

Hypertext & hypermedia

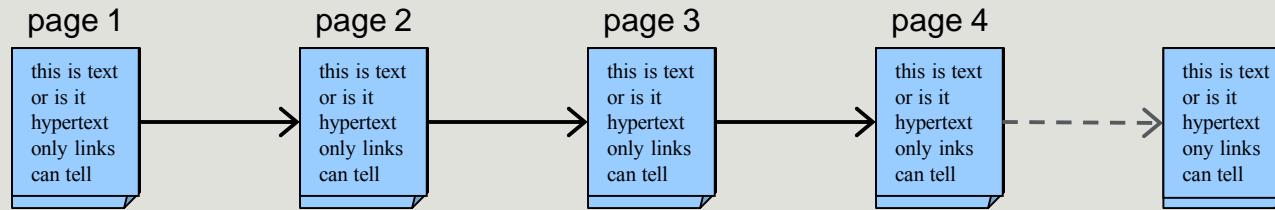
WIOLETA SZWOCH PHD MENG

DEPARTMENT OF INTELLIGENT INTERACTIVE SYSTEMS

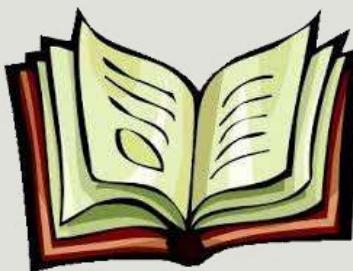
Hypertext

Text

- imposes linearity of reading



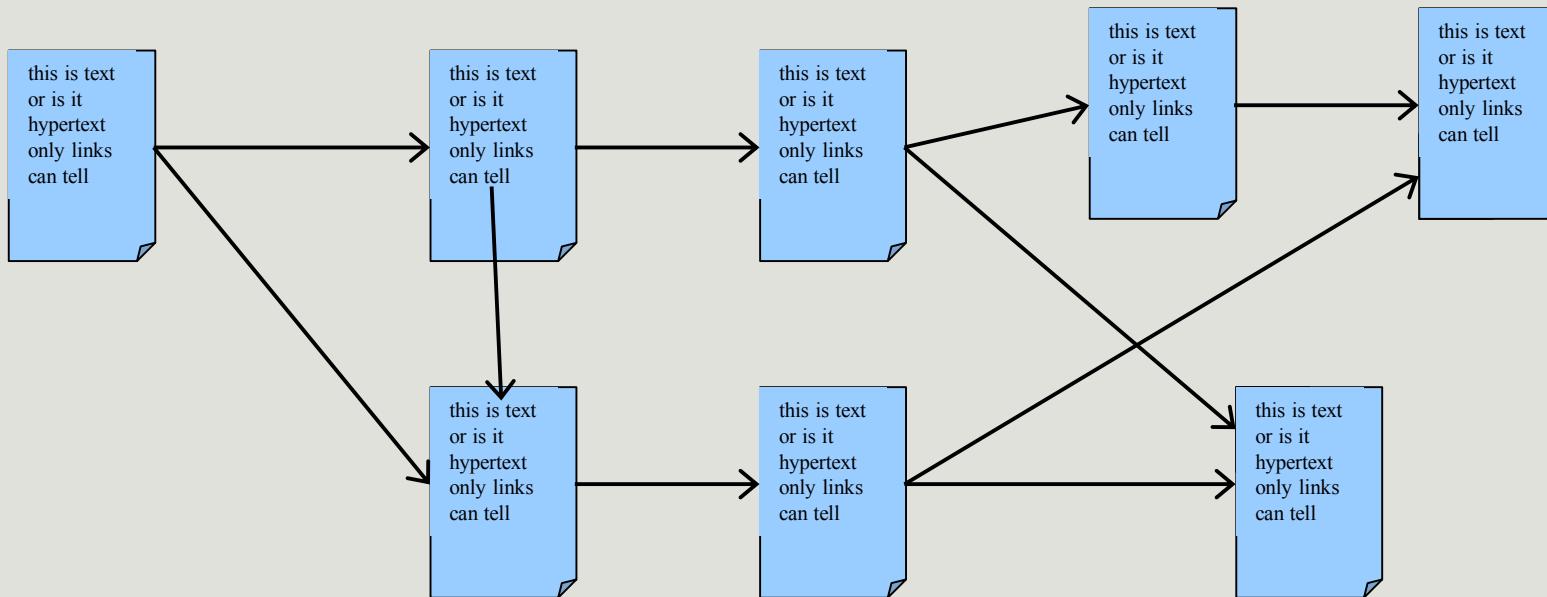
- author knows better



Hypertext

hypertext

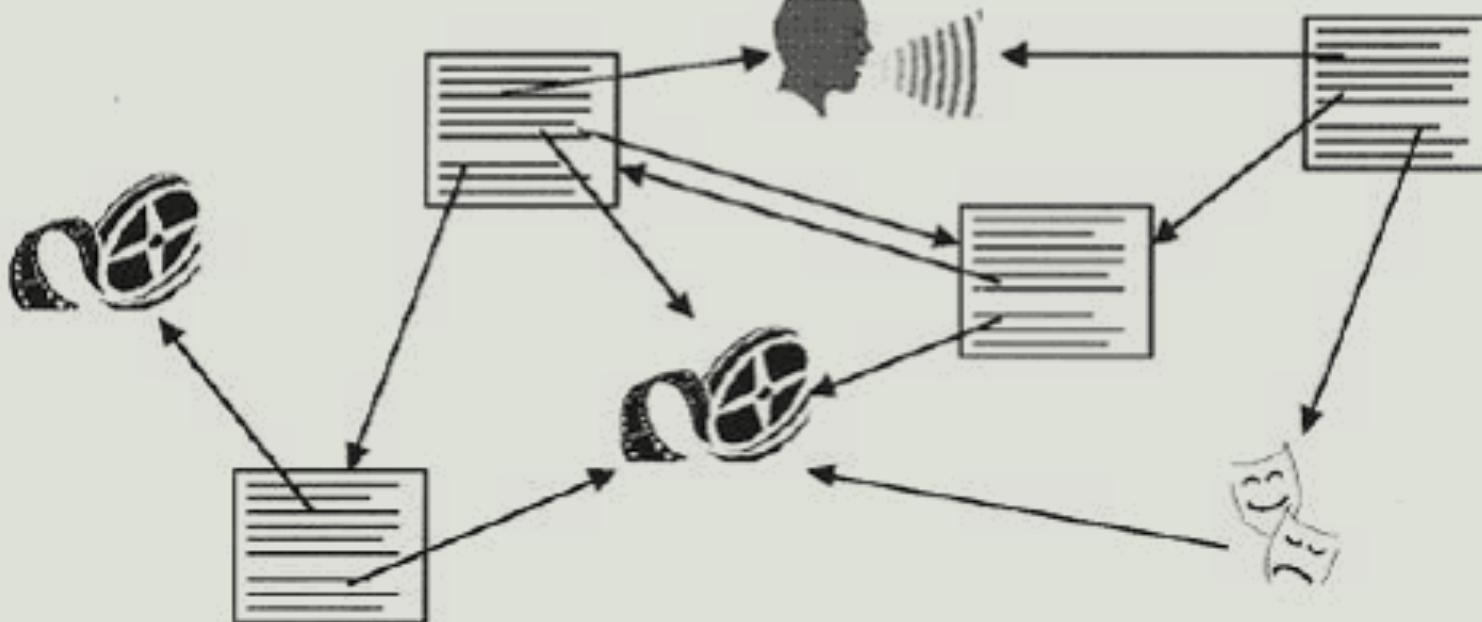
- non-linear text material



Hypertext & hypermedia

hypermedia

- media in a non-linearly organized system
- extension of hypertext
- ideas not products



Hypertext

A database that has active cross-references and allows the reader to ‘jump’ to other parts of the database as desired” Schneiderman, 1989 (Definizione)

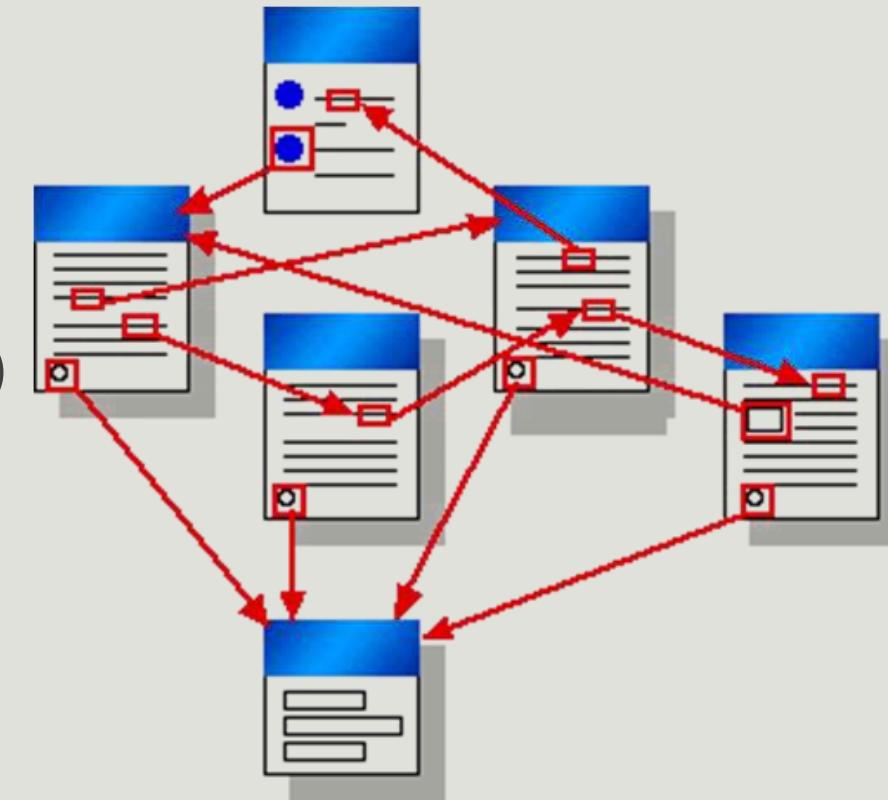
Possibilities

- return
- loop
- marking selected links

Nodes = parti del database (unità di informazione)

Links = collegamenti fra i nodi

Navigation = moving through the hypertext



Nodes

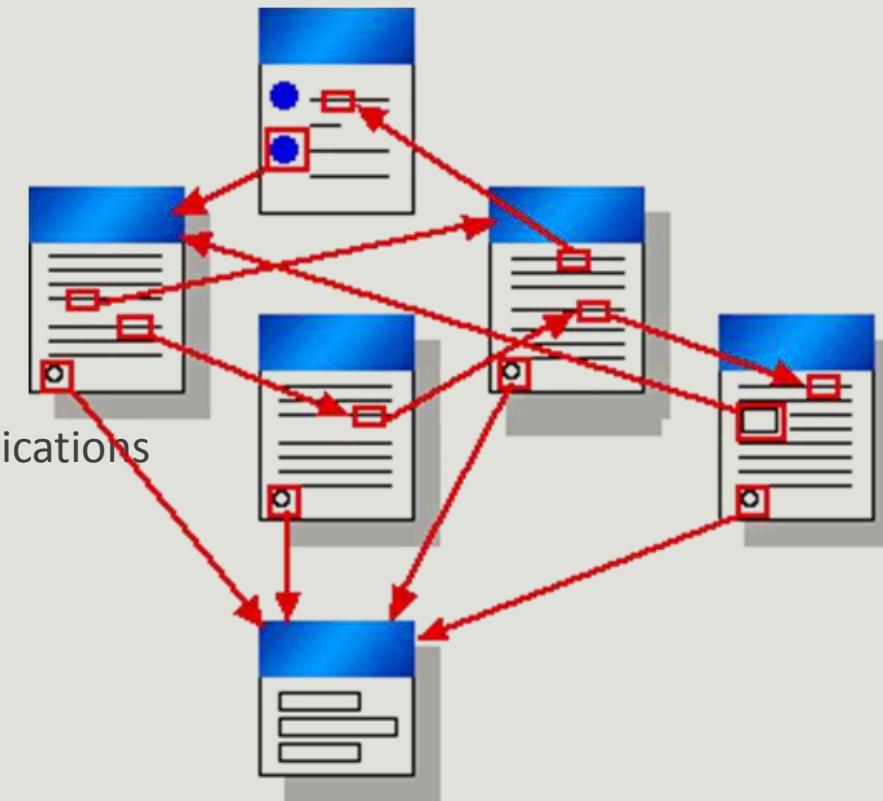
units of information

different names in different systems:

- pages (www)
- cards (HyperCard)
- articles (Hyperties)
- documents

may contain:

- text, graphics, animation, sound, video, images, applications



Links

they form a network of connections between nodes, they bind the hypertext
(lega)

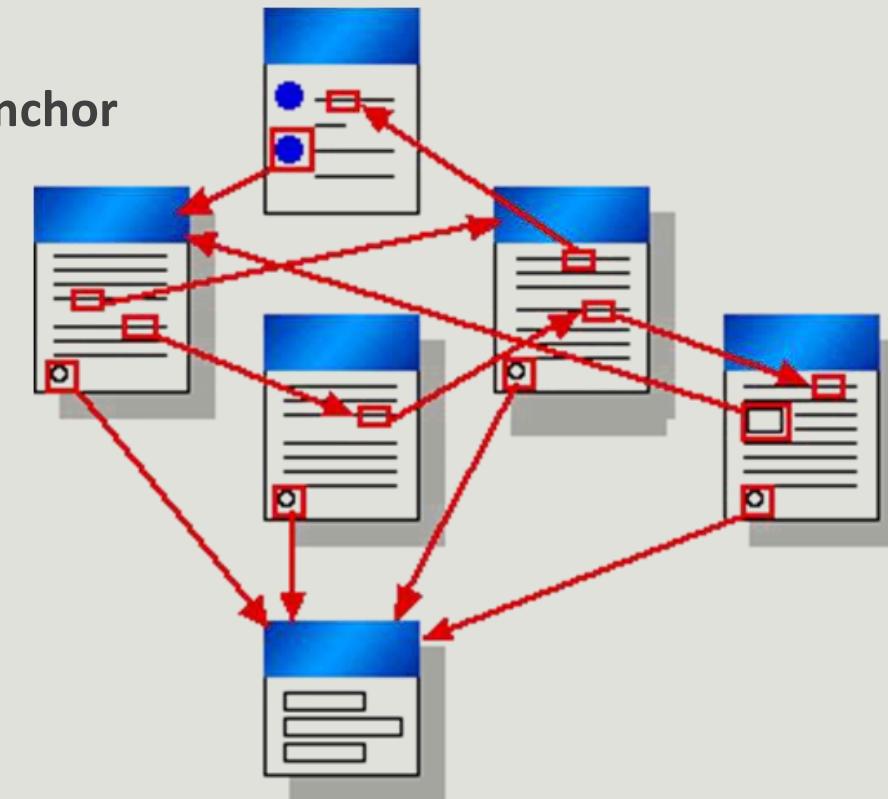
labels, pointers that connecting nodes

associated with a specific part of source node - anchor

what links can do?

- move to a new topic
- show a link
- provide additional information
- display image or video
- run applications

there is no direction in hyperspace



Navigation

the process of moving from one node to another by hypertext

Browsing

Indexing = quando le informazioni sono organizzate

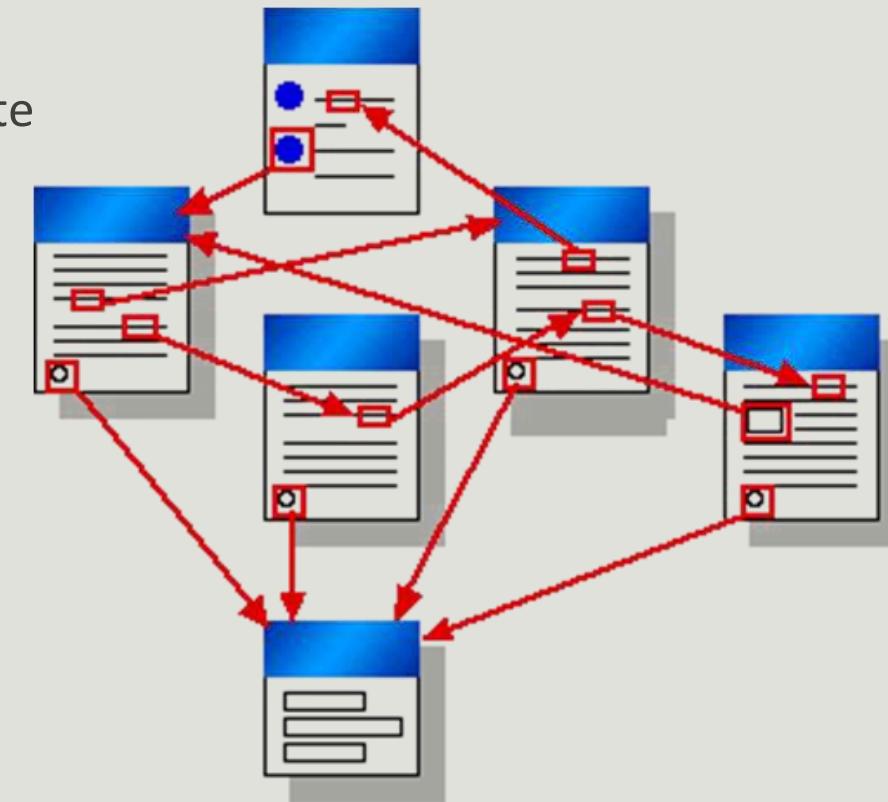
Searching = cercare per esempio per keyword

Filters

Tours

Bookmarks

Path



Application of hypertext

Internet

educational tools

a way to organize content in the database

entertainment

shopping on the Internet

not just web applications:

- encyclopedias,
- dictionaries,
- presentations,
- ...

IS Google MAKING US Stoopid?

rejection of other media (books and the press)

constraints in perception

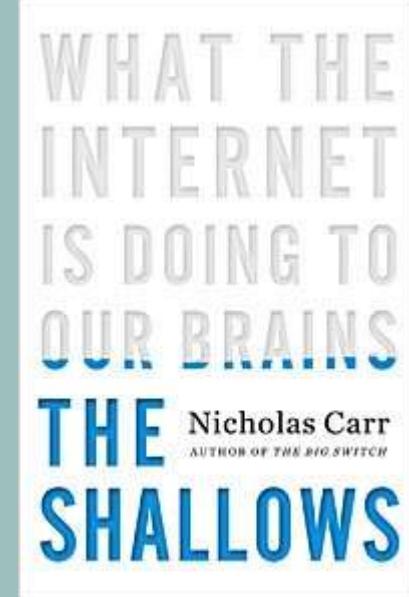
getting shorter and poorer speech

problems with the acquisition of long texts

Internet addiction

problems with concentration

easy access to any information



lack of criticism

Non-linear structure

molti percorsi conducono attraverso il materiale

navigazione associativa (così funziona anche il nostro cervello)

Un link è più semplice di una query



Advantages

- many paths leading through the material
- associative browsing
- control over screen content
- link is easier to use than complex queries

Disadvantages

- easy to lose „lost in hyperspace”
 - knowledge and contents
 - fragmentary information
 - no integration ... confusion
- navigation and structure
 - hyperlinks move across structure – where am I?

Non-linear structure

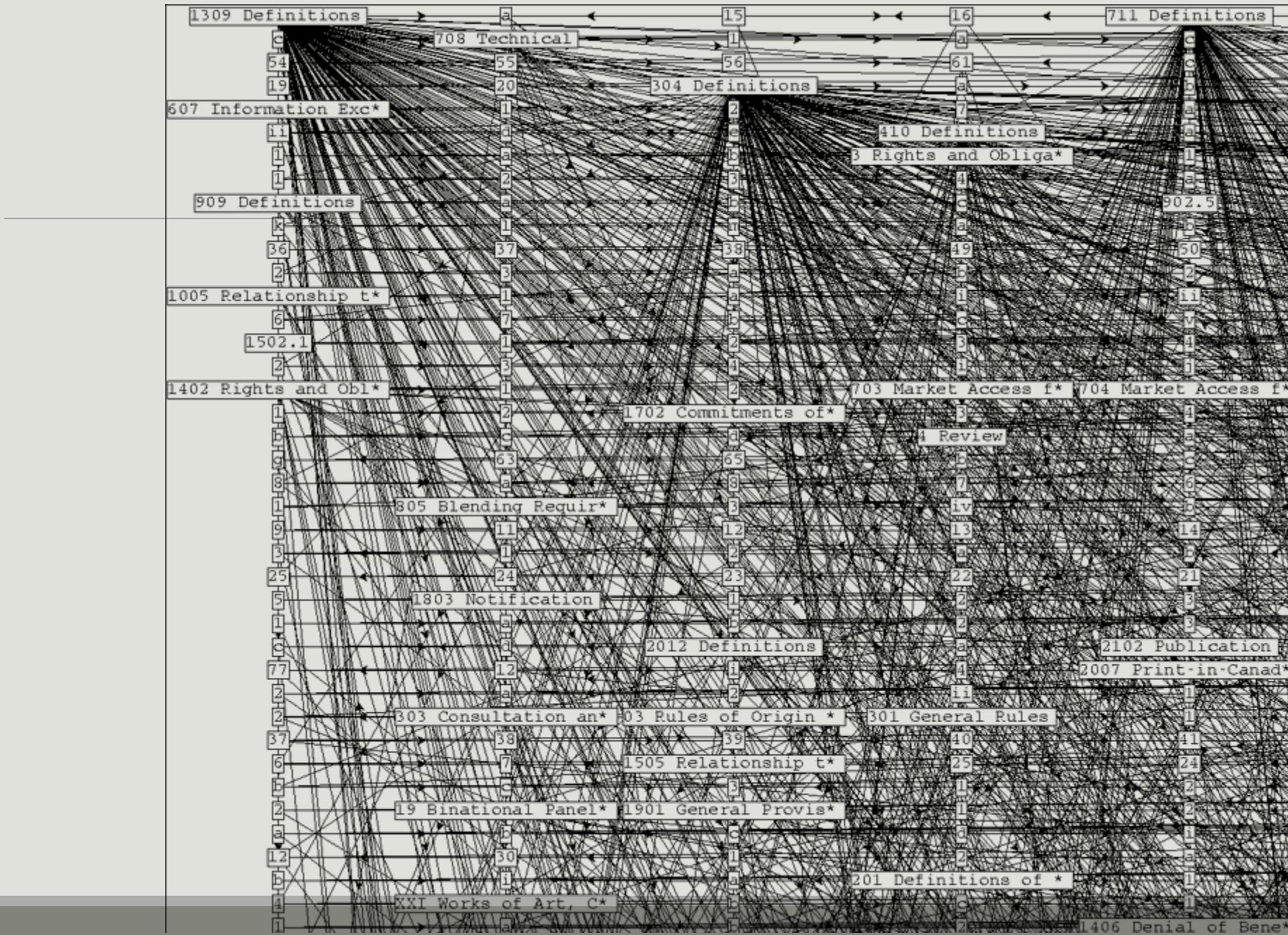


Disadvantages

- easy to lose „lost in hyperspace”
- an interesting node may be hard to find again in the future
- ease of looping
- inability to estimate available information

Advantages

- many paths leading through the material
- associative browsing
- control over screen content
- link is easier to use than complex queries



Hypertext Navigation

Goals

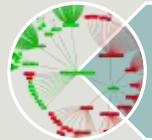
- Find some information
- Learn something

Problems

- Where am I?
- Where can I go from here?
- Where should I go?



Navigation support



whole picture

intera mappa del sistema con tutti i link e i nodi. Potrebbe risultare alla fine fin troppo grande



return path



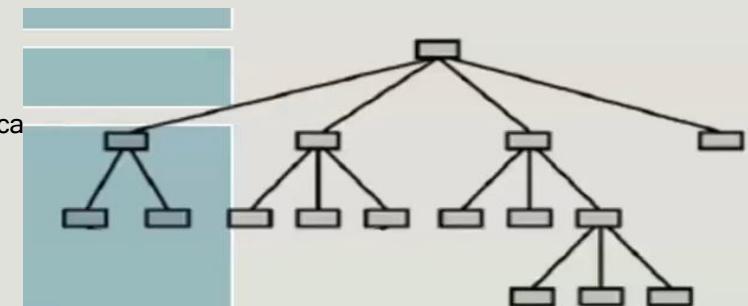
avoiding loops



using structure

using structure

- hierarchy A ogni livello l'informazione diventa più specifica
- sequence
 - forward, backward, home
- tree
 - top, ancestor, descendants, siblings



metaphor

Tipo di architettura di un sistema di ipertesti, proposto da due persone in basso

Architecture of hypertext systems

Attualmente pochi sistemi seguono questo modello, ma è abbastanza semplice da capire

user interface

nodes and links

storage, shared data,
and network access

Presentation Level

***Hypertext Abstract Machine
(HAM) Level***

this is the "enginee" of the system

Database Level

Campbell and Goodman

Architecture of hypertext systems

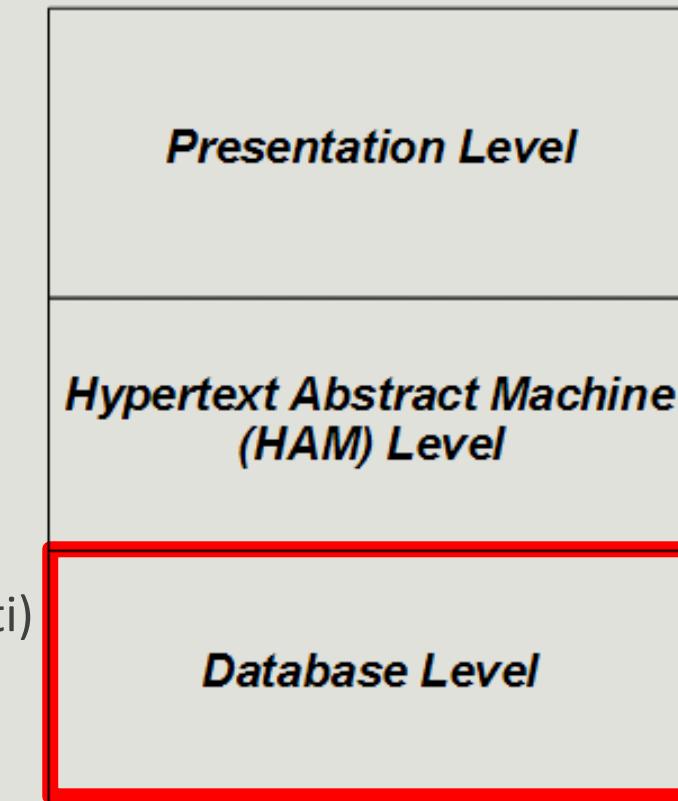
standard database

the information can be located at a local or remote computer

no matter how the information is stored - the speed of access to information is important

multi-user access to the information, various security considerations, including backup

sees the hypertext nodes and links, as just data objects (vede i nodi e i collegamenti, solo come oggetti)



Architecture of hypertext systems

an engine which manages all information about the hypertext and communicates with the application through a byte-stream protocol

knowledge of the form of the nodes and links and would know what attributes were related to each

(conoscenza della forma dei nodi e link e di come sono relazionati)

the ability to transfer information from one hypertext system to the other

Presentation Level

Hypertext Abstract Machine (HAM) Level

Database Level

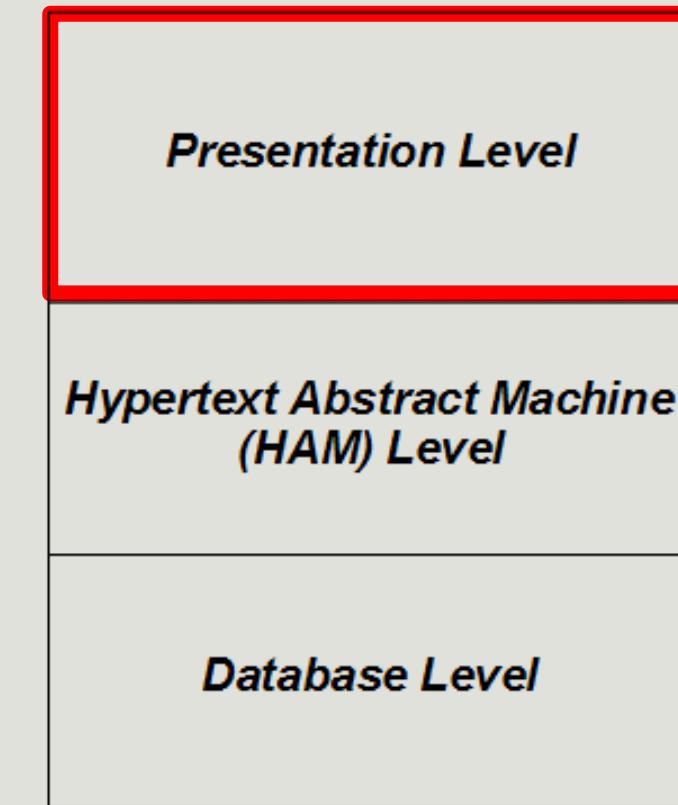
Architecture of hypertext systems

the level determines how to present the information in the HAM level

- nodes and links (how to present nodes and links)

access rights (filtration of information based on who has access)

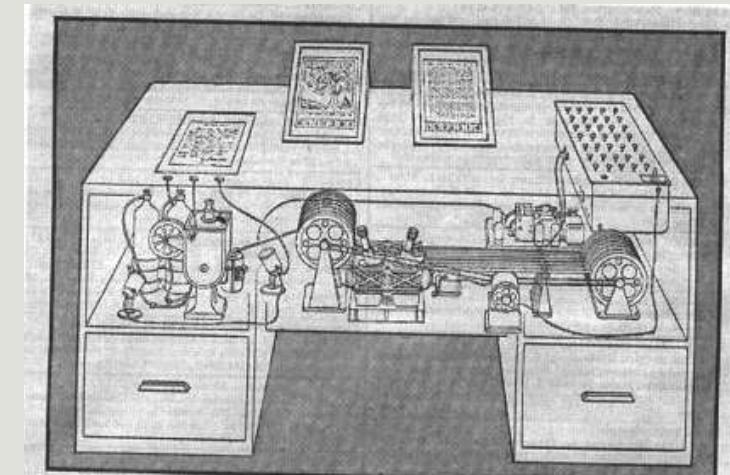
- filtering information



History

Vannevar Bush, MEMEX

- *Memory Extender*
- proposes **Memex** in the article “As We May Think”
- never implemented
- a mechanized device which would enable a user to view all sorts of written material and organize it arbitrarily, adding annotations and links
- ability to create links between items or documents
 - Combining links into **trails** of information relevant to given topics

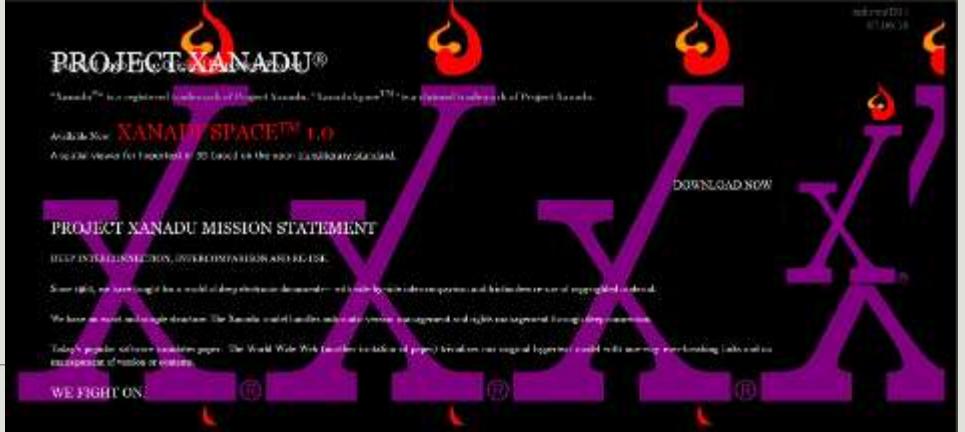


MEMEX in the form of a desk would instantly bring files and material on any subject to the operator's fingertips. Slanting translucent viewing screens magnify supermicro-film filed by code numbers. At left is a mechanism which automatically photographs longhand notes, pictures and letters, then files them in the desk for future reference.

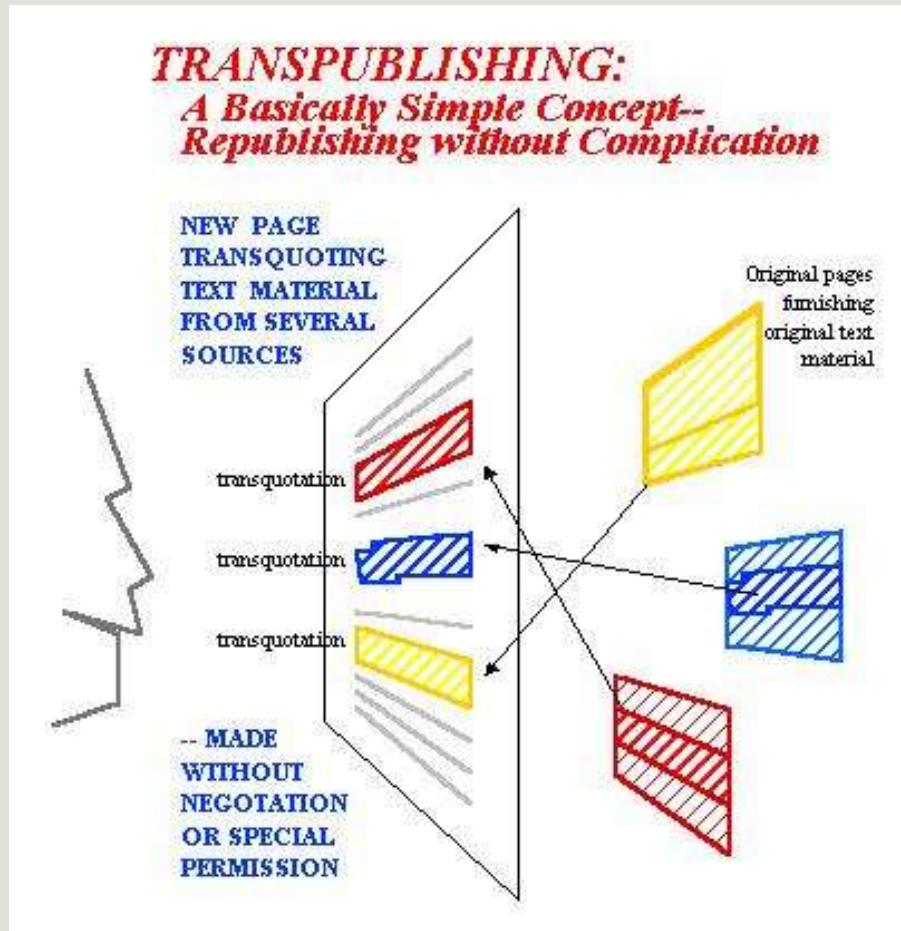
History

Ted Nelson, XANADU (1965), docuverse

- Ted Nelson introduces Xanadu and coins the term ‘hypertext’
- A repository for everything ever written
- Possible to address any substring of any document from any other location
 - Every byte in every document needs its own address
- Text is never deleted
 - All versions can be generated from the latest version
- Author of every document is known and s/he gets royalties based on how many people read how many bytes of author’s work
- XANADU has never been completed



History



each transquotation comes, in effect, from the original publisher; the original publisher supplies the quotation to each user

transclusion is the inclusion of part or all of an electronic document into one or more other documents by hypertext reference

Here are some real transquotations, coming from other pages by permission of their owners.

One of my students wrote a delightful humorous essay; and here is a transquotation from

it:  ... I believe that just as extreme intelligence is a gift, so is extreme stupidity.
(The first special quote-mark goes to the original context; the second special quote-mark goes to the copyright permission statement.)

History

van Dam, HES (1967)

- Hypertext Editing System
- Ran in 128K on an IBM/360 mainframe
- Supported by IBM, who sold to the Houston Manned Spacecraft Center
 - Used to produce documentation for the Apollo space program



History

HyperCard (1987)

HyperTalk - programming language

Time since you were here: NEVER

Current report overview map

```
graph TD; Root[HyperTEXT'87 Workshop] --> People[People]; Root --> Literature[Literature]; Root --> Systems[Systems]; Root --> Applications[Applications]; Root --> ResearchIssues[Research issues]; Root --> Definition[Definition]; ResearchIssues --> CSCW86TripReport[CSCW'86 Trip Report]
```

Current chapter overview map

```
graph TD; Root[Classification of HT systems] --> Hypertext[Hypertext systems]; Root --> NoteCards[NoteCards]; Root --> DocumentExaminer[Document Examiner]; Hypertext --> HyperCard[HyperCard (Apple)];
```

Hypertext systems

Classifying hypertext systems (Frank Halasz)

Frank Halasz from MCC gave the last talk at the workshop. He and the organizing committee should be criticized for not making it the first talk AND the last talk: Part of the talk was a very good survey of what HT really is and a classification of current systems. This material could have filled a whole talk with no problems but was presented with such speed that it left the audience breathless. It would also have made a good platform for the discussions during the conference if it had been presented at the beginning instead of at the end.

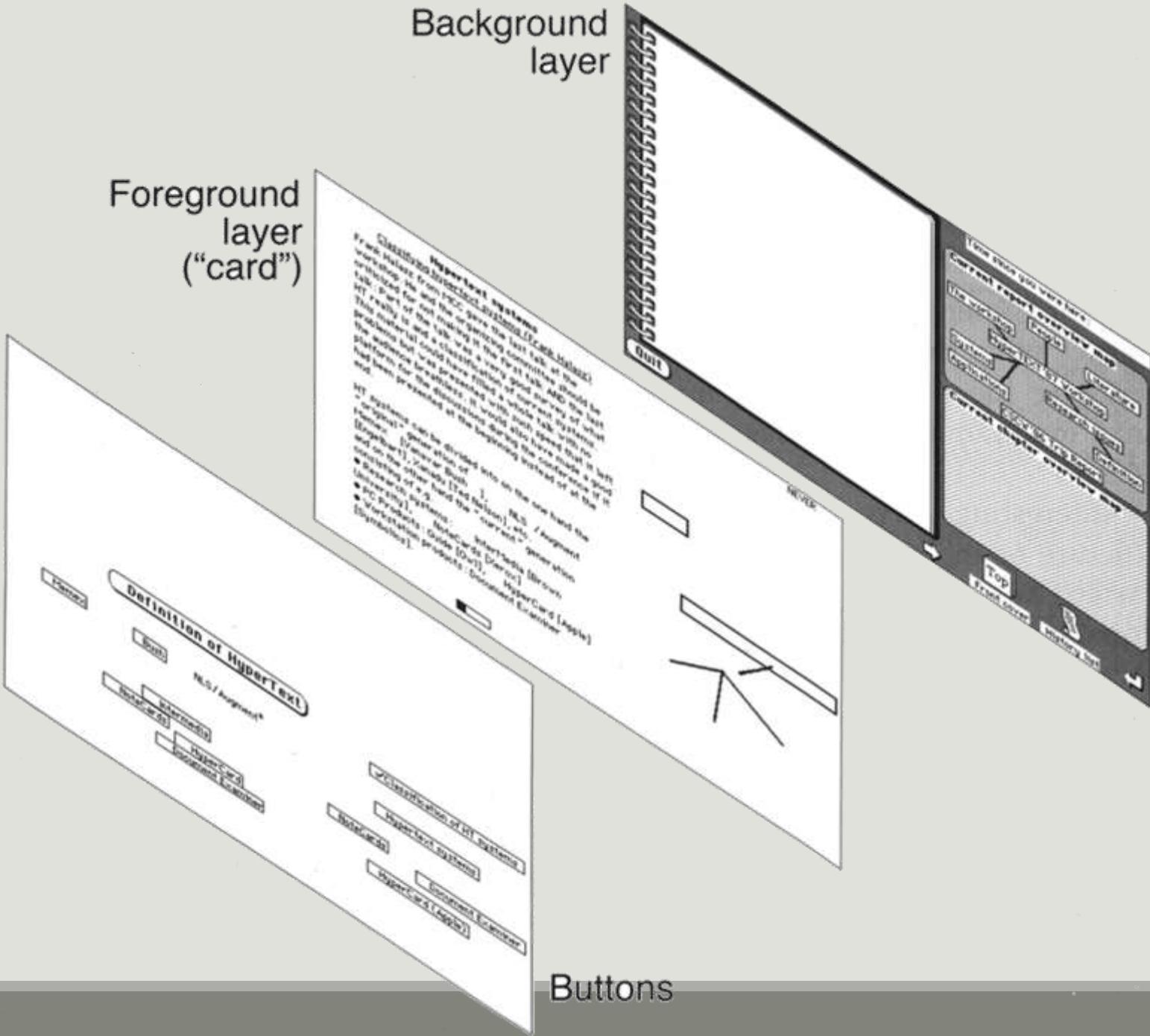
Definition of HyperText

HT systems can be divided into on the one hand the "original" generation of Memex [Vanavar Bush], NLS/Augment* [Engelbart], Xanadu [Ted Nelson], etc. and on the other hand the "current" generation consisting of e.g.

- Research systems: Intermedia [Brown University], NoteCards [Xerox]
- PC Products: Guide [Owl], HyperCard [Apple]
- Workstation products: Document Examiner [Symbolics].

Quit → Front cover History list ← Top





History

Tim Berners-Lee (1989)

project goal: easy exchange of information between scientists using the hypertext system

Web, integration of text documents, graphics, image, sound

he developed the syntax of HTML

the first WorldWideWeb browser

the first server www (1991)

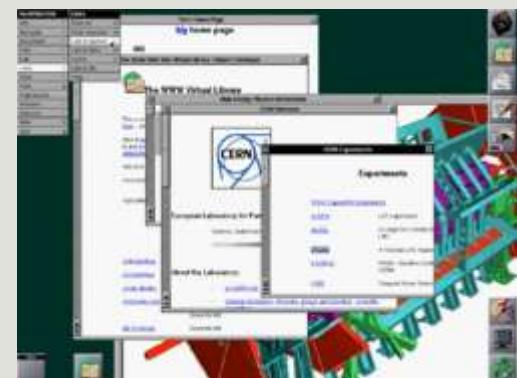
World Wide Web Consortium (W3C)

A screenshot of a web browser window showing the homepage of the first website. The URL is http://info.cern.ch. The page title is "Niezabezpieczona | info.cern.ch". Below the title, there's a menu bar with "Aplikacje", "AM", "angielski", "Ćwiczenia", "Dom", "Finance", and "Inne zakłady". The main content area has the heading "http://info.cern.ch - home of the first website" and the text "From here you can:" followed by a bulleted list:

- Browse the first website
- Browse the first website using the line-mode browser simulator
- Learn about the birth of the web
- Learn about CERN, the physics laboratory where the web was born

A screenshot of a web browser window displaying a document titled "WorldWideWeb: Proposal for a HyperText Project". The document header includes "To:", "Cc:", "From:", and "Date:". The body of the document contains several paragraphs of text, including:

The attached document describes in more detail a Hypertext project.
HyperText is a way to link and access information of various kinds as a web of nodes in which the user can browse or sift. It provides a single user-interface to large classes of information (reports, notes, data-base, on-going documentation and on-line help). We propose a simple scheme incorporating servers already available at CERN.
The project has two phases: firstly we make use of existing software and hardware as well as implementing simple browsers for the user's workstation, based on an analysis of the requirements for information access needs by experiments. Secondly, we extend the application area by also allowing the user to add new material.
Phase one should take 3 months with the full manpower complement; phase two a further 3 months, but this phase is more open-ended, and a review of needs and status will be incorporated into it.



A horizontal timeline arrow pointing from left to right, divided into four segments by vertical dashed lines. The segments are color-coded: light beige for the first two segments and light gray for the last two. Above the timeline, key milestones are listed with their corresponding years and descriptions.

| Date | Event / Description |
|------|---|
| 1945 | Vannevar Bush Memex |
| 1965 | Ted Nelson hypertext |
| 1967 | The HES, Andy van Dam |
| 1978 | Aspen Movie Map , first hypermedia videodisk, Andy Lippman, |
| 1987 | Apple introduces HyperCard, Bill Atkinson |
| 1987 | Hypertext'87 first major conference on hypertext |
| 1991 | WWW first global hypertext, Tim Berners-Lee |

1965 Ted
Nelson
hypertext

1978 Aspen
Movie Map ,
first
hypermedia
videodisk,
Andy
Lippman,

1987
Hypertext'87
first major
conference
on hypertext

Example of hypertext system

www

Web is a “weak” hypertext system

„The Xanadu® project did not "fail to invent HTML". HTML is precisely what we were trying to PREVENT-- ever-breaking links, links going outward only, quotes you can't follow to their origins, no version management, no rights management.

The "Browser" is an extremely silly concept-- a window for looking sequentially at a large parallel structure. It does not show this structure in a useful way.”

Ted Nelson

Hypermedia ≠ World Wide Web ◦ The Web is one type of hypermedia application but does not illustrate all hypermedia concepts.

Hypertext & hypermedia

WIOLETA SZWOCH DR

DEPARTMENT OF INTELLIGENT INTERACTIVE SYSTEMS

Hypertext

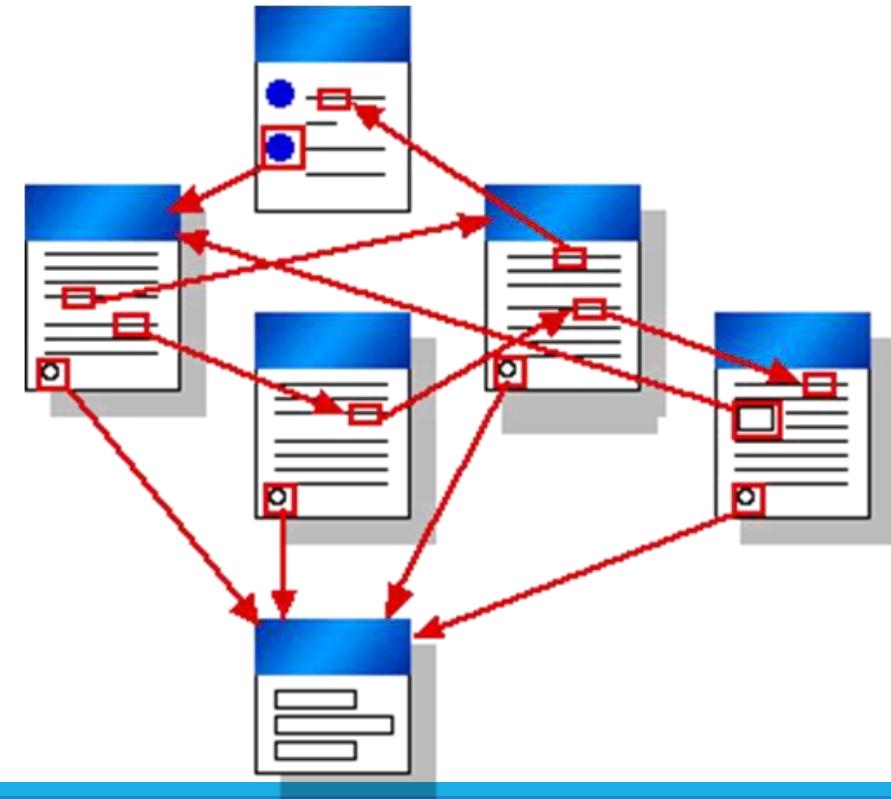
Hypertext - a database that has active cross-references and allows the reader to ‘jump’ to other parts of the database as desired

Schneiderman, 1989

Nodes – units of information

Links – labels connecting nodes

Navigation – process of moving through the hypertext database



Creating web pages



HTML
contents



CSS
presentation



JavaScript
action

Nel corso non vedremo javascript



HTML

HyperText

- used to write content on the hypertext system (web)

Markup

- the document is built on tags

Language

- the document is built according to syntax rules, and based on a defined alphabet - tags

```
5 <main id="content">
6   <div>
7     <div data-drupal-messages-fallback class="hidden"></div>
8   <div id="block-politechnika-gdanska-content" class="block block-system block-system-
9     main-block">
10
11     <article role="article" about="https://eti.pg.edu.pl/en/students">
12
13       <div>
14         <div class="page-content">
15           <div class="container">
16             <div class="row">
17               <div class="col-lg-8 mb-3 mb-lg-0">
18
19                 <div class="block block-layout-builder block-field-blocknodesitebody">
20
21                   <div>           <p>The classes at the Faculty of Electronics, Telecommunication and Informatics of
24 Gdańsk University of Technology in the winter semester 2021/2022 start on the
25 1st October 2021 and will be conducted as follows:          </p>
26
27 <p>Hybrid mode – Undergraduate (BA) and Postgraduate (MA) full-time</p>
28
29   </div>
30
31   </div>
32
33   </div>
34   <div class="col-lg-4">
35
36     <div class="block block-entity-hierarchy-microsite block-entity-
37     hierarchy-microsite-menu">
```

https://eti.pg.edu.pl/en/students

Give PG

Employees Alumni Cooperation ETI community

Students Admission Research Faculty

What are you looking for?

ETI > Students

Students



The classes at the Faculty of Electronics, Telecommunication and Informatics of Gdańsk University of Technology in the winter semester 2021/2022 start on the 1st October 2021 and will be conducted as follows:

Hybrid mode – Undergraduate (BA) and Postgraduate (MA) full-time

Students

- News
- Registrar's Office
- Diploma Thesis
- Freshman
- Erasmus +
- Moja PG Portal

Useful links

Student Affairs Office

ECTS Information Package

HTML

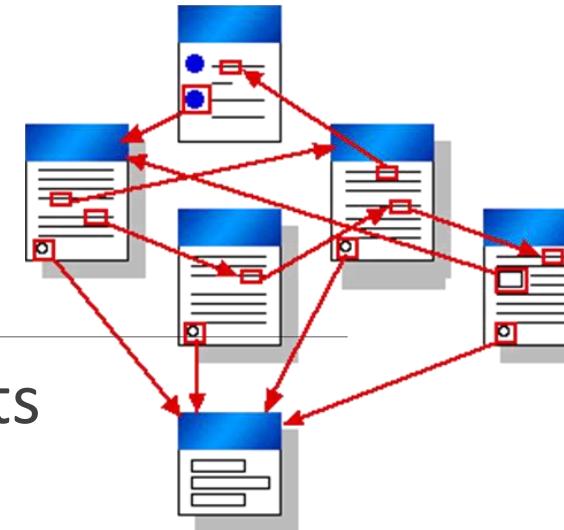
language describing the structure of web pages documents

- content + formatting (graphic layout)
- text, multimedia, hyperlinks

language defining documents

- set of tags
- describe the look of the website
- tags - formatting instructions
- each tag describe different content of the document

a fixed set of tags



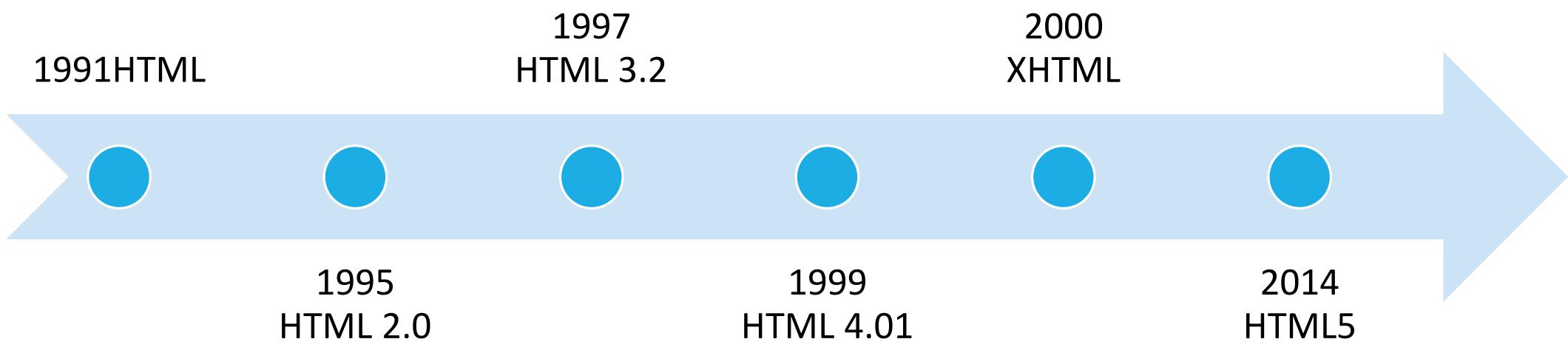
History of HTML

HTML vs XHTML

different writing style

some restrictions

the same tags



HTML - advantages and disadvantages



- widely used, every browser supports HTML
- simply syntax
- easy to learn and use
- you can integrate HTML with CSS, JavaScript, php etc.

- content and presentation
- no information about logical structure of document, it's a language for presentation of the content
- is not extensible, fixed set of tags

Non c'è separazione fra contenuto e forma

Tags, elements

keywords, tags do have a specific meaning

- <tag> <html> <p> <nav> <div> <h1>

tags normally come in pairs:
start tag (opening tag) and
end tag (closing tag)

- <tag> content </tag>

empty element

- <tag/>
- <tag></tag>

Attributes

specificano particolari caratteristiche per un elemento

elements can have
attributes

attributes provide
additional information
about an element

attributes are always
specified in the start tag

- ``
- `<tag attrName="value"> content </tag>`
- `<tag attrName="value"/>`

HTML document structure (basic)

```
<!DOCTYPE html>
<html>
  <head>
    tags that describe the content of the document
  </head>
  <body>
    tags that create the content of the document
  </body>
</html>
```

Spesso ci può essere il
percorso dov'è il file CSS

HTML document structure (basic)

```
<!doctype html>
```

- a declaration, lets the browser know that you are using HTML5

Serve per far capire al browser che è un file html

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

tags that describe the content of the document

```
</head>
```

```
<body>
```

tags that create the content of the document

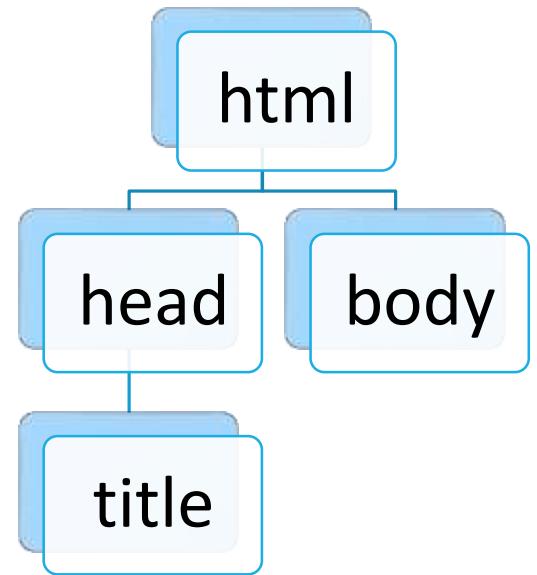
```
</body>
```

```
</html>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

HTML document structure (basic)

```
<!DOCTYPE html>
<html>
  <head>
    tags that describe the content of the document
  </head>
  <body>
    tags that create the content of the document
  </body>
</html>
```



<head>

<title>

- defines a title for the document

<script>

- define a client-side script (JavaScript); functions to handle events

<style>

- style of presentation

Magari il file CSS

<meta>

- metadata about the HTML, information for browsers

<link>

- defines the relationship between a document and an external resource

<head>

```
<meta charset="utf-8">
<title>Document Title</title>
<link rel="stylesheet" href="style.css">
<script src="script.js"></script>
</head>
```

Page title

← → ⌂ file:///F:/PG/HiH/Przykłady/HTMLPage1.htm

Lecture Laboratory Project

Page content

Inspektor Konsola Debugger Sieć Edytor stylów Wydajność Pamięć Dane ...

Szukaj w kodzie HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page title</title>
  </head>
  <body>
    <nav>
      <a href="/wyk/">Lecture</a>
      <a href="/lab/">Laboratory</a>
      <a href="/proj/">Project</a>
    </nav>
    <br>
    Page content
  </body>
</html>
```

Filtruj style :hover .cls + ☀ Nie zaznaczono elementu.

Układ Wyliczone Lokalne zmiany Czcionki

Flexbox Wybierz kontener Flex lub element, aby kontynuować.

Siatka CSS Grid nie jest używany na tej stronie

Model pudelkowy

margin: 0
border: 0
padding: 0
content: 0x0

Właściwości modelu pudelkowego

| | |
|------------|-------------|
| box-sizing | content-box |
| display | none |
| float | none |

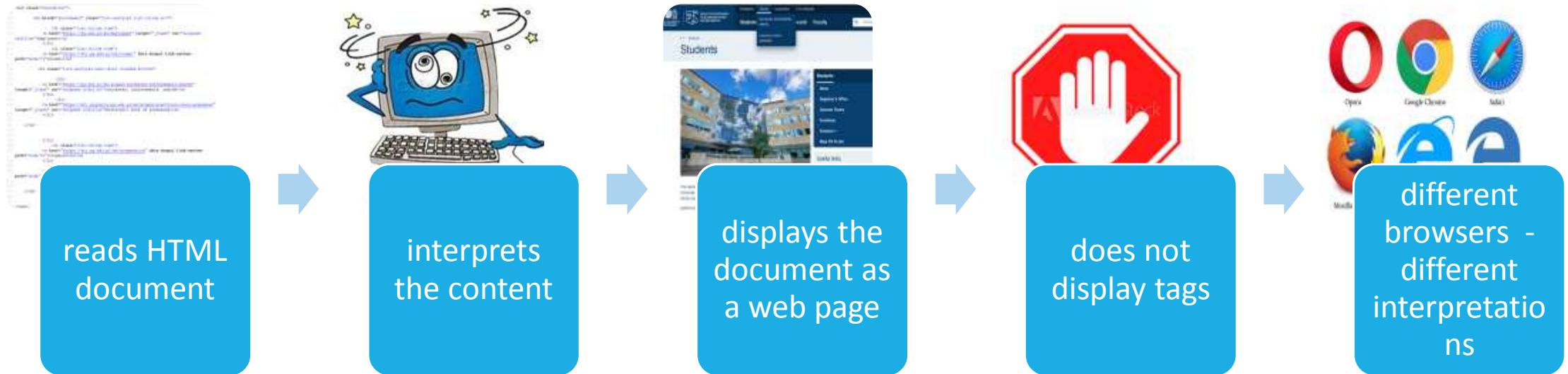
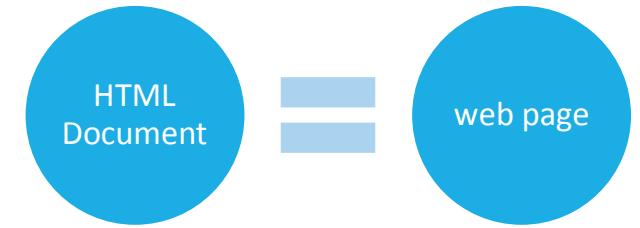
html > head > title

Filtruj zawartość Błędy Ostrzeżenia Wpisy dziennika Informacje Wpisy debugowania CSS XHR Żądania

»

HTML in browsers

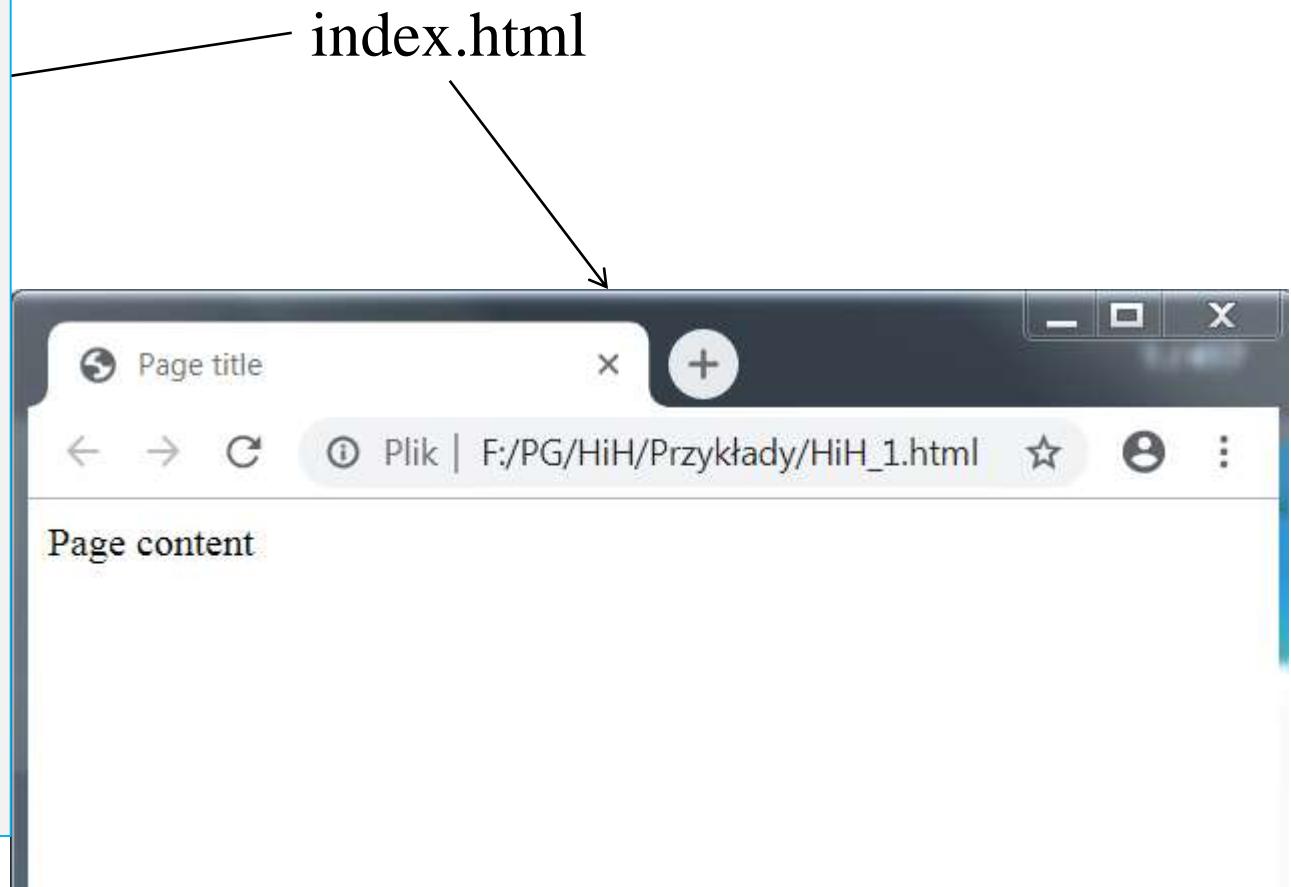
What does the browser do?



HTML in browsers

Per scoprire tutti i tag, arragiati e cerca le reference su internet

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Page title</title>
  </head>
  <body>
    Page content
  </body>
</html>
```



Validation

Per controllare se il codice HTML è corretto

The screenshot shows a web browser window with the URL <http://validator.w3.org/check> in the address bar. The page title is "W3C [Valid] Markup Validation o...". The browser toolbar includes icons for back, forward, search, and various extensions like Bing, Skydrive, and Facebook.

The main content area features the "W3C® Markup Validation Service" logo and the subtext "Check the markup (HTML, XHTML, ...) of Web documents". Below this, there are navigation links "Jump To: Congratulations · Icons".

A green banner at the top states "This document was successfully checked as HTML 4.01 Transitional!".

The form fields are as follows:

| | |
|----------------------|--|
| Result: | Passed |
| File : | <input type="file"/> Przeglądaj... <small>Use the file selection box above if you wish to re-validate the uploaded file ankieta.htm</small> |
| Encoding : | iso-8859-2 <input type="button" value="detect automatically"/> |
| Datatype : | HTML 4.01 Transitional <input type="button" value="detect automatically"/> |
| Root Element: | HTML |

HTML tags – text formatting

heading commands

- `<h1>, <h2>, ..., <h6>`

paragraph

- `<p>Paragraph content</p>`

span

- `text`

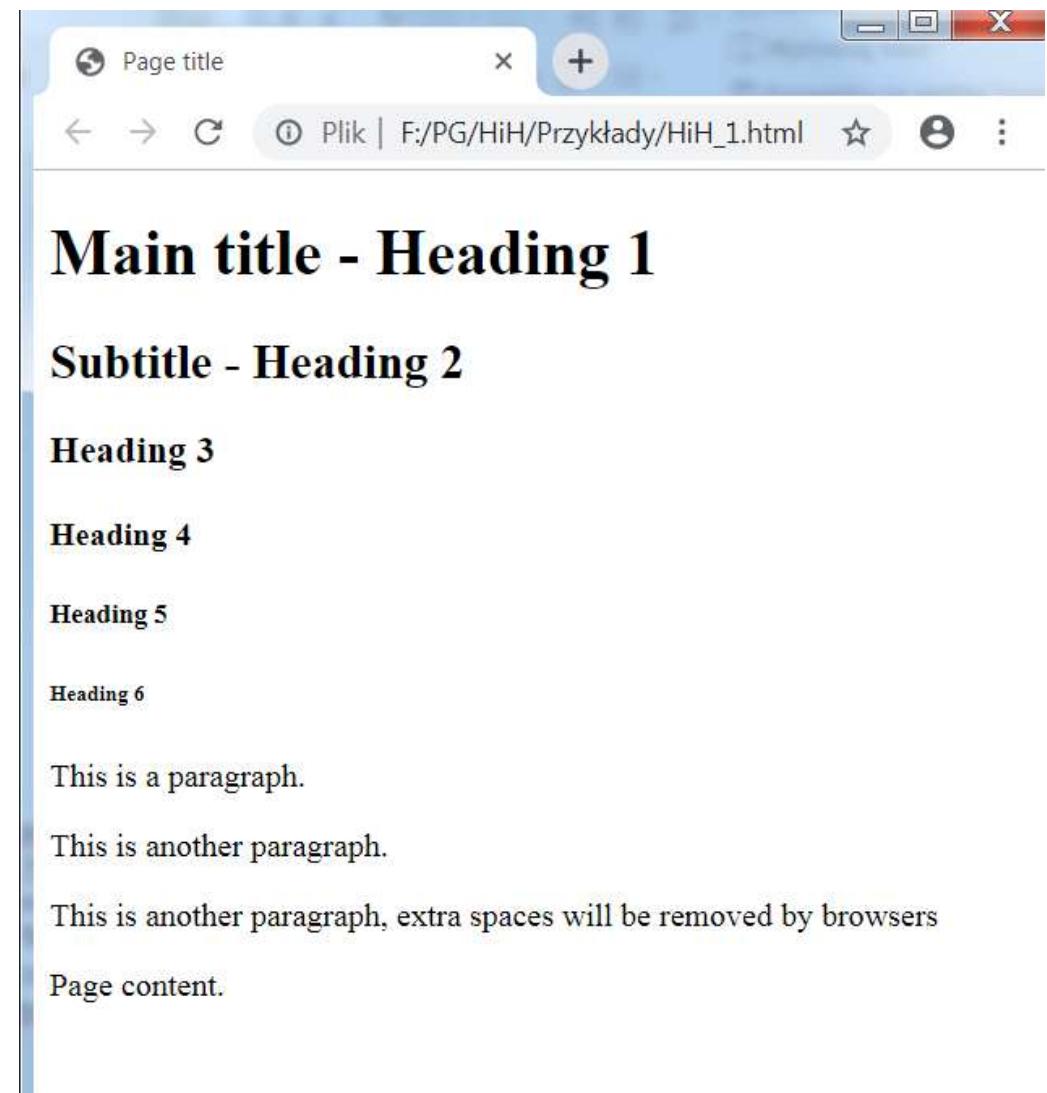
Per separare il testo, non ho capito
in che modo

single line break

- `
`

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Page title</title>
  </head>
  <body>
    <h1>Main title - Heading 1</h1>
    <h2>Subtitle - Heading 2</h2>
    <h3>Heading 3</h3>
    <h4>Heading 4</h4>
    <h5>Heading 5</h5>
    <h6>Heading 6</h6>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
    <p>This is another paragraph,           extra      spaces
       will be removed by browsers</p>
    Page content.

  </body>
</html>
```



Simple text formatting

content-based tags, logical tags

- selecting an area having a specific context or meaning indirect effect on the appearance
- `Bold Text Here`
`Italicized Text Here`

Vedi che entrambe le funzioni portano allo stesso risultato,
anche se il significato è diverso (riasculta)

Bold Text Here
Italicized Text Here

physical style tags

- direct selection of fonts, sizes and colors
- `Bold Text Here`
`<i>Italicized Text Here</i>`

Hyperlink tag

one of the fundamental HTML mechanisms

used for navigation

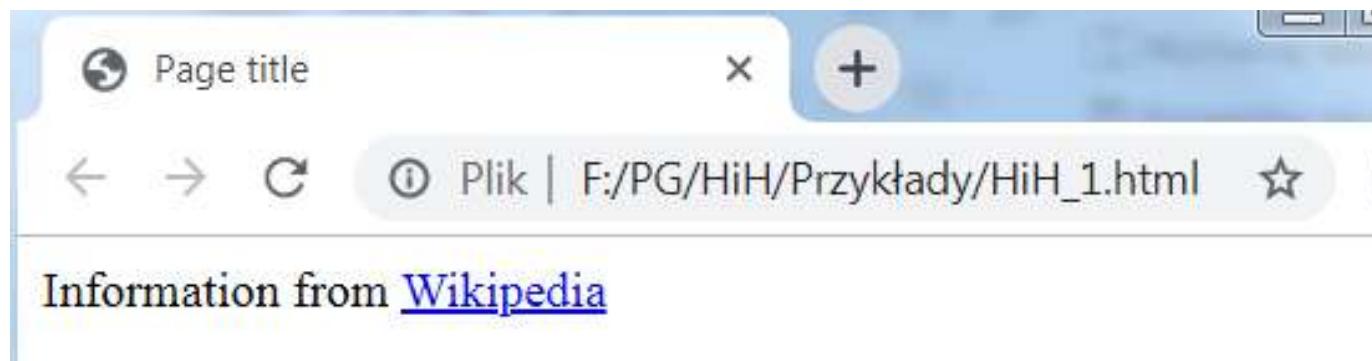
- inside an HTML document
- between different HTML documents

`place we can click `

Hyperlink tag

Information from

 Wikipedia



```
<h1>Table of Contents</h1>
<a href="#Topic1">
    Click to jump to the First Topic
</a>
<a href="#Topic2">
    Click to jump to the Second Topic
</a>

<h2 id="Topic1">First topic</h2>
<p>Content about the first topic</p>
<h2 id="Topic2">Second topic</h2>
<p>Content about the second topic</p>
```

Collegamento a un punto preciso di un'altra pagina

```
<a href="page1.html#Topic1">Click to jump to the First Topic </a>
```

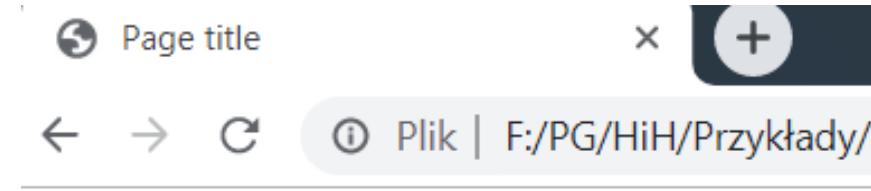


Table of Contents

[Click to jump to the First Topic](#)

[Click to jump to the Second Topic](#)

First topic

Content about the first topic

Second topic

Content about the second topic

Hyperlink tag <a>

```
<a href="https://www.wikipedia.com">  
    Wikipedia  
</a>
```

Open link in a new window or tab:
`<a href="https://www.w3schools.com"
 target="_blank">
 Visit W3Schools
`

```
<a href="/images/myimage.jpg"  
    download="image1">  
    Download the image  
</a>
```

href

target

- `_blank`: a new window or tab
- `_self`: the same window it was clicked in
- `_parent`: the parent window
- `_top`: the full window size of the browser

download

- `filename`

Hyperlink tag

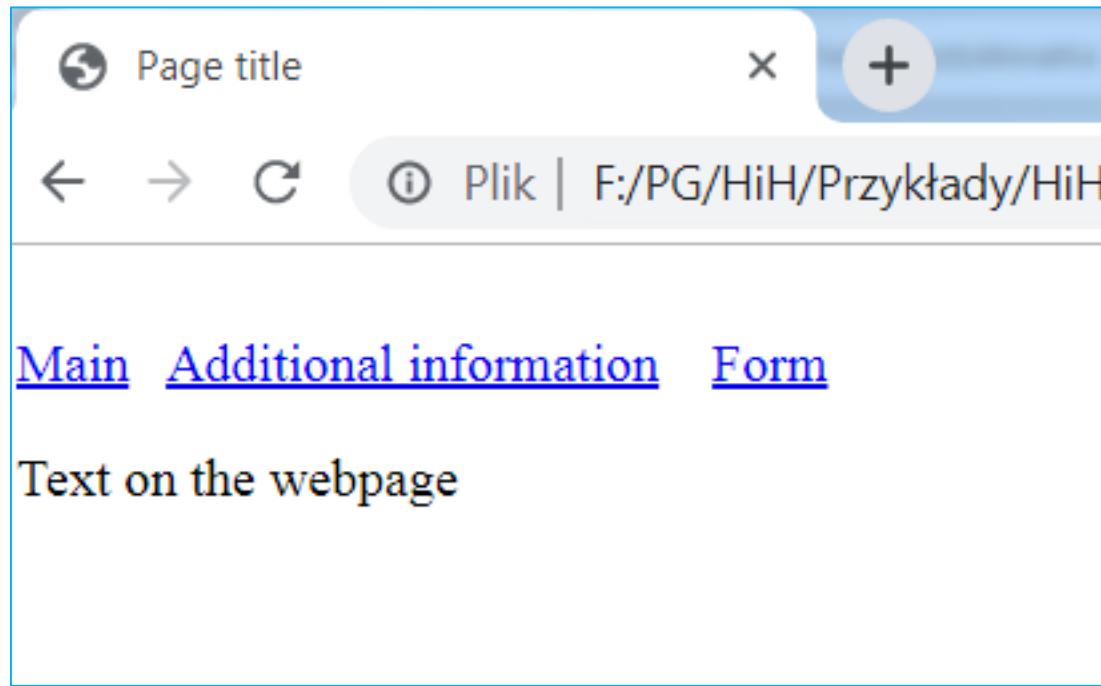
absolute

- always the same location (domain and folder)
- ``
 Text Here
``

relative

- download file in relation to the HTML file
- ``
 Text Here
``

Meglio questa!!! più
versatile



Text on the webpage

```
<a href="Page1.html">Main</a>
<a href="Page2.html">Additional information</a>
<a href="Page3.html">Form</a>
```

Images in HTML

<figure>

<picture>

<map>

Images in HTML



```

```

src

- URL - web address

alt

- alternate text

title

- extra information about an element

**width,
height**

- size modification

Global attributes

attributes that can be used with all HTML elements

class

id

title

style

...



Smile

Non è un sostituto di è qualcosa di più avanzato

Images in HTML

```
<figure>
  
  <figcaption>Smile </figcaption>
</figure>
```

<figure>

- container

<figcaption>

- associate a caption with the image

Images in HTML



In questo caso l'immagine è cliccabile a porta da un'altra parte

```
<a href="page1.html">  
    
</a>
```

Image maps

```

<map name="shapes">
  <area shape="polygon" coords="79,6,5,134,153,134" href="wikipedia.pl">
  <area shape="rectangle" coords="177,6,306,134" href="onet.pl">
  <area shape="circle" coords="397,71,65" href="wp.pl">
</map>
```

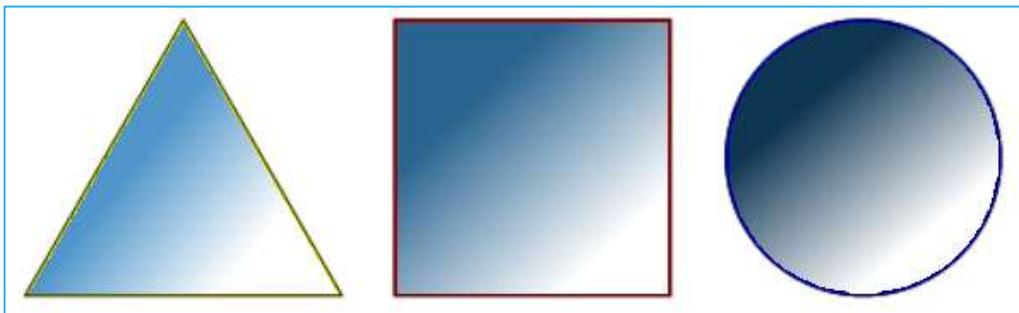


Image maps

an image-map is an image with clickable areas

map <map> attributes and subelements

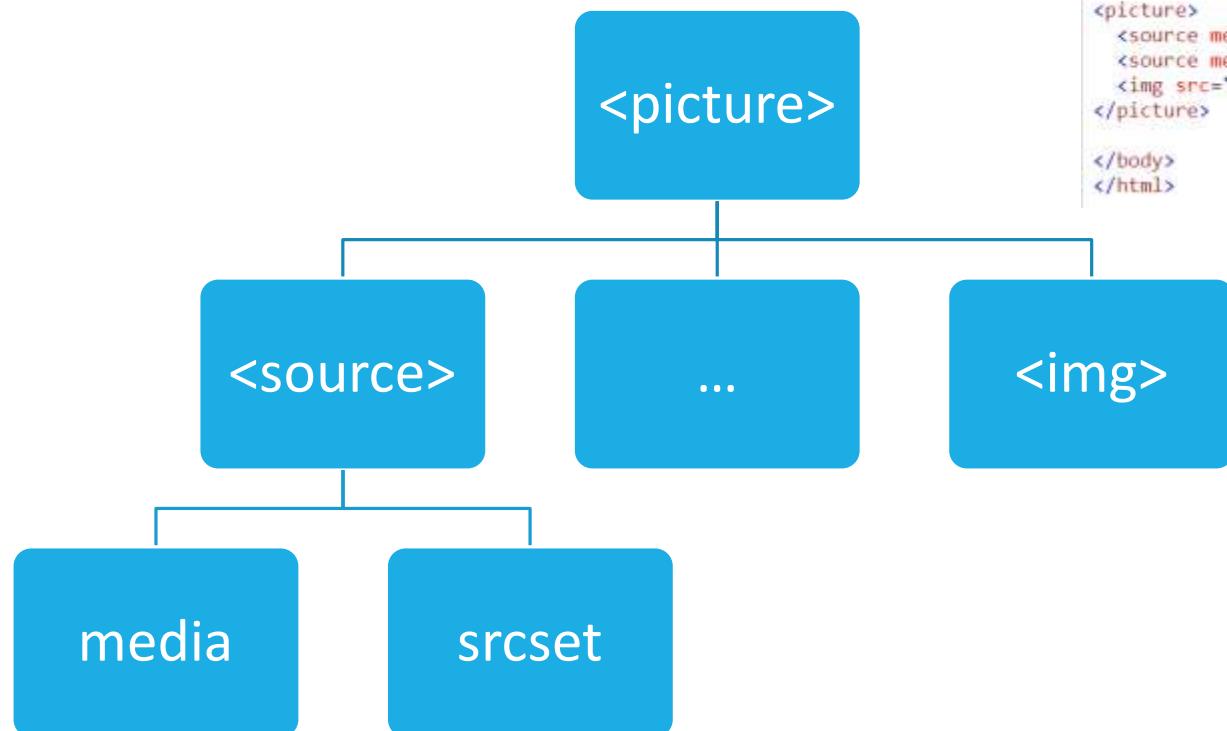
- name
- <area>
 - coords
 - shape
 - href

image attributes

- usemap
 - map name

Un altro tag ancora...

Images in HTML



```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
<body>

<h1>The picture element</h1>

<p>Resize the browser window to load different images.</p>

<picture>
  <source media="(min-width:650px)" srcset="img_pink_flowers.jpg">
  <source media="(min-width:465px)" srcset="img_white_flower.jpg">
  
</picture>

</body>
</html>
```

Result Size: 369 x 559

Get your own website

The picture element

Resize the browser window to load different images.



Result Size: 467 x 559

Get your own website

The picture element

Resize the browser window to load different images.



Tables

<table> ... </table> the whole table

<tr> ... </tr> each row

<td> ... </td> each cell in row

<th> ... </th> cell from the header row

possibility to present numerical data

create tabular data sets

cell contents

- any element: text, image, link, form, etc.

```
<table>
  <tr>
    <th>Test</th>
    <th>Score</th>
  </tr>
  <tr>
    <td>1</td>
    <td>20</td>
  </tr>
  <tr>
    <td>2</td>
    <td>30</td>
  </tr>
  <tr colspan="2">
    <td>Sum: 50</td>
  </tr>
</table>
```

| Test | Score |
|---------|-------|
| 1 | 20 |
| 2 | 30 |
| Sum: 50 | |

colspan per espandere unire due colonne

Lists

organizing information in the form of an enumeration

unordered list

``

ordered list

``

description list

`<dl>`

Lists

unordered list

```
<ul>
  <li>Item</li>
  <li>Next item</li>
  <li>Yet another item</li>
</ul>
```

- Item
- Next item
- Yet another item

Lists

ordered list ``

```
<ol>
  <li>HTML</li>
  <li>XML Schema</li>
  <li>XSL</li>
</ol>
```

1. HTML
2. XML Schema
3. XSL

Lists

ordered list

```
<ol start="3">
  <li>Item</li>
  <li value="7">Next item</li>
  <li>Yet another item</li>
</ol>
```

- 3. Item
- 7. Next item
- 8. Yet another item

```
<ol start="10" type="I" reversed>
  <li>Item</li>
  <li>Next item</li>
  <li>Yet another item</li>
</ol>
```

- Per cambiare il tipo di numerazione
- X. Item
 - IX. Next item
 - VIII. Yet another item

Lists

```
<ul>
  <li>item 1</li>
  <li>item 2
    <ul>
      <li>sub-item 2.1</li>
      <li>sub-item 2.2</li>
    </ul>
  </li>
  <li>item 3</li>
</ul>
```

- item 1
- item 2
 - sub-item 2.1
 - sub-item 2.2
- item 3

Lists

description list <dl> <dt>

```
<dl>
  <dt>hypertext</dt>
  <dd>non-linear text material</dd>
  <dt>hypermedia</dt>
  <dd>media in a non-linearly organized system</dd>
</dl>
```

hypertext

non-linear text material

hypermedia

media in a non-linearly organized system

Comments

<!-- This is a comment -->

<!--

This paragraph

also

is a comment ...

-->

Vogliamo che ci sia interazione con l'utente

Forms

A fragment of an HTML document that contains controls for entering data

- a mechanism that allows the user to transfer data to the web application
- text, radio buttons, checkboxes, ...

The basic method of interaction between the user and the web application

Allows entering data via the user interface (as in desktop applications)

Forms

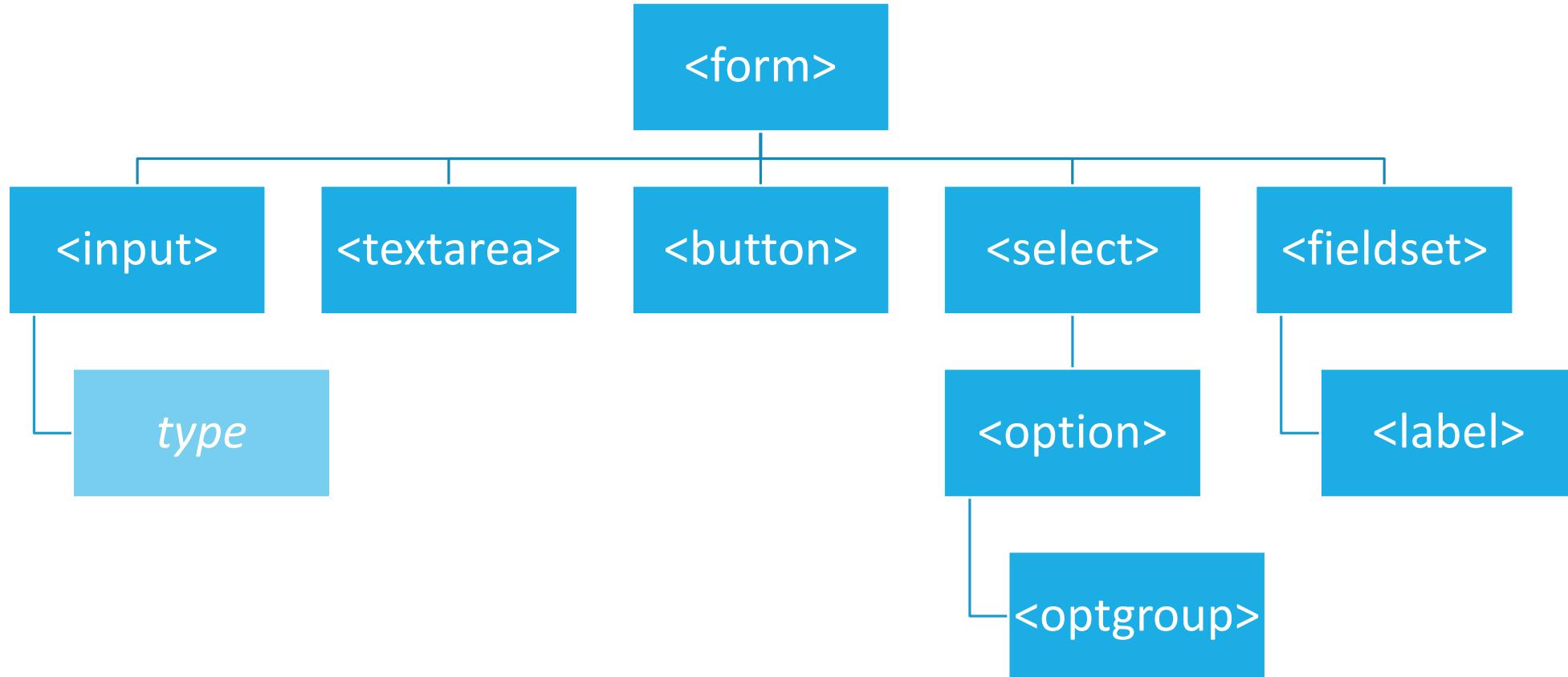
<form> provides a mechanism to interact with the user

- attribute *action* : **where** the data is to be sent (URL)
- attribute *method* : **how** the data is to be sent
 - post
 - get (default)
- attribute *name*

<form action="action.php" method="get">

<form action="action.php" method="post">

Forms



Text box

```
<!DOCTYPE html>
<html>
<body>
  <h2>Registration form</h2>
  <form action="/action.php">
    Name:
    <input type="text" name="name" value="Please, enter your name">
  </form>
</body>
</html>
```

Registration form

Name:

Password

```
<!DOCTYPE html>
<html>
<body>
  <h2>Registration form</h2>
  <form action="/action.php">
    Name:
    <input type="text" name="name" value="Please, enter your name"><br/>
    <label for="pswd">Password:</label>
    <input type="password" name="gender" id="pswd" ><br/>
  </form>
</body>
</html>
```

Label... non hai capito
for e id should be the same

Registration form

Name: Please, enter your name

Password:

Radio button

```
<!DOCTYPE html>
<html>
<body>
<h2>Registration form</h2>
<form action="/action.php">
  Name:
  <input type="text" name="name" value="Please, enter your name"><br/>
  <label for="pswd">Password:</label>
  <input type="password" name="gender" id="pswd" ><br/>
  Gender:
  <label for="male">Male</label>
  <input type="radio" name="gender" id="male" value="male" checked>
  <label for="female">Female</label>
  <input type="radio" name="gender" id="female" value="female">
</form>
</body>
</html>
```

Registration form

Name: Please, enter your name

Password:

Gender: Male Female

Checkbox

```
<label for="male">Male</label>
<input type="radio" name="gender" id="male" value="male" checked>
<label for="female">Female</label>
<input type="radio" name="gender" id="female" value="female"><br/>
Programming languages:
<input type="checkbox" name="c" value="c" checked>C++
<input type="checkbox" name="java" value="Java">Java
</form>
</body>
</html>
```

Registration form

Name: Please, enter your name

Password:

Gender: Male Female

Programming languages: C++ Java

Select list

```
Programming languages:  
<input type="checkbox" name="c" value="c" checked>C++  
<input type="checkbox" name="java" value="Java">Java <br/>  
Semester  
<select>  
  <option value="1">1</option>  
  <option value="2">2</option>  
  <option value="3">3</option>  
</select>  
</form>  
</body>  
</html>
```

```
<select name="colorlist" id="colorlist">  
  <option value="0">Choose a color</option>  
  <option value="red">Red</option>  
  <option value="blue">Blue</option>  
  <option value="green">Green</option>  
  <option disabled value="orange">Orange</option>  
</select>
```

Registration form

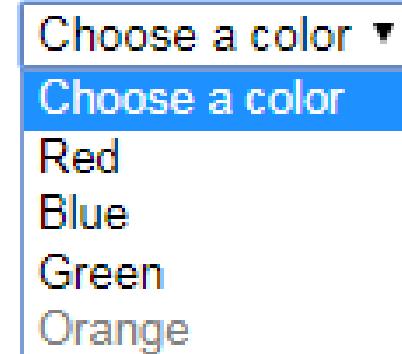
Name:

Password:

Gender: Male Female

Programming languages: C++ Java

Semester



Text area

```
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
</select><br/><br/>
<textarea rows="7" cols="70">
    please, write your comment
</textarea>
</form>
</body>
</html>
```

Registration form

Name:

Password:

Gender: Male Female

Programming languages: C++ Java

Semester

please, write your comment

File field

```
please, write your comment  
</textarea><br/>  
attach file:  
  
</form>  
</body>  
</html>
```

Registration form

Name: Please, enter your name
Password:
Gender: Male Female
Programming languages: C++ Java
Semester

please, write your comment

attach file: Nie wybrano pliku

Submit and Reset buttons

```
    ...
</textarea><br/>
attach file:
<input type="file" name="file"><br/>
<input type="submit" value="Submit">
<input type="reset">
</form>
</body>
</html>
```

Registration form

Name:

Password:

Gender: Male Female

Programming languages: C++ Java

Semester

please, write your comment

attach file: Nie wybrano pliku

Forms

form layout

- <div>, <p>
- <fieldset>, <legend>
- <label>

```
<form method="post" action="b.txt">
  <fieldset>
    <legend>Personal data:</legend>
    <label for="name"> Name and surname: </label>
    <input type="text" id="name"><br>
    email: <input type="email"><br>
    <input type="submit" value="Send" />
    <button type="button">Confirm</button>
  </fieldset>
</form>
```

Personal data:

Name and surname:

email:

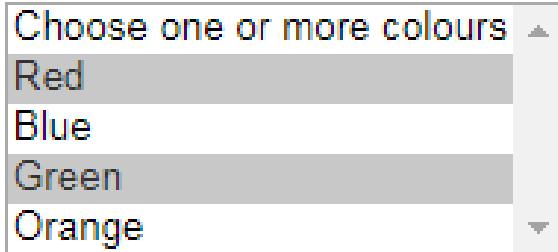
Attributes of form tags

| name | value | readonly (yes no) | checked (yes no): |
|--|---|--|---|
| <ul style="list-style-type: none">• name of the form's element | <ul style="list-style-type: none">• default value of the field (text box, password)• label of the button (buttons),• the value of the element sent to application (checkbox, radio buttons) | <ul style="list-style-type: none">• specifies that the input field cannot be changed | <ul style="list-style-type: none">• specifies that the element is checked |

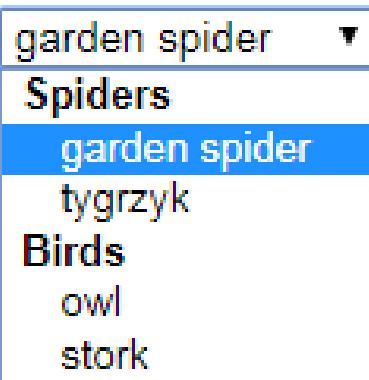
Attributes of form tags

Enter your surname

```
<input ... value="Enter your surname"/>
```



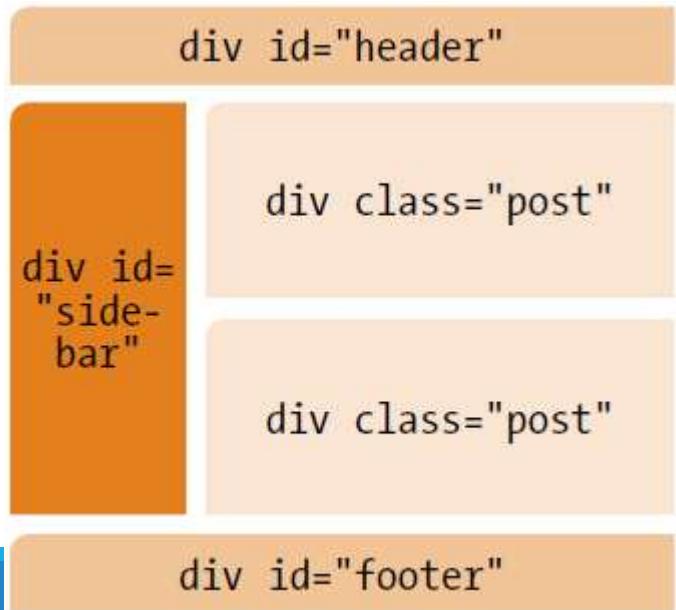
```
<select ... multiple="yes" size="4">
```



```
<select>
<optgroup label="spiders">
  <option value="krzyżak">garden spider</option>
  <option value="tygrzyk">tygrzyk</option>
</optgroup>
```

Page division

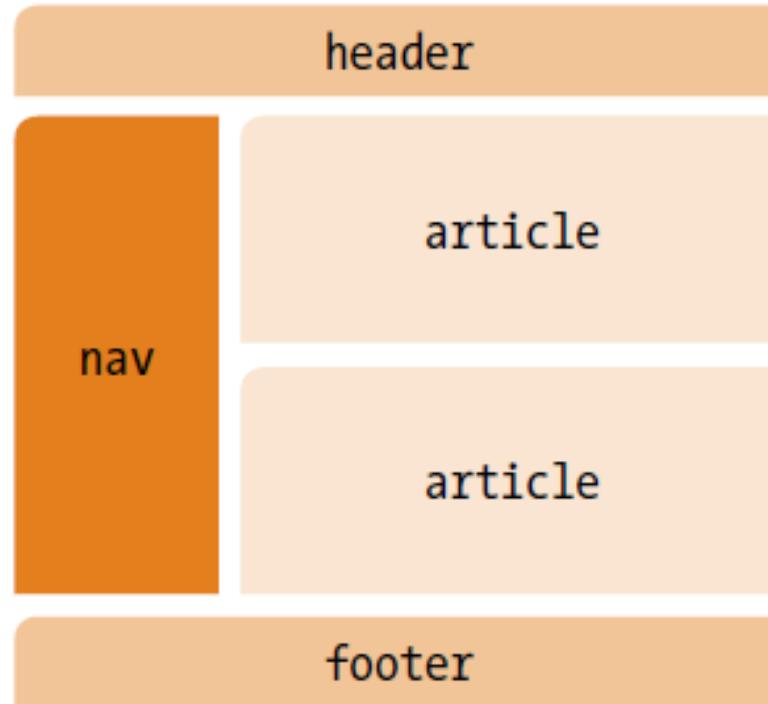
- <div>
- defines a section in a HTML document
- used to group block-elements to format them with styles



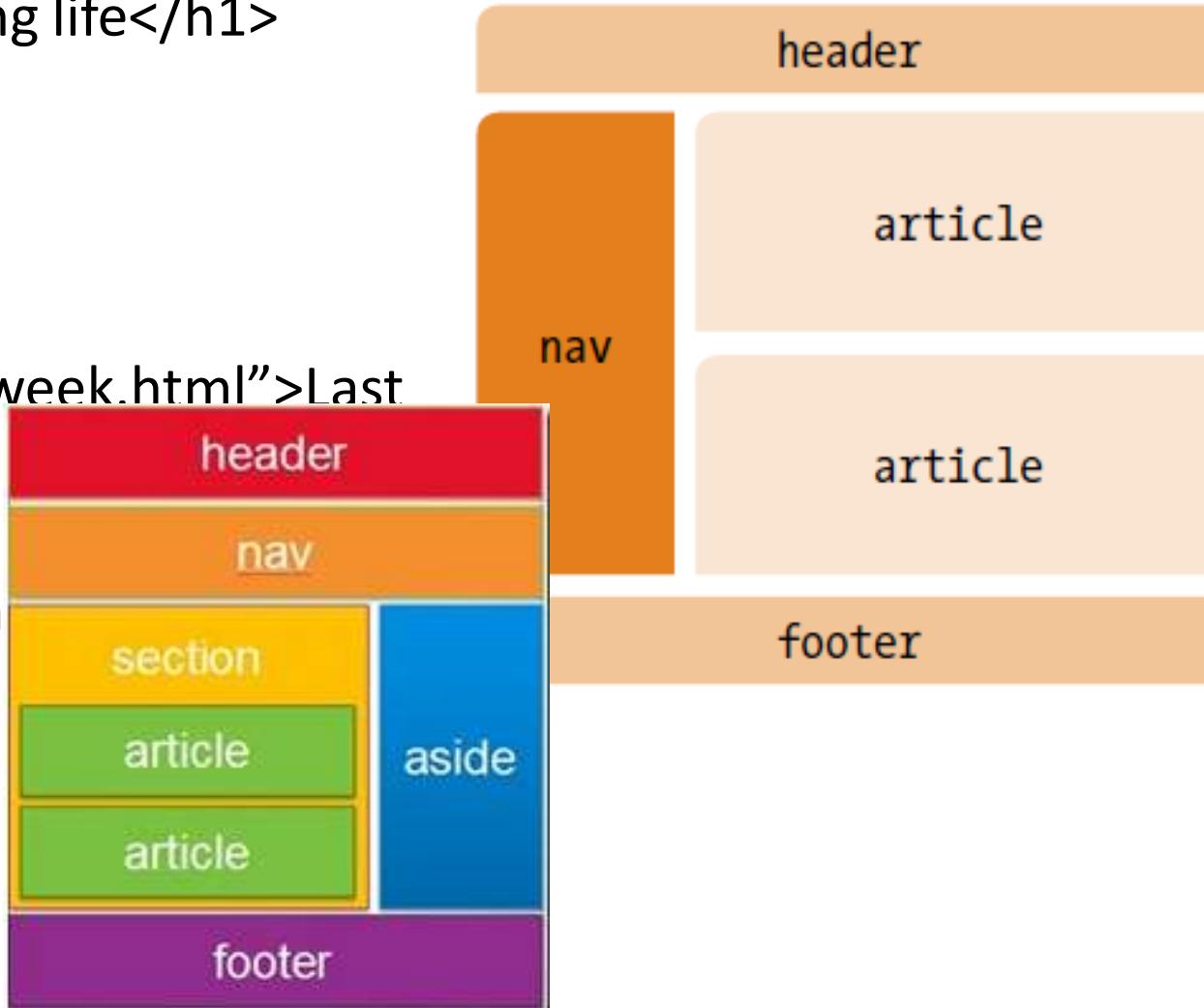
```
<div id="header">
  <h1>My interesting life</h1>
</div>
<div id="sidebar">
  <h2>Menu</h2>
  <ul>
    <li><a href="last-week.html">Last week</a></li>
    <li><a href="archive.html">Archives</a></li>
  </ul>
</div>
```

```
<div id= "header">  
  <h1>My interesting life</h1>  
  </div>  
  <div id= "nav">  
    <h2>Menu</h2>  
    <ul>  
      <li><a href="last-week.html">Last  
        week</a></li>  
      <li><a  
        href="archive.html">Archives</a><  
        /li>  
    </ul>  
  </div>
```

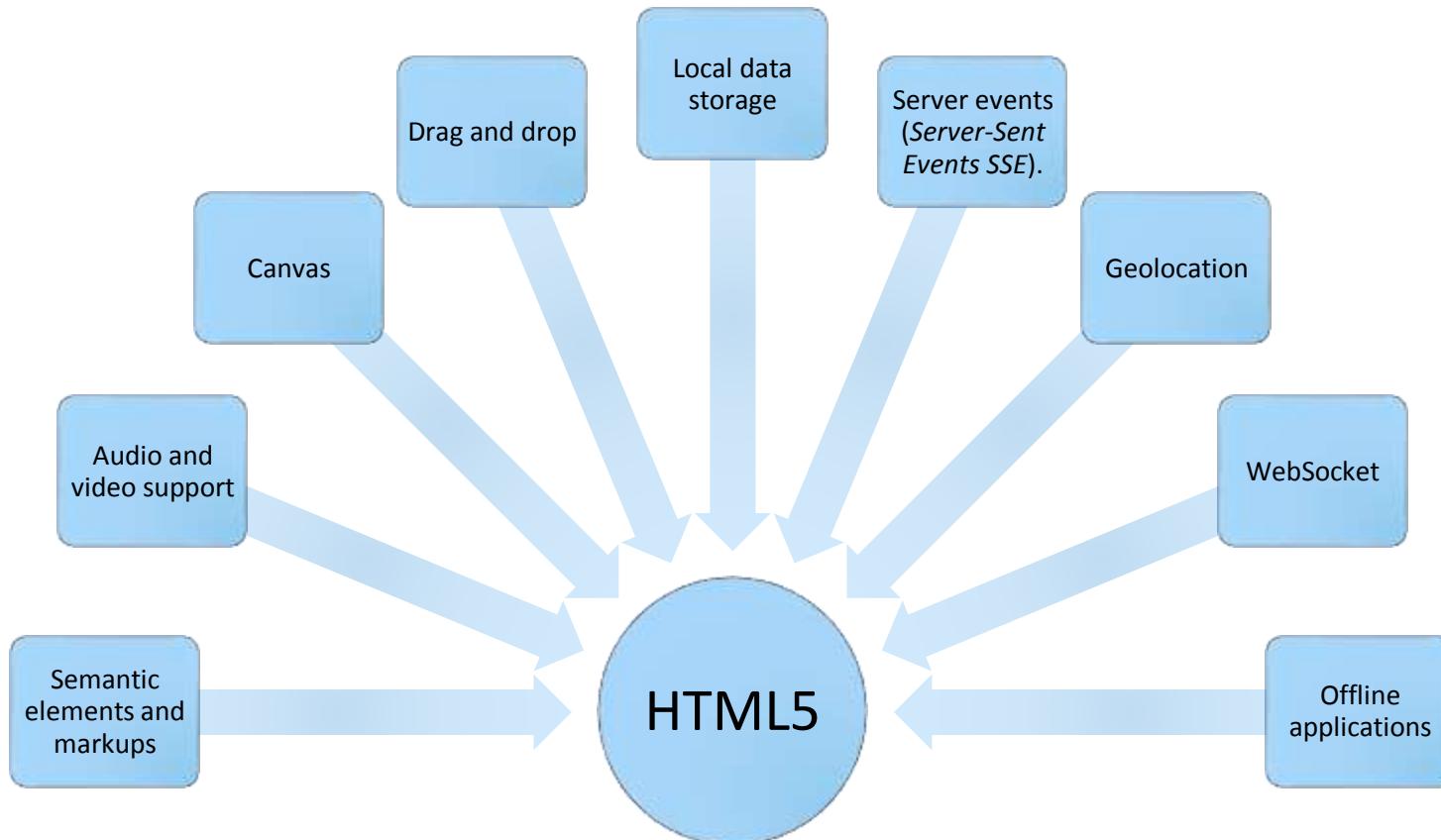
div id=" header"



```
<header>
<h1>My interesting life</h1>
</header>
<nav>
<h2>Menu</h2>
<ul>
<li><a href="last-week.html">Last
week</a></li>
<li><a
href="archive.htm
/>/li>
</ul>
</nav>
```



HTML5: new features



Canvas



Possibility of drawing

- The <canvas> tag is only a container for graphics
- A canvas is a rectangular area of a raster graphics
- JavaScript is needed for drawing on canvas

```
<canvas id="my-first-canvas" width="360" height="240">
```

Here we give the text displayed when the browser does not support canvas.

```
</canvas>
```

```
<script>
```

```
var c = document.getElementById('my-first-canvas');
var ctxt = c.getContext('2d');
var grd=ctxt.createLinearGradient(0,0,175,50);
grd.addColorStop(0,"#FF0000");grd.addColorStop(1,"#0000FF");
ctxt.fillStyle=grd; ctxt.fillRect(0,0,175,50);
</script>
```

Canvas

- **Functionality**

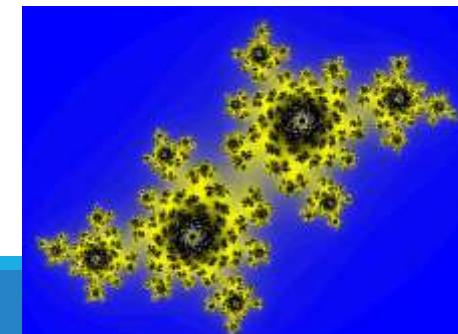
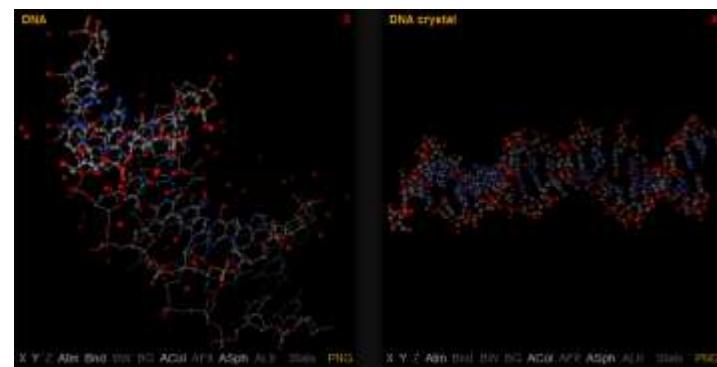
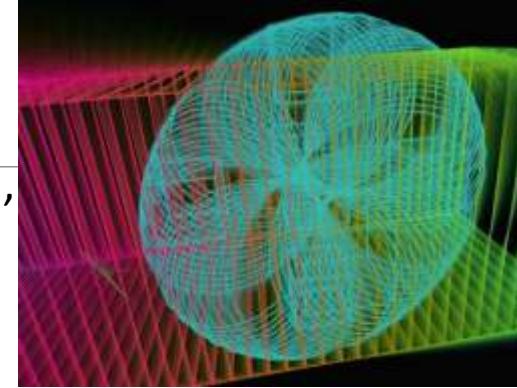
- drawing rectangles and paths (lines, bows, curves Beziera, ...),
- filling figures with a color, pattern or gradient,
- drawing subtitles,
- support for transparency,
- transformations (shifting, scaling, skewing etc.),
- embedding raster images (PNG, JPEG, GIF),
- shading,
- ...

- **Usage**

- graphs and charts,
- games,
- ...

- **SVG**

- the ability to put code in HTML



<video>

The controls attribute adds video controls, like play, pause, and volume.

<source>

- The browser will use the first recognized format



```
<video controls width="360" height="240" poster="placeholder.jpg">
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.webm" type="video/webm">
  <source src="movie.ogv" type="video/ogg">
  <object type="application/x-shockwave-flash" width="360"
height="240" data="player.swf?file=movie.mp4">
    <param name="movie" value="player.swf?file=movie.mp4">
    <a href="movie.mp4">Download the movie</a>
  </object>
Przeglądarka nie obsługuje znacznika video
</video>
```

<audio>

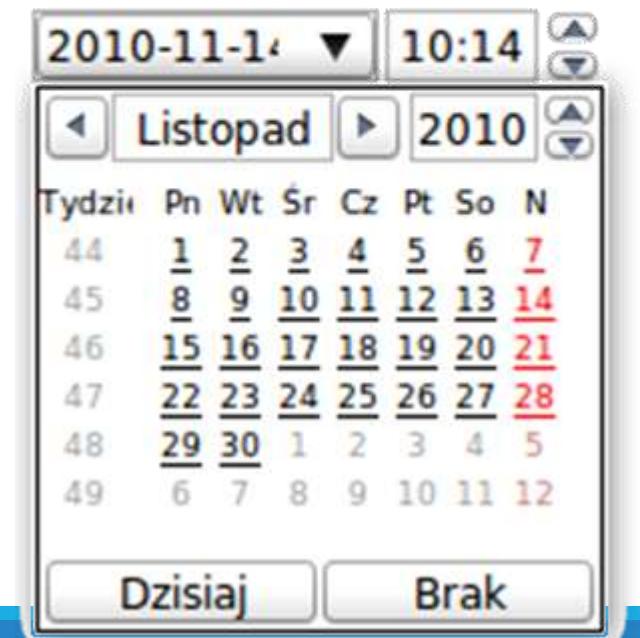
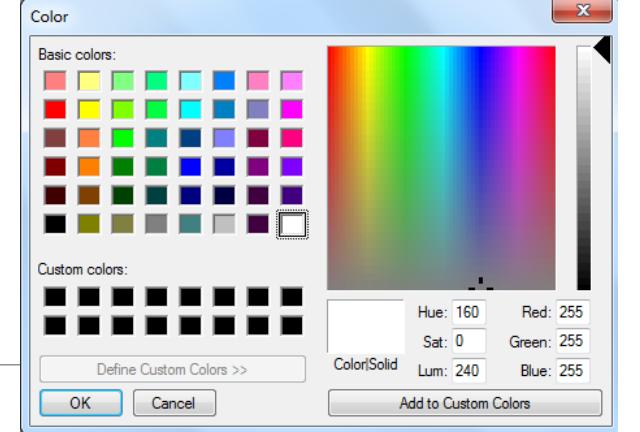
similar to the tag <video>



```
<audio controls>
  <source src="pliczek.ogg" type="audio/ogg">
  <source src="pliczek.mp3" type="audio/mpeg">
    <object type="application/x-shockwave-flash"
data="player.swf?soundFile=pliczek.mp3">
      <param name="movie" value="player.swf?soundFile=pliczek.mp3">
      <a href="pliczek.mp3">Download the song</a>
    </object>
</audio>
```

New controls

```
<input name="url" type="url">  
  
<input name="number" type="number" min="1" max="5">  
  
<input name="number" type="range" min="1" max="5">  
  
<input name="date" type="date">
```



Geolocation API

- The HTML Geolocation API is used to locate a user's position.
- User permission is required
- Location
 - based on the client's IP address
 - based on the near BTS signal strength measurement
 - based on built-in GPS
- Available
 - geographic coordinates, user movement speed, direction of movement ...

Hypertekst & hypermedia

DR WIOLETA SZWOCH

DEPARTMENT OF INTELLIGENT INTERACTIVE SYSTEMS

Tworzenie stron

HTML

content

CSS

presentation

JavaScript

action

CSS Cascading Style Sheet

Mechanism, language for defining the style of presentation of documents on the Internet, defining their graphic layout

The preferred way to describe the presentation of HTML documents, ...

It allows for consistent formatting of a set of documents

Easier modification of the pages' appearance

It does not exist independently - closely related to the language of the document's structure description

Why CSS?

separation of the design part from the content part.

control over the method of the final presentation of the document

shorter time for the "care" of the page by the author

different formatting for different devices

Adding styles to documents

External style sheets

Internal CSS

Inline styles

Adding styles to documents

metodo
raccomandato

External
style sheets

```
<html>
  <head>
    <title>My hobby</title>
    <link rel="Stylesheet" type="text/css" href="Style.css">
  </head>

  <body>
    body
    {
      margin-top : 5px;
      padding : 0;
      text-align : left;
      background : #333333;
      border-radius: 20px ;
    }
    h1 {
      text-align:left;
      vertical-align:middle;
      font-family: 'Palatino Linotype', "Times New Roman", Verdana;
      color: #555555;
      font-style: italic;
      font-weight: bold;
    }
    ul {
      margin: 0;
      padding: 0;
      list-style: none;
      width: 200px;
      border-bottom: 1px solid #565656;
    }
    ul li {
      position: relative;
    }
```

Adding styles to documents

Internal CSS

```
<!DOCTYPE html>
<html>
<head>
<title>My hobby</title>
<style>
body
{
    margin-top : 5px;
    padding : 0;
    text-align : left;
    background : #333333;
    border-radius: 20px ;
}
h1 {
    text-align:left;
    vertical-align:middle;
    font-family: 'Palatino Linotype', "Times New Roman", Verdana;
    color: #555555;
    font-style: italic;
    font-weight: bold;
}
ul {
    margin: 0;
    padding: 0;
    list-style: none;
    width: 200px;
    border-bottom: 1px solid #565656;
}
```

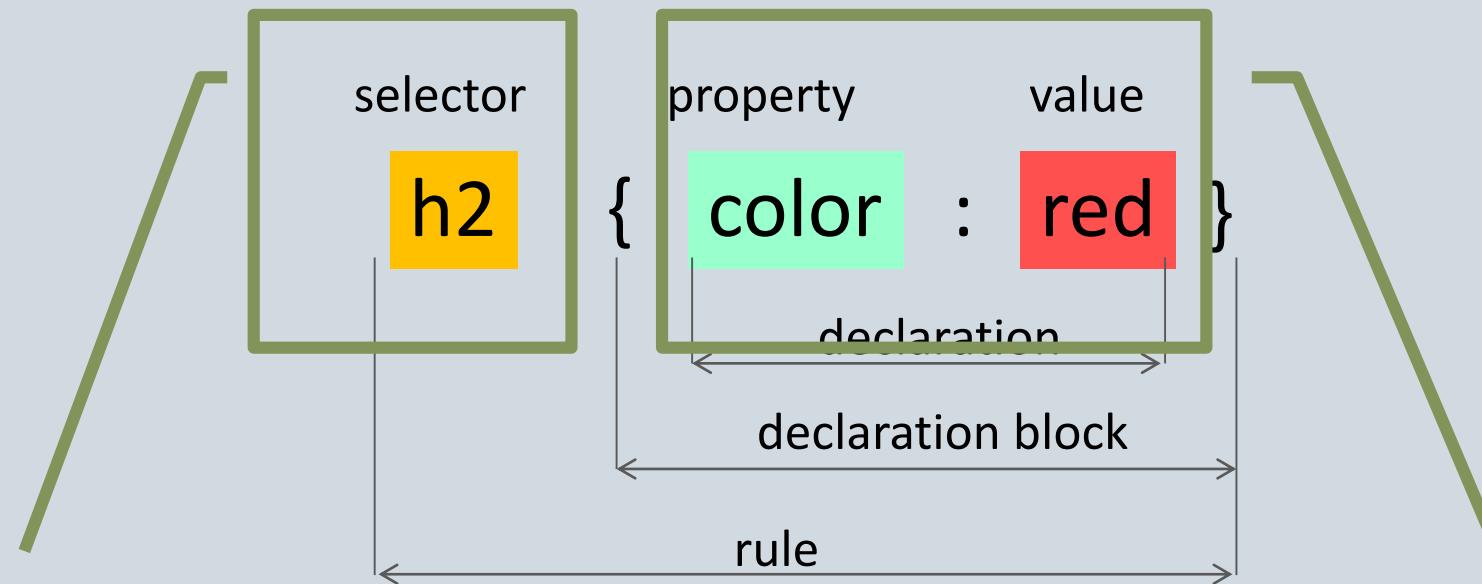
Dai non usare MAI (ma quello con la priorità più alta)

Adding styles to documents

Inline styles

```
<body>
    <div style="text-align:center; border: 1px solid black;">
        An example of inline style
    </div>
```

Rules syntax



indicates elements of the document,
to be formatted

describes the formatting
(formatting commands)

Selector:

name of the elements (h1, p, div, a, ...)
class (.left)
identifier (#left)

Rules syntax

grouping of selectors

```
p, h1 { color : red }
```

grouping of declarations

```
p { color : red ; font-style : italic ; }
```

Types of selectors

simple selector

```
<html>
  <head>
    <title>My hobby</title>
    <style>
      p { color:red;      font-size:14pt; }
    </style>
  </head>
<body>
  HTML file wits CSS rules
  <p>
    An example of a formatted paragraph.
  </p>
</body>
</html>
```

HTML file wits CSS rules

An example of a formatted paragraph.

Types of selectors

universal selector

```
* { color:red; font-size:14pt; }
```

The universal selector, written "*", matches the name of any element type

- Use “**Universal selector**” if you want all the tags in your web documents have some common style (for example, all tags don’t have any margin)

Leggi il testo! è autoesplicativo: span funziona solo all'interno di p! (come se fosse un figlio)

Types of selectors

Descendants

```
p span { color:red; font-size:14pt; }
```

```
<p> paragraph <span> with descendants selector </span> </p>
<div> here the rule for span <span> doesn't work </span> </div>
```

paragraph with descendants selector

here the rule for span doesn't work

Types of selectors

Child

```
p > span { color:red; font-size:14pt; }
```

```
<p> paragraph <span> with descendants selector </span> </p>
<div> here the rule for span <span> doesn't work </span> </div>
```

paragraph with descendants selector

here the rule for span doesn't work

Types of selectors

sibling

```
h2 + p { color:red; font-size:14pt }
```

```
<h2>Story</h2>
<p>First paragraph</p>
<p>Second paragraph</p>
<p>Third paragraph</p>
```

Story

First paragraph

Second paragraph

Third paragraph

Se devi usarlo più di una volta

Types of selectors

Class

```
.red { color:red; font-size:14pt }  
  
h1.blue { color:blue }
```

- Use “**class selector**” if you want to apply the style for many (but not all) occurrences of a certain tag; *OR* if you want to apply the style for more than one type of tags

```
<h1 class="blue">Header</h1>  
<h2 class="red">Story</h2>  
<p>First paragraph</p>  
<p class="red">Second paragraph</p>  
<p>Third paragraph</p>
```

Header

Story

First paragraph

Second paragraph

Third paragraph

Types of selectors

ID

```
#red { color:red; font-size:14pt }

h1#blue { color:blue }
```

- Use “**ID selector**” if you want to apply the style for only one occurrence of a certain tag

```
<h1 id="blue">Header</h1>
<h2>Story</h2>
<p>First paragraph</p>
<p id="red">Second paragraph</p>
<p>Third paragraph</p>
```

Header

Story

First paragraph

Second paragraph

Third paragraph

Types of selectors

pseudo-classes

```
a {  
    font-size:12pt;  
    text-decoration:none;  
}  
  
a:link  
{  
    color:green;  
}  
a:visited  
{  
    color:yellow;  
}  
a:hover  
{  
    color:pink;  
    text-decoration:underline;  
}
```

[Wydział ETI PG](https://www.eti.pg.gda.pl)

Wydział ETI PG

Wydział ETI PG

Types of selectors

pseudo-element

```
p::first-letter { color:red; font-size:24pt }
```

```
<h2>Story</h2>
<p>First paragraph</p>
<p>Second paragraph</p>
<p>Third paragraph</p>
```

Story

F irst paragraph

S econd paragraph

T hird paragraph

Types of selectors

selektory atrybutu

```
<style>
  input[type="submit"] {color:red;}
</style>
```

Dane osobowe:

Nazwisko i imię:

email:

Send **Potwierd**

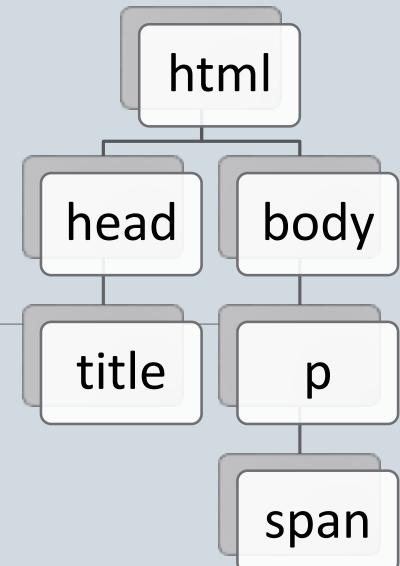
Inheritance

```
p {color:green}
```

Story

paragraph with span

```
<p> paragraph <span> with span </span> </p>
```



Text properties

- text-indent, text-align, text-decoration, text-shadow, letter-spacing, text-transform
- word-spacing
- letter-spacing
- text-decoration (none, underline, overline, line-through, blink)
- vertical-align
- text-transform (capitalize, uppercase, lowercase)
- text-align
- text-indent
- line-height

Font properties

font (serif, ...)

font-style (normal, italic, ...)

font-variant (normal, small-caps, ...)

font-weight (normal, bold, 100, ...)

font-size (small, medium, large, ...)

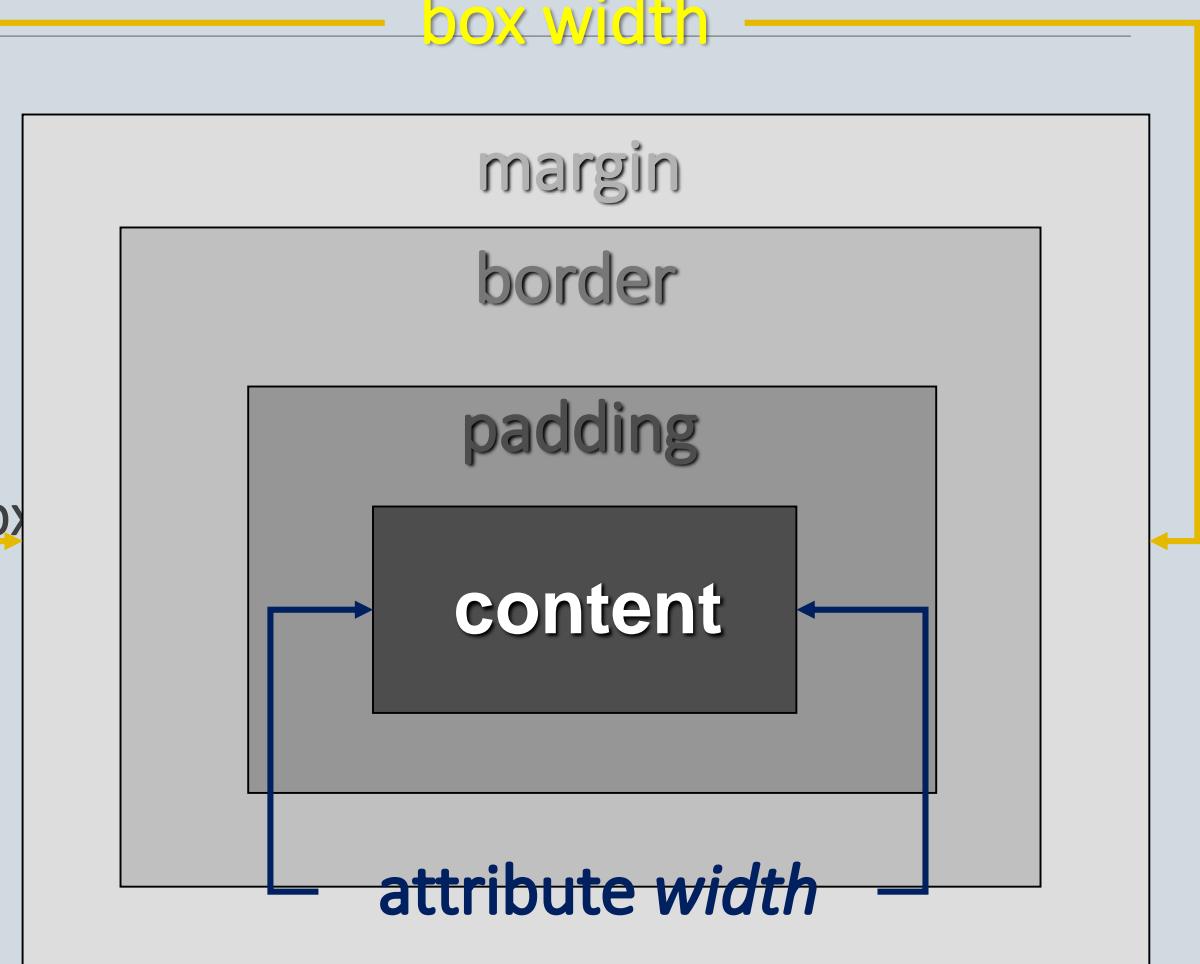
font-family

Color properties

- color
- background
- background-color
- background-image
- background-repeat
- background-attachment
- background-position

The box model

padding : 15px;
margin : 30px;
border-left : 1px;
margin-top : 100px



Positioning

static

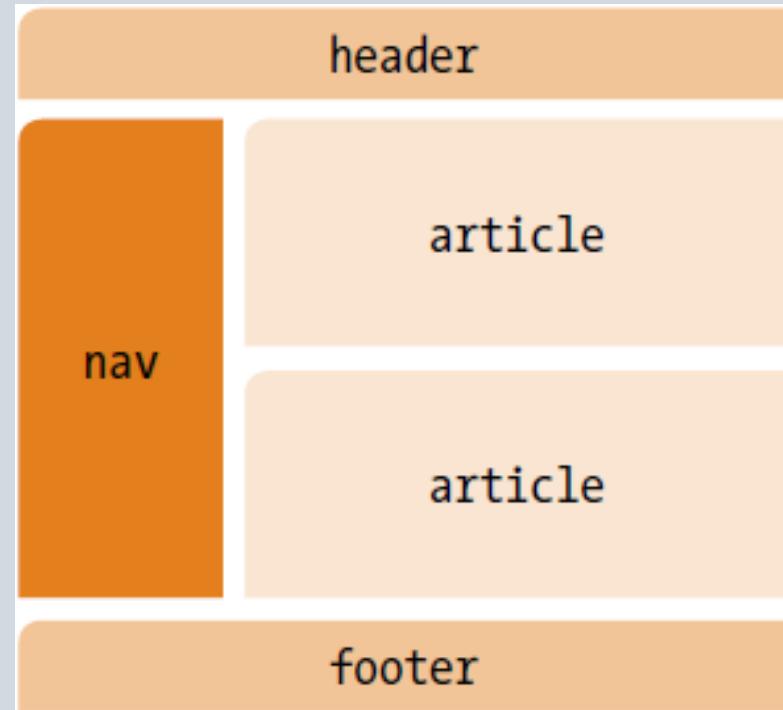
absolute

relative

fixed

z-index

float

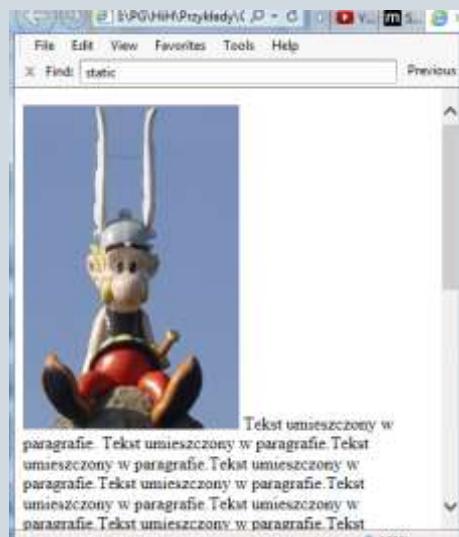


Positioning

static

- normal positioning

```
selector { position: static; other parameters }
```



```
img  
{  
position: static;  
left:70px;top:50px  
}
```

E' invisibile agli altri elementi

Positioning

absolute



- position of the element is determined by specifying the distance from the edges of the container (*left, right, top, bottom*).

selector { position: absolute; other parameters }



```
img
{
  position: absolute;
  left:70px;top:50px
}
```

Gli altri elementi sanno dell'elemento

Positioning

relative

- the position of the element is moved by the values specified in left, right, top and bottom relative to the initial position.

selector { position: relative; other parameters }



```
img  
{  
  position: relative;  
  left:70px;top:50px  
}
```

Positioning

fixed

- an element locked in a specific place on the screen, constantly visible and does not scroll with the rest of the page.

selector { position: fixed; other parameters }

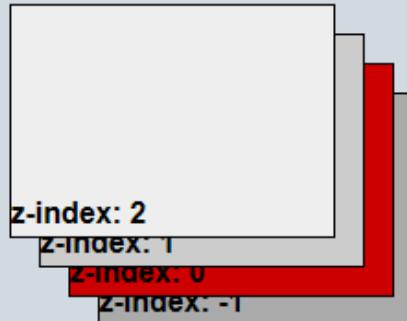


```
img  
{  
    position: fixed;  
    left:70px;top:50px  
}
```

Positioning

overlaping (z-index)

- defines the overlapping of elements



selector { position:type; other parameters; z-index: number }



```
img
{
    position:fixed;
    left:70px;top:50px
    z-index:-1;
}
```

Positioning

float

- setting a given element relative to elements that are adjacent to it



```
selector { float: right/left/none; }
```



```
img  
{  
  float: right;  
  left:70px;top:50px  
}
```

Transformations

translation

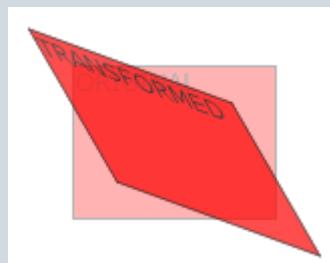
rotation

scale

skew

matrix

```
div {  
    -ms-transform: rotate(30deg); /* IE 9 */  
    -webkit-transform: rotate(30deg); /* Chrome, Safari, Opera */  
    transform: rotate(30deg);  
}
```



```
div {  
    -ms-transform: skew(30deg,20deg); /* IE 9 */  
    -webkit-transform: skew(30deg,20deg); /* Chrome, Safari, Opera */  
    transform: skew(30deg,20deg);  
}
```

```
div {  
    -ms-transform: translate(50px,100px); /* IE 9 */  
    -webkit-transform: translate(50px,100px); /* Chrome, Safari, Opera */  
    transform: translate(50px,100px);  
}
```



Transitions

effects that let an element gradually change from one style to another

you must specify

- property
- duration



```
<style>
div {
    width: 120px;
    height: 50px;
    background: blue;
    /* For Safari 3.1 to 6.0 */
    -webkit-transition-property: width;
    -webkit-transition-duration: 1s;
    -webkit-transition-timing-function: linear;
    -webkit-transition-delay: 2s;
    /* Standard syntax */
    transition-property: width;
    transition-duration: 1s;
    transition-timing-function: linear;
    transition-delay: 1s;
}
div:hover {
    width: 200px;
}
</style>
</head>
<body>
<p>  </p>
<div>  Tekst</div>
<p></p>
</body>
```

Animations

the @keyframes rules specify the animation

the need to link the rule with the selector

- animation name
- duration



```
<style>
div {
    width: 100px;
    height: 100px;
    background: red;
    position: relative;
    /* Chrome, Safari, Opera */
    /* Standard syntax */
    animation-name: myfirst;
    animation-duration: 5s;
    animation-timing-function: linear;
    animation-delay: 2s;
    animation-iteration-count: infinite;
    animation-direction: alternate;
    animation-play-state: running;
}

/* Chrome, Safari, Opera */
/* Standard syntax */
@keyframes myfirst {
    0% {background:red; left:0px; top:0px;}
    25% {background:yellow; left:200px; top:0px;}
    50% {background:blue; left:200px; top:200px;}
    75% {background:green; left:0px; top:200px;}
    100% {background:red; left:0px; top:0px;}
}
</style>
</head>
<body>
<div></div>
</body>
```

XML

Extensible Markup Language

DR WIOLETA SZWOCH

DEPARTMENT OF INTELLIGENT INTERACTIVE SYSTEMS

Agenda

XML fundamentals

XML document

Markup Language

GML

- 1969 E.Mosher, R.Lorie, Ch. F. Goldfarb
- first markup language

Standard General Markup Language (SGML)

- ISO standard of exchange and storage of the data (1986)
- separating form (how it looks) from content (what it is)
- tags do not determine style of presentation
- set of tags is defined in separate declaration - DTD



Are languages to create XML documents and are instances of XML meta languages

HTML

vs

XML

web description
language

the meaning of tags
and attributes is
predetermined

instance of SGML

meta-language for
defining languages

the meaning of tags
and attributes is
determined by the
user or application

subset of SGML

XML - eXtensible Markup Language

Extensible – tags used in documents are designed by the user (document schema designer)

Language – the document is built according to syntax rules, and based on a defined alphabet - tags

Markup – the document is built on tags

Non sappiamo con interpretar i numeri: l'HTML non interpreta i numeri

Why XML?

Problems with HTML

- the set of tags is strictly (rigorosamente) defined
- is used to describe the way of presentation not data
- what does the code mean?

<td> 3472096</td>

<p>Wioleta Szwoch

HTML code

 Department of Intelligent
Interactive Systems
 3472096

Wioleta Szwoch

Department of Intelligent Interactive Systems
3472096

processing by the browser

Is the algorithm able to interpret this code?

```
< employee>
<name> Wioleta Szwoch</name>
<department>
    Department of Intelligent Interactive Systems
</ department >
<phone>3472096</phone>
</ employee>
```

code processing by the browser?

Wioleta Szwoch

Department of Intelligent Interactive Systems

phone: 347 20 96

Wioleta Szwoch

Department of Intelligent Interactive Systems

phone: 347-20-96

Wioleta Szwoch

Department of Intelligent Interactive Systems

3472096

XML is not ...

A replacement for HTML

- but HTML can be generated from XML

A presentation format

- but XML can be converted into one

A programming language

- but it can be used with almost any language

A network transfer protocol

- but XML may be transferred over a network

A database

- but XML may be stored into a database

But then – what is it?

XML is a meta markup language
for text documents / textual data



XML allows to define languages
(„applications“) to represent text
documents / textual data

```
< employee>
  <name> Wioleta Szwoch</name>
  <department>
    Department of Intelligent Interactive Systems
  </ department >
  <phone> 3472096 </phone>
</ employee>
```

- Easy to understand for human users
- Very expressive (semantics along with the data)
- Well structured, easy to read and write from programs

< a1>

<b20> Wioleta Szwoch</b20>

<cX>

Department of Intelligent Interactive Systems

</ cX>

<tX> 3472096 </tX>

</ a1>

- Hard to understand for human users
- Not expressive (semantics along with the data)
- Well structured, easy to read and write from programs

```
< data>
```

```
12n23ng3h4j5ats7899ducyytt6n33n3hdgsgc237675
```

```
</ data>
```

- **Impossible** to understand for human users
- **Not** expressive (**no** semantics along with the data)
- **Unstructured**, read and write only with special programs

XML Document

The XML document can optionally have an XML declaration.

- XML version
- the character encoding used in the document
 - UTF-8; UTF-16; ISO-8859-1; ISO-8859-2; windows-1250
- processing external DTD

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
```

```
<person>
  Some information about
    <FirstName>Anna</FirstName>
    <LastName>Kowalska</LastName>
</person>
```

XML documents

What's in an XML document?

elements

attributes

declaration

entity

processing instructions

XML declaration (FACOLTATIVA)



```
<?xml version="1.0" encoding="UTF-8"?>
<books>
    <book category="fantasy">
        <title>The colour of magic</title>
        <author>Terry Pratchett</author>
        <year>2012</year>
        <price>30.00</price>
    </book>
    <book category="fantasy">
        <title>Guards! Guards!</title>
        <author>Terry Pratchett</author>
        <year>2012</year>
        <price>30.00</price>
    </book>
</books>
```

type version encoding processing
external DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<books>
    <book category="fantasy">
        <title>The colour of magic</title>
        <author>Terry Pratchett</author>
        <year>2012</year>
        <price>30.00</price>
    </book>
    <book category="fantasy">
        <title>Guards! Guards!</title>
        <author>Terry Pratchett</author>
        <year>2012</year>
        <price>30.00</price>
    </book>
</books>
```

root (parent of the other elements):
tutto è contenuto all'interno

element
(contenitore di informazioni)

```
<?xml version="1.0" encoding="UTF-8"?>
<books>
    <book category="fantasy">
        <title>The colour of magic</title>
        <author>Terry Pratchett</author>
        <year>2012</year>
        <price>30.00</price>
    </book>
    <book category="fantasy">
        <title>Guards! Guards!</title>
        <author>Terry Pratchett</author>
        <year>2012</year>
        <price>30.00</price>
    </book>
</books>
```

attribute: part of element that contains
more information about the element

element content

attribute content

XML - element

Basic concept - defines the meaning of the content of information

```
<name> Kowalski Jan </name>
```

Element names are case-sensitive

```
<name> Kowalski Jan </Name >
```

The tag name can contain:

- letters, numbers
- - . (not the beginning)
- _ :

```
_name.i::1-2
```

```
<NAME></NAME>  
<Name></NAmE>  
<name ></ name >
```

XML - elements

An element can contain:

- text
- attributes
- other elements
- or a mix of the above

Empty element

```
<person>  
    Definition for a person  
    <name>Anna</name>  
    <surname>Kowalska</surname>  
    with mixed value  
</person>
```

```
<link href='http://www.wp.pl'> </link>
```

empty element tag

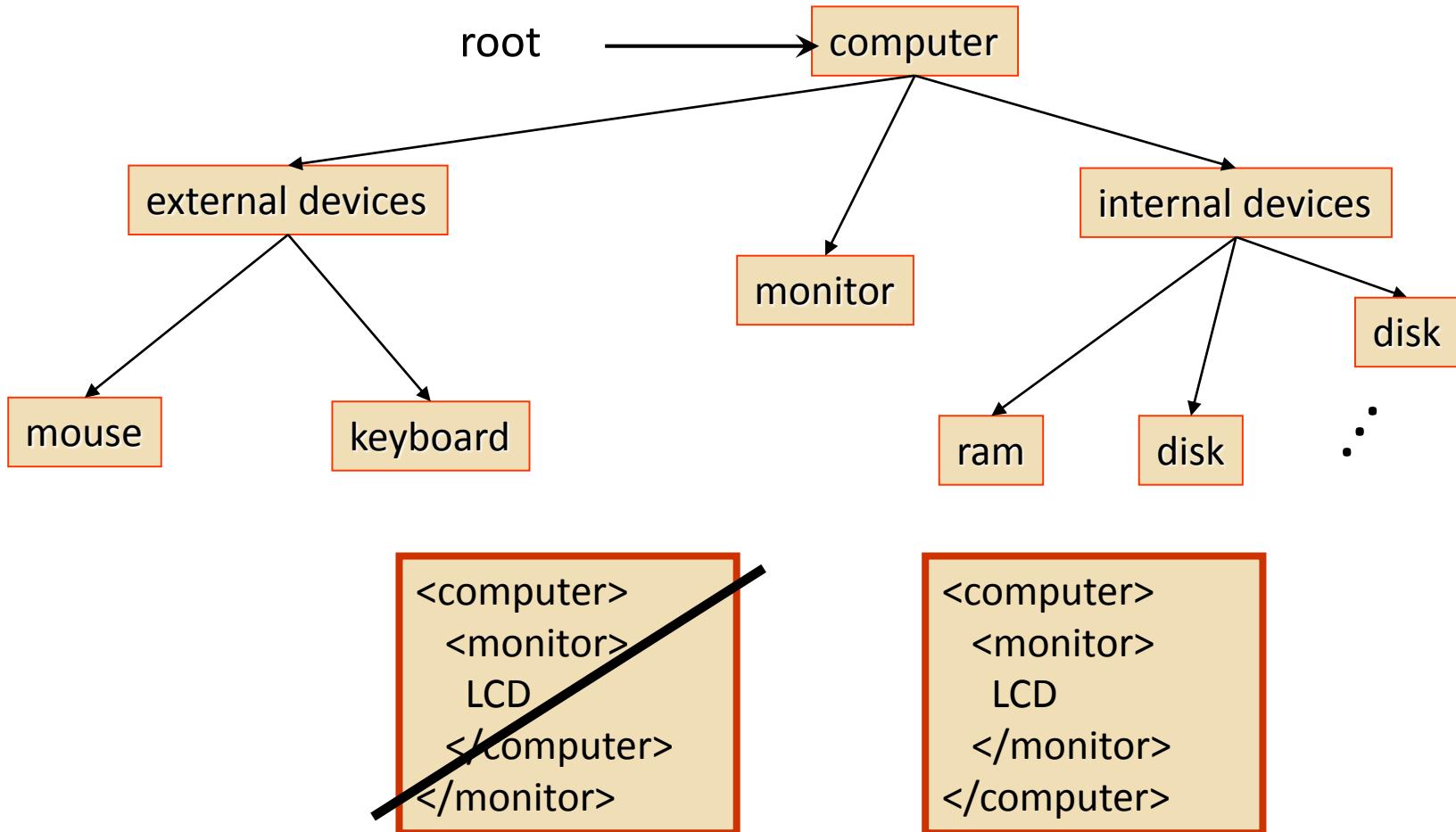
Sono corrette entrambe le versioni

```
<link href='http://www.wp.pl' />
```

attribute

Solo gerarchia, nulla da segnalare

XML – elements hierarchy



XML - attribute

any order of attributes

qualsiasi ordine di attributi

In one element **only different attribute names**

In un elemento solo nomi di attributi diversi

limitations on the name of the attribute as on the element

limitazioni sul nome dell'attributo come sull'elemento

attributes have no structure, **simply strings** (while elements can have subelements)

gli attributi non hanno struttura, semplicemente stringhe (mentre gli elementi possono avere sottoelementi)

```
<surname maiden_name ="no"> Kowalski </surname>
```

~~```
<name which="first" which="second"> Jan </name>
```~~

# XML - declarations

---

```
<!
<message>
<!
 Bad nesting <comp> <mon> LCD </comp> </mon>
</message>
[
```

```
<!-- an example of comment and CDATA section -->
<message >
 <![CDATA[Bad nesting <comp> <mon> LCD </comp> </mon>]]>
</message >
```

## basic declaration

- comments   <!-- comment content -->
  - CDATA       <![CDATA[text]]>
- declaration used in DTD
- <!DOCTYPE ...> <!ATTLIST ...> ...

Caracter Data: you let to know to the parser that a particular contains no markup and shoud be treated as a regular text

# XML - entity

dividing the document into smaller parts

```
<?xml version="1.0"?>
<!DOCTYPE list [
 <!ENTITY jnowak SYSTEM "jnowak.ent">
 <!ENTITY jkowalski SYSTEM "jkowalski.ent">
]>
<list>
 &jnowak;
 &jkowalski;
</list>
```

The diagram illustrates the expansion of XML entities. On the left, the XML code defines two entities: `jnowak` and `jkowalski`, each pointing to an external file (`jnowak.ent` and `jkowalski.ent`). Red arrows point from the entity references in the code to their respective expanded forms on the right. The expanded form for `jnowak` contains a `<data>` block with a `<person>` block for Jan Nowak, including a telephone number. The expanded form for `jkowalski` contains a similar `<data>` block with a `<person>` block for Jan Kowalski, also including a telephone number.

```
<data>
 <person>
 Jan Nowak
 </ person >
 <telephone>
 +48 (58) 347-1234
 </telephone >
 </data>
```

```
<data>
 <person>
 Jan Kowalski
 </ person >
 <telephone>
 +48 (58) 347-1534
 </telephone >
 </data>
```

# XML - processing instructions

allow documents to contain instructions for applications

<? target instructions?>

can be used to pass the information to applications (rarely used)

they are mostly used to link XML documents to a stylesheet so such process i will you on class and project.  
Let's the parser run a piece of code

```
<?xml-stylesheet type="text/xsl" href="journey.xslt"?>
```

```
<?php
 mysql_connect("base.pl", "kowalski", "password");
 mysql_close()
?>
```

# XML – a good style

---

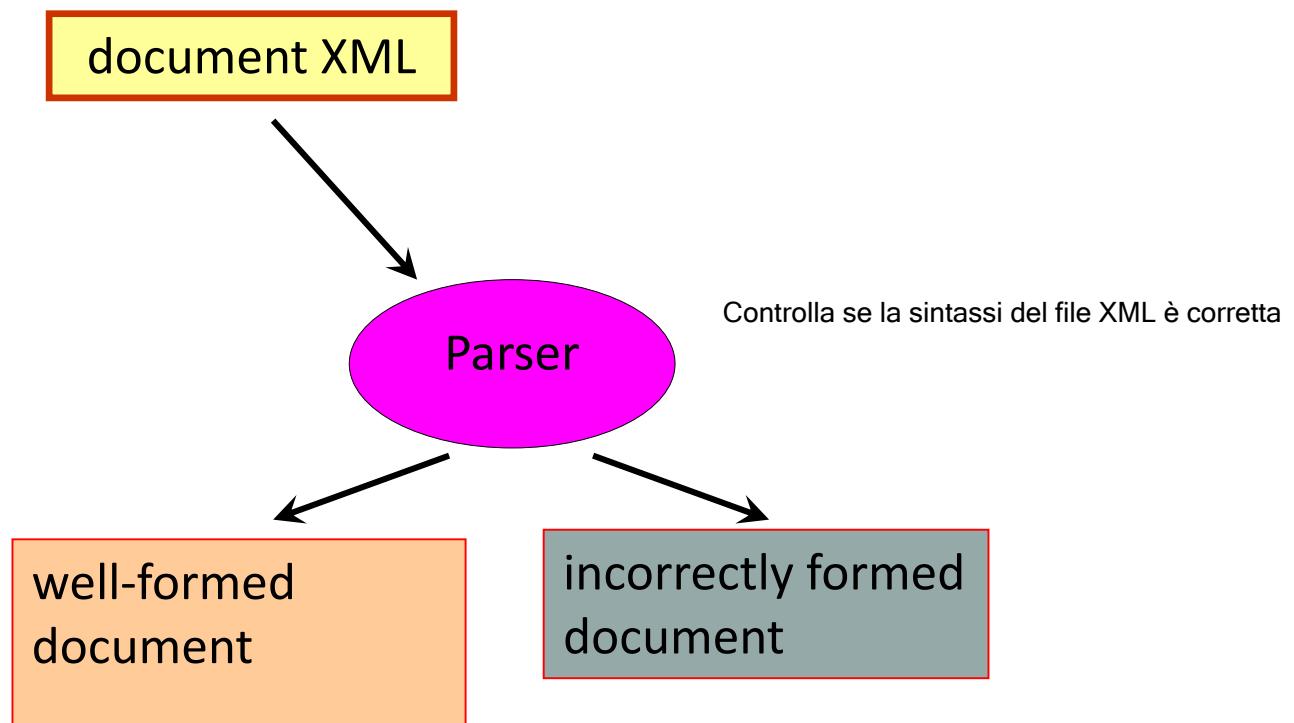
the appearance of the document

```
<?xml version="1.0" encoding="UTF-8"?>
<books>
 <book category="fantasy">
 <title>The colour of magic</title>
 <author>Terry Pratchett</author>
 <year>2012</year>
 <price>30.00</price>
 </book>
 <book category="fantasy">
 <title>Guards! Guards!</title>
 <author>Terry Pratchett</author>
 <year>2012</year>
 <price>30.00</price>
 </book>
</books>
```

# Correctness of the document

---

well-formed document



# Correctness of the document

---

## *WELL-FORMED DOCUMENT*

There must be exactly one root element

Every start-tag has a matching end-tag

Attribute values must be quoted

Elements may nest, but must not overlap

...

# XML (*Extensible Markup Language*)

---

10 II 1998 r XML 1.0

meta language

- a set of rules on the basis of which other languages can be created

extensible, formalized

designed to store and transport data

# XML (*Extensible Markup Language*)

---

open standard

not licensed

text language

flexible

- flexible document structure

self-describing

contextual

- the ability to store data with their meaning

# XML (*Extensible Markup Language*)

---

platform independent

hierarchical

separate form (how it looks) from content (what it is)

the possibility of a varied presentation

- XSLT, FO

modular

# Uses of XML

---

## Web pages

- WML, XHTML
- greater formalization of information, its processing

XML can be used to exchange the information between organizations and systems

## Description of resources

- information retrieval, resource descriptors
  - RDF (*Resource Description Framework*), OWL (*Web Ontology Language*)
  - WSDL (*Web Services Description Language*)

Representation of semi-structural information

Databases - intermediate layer

Guarda quanto è complesso scrivere una semplice formula con MathXML

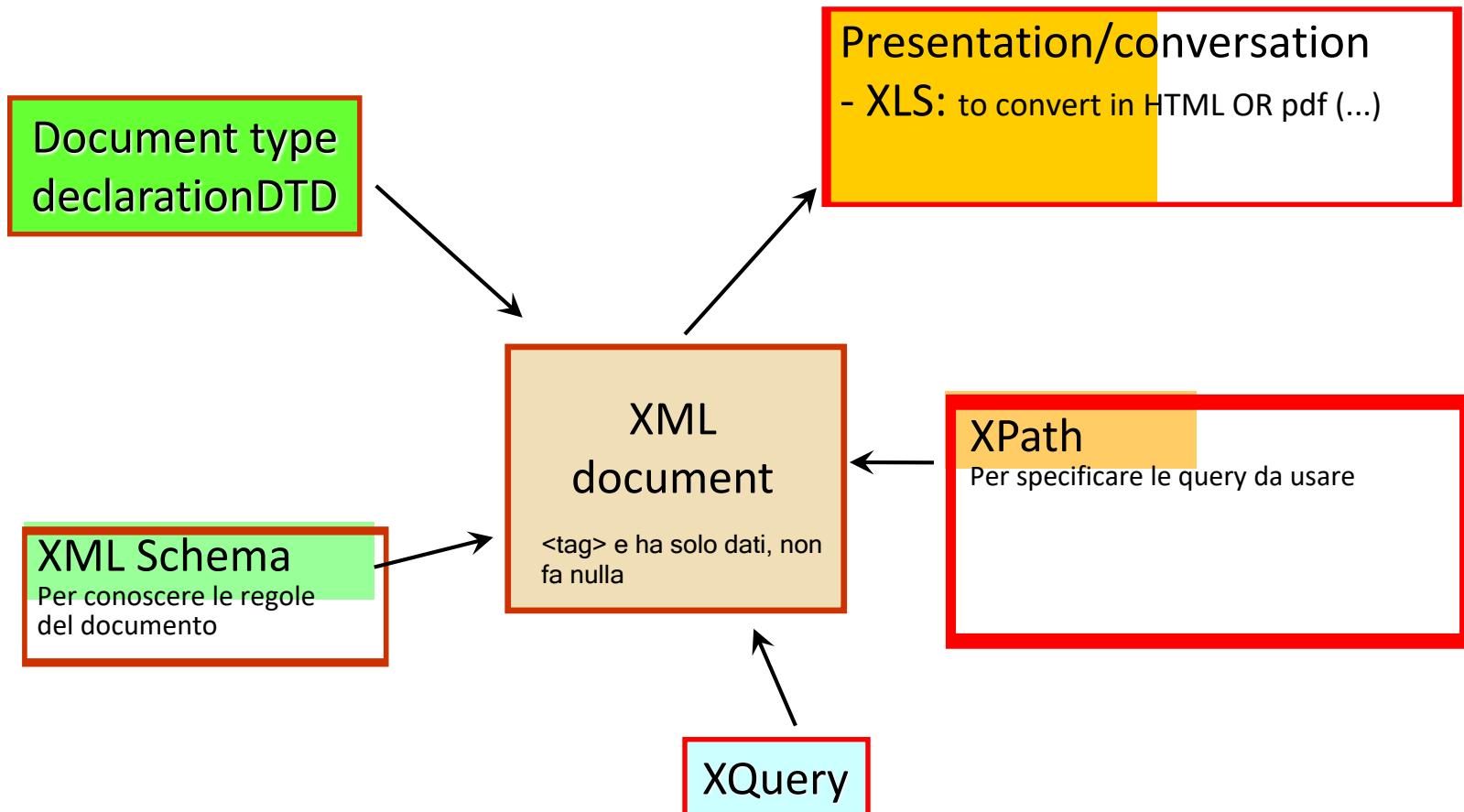
$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

```
<math mode="display" xmlns="http://www.w3.org/1998/Math/MathML">
<semantics>
 <mrow>
 <mi>x</mi>
 <mo>=</mo>
 <mfrac>
 <mrow>
 <mo form="prefix">−j<!-- - --></mo>
 <mi>b</mi>
 <mo>±j<!-- ± --></mo>
 </mrow>
 <msqrt>
 <msup>
 <mi>b</mi>
 <mn>2</mn>
 </msup>
 <mo>−j<!-- - --></mo>
 <mn>4</mn>
 <mo>⁢j<!-- ⁢ --></mo>
 <mi>a</mi>
 <mo>⁢j<!-- ⁢ --></mo>
 <mi>c</mi>
 </msqrt>
 </mrow>
 <mrow>
 <mn>2</mn>
 <mo>⁢j<!-- ⁢ --></mo>
 <mi>a</mi>
 </mrow>
 </mfrac>
</mrow>
<annotation encoding="TeX">
 x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}
</annotation>
<annotation encoding="StarMath 5.0">
 x={-b plusminus sqrt {b^2 - 4 ac}} over {2 a}
</annotation>
</semantics>
</math>
```

Che sarebbe cortissima in altri linguaggi

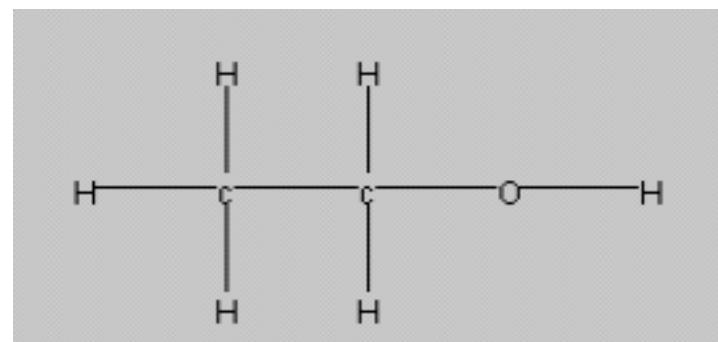
```
x=\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}
```

# XML document



Esempio dello standard di XML per descrivere le formule chimiche (nelle slides dopo ci sono altri esempi)

```
<CML:molecule id="test">
 <CML:atomArray builtin="elsym">
 c c O H H H H H</CML:atomArray>
 <CML:atomArray builtin="x2" type="float">
 0 0 0 0 0 -2 -2 2 2</CML:atomArray>
 <CML:atomArray builtin="y2" type="float">
 0 2 4 -2 6 0 2 0 2</CML:atomArray>
 <CML:bondArray builtin="atid1">
 1 1 1 1 2 2 2 3</CML:bondArray>
 <CML:bondArray builtin="atid2">
 2 4 6 8 7 9 3 5</CML:bondArray>
 <CML:bondArray builtin="order" type="integer">
 1 1 1 1 1 1 1 1
 </CML:bondArray>
</CML:molecule>
```

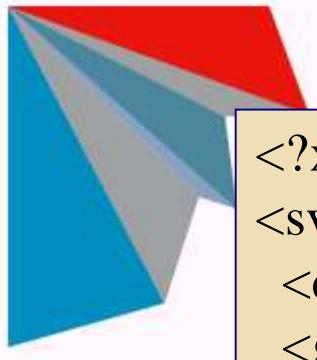


```

<?xml version="1.0"?>
<math xmlns="http://www.w3.org/1998/Math/MathML"
 style="color: #000000; font-family: times">
<mrow>
 <mi>z</mi>
 <mo>=</mo>
 <mfrac>
 <mrow>
 <mn>2</mn>
 <mo>⁢</mo>
 <msub>
 <mi>M</mi>
 <mn>i</mn>
 </msub>
 </mrow>
 <msqrt>
 <mrow>
 <msup>
 <mi>x</mi>
 <mn>3</mn>
 </msup>
 <mo>+</mo>
 <mn>5</mn>
 <mo>-</mo>
 <mi>y</mi>
 </mrow>
 </msqrt>
 </mfrac>
</mrow>
</math>

```

$$z = \frac{2M_i}{\sqrt{x^3 + 5 - y}}$$



```
<?xml version="1.0" standalone="no"?>
<svg width="100%" height="100%" xmlns='http://www.w3.org/2000/svg'>
 <desc></desc>
 <g id="polygonMask" style="stroke: none;">
 <polygon style="fill: #0192BF;" points="30,60 30,230 108,208"
 stroke="1"/>
 <polygon style="fill: #9EA4A6;" points="30,60 108,208 125,155"
 stroke="1"/>
 <polygon style="fill: #89A5C7;" points="30,60 125,155 144,161"
 stroke="1"/>
 <polygon style="fill: #488B9B;" points="30,60 144,161 138,115"
 stroke="1"/>
 <polygon style="fill: #9EA4A6;" points="30,60 138,115 180,114"
 stroke="1"/>
 <polygon style="fill: #EC1608;" points="30,60 180,114 160,60"
 stroke="1"/>
 </g>
</svg>
```



```
<bar barid="treb-11"> F 5(3F En Eb D C Eb D C B) G
 5(En Eb D C B An G F C)
 <notation>
 <slurend end="slur11" beat="3" />
 </notation>
</bar>
<bar barid="treb-12"> D:2 D (5B:2)3
</bar>
```

# Summary

---

## XML

- **meta language = we can create another languages using XML**
- **separate form (appearance) from content = il modo di presentare è diviso dai dati. XML è solo dati**
- text
- expandable
- flexible
- self-describing
- contextual
- open standard

DTD -> lo useremo nel progetto  
XML Schema -> nel prossimo laboratorio

# XSDL XML Schema Description Language

---

DR WIOLETA SZWOCH  
DEPARTMENT OF INTELLIGENT INTERACTIVE  
SYSTEMS

Perchè ne abbiamo bisogno? -> le stesse informazioni (nome, cognome e data) possono essere salvate in più modi. Se ogni persona crea la sua versione è un problema. Bisogna creare delle regole.

# XML Schema

---

An XML Schema describes the structure of an XML document.

dictionary for XML file

2001 standard XML Schema

W3C XML Schema specification

- Part 0 fundamentals
- Part 1 structures                elements, attributes, namespaces
- Part 2 data types

# Purpose of XML Schema

---

data validation

- elements and attributes structure
- elements order
- values of elements and attributes

system documentation

modifying data

application-specific information

recipe for language

validator control correctness of the document ↗

the programmer analyzes only the correct documents, he does not have to examine erroneous cases

la stessa cosa può essere descritta in più modi con XML schema

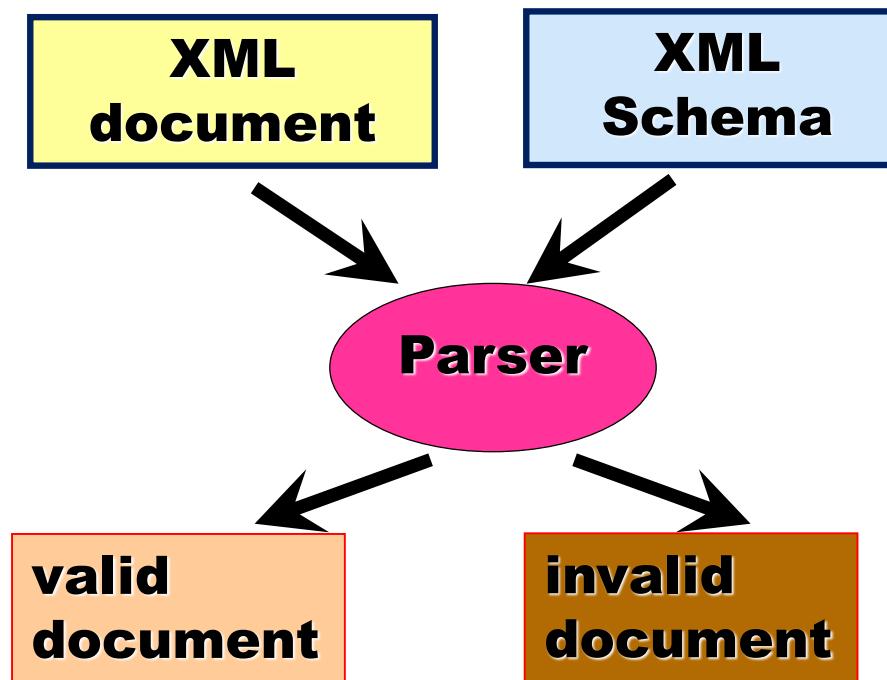
Come possiamo controllare se il nostro XML file va bene con l'XML Schema che abbiamo

# Correctness of the document

---

well-formed document

valid document



# Correctness of the document

---

## *WELL-FORMED*

There must be exactly one root element

Every start tag has a matching end tag

Attribute values must be quoted

Elements may nest, but must not overlap

...

## *VALID*

is well- formed

the definition of the document exists (DTD, XML Schema, . . . )

the content of the XML document is consistent with the definition of the document

```
<?xml version="1.0" encoding="UTF-8"?>
<osoby xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="Osoby.xsd">
 <osoba>
 <nazwisko>Kowalski</nazwisko>
 <imie>Jan</imie>
 <data_ur>1980.11.11</data_ur>
 <wyksztalcenie>wyższe</wyksztalcenie>
 <miejscze_pracy>Intel</miejscze_pracy>
 </osoba>
 <osoba>
 <nazwisko>Nowacki</nazwisko>
 <imie>Eustachy</imie>
 <data_ur>1998.10.10</data_ur>
 <wyksztalcenie>podstawowe</wyksztalcenie>
 <miejscze_nauki>Szkoła Podstawowa</miejscze_nauki>
 </osoba>
</osoby>
```

XML FILE valid with XML schema

```
<?xml version="1.0" encoding="UTF-8"?>
<osoby xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="Osoby.xsd">
 <osoba nazwisko="Kowalski">
 <imie>Jan</imie>
 <data_ur>1980.11.11</data_ur>
 <miejscze_pracy>Intel</miejscze_pracy>
 </osoba>
 <osoba nazwisko="Nowacki">
 <imie>Eustachy</imie>
 <data_ur>1998.10.10</data_ur>
 </osoba>
</osoby>
```

XML FILE not valid with  
XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
 <xs:element name="osoby">
 <xs:complexType>
 <xs:sequence maxOccurs="unbounded">
 <xs:element name="osoba">
 <xs:complexType>
 <xs:sequence>
 <xs:element name="nazwisko" type="xs:string"/>
 <xs:element name="imie" type="xs:string"/>
 <xs:element name="data_ur" type="xs:string"/>
 <xs:element name="wyksztalcenie" type="xs:string"/>
 <xs:choice>
 <xs:element name="miejscze_pracy" type="xs:string"/>
 <xs:element name="miejscze_nauki" type="xs:string"/>
 </xs:choice>
 </xs:sequence>
 </xs:complexType>
 </xs:element>
 </xs:sequence>
 </xs:complexType>
 </xs:element>
</xs:schema>
```

XML SCHEMA FILE

# XML Schema

---

XML Schema is an external file

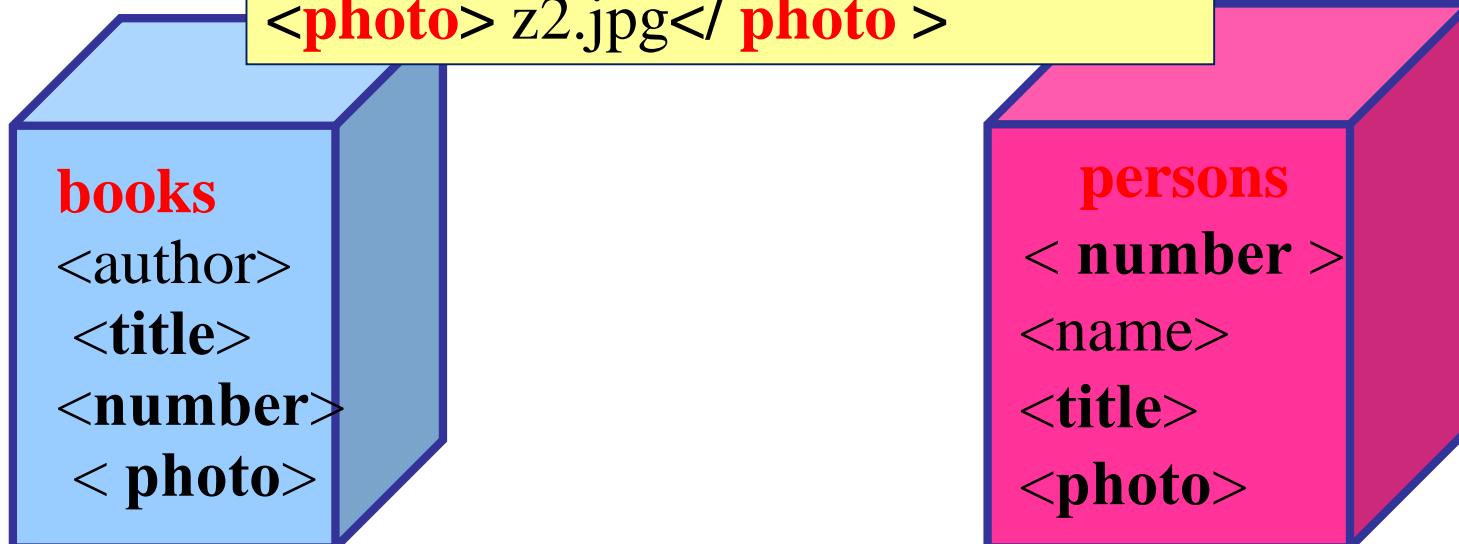
XML Schema defines classes of XML documents

- XML Schema
- XML Data,
- Document Content Description (DCD),
- Schema for Object-oriented XML (SOX)
- Schematron
- TREX
- RELAX, XDR, HOOK, DSD, Assertion Grammars

Problema: se vogliamo unire due file ma che hanno gli stessi attributi non sappiamo come fare a distinguarli. In questo caso abbiamo il problema con title, number and photo. Purtroppo la slide successiva non la fa vedere, ma per capire basta aggiungere prima del elemento in nome de file seguito dai due punti e poi il tag. Esempio: <books:title> ... </books:title>

# XML namespaces

```
<number> 1</ number >
<author> Kowalski</author>
<title> Profesor </ title >
< photo > z1.jpg</ photo >
<name> Nowak</ name>
< title > Profesor </ title >
< number > 1</ number >
<photo> z2.jpg</ photo >
```



# XML namespaces

---

necessary when we have a name conflict because we use different markup languages

Marker language treated as a set of names - namespace

## Namespace

- namespace is defined by **xmlns:prefix**
- prefix is used for every tag or attribute from the namespace
- identified by URI
- each tag from the namespace has a prefix

Un namespace si devinisce con `xmlns:"nome_del_namespace"`.

Qui ci sono tutti i casi, penso li capirai con il tempo.

# XML namespaces

## namespace in document

- no namespace
- one
- many
- local
- default

```
<?xml version="1.0"?>
<book>
 <title> Digital documents </title>
</book>
```

```
<?xml version="1.0"?>
<bk:book xmlns:bk='http://www.books.org/books'>
 <bk:title> Digital documents </bk:title>
</bk:book>
```

```
<?xml version="1.0"?>
<bk:book xmlns:bk='http://www.books.org/books'
 xmlns:isbn='urn:ISBN:0-395-36341-6'>
 <bk:title> Digital documents </bk:title>
 <isbn:number>1568491379</isbn:number> </bk:book>
```

```
<?xml version="1.0"?>
<bk:book xmlns:bk='http://www.books.org/books'>
 <bk:title> Digital documents </bk:title>
 <isbn:number xmlns:isbn='urn:ISBN:0-395-36341-6'>
 1568491379
 </isbn:number>
</bk:book>
```

```
<?xml version="1.0"?>
<book xmlns='http://www.books.org/books'>
 <title> Digital documents </title>
 <isbn:number xmlns:isbn='urn:ISBN:0-395-36341-6'>
 1568491379
 </isbn:number>
</book>
```

Usa gli standard!!!!

# XML standard namespaces

---

| <b>language</b> | <b>prefiks</b> | <b>URI</b>                                                                                        |
|-----------------|----------------|---------------------------------------------------------------------------------------------------|
| HTML            | html:          | <a href="http://www.w3.org/TR/REC-html40">http://www.w3.org/TR/REC-html40</a>                     |
| XML Schema      | xsd:           | <a href="http://www.w3.org/2001/XMLSchema-instance">http://www.w3.org/2001/XMLSchema-instance</a> |
| XSLT            | xsl:           | <a href="http://www.w3.org/1999/XSL/Transform">http://www.w3.org/1999/XSL/Transform</a>           |
| XSL             | fo:            | <a href="http://www.w3.org/1999/XSL/Format">http://www.w3.org/1999/XSL/Format</a>                 |
| Xlink           | xlink:         | <a href="http://www.w3.org/1999/xlink">http://www.w3.org/1999/xlink</a>                           |

# XML Schema

---

## XML Schema namespace

- <http://www.w3.org/2001/XMLSchema>
- there are all Schema components
  - element, attribute, schema, complexType, string, sequence ...

Questi due schemi sono assolutamente identici al 1 ha il namespace di default senza prefisso, il 2 ha il prefisso standard per XML Schema

# XML Schema

---

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
 <element name="Person">
 <complexType>
 ...
 </complexType>
 </element>
 ...
</schema>
```

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <xsd:element name="Person">
 <xsd:complexType>
 ...
 </xsd:complexType>
 </xsd:element>
 ...
</xsd:schema>
```

XML Schema è pur sempre un file XML quindi anche qui ci sono le solite regole tipo la versione e il root ecc...

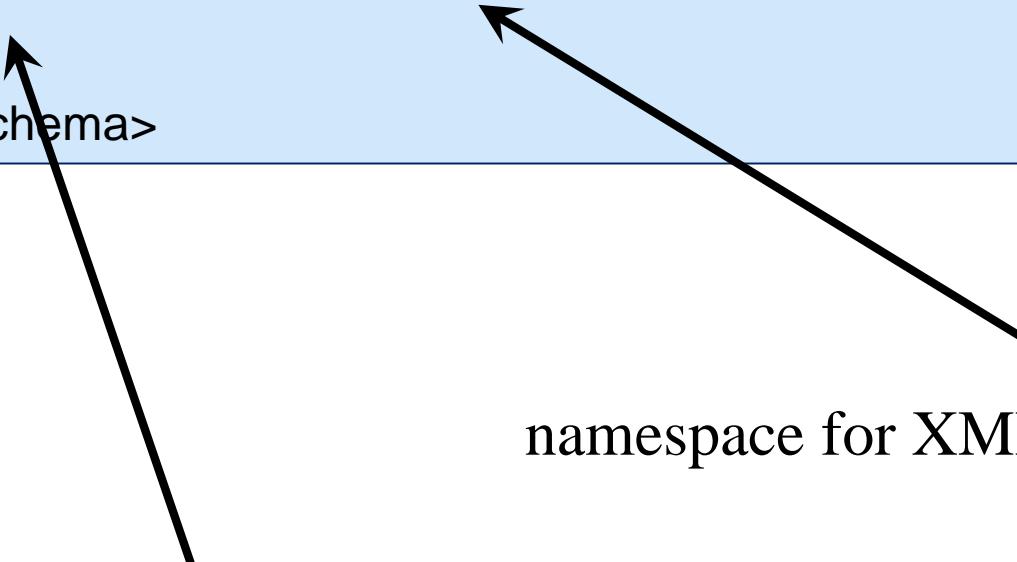
# XML Schema

---

Structure of the XML Schema document

```
<?xml version="1.0" ?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
....
</xsd:schema>
```



namespace for XML Schema

the main content of the scheme

# XML Schema

---

## element declaration

- type, numer of elements, hierarchy

File XML, nel file XML Schema dovremo essere in grado di definire elementi e attributi

## attribute declaration

## new types definition

```
<person surname='Nowak'>
 <name> Anna </name>
 <name> Marta </name>
 <data> 1980.01.01 </data>
</ person>
```

Blu = XML Schema

Yellow = XML

# Element declaration

---

## Element

```
<xsd:element name="Name_of_the_element" type="Data_Type" ...
attributes/>
```

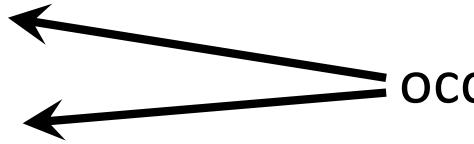
```
< Name_of_the_element >
 element content according to Data_Type
</ Name_of_the_element >
```

```
<xsd:element name="surname" type="xsd:string" />
```

```
<surname> Kowalski </ surname>
```

# Elements attributes

---

- name
  - type
  - id
  - minOccurs
  - maxOccurs
  - ref
- occurrence
- 
- ```
graph LR; A[minOccurs] --> C[occurrence]; B[maxOccurs] --> C;
```

```
<xsd:element name="surname" type="xsd:string" />
```

Element attributes

minOccurs, maxOccurs

- default value
 - minOccurs = "1"
 - maxOccurs = "1"
- optional
 - minOccurs = "0"
- Unspecified number of occurrences
 - maxOccurs = "unbounded"
(non ci sono vincoli)

```
<schema xmlns="http://www.w3.org/2001/XMLSchema">
...
<element name="name" type="string"
        minOccurs="0" maxOccurs="unbounded" />
<element name="surname" type="string"
        minOccurs="1" maxOccurs="1" />
<element name="phone" type="long"
        minOccurs="1" maxOccurs="3" />
...
</schema>
```

Element declaration

Elements

- Global
- Local
- References

Global elements

Placed directly in the main element

Messi direttamente nella root del file

Visible in the whole scheme

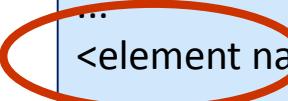
Sono visibili in tutto lo schema

They have a name

They have a type

They can have optional attributes

```
<FirstGlobalElement>  
    content  
</ FirstGlobalElement >
```

```
<schema xmlns="http://www.w3.org/2001/XMLSchema>  
...  
  
<element name="FirstGlobalElement" type="string" />  
...  
</schema>
```

Local elements

They are defined in the context of other elements

They have a name, type, they can have optional attributes

XML SCHEMA

Se un elemento è locale, è invisibile nel resto dello schema

XML FILE (rispetta le condizioni dell'XML SCHEMA)

```
< FirstGlobalElement >
    < LocalElement >1.2 </ LocalElement >
</ FirstGlobalElement >
< SecondGlobalElement >
    < LocalElement >plain text </ LocalElement >
</ SecondGlobalElement >
```

```
<schema xmlns="http://www.w3.org/2001/XMLSchema"
        <element name="FirstGlobalElement">
        ...
            <element name="LocalElement" type="float" />
        ...
            </element>
        <element name="SecondGlobalElement" >
        ...
            <element name="LocalElement" type="string" />
        ...
            </element>
        </schema>
```

We can declare a global element and refer to it in another element

References to the elements

They have the ref attribute instead of the name attribute

They do not have any type attribute

They allow multiple use of element declaration

They can only refer to global elements(?)

```
<schema xmlns="http://www.w3.org/2001/XMLSchema>
  <element name="FirstElement" >
    ...
    <element ref="SecondElement" />
    ...
  </element>
  ...
  <element name="SecondElement" type="string" />
</schema>
```

```
<FirstElement>
  <SecondElement>wartość</SecondElement>
</ FirstElement >
```

Declaration v.s. Definition

Declarations – enable element and attributes with specific names and types to appear in document instance

- element declaration
- attribute declaration

thanks to declaration elements and attributes with specific name and type can appear in a document instance (in a XML file)

Definitions - create a new type

- simple type definitions
- complex type definitions
- attribute group, model group definitions

Elements - fixed and default value

fixed = come default ma non puoi rimodificarlo

```
<xsd:element name= " number" fixed= "1.0" />
```

< number >1.0</ number >

< number />

< number >2.0</ number >

default = valore di default che ha l'attributo se è vuoto

```
<xsd:element name= " number" type= "decimal" default= "1.0" />
```

< number >1.0</ number >

< number />

< number >2.0</ number >

E' un errore avere sia default che fixed

Data types

built-in = are part of the standard

user-defined = are types that the user have defined

simple type: when you want to create a new type that is a refinement of a built-it type

complex type = can contain sub element. Use when you want to define child elements and/or attributes of an element

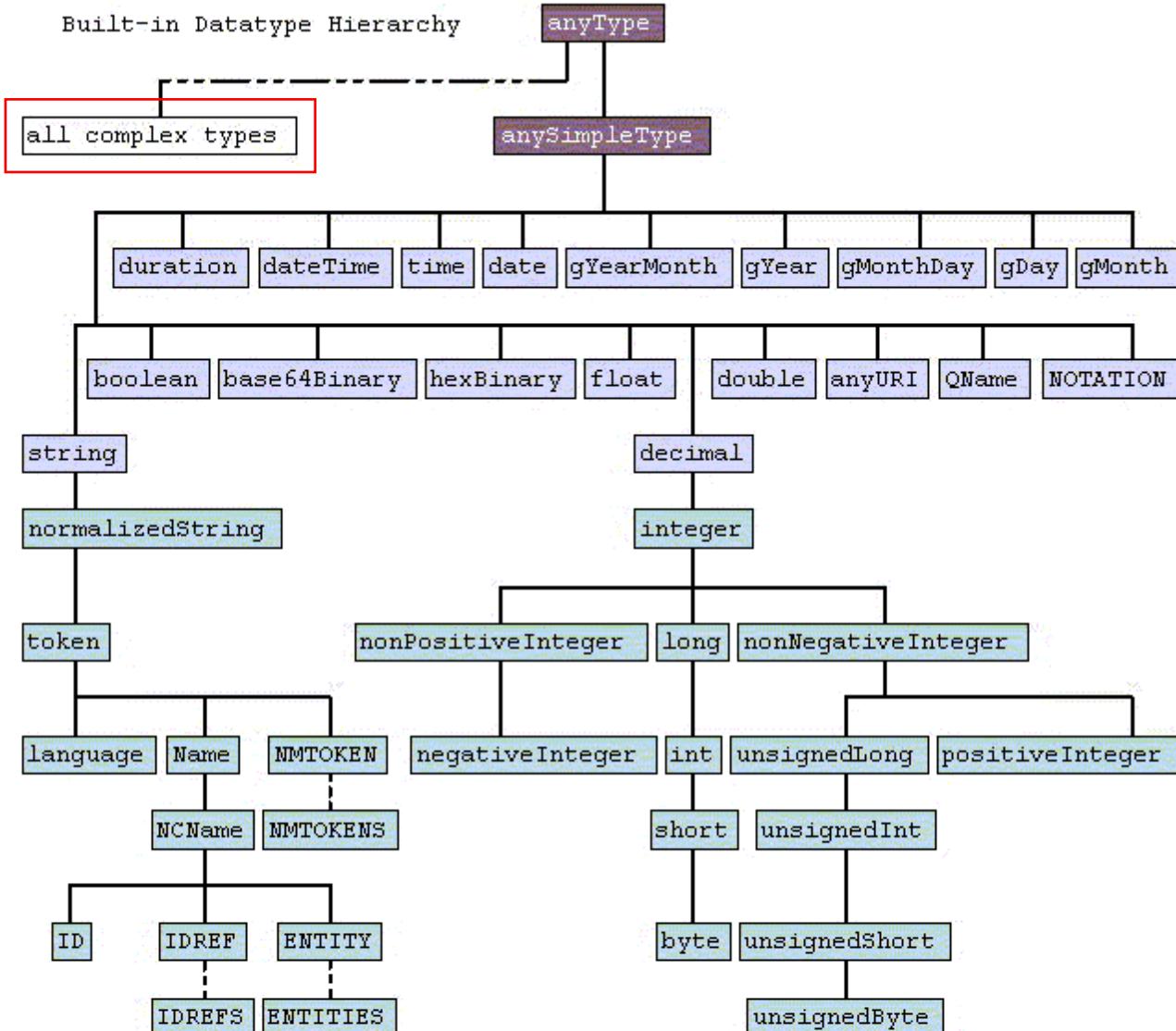
primitive

- they do not require defining derived
- derived from other types

derived

- derived form other types

Built-in Datatype Hierarchy



ur types



derived by restriction



built-in primitive types



derived by list



built-in derived types

derived by extension or
restriction

complex types

Si sfrutta la keyword complexType, ma non hai capito bene MA DOPO C'E' UN ESEMPIO.

Complex types

Only the type defined globally
and named can be used many times

```
<xsd:element name="Student">  
  <xsd:complexType name="personType">  
    ...  
  </xsd:complexType>  
</xsd:element>
```

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">  
  <xsd:element name="Student">  
    <xsd:complexType>  
      ...  
    </xsd:complexType>  
  </xsd:element>
```

Anonymous type

(può essere anonimo solo un
elemento)

```
<xsd:element name="Teacher" type="personType" />  
  <xsd:complexType name="personType">  
    ...  
  </xsd:complexType>  
</xsd:schema>
```

Devi prima definirlo nella root come è stato fatto
qui con name. Qui abbiamo definito il type come "personType" lo possiamo utilizzare più volte.

Named type

Complex types

```
<xsd:element name="Student" type="personType" />
<xsd:element name="Teacher" type="personType" />
<xsd:complexType name="personType">
    .....
    <xsd:element name="Surname" type="xsd:string" />
    <xsd:element ref="Name" />
    <xsd:element name="Phone" type="xsd:long" />
    .....
</xsd:complexType>
<xsd:element name="Name" type="xsd:string" />
```

Complex types

Order indicators

- way of occurrence of subelements in the complex type

- sequence

```
<xsd:element name="Student" type="personType" />
<xsd:element name="Teacher" type="personType" />
<xsd:complexType name="personType">
    .....
    <xsd:element name="Surname" type="xsd:string" />
    <xsd:element ref="Name" />
    <xsd:element name="Phone" type="xsd:long" />
    .....
</xsd:complexType>
<xsd:element name="Name" type="xsd:string" />
```

- choice

- all

The sentence in the XML document must appear in the order they are declared in the schema file

Order indicators

sequence

- strict order of sub-elements

```
<xsd:complexType name="personType">
  <xsd:sequence>
    <xsd:element name="Surname" type="xsd:string" />
    <xsd:element name="Name" type="xsd:string" />
    <xsd:element name="Phone" type="xsd:long" />
  </xsd:sequence>
</xsd:complexType>
```

- Equivalent in DTD: (non ci interessa ora)

```
<!ELEMENT xyz (Surname, Name, Phone)>
```

Order indicators

choice (LO USI PER LE ALTERNATIVE)

- only one of the declared child elements

```
<xsd:complexType name="Dane">
  <xsd:choice>
    <xsd:element name="Student" type="xsd:string" />
    <xsd:element name="Assistant" type="xsd:string" />
    <xsd:element name="Professor" type="xsd:string" />
  </xsd:choice>
</xsd:complexType>
```

- Equivalent in DTD:

```
<!ELEMENT xyz (Student | Assistant | Professor)>
```

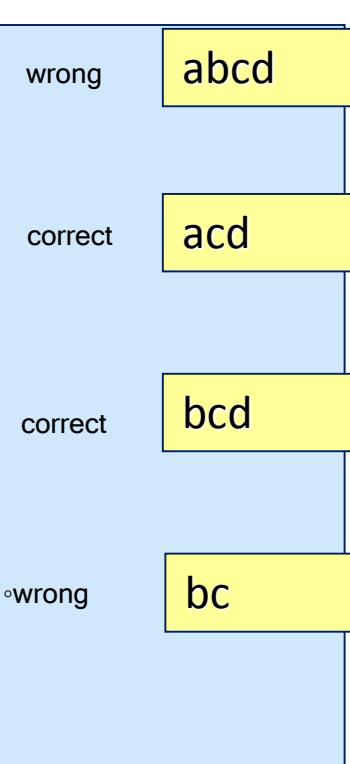
Combination of choose and sequence

Order indicators

Nesting choice and sequence

```
<xsd:sequence>
  <xsd:choice>
    <xsd:element name="a" type="xsd:long" />
    <xsd:element name="b" type="xsd:string" />
  </xsd:choice>
  <xsd:sequence>
    <xsd:element name="c" type="xsd:int" />
    <xsd:element name="d" type="xsd:float" />
  </xsd:sequence>
</xsd:sequence>
```

```
<!ELEMENT xyz ((a | b), (c, d))>
```



Fa vari esempi con questo, dalla slide così non puoi capire più di tanto (per ascoltare L.4 01:08:00

Order indicators

Repeating the sequence/choice

```
<xsd:sequence maxOccurs="unbounded">  
  <xsd:choice minOccurs="0" maxOccurs="1">  
    <xsd:element name="a" type="xsd:long" />  
    <xsd:element name="b" type="xsd:string" />  
  </xsd:choice>  
  <xsd:sequence minOccurs="0" maxOccurs="unbounded">  
    <xsd:element name="c" type="xsd:int" />  
    <xsd:element name="d" type="xsd:float" />  
  </xsd:sequence>  
</xsd:sequence>
```

```
<!ELEMENT xyz ((a | b)?, (c, d)*)+>
```

Order indicators

all

- all elements in any order (devono apparire)

```
<xsd:all>
  <xsd:element name="Surname" type="xsd:string" />
  <xsd:element ref="Name" />
  <xsd:element name="Phone" type="xsd:long" />
</xsd:all>
```

- limitations all
 - maxOccurs = "1", minOccurs = "0" or "1"
 - all cannot be nested within sequence, choice, all
 - in all only elements
- increasing the computational complexity of validating parsers
- Equivalent in DTD:

????

Su questo argomento lei ha molte più slide su cui spiega (data oriented vs document oriented)

Mixed content type

It supports all properties of the complex type:

- minOccurs, maxOccurs, sequence ...

```
<complexType name="MyName2" mixed="true">
  <sequence>
    <element name="FirstName" maxOccurs="2" type="string" />
    <element name="LastName" type="string" />
  </sequence>
</complexType>
```

```
<person>
  Definicja dla osoby
  <FirstName>Anna</FirstName> bez innych imion
  <FirstName>Empty</FirstName>
  <LastName>Kowalska</LastName> z mieszaną zawartością
</person>
```

anyType

The elements by default are "any type" *anyType*

- The following declarations are equivalent

```
<xsd:element name="Data" type="xsd:anyType" />
```

```
<xsd:element name="Data" />
```

```

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <books>
    <author>
      <name>student's name</name>
      <surname>student's surname</surname>
    </author>
    <study kind="lecture" >
      <activities id="1">
        <topic>Hypertext and hypermedia</topic>
        <range>
          <component>Hypertext & hypermedia</component>
          <component>HTML CSS</component>
          <component>XML</component>
          <component>XML Schema</component>
          <component>DTD</component>
          <component>XSLT</component>
          <component>FO</component>
        </range>
        <score>30</score>
      </activities>
    </study>
    <study kind="laboratory" obligatory="yes">
      <activities id="1">
        <topic>HTML + CSS</topic>
        <range>
          <component>structure of the page</component>
          <component>links</component>
          <price>29.99</price>
        </range>
      </activities>
    </study>
  </books>
</xs:schema>

```

Manca type="BookType" nella primo elemento!
 quindi diventa "anytype" e lo schema sotto è
 come viene letto.

Nota che il file XML è valido con la
 dichiarazione sotto, perchè abbiamo usato
 appunto "anytype". L'unico vincolo è che il
 nome del root sia "Books"

--> NON USARE MAI ANYTYPE

```

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <books>
    <book category="web">
      <title>Learning XML</title>
      <author>Erik T. Ray</author>
      <year>2003</year>
      <price>39.95</price>
    </book>
  </books>
</xs:schema>

```

Element any

The <any> element enables us to extend the XML document with elements not specified by the schema.
Puoi scrivere altro dopo il cognome.

```
<complexType name="personType" mixed="true">
    <sequence>
        <element name="FirstName" maxOccurs="2" type="string" />
        <element name="LastName" type="string" />
        <any minOccurs="0"/>
    </sequence>
</complexType>
```

Empty content

element may be empty

Dichiaro un elemento complesso ma dentro non ci metto nulla

```
<xsd:element name="emptyElement">  
  <xsd:complexType />  
</xsd:element>
```

```
<emptyElement />
```

Poi c'è una slide che non ha messo qui

Element group

the group element is used to define a group of elements to be used in complex type definitions.

named groups of model fragments of content models that you can reuse

benefits

- indicates that some complex types have similar subelements
- allows you to create more compact schema

The group can not contain both elements and attributes

Groups must be declared globally

Esempio of group element:

```
<complexType name="StudentType" mixed="true">
  <sequence>
    <group ref="MyGroup" />
  </sequence>
</complexType>

<complexType name="TeacherType" mixed="true">
  <sequence>
    <element name="Title" maxOccurs="3" type="string" />
    <group ref="MyGroup" />
  </sequence>
</complexType>

<group name="MyGroup" >
  <sequence>
    <element name="FirstName" type="string" />
    <element name="LastName" type="string" />
  </sequence>
</group>
```

Simple types

Creating your **own** simple datatypes

Simple types inherit (ereditare) from:

- built-in types
- other simple types

Only simple content (not sub element or attributes are allowed in simple types)

General form of creating a new datatype (SIMPLETYPE) by specifying facet values

ESEMPIO A PAGINA DOPO

Facets:

- length
- minlength
- maxlength
- pattern
- enumeration
- minInclusive
- maxInclusive
- minExclusive
- maxExclusive

```
<simpleType name="name">  
  <restriction base="source">  
    <facet value="..." />  
    <facet value="..." />  
    ...  
  </restriction>  
</simpleType>
```

Sources:

- string
- boolean
- number
- float
- double
- duration
- dateTime
- time

...

...

Example of creating a new datatype by specifying facet values

```
<xsd:simpleType name="TelephoneNumberType">
  <xsd:restriction base="xsd:string">
    <xsd:length value="8"/>          Deve essere lunga 8 caratteri
    <xsd:pattern value="\d{3}-\d{4}"/>  Deve poi seguire la regola:
  </xsd:restriction>                3numeri - 4 numeri
</xsd:simpleType>
```

Facets of the string

The string primitive datatype has six optional facets:

- length
- minLength
- maxLength
- pattern
- enumeration
- whitespace (legal values:
preserve, replace, collapse)

```
<simpleType name="AustrianZIPCode">
  <restriction base="string">
    <length value="4" />    La lunghezza sarà 4
  </restriction>
</simpleType>
<simpleType name="InternationalZIPCode" >
  <restriction base="string">
    <minLength value="4" />
    <maxLength value="6" />
  </restriction>
</simpleType>
```

la lunghezza sarà compresa fra 4 e 6

enumeration

- List of predefined values (per forzare il valore fra alcuni scelti)

```
<day> Tuesday </day>
```

```
<simpleType name="dayType" >
  <restriction base="string">
    <enumeration value="Monday" />
    <enumeration value="Tuesday" />
    <enumeration value="Wednesday" />
    <enumeration value="Thursday" />
    <enumeration value="Friday" />
  </restriction>
</simpleType>
```

pattern

The value of pattern must be a regular expression

Si definisce un set di valori validi

[0-9] --> range di valori

{3} --> quanti valori devi mettere

esempio: [0-9]{2} --> 2 valori fra 0 e 9

```
<xsd:simpleType name="typNIP">  
  <xsd:restriction base="xsd:string">  
    <xsd:pattern value="([0-9]{3}-[0-9]{2}-[0-9]{2}-[0-9]{3})|  
      ([0-9]{3}-[0-9]{3}-[0-9]{2}-[0-9]{2})"/>  
  </xsd:restriction>  
</xsd:simpleType>
```

111-22-33-444-555-666-77-88

Regular expressions

- \p{L} A letter, from any language
- \p{Lu} An uppercase letter, from any language
- \p{Ll} A lowercase letter, from any language
- \p{N} A number - Roman, fractions, etc
- \p{Nd} A digit from any language
- \p{P} A punctuation symbol

? --> means 0 or 1 times
* --> means 0 or many times
+ --> means 1 or many times

```
<xsd:simpleType name="money">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="\p{Sc}\p{Nd}+(\.\p{Nd})?" />
  </xsd:restriction>
</xsd:simpleType>    Currency sign      Digit from any language
<xsd:element name="cost" type="money"/>
```

```
<cost>$45.99</cost>
<cost>¥300</cost>
```

Ci sono molti altri esempi durante la lezione

Facets of the integer datatype

The integer datatype has 8 optional facets:

totalDigits

pattern

whitespace

enumeration

maxInclusive

maxExclusive

minInclusive

minExclusive

```
<simpleType name="gradeType" >
  <restriction base="Integer">
    <minInclusive value="1" />
    <maxInclusive value="5" />
  </restriction>
</simpleType>
```

I valori possono andare da 1 a 5

Facets of the decimal datatype

The decimal datatype has 9 optional facets:

- totalDigits
- fractionDigits
- pattern
- whitespace
- enumeration
- maxInclusive
- maxExclusive
- minInclusive
- minExclusive

```
<simpleType name="4digitnumberType">
  <restriction base="decimal">
    <totalDigits value="4" />
  </restriction>
</simpleType>
```

Massimo numero di cifre in generale(?)


```
<simpleType name="4plus2digitnumberType" >
  <restriction base="decimal">
    <totalDigits value="6" />
    <fractionDigits value="2" />
  </restriction>
</simpleType>
```

Massimo numero di cifre dopo la virgola

Multiple Facets - "and" them together, or "or" them together?

```
<xsd:simpleType name="TelephoneNumberType">
  <xsd:restriction base="xsd:string">
    <xsd:length value="8"/>
    <xsd:pattern value="\d{3}-\d{4}"/>
  </xsd:restriction>
</xsd:simpleType>
```

An element declared to be of type TelephoneNumber must be a string of length=8 *and* the string must follow the pattern: 3 digits, dash, 4 digits.

```
<xsd:simpleType name="shapeType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="circle"/>
    <xsd:enumeration value="triangle"/>
    <xsd:enumeration value="square"/>
  </xsd:restriction>
</xsd:simpleType>
```

An element declared to be of type shape must be a string with a value of *either* circle, *or* triangle, *or* square.

Patterns, enumerations => "or" them together All other facets => "and" them together

Fixing a facet value

```
<xsd:simpleType name= "daysType">
  <xsd:restriction base="xsd:int">
    <xsd:minInclusive value="1" fixed="true"/>
    <xsd:maxInclusive value="30"/>
  </xsd:restriction>
</xsd:simpleType>
```

Provi a cambiare valore in un elemento derivato , ma non puoi cambiare il valore nell' elemento padre
è **fixed="true"**

```
<xsd:simpleType name= "myDaysType">
  <xsd:restriction base=" daysType ">
    <xsd:minInclusive value="7"/>
    <xsd:maxInclusive value="30"/>
  </xsd:restriction>
</xsd:simpleType>
```

Summary (riassunto) of declaring elements

```
<xsd:element name="name" type="type" minOccurs="int," maxOccurs="int"/>
```

```
<xsd:element name="name" minOccurs="int" maxOccurs="int">
  <xsd:complexType>
    ...
  </xsd:complexType>
</xsd:element>
```

```
<xsd:element name="name" minOccurs="int" maxOccurs="int">
  <xsd:simpleType>
    <xsd:restriction base="type">
      ...
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Lists

```
<numbers>2 10 6 1 15 150</numbers>
```

The list element defines a simple type element as a list of values of a specified data type.

The list can only contain simple types

You cannot create a list of lists

All list items of the same type

List elements in XML documents must be separated by whitespace

facets allowed on list

- *length, minLength, maxLength*
- they determine the length of the list
- *enumeration, pattern*
- they specify the values of t

```
<xsd:simpleType name="numbersType">
  <xsd:list itemType="xsd:positiveInteger "/>
</xsd:simpleType>
```

```
<xsd:element name="numbers" type="numbersType "/>
```

Puoi unire ogni tipo di file, ma solo simple type. Qui possiamo vedere due versioni della stessa cosa Ma nel secondo modo i simpletype non sono riutilizzabili perchè sono all'interno.

Union

Connecting any simple types into one

You can combine different types

```
<simpleType name="Type1" >  
  <restriction base="string">  
    <enumeration value="One" />  
    <enumeration value="Two" />  
  </restriction>  
</simpleType>  
<simpleType name="Type2" >  
  <restriction base="positiveInteger">  
    <maxInclusive value="2"/>  
  </restriction>  
</simpleType>  
<simpleType name="unionType" >  
  <union memberTypes="Type1 Type2">  
</simpleType>  
<element name="class" type="unionType" />
```

```
<class>1</class>  
<class>One</class>
```

```
<simpleType name="unionType" >  
  <union>  
    <simpleType>  
      <restriction base="string">  
        <enumeration value="One" /> ...  
      </restriction>  
    </simpleType>  
    <simpleType>  
      <restriction base="positiveInteger">  
        <maxInclusive value="2"/>  
      </restriction>  
    </simpleType>  
  </union>  
</simpleType>
```

Ci sono due vie: restringere un type oppure aggiungere valori a un base type

Derived complex types

restriction

- the set of values of the new type is a subset
- similar to simple types

extension

- adding additional elements to the base type

Base attributes indicates the base type

Derive by extension

```
<complexType name="personType" >
<sequence>
    <element name="name" type="string" />
    <element name="surname" type="string" />
</sequence>
</complexType>
```

```
<complexType name="extendedType" >
<complexContent>
    <extension base="personType">
        <sequence>
            <element name="email" type="string" />
        </sequence>
    </extension>
</complexContent>
</complexType>
```

Protremmo volere che da un elemento non si possa derivare nulla

Derive by extension

```
<complexType name="personType" final="#all" >
<sequence>
    <element name="name" type="string" />
    <element name="surname" type="string" />
</sequence>
</complexType>
```

```
<complexType name="extendedType" >
<complexContent>
    <extension base="personType">
        <sequence>
            <element name="email" type="string" />
        </sequence>
    </extension>
</complexContent>
</complexType>
```

dARA' ERRORE!!!

Element of the restricted type must be a valid element of the base type.

NOTA: Se vuoi eliminare un elemento nell'elemento derivato, esso deve necessariamente essere opzionale nel base type

Derive by restriction

```
<xsd:complexType name="Publication">
  <xsd:sequence>
    <xsd:element name="Title" type="xsd:string" maxOccurs="unbounded"/>
    <xsd:element name="Author" type="xsd:string" maxOccurs="unbounded"/>
    <xsd:element name="Date" type="xsd:gYear"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name= "SingleAuthorPublication">
  <xsd:complexContent>
    <xsd:restriction base="Publication">
      <xsd:sequence>
        <xsd:element name="Title" type="xsd:string" maxOccurs="unbounded"/>
        <xsd:element name="Author" type="xsd:string"/>
        <xsd:element name="Date" type="xsd:gYear"/>
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

Attributes

The attribute declarations always come last, after the element declarations.
(le dichiarazione di attributi viene sempre dopo quella degli elementi)

Declared locally or globally

Always simple type

```
<attribute name="attr1" type="string" />
<element name="testelement">
  <complexType>
    <sequence>
      ...
    </sequence>
    <attribute ref="attr1" />
    <attribute name="attr2" type="string" />
  </complexType>
</element>
```

Global attribute

reference

Local attribute

Attributes

default = automaticamente aggiunto all'attributo

fixed = automaticamente aggiunto all'attributo, ma non si può cambiare

Questi attributi hanno senso solo all'interno di elementi non negli attributi globali

use= "optional"

use= "required"

use= "prohibited"

Si capisce l'attributo di questo dal video (si usa per gli elementi derivate (quelli ristretti))

```
<xs:attribute name="lang" type="xs:string" use="required" fixed="EN"/>
```

anyAttribute

Permette di aggiungere attributi che non sono specificati dallo schema

```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
      <xs:any minOccurs= "0" />
    </xs:sequence>
    <xs:anyAttribute/>
  </xs:complexType>
</xs:element>
```

E' usato per raggruppare un gruppo di attributi

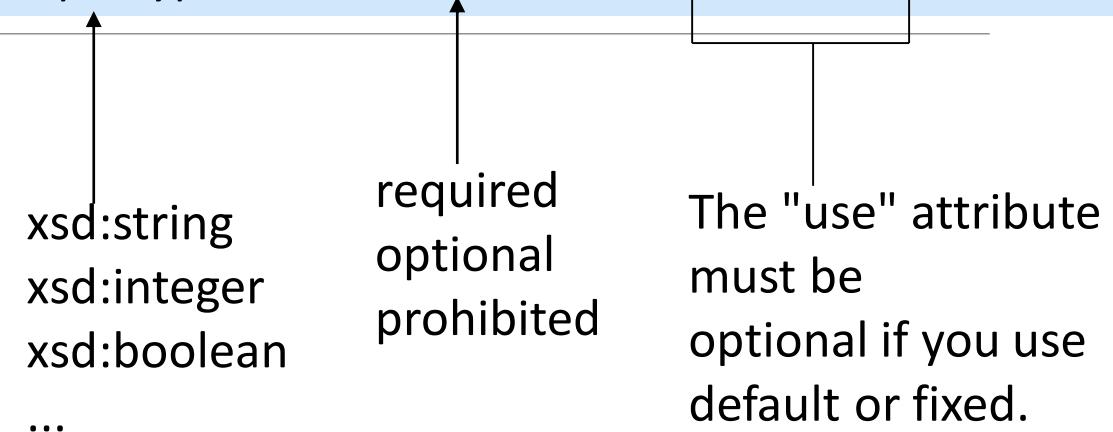
attributeGroup

```
<element name="testelement2">
    ...
    <attributeGroup ref="MyAttrGroup1" />
    ...
</element>
<attributeGroup name="MyAttrGroup1">
    <attribute name="attr1" type="long" />
    <attribute name="attr2" >
        <simpleType> ... </simpleType>
    </attribute>
</attributeGroup>
```

Ci sono due modi:

Summary of Declaring Attributes

```
<xsd:attribute name="name" type="simple-type" use="how-its-used" default/fixed="value" />
```

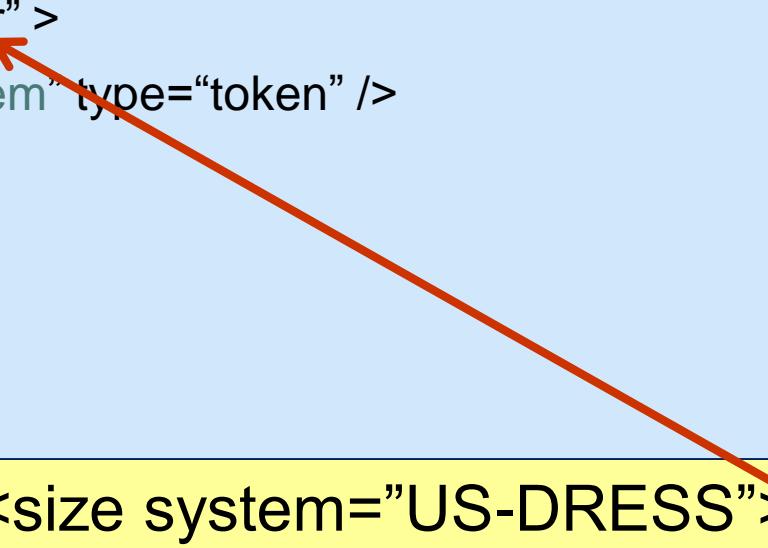


```
<xsd:attribute name="name" use="how-its-used" default/fixed="value">  
  <xsd:simpleType>  
    <xsd:restriction base="simple-type">  
      <xsd:facet value="value" />  
      ...  
    </xsd:restriction>  
  </xsd:simpleType>  
</xsd:attribute>
```

C' E' UNA SLIDE MOLTO IMPORTANTE SUL VIDEO, COPIALA!!!

```
<element name="size">  
<complexType>  
  <simpleContent>  
    <extension base="integer" >  
      <attribute name="system" type="token" />  
    </extension>  
  </simpleContent>  
</complexType>  
</element>
```

NON HAI CAPITO MOLTO BENE...



```
<size system="US-DRESS">10</size>
```

```
<element name="size">  
<complexType>  
  <attribute name="system" type="token" />  
</complexType>  
</element>
```

```
<size system="US-DRESS"/>
```

Include and import are used to add multiple schemas to a document

Include and import

include

```
<schema xmlns="http://www.w3.org/2001/XMLSchema  
         targetNamespace="http://www.example.org/NS2">  
    <include schemaLocation="SchemaToInclude.xsd" />  
  </schema>
```

import

```
<schema xmlns="http://www.w3.org/2001/XMLSchema  
         targetNamespace="http://www.example.org/NS2"  
         xmlns:imp1="http://www.importme.org/Import1">  
    <import namespace="http://www.importme.org/Import1  
              /schemas/Import1.xsd" />  
    <element name="test1" type="imp1:testType" />  
  </schema>
```

Annotations

for people <documentation>

for applications <appinfo>

Nulla a effetto sull Schema validation. Ma non puoi metterle ovunque: solo prima e dopo ogni elemento globale (ma alla fine si può lo stesso..) --> la slide che lo spiega qui non c'è)

```
<xsd:annotation>
  <xsd:documentation>
    This text is intended for humans.
  </xsd:documentation>
  <xsd:appinfo>
    <someXML>any well-formed XML</someXML>
  </xsd:appinfo>
</xsd:annotation>
```

nil (annullabile) content

attribute nillable

indicates that the element may not have content

in XML xsd:nill="true"

```
<complexType name="Price">
  <sequence>
    <element name="amount" type="integer" />
    <element name="currency" type="string" nillable="true" />
  </sequence>
</complexType>
```

```
<Price>
  <amount>100</amount>
  <currency xsd:nil="true" />
</Price>
```

XML Schema design methods

We can use 3 strategies:

Define the type of each element using a local type

Define a series of named complex and simple types at the top level of the XML Schema document and use those names to indicate the types to be used for the elements

Define a series of elements and groups of code at the top level of the Schema definition and then refer to those element definitions using the attribute ref

```

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="books">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="book" maxOccurs="unbounded" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="title" type="xs:string"/>
              <xs:element name="author" type="xs:string"/>
              <xs:element name="year" type="xs:short"/>
              <xs:element name="price" type="xs:float"/>
            </xs:sequence>
            <xs:attribute name="category" type="xs:string" use="optional"/>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

hCl sono varie possibilità, qui è definito tutto localmente

```

<?xml version="1.0" encoding="UTF-8"?>
<books>
  <book category="fantasy">
    <title>The colour of magic</title>
    <author>Terry Pratchett</author>
    <year>2012</year>
    <price>30.00</price>
  </book>
  <book category="fantasy">
    <title>Guards! Guards!</title>
    <author>Terry Pratchett</author>
    <year>2012</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title>Harry Potter</title>
    <author>J. K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</books>

```

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="books" type="booksType"/>
<xs:complexType name="bookType">
  <xs:sequence>
    <xs:element name="title" type="xs:string"/>
    <xs:element name="author" type="xs:string"/>
    <xs:element name="year" type="xs:short"/>
    <xs:element name="price" type="xs:float"/>
  </xs:sequence>
  <xs:attribute name="category" type="xs:string" use="optional"/>
</xs:complexType>
<xs:complexType name="booksType">
  <xs:sequence>
    <xs:element name="book" type="bookType" maxOccurs="unbounded" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
</xs:schema>
```

Altra strategia...

```
<?xml version="1.0" encoding="UTF-8"?>
<books>
  <book category="fantasy">
    <title>The colour of magic</title>
    <author>Terry Pratchett</author>
    <year>2012</year>
    <price>30.00</price>
  </book>
  <book category="fantasy">
    <title>Guards! Guards!</title>
    <author>Terry Pratchett</author>
    <year>2012</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title>Harry Potter</title>
    <author>J. K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</books>
```

```

<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="title" type="xs:string"/>
  <xs:element name="author" type="xs:string"/>
  <xs:element name="year" type="xs:short"/>
  <xs:element name="price" type="xs:float"/>
  <xs:element name="book">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="title"/>
        <xs:element ref="author"/>
        <xs:element ref="year"/>
        <xs:element ref="price"/>
      </xs:sequence>
      <xs:attribute name="category" type="xs:string" use="optional"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="books">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="book" maxOccurs="unbounded" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

Altra strategia

```

<?xml version="1.0" encoding="UTF-8"?>
<books>
  <book category="fantasy">
    <title>The colour of magic</title>
    <author>Terry Pratchett</author>
    <year>2012</year>
    <price>30.00</price>
  </book>
  <book category="fantasy">
    <title>Guards! Guards!</title>
    <author>Terry Pratchett</author>
    <year>2012</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title>Harry Potter</title>
    <author>J. K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</books>

```

XSL

DR WIOLETA SZWOCH

DEPARTMENT OF INTELLIGENT INTERACTIVE SYSTEMS

XPath (*XML Path Language*)

standard for identifying parts of an XML document

An unambiguous description of the element's address in the XML file

Used in other standards, does not exist alone

Path expressions are used to navigate in XML documents

- syntax similar to Unix file system paths
- the ability to extract the necessary nodes

XPath contains a library of standard functions

standard per identificare parti di un documento XML

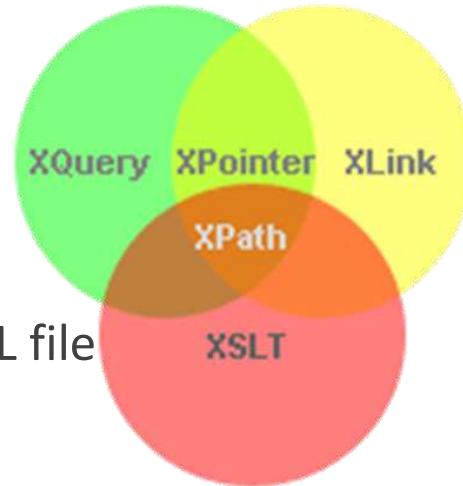
Una descrizione univoca dell'indirizzo dell'elemento nel file XML

Utilizzato in altri standard, non esiste da solo

Le espressioni di percorso vengono utilizzate per navigare nei documenti XML:

- sintassi simile ai percorsi del file system Unix
- la capacità di estrarre i nodi necessari

XPath contiene una libreria di funzioni standard



Nodes

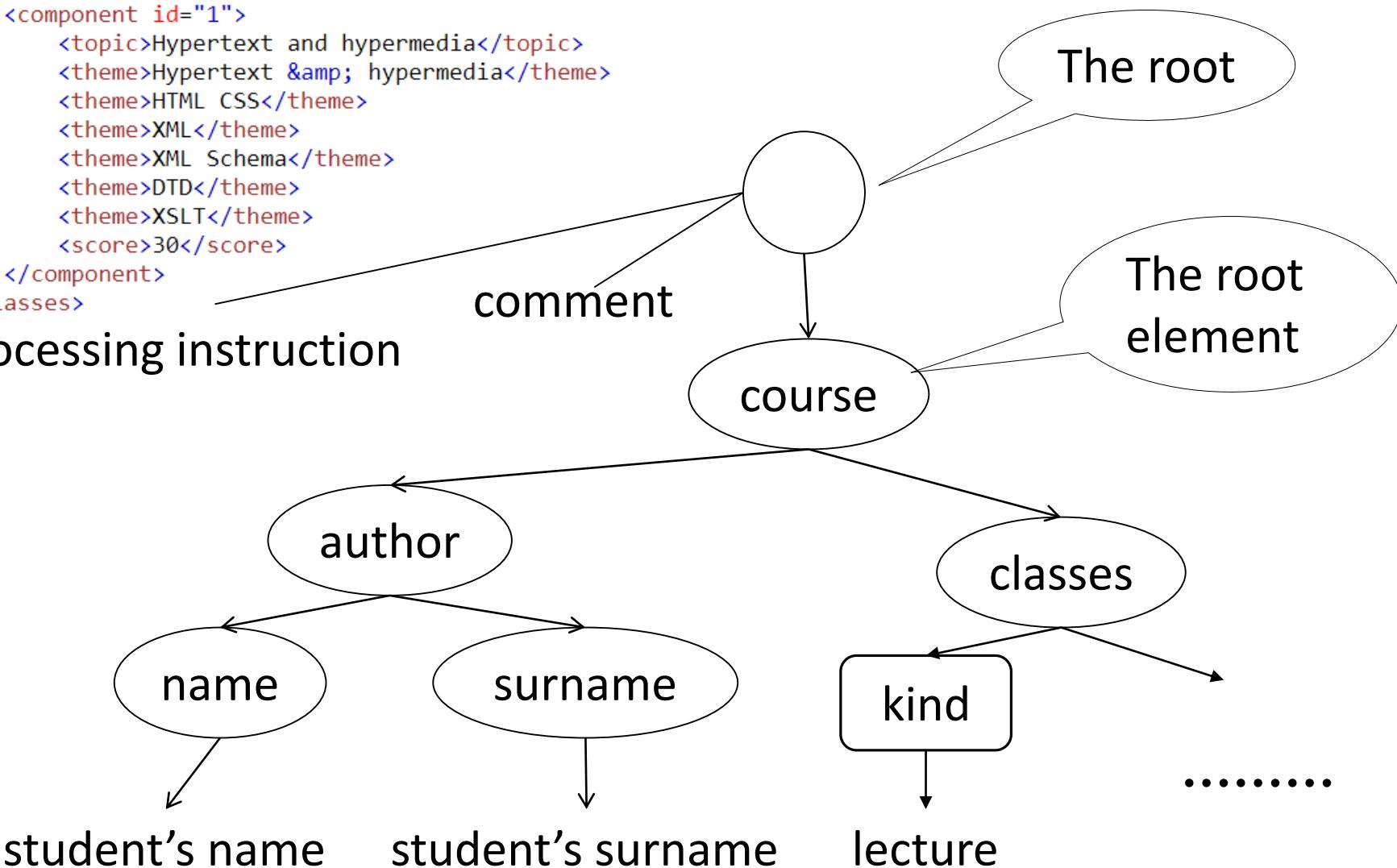
XML document

- Tree structure with nodes
- Each node represents part of XML document
 - Seven types
 - Root
 - Element
 - Attribute
 - Text
 - Comment
 - Processing instruction
 - Namespace

Data model for XPath

```
<course>
  <author>
    <name>student's name</name>
    <surname>student's surname</surname>
  </author>
  <classes kind="lecture" >
    <component id="1">
      <topic>Hypertext and hypermedia</topic>
      <theme>Hypertext & hypermedia</theme>
      <theme>HTML CSS</theme>
      <theme>XML</theme>
      <theme>XML Schema</theme>
      <theme>DTD</theme>
      <theme>XSLT</theme>
      <score>30</score>
    </component>
  </classes>
```

processing instruction



```
<course>
  <author>
    <name>student's name</name>
    <surname>student's surname</surname>
  </author>
  <classes kind="lecture" >
    <component id="1">
      <topic>Hypertext and hypermedia</topic>
      <theme>Hypertext & hypermedia</theme>
      <theme>HTML CSS</theme>
      <theme>XML</theme>
      <theme>XML Schema</theme>
      <theme>DTD</theme>
      <theme>XSLT</theme>
      <score>30</score>
    </component>
  </classes>
```

relative path

step/step/...

absolute path

/step/step/...

A step consists of:

an axis

a node-test

zero or more predicates

XPath is a syntax for "addressing" into a document.
They are "path expressions".
XPath expressions have a directory-path-like syntax.

Step – full syntax:

axis::node-test [predicate1] [predicate2] ...

axis – direction in document tree

node-test – selecting nodes by kind,
name, or type

predicates – (0 or more) additional logical
conditions for filtering

```
<course>
  <author>
    <name>student's name</name>
    <surname>student's surname</surname>
  </author>
  <classes kind="lecture" >
    <component id="1">
      <topic>Hypertext and hypermedia</topic>
      <theme>Hypertext & hypermedia</theme>
      <theme>HTML CSS</theme>
      <theme>XML</theme>
      <theme>XML Schema</theme>
      <theme>DTD</theme>
      <theme>XSLT</theme>
      <score>30</score>
    </component>
  </classes>
```

XPath is a syntax for "addressing" into a document.
They are "path expressions".
XPath expressions have a directory-path-like syntax.

"/" represents the Document info item (root)

* matches any element

author/*

@ means attribute

classes/@kind

```
<course>
  <author>
    <name>student's name</name>
    <surname>student's surname</surname>
  </author>
  <classes kind="lecture" >
    <component id="1">
      <topic>Hypertext and hypermedia</topic>
      <theme>Hypertext & hypermedia</theme>
      <theme>HTML CSS</theme>
      <theme>XML</theme>
      <theme>XML Schema</theme>
      <theme>DTD</theme>
      <theme>XSLT</theme>
      <score>30</score>
    </component>
  </classes>
```

classes//theme //theme

The result of evaluating an XPath expression is a Node Set

"///" matches elements that aren't direct children

XPath

XPath treat XML as a tree of elements

Root of tree – document node main element (aka document element) is not the root

„Leaf” may be:

tag, attribute, processing instruction, comment, text, namespace

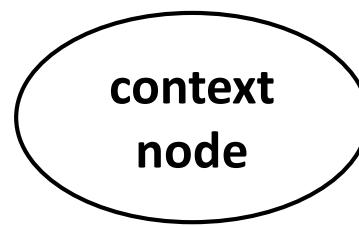
“leaves” are related by “branches” - axes

Nodes

Relations between nodes

child, parent, descendant , ancestor, sibling

```
<course>
  <author>
    <name>student's name</name>
    <surname>student's surname</surname>
  </author>
  <classes kind="lecture" >
    <component id="1">
      <topic>Hypertext and hypermedia</topic>
      <theme>Hypertext & hypermedia</theme>
      <theme>HTML CSS</theme>
      <theme>XML</theme>
      <theme>XML Schema</theme>
      <theme>DTD</theme>
      <theme>XSLT</theme>
      <score>30</score>
    </component>
  </classes>
```



You can address the context node as '.'

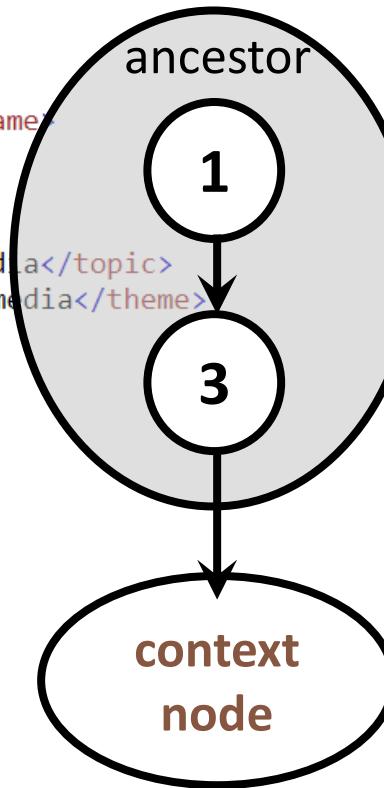
./@*

The context node is implicit.

range/component \equiv ./range/component

The context node does not have to be an element.

```
<course>
  <author>
    <name>student's name</name>
    <surname>student's surname</surname>
  </author>
  <classes kind="lecture" >
    <component id="1">
      <topic>Hypertext and hypermedia</topic>
      <theme>Hypertext & hypermedia</theme>
      <theme>HTML CSS</theme>
      <theme>XML</theme>
      <theme>XML Schema</theme>
      <theme>DTD</theme>
      <theme>XSLT</theme>
      <score>30</score>
    </component>
  </classes>
```



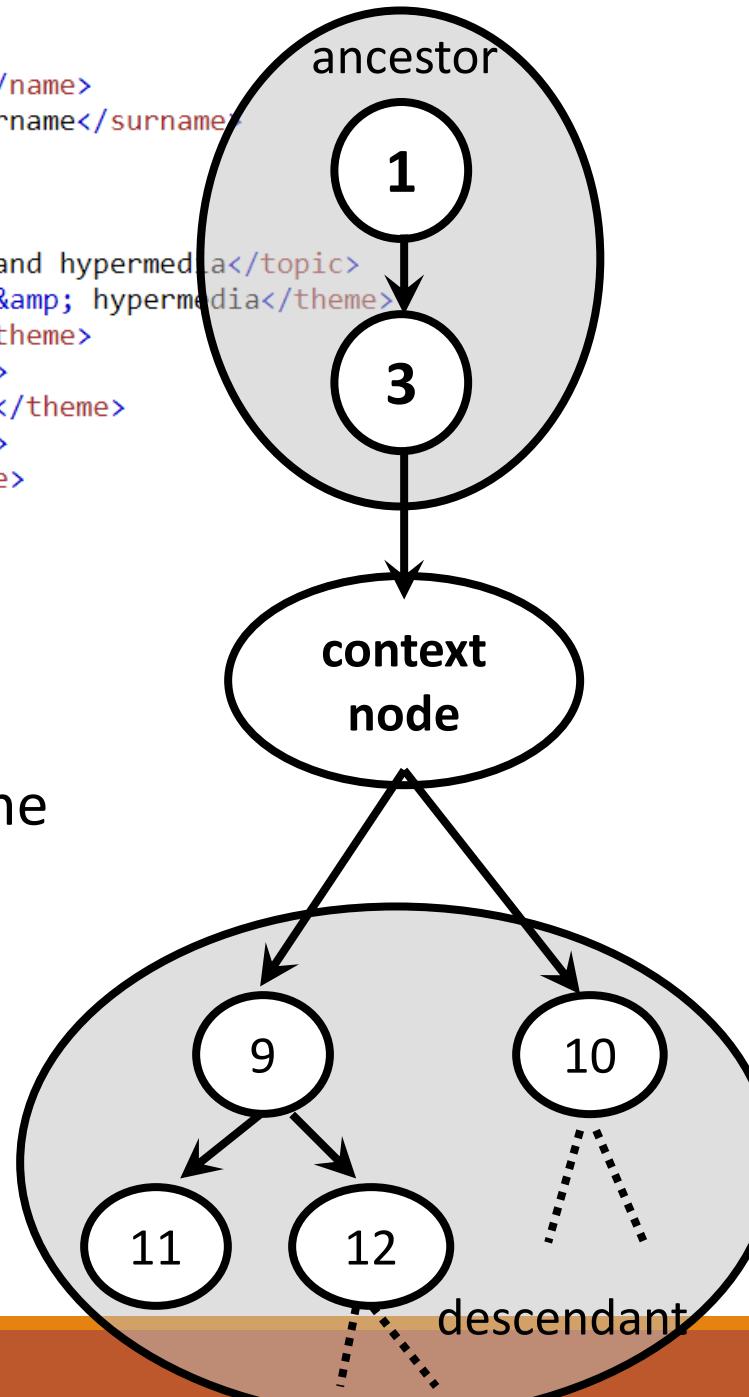
From the context node you can access your parent and ancestors.

'..' represents the parent.

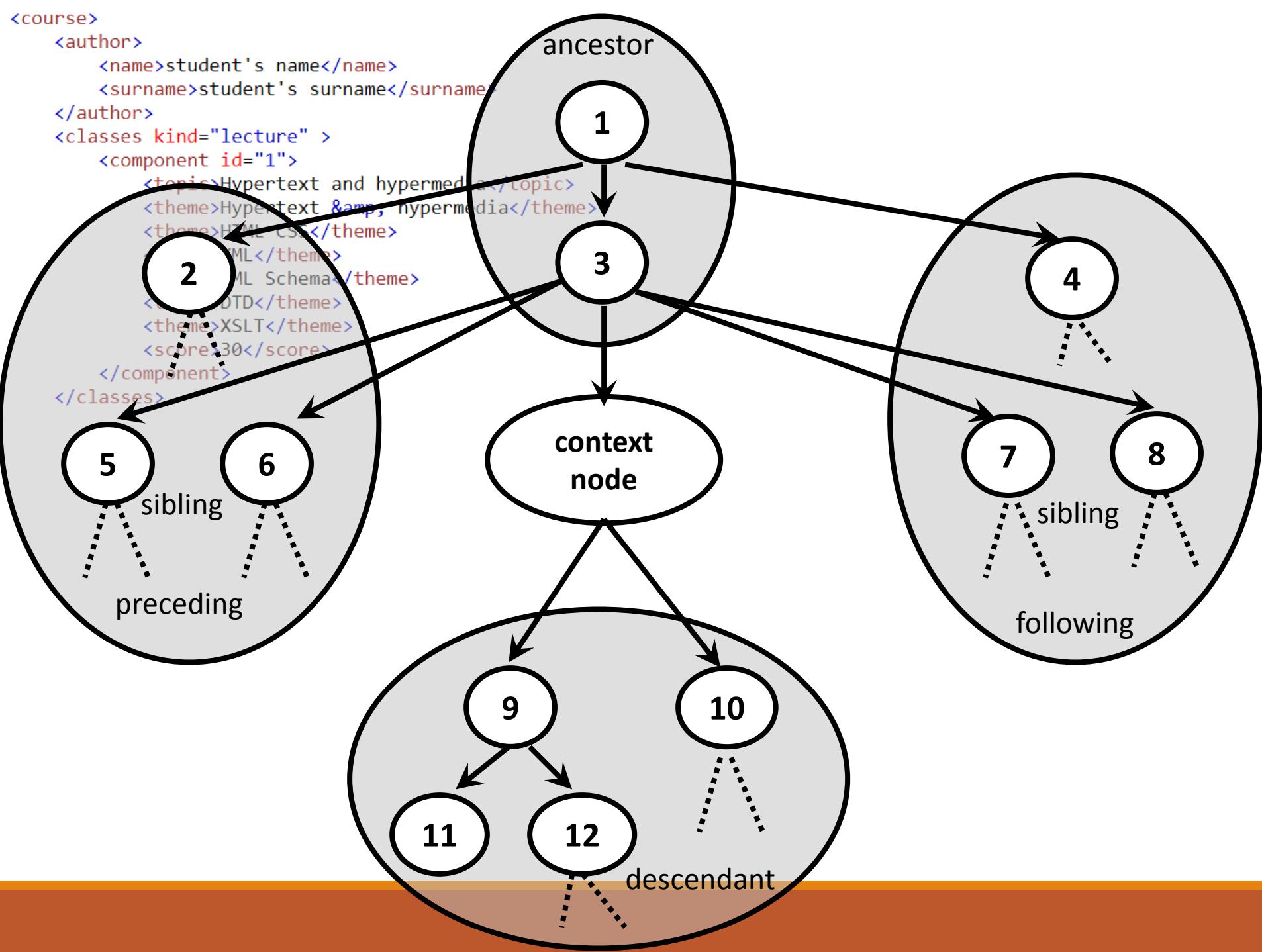
You can go back many levels: ../../classes

```
<course>
  <author>
    <name>student's name</name>
    <surname>student's surname</surname>
  </author>
  <classes kind="lecture" >
    <component id="1">
      <topic>Hypertext and hypermedia</topic>
      <theme>Hypertext & hypermedia</theme>
      <theme>HTML CSS</theme>
      <theme>XML</theme>
      <theme>XML Schema</theme>
      <theme>DTD</theme>
      <theme>XSLT</theme>
      <score>30</score>
    </component>
  </classes>
```

component/theme



```
<course>
  <author>
    <name>student's name</name>
    <surname>student's surname</surname>
  </author>
  <classes kind="lecture" >
    <component id="1">
      <topic>Hypertext and hypermedia</topic>
      <theme>Hypertext & hypermedia</theme>
      <theme>HTML/CSS</theme>
      <theme>XML</theme>
      <theme>DTD</theme>
      <theme>XSLT</theme>
      <score>30</score>
    </component>
  </classes>
```



Axis

self

child

descendant

parent

ancestor

following-sibling

preceding-sibling

following

preceding

attribute

namespace

descendant-or-self

ancestor-or-self

Node test

by kind of node

node()

text()

comment()

processing-instruction()

attribute()

by name

./classes/@kind

expands to

self::node()/

descendant-or-self::node()/ child::classes/

attribute::kind

Some built-in XPath functions

Text

- concat(s1, s2, ...) substring(s, pos, len)
starts-with(s1, s2) contains(s1, s2) string-length(s) translate(s, t1, t2)

Numbers

- floor(x) ceiling(x) round(x)

Nodes

- name(n?) local-name(n?) namespace-uri(n?)

Sequences

- count(S) sum(S) min(S) max(S) avg(S) empty(S) reverse(S) distinct-values(S)

Context

- current() position() last()

Operators

Arithmetic

+ - * div mod

Logical values

and or

Comparison operators

= != < <= > >=

Predicates

additional logical conditions for filtering

/path[predicate]

- they allow you to check properties that can not be expressed in the nodes themselves
- any XPath expression
- predicates may contain functions and operators
- only nodes for which the predicate is true are included in the result

```
<course>
  <author>
    <name>student's name</name>
    <surname>student's surname</surname>
  </author>
  <classes kind="lecture" >
    <component id="1">
      <topic>Hypertext and hypermedia</topic>
      <theme>Hypertext & hypermedia</theme>
      <theme>HTML CSS</theme>
      <theme>XML</theme>
      <theme>XML Schema</theme>
      <theme>DTD</theme>
      <theme>XSLT</theme>
      <score>30</score>
    </component>
  </classes>
```

component/theme[last()]

component/theme[position()<3]

/course/author/name

author | classes/component/topic

theme[3]

classes[@kind='lecture']

XSL (*eXtensible Stylesheet Language*)

SGML(1986) (→ XML)

DSSSL (→XSL)

- *Document Style and Semantics Specification Language*
- language for processing and transforming SGML documents into a form that can be displayed or printed
- **A HUGE MONSTER OF A LANGUAGE**

XSL (eXtensible Stylesheet Language)

XSLT (XSL Transformation)

A language that allows you to transform and display data from XML documents
What would have happened if there hadn't been an XSLT

XML document without XSL

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="D:\XSL\Transform.xsl"?>
<!DOCTYPE person [
  <!-- person element -->
  <!-- address element -->
  <!-- telephone element -->
  <!-- email element -->
  <!-- spouse element -->
  <!-- hobbies element -->
  <!-- residence element -->
  <!-- birthdate element -->
  <!-- deathdate element -->
]>
<person>
  <name>John Smith</name>
  <address>123 Main Street, Anytown, USA</address>
  <telephone>555-1234</telephone>
  <email>john.smith@example.com</email>
  <spouse>
    <name>Jane Doe</name>
    <address>123 Main Street, Anytown, USA</address>
    <telephone>555-1234</telephone>
    <email>jane.doe@example.com</email>
  </spouse>
  <hobbies>
    <hobby>Gardening</hobby>
    <hobby>Cooking</hobby>
  </hobbies>
  <residence>
    <street>123 Main Street</street>
    <city>Anytown</city>
    <state>USA</state>
    <zip>12345</zip>
  </residence>
  <birthdate>1970-01-01</birthdate>
  <deathdate>2050-12-31</deathdate>
</person>
```

XSL FO (XSL Formating Objects)

XML document with XSL

Kontrolne informacje o państwie:

| | |
|-----------------------------------|--------------------------|
| Nazwa państwa: | Typy: |
| Nazwa państwa o języku urzędowym: | Uzbecka Miejska-Arabia |
| Język urzędowy: | arabski |
| Język nar.: | + angielski |
| Ilość mieszkańców: | + francuski |
| Powierzchnia: | 883 000 km ² |
| Dane przyjazne: 2004-07-07 | Data odjazdu: 2004-07-15 |



Odwiedzone przez nasze rejsantów to Durgabatia.

Durgabatia leży nad laskowymi, kryształowo czystymi Morzem Czerwonym o lśniącym kolorze podwodne fale raf koralowych w odległości 500 km od Kairu i 270 km od Lukasu. Dookoła wspaniałego ujęcia znajdują się obiekty turystyczne wykorzystywane w Egipcie. Miasto rozciąga się na ok. 35 km długości, na której w większości znajdują się luksusowe hotele.

Treść tekstu:

Skorot jest zawsze dopuszczalny od kilometra lat. Aż trudno w to uwierzyć, ale do góry od lat siedemdziesiątych ta spokojna osada rybacka jest rozbudowywana i luzy obrzeże mieszkańców.

Kilometr:

Displaying XML using CSS

```
<?xml version="1.0" encoding="utf-8"?>
<?xmlstylesheet type="text/css" href="p1.css" ?>
<pajeczaki>
  <pajaki gatunek="tygrzyk" chroniony="tak" wyst_polska="tak">
    <nazwa jezyk="polaska">Tygrzyk Paskowany</nazwa>
    <nazwa jezyk="łacińska">Argiope bruennichi</nazwa>
    <opis czego_dotyczy="Cechy">
      Samica: jej głowotułów pokryty jest gęstym srebrzystym włosem
    </opis>
    <opis czego_dotyczy="Pokarm">
      Pająk ten jest bardzo "wybredny", gdyż poluje tylko na owady
    </opis>
  </pajaki>
  <skorpiony gatunek="polny" chroniony="tak" wyst_polska="nie">
    <nazwa jezyk="polaska">Skorpion Polny</nazwa>
    <nazwa jezyk="łacińska">Buthus Occitanus</nazwa>
    <opis czego_dotyczy="Cechy">
      Posiadając małe kleszcze i gruby ogon, ten bardzo jadowity g
    </opis>
    <opis czego_dotyczy="Pokarm">
      Osobnik należy do niezwykle szybkich i agresywnych skorpionów
    </opis>
  </skorpiony>
</pajeczaki>
```

```
pajeczaki
{
background-color: #ffffff;
width: 100%;
}
pajaki, skorpiony
{
display: block;
margin-bottom: 30pt;
margin-left: 0;
}
nazwa
{
display: block;
color: #FF0000;
font-size: 20pt;
}
opis
{
display: block;
color: #0000FF;
font-size: 14pt;
}
```

Displaying XML using CSS

The screenshot shows a browser window with the following content:

```
<?xml version="1.0" encoding="utf-8"?>
<?xmlstylesheet type="text/css" href="p1.css" ?>


file:///I:/PG/HiH/Przykłady/xslt/Pajaki_1.xml


```

Tygrzyk Paskowany *Argiope bruennichi*

Samica: jej głowotułów pokryty jest gęstym srebrzystym włosem, a na jaskrawożółtym odwłoku posiada długie, czerwone paski. Samiec: jest bardzo niepozorny, jego srebrzystoszary odwłok urozmaica jedynie szara podłużna łata na środku. Pająk ten jest bardzo "wybredny", gdyż poluje tylko na owady z rzędu prostoskrzydłych i ważek, których

Skorpion Polny *Buthus Occitanus*

Posiadając małe kleszcze i grubego ogona, ten bardzo jadowity gatunek jest typowym przedstawicielem radykalnej odmiany tego gatunku, charakteryzującej się bardziej brązowym zabarwieniem ciała. Natomiast odmiana ta jest bardziej spłaszciona i ma mniejsze kolce na ogonie. Dymorfizm płciowy jest słabo rozróżnialny u tych osobników. Liczba "zębów" na samca są dłuższe.

Osobnik należy do niezwykle szybkich i agresywnych skorpionów, będąc dokonalem myśliwym. Żywi się głównie innymi skorpionami, ale również innymi zwierzętami.

XSLT

Extensible Stylesheet Language Transformations

CSS – Cascading Style Sheet

- style sheet determines the style or the appearance of tags (HTML or XML)
- we can define fonts, margins, colours, borders, ...

XSLT stylesheet

- a document that generates data based on XML document, the resulting document may or may not contain formatting information
- a complete high-level language for manipulating an XML documents
- it does not replace existing programming languages but complements them

Advantages

- Independent of programming. Transformations are written in a separate XSL file which is an XML document.
- Output can be changed by simply modifying the transformations in an XSL file. No need to change any code. So Web designers can edit the stylesheet and can see the change in the output quickly.

XSLT

Extensible Stylesheet Language Transformations

the language of transforming XML trees

the basic processing paradigm is pattern matching

declarative language

data-driven language (push processing model)

- the code is executed in response to the data;
- the code is executed nondeterministically

XSLT

importance XSLT

- allows you to work with XML documents
 - we do not have to write programs to process XML
- different document type from XML
(HTML, ...)
- the same transformation can be applied to many XML documents
- different XSLT can be applied to the same XML document

Example

```
<?xml version="1.0" encoding="iso-8859-2"?>
<!--Temat: Drzewo Genealogiczne (XML + XSD)-->
<?xml-stylesheet type="text/xsl" href="bk-drzewo.xsl"?>
<drzewo xmlns:xsi="http://www.w3.org/2001/XMLSchema-in
    <osoba plec="kobieta">
        <imie>Halina</imie>
        <nazwisko>Abecka</nazwisko>
        <rodowe><></rodowe>
        <data>1940-08-14</data>
        <miejsce>Reda</miejsce>
        <informacje>Babcia Halina przez wiekszo
        <partner>
            <osoba plec="mezczyzna">
                <imie>Wiktor</imie>
                <nazwisko>Abecki</nazwisko>
                <data>1937-07-03</data>
                <miejsce>Zakopane</miejsce>
                <informacje>Dziadka Wi
            </osoba>
        </partner>
    <dzieci>
```

| | |
|---|--|
|  | Halina Abecka |
| | Data urodzenia: 1940-08-14 |
| | Miejsce urodzenia: Reda |
| | Informacje: Babcia Halina przez wiekszo |
|  | Ewa Babecka |
| | Data urodzenia: 1961-10-10 |
| | Miejsce urodzenia: Wejherowo |
| | Informacje: Moja mama, dzwoni do mnie raz w tygodniu. |
|  | Jola Cebcka |
| | Data urodzenia: 1964-09-21 |
| | Miejsce urodzenia: Wejherowo |
| | Informacje: Ciocia Jola to najbardziej wykrecona osoba z rodziny. |
|  | Katarzyna Ebecka |
| | Data urodzenia: 1962-07-07 |
| | Miejsce urodzenia: Wejherowo |
| | Informacje: Ciocia Kasia mieszka niedaleko ode mnie i często do nas wpada. |

| | |
|---|---|
|  | Halina Abecka |
| | Nazwisko rodowe: XXXX |
| | Data urodzenia: 1940-08-14 |
| | Miejsce urodzenia: Reda |
| | Informacje: Babcia Halina przez wiekszo |
|  | Wiktor Abecki |
| | Data urodzenia: 1937-07-03 |
| | Miejsce urodzenia: Zakopane |
| | Informacje: Dziadka Wiktor juz nie ma na tym swiecie. |
|  | Ewa Babecka |
| | Nazwisko rodowe: Abecka |
| | Data urodzenia: 1961-10-10 |
| | Miejsce urodzenia: Ostroda |
| | Informacje: Moja mama, dzwoni do mnie raz w tygodniu. |
|  | Edward Babecki |
| | Data urodzenia: 1960-03-15 |
| | Miejsce urodzenia: Gdynia |
| | Informacje: Moj Tata-sponsor, wymienia ze mna jakies 30 |
| | pełnych zdan w ciągu roku. |
|  | Jan Szymon Babecki |
| | Data urodzenia: 1982-02-15 |
| | Miejsce urodzenia: Ostroda |
| | Informacje: |
|  | Maciej Mikolaj Babecki |
| | Data urodzenia: 1984-10-03 |
| | Miejsce urodzenia: Ostroda |
| | Informacje: Brachol Maciek to zapalony hiphopowy DJ, w |
| | wolnym czasie studiuje na PG. |



Halina Abecka

Nazwisko rodowe: XXXX
 Data urodzenia: 1940-08-14
 Miejsce urodzenia: Reda
 Informacje: Babcia Halina przez wiekszosc czasu odpoczywa.



Wiktor Abecki

Data urodzenia: 1937-07-03
 Miejsce urodzenia: Zakopane
 Informacje: Dziadka Wiktora juz nie ma na tym swiecie.



Ewa Babecka

Nazwisko rodowe: Abecka
 Data urodzenia: 1961-10-10
 Miejsce urodzenia: Ostroda
 Informacje: Moja mama, dzwoni do mnie raz w tygodniu.



Edward Babecki

Data urodzenia: 1960-03-15
 Miejsce urodzenia: Gdynia
 Informacje: Moj Tata-sponsor, wymienia ze mna jakies 30 pełnych zdan w ciągu roku.



Jan Szymon Babecki

Data urodzenia: 1982-02-15
 Miejsce urodzenia: Ostroda
 Informacje:



Maciej Mikolaj Babecki

Data urodzenia: 1984-10-03
 Miejsce urodzenia: Ostroda
 Informacje: Brachol Maciek to zapalony hiphopowy DJ, w wolnym czasie studiuje na PG.

XML

```
<?xml version="1.0" encoding="iso-8859-2"?>
<!--Temat: Drzewo Genealogiczne (XML + XSD)--&gt;
&lt;?xmlstylesheet type="text/xsl" href="bk-drzewo.xsl"?&gt;
&lt;drzewo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="bk-drzewo.xsd"&gt;
  &lt;osoba plec="kobieta"&gt;
    &lt;imie&gt;Halina&lt;/imie&gt;
    &lt;nazwisko&gt;Abecka&lt;/nazwisko&gt;
    &lt;rodowe&gt;&lt;!--&gt;&lt;/rodowe&gt;
    &lt;data&gt;1940-08-14&lt;/data&gt;
    &lt;miejsce&gt;Reda&lt;/miejsce&gt;
    &lt;informacje&gt;Babcia Halina przez wiekszosc czasu odpoczywa.&lt;/informacje&gt;
    &lt;partner&gt;
      &lt;osoba plec="mezczyzna"&gt;
        &lt;imie&gt;Wiktor&lt;/imie&gt;
        &lt;nazwisko&gt;Abecki&lt;/nazwisko&gt;
        &lt;data&gt;1937-07-03&lt;/data&gt;
        &lt;miejsce&gt;Zakopane&lt;/miejsce&gt;
        &lt;informacje&gt;Dziadka Wiktora juz nie ma na tym swiecie.&lt;/informacje&gt;
      &lt;/osoba&gt;
    &lt;/partner&gt;
  &lt;/osoba&gt;
  &lt;osoba plec="kobieta"&gt;
    &lt;imie&gt;Ewa&lt;/imie&gt;
    &lt;nazwisko&gt;Babecka&lt;/nazwisko&gt;
    &lt;rodowe&gt;Abecka&lt;/rodowe&gt;
    &lt;data&gt;1961-10-10&lt;/data&gt;
    &lt;miejsce&gt;Ostroda&lt;/miejsce&gt;
    &lt;informacje&gt;Moja mama, dzwoni do mnie raz w tygodniu.&lt;/informacje&gt;
  &lt;/osoba&gt;
&lt;/drzewo&gt;</pre>

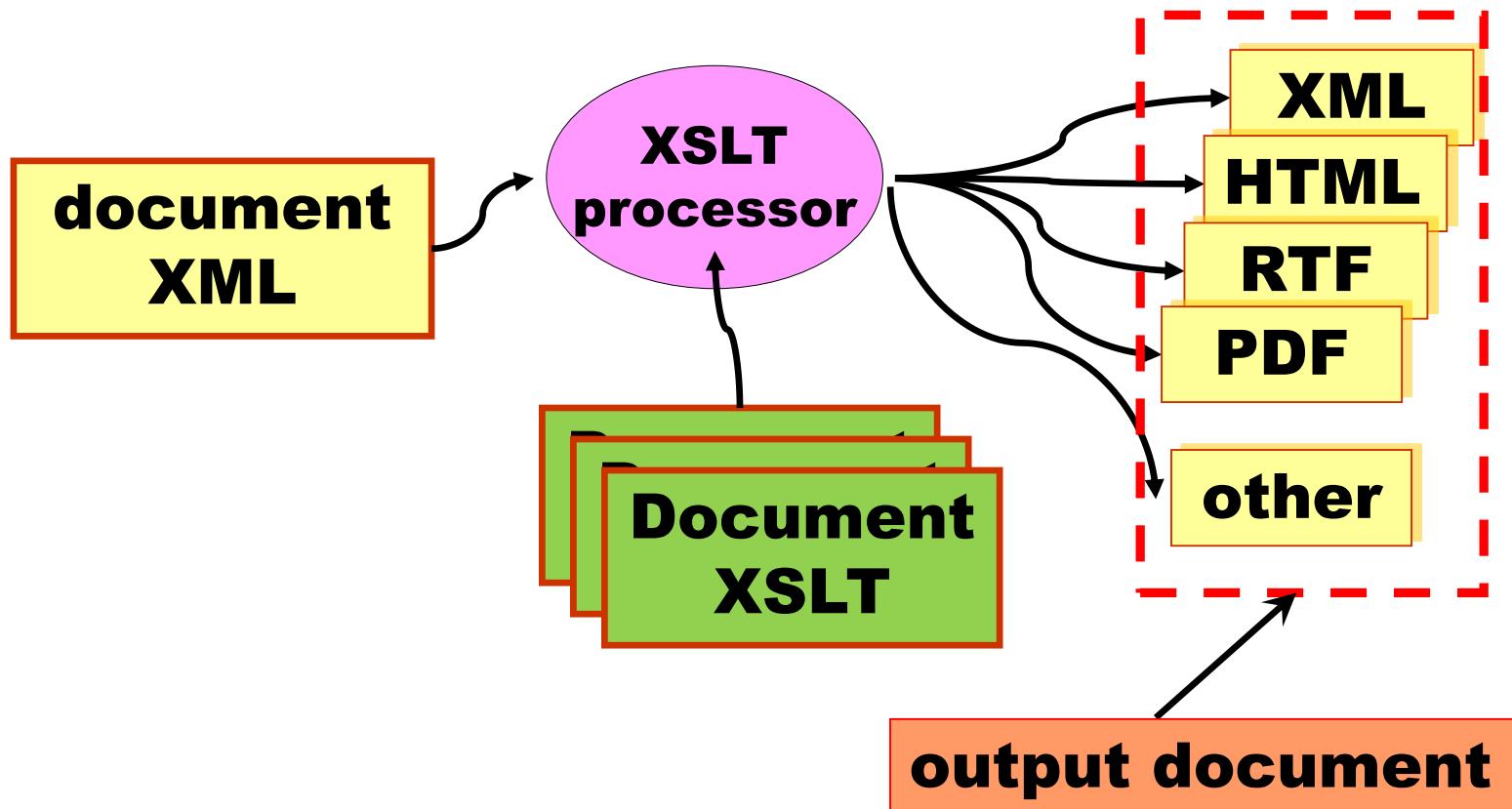
```

XSLT

```
<?xml version="1.0" encoding="iso-8859-2"?>
<!--Temat: Drzewo Genealogiczne (XML + XSD)--&gt;
&lt;?xmlstylesheet type="text/xsl" href="bk-drzewo.xsl"?&gt;
&lt;drzewo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="bk-drzewo.xsd"&gt;
  &lt;osoba plec="kobieta"&gt;
    &lt;imie&gt;Anna&lt;/imie&gt;
    &lt;nazwisko&gt;Nowak&lt;/nazwisko&gt;
    &lt;rodowe&gt;&lt;!--&gt;&lt;/rodowe&gt;
    &lt;data&gt;1940-08-14&lt;/data&gt;
    &lt;miejsce&gt;Reda&lt;/miejsce&gt;
    &lt;informacje&gt;Babcia&lt;/informacje&gt;
    &lt;partner&gt;
      &lt;osoba plec="mezczyzna"&gt;
        &lt;imie&gt;Andrzej&lt;/imie&gt;
        &lt;nazwisko&gt;Nowak&lt;/nazwisko&gt;
        &lt;data&gt;1940-07-03&lt;/data&gt;
        &lt;miejsce&gt;Torun&lt;/miejsce&gt;
        &lt;informacje&gt;Dziadek&lt;/informacje&gt;
      &lt;/osoba&gt;
    &lt;/partner&gt;
  &lt;/osoba&gt;
  &lt;dziedzic1&gt;
    &lt;osoba plec="kobieta"&gt;
      &lt;imie&gt;Katarzyna&lt;/imie&gt;
      &lt;nazwisko&gt;Lusnia&lt;/nazwisko&gt;
      &lt;rodowe&gt;Nowak&lt;/rodowe&gt;
      &lt;data&gt;1963-10-10&lt;/data&gt;
      &lt;miejsce&gt;Kalisz&lt;/miejsce&gt;
      &lt;informacje&gt;Mama&lt;/informacje&gt;
      &lt;partner&gt;
        &lt;osoba plec="mezczyzna"&gt;
          &lt;imie&gt;Jan&lt;/imie&gt;
          &lt;nazwisko&gt;Lusnia&lt;/nazwisko&gt;
          &lt;data&gt;1960-03-15&lt;/data&gt;
          &lt;miejsce&gt;Gdynia&lt;/miejsce&gt;
          &lt;informacje&gt;Tata&lt;/informacje&gt;
        &lt;/osoba&gt;
      &lt;/partner&gt;
    &lt;/osoba&gt;
  &lt;/dziedzic1&gt;
  &lt;osoba plec="mezczyzna"&gt;
    &lt;imie&gt;Edward&lt;/imie&gt;
    &lt;nazwisko&gt;Szymon&lt;/nazwisko&gt;
    &lt;rodowe&gt;&lt;!--&gt;&lt;/rodowe&gt;
    &lt;data&gt;1982-02-15&lt;/data&gt;
    &lt;miejsce&gt;Ostroda&lt;/miejsce&gt;
    &lt;informacje&gt;&lt;/informacje&gt;
  &lt;/osoba&gt;
  &lt;osoba plec="kobieta"&gt;
    &lt;imie&gt;Hanna&lt;/imie&gt;
    &lt;nazwisko&gt;Lusnia&lt;/nazwisko&gt;
    &lt;data&gt;1995-05-29&lt;/data&gt;
    &lt;miejsce&gt;Ostroda&lt;/miejsce&gt;
    &lt;informacje&gt;Siostra&lt;/informacje&gt;
  &lt;/osoba&gt;
&lt;/drzewo&gt;</pre>

```

XSLT



XSLT

From a single XML file, we are able to generate:

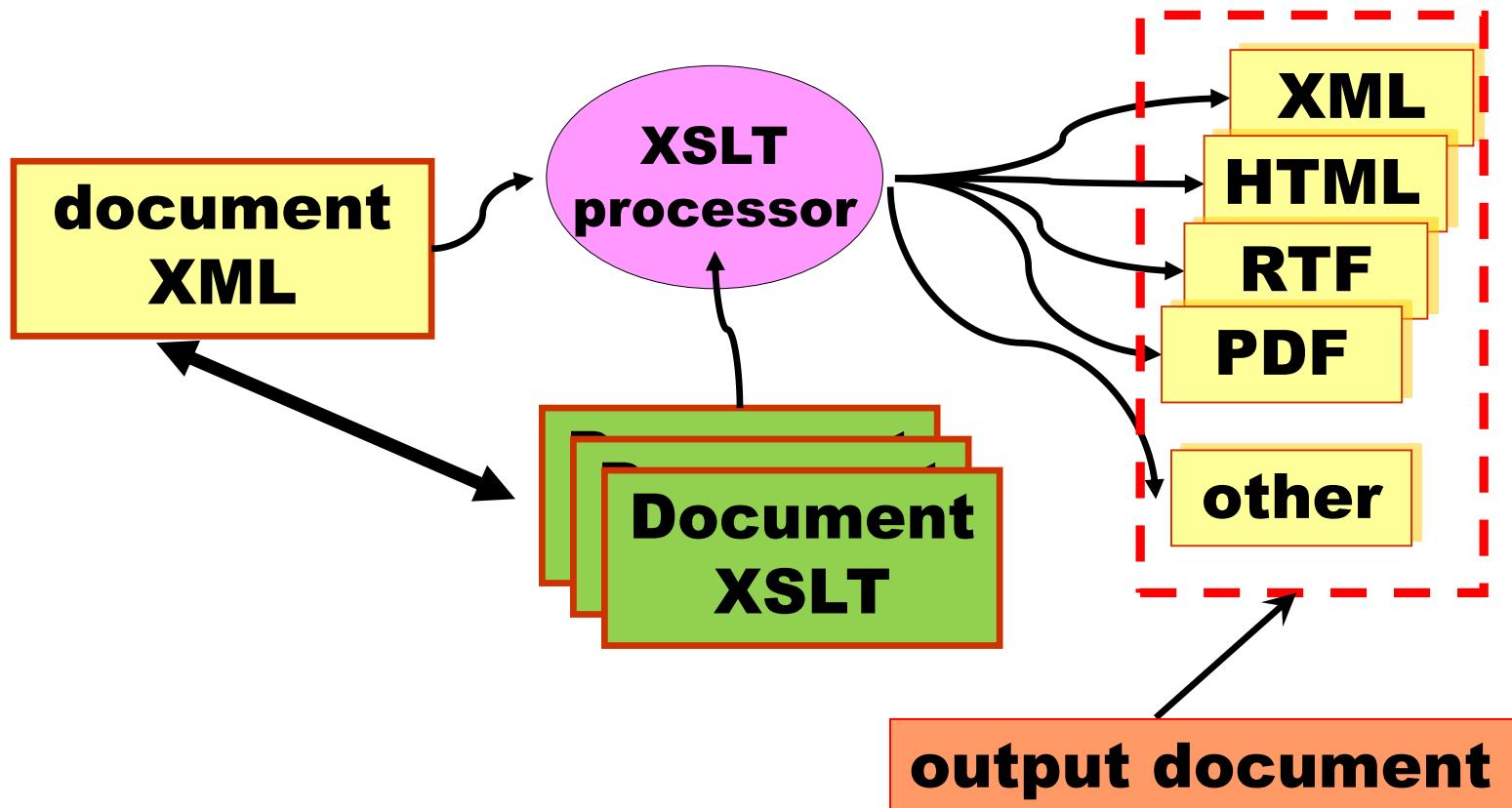
- an HTML webpage, to publish the data on the web
- a CSV file, to easily import into a database/spreadsheet
- a PDF file for printing
- another XML file, with different tag names or layout
- other text files

XSLT

XSLT processors

- XT (*James Clark*)
 - One of the first
 - Created in Java
- Xalan (*Apache*)
 - Easy to use in Java and C++ environments
- Msxml (*Microsoft*)
 - IE
- Saxon
- LotusXSL, Koala XSL Engine, iXSLT, XT, ...

XSLT



XSLT

command line (eg Xalan)

- `java org.apache.xalan.xslt.Process -IN file.xml -XSL file.xsl -OUT file.html`

in XML file

- `<?xmlstylesheet type= " text/xsl" href= " sheet.xsl"?>`

Stylesheet processing

XSLT processor get a document and a stylesheet

It starts a (breadth-first) traversal at the **root node** and checks to see if there is a template match

- If so, it applies the template and looks for an “xsl:apply-templates” element
 - If such an element exists, it continues the traversal
 - if no such element exists, the traversal stops
- If not, it traverses down the tree looking for a template match with some other node of the tree

Structure of the XSLT document

XML declaration

xsl:stylesheet tag

- xmlns:xsl attribute: URI for XSLT namespace
 - xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
- version

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="1.0">
    ...templates and transformation rules go here ...
</xsl:stylesheet>
```

Structure of the XSLT document

xsl:attribute-set

it defines a set of attributes

xsl:decimal-format

defines how the numbers are displayed

xsl:import

import style sheets

xsl:include

include style sheets

xsl:key

key

xsl:namespace-alias

indicates the namespace in the resulting document

xsl:output

determines the type of result document

xsl:param

allows you to create parameters

xsl:preserve-space

retains white characters in the specified element

xsl:strip-space

removes whitespace from the specified element

xsl:template

defines the template

xsl:variable

defines variables

Structure of the XSLT document

xsl:apply-imports

xsl: apply-templates

xsl: attribute

xsl:call-template

xsl:choose

xsl:comment

xsl:copy

xsl:copy-of

xsl:element

xsl:fallback

xsl:for-each

xsl:if

xsl:message

xsl: number

xsl: otherwise

xsl:processing-instruction

xsl:sort

xsl:text

xsl:value-of

xsl:when

xsl:with-param

Structure of the XSLT document

hello.xsl

```
<?xml version="1.0"?>
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="text"/>
  <xsl:template match="/">
    <b><xsl:value-of
      select="hi"/></b>
  </xsl:template>
</xsl:stylesheet>
```

hello.xml

```
<?xml version="1.0"?>
<hi>Hello XSLT!</hi>
```

Hello XSLT!

The format of the output document

xsl:output

attributes

- *method* defines the output format
 - html, xml, text
- *encoding, version, ...*

default method: xml

XSLT extraction of information

xsl:value-of

- ***select***
 - determines the node from which we want to get the information
 - the attribute contains an XPath expression or XSLT function

xsl:value-of - only **first node** from set

```
<xsl:value-of select="node"/>
```

```
<?xml version="1.0" encoding="iso-8859-2"?>
<ptaki xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="ptaki.xsd">
  - <cptak numer="1">
    - <nazwa>
      <polska>Zimorodek</polska>
      <angielska>Common Kingfisher</angielska>
    </nazwa>
    <gromada>ptaki</gromada>
    - <wystepowanie>
      <region_swiata>Południowa  
Malajska.</region_swiata>
      <środowisko_zycia>Iądowe</środowisko_zycia>
      <Polska>tak</Polska>
    </wystepowanie>
    - <opis>
      <cechy_charakterystyczne>B  
polyskiem, srodek grzbietu i sterowki bletkite, gardlo i policzki biale, ciemne lotki, niebiesko-zielone ze  
choragiewki i cynamonowy spod ciala. Ma nieproporcjonalnie dlugi i mocny dzioeb oraz bardzo krotkie n  
Lata szybkim, prostoliniowym lotem, zwykle nisko nad woda, odzywajac sie wysokim przenikliwym  
glosem.</cechy_charakterystyczne>
      <pozywienie>Drobne rybki, ktore zimorodek zdobywa na wodzie</pozywienie>
    - <wymiary>
      <dlugosc_ciala>17cm</dlugosc_ciala>
      <rozpietosc_skrzydele>28cm</rozpietosc_skrzydele>
      <waga>56g</waga>
    </wymiary>
  </opis>
  <chroniony>tak</chroniony>
  - <galeria>
    <zdjecie numer="1">img/zimorodek_1.jpg</zdjecie>
    <zdjecie numer="2">img/zimorodek_2.jpg</zdjecie>
    <zdjecie numer="3">img/zimorodek_3.jpg</zdjecie>
  </galeria>
```

Zdjęcia



2018-10-12

```
<h3>Zdjęcia</h3>
<div id="zdjęcia">
  <xsl:for-each select="galeria/zdjęcie">
    <img>
      <xsl:attribute name="src" />
      <xsl:value-of select=". />
    </xsl:attribute>
  </img>
  </xsl:for-each>
</div>
<p>
  <br />
  <h4>
    <xsl:value-of select="data_modyfikacji"/>
  </h4>
</p>
<br />
</xsl:template>
```

XSLT - templates

xsl:template

A template contains rules to apply when a specified node is matched.

The template can be called by matching the node of the XML document tree to the *match* attribute

Nodes are matched and not selected, they are processed as they are encountered by the XML processor

XSLT - templates

xsl:template

- o match
- o mode
- o name

```
<xsl:template match= "XPath-expression">  
...  
</xsl:template>
```

ogólny format szablonu

```
<?xml version="1.0"?>  
<R>  
  <A>1</A>  
  <B>2</B>  
  <A>3</A>  
</R>
```

```
<xsl:template match="R">  
...  
</xsl:template>
```

```
<xsl:template match="A">  
...  
</xsl:template>
```

```
<xsl:template match="B">  
...  
</xsl:template>
```

XSLT - templates

xsl:template

- match
- mode
- name

using templates

- xsl:apply-templates
- xsl:call-template

templates must not be nested

XSLT - templates

the template element must have one of the attributes:

- **match**
 - defines the transformation for the nodes described by this attribute

`<xsl:template match="...">>...</xsl:template>`

- **name**
 - gives the template a name

`<xsl:template name="...">>...</xsl:template>`

XSLT - templates

template with match attribute

- template definition:

```
<xsl:template match="node">...</xsl:template>
```

- using template:

- all of the children of the current node

```
<xsl:apply-templates/>
```

- selecting the nodes to which the template is used

```
<xsl:apply-templates select="..." />
```

```
<?xml version="1.0"?>
<R>
  <A>1</A>
  <B>2</B>
  <A>3</A>
</R>
```

XSLT - templates

template with name attribute

- template definition:

```
<xsl:template name="dosomething">...</xsl:template>
```

- using template :

```
<xsl:call-template name=" dosomething "/>
```

looks like a traditional programming language

we have full control over the way of execution the code

XSLT - templates

```
...
<xsl:template match="mammals">
<h1><xsl:apply-templates/></h1>
</xsl:template>
```

```
...
<xsl:template match="birds">
<h1><xsl:apply-templates/></h1>
</xsl:template>
```

```
...
```

```
...
<xsl:template match="mammals | birds">
<h1><xsl:apply-templates/></h1>
</xsl:template>
```

```
...
```

XSLT - templates

If no rule matches the node?

built-in template

- for document root and elements:
 - apply templates to children
- for text nodes and attributes:
 - copy text value to result
- for comments and processing instructions
 - do nothing

XSLT - templates

template with mode

- possibility to process the same set of nodes many times

XSLT - templates

```
<xsl:template match="root">
    <xsl:apply-templates mode="tryb1"/>
    <xsl:apply-templates mode="tryb2"/>
</xsl:template>

<xsl:template match="elem" mode="tryb1">
    <xsl:text>tryb1:</xsl:text>
    <xsl:value-of select=".."/>
</xsl:template>

<xsl:template match="elem" mode="tryb2">
    <xsl:text>tryb2:</xsl:text>
    <xsl:value-of select=".."/>
</xsl:template>
```

Resolving matching rules conflicts

Which template will be used?

there is no template for the given context

- built-in template will be used

there is one template for the given context

- this template will be used

there are 2 templates for the given context

- a more specific template will be applied

there are 2 templates for the given context, both equally specific

- a template with higher priority will be applied

there are 2 templates: the same context, priority, both equally specific

- a template that appears later will be applied

XSLT

The XSLT language allows you to

- conditional processing,
- looping
- parameterisation
 - data from outside the xml and xsIt file can affect the result

XSLT - conditional processing

xsl:if

- **test**

no else

```
<xsl:if test="gender='woman'>  
    <font class="napis">Wife</font>  
</xsl:if>  
  
<xsl:if test="gender='man'>  
    <font class="napis">Husband</font>  
</xsl:if>
```

```

<xsl:if test="$srodomiesko='wodne' ">
    <xsl:attribute name="style">
        background:url(img/tab/morze.jpg) center no-repeat;
    </xsl:attribute>
    wodne
</xsl:if>
<xsl:if test="$srodomiesko='ladowe' ">
    <xsl:attribute name="style">
        background:url(img/tab/ziemia.jpg) center no-repeat;
    </xsl:attribute>
    ladowe
</xsl:if>

```

| Masy | Gromada | Czy lata? | Srodowisko życia | Region |
|-------|---|--|---|---------------|
| masa | | | | |
| 0.3kg |  |  |  | Różnie |
| 20kg |  |  |  | Mauritius |
| 250kg |  |  |  | Nowa Zelandia |

XSLT

xsl:choose : conditional processing

- xsl:when : determines the condition
 - test
- xsl:otherwise : optional, final instructions, used if no condition satisfied
- only first branch with satisfied condition processed.

```
<xsl:choose>
    <xsl:when test="@gender='woman' ">
        
    </xsl:when>
    <xsl:when test="@ gender='man' ">
        
    </xsl:when>
</xsl:choose>
```

XSLT - conditional processing

```
<xsl:choose>
    <xsl:when test="position()=first()">
        Do something for first element
    </xsl:when>
    <xsl:when test="position()=last()">
        Do something for last element
    </xsl:when>
    <xsl:otherwise>
        Do something for other elements
    </xsl:otherwise>
</xsl:choose>
```

XSLT - loop

xsl:for-each

- select

the XSLT processor processes all nodes corresponding to the pattern given in the *select* attribute

possibility of sorting - xsl:sort

```
<xsl:for-each select="XPath expression">  
    some instructions  
</xsl:for-each>
```

Sorting

xsl:sort

attributes

- **select**
 - sorting by element or attribute
- **order**
 - *ascending, descending*
- **case-order**
 - specifies letter size priority, *upper-first, lower-first*
- **lang**
- **data-type**
 - sorting letters or numbers (*text, number*)

possibility of multiple sorting criteria

Sorting

xsl:sort

- xsl:for-each
- xsl:apply-templates

xsl:sort the first instruction in for-each

```
<xsl:apply-templates>
    <xsl:sort select="wezel"/>
</xsl:apply-templates>
```

```
<xsl:for-each select="osoba">  
    <xsl:sort select="@plec"/>  
    <xsl:sort select="nazwisko"/>  
    <xsl:sort select="imie"/>  
    <center>  
        <xsl:choose>  
            <xsl:when test="@plec='kobieta'">  
                <table class="kobieta" align="center"> <xsl:call-template name="tabelka"/>  
            </table>  
        </xsl:when>  
        <xsl:when test="@plec='mezczyzna'">  
            <table class="mezczyzna" align="center"> <xsl:call-template name="tabelka"/>  
        </table>  
        </xsl:when>  
    </xsl:choose>  
    </center>  
</xsl:for-each>
```

Numbering

xsl:number

it can be located anywhere in the templates also in for-each element

defining the form of the number - attribute *format*

- 1 1,2,3...
- 01 01,02,03,...
- a a,b,c,...z,aa...
- B A,B,C...
- i i,ii,iii,iv,...
-

Numbering

level attribute

- = "any"
 - continuous numbering of elements regardless of their parent element
- = "multiple"
 - multi-level numbering (eg 1.2; 3.2.4, ...)

the possibility of grouping numbering

- grouping-separator , grouping-size
 - grouping-separator="." , grouping-size="3"
 - 12345678 -> 12.345.678

```
<xsl:template match="odsylacze">
<h3>Ciekawe linki:</h3>
    <xsl:for-each select="odsylacz">
        <xsl:number value="position()" format="I"/>
        <a class="obrazek">
            <xsl:attribute name="href"><xsl:value-of select="@href"/></xsl:attribute>
            <xsl:attribute name="title"><xsl:value-of select="@alt"/></xsl:attribute>
            <xsl:text> </xsl:text><xsl:value-of select=". "/><br/>
        </a>
    </xsl:for-each>
</xsl:template>
```

Opis motylka:

Przednie skrzydło czarne z ukośnymi, białożółtymi paskami. Tylne skrzydło pomarańczowoczerwone z czarnymi plamkami. Tylne skrzydło pomarańczowoczerwony z rzędem drobnych, czarnych plamek na stronie grzbietowej. Motyle latają głównie w czasie jesiennym na różnych roślinach zielnych takich jak jasnota, pokrzywa. Po przezimowaniu żerują na leszczynie, malinie, wiciokrzewie. Samice składają jaja w małych złożach na krawędzi liści roślin żywicielskich.



Ciekawe linki:

- I Moths and Butterflies of Europe
- II UK Moths
- III Schmetterlingsseiten von Jürgen Rodeland
- IV Insekten Box

Variables

xsl:variable

definition

- <xsl:variable name="VarName">value</xsl:variable>
- <xsl:variable name="VarName" select ="'value'"/>

reference to variable

- <xsl:value-of select="\$VarName"/>

Variables

local

```
<xsl:template name="...">  
    <xsl:variable name="...">...</xsl:variable>  
</xsl:template>
```

global

```
<xsl:stylesheet ...>  
    <xsl:variable name="...">...</xsl:variable>  
    ...  
</xsl:stylesheet >
```

Variables??

Constant (*name given to a certain value*)

their values can not be modified (*read only*)

advantages of variables

- they make reading the code easier
 - complex expression saved as a variable
 - the ability to break complex expressions into parts
- reuse
 - performance enhancement especially for complex expressions that result in a fragment of the tree
- saving the value of nodes that are currently unavailable

Variables

simple

- they contain single values
- usually used to insert the same values in many places in a document

complex

- they contain sets of nodes and tree fragments

```

<xsl:template match="zwierzak">
    <!-- tytuł: -->
    <h2><xsl:value-of select="nazwa/polska"/>, (<i><xsl:value-of select="nazwa/lacinska"/></i>)</h2>
    
    <xsl:apply-templates select="wystepowanie"/> <!-- szablon wystepowania -->
    <xsl:apply-templates select="opis"/> <!-- szablon opisu -->

    <xsl:variable name="nazwa_zwierza">      <!-- zmienna nazwy -->
        <xsl:value-of select="nazwa/polska"/>
    </xsl:variable>

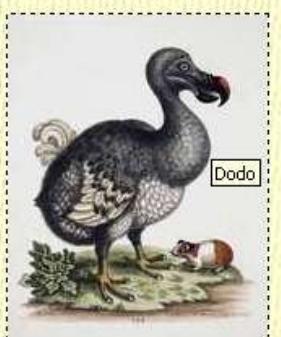
    <h4>Zdjęcia:</h4>
        <xsl:for-each select="zdjecia">
            
                <xsl:comment><xsl:value-of select=". "/></xsl:comment>
            </img>
        </xsl:for-each>
    <br/>
    <xsl:if test="position() != last()">      <!-- linie rozdzielające zwierzaki -->
        <br/><br/><hr/><br/>

```

Opis:

- Wygląd: Duży, masywny ptak o uwstecznionych skrzydłach z charakterystycznym żółtym dziobem i krótkimi nogami.
- Dodatkowe informacje: Pierwszy raz opisany został przez zeglarza i przyrodnika Georges'a Cuviera w 1825 roku.
- Pokarm: Brak dostatecznie wiarygodnych danych. Według przekazu ze żywił się kamieniami i żelazem (prawdopodobnie dodo polkłykały kamienie).
- Wymiary:
 - Długość ciała: 75 cm
 - Masa: 20 kg

Zdjęcia:

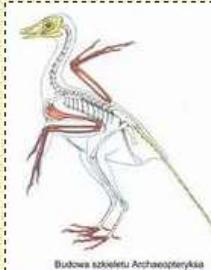


Dodo

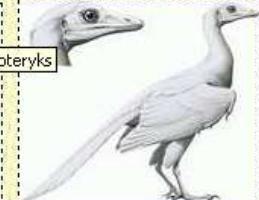
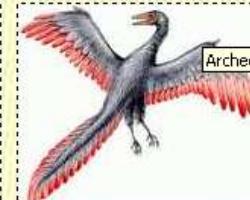
Opis:

- Wygląd: Wygląd stanowi zagadkę, jak to u zwierząt kopalnych bywa. Wiadomo, że posiadały skrzydła i ogon, ale nie miały skór skrzydłowych.
- Dodatkowe informacje: Gatunek znany jest z sześciu skamielin znalezionych w litograficznych okolicach miejscowości Solnhofen w Bawarii.
- Pokarm: Owady
- Wymiary:
 - Długość ciała: 45 cm
 - Masa: 0.3 kg

Zdjęcia:



Budowa szkieletu Archeopteryksa



Archeopteryks

```

<xsl:variable name="bgcolor">
  <body>#cccccc</body>
  <table>#ffffff</table>
  <row>#cccccc</row>
  <altrow>#ffffff</altrow>
</xsl:variable>

<xsl:template match="/">
  <html>
    <body bgcolor="{$bgcolor/body}">
      <xsl:apply-templates />
    </body>
  </html>
</xsl:template>

<xsl:template match="cars">
  <table bgcolor="{$bgcolor/table}" width="75%">
    <xsl:for-each select="car">
      <tr>
        <xsl:attribute name="bgcolor">
          <xsl:choose>
            <xsl:when test="position() mod 2 = 0">
              <xsl:value-of select="$bgcolor/altrow" />
            </xsl:when>
            <xsl:when test="position() mod 2 = 1">
              <xsl:value-of select="$bgcolor/row" />
            </xsl:when>
          </xsl:choose>
        </xsl:attribute>
        ...
      </tr>
    </xsl:for-each>
  </table>
</xsl:template>

```

| | | |
|-------|------------|------|
| Focus | Ford | 2000 |
| Golf | Volkswagen | 1999 |
| Camry | Toyota | 1999 |
| Civic | Honda | 2000 |
| Prizm | Chevrolet | 2000 |

Formatting numbers

Converting numerical values into strings

format-number(number, format_pattern, dec_format)

value to format

{prefix}number{.fraction}{suffix}

0 Digit

Digit, zero shows as absent

. The position of the decimal point (###.##)

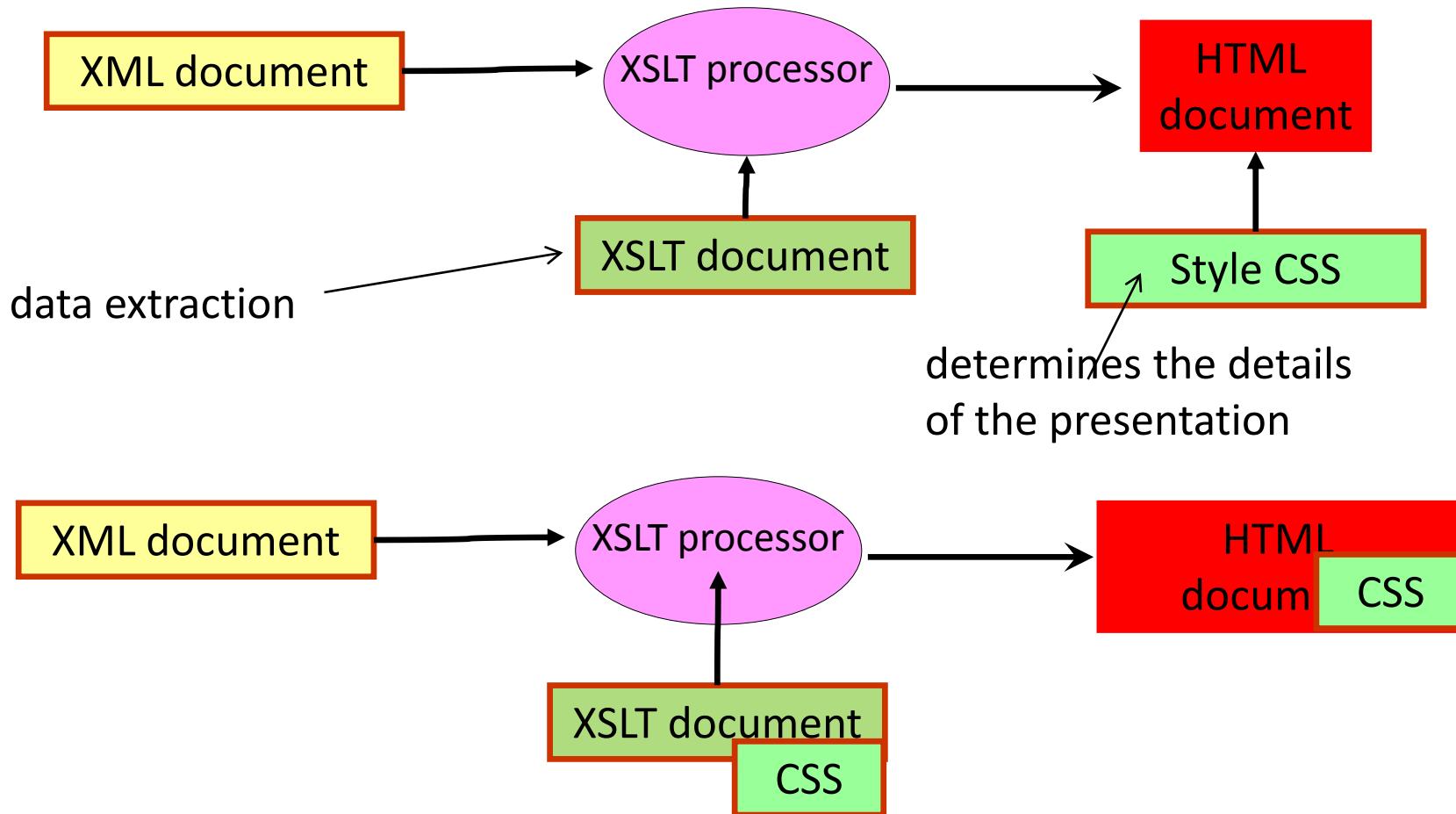
, The group separator for thousands (###,###.##)

% Displays the number as a percentage (##%)

name of the user-defined format

```
<xsl:decimal-format  
    name="formatname"  
    decimal-separator=""  
    grouping-separator=""  
    infinity=""  
    minus-sign=""  
    NaN=""  
    percent=""  
    per-mille=""  
    zero-digit=""  
    digit=""  
    pattern-separator="" />
```

XSLT – using CSS



```

body
{
    background-color:cornflowerblue;
}

a
{
    color:indigo;
    text-decoration:underline;
    font-weight:bold;
}

a:link
{
    color:indigo;
    text-decoration:underline;
}

h1
{
    font-size:xx-large;
    font-family:'Courier New','Arial';
    color:navy;
}

h2
{
    font-size:x-large;
    font-family:'Arial';
    color:purple;
}

h3
{
    font-size:large;
    font-family:'Courier New','Times New Roman';
    color:lime;
}

p
{
    font-size:normal;
    font-family:'Courier New';
}

table
{
    border-style:dotted;
    border-width:2px;
    border-color:red;
}

```

(Słoń afrykański)



[\[http://www.ekologia.gemapro.vip.alpha.pl/elefante.html\]](http://www.ekologia.gemapro.vip.alpha.pl/elefante.html) [\[http://pl.wikipedia.org/wiki/S%C5%82on_afryka%C5%84ski\]](http://pl.wikipedia.org/wiki/S%C5%82on_afryka%C5%84ski)

Występuje w:

Środowisko

lądowe

Polska

nie

Słoń afrykański zamieszkuje afrykańską sawannę oraz południowych krańców Sahary po Namibię, północną Botwanę i północną część Afryki.

Słoń afrykański jest większy od swego krewniaka, sonia indyjskiego (azjatyckiego), ma również większe uszy. Linia grzbietu wklęsła, trąba zakończona dwoma palczastymi wyrostkami. Cioczy u samicy, u samca dochodzą do 20 kg. Młode sionie są szarawoczarne, z wiekiem ich barwa zmienia się na rózową. Skórę mają pomarszczoną, pokrytą rzadkimi grubymi włosami i spłaszczonego koniec ogona.

Samica:

[http://www.ekologia.gemapro.vip.alpha.pl/elefante.html] [http://pl.wikipedia.org/wiki/S%C5%82on_afryka%C5%84ski]

```

<?xml version="1.0" encoding="windows-1250"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:xs="http://www.w3.org/2005/xpath-functions" xmlns:fn="http://www.w3.org/2005/xpath-functions/functions">
<xsl:output method="html"/>

<xsl:template match="/">
    <xsl:apply-templates/>
</xsl:template>

<xsl:template match="zwierzeta">
    <html>
        <head>
            <link rel="stylesheet" type="text/css" href="styl2.css"/>
            <title>Moje zwierzęta</title>
        </head>
        <body>
            <xsl:for-each select="zwierze">
                <h1><xsl:value-of select="nazwa_lacinska"/></h1>
                <h2>(<xsl:value-of select="gatunek"/>)</h2>
            </xsl:for-each>
        </body>
    </html>
</xsl:template>

```

| Państwo | Region | Miasto | Nocleg |
|----------|--|--|--|
| Filipiny | Nazwa: <i>Filipja</i> Opis <i>Wyspa jednym małym miastem:</i> Charakter <i>Morski</i> www http://www.filipja.com | Nazwa: <i>Filipia</i> Wielkość: <1000m ² Mieszkancy: <1000 Cena 11.40 USD | Rodzaj: <i>Pole namiotowe</i> Ulica: <i>Dolink</i> |
| | Nazwa: <i>Nowagwineandia</i> Opis <i>Śliczny region z jednym małym miastem i mnóstwem jezior:</i> Charakter <i>Pojeziorze</i> www http://www.gwineandia.com | Nazwa: <i>Gwineanka</i> Wielkość: <1000m ² Mieszkancy: <1000 Cena 21.20 USD | Rodzaj: <i>Pole namiotowe</i> Ulica: <i>Swinukowa</i> |
| | Nazwa: <i>Boliwiandia</i> Opis <i>Śliczny gorski region z jednym dużym miastem:</i> Charakter <i>Wieżki</i> www http://www.boliwiandia.com | Nazwa: <i>Cosiewo</i> Wielkość: <1000m ² Mieszkancy: 1000 - 5000 Cena 31.20 USD | Rodzaj: <i>Kwatera prywatna</i> Ulica: <i>Repe</i> |
| | Nazwa: <i>Malearandia</i> Opis <i>Śliczny nizinny region z wielkim miastem:</i> Charakter <i>Pasterski</i> www http://www.malearandia.com | Nazwa: <i>Male</i> Wielkość: >10000m ² Mieszkancy: >10000 Cena 61.20 USD | Rodzaj: <i>Hotel</i> Ulica: <i>Trutka</i> |

```

<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/xsl/transform" version="1.0">
  <xsl:output method="html"/>
  <xsl:strip-space elements="*"/>
  <xsl:template match="/">
    <html>
      <head>
        <title>Miejsca Wakacyjne</title>
      </head>
      <style>
        BODY {SCROLLBAR-FACE-COLOR: #64B09B; SCROLLBAR-HIGHLIGHT-COLOR: #64B09B; SCROLLBAR-3DLIGHT-COLOR: #000000; SCROLLBAR-ARROW-COLOR: #FFFFFF; SCROLLBAR-TRACK-COLOR: #778CA4; SCROLLBAR-DARKSHADOW-COLOR: #000000; }.td_dddddCopy2 { background: #FFD991; border: 1px solid; border-color: #FFFFFF #A7A7A7 #FFFFFF; font: 10px Verdana, Arial, Helvetica, sans-serif; color: #000000 }
        .menu_link { font: normal 10px Verdana, Arial, Helvetica, sans-serif; color: #000000; text-decoration: none }
        .menu_link:hover { font: 10px Verdana, Arial, Helvetica, sans-serif; color: #FFFFFF; text-decoration: underline }
        .table_border { border: 1px solid; border-collapse: separate; border-color: #FFCC34 #A7B4C5 #FFCC34 #FFCC34 }
        .table_border2 { border-collapse: separate; border-color: #000000 #A7B4C5 #FFCC34 #FFCC34 }
        .border { border-right-style: solid; border-right-width: 1; border-right-color: #FFCC34 }
      </style>
    <body>
      <center>
        <table border="1">
          <tr>
            <xsl:attribute name="style">background-color: red; color: black; font-weight: bold; font-family: Lucida;</xsl:attribute>
            <td><b><1><center><b>Państwo</b></center></1></b></td>
            <td colspan="2"><b><1><center><b>Region</b></center></1></b></td>
            <td colspan="2"><b><1><center><b>Miasto</b></center></1></b></td>
            <td colspan="2"><b><1><center><b>Nocleg</b></center></1></b></td>
          </tr>
          <xsl:for-each select="wakacje/etap">
            <xsl:sort select="nocleg/cena" />
            <tr>
              <xsl:attribute name="style">background-color: #FFFFDB; font-weight: bold; font-family: Lucida;</xsl:attribute>
              <td rowspan="4"><xsl:value-of select="państwo"/></td>

```

Changing the structure of the output document

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<?xmlstylesheet type="text/xsl" href="Pajeczaki2.xsl"?>
<pajeczaki xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <pajaki gatunek="tygrzyk" chroniony="tak" wyst_polska="tak">
    <nazwa jezyk="polska">Tygrzyk Paskowany</nazwa>
    <nazwa jezyk="łacińska">Argiope bruennichi</nazwa>
    <gromada>Pajeczaki</gromada>
    <wystepowanie>
      <swiat>...</swiat>
      <tereny>...</tereny>
      <srodowisko typ="lądowe" />
    </wystepowanie>
    <opis czego="Cechy">Samica: jej głowotułów pokryty jest gęstym s
    </opis>
    <opis czego="Pokarm">Pająk ten jest bardzo "wybredny", gdyż polu
    </opis>
    <opis czego="Wymiary">Samica osiąga długość ok. 25 mm. Samiec je
    </opis>
    <zdjecie zrodlo="Gfx\tygrzyk10_mlody2.jpg"></zdjecie>
    <zdjecie zrodlo="Gfx\tygrzyk12_ml_samica.jpg"></zdjecie>
    <zdjecie zrodlo="Gfx\tygrzyk12_ml_samiec.jpg"></zdjecie>
    <zdjecie zrodlo="Gfx\tygrzyk14_obiad.jpg"></zdjecie>
    <linki adres="http://pl.wikipedia.org/wiki/Tygrzyk_paskowany">wi
    <linki adres="http://www.salamandra.org.pl/magazyn/b08a07.html">
    <linki adres="http://www.eko.org.pl/otop-leszno/tygrzyk.php">eko
    <linki adres="http://www.bocian.org.pl/biuletyn/2001/b2a23.php">
    <linki adres="http://grzesiak.kei.pl/jurek/lista161.html">grzesi
    <data>2012-12-01</data>
  </pajaki>
```

```
<?xml version="1.0" encoding="UTF-16"?>
- <zwierzeta>
  - <tygrzyk>
    <opis>Samica: jej głowotułów pokryty jest gęstym s
    </opis>
    <zdjecie>Gfx\tygrzyk10_mlody2.jpg</zdjecie>
    </tygrzyk>
  - <krzyzak>
    <opis>Na odwłoku jasne plamy układają się w c
    odcień brązu. Cechą charakterystyczną jest
    pozbawiony nóg, przeważnie jest pękaty. S
    <zdjecie>Gfx/krzyzak.jpg</zdjecie>
    </krzyzak>
  - <polny>
    <opis>Posiadając małe kleszcze i gruby ogon, to
    mniej jadowita odmiana tego gatunku, char
    należy do najbardziej jadowitych skorpionó
    od 19 do 30, natomiast u samca 25 do 36. P
    <zdjecie>Gfx/skopion1.jpg</zdjecie>
    </polny>
  </zwierzeta>
```

Changing the structure of the output document

creating new elements and attributes

- <xsl:element>
- <xsl:attribute>

Copying elements from a source document

- <xsl:copy>
 - a copy of the current node without attributes and child nodes
- <xsl:copy-of>
 - copy of the current node with children nodes and attributes

```
<document>
```

...to a physical condition of a document
may also be blurred with stains,
corners and other noise-like effects.

```
<paragraph/>
```

In order to tackle these problems
degrees of fatigue.

```
<paragraph/>
```

A set of selected documents may be satisfactory for some of
these aspects, and at the same time ... defined quality metrics to
measure the indicated document aspects.

```
<paragraph/>
```

A methodology for measuring quality of documents in different
phases is...was used with relative wide band

```
<paragraph/>
```

Other issues...can be worked out with the QE

```
<paragraph/>
```

Quality improvement that can be really obtained there requires
adding to the DDLC...

```
</document>
```

```
<xsl:template match="paragraph">
```

```
  <p>
```

```
  -----  
  </p>
```

```
</xsl:template>
```

```
<xsl:template match="paragraph">
```

```
  <xsl:element name="p">
```

```
  -----
```

```
  </xsl:element>
```

```
</xsl:template>
```

Creating new elements

Element or attribute name established at runtime

```
<xsl:template match=".....">
  <xsl:element name="{node_of_the_document_tree}">
  -----
  </xsl:element>
</xsl:template>
```

```

```

```
<xsl:element name="img">
  <xsl:attribute name="src">
    <xsl:value-of select="@zrodlo"/>
  </xsl:attribute>
  <xsl:attribute name="width">
    150
  </xsl:attribute>
</xsl:element>
```

Text

we write the text

<xsl:text>

```
<xsl:value-of select="nazwisko">  
  <xsl:text>  </xsl:text>  
  
<xsl:value-of select="imie">  
  <xsl:text>  </xsl:text>
```

Push-me Pull-you Stylesheets

INPUT-DRIVEN (CALLED PUSH)

walk the input tree

match elements in input tree

do something when you find a
match

STYLESHEET DRIVEN (CALLED PULL)

more like a typical computer
program

walk the stylesheet (which
specifies the order of the output
document)

when it asks for data, go get it
from the input tree

Push-me Pull-you Stylesheets

Input-driven (called Push)

Stylesheet driven (called Pull)

Why Programmers Will Like Pull?

They are used to controlling the order of execution

They don't trust the stylesheet to "do it right"

They need recursion, so they force it (instead of using it)

They fight the template design

Why Pull Can Be a Problem?

Style sheets tell the processor what to do when it finds something. So processor controls the finding of things in the input tree (when to do it).

Pull style sheets control both what and when, and fight the design of the language

Pull: The Big Beginner Mistake

Pull should be used sparingly for special situations only

- Best used on very regular, predictable structures (data)

Beginners use it for everything

How to notice this problem in stylesheets

- using `<xsl:for-each>` to force recursion (instead of using templates, which recurse as designed)
- using `<xsl:value-of>` which
 - processes the text inside an element
 - (instead of `<xsl:apply-templates>`, which processes both text and the embedded tags inside an element)

XSLT summary

declarative, data-driven language of transforming XML trees

the basic processing paradigm is pattern matching

Possibility of conditional processing, looping, parameterization, sorting, numbering, ...

Various types of output documents

DTD

DR WIOLETA SZWOCH

DEPARTMENT OF INTELLIGENT INTERACTIVE SYSTEMS

Document Type Definition

DTD defines the document description language

a formal description of the document's structure, containing information about elements (tags), their additional properties (attributes) and dependencies between elements that connect them to a tree structure

A DTD specifies a grammar for the document

- Constraints on structures and values of elements, attributes, etc.

Document Type Definition

DTD defines the document description language

Advantages:

- The possibility of validation
- The possibility of automatic conversion
- Documentation
- Creating multiple documents according to one recipe

An Internal DTD Declaration

An External DTD Declaration

Tree.XML:

```
<?xml version=1.0? encoding="UTF-8" >
<!DOCTYPE tree [
    <!ELEMENT tree (#PCDATA)>
]>
...
```

Tree.DTD:

```
<?xml version="1.0" encoding="UTF-8" ?>
<ELEMENT tree (#PCDATA)>
...
```

Tree.XML:

```
<?xml version=1.0 standalone= " yes" ?>
<!DOCTYPE tree SYSTEM " Tree.dtd">
...
```

An External DTD Declaration

Private

```
<!DOCTYPE name_of_the_root_element SYSTEM "URL_DTD">  
<!DOCTYPE tree SYSTEM "Tree.dtd">
```

Public

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

Document Type Definition

Elements

Atributes

Entities

Elements

Element declaration

```
<!ELEMENT name_of_element type_of_the_content>
```

```
<!ELEMENT name (#PCDATA)>
```

Element content

- Content type
 - chars (#PCDATA)
 - elements
 - mixed
- EMPTY
- ANY

`<name_of_element>`
content according to type
`</ name_of_element >`

```
<name>  
    Kowalski  
</name>
```

```
<person>  
    Kowalski  
</person>
```

Order indicators

- sequence
- choice
- sequence and choice connection

```
<computer>
    < out_dev ></ out_dev >
    < in_dev ></ in_dev >
</ computer>
```

```
<computer>
    < out_dev ></ out_dev >
</computer>
```

```
<!ELEMENT computer (out_dev, in_dev)>
<!ELEMENT computer (out_dev | in_dev)>
<!ELEMENT computer (model, (out_dev | in_dev), nr)>
```

```
<komputer>
    <model> </ model >
    <out_dev ></ out_dev >
    <nr> </nr>
</ komputer>
```

Number of occurrences

| No symbol | One and only one | Element |
|-----------|-----------------------------|-------------------------------------|
| ? | Zero or one | optional element |
| * | zero or any number of times | Optional element iteration |
| + | one or more times | iteration of the obligatory element |

```
<computer>
    <disk> 1 </disk>
    <disk> 2 </disk>
    <disk> 3 </disk>
</computer>
```

```
<!ELEMENT computer (disk, disk, disk)>
```

```
<!ELEMENT computer (disk*)>
```

Content type

Char content

```
<!ELEMENT computer (#PCDATA)>
```

```
<computer> my computer </computer>
```

Mixed content

```
<!ELEMENT computer (#PCDATA | ram | disk | monitor)*>
```

```
<computer>
    computer with
    <ram> 2 </ram>
    <monitor></monitor>
</computer>
```

Content type

ANY content

- any element declared in the DTD

```
<!ELEMENT computer ANY>
```

EMPTY content

- must be empty

```
<!ELEMENT point EMPTY>
```

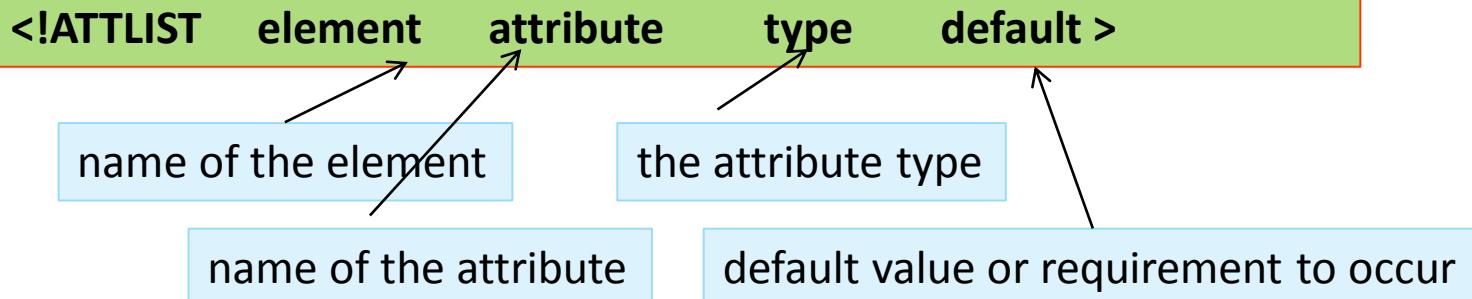
```
<point/>
```

```
<!ELEMENT computer (#PCDATA)>
```

```
<computer> </computer>
```

Attributes

attribute declaration



types for attribute

- CDATA
- ID
- IDREF
- ...

`<car colour= "black" >...</car>`

Attribute types

string

```
<!ATTLIST document no CDATA >
```

```
<document no="1.2.3"> ... </document >
```

enumeration

```
<!ATTLIST document status (project | edition | ready) >
```

```
<document status="edition"> ... </document>
```

modular

```
<!ENTITY % status "status (project | edition | ready)">
```

```
<!ENTITY % no "no CDATA">
```

```
<!ATTLIST document %no; %status;>
```

Default values

accepted if no value is given in the document

```
<!ATTLIST document status (project | edition | ready) 'edition'>
```

described by the keyword

- #REQUIRED
- #IMPLIED
- #FIXED the value cannot be changed

```
<!ATTLIST document status (project | edition | ready) #REQUIRED>
<!ATTLIST document status (project | edition | ready) #IMPLIED>
<!ATTLIST document status (project | edition | ready) #FIXED '14.11'>
```

Global attributes

Each element can use them

```
<!ENTITY % x.global  
' status (project | edition | ready) #REQUIRED  
no    CDATA           #IMPLIED '>  
...  
<!ELEMENT document (#PCDATA | chapter)*>  
<!ATTLIST document %x.global;>  
...  
<!ELEMENT chapter (#PCDATA)>  
<!ATTLIST chapter %x.global;>  
...
```

DTD - limitations

No default values for elements

DTD does not support namespace

No possibility to limit the text content of documents

No data types other than textual

Inability to define own types of elements

Poor modularity

Non-compliance with XML specification