Contents

1 Programowanie klienta TCP

Klasa gniazda sieciowego do wykorzystanie w programie klienckim TCP to Socket. Gniazdo tworzymy podając adres serwera w postaci String oraz port serwera jako Integer.

```
Socket socket = new Socket("127.0.0.1", 1000);
```

Metoda getOutputStream() zwraca nam strumień wyjściowy. Wykorzystując go w konstruktorze klasy PrintWriter możemy wysyłać dane.

```
PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
```

W powyższym przykładzie wartość true to parametr boolean autoFlush, który sprawia, że metody println, printf oraz format będą czyściły buffor wyjściowy.

W przypadku, gdy chcemy wysłać jakiś napis wykorzystujemy obiekt PrintWriter i wywołujemy jego metodę println().

```
out.println("To jest moja wiadomość");
```

Do odbierania i wyświetlania danych możemy wykorzystać obiekt klasy BufferedReader. Może on przyjąć obiekt InputStreamReader, który może być strumieniem wejściowym z naszego gniazda. Strumień wejściowy gniazda otrzymamy wykonując metodę getInputStream().

```
BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
```

Do odczytania danych przychodzących do naszego klienta możemy wykorzystać metodę readLine(). Poniżej przykład przypisania danych przychodzących do obiektu klasy String.

```
String message = in.readLine();
```

Przydatna może okazać się również funkcja Socket.close(), która pozwala na zamknięcie gniazda tym samym zamyka ona połączenie.

```
socket.close();
```

2 Programowanie serwera TCP

W przypadku serwera TCP używmay innego gniazda. Takie gniazdo dostarcza nam klasa ServerSocket. Przyjmuje ona numer portu na którym nasłuchuje serwer. Port jest tutaj ponownie zmienną typu Integer.

```
ServerSocket serverSocket = new ServerSocket(port)
```

Żeby pozwolić klientowi na podłączenie się do naszego serwera musimy go zaakceptować. Pozwala nam na to metoda accept(). Zwraca ona obiekt typu Socket.

```
Socket clientSocket = serverSocket.accept();
```

A więc wykorzystując gniazdo serwera i wywołując na nim metodę accept() w momencie, gdy klient podłączy się do naszego serwera ta metoda zwróci nam gniazdo sieciowe dla naszego klienta. Będziemy używać tego gniazda żeby komunikować się z naszym klientem.

Następnie dla każdego klienta możemy postąpić tak samo jak w przypadku programowania klienta TCP. Czyli wykorzystując nowo otrzymane gniazdo klienckie utworzyć obiekty PrintWriter oraz BufferedReader. Dzięki temu będziemy mogli przesyłać danemu klientowi dane oraz od niego te dane odbierać.

3 Kod clientexample.java i serverexample.java

Napisałem prosty kod w którym serwer oczekuje na klientów w nieskończoność. Po nawiązaniu połączenia klient wysyła do serwera wiadomość "hello" na którą serwer odpowiada. Dodałem trochę komentarzy mam nadzieje, że będzie to jasne.

W razie pytań proszę o maila na adres: maciej.miroslaw@pjwstk.edu.pl