JAVA 2

Networking

Teme

- Mrežno programiranje u Javi
- URL
- TCP
- *Socket*
- UDP
- *DatagramSocket*

Mrežno programiranje u Javi

- *Networking API* – omogućuje razvoj aplikacija koje se izvršavaju na mrežno povezanim računalima
- J2SE – od samih početaka, Java je dizajnom orijentirana mrežno
- ***java.net*** paket
- API iznimno pojednostavljen naspram drugih jezika
- objekti se serijaliziraju prilikom prijenosa

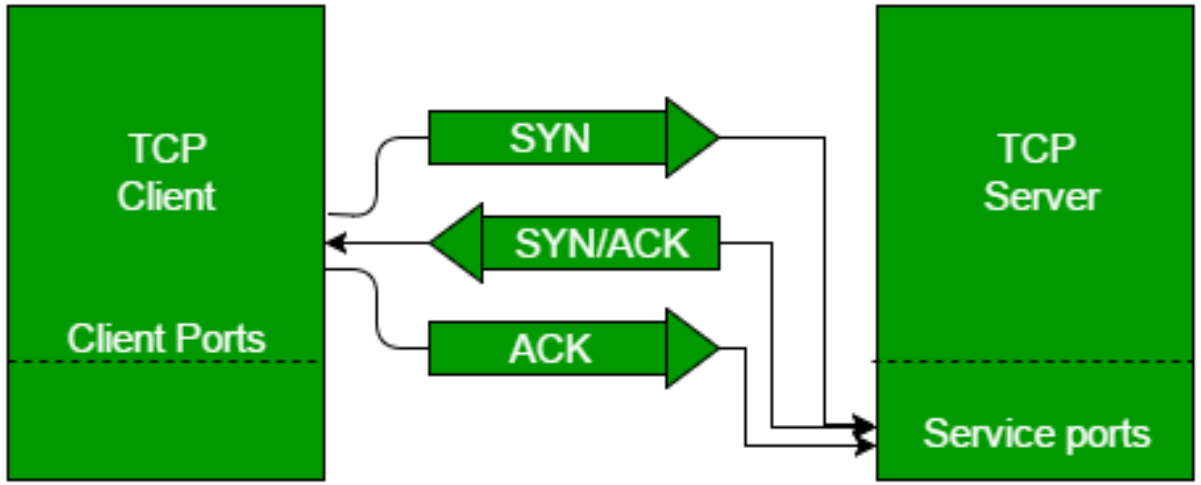
URL

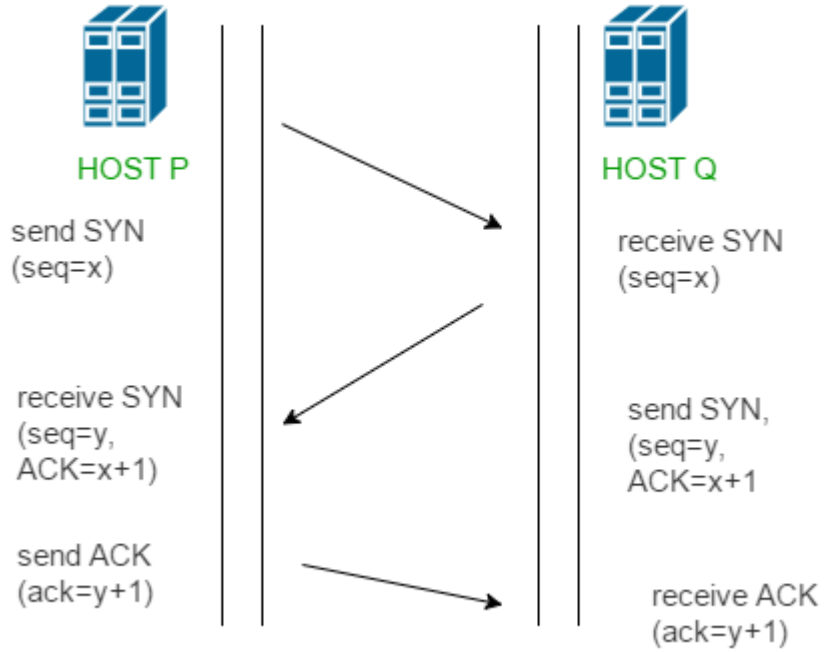
- ***U**niform **R**esource **L**ocator* – jedinstveno određuje lokaciju resursa
- zajedno sa URN (*name*) čini specijalizaciju URI (*identifier*)
- komponente:
  - *protocol* (http, ftp...)
  - *resource name*
    - host name – ime mašine na kojem je resurs
    - filename – putanja do resursa
    - port number – opcionalan broj porta servisa za pristup resursu
    - reference – opcionalna refererenca na #
- ***java.net.URL*** – pristup komponentama, otvaranje konekcije i potom pristup komunikacijskim tokovima

TCP

- ***T**ransmission **C**ontrol **P**rotocol* – komunikacijski protokol
- pouzdana komunikaciju klijenta i poslužitelja
- uspostavlja *handshake* između klijenta i poslužitelja
  - Korak 1 (SYN) – klijent šalje ***S**ynchronize Sequence Number* segment
  - Korak 2 (ACK + SYN) – poslužitelj potvrđuje primitak i šalje svoj SYN
  - Korak 3 (ACK) – klijent potvrđuje primitak
- ***P**ositive **A**cknowledgement with **R**e-transmission* (PAR):
  - **P**rotocol **D**ata **U**nit (PDU) segment se šalje uzastopno dok primitak nije potvrđen
- HTTP, FTP, SMTP...

TCP



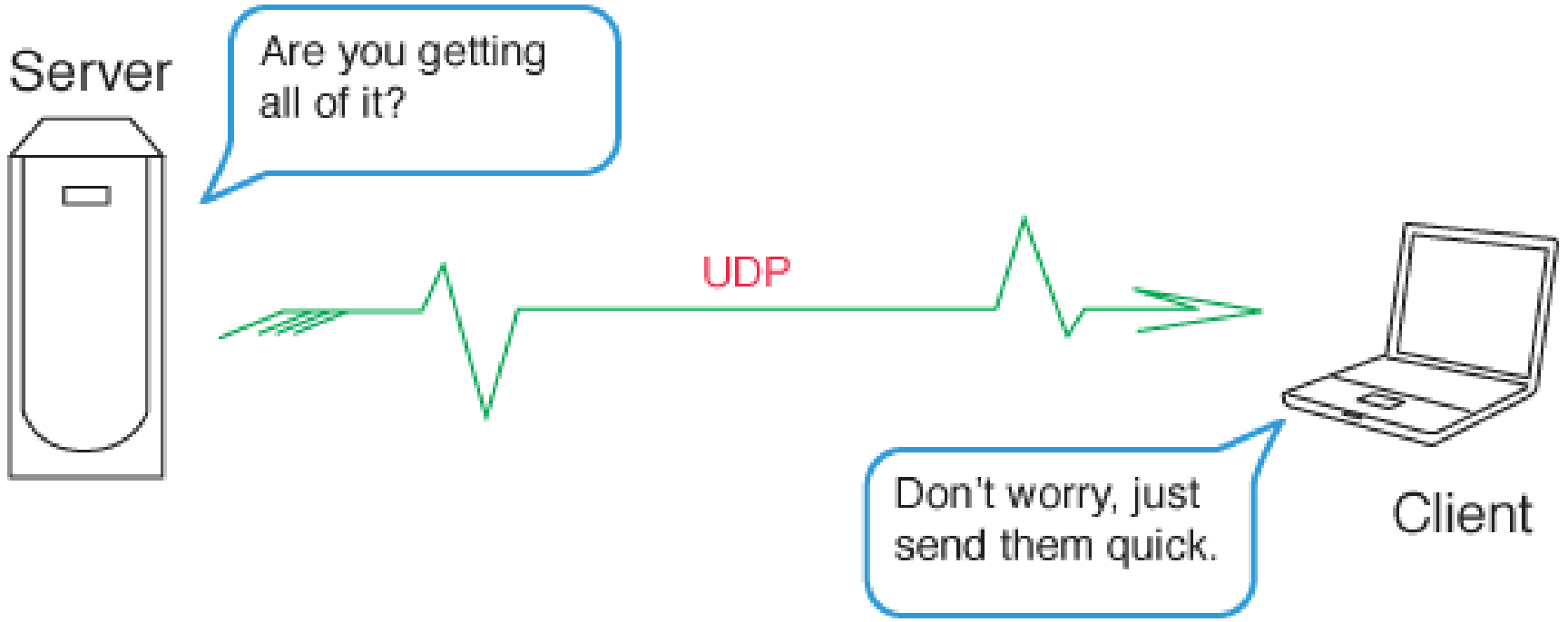


Izvor: https://www.geeksforgeeks.org/tcp-3-way-handshake-process/

UDP

- ***U**ser **D**atagram **P**rotocol* – komunikacijski protokol
- mala kašnjenja (low latency), tolerira greške (*loss tolerating*)
- različit od TCP:
  - nepouzdan
  - brz
  - *lightweight*
  - paketi mogu stići na klijenta van redoslijeda
  - datagram paketi se ne šalju ponovno u slučaju gubitka
- omogućuje *broadcast* i *multicast*
- DNS, DHCP...

UDP

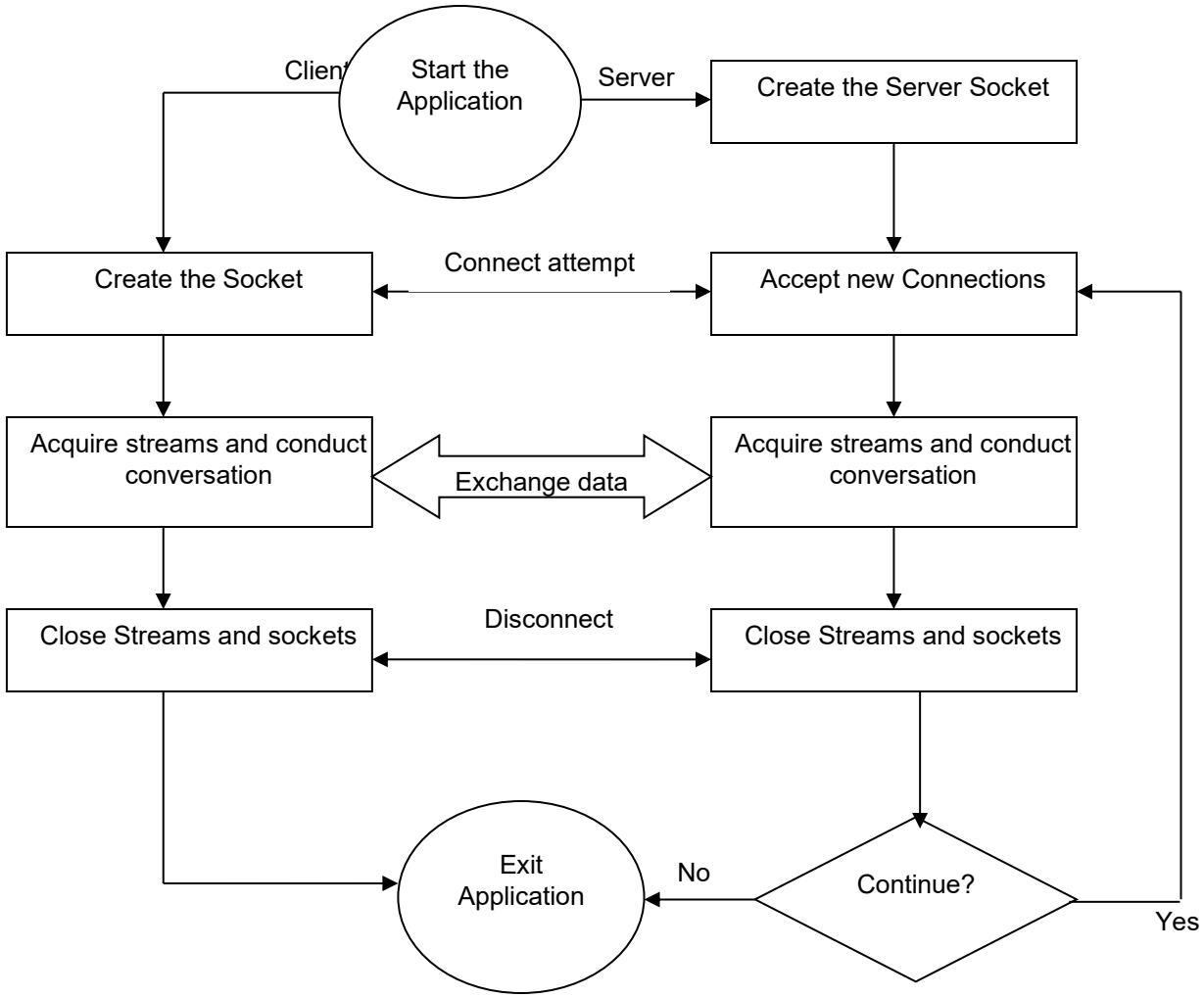


Izvor: http://classes.dma.ucla.edu/Fall05/161A/projects/Joshua/Project1/udp.htm

Socket

- TCP komunikacijski mehanizam
- prilikom stvaranja konekcije, otvara se *Socket*
- koraci
  - poslužitelj na određenom portu instancira *ServerSocket*
  - poslužitelj osluškuje na portu i čeka klijentsku konekciju
  - klijent instancira *Socket*, specificirajući poslužitelja i port – dobiva *random* port
  - prilikom uspostavljanja komunikacije, poslužitelj *accept()* metodom pristupa referenci klijentskog *Socketa*
  - otvaraju se tokovi za komunikaciju
- zatvaranjem *Socketa*, zatvaraju se i tokovi komunikacije

Socket


Client
Start the Application
Server
Create the Server Socket
Create the Socket
Connect attempt
Accept new Connections
Acquire streams and conduct conversation
Exchange data
Acquire streams and conduct conversation
Close Streams and sockets
Disconnect
Close Streams and sockets
Exit Application
No
Continue?
Yes

DatagramSocket

- UDP komunikacija – *DatagramPacket* šalje se između *DatagramSocket*-a
- koraci
  - poslužitelj na određenom portu instancira *DatagramSocket*
  - poslužitelj osluškuje na portu i čeka klijentski paket
  - klijent instancira *DatagramSocket*, a potom i *DatagramPacket* u kojem definira poslužitelja i port i šalje ga – dobiva *random* port
  - kada stigne paket klijenta, poslužitelj iščitava podatke, ali i podatke o klijentu (adresu i port)
  - poslužitelj kreira *DatagramPacket* u kojem definira klijenta i port te ga šalje

Demo

- Project

Izvor:http://www.jnhsolutions.com/contact-us/request-a-demo/

Hvala na pažnji!