Internet, intranet and Web — Lecture II

World Wide Web: standards, protocols, documents

Marco Solieri marco.solieri@lipn.univ-paris13.fr
Info et Réseaux en Apprentissage, Sup Galilée, Université Paris 13

November 6th, 2014

Outline

Contents

1 World Wide Web

Hypermedia

Ted Nelson, 1960s:

- removal of predeterminedness of text's sequence: hypertext
- (never-ended) implementation of needed technologies: Xanadu project

Definition 1 (Hypertext)**.** A text with accessible references (hyperlinks) to other text.

Timothy Barners-Lee, 1980s:

- simplification of hypertext concept
- project of needed technologies: WorldWideWeb

Definition 2 (World Wide Web)**.** A system of hypermedia accessed via the Internet and realized with a client-server architecture.

Web client-server architecture

Client (browser)

- user interface for read-only access to hypermedia:
  - visualize text and image,
  - play sound and video;
- retrieval of documents from the server;
- extendible via software components:
  - plugin: locally stored
  - scripts: remotely downloaded (JavaScript programming language);

Server

- transfer of hypermedia to client
- access to hypermedia, either:
  - local (e.g. file) or remote (e.g. record in DB)
  - static or dynamic (generated by software)

Basics of Web

Resource identifier (URI)

- identification for hypermedia and anything else,
- target for hyperlinks.

Communication protocol (HTTP)

- client-server stateless communication
- access to resources on the Internet

Document language (XHTML)

- realize hypertext and hypermedia
- hyperlinks support

2 Uniform Resource Identifier

2.1 Features

Resources

Definition 3 (Resource)**.** An object available on the World Wide Web.

What a resource could be:

- a file stored in a filesystem (e.g. a JPEG photo)
- a record of data (e.g from a DB)
- a file output of an application (e.g. a PDF document)
- a concrete object (e.g. a person or a book)
- an abstract concept (e.g. a grammar of a language)
- ...

Uniformity (or universality)

Common syntax for resources that could be located:

- Web, accessible via HTTP
- Internet, accessible via the appropriate protocol (e.g. FTP)

Simple:

- protocol independence,
- self-contained (include any information needed)
- cost-effectiveness in storage and communication (string format),

2.2 Definition

Types

Name (URN) unique, permanent and non-repudiable tag

Locator (URL) information for effective access

Syntax

Definition 4 (URI syntax)**.** schema : [// authority] path [? query] [# fragment]

Where

Schema arbitrary string, protocol name in case of URL (IANA)

authority hierarchical name of responsible a subspace of names, where form is [userinfo @] host [: port]

where host is a domain name or an IP address

Path hierarchical name of resource, where separator is /

Query specifications of resource, (typically) where separator is &, form is parameter=value and space is +

Fragment secondary resource: internal of or relative to the primary

Examples

Example 5 (URIs).

- `http://www.ietf.org/rfc/rfc2396.txt`
- `ftp://ftp.is.co.za/rfc/rfc1808.txt`
- `cid:foo4%25foo1@bar.net`
- `mailto:John.Doe@example.com`
- `news:comp.infosystems.www.servers.unix`
- `file:///home/john/Documents/file.tex`
- `urn:isbn:0-486-27557-4`

Special characters

Reserved characters: `; / : @ & = + $ ,`
Escaped characters (with %NN):

- control characters, i.e. ASCII < 32
- non ASCII, i.e. Latin-1 > 127
- unwise characters: `{ } | \ ^ [ ] ‘`
- delimiters: ␣ `< > # % "`
- reserved characters used with different meaning

2.3 Dynamics

Operations on URIs

Resolution

- generation of the corresponding absolute URL
- input:
  - an URI reference (i.e. a relative URI)
  - an URI which is not an URL

  output: an URL

Dereferencing

- retrieval of the corresponding resource
- input: an URL
- output: a resource

3 HyperText Transfer Protocol

3.1 Connections

HTTP features

Client-server arch. the client opens the connection and request a service, the server replies and closes the connection.

Data independence support for transfer of HTML document and any other format, via content negotiation.

Statelessness any HTTP connection must contain any information needed for the response.

Caching support for implementation of various caching policies and tools.

Authentication specifications for various techniques of user authentication.

Roles of a HTTP communication

Necessary roles:

User agent The client which initiates the HTTP request (i.e. a browser or a bot)

Origin server The server who owns the resource

Extra roles (possible):

Proxy an application acting both as server and a client and controlling the communication

- transparently (e.g. caching), or
- not transparently (e.g. verification, filtering, enriching).

Gateway an application acting as the origin server (e.g. load balancing or load layering)

Connection and persistence

Definition 6 (HTTP 1.0 connection)**.** A request by client and a reply by server.

Cons of having a distinct TCP connection for each HTTP request:

- network overhead,
- computation overhead,
- time overhead.

HTTP 1.1 (IETF RFCs 2616, 2617) introduces connection persistence

Definition 7 (HTTP 1.1 connection)**.** A sequence alternating a request and a reply.

Definition 8 (HTTP 1.1 pipelining connection)**.** A sequence alternating requests and ordered replies.

3.2 Requests

Request

Definition 9 (HTTP Request)**.** A MIME message with the following syntax:

Method URI Version `CRLF`

[Header `CRLF`]*

`CRLF`

[Body]

Where

Method the type of action requested

Version one of `HTTP/1.0` and `HTTP/1.1`

Header parameters of transmission, entity and request

Request methods

Main methods:

GET Retrieve a representation of a resource (since HTTP 0.9). Could be:

- conditional, when specifying a criterion (e.g. If-match, If-modified-since);
- partial, when specifying a portion of a request.

HEAD Retrieve server's information about a resource. Ask for a reply message without body: headers only.

POST Relate an information to a resource Used for data submission (e.g. from a form).

PUT Insert a resource. Create a new resource or substitute the old one.

DELETE Remove a resource and any related information.

Note: PUT and DELETE offer no access control (see WebDAV).

3.3 Headers

Common headers: transmission

Main transmission headers

Date date and time of the transmission.

MIME-Version version of MIME used (i.e. 1.0).

Transfer-Encoding encoding format

Cache-Control caching policy requested or preferred

Connection type of connection to use (e.g. persistent or not)

Via used by proxies and gateways

Message headers: entity
Main entity headers (about the message body)

Content-Type MIME type, mandatory if has a body

Content-Lenght size in bytes, mandatory

Content-Encoding encoding format

Content-Language human language

Content-Location URL

Content-MD5 MD5 digest value

Content-Range portion

Expires date of invalidation

Last-Modified date and time of last modification, mandatory

Requests

Request message headers
Main request headers

- generic headers,
- entity headers about the related entity
- reply-specific headers, e.g.

  User-Agent client description (e.g. name, version, OS)

  Referer URL of the resource linking the requested one

  Host domain name and port used, mandatory in HTTP 1.1

  Accept accepted MIME versions

  Accept-Charset accepted character sets

  Accept-Encoding accepted encoding formats

  Accept-Language accepted human languages

  If-Modified-Since minimum accepted date (only if newer than)

  If-Unmodified-Since maximum accepted date (only if older than)

Example request

Example 10 (Click on `http://ms.xt3.it/teaching.xhtml#iweb`). `GET /teaching.xhtml\#iweb HTTP/1.1`

```
Host: ms.xt3.it
Connection: keep-alive
User-Agent: Mozilla/5.0 (X11; Linux i686)
  AppleWebKit/535.19 (KHTML, like Gecko)
  Chrome/18.0.1025.151 Safari/535.19
Accept: text/html,application/xhtml+xml,
  application/xml,*/*
Accept-Encoding: gzip,deflate,sdch
Accept-Language: it,en-US,en
Accept-Charset: ISO-8859-1,utf-8,*
```

Properties of request methods
Definition 11 (Safety)**.** The method has no side effects on the internal state of the server.

- then the method can be executed by an intermediate node (e.g. a caching proxy)

Definition 12 (Idempotence)**.** Repetitions of the method are equivalent to a single one.

- then the method can be re-executed

Properties of methods

	GET	HEAD	POST	PUT	DELETE
Safety	✓	✓			
Idempotence	✓	✓		✓	✓

3.4 Replies

Reply

Definition 13 (HTTP Reply)**.** A MIME message with the following syntax:

Version Status_code Reason_phrase `CRLF`

[Header `CRLF`]*

`CRLF`

Body

Where

Version one of `HTTP/1.0` and `HTTP/1.1`

Status_code three-digits numeric code

Reason_phrase human-readable string

Status codes
Catogories and codes

1xx Informational wait for completion of request: 00 continue.

2xx Successuful requested action received and accepted: 00 Ok, 01 Created.

3xx Redirection requested action received, another action should be performed: 01 Moved permanently, 02 Found, 03 See other, 04 Not modified.

4xx Client error requested action cannot be performed because of client: 00 Bad request, 01 Unauthorized, 03 Forbidden, 04 Not found.

5xx Server error requested action cannot be performed because of server: 00 Internal server error, 01 Not implemented.

Reply headers
Headers in a reply could be:

- generic headers,
- entity headers if contains an entity (Content-Type and Content-lenght mandatory)
- reply-specific headers, e.g.

  Server server description (e.g. name, version, OS)

  WWW-Authenticate authentication method and parameters

Example reply

Example 14 (Reply from server).
```
HTTP/1.1 200 OK
Vary: Accept-Encoding
Content-Encoding: gzip
Last-Modified: Fri, 04 Nov 2011 00:27:06 GMT
Content-Type: text/html
Accept-Ranges: bytes
Content-Length: 1924
Date: Sat, 11 May 2012 07:21:33 GMT
Server: lighttpd/1.4.28

<html>
    ...
```

4 HTTP extensions

4.1 Authentication and security

Client authentication
Information for challenge-response protocol

Realm description of the restricted-access area,

Credentials pair: username and password.

Authentication process:

1. client send a request for a restricted resource;
2. server replies with code 401 and a WWW-Authenticate header:
   - authentication method,
   - authentication challenge;
3. client get username and password from user;
4. client send the request again, with response (and will add it to any other request of that realm);
5. server verify response and
   - accepts and serves the request, or
   - replies with 403 Forbidden.

Client authentication methods
Basic authentication method:

- since HTTP 1.0,
- challenge: realm,
- response: Base64 encoding of `username:password`.

Problem
Secrets sent in cleartext on the underlying channel!

Digest access authentication method

- since HTTP 1.1
- challenge: realm, nonce
- response: Hash(`realm`, `nonce`, `username`, `password`)

HTTPS: HTTP Secure

Alert
HTTP with digest access authentication still insecure, exposed to:

- eavesdropping,
- man-in-the-middle attacks.

Secure the underlay channel with SSL/TLS (IETF RFC 2818):

Confidentiality SSL/TLS encryption

Server authentication SSL/TLS key certificate

Client authentication Basic HTTP method (now secured) or Digest method (SSL/TLS certification is costly)

4.2 State management

State management: cookies
State management in a stateless protocol

- Enrich the connection with a persistent piece of data: the cookie.
- Add special-purpose headers for handling.
- Proposed extension by Netscape
- Standardized by IETF RFC 6265

Cookie handling process:

1. server adds to the first response a Set-Cookie header containing the cookie
2. server now start associating the unique cookie to the client
3. client stores the cookie and will always add it to requests for that server with a Cookie header.

Cookies
Content:

Comment description

Domain domain name for which is valid

MaxAge duration in seconds of validity

Path URI for which is valid

Secure need for secure channel (HTTPS)

Version

Uses:

Session management storage of session data, e.g. shopping basket, web login, persistent login.

Personalization storage of user data, e.g. site customization, user preferences.

Tracking storage of user behaviors, e.g. site navigation history.

Cookies and user privacy
Third-party tracking cookies

- every connection could request to use cookies,
- advertisement enterprises buy banners in web pages,
- ads contains a third-party web resource, dynamically loaded,
- then: ads enterprises can use cookies to perform "web analytics" – tracking and profiling users across multiple websites.

(Partial) countermeasures for a (little) more private browsing:

- keep cookie storage clean of unnecessary cookies,
- disable setting of third-party cookie on the browser,
- disable loading of ads in general.

5 Web architectures

Static web
Every resource is physical stored:

- a web page can be composed of different resources
- every resource has a different URL
- client requests each resource and render the page

Pro: easily implementable.
Con: no automation, no integration between resources.

Dynamic web
Most of resources are dynamically generated.
The website as a three-layers application:

1. storage: files for binary data, DBMS for structured data
2. application: programs for the HTML generation
3. browser: HTML rendering

Two main approaches for implementing the application:

Embedded code code inside HTML

Full application code and HTML templates

Embedded code
Script program inserted inside HTML documents as comment:

- HTML as template,
- code as the application.

The web server:

- executes the interpreter on the content of special comments,
- replaces the comment with the output from the interpreter.

Pros:

- powerful: code integration with document,
- cost-effective: easy to use.

Cons:

- weak architecture: application and presentation dependence,
- hard to design: application can become sparse.

Full application
Stronger separation between application logic and presentation:

- source files for applications,
- template files for presentation.

Common approaches:

- static templates, read by the program and then merged with the dynamically-generated HTML;
- a piece of program first generate the templates, later used by the main module;

- a full template engine invokes the application, that outputs lot of parameters, and inserts them into the templates.

Pros:

- independence between application logic and presentation.

Cons:

- separation application/presentation still not complete.

Four-layers web
The website as a four-layers application:

1. storage: files for binary data, DBMS for structured data
2. application-logic: programs for content generation (e.g. XML)
3. presentation: programs for content presentation (e.g. XSLT)
4. browser: HTML rendering

Pros:

- full separation application/presentation,
- modular and simple architecture.

Cons:

- expensive: each step of the application triggers the whole process: page regeneration, transmission, and rendering,
- difficult naming and caching: each step of the application need its own URL.

Four-layers web 2.0
Browser loads:

- a simpler HTML page, with missing parts (template),
- JavaScript code (piece of program),
- some Ajax libraries (common implementation).

The client-side program can implement:

- presentation only: talks with the application-logic layer on server-side via XML,
- presentation and application-logic: almost everything on client-side, except for server requests and storage queries

6 Web markup

Web markup
The five levels of web markup

Content the text with spaces, punctuations ...

Structure paragraphs, lists, tables, emphasis, citations ...

Linking anchors for creating hyperlinks

Semantics meaning of contents, relations to other concepts ...

Presentation double interline for paragraph separation, bullets for lists, red bold for emphasis, big centered for title ...

6.1 Brief history of web markup

Ancient history

- Birth (1991-1995)
  - SGML-like definition by Timothy Barners Lee (HTML 1)
  - Refined version standardized by IETF (HTML 2)
  - Markup language for hypertext: document structuring, text formatting, hyperlinks, images ...
- First browser war (1994-1998)
  - Market war: Netscape Navigator vs Microsoft Explorer
  - Proprietary extensions to HTML: scripting (JavaScript), style tags
  - W3C foundation for improving effective standardization
  - Document tests as browser visualization (it works), instead of correctness (it is validated)

Middle ages

- The last word: HTML 4 (1997-1999)
  - After lot of intermediate versions, HTML 4
  - Several features added: i18n, style sheets, frames ...
  - Separation between structure and presentation (Strict DTD)
  - Most recent and most used standard: version 4.01

- Format corruption for HTML and CSS (1997-present)
  - Web pages written as tag soup
  - Transitional DTD still common
  - Browser with quirk mode parsing, each with its own algorithm

  Aim change (1999-present)
  - from a markup language for hypermedia
  - to a language for web application.

Late middle ages (yup, present)

- Serialization (2000-2006)
  - Reformulation of HTML 4.01 in XML 1.0 (XHTML 1)
  - Modularized for user agent targeting (XHTML 1.1)
  - Complete redesign, without backward compatibility, without participation of players from browsers, search engines, CMSs (abandoned draft XHTML 2)
- Defeat of the systematic approach (2004-present)
  - WHAT Working Group opens and W3C reopens too for developing together a new HTML version (*He who ships working code wins*)
  - Not a new language, but changes to HTML 4:
    * specifies common (good or bad) practices as standard adds new features (web application, multimedia)
    * parsing algorithm specified :-)
    * a living standard (even if W3C considers it as draft)
    * serialized syntax XHTML 5

6.2 Morals

A moral

> Browser vendors showed us some cool stuff that we will soon be able to do on the web. Text that runs at an angle, boxes with round corners, logos that spin, graphics that pulse to the beat of the music, forms that (wow!) check that you've entered a valid date. And everyone tweeted 'Wow, that's cool'.
>
> Well, it's not cool. It's a tragedy.
>
> Michael Kay (W3C editor)Nov 4th, 2010

Our moral

We want to deliver:

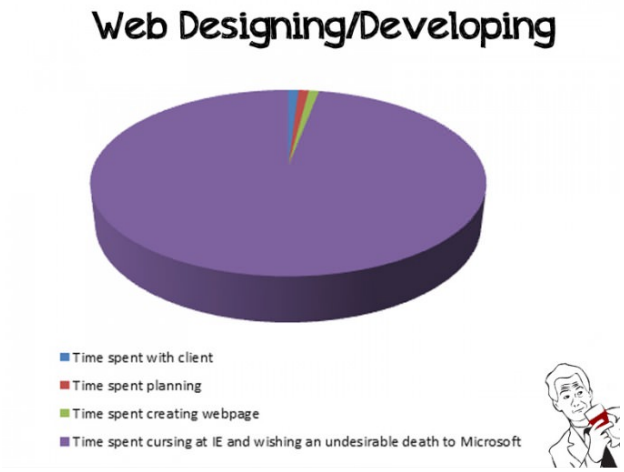
- layered design,
- strictly-validated and fully-interoperable pages.

For markup we can use:

- XHTML 1, the best language so far;
- XHTML 5, changeable and with limited support: still partial, to be checked (http://www.modernizr.com/)

Yes, we can!

Nevertheless...



True story.

7 XHTML 1 and HTML 4 markup

7.1 Document type and basic structure

Document type

Many markup languages

- Each language is formally defined in a DTD (Document Type Definition):
  - HTML 4.01: Strict, Transitional or Frameset
  - XHTML 1.0: Strict, Transitional or Frameset
  - XHTML 1.1: Basic 1.1, Mobile 1.2
- Each document has a statement of conformity to a particular DTD: a document type declaration.

XML serialization
Motivations:

- stricter syntax,
- faster parsing
- easier manipulation,
- extensibility.

Differences to HTML:

- correct nesting,
- case sensitiveness,
- closing tags (`<tag>` ... `</tag>`) or empty elements (`<br/>`),
- quoted attributes.

Example

Example 15 (Document type declarations). HTML 4.01 Strict document header:

```
<!DOCTYPE HTML
    PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/
      strict.dtd">
```

XHTML 1.0 Strict document header:

```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE html
      PUBLIC "-//W3C//DTD XHTML
        1.0 Strict//EN"
      "http://www.w3.org/TR/xhtml1/DTD/
        xhtml1-strict.dtd">
  <html xmlns="http://www.w3.org/1999/
      xhtml" xml:lang="en" lang="en">
```

Document structure tags
`<html>` root of the document

- `<head>` container for processing information and metadata
  - `<base>` base URL for all relative links,
  - `<meta>` metadata, e.g. author, keywords, HTTP headers,
  - `<style>` definition or reference to CSS,
  - `<script>` definition or reference to script;
- `<body>` container for displayable content of the document
  - Block elements
    - rectangular shape,
    - adjustable margin and size,
    - not broken into multiple lines.
  - Inline elements
    - part of the text flow,
    - necessarily included in a block element.

7.2 Inline elements

Anchor

- Creation a relation between the text and an URL.
- Text tagged `<a>`.
- With respect to the URL, text could become:

Target
- attribute: `id`
- value: name of the fragment identifier
- use: make the text linkable with the name

Origin
- attribute: `href`
- value: URL
- use: make the text a link to the URL

Phrase elements
Main phrase elements and semantic:

- `<em>` emphasized,
- `<strong>` strongly emphasized,
- `<abbr>` explanation of an abbreviation,
- `<dfn>` definition of a single term,
- `<code>` snippet of source code.

Other elements and semantic:

- `<br />` forced line break,
- `<q>` quotation,
- `<span>` arbitrary: customized presentation and/or behaviour.

7.3 Block elements

Structure and quotation
Document structure (pretending)

- Heading block
  - tagged `<hN>`
  - where $1 \leq N \leq 6$ and top level is 1.
- Paragraph block
  - tagged `<p>`.

Quotation

- `<blockquote>` block of quotation

Customized block

- tagged `<div>`
- arbitrary semantic, for customized presentation and/or behaviour

Listing

Unordered list

- tagged `<ul>`
- items tagged `<li>`

Ordered list

- tagged `<ol>`
- items tagged `<li>`

Definition list

- tagged `<dl>`
- term item tagged `<dt>`
- definition tagged `<dd>`

7.4 Tables

Table

- Table tagged `<table>`, containing
  - (possibly) a caption tagged `<caption>`
  - rows tagged `<tr>`, containing:
    - header cells tagged `<th>`
    - data cells tagged `<td>`
  - possible partition with:
    - header group tagged `<thead>`
    - footer group tagged `<tfoot>`
    - body group tagged `<tbody>`
- Table element attributes
  - a brief description in `summary`
  - presentational aspect: `border`, `width` ...
- When use a table?
  - to create a table,
  - not to lay the page out.

Table and cell spanning

- A cell can span next rows or columns: attr. `rowspan`, `colspan`.

Example 16 (Table with spanning cells).

```
<table border="1" cellpadding="3"
  width="75%" summary="Spanning cells">
  <tr>
    <td>Cell A</td>
    <td colspan="2">Cell B</td>
  </tr>
  <tr>
    <td colspan="2">Cell C</td>
    <td rowspan="2">Cell D</td>
  </tr>
  <tr>
    <td>Cell E</td>
    <td>Cell F</td>
  </tr>
</table>
```

Cell placement is A: 1,1 - B: 1,2-3 - C: 2,1-2 - D: 2-3,3 - E: 3,1 - F: 3,2

7.5 Images

Images

- Typical image formats: JPEG, PNG, GIF
- Empty element `<img />`
- Main attributes:

  src URL of the image (mandatory)
  alt text alternative to image visualization
  width forced width for visualization of the image
  height forced height for visualization of the image
  name name to be referred to the image
  usemap image is a client-side map
  ismap image is a server-side map
- Deprecated style attributes: align, border, vspace, hspace.

7.6 Forms

Form

Goals:

1. asking information to the user
2. specification of information processing
   - possible validation: client side, with JavaScript
   - processing: server side, with HTTP request to application

Root element `<form>`

- attribute `method`: the HTTP method of the request to be performed, `GET` or `POST`,
- attribute `action`: URL of the HTTP request, typically of a script.

Input element

Tag `<input />`

- empty element
- attribute `type`, main possible values:
  - `text`: single line of text
  - `password`: single line of text to be hidden by the browser
  - `button`: button
  - `submit`: button for form committing
  - `reset`: button for form resetting
  - `checkbox`: checkbox for (multiple) selection
  - `radio`: radio buttons for single selection
  - `file`: file upload (to be selected)
- attribute `value` semantics:
  - default value, for text types
  - descriptive text, for buttons types
  - associated value, for selection types

Selection, button, text area elements

Selection element

- tag `<select>`,
- the content is a set of elements tagged `<option>`,
- elements could be grouped in elements tagged `<optgroup>`.

Button element

- tag `<button>`,
- has content,
- more customizable than `<input type="submit">`.

Text area element

- tag `<textarea>`
- multiple-lines text input
- size specified with mandatory attributes `cols` and `rows`

Summarizing example

Example 17 (Subscription form).

```
<form action="add_subscriber.cgi" method="post"><p>
    First name: <input type="text" name="firstname" /> <br />
    Last name: <input type="text" name="lastname" /> <br />
    Email: <input type="text" name="email" /> <br />
    <input type="radio" name="sex" value="M" /> Male <br />
    <input type="radio" name="sex" value="F" /> Female <br />
    <button name="submit" value="submit"
            type="submit" onclick="verify()">
      Send
      <img src="/icons/wow.gif" alt="wow">
    </button>
    <button name="reset" type="reset">
      Reset
      <img src="/icons/oops.gif" alt="oops">
    </button>
</p></form>
```

Outline

Contents