Διάλεξη 23: Εισαγωγή στην Δικτύωση (Networking)

Στην ενότητα αυτή θα μελετηθούν τα εξής επιμέρους θέματα:

- Διευθύνσεις και Θύρες
- Sockets και ServerSockets

Διδάσκων: Παναγιώτης Ανδρέου

Εισαγωγή

- Η επικοινωνία μεταξύ δύο υπολογιστών στο διαδίκτυο επιτυγχάνεται με δύο πρωτόκολλα:
  - Transmission Control Protocol (TCP)
    - Αξιόπιστη ροή δεδομένων
    - Η σειρά των δεδομένων παίζει ρόλο
  - User Datagram Protocol (UDP)
    - Μη-αξιόπιστη ροή δεδομένων (πιο αποδοτική)
    - Η σειρά των δεδομένων δεν παίζει ρόλο

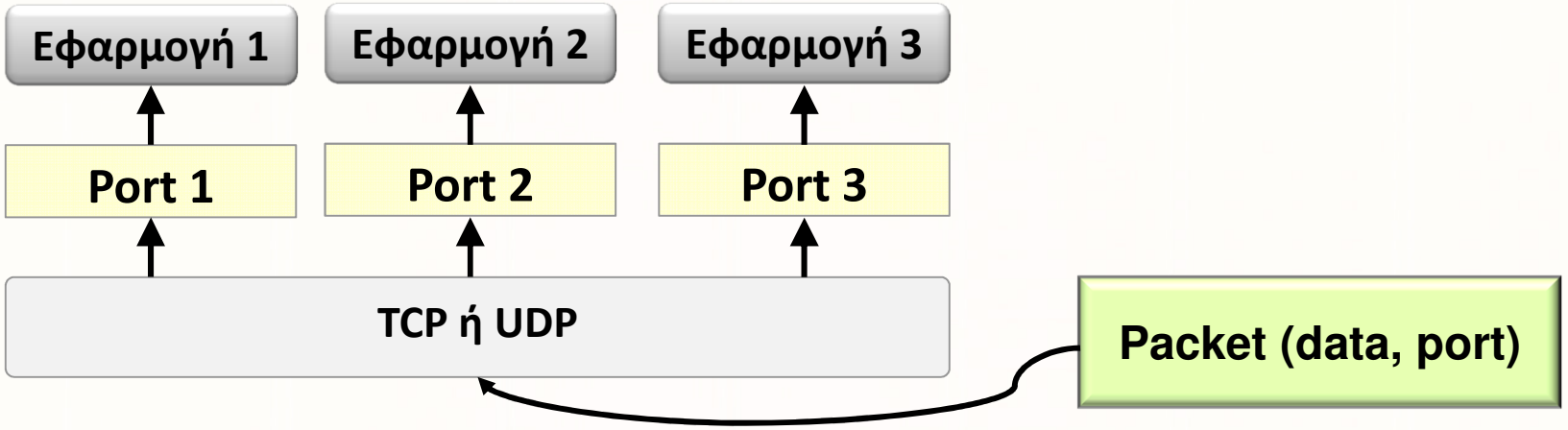
- Η βιβλιοθήκη java.net περιέχει κλάσεις για την διεκπεραίωση της δικτύωσης

Διευθύνσεις και Θύρες (Addressing and Ports)

- Η διεύθυνση κάποιου υπολογιστή στο διαδίκτυο είναι μία μοναδική διεύθυνση (π.χ., 109.105.224.164) χρησιμοποιώντας το **Internet Protocol (IPv4 ή IPv6)**
- **Ερώτηση:** Τι συμβαίνει αν έχουμε 3 υπολογιστές στο ίδιο σπίτι;
- **Απάντηση:** Χρησιμοποιούμε ένα διακομιστή (router) ο οποίος έχει μοναδική διέυθυνση στο διαδίκτυο αλλά δημιουργεί επίσης ένα τοπικό δίκτυο στο σπίτι μας με τοπικές διευθύνσεις
  - Παράδειγμα:
    - Router: External IP 109.105.224.164/Internal IP 192.168.0.1
    - PC1: 192.168.0.2
    - PC2: 192.168.0.3
    - PC3: 192.168.0.4
- **Ερώτηση:** Τι συμβαίνει αν κάποια δεδομένα που φθάνουν σε ένα υπολογιστή προορίζονται για μία εφαρμογή Α και κάποια άλλα για άλλη εφαρμογή Β;

Διευθύνσεις και Θύρες (Addressing and Ports) (συν.)

- **Απάντηση:** Επιτυγχάνεται με τη χρήση ξεχωριστών θηρών (ports)



- Παραδείγματα ports:
  - 21: FTP
  - 22: SSH
- Δικτύωση με κλάσεις στην Java
  - Διευθύνσεις (URLs): URL, URLConnection
  - Επικοινωνία με TCP: Socket, ServerSocket classes
  - Επικοινωνία με UDP: DatagramPacket, DatagramSocket, MulticastSocket

Uniform Resource Locator (URL)

- Ορισμός: **URL είναι ακρώνυμο για *Uniform Resource Locator* και είναι μία αναφορά (διεύθυνση) σε ένα πόρο στο διαδίκτυο**
- Συστατικά ενός URL
  - Host Name: Το όνομα (διεύθυνση) της μηχανής
  - Filename: Το μονοπάτι για κάποιο αρχείο
  - Port Number: Η θύρα επικοινωνίας
  - Reference: Μία αναφορά για μία συγκεκριμένη τοποθεσία (anchor)

- Δημιουργία ενός URL (κλάσεις URL και URI)

```
URL myURL = new URL("http://example.com/test/");

URI myURI = new URI("http", "example.com", "/test/", "");
URL myURL = uri.toURL();
```

- Χρήσιμες Μεθόδοι:
  getProtocol, getHost, getPort, getPath, getFile

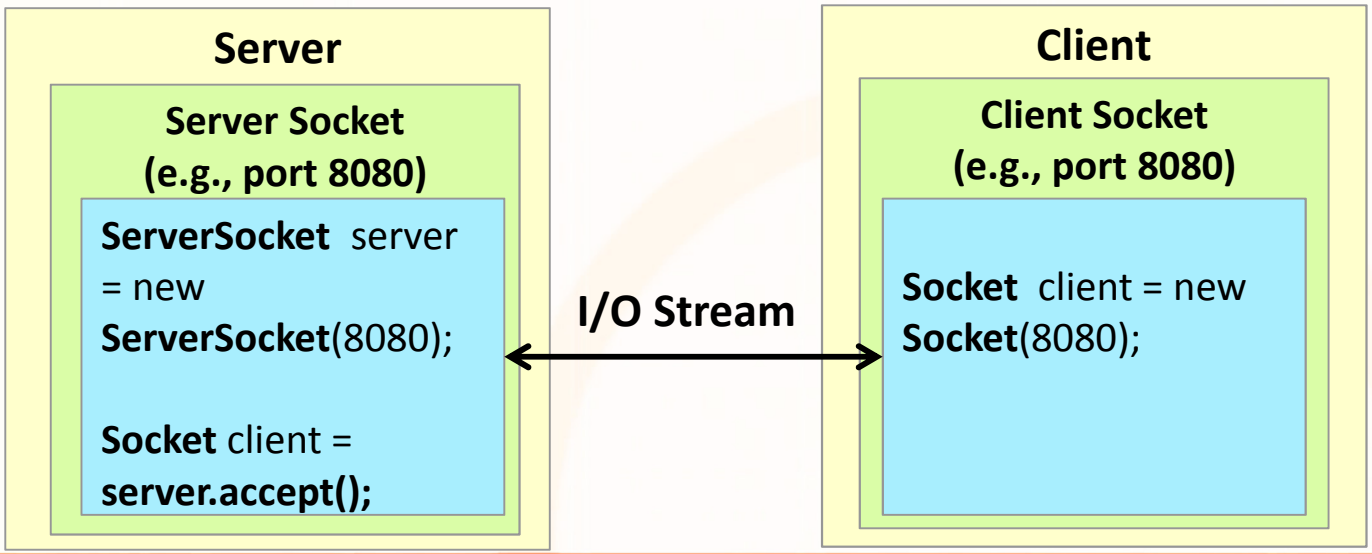
Χρήση Ροών με URL

- Επιτρέπεται η απευθείας χρήση java.io.InputStream από ένα URL
- Παράδειγμα (εκτύπωση του HTML κώδικα μίας ιστοσελίδας):

```
URL cs = null;
try {
cs = new URL("http://www.cs.ucy.ac.cy/index.php");
BufferedReader in = new BufferedReader(new
                    InputStreamReader(cs.openStream()));
String inputLine;
while ((inputLine = in.readLine()) != null)
            System.out.println(inputLine);
        in.close();
}
catch (MalformedURLException e) { }
catch (IOException e) { }
```

Μοντέλο Πελάτη/Εξυπηρετητή (Client/Server model)

- **Εξυπηρετητής (server)**
  - «Ακούει» συνεχώς για συνδέσεις από πελάτες με ServerSocket(s) συγκεκριμένα ports
  - Πρέπει να τρέχει πριν να δημιουργήσει σύνδεση ο πελάτης
- **Πελάτης (client)**
  - Συνδέεται με Socket(s) σε υφιστάμενους εξυπηρετητές
- **Όταν επιτευχθεί η επικοινωνία, μετά τα δεδομένα μεταδίδονται με I/O Streams όπως για τα αρχεία**



Παράδειγμα: Client/Server

Εξυπηρετητής (server)

```
try {
  // Create a server socket
  ServerSocket serverSocket = new
                          ServerSocket(8000);
  print("Server started ");
  // Listen for a connection request
  Socket socket = serverSocket.accept();

  // Create data input and output streams
  DataInputStream inputFromClient = new
    DataInputStream(
      socket.getInputStream());
  DataOutputStream outputToClient = new
    DataOutputStream(
      socket.getOutputStream());
  while (true) {
    // Receive radius from the client
    double radius =
      inputFromClient.readDouble();
    // Compute area
    double area = radius * radius * Math.PI;
    // Send area back to the client
    outputToClient.writeDouble(area);
  }
catch (IOException ex) { }
```

Πελάτης (client)

```
private DataOutputStream toServer;
private DataInputStream fromServer;
try {
  // Create a socket
  Socket socket = new Socket("localhost",
                                        8000);
  // Create an input stream to
  // receive data from the server
  fromServer = new DataInputStream(
  socket.getInputStream());
  // Create an output stream to
  // send data to the server
  toServer = new DataOutputStream(
    socket.getOutputStream());
}
catch (IOException ex) { }
```

Πληροφορίες για τους πελάτες (κλάση InetAddress)

- **Πληροφορίες για τους πελάτες** μπορούν να αποκτηθούν με την κλάση **InetAddress**
- Η κλάση InetAddress μοντελοποιεί μία διεύθυνση IP
- Δημιουργία και αρχικοποίηση
  InetAddress inetAddress = socket.getInetAddress();

- Παράδειγμα
  Εκτύπωση ονόματος και IP διεύθυνσης

```
System.out.println("Client's host name is " +
  inetAddress.getHostName());
System.out.println("Client's IP Address is " +
  inetAddress.getHostAddress());
```

Εξυπηρέτηση Πολλαπλών Πελατών

- Επιτυγχάνεται με τη χρήση threads
- Για κάθε σύνδεση δημιουργείται ένα thread
- Παράδειγμα
- while (true) {

  Socket socket = serverSocket.accept();

  Thread thread = new ThreadClass(socket);

  thread.start();

  }

- Πιο σωστά με την χρήση εσωτερικών κλάσεων

  // Create a new thread for the connection
  HandleAClient task = **new** HandleAClient(socket);
  new Thread(task).start();

Inner class

```
class HandleAClient implements Runnable {

  private Socket socket; // A connected socket

  public HandleAClient(Socket socket) {
    this.socket = socket;
  }

  public void run() {
    try {
      // Create data input and output streams
      DataInputStream inputFromClient =
        new DataInputStream(
      socket.getInputStream());

      DataOutputStream outputToClient =
        new DataOutputStream(
      socket.getOutputStream());

      // Continuously serve the client
      while (true) {
        // Receive radius from the client
        double radius = inputFromClient.readDouble();

        // Compute area
        double area = radius * radius * Math.PI;

        // Send area back to the client
        outputToClient.writeDouble(area);
      }
    }
    catch(IOException e) { }
  }
}
```