'Hands-on' tools for internet programming

DD1335 (gruint10)

Serafim Dahl

serafim@nada.kth.se
What is this course about?

- Providing tools for hands-on internet programming
- There are only 9 lectures – do show up, please!
Welcome

Lectures are about

▶ Basics on the internet
  ▶ Protocols, addresses, hosts
  ▶ HTML, markup
  ▶ Internet connections, servers (Java)
▶ Server-Side Internet Programming
  ▶ CGI, Servlets (Java)
  ▶ Java Server Pages (JSP) and other scripting (ASP)
  ▶ 3-tier systems: JDBC (Java-SQL)
▶ Client-Side Internet Programming
  ▶ Javascript
  ▶ CSS
  ▶ Applets (Java) and maybe some other technique(s)
▶ Other Issues
  ▶ XML, Web Services, Semantic Web
  ▶ PHP and other scripting languages
Labs

- Principles: wide, not deep.
- A lot to do, but all easy, mostly with a template to start from
- Net and programming basics (Lab 1), Net connections (Lab 2)
- Server side (Lab 3, Lab 4), Client side (Lab 5)

Projects

- You define your projects.
- You form the project groups.
- Send me an email with a 5-line project idea and names of group members
- The project must be an interactive WWW system. Simple HTML pages are not enough
- Required: server-side programming (e.g. shopping baskets, booking systems, resource allocation)
- Required: JavaScript (e.g. client-side checking of user input, etc)
- Not much technical complexity, but a high editorial quality (good layout, including CSS), making the best of the Internet medium
- Make groups of 3 to 6 people
Welcome

Administration

- Course codes: gruint10
- Register on the course (for admin of course element results):
  Log in to some computer
  Start a web browser and connect to
  https://rapp.nada.kth.se/rapp and login
  Activate the course instance ”gruint10”
- To get info apart from that on the web
  course join gruint10
Introduction to the internet

Content
A little on:
- network concepts
- web concepts
- internet addresses
- sockets

References:
- Harold: Java Network Programming
- Hall: Core Web Programming
- Deitel, et al: Internet and the World Wide Web How to Program
- Ince: Developing Distributed and E-Commerce Applications
Programming network applications

- Why network applications?
  - Alongside the technical "evolution", communication between application and also between parts of applications residing on different computer become more and more common
  - Examples of asynchronously communicating applications: web browsers, e-mail, news.
  - Some other examples: Distributed databases, sound, radio, video and internet telephony.

- Need for applications where the participants are aware of each others:
  - Shared bulletin boards, whiteboards, shared word processors, control systems (eg. robots) and (not the least) games (like runescape and world of warcraft).

- There is support in the networks, where we will look closer on the internet.
Programming network applications

- Large amounts of internet sites
  - Auctions, advertising, commerce, portals with collections of sites concerning business, music, film, software, info, reports of various kinds books, search engines, education, . . .

- Kinds of application programs
  - E-mail
  - News
  - Web based databases
  - Client-server, peer-to-peer
  - Telephone
  - Video
  - . . .
Network basics

Networks

A network is in this respect a collection of interconnected computers and/or other kinds of equipment

Terminology:

- **node**, a machine that is connected to the network (computer, printer, bridge, vending machine, ...)
- **host**, a fully autonomous computer connected to the network
- **address**, each node has a unique address (a number of bytes)
- **packet**, modern networks are packet based, meaning that the information is broken down to and sent as small chunks, each chunk of information handled separately.
- **protocol**, rules, specifying how to perform communication
Internet

Internet is the most know and most wide spread network.

- Designed to be robust (errors are unusual)
- First version 1969, ARPANET, designed by ARPA, a DoD unit.
- 1983 there were 562 computers on the ARPANET
- 1986 there were 5000 computers
- 1987 – 28000,
- 1989 – 100000,
- 1990 – 300000,
- 2009 – 1.67 billion (a rough estimate on June 30)
Layers

A network is built as a set of layers

- Application programs work mainly in the upper layer
- Eventually in the transport layer (in distributed applications)
- Other layers are normally of no concern
IP, TCP, UDP

- **IP**, Internet Protocol
  the network layer protocol (the reason for the name "Internet")

- **TCP**, Transport Control Protocol
  a connection based protocol which insures a correct data exchange between two nodes

- **UDP**, User Datagram Protocol
  a protocol which allows the transmission of independant packets from one node to another with no guarantee concerning delivery or order of delivery
IP address, DNS

- **IP address**. Each machine is identified by a unique 4-byte number
  - Many computers have a fixed number, others get a dynamically assigned number at connection time
  - 1995 the use of the internet "exploded" and as there are not enough 4-byte numbers (you get a "lousy" \(2^{32} = 4294967296\) addresses), IPv6 was created giving \(2^{128} = 340282366920938463463374607431768211456\) adresses. Ought to be enough for some time . . .

- **DNS, Domain Name Server**
  - IP-addresses are hard to remember and thus DNS was created to allow symbolic (textuel) names that are looked up and translated to IP-addresses
  - Eg.: `www.nada.kth.se` is translated to `130.237.225.40`
Every computer with an IP-address has 65536 logical ports for communication over the internet.

Some are reserved
- ports number 0-1023 are reserved (for what and by whome may be seen in the file `/etc/services` (on UNIX/Linux)
- eg. the following:
  - port 7 for echo
  - port 20-21 for ftp
  - port 23 for telnet
  - port 25 for smtp (send e-mail)
  - port 80 for http (web server)
  - port 110 for POP3 (read e-mail)
Intranet

There are other networks with the same structure. Local networks are usually called *intranet*. They may link to the internet with special "bridges". Sometimes the bridge uses filtering devices to restrict the data traffic between the networks.
The client-server model

Today, the *client-server* model is the prevailing when constructing distributed, cooperating application programs.

- a client asks a server for a service (as eg. information about the time)
- a server accomplishes the corresponding task and delivers the service (like sending time info, sending a file from its local file system, eg. a web page)
- both following a protocol that enables asking for and providing services over the network
The client-server model . . .

- Not all kinds of application programs fit into the client-server model. Some act simultaneously as both client and server and, if both "ends" of a communication do, that communication is called "peer-to-peer". Eg:
  - a shared editor
  - a game (runescape, world of warcraft, . . .)
  - a telephone connection
Some internet standards have been developed publicly already from the prototype stage.

Their protocols are publicly accessible on the internet.

These protocols fit into the following categories:

- Mandatory – each host *must* implement them, eg. IP
- Recommended – that *ought to be* implemented, eg. TCP, SMTP, UDP, TelNet, …
- Optional, like MIME
- Restricted, that are necessary only in special cases
- Not recommended, that should not be implemented
- Historical (obsolete, deprecated)
- Informative, that may have been constructed outside the RFC but still are useful without delivering an established protocol
HTTP, HTML, XHTML, MIME

- HTTP, HyperText Transfer Protocol,
  - a standard protocol for the communication between a web server and a web client (web browser)
- HTML, HyperText Markup Language
  - the first generation standard language for the construction of web pages, a subset to SGML with extra error tolerance
  - XHTML, eXtensible HTML,
    - second generation language for the construction of web pages, HTML as a strict subset to XML
- MIME, Multipurpose Internet Mail Extension
  - an open standard that determines how multimedia objects are to be transmitted by e-mail
Network basics

URL, URI, URN

▶ URI, Uniform Resource Identifier
  ▶ define how to uniquely identify a resource on the internet
  ▶ is divided into the subgroups URL and URN

▶ URL, Uniform Resource Locator
  ▶ a reference for an address on the internet
  ▶ looks like: protocol://host[:port]/path/file[#section]
  ▶ eg:
    http://www.csc.kth.se:8080/dd1335/gruint09/labs/#lab2

▶ URN, Universal Resource Name
  ▶ a "pointer" to a resource without specifying its exact position, eg. the search
    for a certain kind of documents may deliver the set of URLs (the positions of
    all the documents)
SGML & HTML

- SGML, Standard Generalized Markup Language
  - Was created in the 1970s
  - Describes the semantics of a text rather than its presentation

- HTML, HyperText Markup Language
  - Was created from SGML early in the 1990s
  - Describes how to present a text rather than its semantics
  - Is "lingua franca" for presentation of hypertext on the web
HTTP

- HTTP, HyperText Transport Protocol
  - a standard describing how a web client and a web server should exchange data
  - uses MIME to decode data
  - uses TCP/IP for the transmission of data
  - The client sends a message once the communication has been established
    
    eg. GET /index.html HTTP/1.1
  - the web server responds by sending the file index.html to the client
MIME, Multipurpose Internet Mail Extension

- an open standard for how to send multimedia objects by e-mail
- denotes the type of data that is transmitted, eg.
  - text/plain, text/html
  - news
  - application/postscript, application/pdf
  - zip
  - image/gif, image/jpeg, image/tiff, image/x-bitmap
  - audio/basic, audio/mpeg
  - video/mpeg, video/quicktime, video/x-msvideo