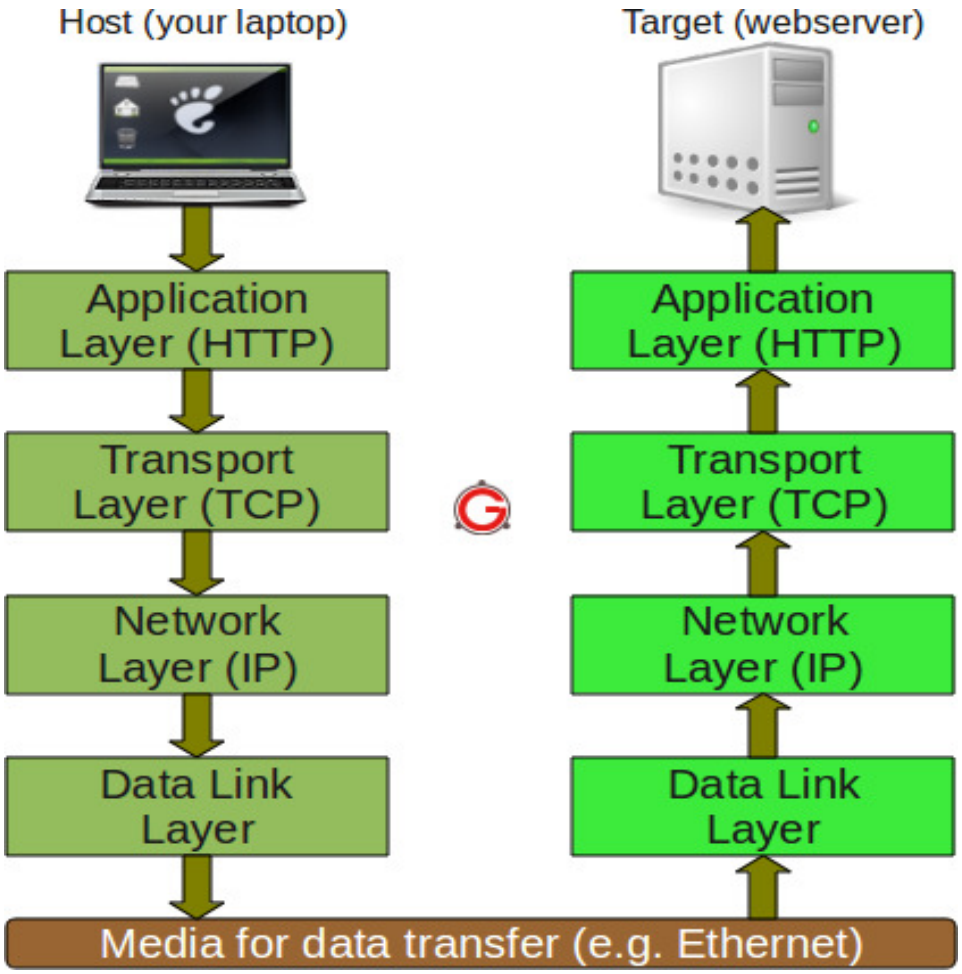
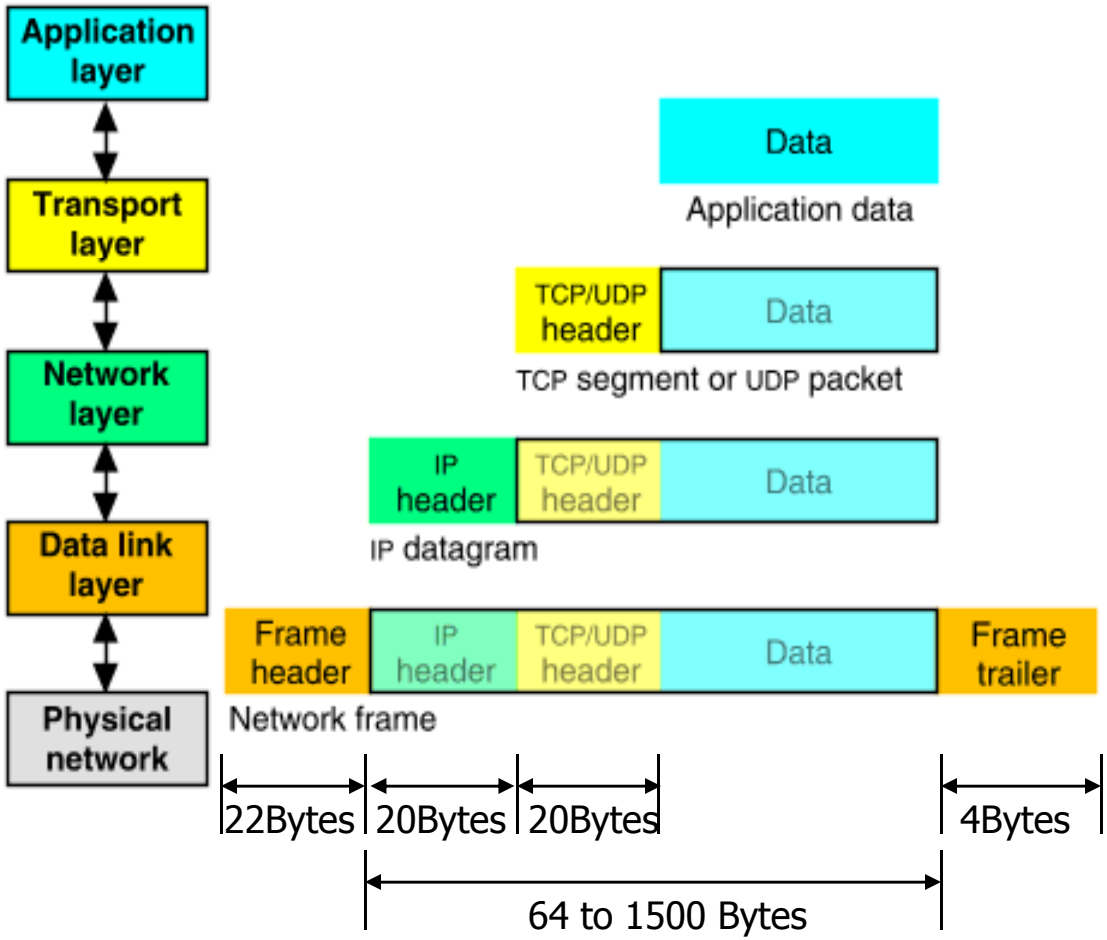
Socket programming

Sockets & Streams

TCP IP Protocol Stack



TCP IP Protocol Stack



Το επίπεδο μεταφοράς

- Υπάρχουν δύο προσεγγίσεις.
  - Συνδεσμοστρεφής (connection oriented) υπηρεσία μεταφοράς
    - Εγκατάσταση σύνδεσης, μεταφορά δεδομένων τερματισμός σύνδεσης-αξιοπιστία (παράδειγμα: TCP).
  - Ασυνδεσμική (connectionless) υπηρεσία μεταφοράς
    - Όχι αξιοπιστία (παράδειγμα: UDP).

User Datagram Protocol (UDP)

- Προσφέρει ελάχιστες υπηρεσίες.
  - Πολυπλεξία.
  - Ελάχιστο έλεγχο σφαλμάτων.

- Δεν υποστηρίζει
  - Έλεγχο ροής ή συμφόρησης.
  - Έλεγχο σφαλμάτων.
  - Επαναμετάδοση σε περίπτωση σφαλμάτων.
  - Παραλαβή δεδομένων στην σωστή σειρά.

User Datagram Protocol (UDP)

- Χρησιμότητα UDP
  - Δεν εγκαθιστά συνδέσεις (η διαδικασία εγκατάστασης εισάγει καθυστέρηση)
  - simple: όχι connection states σε sender και receiver
  - Μικρό segment header
  - Δεν υπόκειται σε έλεγχο συμφόρησης:
    - UDP can blast away as fast as desired

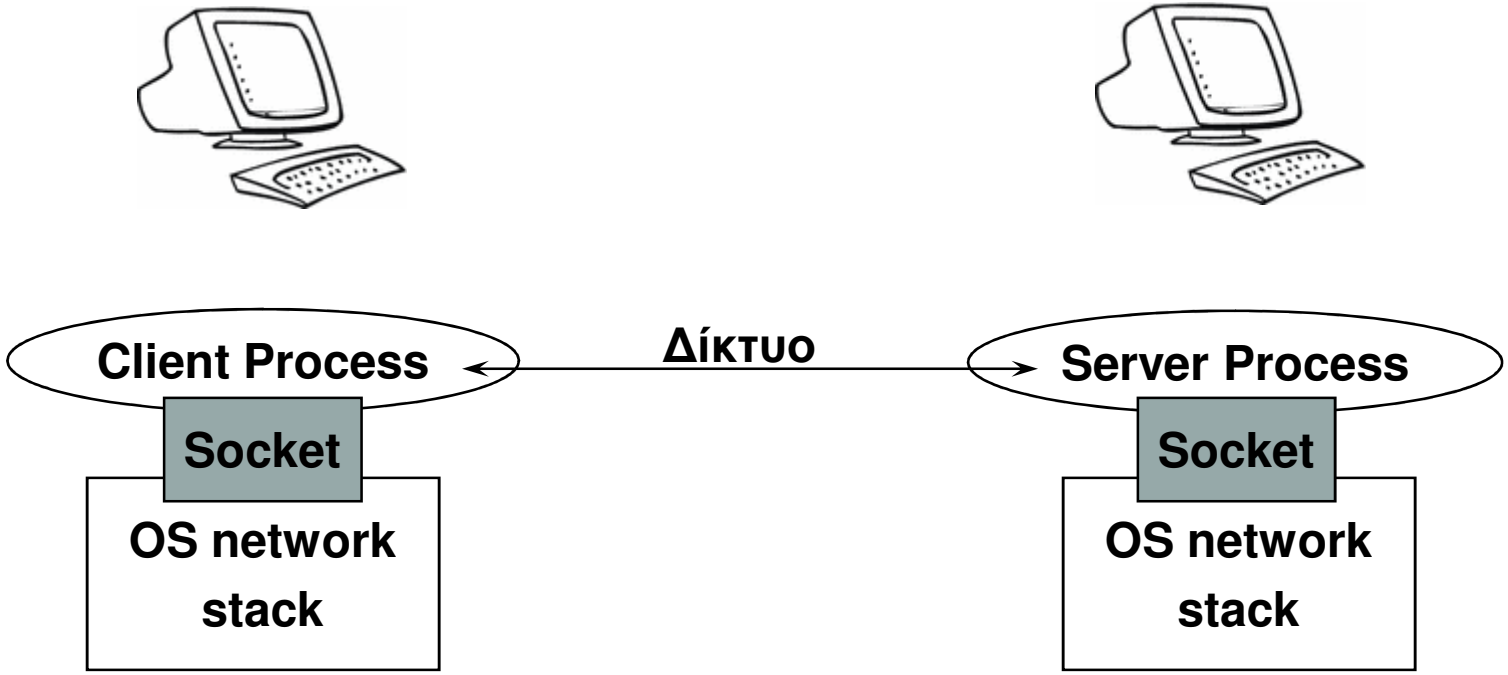
User Datagram Protocol (UDP)

- Συχνά χρησιμοποιείται για streaming multimedia apps
  - loss tolerant
  - rate sensitive
- Άλλες χρήσεις UDP
  - DNS
  - SNMP
- Για reliable transfer over UDP: Προσθήκη reliability στο application layer
  - application-specific error recovery

Transmission Control Protocol (TCP)

- Προσφέρει τα παρακάτω:
  - Εγγυάται την παράδοση των μηνυμάτων.
  - Παραδίδει τα μηνύματα με τη σωστή σειρά.
  - Παραδίδει μόνο ένα αντίγραφο.
  - Υποστηρίζει αυθαίρετο μέγεθος μηνυμάτων.
  - Υποστηρίζει το συγχρονισμό μεταξύ των επικοινωνούντων.
  - Επιτρέπει το έλεγχο της ροής προς το παραλήπτη.
  - Πραγματοποιεί έλεγχο συμφόρησης.
  - Υποστηρίζει την πολυπλεξία πολλαπλών χρηστών.
  - Τα παραπάνω γίνονται με διαφάνεια στους προγραμματιστές της εφαρμογής.
    - Οι οποίοι απλά καλούν διαδικασίες βιβλιοθήκης για τη χρήση των υπηρεσιών του TCP (προγραμματισμός υποδοχών –socket programming).

Διαδιεργασιακή επικοινωνία με Sockets



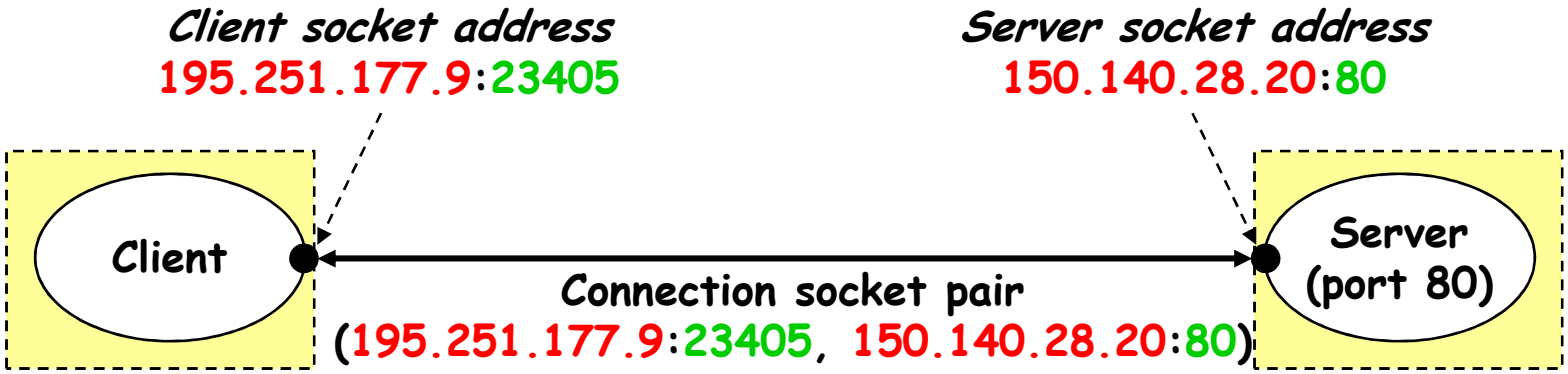
- Διεπαφή (προγραμματιστική αφαίρεση) που παρέχει το ΛΣ για την πρόσβαση στο υποσύστημα επικοινωνίας (π.χ. UDP/TCP)
- Υλοποιούμε I/O στο δίκτυο μέσω των sockets περίπου όπως υλοποιούμε I/O στο δίσκο μέσω των περιγραφέων αρχείων.

Επικοινωνιακή σύνδεση (γενικά)

- Η επικοινωνιακή σύνδεση μεταξύ δύο διεργασιών περιγράφεται με μια πεντάδα της μορφής: (*πρωτόκολλο, τοπική-διεύθυνση, τοπική-διεργασία, απομακρυσμένη-διεύθυνση, απομακρυσμένη-διεργασία*).
  - Το *πρωτόκολλο* αναφέρεται στο σύνολο των κανόνων που διέπουν την επικοινωνία (π.χ. UDP, TCP).
  - Η *τοπική-διεύθυνση* και *απομακρυσμένη-διεύθυνση*, προσδιορίζουν τις διευθύνσεις επιπέδου δικτύου (π.χ. IP) των Η/Υ, στους οποίους εκτελούνται οι διεργασίες.
  - Η *τοπική-διεργασία* και *απομακρυσμένη-διεργασία*, προσδιορίζουν την ταυτότητα των διεργασιών που θα επικοινωνούν (στα UDP, TCP έναν 16-bit ακέραιο αριθμό, ο οποίος αναπαριστά την θύρα (port number) της διεργασίας)
- Το socket περιγράφει το ένα άκρο (endpoint) μιας επικοινωνιακής σύνδεσης.
  - Περιλαμβάνει την τριάδα (*πρωτόκολλο, διεύθυνση, αριθμό θύρας*) για το ένα άκρο της επικοινωνιακής σύνδεσης.
- Η διαδιεργασιακή επικοινωνία με sockets απαιτεί την μετάδοση μηνυμάτων μεταξύ ενός socket στη μια διεργασία και ενός socket στην άλλη διεργασία.

TCP σύνδεση

- Προσδιορισμός της διεύθυνσης του host στο δίκτυο
  - IP address
- Ταυτότητα διεργασίας
  - Port number
- Το ζεύγος (*IP address* , *Port number*) ορίζει ένα "*socket-address*"



Client host address
195.251.177.9

Server host address
150.140.28.20

Η θύρα 23405 έχει δοθεί από το OS

Η θύρα 80 είναι μια καλά καθορισμένη θύρα για πρόσβαση σε Web servers

Sockets στην Java

Συλλογή από κλάσεις και interfaces στο πακέτο java.net

Βασικές κλάσεις για TCP επικοινωνία:

- **Socket**: client-side & server-side
- **ServerSocket**: server-side

- Βασικές κλάσεις για UDP επικοινωνία :
  - **DatagramSocket**
  - **DatagramPacket**

Socket class methods

- Socket(InetAddress address, int port)
  - Δημιουργεί ένα TCP stream socket και το συνδέει στο port και την IP διεύθυνση που καθορίζονται ως παράμετροι.
- Socket(String host, int port)
  - Δημιουργεί ένα TCP stream socket και το συνδέει στο port του host, του οποίου το όνομα καθορίζεται στις παραμέτρους.
- InputStream getInputStream()
  - Επιστρέφει ένα stream εισόδου για αυτό το socket.
- OutputStream getOutputStream()
  - Επιστρέφει ένα stream εξόδου αυτού του socket
- InetAddress getLocalAddress()
  - Επιστρέφει την τοπική διεύθυνση με την οποία το socket αρχικοποιήθηκε.
- int getLocalPort()
  - Επιστρέφει τον τοπικό αριθμό θύρας με τον οποίο το socket αρχικοποιήθηκε.
- InetAddress getInetAddress()
  - Επιστρέφει την διεύθυνση του απομακρυσμένου Η/Υ στην οποία το τοπικό socket είναι συνδεδεμένο.
- int getPort()
  - Επιστρέφει τον αριθμό θύρας της απομακρυσμένης διεργασίας στο socket της οποίας το τοπικό socket έχει συνδεθεί.
- void close()
  - Κλείνει αυτό το socket (απελευθερώνει το socket description).

ServerSocket class methods

- ServerSocket(int port)
  - Δημιουργεί ένα server socket στη θύρα της οποίας ο αριθμός καθορίζεται απ' την παράμετρο. Αν ο αριθμός είναι το 0, τότε αφήνεται στο σύστημα η επιλογή μιας ελεύθερης (μη χρησιμοποιούμενης) θύρας.
- ServerSocket(int port, int backlog)
  - Δημιουργεί ένα server socket στην θύρα της οποίας ο αριθμός καθορίζεται απ' την πρώτη παράμετρο, ενώ επιτρέπει τόσες συνδέσεις να περιμένουν στην ουρά, μέχρι να γίνουν αποδεκτές, όσες η δεύτερη παράμετρος.
    - Εαν οι Pending αιτήσεις >backlog, το ειδοποιείται ο αιτών και το ΛΣ αναλαμβάνει την επαναυποβολη διάφανα προς την εφαρμογή
    - Ο πρώτος constructor χρησιμοποιεί κάποιο default backlog size
- Socket accept()
  - Ακούει για πιθανή αίτηση σύνδεσης ενός πελάτη με τον server και την αποδέχεται. Ένα νέο socket αντικείμενο δημιουργείται και χρησιμοποιείται για την ανταλλαγή μηνυμάτων με τον πελάτη.
- void close()
  - Κλείνει αυτό το server socket.
- InetAddress getInetAddress()
  - Επιστρέφει την τοπική διεύθυνση αυτού του server socket.
- int getLocalPort()
  - Επιστρέφει τον αριθμό της θύρας που αυτό το server socket «ακούει».

I/O μέσω Streams

- Stream = γενικός μηχανισμός για I/O δεδομένων
- Input Stream = αντικείμενο που μια εφαρμογή μπορεί να χρησιμοποιήσει για να διαβάσει μια ακολουθία δεδομένων
- Output Stream: αντικείμενο που μια εφαρμογή μπορεί να χρησιμοποιήσει για να γράψει μια ακολουθία δεδομένων
- Και στις δυο περιπτώσεις τα data μεταφέρονται ως bytes

Java Streams

- Συλλογή από stream κλάσεις και interfaces στο πακέτο **`java.io`**
- Στην κορυφή της ιεραρχίας ορίζονται δυο abstract κλάσεις:
  - `InputStream`
  - `OutputStream`

Διάφορες κλάσεις τύπου Stream

- Byte Array Streams
  - `ByteArrayInputStream & ByteArrayOutputStream`
    - `I/O γίνεται σε πίνακες`
- File Access Streams
  - `FileInputStream & FileOutputStream`
    - `I/O γίνεται σε αρχεία (raw bytes)`
- Filter Streams (wrappers σε κλάσεις Streams):
  - `FilterInputStream & FilterOutputStream`
    - `BufferedInputStream`
    - `BufferedOutputStream`
    - `DataInputStream`
      - Μετατρέπει τα δεδομένα στον γηγενή τύπο τους καθώς τα λαμβάνει
    - `DataOutputStream`
      - Γράφει πρωτογενείς τύπους δεδομένων σε ένα output stream

Βήματα εφαρμογής TCP client

Λειτουργικό βήμα	Περιγραφή
Socket()	Αρχικοποίηση του socket αντικειμένου και σύνδεση στον server
getInputStream() και getOutputStream()	Σύνδεση του socket με τον στάνταρ μηχανισμό εισόδου/εξόδου στη Java, δηλαδή τα streams
read() και write() readUTF() και writeUTF()	Ανάγνωση και εγγραφή στο συνδεδεμένο με το socket stream
close()	Κλείνει το socket και απελευθερώνεται ο αντίστοιχος πόρος του συστήματος

Βήματα εφαρμογής TCP server

Λειτουργικό βήμα	Περιγραφή
ServerSocket()	Αρχικοποίηση του listening socket αντικειμένου
accept()	Αναμονή και αποδοχή σύνδεσης νέου πελάτη
new Thread	Δημιουργία καινούργιου thread για την εξυπηρέτηση ενός πελάτη (αυτό το βήμα ακολουθείται ανάλογα με τις προδιαγραφές της εφαρμογής)
getInputStream() και getOutputStream()	Το καινούργιο socket που επιστρέφεται από την accept συνδέεται με τον στανταρ μηχανισμό εισόδου/εξόδου στη Java, δηλαδή τα streams
read() και write() readUTF() και writeUTF()	Ανάγνωση και εγγραφή στο συνδεδεμένο με το socket stream
close()	Κλείνει το socket και απελευθερώνεται ο αντίστοιχος πόρος του συστήματος

Περίγραμμα εφαρμογής

```
Socket s = new Socket(host, serverPort);



…


DataInputStream in = new
     DataInputStream(s.getInputStream());
DataOutputStream out = new
     DataOutputStream(s.getOutputStream());

…

out.write(…);

…

in.read(…);
```

client-side

```
ServerSocket listenSocket = new
          ServerSocket(serverPort);

…
Socket s = listenSocket.accept();

…
DataInputStream in = new
     DataInputStream(s.getInputStream());
DataOutputStream out = new
     DataOutputStream(s.getOutputStream());

…

in.read(…);

…

out.write(…);
```

server-side

Μετάδοση αντικειμένων

- Για την αποστολή/λήψη δεδομένων βασικών τύπων (integer, long, char, boolean, byte, float, double) μπορούν να χρησιμοποιηθούν οι αντίστοιχες μέθοδοι των κλάσεων DataInputStream/DataOutputStream (π.χ. writeInt()/readInt())
- Για strings μπορούν να χρησιμοποιηθούν οι μέθοδοι writeUTF()/readUTF().
- Για αντικείμενα κλάσεων, οι κλάσεις θα πρέπει να υλοποιούν το Serializable interface και θα πρέπει να χρησιμοποιηθούν μέθοδοι των κλάσεων ObjectInputStream/ObjectOutputStream
  - writeObject()
  - readObject()