TDDD97 - Web Programming
Introduction

Sahand Sadjadee
Dept. of Computer and Information Science
Linköping University

Outline

- Course overview
  - Lectures
  - Programming assignments
  - Lab sessions/Camedin
- Background
  - Internet
  - Web
- Architecture of web applications

Course overview

Lectures

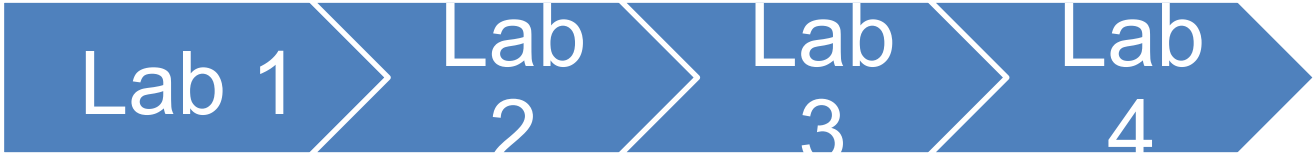
- Lectures(4)
  - Introduction
  - Client-side development
  - Server-side development
  - Client-server communication
- Lessons(3)
  - Step-by-step development of an example web application

Lectures and Lessons timeline

1. Lecture 01 -> **Introduction(The course overview, web and internet background, architecture)**
2. Lecture 02 -> **client-side development(HTML, CSS, JavaScript, Selenium, RWD)**
3. Lesson 01 -> **Getting started with "The Phonebook", Going through lab 01**
4. Lecture 03 -> **server-side development(Python, Flask, SQL, Rest API, Validation, Security)**
5. Lesson 02 -> **Developing The backend of Phonebook, Going through lab 02**
6. Lecture 04 -> **Client-Server communication(AJAX, Websockets, JSON, Project Overview)**
7. Lesson 03 ->**Completing Phonebook, Going through lab 03**

Programming Assignments

- Lab 1: **Client-side technologies: HTML, CSS, JavaScript**
- Lab 2: **Server-side technologies**
- Lab 3: **Client-server communication**
- Lab 4: **Mini-project**



Programming Assignments (cont.)

- No written exam => more lab work
- Expect to work hard for these credits
- If you for some reason feel uncomfortable about this, consider dropping this course *now*.

About the Workload

Comments from previous students

- “It was hard work but I really learned a lot!”
- “Working hard actually brings results.”
- “Much more work when you work alone.”

Some info...

- Groups of two students
  - Working alone is ok, too
- Sign up for the labs in the webreg system
- Most students in year 3–4 => we expect more
  - Independent work
  - Information search on your own
  - Work from less detailed instructions
  - Individual/independent/unique solutions

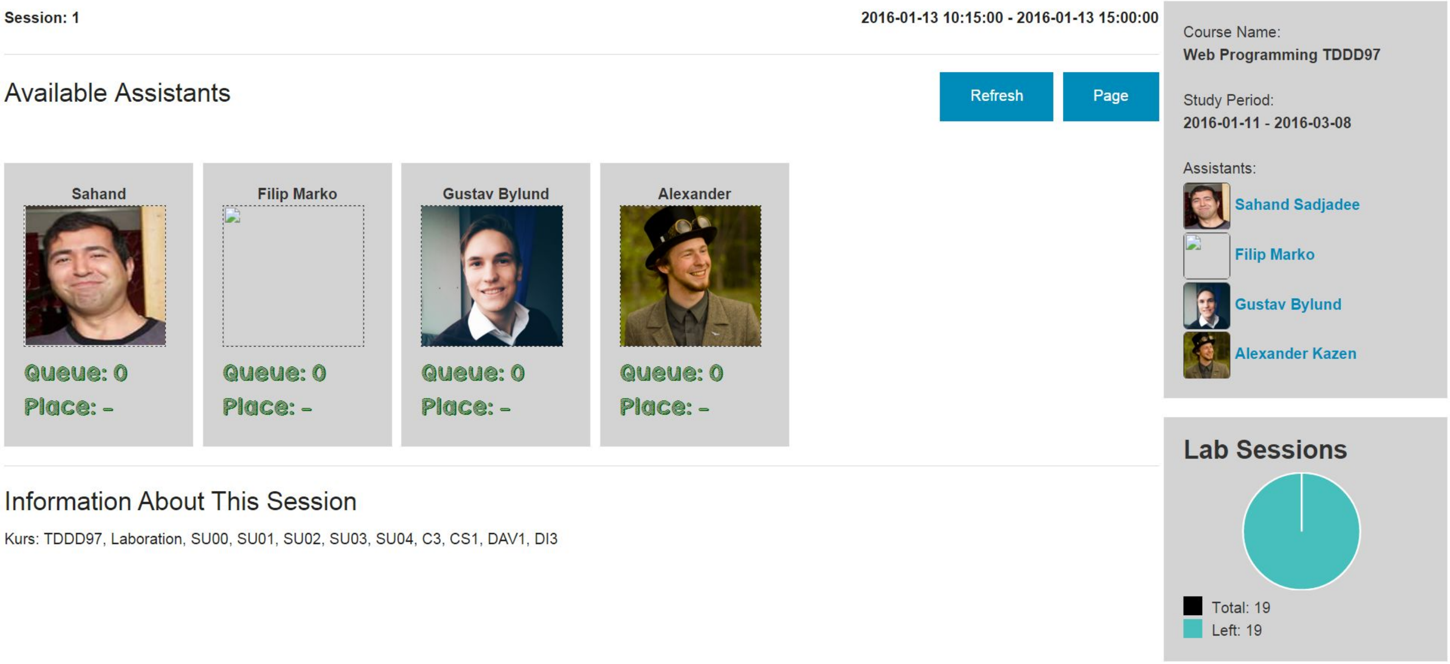
Lab sessions

- 22+2 assisted lab sessions.
- 7 rooms booked for every lab session.
- 78 available stations.
- You can also bring your own laptop.
- Each lab can be presented at any lab session
- There are two lab sessions in the end of course which are only dedicated for presentations.

Assistants

- **Dennis Persson**
- **Hampus Arvå Linhem**
- **Iliyas Jorio**
- **Jani Jokinen**
- **Mattias Lantz Cronqvist**
- **Sokrates Lamprou**
- **Tore Haglund**

Use Camedin for getting help from the available assistants



Session: 1

2016-01-13 10:15:00 - 2016-01-13 15:00:00

Available Assistants

Sahand	Filip Marko	Gustav Bylund	Alexander
Queue: 0	Queue: 0	Queue: 0	Queue: 0
Place: -	Place: -	Place: -	Place: -

Information About This Session

Kurs: TDDD97, Laboration, SU00, SU01, SU02, SU03, SU04, C3, CS1, DAV1, DI3

Course Name:
Web Programming TDDD97

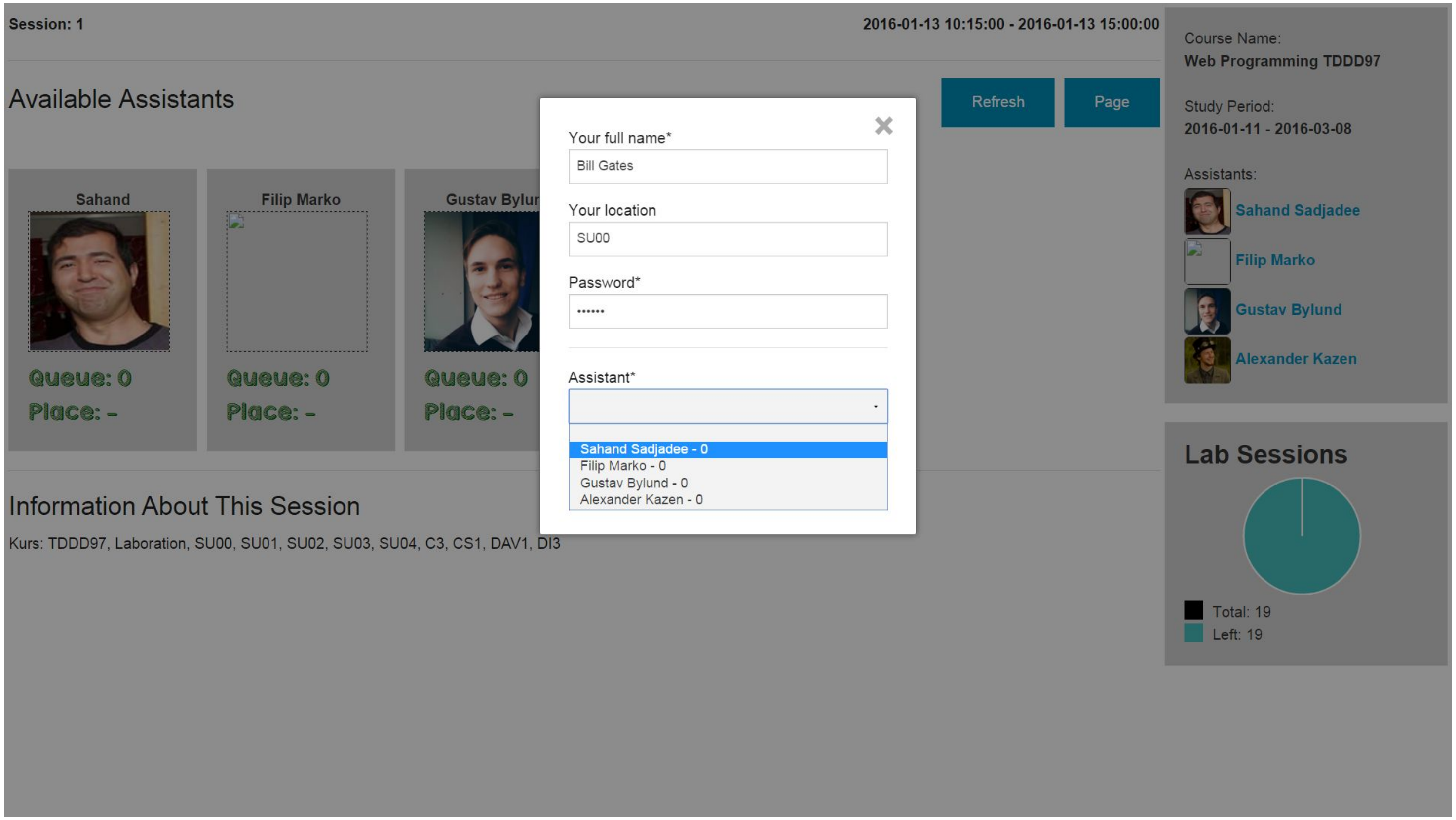
Study Period:
2016-01-11 - 2016-03-08

Assistants:
Sahand Sadjadee
Filip Marko
Gustav Bylund
Alexander Kazen

Lab Sessions

■ Total: 19
🟦 Left: 19

Cont.



Session: 1

2016-01-13 10:15:00 - 2016-01-13 15:00:00

Available Assistants

Sahand
Queue: 0
Place: –

Filip Marko
Queue: 0
Place: –

Gustav Bylur
Queue: 0
Place: –

Your full name*
Bill Gates

Your location
SU00

Password*
••••••

Assistant*

Sahand Sadjadee - 0
Filip Marko - 0
Gustav Bylund - 0
Alexander Kazen - 0

Information About This Session

Kurs: TDDD97, Laboration, SU00, SU01, SU02, SU03, SU04, C3, CS1, DAV1, DI3

Course Name:
Web Programming TDDD97

Study Period:
2016-01-11 - 2016-03-08

Assistants:
Sahand Sadjadee
Filip Marko
Gustav Bylund
Alexander Kazen

Lab Sessions

■ Total: 19
🟩 Left: 19

Continuation Project Course: TDDD27

- Given in study period VT2; you get 6 credits
- Goal
  - Learn Web programming for real
  - Learning how to find documentation
- Tasks
  - Project proposal approved by assistant (before starting)
  - Project presentation at seminar
- Reporting
  - Brief document describing the program (e.g., HTML page)
  - Source code and corresponding documentation
- Note: The fact that TDDD27 is the advanced course does not mean that TDDD97 is an easy course

Continuation Course: TDP024

- Enterprise Systems
- For the IP curriculum; given in Swedish
  - Other students can apply, but Swedish is a requirement
- Service Oriented Architectures (SOA)
- Web Services
- Labs
  - Literature search
  - SOA service
  - Exception handling
  - Security

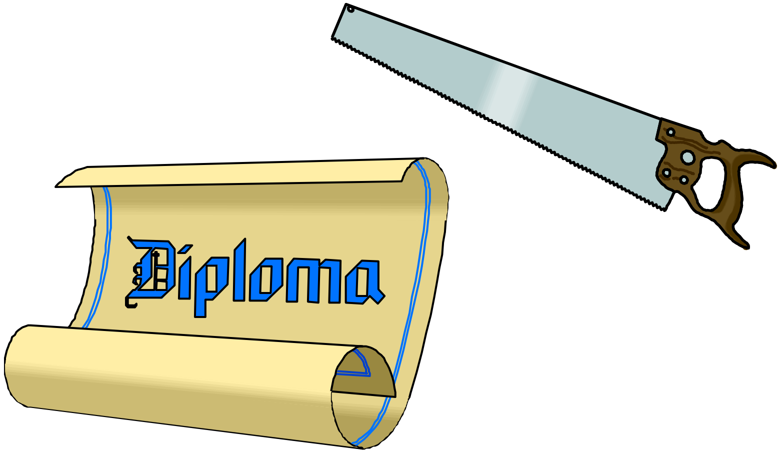
Important Dates

- Sign up for labs in webreg by January 01
  - Required in addition to course registration
- Labs completed and demonstrated by March 19
  - Demo for lab assistant
- Adjustments and code submitted by March 23
  - Course completed
- What if I miss the deadline date?
  - Second chance June 10-13

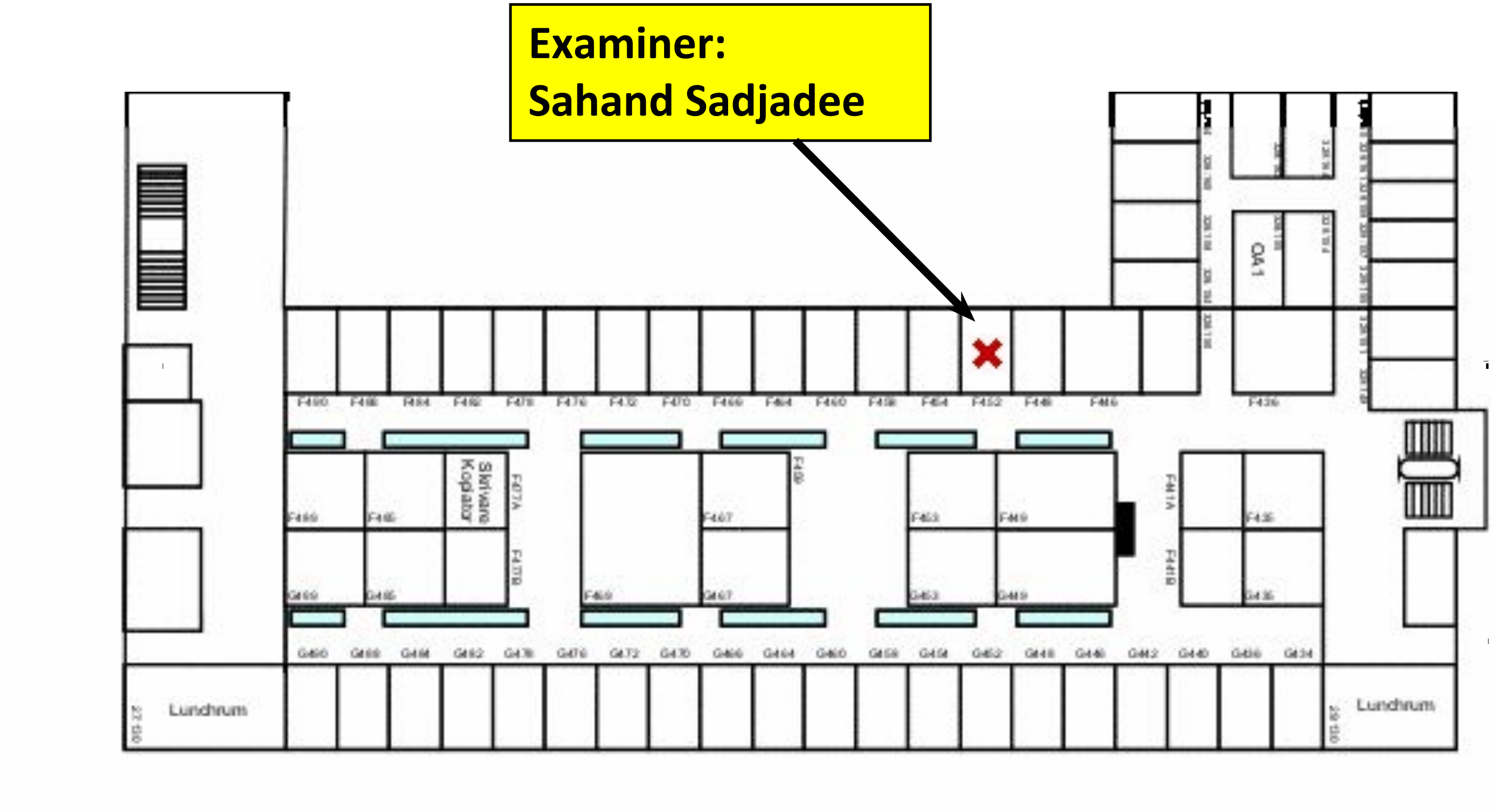
https://www.lith.liu.se/for-studenter/terminstider?l=sv

TDDD97 Examination

- Completed laboratory assignments (all labs)
- Grades: 3, 4, 5
  - Corresponding ECTS for foreign students
  - Grading is based on the mini-project (lab 4)



Office Map: Building E, 2nd floor



Examiner:
Sahand Sadjadee

Course home page

- Course home page: http://www.ida.liu.se/~TDD97/
  - Course information
  - Lecture slides
  - Programming-assignment instructions
  - Links to relevant web programming resources

Background

Brief History of the Internet

- Late 1960s: ARPAnet
  - Founded by ARPA
  - Original ARPAnet designed for 64 nodes
- Early 1980s: Internet
  - Several interconnected networks
  - Transmission Control Protocol (TCP), Internet Protocol (IP)
- Early applications:
  - Remote login: Telnet
  - E-mail: SMTP
  - File transfer: FTP
  - Discussion groups: NEWS

Internet (cont.)

- Tremendous growth of the Internet
  - 1983: 562 Computers
  - 1993: 1,313,000 Computers
  - 1994: 2,217,000 Computers
  - 1996: 14,352,000 Computers
- Problems:
  - Finding information
  - Not user friendly
  - Several different protocols and formats

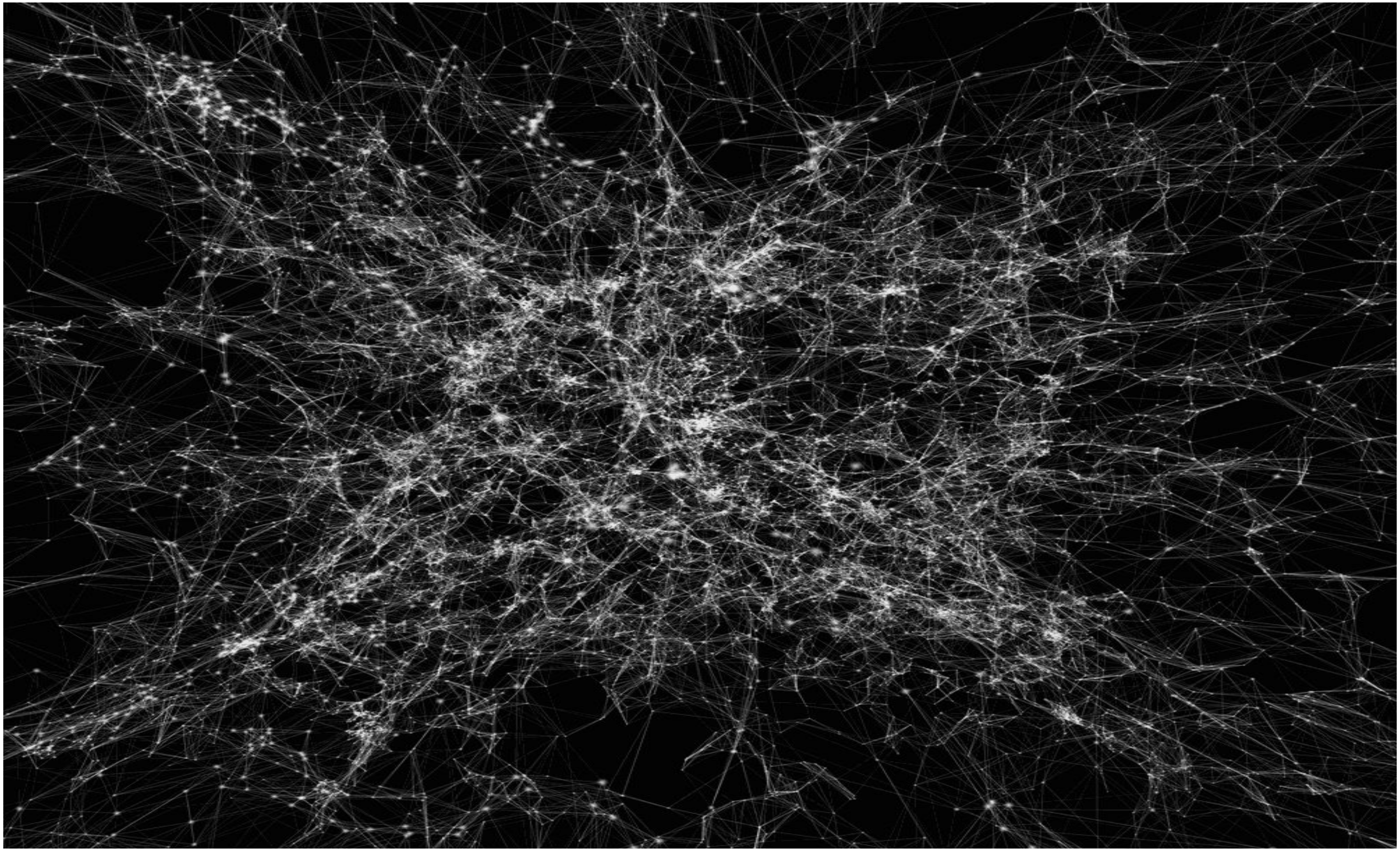
World Wide Web

The World Wide Web is an information space where documents and other web resources are identified by Uniform Resource Locators, interlinked by hypertext links, and can be accessed via the Internet.

Year: 1989/1990
Inventor: Tim Berners-Lee, Robert Cailliau

World Wide Web

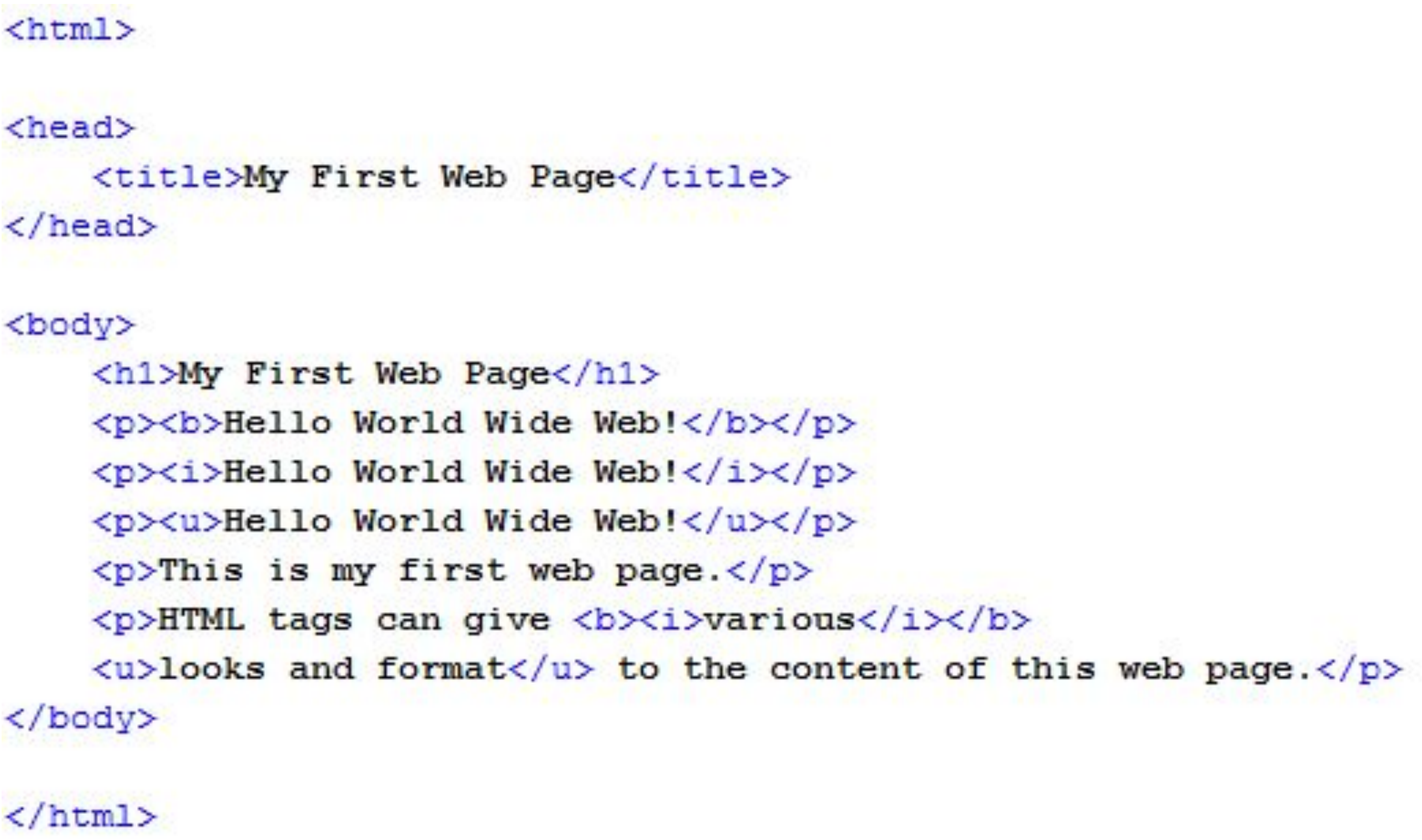
The Cosmic Web Paper by the Barabasi Lab



World Wide Web

- Just another Internet application
- Graphical interface to the Internet
- Protocols:
  - Uniform Resource Locator (URL)
  - HTTP, HTTPS
- HTML
  - Hypertext markup language
  - Based on Standard General Markup Language (SGML)
  - Tags
  - Hyperlinks
  - Graphics
  - Forms and CGI scripts

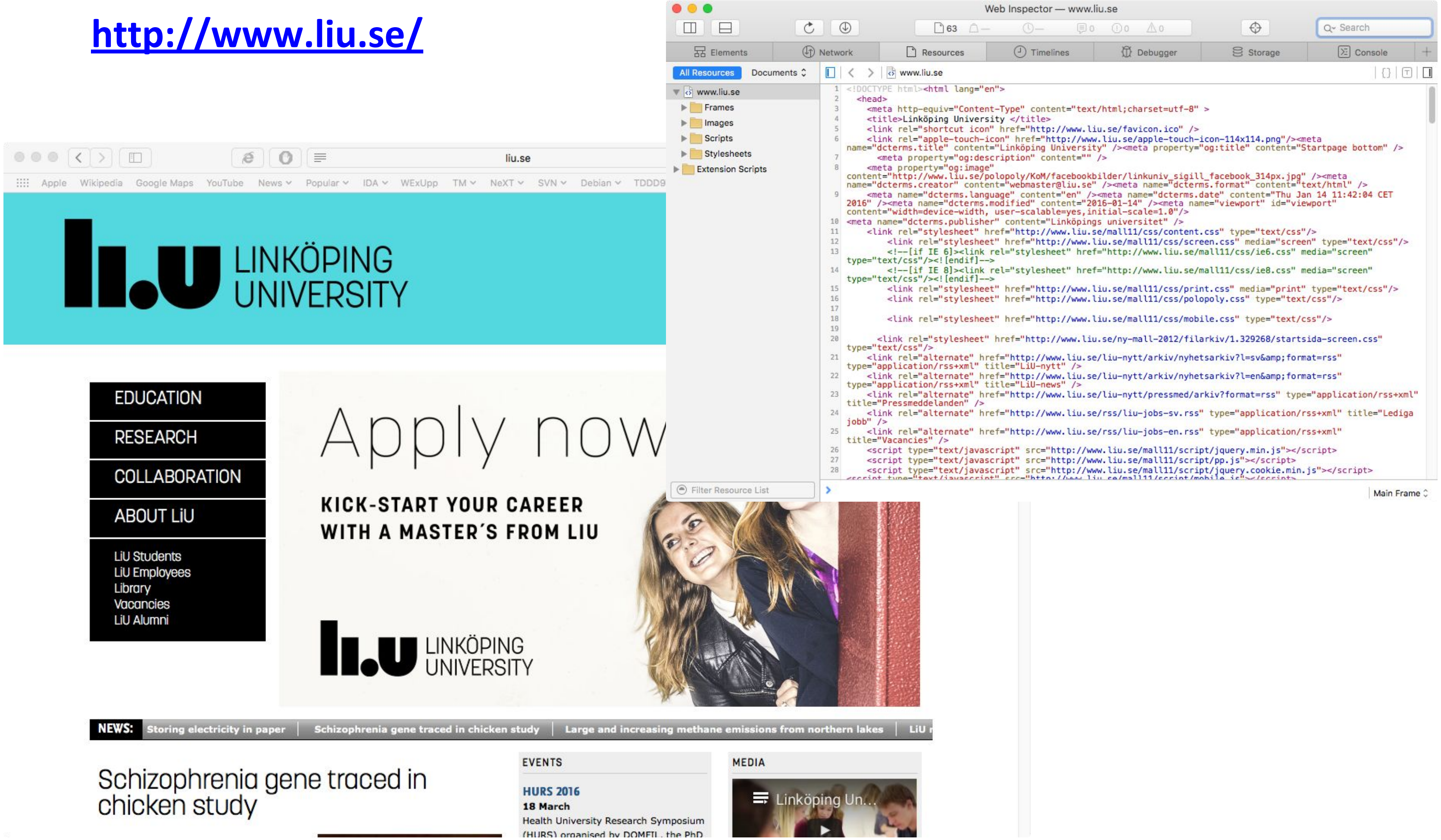
Sample HTML in a static web page



```
<html>

<head>
    <title>My First Web Page</title>
</head>

<body>
    <h1>My First Web Page</h1>
    <p><b>Hello World Wide Web!</b></p>
    <p><i>Hello World Wide Web!</i></p>
    <p><u>Hello World Wide Web!</u></p>
    <p>This is my first web page.</p>
    <p>HTML tags can give <b><i>various</i></b>
    <u>looks and format</u> to the content of this web page.</p>
</body>

</html>
```

Sample HTML

http://www.liu.se/



```
<!DOCTYPE html><html lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" >
    <title>Linköping University </title>
    <link rel="shortcut icon" href="http://www.liu.se/favicon.ico" />
    <link rel="apple-touch-icon" href="http://www.liu.se/apple-touch-icon-114x114.png"/><meta
name="dcterms.title" content="Linköping University" /><meta property="og:title" content="Startpage bottom" />
      <meta property="og:description" content="" />
    <meta property="og:image"
content="http://www.liu.se/polopoly/KoM/facebookbilder/linkuniv_sigill_facebook_314px.jpg" /><meta
name="dcterms.creator" content="webmaster@liu.se" /><meta name="dcterms.format" content="text/html" />
    <meta name="dcterms.language" content="en" /><meta name="dcterms.date" content="Thu Jan 14 11:42:04 CET
2016" /><meta name="dcterms.modified" content="2016-01-14" /><meta name="viewport" id="viewport"
content="width=device-width, user-scalable=yes,initial-scale=1.0"/>
<meta name="dcterms.publisher" content="Linköpings universitet" />
    <link rel="stylesheet" href="http://www.liu.se/mall11/css/content.css" type="text/css"/>
        <link rel="stylesheet" href="http://www.liu.se/mall11/css/screen.css" media="screen" type="text/css"/>
        <!--[if IE 6]><link rel="stylesheet" href="http://www.liu.se/mall11/css/ie6.css" media="screen"
type="text/css"/><![endif]-->
        <!--[if IE 8]><link rel="stylesheet" href="http://www.liu.se/mall11/css/ie8.css" media="screen"
type="text/css"/><![endif]-->
        <link rel="stylesheet" href="http://www.liu.se/mall11/css/print.css" media="print" type="text/css"/>
        <link rel="stylesheet" href="http://www.liu.se/mall11/css/polopoly.css" type="text/css"/>

        <link rel="stylesheet" href="http://www.liu.se/mall11/css/mobile.css" type="text/css"/>

      <link rel="stylesheet" href="http://www.liu.se/ny-mall-2012/filarkiv/1.329268/startsida-screen.css"
type="text/css"/>
    <link rel="alternate" href="http://www.liu.se/liu-nytt/arkiv/nyhetsarkiv?l=sv&amp;format=rss"
type="application/rss+xml" title="LiU-nytt" />
    <link rel="alternate" href="http://www.liu.se/liu-nytt/arkiv/nyhetsarkiv?l=en&amp;format=rss"
type="application/rss+xml" title="LiU-news" />
    <link rel="alternate" href="http://www.liu.se/liu-nytt/pressmed/arkiv?format=rss" type="application/rss+xml"
title="Pressmeddelanden" />
    <link rel="alternate" href="http://www.liu.se/rss/liu-jobs-sv.rss" type="application/rss+xml" title="Lediga
jobb" />
    <link rel="alternate" href="http://www.liu.se/rss/liu-jobs-en.rss" type="application/rss+xml"
title="Vacancies" />
    <script type="text/javascript" src="http://www.liu.se/mall11/script/jquery.min.js"></script>
    <script type="text/javascript" src="http://www.liu.se/mall11/script/pp.js"></script>
    <script type="text/javascript" src="http://www.liu.se/mall11/script/jquery.cookie.min.js"></script>
<script type="text/javascript" src="http://www.liu.se/mall11/script/mobile.js"></script>
```

Web pages

- A web page is a document which its content can be read, interpreted and presented by **web browsers**.
- Mostly contains HTML, CSS and javascript code. It can also contain references to media files.
- Web pages can be entirely located at **web servers** OR entirely/partially generated at server-side.
- A web page does not contain media files but references to media files.

Static web pages

- Only contains HTML/Javascript/CSS.
- Shall look the same no matter how many times downloaded, by who, when or how.

Dynamic web pages

- The content, html/css/javascript, is generated at the server-side.
- Data, known as parameters, is sent to the server upon request for a dynamic web page. Depending on the parameters the response is different and the returned web page looks partially differently.

Web servers

- A web server is a software system which receives requests from clients and in response returns web pages and other types of files.
- A computer running a web server software is also called web server.
- The received requests are in HTTP/HTTPS language(protocol).

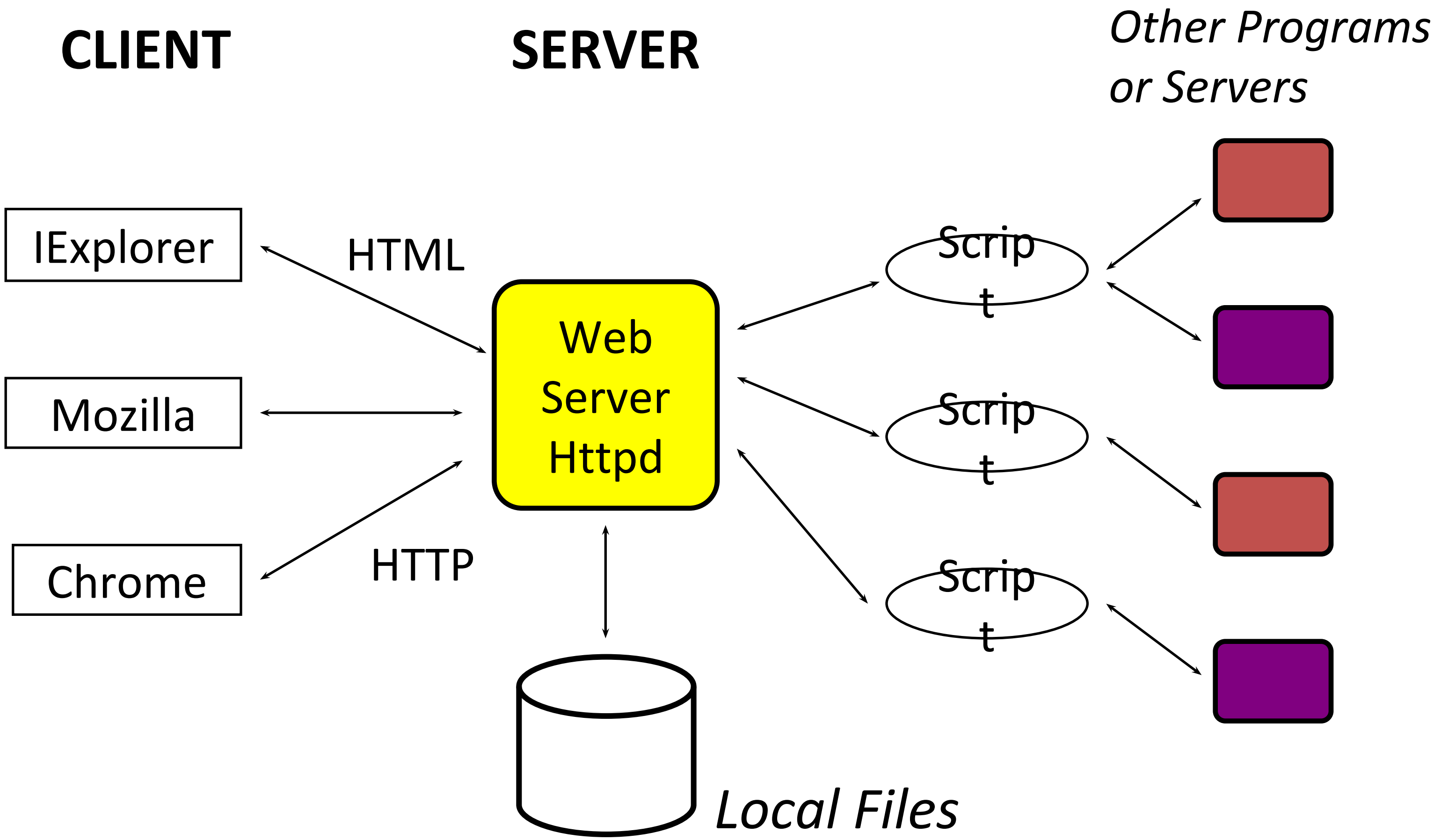
Application servers

- An application server is a software system which is capable of executing programs in certain languages. For example, Java or Python.
- The web server communicates with the application server to get some of the non-HTML code in the **dynamic web pages** executed.
- Application servers need to have access to some type of database system.

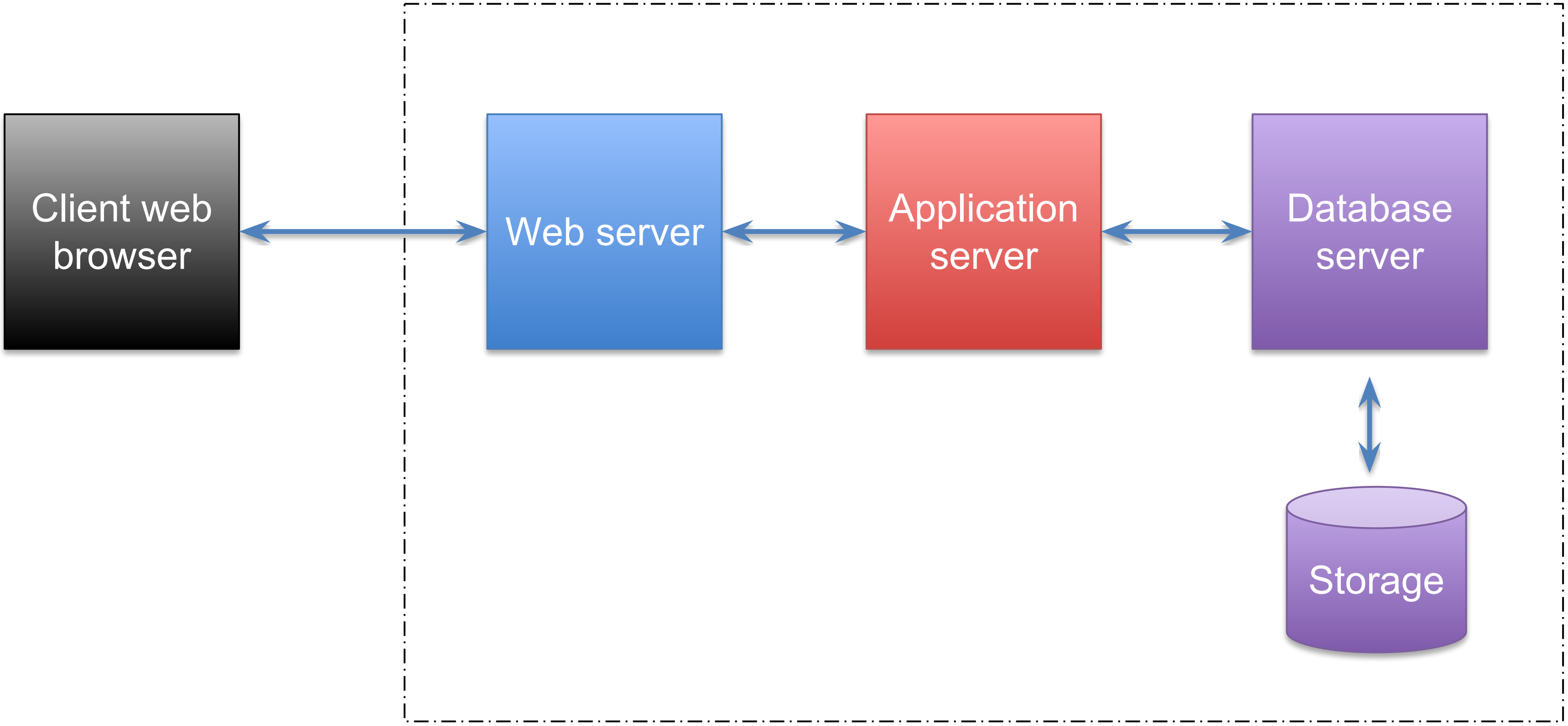
Web browsers(clients)

- Web browsers are desktop/mobile applications which are capable of reading, interpreting and visualizing web pages.
- Web browsers can be used for sending web requests using HTTP/HTTPS protocols.
- The user can use the address bar or the links inside of one already downloaded web page to send a request to the web server.
- When a web page is downloaded all the references, media files, inside of the web page shall be automatically translated to other requests sent to the server.

Web Server Configuration



Multi-tier systems



Web Programmig

- Producing dynamic web content
- Several techniques
  - Competing technologies
  - Rapidly evolving
- Client-side dynamics
  - JavaScript, Applets, Flash, etc.
- Server-side dynamics
  - CGI, JSP, JSF, ASP, PHP, XML, databases, etc.

Web Programming (cont.)

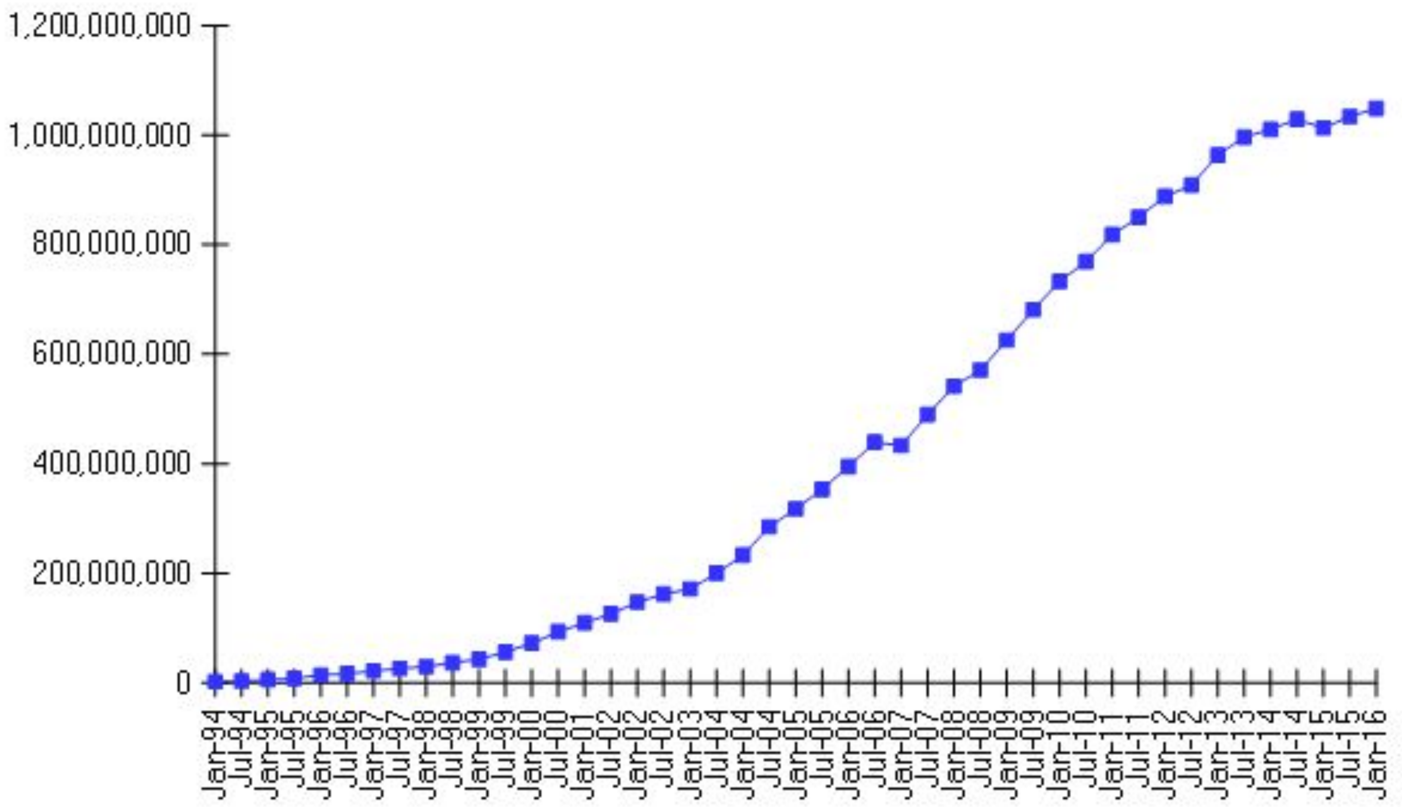
- Several generations of techniques
  - For example, CGI versus JSP versus SPA/RESTful API
- Some techniques maturing
  - Relatively stable basic technology (e.g., HTTP and HTML)
- What is important to learn?
  - Specific knowledge becoming obsolete quickly
- Goal this course:
  - Overview of existing techniques (e.g., HTML, CSS, JavaScript)
  - Deeper knowledge of specific techniques (e.g., Python/FLASK)
  - “Learn how to learn”

The Evolving Web

- Not just new technologies
- The Web is changing every day
  - Services provided
  - Usage pattern
  - Demographics
  - Traffic volume
- Internet and Web
  - Growth!!!
  - Stagnation?
  - Decline??
- See also Hobbes' Internet Timeline at http://www.zakon.org/robert/internet/timeline/

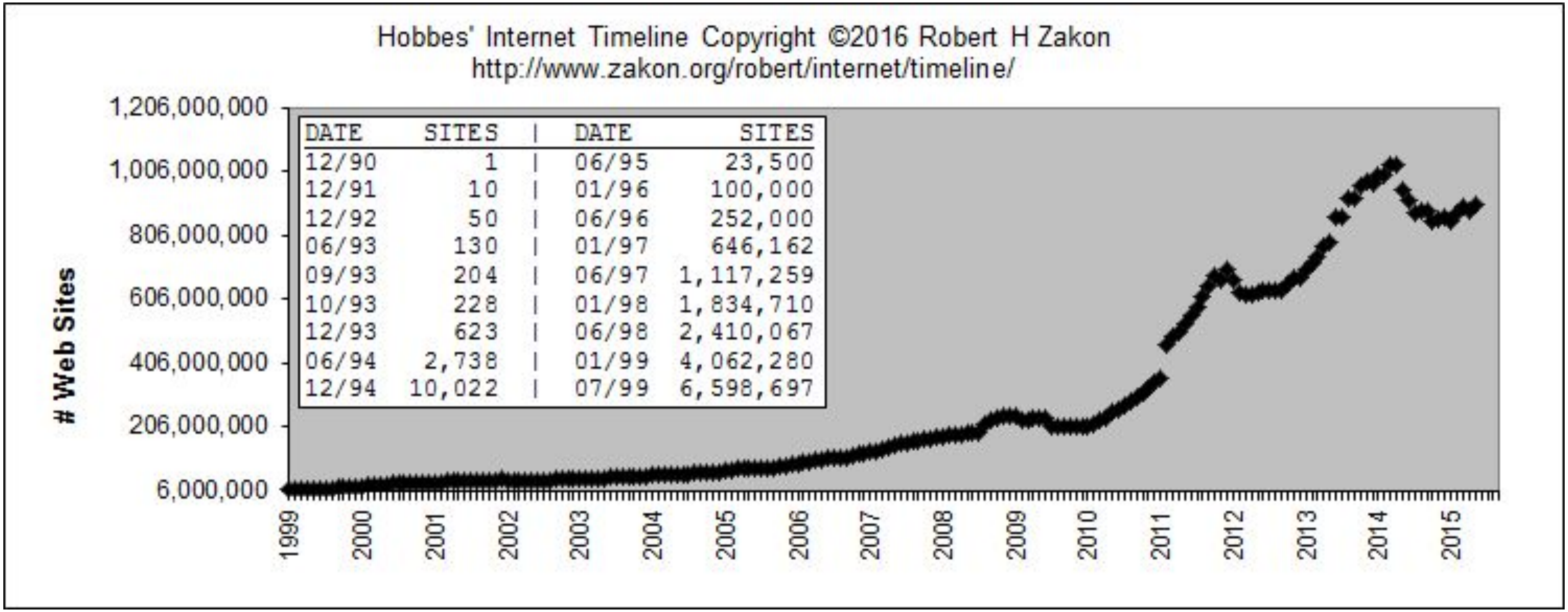
Growth of Hosts on the Internet

Internet Domain Survey Host Count



Source: Internet Systems Consortium (www.isc.org)

Web Growth

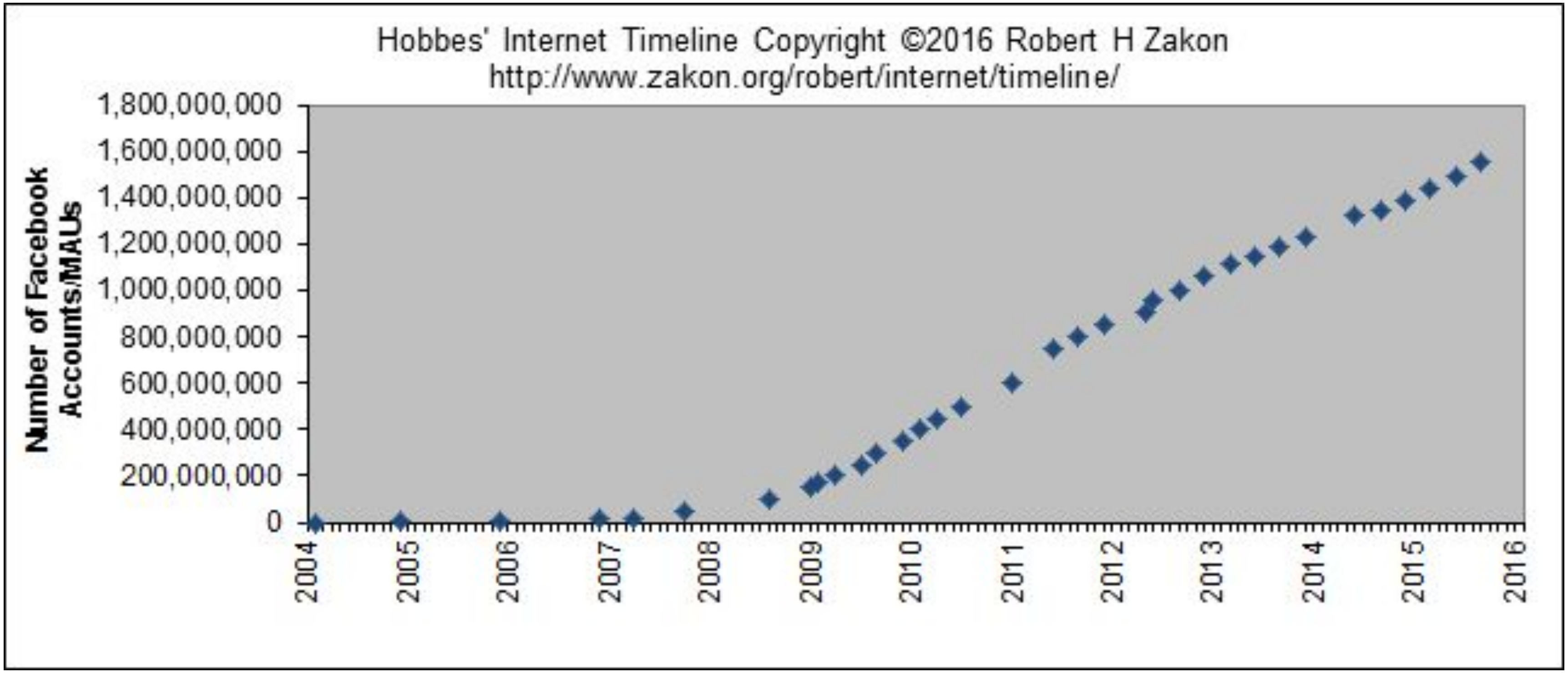


Hobbes' Internet Timeline Copyright ©2016 Robert H Zakon
http://www.zakon.org/robert/internet/timeline/

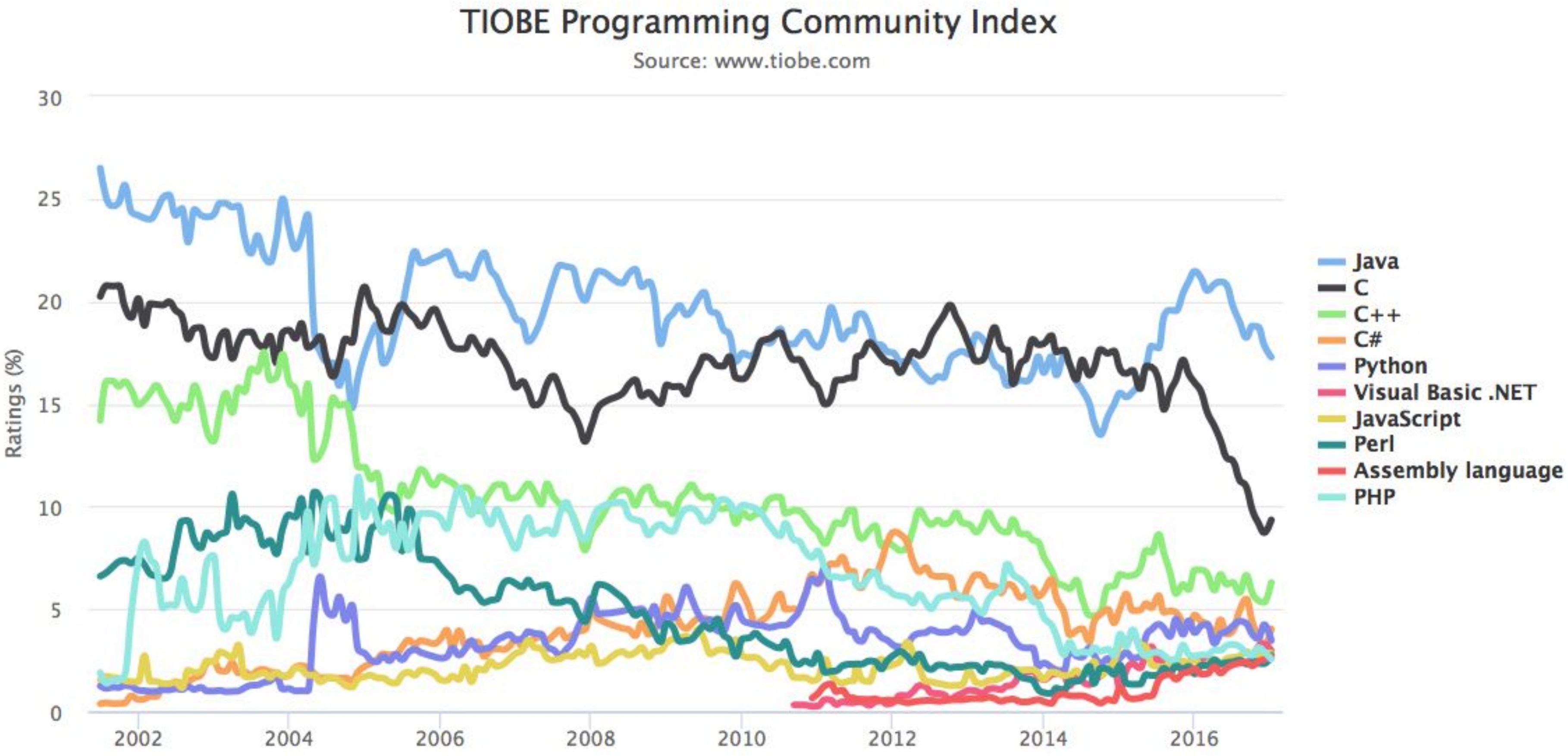
DATE	SITES	DATE	SITES
12/90	1	06/95	23,500
12/91	10	01/96	100,000
12/92	50	06/96	252,000
06/93	130	01/97	646,162
09/93	204	06/97	1,117,259
10/93	228	01/98	1,834,710
12/93	623	06/98	2,410,067
06/94	2,738	01/99	4,062,280
12/94	10,022	07/99	6,598,697

- Sep'12 drop caused by large network of link farmed domains disappearing from under the .com TLD
- Nov'14-Jan'15 drop largely caused by parked domains, with half attributed to a single IP ceasing to host them

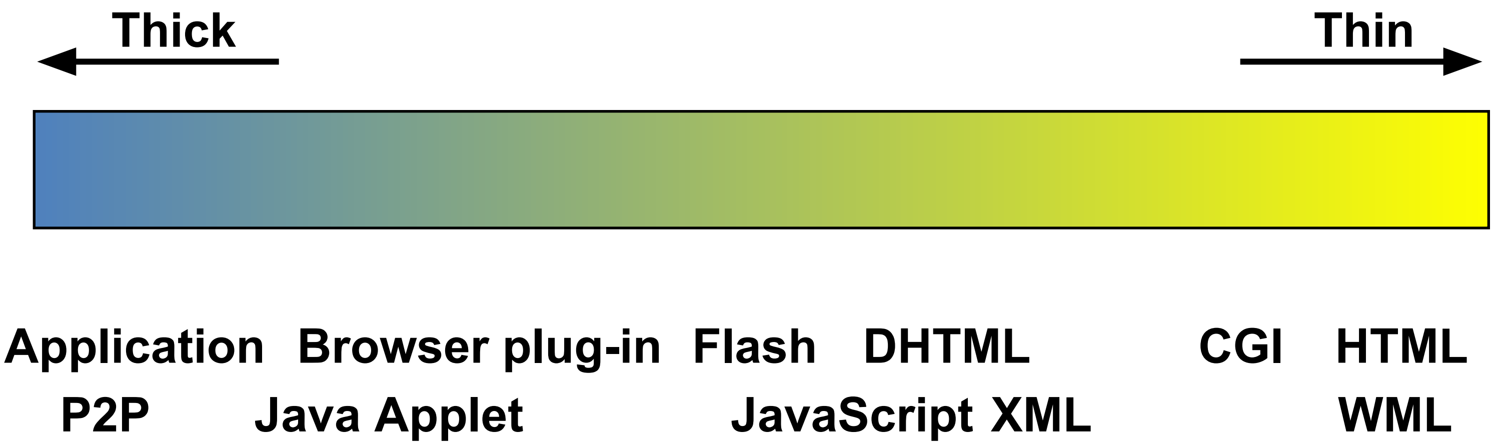
Facebook growth



Programming languages



Thin versus Thick Clients



Thick

Thin

Application P2P

Browser plug-in Java Applet

Flash

JavaScript

DHTML XML

CGI

HTML WML

More work on the client-side

More work on the server-side

Architecture of web applications

CGI programming for web(dynamic)

The web page is entirely generated by a programming script at the server-side. CGI programming is now obsolete.

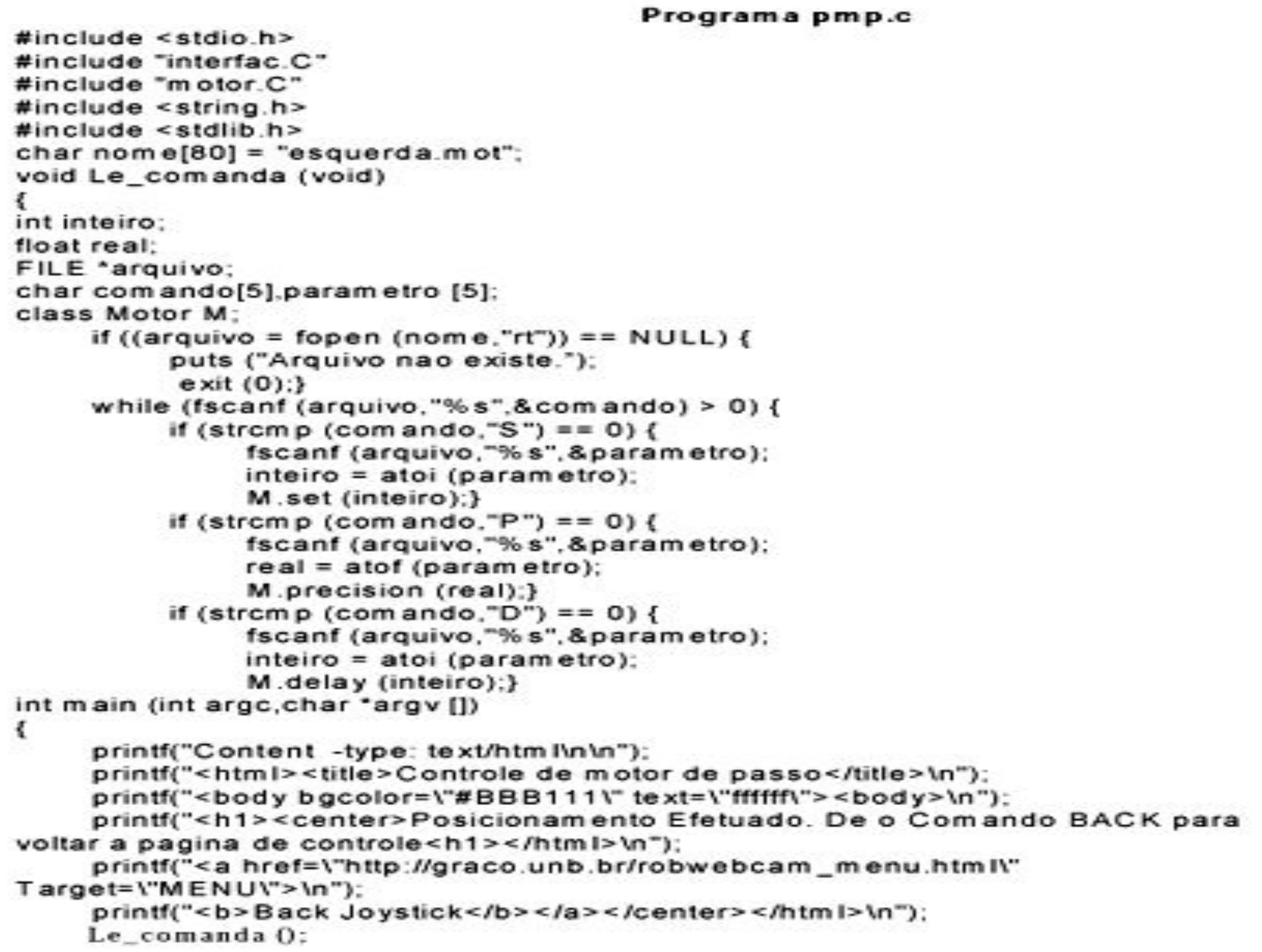
Problems:

- those static parts in each page which stay the same through time and for different users and settings also need to be generated. This causes more workload at the server-side and at the same time graphic designers as an independent team who may have no knowledge about programming shall have problems doing their part.
- The problems mentioned for server-side templating.

CGI programming for web(dynamic)

- A dynamic website is most likely composed of a combination of static web pages and dynamically generated web pages if CGI has been the choice, adding CGI programming is now obsolete.

CGI programming for web(dynamic)



Programa pmp.c

```
#include <stdio.h>
#include "interfac.C"
#include "motor.C"
#include <string.h>
#include <stdlib.h>
char nome[80] = "esquerda.mot";
void Le_comanda (void)
{
int inteiro;
float real;
FILE *arquivo;
char comando[5],parametro [5];
class Motor M;
        if ((arquivo = fopen (nome,"rt")) == NULL) {
                puts ("Arquivo nao existe.");
                 exit (0);}
        while (fscanf (arquivo,"%s",&comando) > 0) {
                if (strcmp (comando,"S") == 0) {
                        fscanf (arquivo,"%s",&parametro);
                        inteiro = atoi (parametro);
                        M.set (inteiro);}
                if (strcmp (comando,"P") == 0) {
                        fscanf (arquivo,"%s",&parametro);
                        real = atof (parametro);
                        M.precision (real);}
                if (strcmp (comando,"D") == 0) {
                        fscanf (arquivo,"%s",&parametro);
                        inteiro = atoi (parametro);
                        M.delay (inteiro);}
int main (int argc,char *argv [])
{
        printf("Content -type: text/html\n\n");
        printf("<html><title>Controle de motor de passo</title>\n");
        printf("<body bgcolor=\"#BBB111\" text=\"ffffff\"><body>\n");
        printf("<h1><center>Posicionamento Efetuado. De o Comando BACK para
voltar a pagina de controle<h1></html>\n");
        printf("<a href=\"http://graco.unb.br/robwebcam_menu.html\"
Target=\"MENU\">\n");
        printf("<b>Back Joystick</b></a></center></html>\n");
        Le_comanda ();
```

Figure 4. Program CGI in " C " (pmp.c) for actuator of step motor through parallel interface used in the RobWebCam system (Álvares & Romariz, 1998).

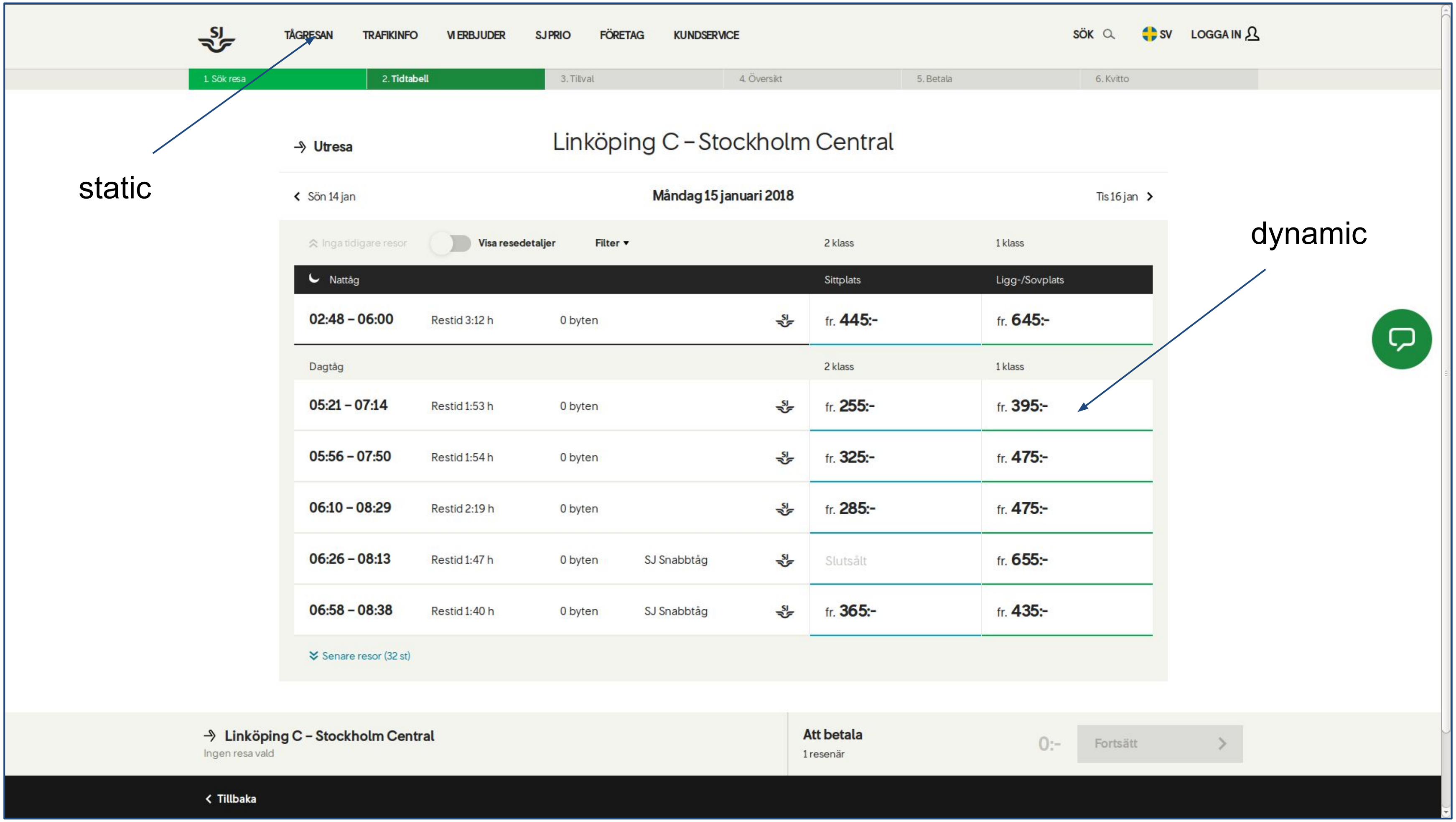
Server-side templating(dynamic)

Every dynamic page has its own static part which resides on the server. The dynamic part which can vary for different users and settings is generated inside of the page. The most common architecture nowadays. The most popular technologies in different languages: JSP, ASP, PHP, ASP .NET. AJAX can also be combined for better user experience. How AJAX can improve user experience?

Problems:

- Every request leads to a page refresh which disrupts user experience.
- For developing mobile apps the backend cannot be used as the data comes along with html tags.

Server-side templating(dynamic)



static

dynamic

TÅGRESAN TRAFIKINFO VI ERBJUDER SJ PRIO FÖRETAG KUNDSERVICE

SÖK SV LOGGA IN

1. Sök resa | 2. Tidtabell | 3. Tillval | 4. Översikt | 5. Betala | 6. Kvitto

Utresa

Linköping C – Stockholm Central

Sön 14 jan | **Måndag 15 januari 2018** | Tis 16 jan

Inga tidigare resor | Visa resedetaljer | Filter

				2 klass	1 klass
Nattåg				Sittplats	Ligg-/Sovplats
02:48 – 06:00	Restid 3:12 h	0 byten		fr. **445:-**	fr. **645:-**
Dagtåg				2 klass	1 klass
05:21 – 07:14	Restid 1:53 h	0 byten		fr. **255:-**	fr. **395:-**
05:56 – 07:50	Restid 1:54 h	0 byten		fr. **325:-**	fr. **475:-**
06:10 – 08:29	Restid 2:19 h	0 byten		fr. **285:-**	fr. **475:-**
06:26 – 08:13	Restid 1:47 h	0 byten	SJ Snabbtåg	Slutsålt	fr. **655:-**
06:58 – 08:38	Restid 1:40 h	0 byten	SJ Snabbtåg	fr. **365:-**	fr. **435:-**

Senare resor (32 st)

Linköping C – Stockholm Central
Ingen resa vald

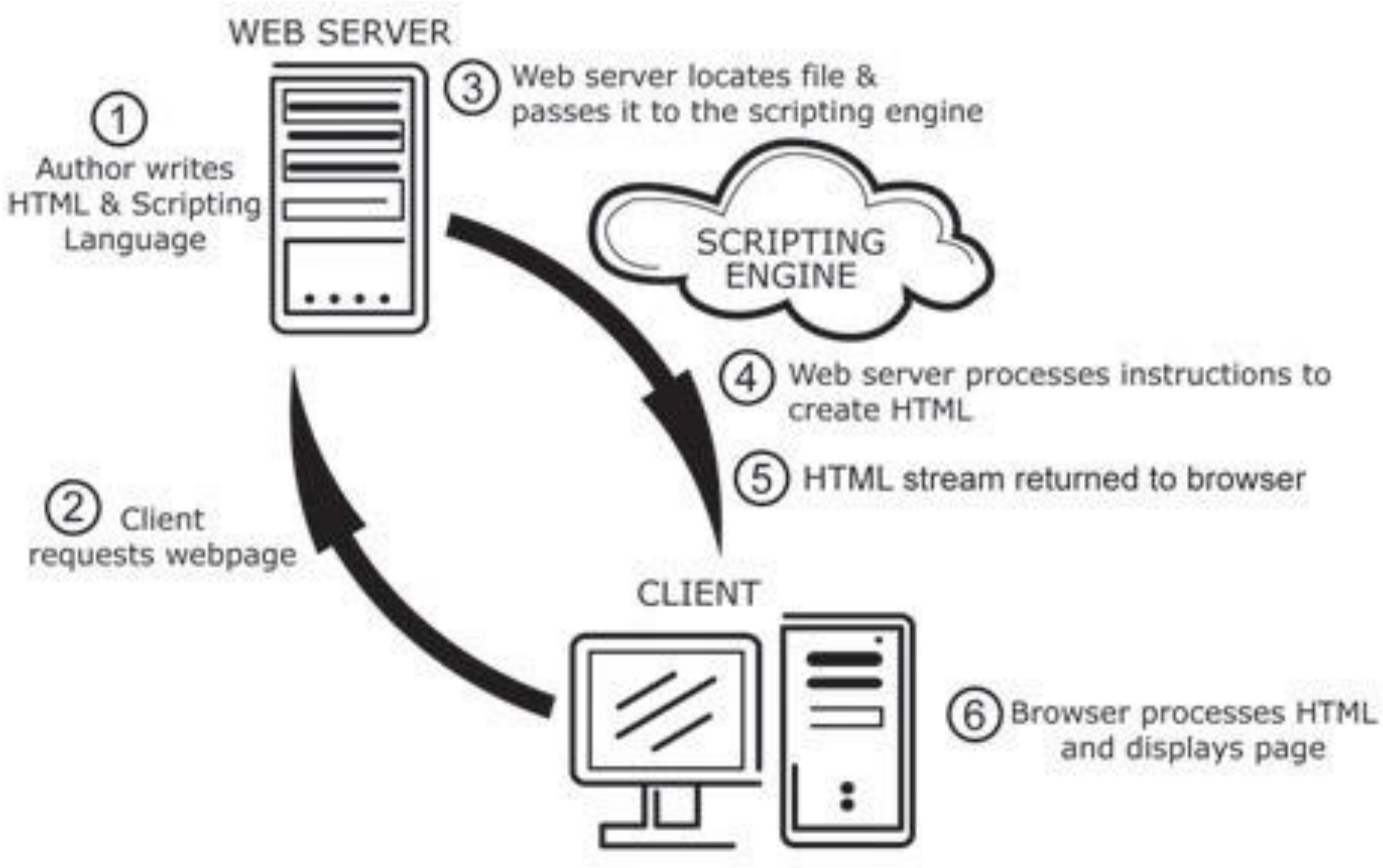
Att betala
1 resenär

0:-

Fortsätt

Tillbaka

Server-side templating(dynamic)


WEB SERVER
1 Author writes HTML & Scripting Language
3 Web server locates file & passes it to the scripting engine
SCRIPTING ENGINE
4 Web server processes instructions to create HTML
5 HTML stream returned to browser
2 Client requests webpage
CLIENT
6 Browser processes HTML and displays page

https://www.search3w.com/hello-world-2/

Server-side templating(dynamic)

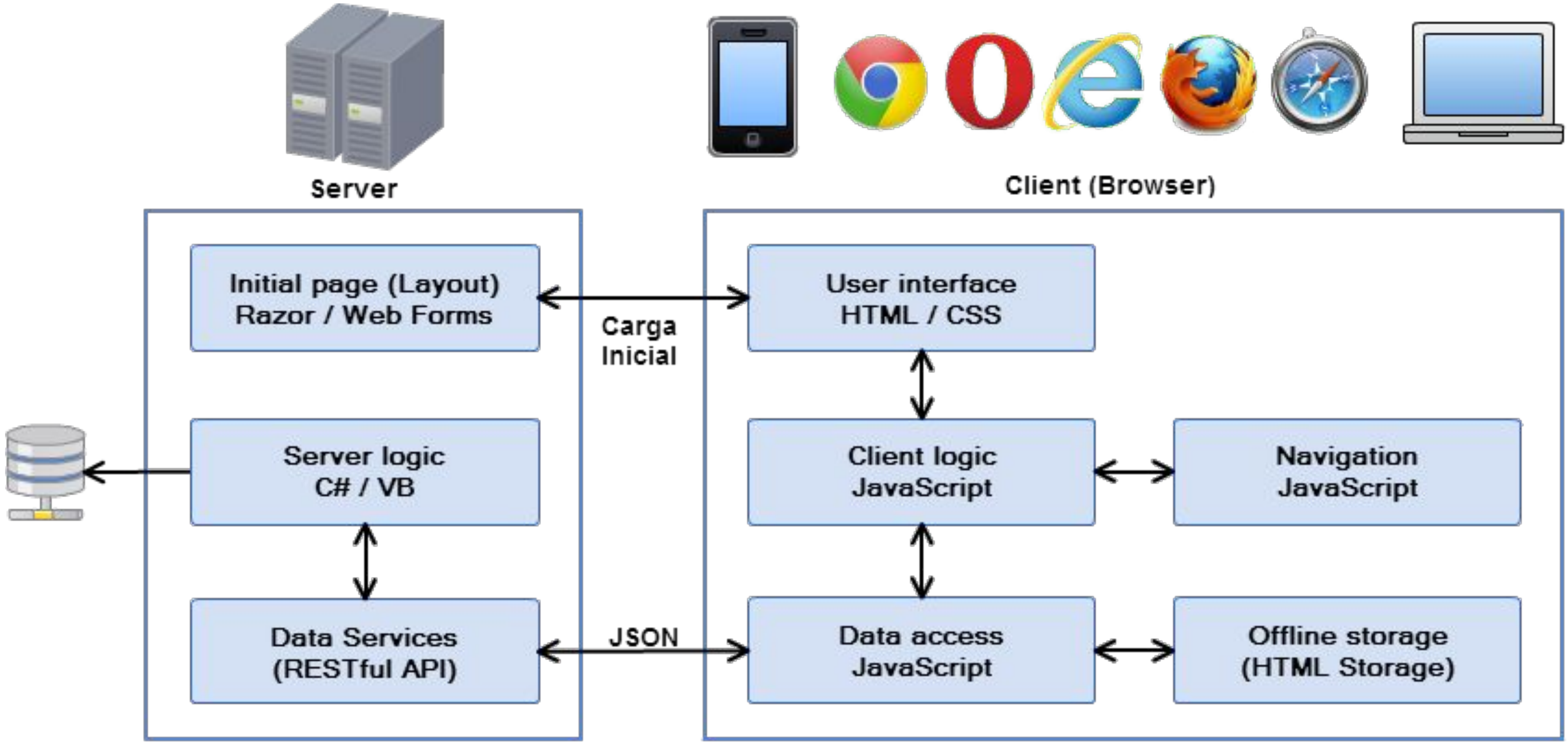
Dynamic websites are most likely composed of a combination of static and dynamic web pages.

Single Page Application(SPA)

Single-Page Applications (SPAs) are Web apps that load a single HTML page and dynamically update that page as the user interacts with the app. SPAs use **AJAX** and **HTML5** to create fluid and responsive Web apps, without constant page reloads. However, this means much of the work happens on the client side, in **JavaScript**.

Keywords: Thick-clients, client-side templating, Javascript, REST API

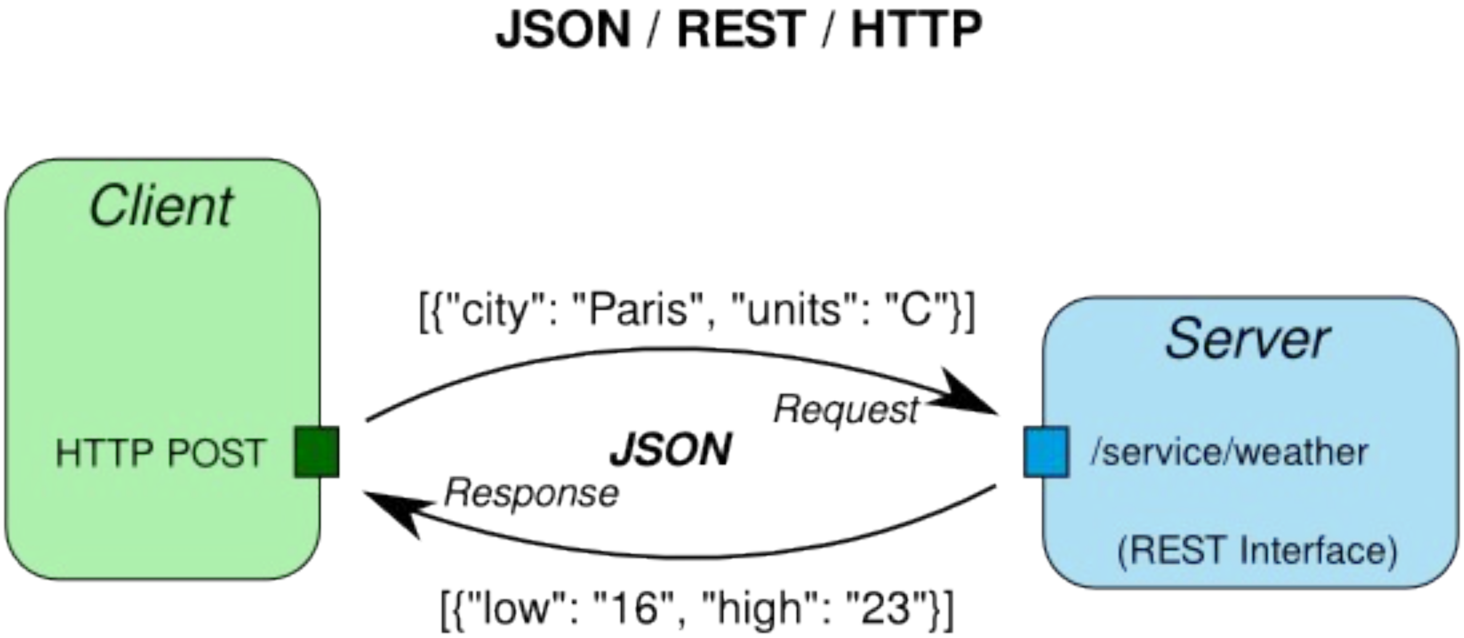
Single Page Application(SPA)



https://www.sitecorenutsbolts.net/2015/12/07/Single-Page-Applications-in-Sitecore-Part-I/

RESTful API/JSON

A set of server-side functions which return only the requested data in JSON. JSON is a data format.



http://linkeddataorchestration.com/2014/01/28/data-modeling-for-apis-part-2-rest-and-json/

HTML5

HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and current major version of the HTML standard.

Improved for SPA and client-side templating at HTML, javascript and CSS level.

AJAX

- Update a web page without reloading the page
- Request data from a server - after the page has loaded
- Receive data from a server - after the page has loaded
- Send data to a server - in the background

Summary (Lecture 1)

- Course overview
  - Lectures
  - Programming assignments
  - Lab sessions/Camedin
- Background
  - Internet
  - Web
- Architecture of web applications