
   - przyciski *Zarejestruj* i *Wyrejestruj*.
2. Użytkownik może zarejestrować się w serwisie komunikatora jako nadawca, wpisując swoje imię i naciskając przycisk *Zarejestruj*.
3. W rozwijalnej liście pojawiają się imiona pozostałych zarejestrowanych użytkowników (jako odbiorcy), z których użytkownik może wybrać jednego i połączyć się z nim, naciskając przycisk *Połącz*.
4. Nadawca przesyła wybranemu odbiorcy tekst napisany w polu nadawcy, przez naciśnięcie klawisza *enter*. Wiadomość ta pojawia się wtedy w polu odbiorcy u nadawcy i u odbiorcy, poprzedzona imieniem nadawcy i dwukropkiem.
5. Użytkownik może rozłączyć się z danym użytkownikiem, naciskając przycisk *Rozłącz*.
6. Użytkownik może wyrejestrować się z serwisu, naciskając przycisk *Wyrejestruj*.

Wymagania: Należy wykonać program komunikatora i program serwisu, wykorzystując programowanie rozproszone oparte na RMI.

Program serwisu ma działać w tle oraz:

- rejestrować i wyrejestrowywać namiastki potencjalnych użytkowników (nadawców/odbiorców);
- być zarejestrowany w rejestrze RMI, aby wszyscy użytkownicy uruchamiający swoje komunikatory mogli z niego korzystać w sposób zdalny;
- dostarczać imiona zarejestrowanych użytkowników do wyświetlenia w liście imion.

Implementacja serwisu zapisów ma korzystać z następującego zdalnego interfejsu:

```
import java.rmi.Remote;
import java.rmi.RemoteException;
public interface SInterface extends Remote{
  int zapisz (KInterface u, String n) throws RemoteException;
  int wypisz(String n) throws RemoteException;
  String[] listuj() throws RemoteException;
  KInterface pobierz(String n) throws RemoteException;}
```

Implementacja komunikatora ma korzystać z następującego zdalnego interfejsu:

```
import java.rmi.Remote;
import java.rmi.RemoteException;
public interface KInterface extends Remote{
  void pisz (String s) throws RemoteException;}
```

Metoda `int zapisz (KInterface u, String n) throws RemoteException;` rejestruje namiastkę *u* pod nazwą *n* w serwisie. Jeśli operacja się powiodła, to metoda zwraca *1*, jeśli nie, to zwraca *0* (np. gdy w serwisie już jest zarejestrowana jakaś namiastka o nazwie *n*).

Metoda `int wypisz(String n) throws RemoteException;` wyrejestrowuje namiastkę zarejestrowaną pod nazwą *n* z serwisu. Jeśli operacja się powiodła, to metoda zwraca *1*, jeśli nie, to zwraca *0* (np. gdy w serwisie nie ma namiastki o nazwie *n*).

Metoda `String[] listuj() throws RemoteException;` pobiera z serwisu tablicę nazw wszystkich zarejestrowanych namiastek.

Metoda `KInterface pobierz(String n) throws RemoteException;` pobiera z serwisu namiastkę zarejestrowaną pod nazwą *n*. **Uwaga:** Metoda ta może zwrócić namiastkę, którą za chwilę miał właśnie wyrejestrować jej właściciel; albo może zwrócić błędną wartość, jeśli w argumencie podano nazwę niezarejestrowanej namiastki.

Metoda `void pisz (String s) throws RemoteException;` przesyła tekst *s* do komunikatora odbiorcy.