



Ca' Foscari
University
of Venice
Department
of Humanities

[ve]dph

Venice Centre for
Digital and Public
Humanities

CLARIN-IT



Research Institute for
Digital and Cultural Heritage



TEXTUAL MARKUP

HTML, CSS, XML

(typographic, structural, semantic markup)

strand #1

digital textual scholarship

FEDERICO BOSCHETTI
& ANGELO MARIO DEL GROSSO,
CNR-ILC / VENICE CENTRE FOR
DIGITAL AND PUBLIC
HUMANITIES

JULY 6, 2020

VeDPH Seminar – Privacy Notice



Si segnala che **il seminario verrà videoregistrato**. Pertanto, l'eventuale e volontaria partecipazione (via chat o attivando la telecamera) permetterà la raccolta di dati personali (quali, ad esempio, l'immagine, la voce e il contenuto dell'eventuale intervento). La registrazione del seminario rimarrà disponibile sul canale YouTube di VeDPH per permettere a chi non ha potuto partecipare live di assistere allo stesso.

Ulteriori informazioni, anche in relazione ai diritti riconosciuti dalla normativa privacy, sono disponibili alla pagina:

<https://www.unive.it/pag/34665/>.



Please notice that **the seminar will be recorded**. Therefore, any voluntary participation (via chat or by activating the camera) will consent the collection of personal data (such as, for example, the image, voice and content of any speech). The registration of the seminar will remain available on the VeDPH YouTube channel to let those who could not participate live to attend it.

Further information, also in relation to the rights recognized by the privacy legislation, are available on the page:

<https://www.unive.it/pag/34665/>.

Introduction

We will study the principles of textual markup and the basic elements of three formalisms created by the W3C (<http://w3c.org>):

- **HyperText Markup Language (HTML)**
- **Cascading Style Sheets (CSS)**
- **eXtensible Markup Language (XML)**

HTML

The source code of a web page

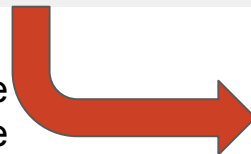


HomePage | Figure Retiche | Composi | Entità nominate | Descrizione del progetto

Eschilo, I Persiani (1-268)

Tr. di G. Fraccaroli	Testo greco	Tr. degli studenti del Liceo Gargallo (SR)
CORO 1 Ecco i canuti de la Persa gente, 2 Che uscì la patria a conquistar de' Greci: 3 Ne le sedi per molto oro opulente 4 Serse ne elesse a sostener sue veci, 5 Serse di Dario, Serse onnipotente. —	Χορός 1 Τόδε μὲν Περσῶν τῶν οἰχομένων 2 Ἑλλάδ' ἐς αἶαν πιστὰ καλεῖται, 3 καὶ τῶν ἀφ' ὧν καὶ πολυχρύσων 4 ἐδράνων φύλακες, 5 κατὰ πρεσβείαν οὓς αὐτὸς ἄναξ 6 Ξέρξης βασιλεὺς Δαρείου γένης 7 εἴλετο χώρας ἐφορεύειν.	CORO 1 Siamo i fedeli, i custodi dei sontuosi palazzi reali 2 colmi d'oro, quelli che lo stesso signore Serse, 3 Re, figlio di Dario, scelse perché vecchi, 4 per sorvegliare la regione, 5 quando i Persiani partirono per la terra ellenica.
	1 Τόδε μὲν Περσῶν ... / 65 χρόνον τρομῶνται : L'ingresso e la disposizione del coro sulla scena sono sottolineati dalla rigidità e dalla lentezza del metro anapestico. Subito dopo viene contestualizzato il luogo, la reggia di Susa. Segue il lungo e dettagliato catalogo degli eroi in stile epico. 1 Τόδε ... / 153 προσαυδῶν : Si tratta della Parodo. La tragedia non presenta un prologo ma inizia con la Parodo che ha struttura tripartita. Due sezioni anapestiche delimitano una sezione lirica. 2 ἔς : εἰς 3 πολυχρύσων = 3 per molto oro opulente : La ricchezza dell'esercito e della flotta è ancora più marcata dal quadruplice utilizzo dell'aggettivo. 3 πολυχρύσων : I compositori omerici che presentano il prefisso in πολυ- hanno una funzione iperbolica; questi trasmettono infatti l'immagine di un regno all'insegna della grandezza e della maestosità, l'Impero Persiano. 5 κατὰ πρεσβείαν = 1 canuti : Il termine indicante la causa, πρεσβεία, viene sostituito nella traduzione dal termine indicante l'effetto.	
6 Ah! ma l' anima mia turbano bieci 7 Augùri, e il dubbio la sgomenta ... oh il giorno 8 Dunque mai non verrà del suo ritorno?	8 ἄμφι δὲ νόστω τῷ βασιλεῖ 9 καὶ πολυχρύσου στρατιάς ἤδη 10 κακόμανης ἄγαν ὀρσολοπέται 11 θυμός, ἔσωθεν δὲ βαυδεῖ.	6 Per il ritorno del re 7 e dell'esercito armato d'oro, 8 l'animo, ormai profeta di sventure, 9 è profondamente angosciato e pieno di sgomento.
6 l'anima ... / 7 Augùri : Fraccaroli nella sua traduzione elimina il concetto dell'animo che si turba nella profondità di sé stesso. 8 Dunque ... ritorno : Fraccaroli traduce i versi 8 e 9 dopo il verso 11 trasformando un'affermativa in un'interrogativa dal tono fortemente patetico.		

view page
source



```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <html>
3 <head>
4 <title>Eschilo, I Persiani</title>
5 <meta charset="UTF-8"/>
6 <link rel="stylesheet" href="./style.css"/>
7 </head>
8 <body>
9 <center>
10 
11 </center>
12 <br/>
13 <div class="menu">
14 <span class="gray">HomePage</span> | <a href="./figures.html">Figure Retiche</a>
15 <a href="overview.html">Descrizione del progetto</a>
16 </div>
17 <br/>
18 <center>
19 <h2>Eschilo, <em>I Persiani (1-268)</em>
20 </h2>
21 </center>
22 <table class="parallelText">
23 <colgroup>
24 <col class="frCo1"/>
25 <col class="grCo1"/>
26 <col class="garCo1"/>
27 </colgroup>
28 <thead>
29 <tr>
30 <th>Tr. di G. Fraccaroli</th>
31 <th>Testo greco</th>
32 <th>Tr. degli studenti del Liceo Gargallo (SR)</th>
33 </tr>
34 </thead>
35 <tbody>
36 <tr>
37 <td class="tdFrText">
38 <strong>
39 <spanText>CORO</spanText>
40 </strong>
41 <br/>
42 <span class="nFra">1 </span>
43 <span>Ecco i canuti de la Persa gente,</span>
44 <br/>
45 <span class="nFra">2 </span>
46 <span>Che uscì la patria a conquistar de' Greci:</span>
47 <br/>
48 <span class="nFra">3 </span>
49 <span>Ne le sedi per molto oro opulente</span>
50 <br/>
51 <span class="nFra">4 </span>
52 <span>Serse ne elesse a sostener sue veci,</span>
53 <br/>
54 <span class="nFra">5 </span>
55 <span>Serse di Dario, Serse onnipotente. —</span>
56 </tr>

```

Markup

Markup languages **represent, describe** data, whereas programming languages **process** data.

Markup languages do not **perform** arithmetic or logical operations

Typically a markup language uses **tags** (e.g. `<p>` and `</p>`) delimited by **metacharacters** (e.g. `<` and `>`) to record **secondary information** (e.g. layout specifications) about **primary information** (e.g. textual content, expressed by character sequences)

HTML describes interlinked documents exchanged on the web

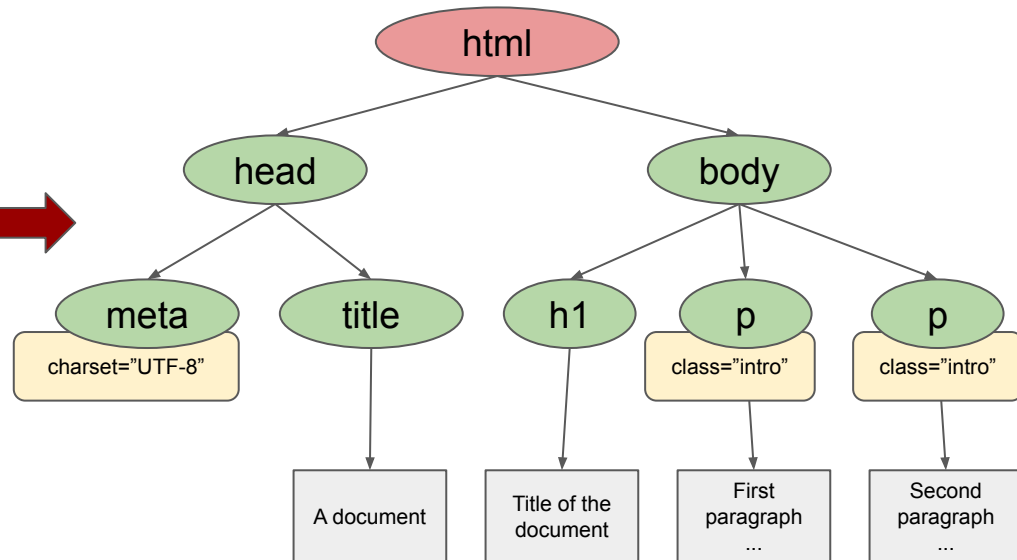
Elements and attributes

Any HTML document has a hierarchical structure that is represented by a tree of nodes rooted in the HTML element

```

<html>
  <head>
    <meta charset="UTF-8"/>
    <title>A document</title>
  </head>
  <body>
    <h1>Title of the document</h1>
    <p class="">First paragraph .... </p>
    <p class="">Second paragraph ... </p>
    ...
  </body>
</html>

```



Elements and attributes

Elements are data containers. If they have textual or mixed content, they are delimited by a **starting tag** and an **ending tag**

`<p>First paragraph
with two lines</p>`

if they are **empty**,
they are constituted by an **empty tag**

Starting tag syntax: `< + tagName + >`

Ending tag syntax: `< + / + tagName + >`

Empty tag syntax: `< + tagName + / + >`

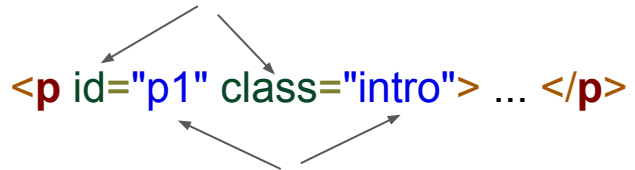
Elements and attributes

Elements may have **attributes**.

Attributes are constituted by two parts:

attribute name

`<p id="p1" class="intro"> ... </p>`



and **attribute value**.

Attribute syntax:

attributeName + = + " + attributeValue + "

Also empty elements may have attributes

``

<https://allthetags.com>

The head

Within the **html root element**, an html document contains a section for metadata: **head**, and a section for textual and (possible) multimedia content: **body**

head contains the title of the document, a list of **meta** elements e.g. with character encoding, **link** with stylesheet reference, etc. and it may contain also **scripts** to provide the document with commands to process contents

```
<html>
  <head>
    <title>Minimalist document</title>
    <meta charset="UTF-8"/>
    <link rel="stylesheet" href="style.css"/>
    <script type="text/javascript" src="main.js"></script>
  </head>
  <body>
    <!-- ... -->
  </body>
</html>
```

Character encoding

Character encoding of the entire document is declared in the head:

```
<head>  
  <meta charset="UTF-8"/>  
</head>
```

UTF-8 is one of the most efficient serializations for **unicode**

“Unicode is the universal character encoding, maintained by the Unicode Consortium (<http://unicode.org>). This encoding standard provides the basis for processing, storage and interchange of text data in any language. [...] Unicode covers all the characters for all the writing systems of the world, modern and ancient. It also includes technical symbols, punctuations, and many other characters used in writing text.” (<http://unicode.org>)

In HTML (special) characters can be encoded also as entities, e.g.

- < → <
- > → >
- & → &
- " → "

(more entities: <https://bit.ly/2ZDyqNY>)

Semantic elements & outer document structure

Inside the body is possible to define a **header**, a menu (**nav**), a division in **articles** or **sections** and a **footer**

```
<body>
  <header>Header info here</header>
  <nav>
    <a href="hp.html">HomePage</a> |
    <a href="a.html">About</a>
  </nav>
  <article>
    <h2>First article</h2>
    <p>An article about Marco Polo</p>
  </article>
  <article>
    <h2>Second article</h2>
    <p>Another article about Marco Polo</p>
  </article>
  <footer>Footer info here</footer>
</body>
```



Header info here

[HomePage](#) | [About](#)

First article

An article about Marco Polo

Second article

Another article about Marco Polo

Footer info here

Inner document structure

The inner document structure is articulated in text blocks or divisions (`<div>...</div>`), which usually are constituted by paragraphs (`<p>...</p>`)

```
<div>  
  <h2>First division</h2>  
  <p>First paragraph</p>  
  <p>Second paragraph</p>  
</div>  
<div>  
  <h2>Second division</h2>  
  <p>Just one paragraph</p>  
</div>
```



First division

First paragraph

Second paragraph

Second division

Just one paragraph

Lists

Ordered lists (` ... `) and unordered lists (` ... `) contain list items (` ... `)

```
<ol>
  <li>First ordered item</li>
  <li>Second ordered item</li>
  <li>Third ordered item</li>
</ol>
<ul>
  <li>First unordered item</li>
  <li>Second unordered item</li>
  <li>Third unordered item</li>
</ul>
```



1. First ordered item
 2. Second ordered item
 3. Third ordered item
- First unordered item
 - Second unordered item
 - Third unordered item

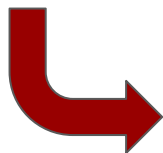
Formatting

Typographic bold and italic are marked as **strong** and **em**(phasis).

<p>This is regular,

this is usually rendered as bold

and this is usually rendered in italic.
</p>



This is regular,
this is usually rendered as bold
and *this is usually rendered in italic.*

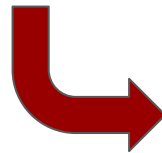
Multimedia

A HTML document can embed multimedia content, such as images, audio, video, etc.

The most frequent multimedia content is an image (``)

```

```



Tables

Tables (**<table>...</table>**) are constituted by table rows (**<tr>...</tr>**), which contain table headers (**<th>...</th>**) or table data (**<td>...</td>**).

```
<table>
  <tr>
    <th>1st col</th>
    <th>2nd col</th>
  </tr>
  <tr>
    <td>1r1c</td>
    <td>1r2c</td>
  </tr>
  <tr>
    <td>2r1c</td>
    <td>2r2c</td>
  </tr>
</table>
```



1st col	2nd col
1r1c	1r2c
2r1c	2r2c

Forms

Forms provide documents with interaction between the user, the front-end (client side, on the browser) or the back-end (server side) of the application.

```
<form action="/proc.jsp">  
  <label>Author:</label>  
  <input type="text" id="author"/><br/>  
  <label>Title:</label>  
  <input type="text" id="title"><br/>  
  <input type="submit" value="Submit">  
</form>
```



Author:

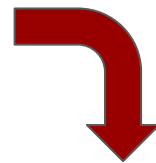
Title:

Form widgets

```

<form id="form1" action="/script.jsp">
  <label>text box</label> <input id="tb" type="text"/><br/>
  <label>password</label> <input id="pwd" type="password"/><br/>
  <label>text area</label> <textarea id="in3"></textarea><br/>
  <label>checkbox 1</label> <input id="cb1" type="checkbox" value="1"/>
  <label>checkbox 2</label> <input id="cb2" type="checkbox" value="2" checked="true"/>
  <label>checkbox 3</label>
  <input id="cb3" type="checkbox" value="3"/><br/>
  <label>radio 1</label>
  <input id="r1" name="group1" type="radio" value="1"/>
  <label>radio 2</label>
  <input id="r2" name="group1" type="radio" value="2"/>
  <label>radio 3</label>
  <input id="r3" name="group1" type="radio" value="3"/><br/>
  <label>select</label>
  <select id="sel">
    <option id="opt1" value="1">First choice</option>
    <option id="opt2" value="2">Second choice</option>
    <option id="opt3" value="3">Third choice</option>
  </select>
</form>

```



text box

password

text area

a long text

checkbox 1 ☐ checkbox 2 ☒ checkbox 3 ☐

radio 1 ☐ radio 2 ☐ radio 3 ☒

select

First choice ▼

CSS

CSS

Cascading Style Sheets (CSS) describe how HTML elements must be **rendered**, i.e. **graphically represented** by browsers on different **media devices** (e.g. desktops, laptops, tablets or smartphones) and different **operative systems** (e.g. Linux, Mac, Windows, Androids) in similar ways but with the necessary adaptation due to screen size, screen ratio and so on.

A CSS is constituted by a list of **rules**

```
.abstract{  
    color: blue;  
}  
.intro{  
    color: red;  
}
```

Any **rule** is constitute by a **selector** and a list of **declarations**. Each declaration is formed by an **attribute name** and an **attribute value**.

Selectors & declarations

Selectors are necessary to identify an element or a set of elements

Most frequent selectors

- **element** (e.g. **p**)
- **#id** (e.g. **#w31**)
- **.class** (e.g. **.row**)

List of selectors and their combinations

- <https://bit.ly/2ZEqprQ>

F. Boschetti & A.M. Del Grosso, *Textual Markup*

Declarations concern any aspect of the graphical representation of an element

Example: how to render a paragraph of class `quot` in italic, gray, justified and smaller than the other paragraphs in the same div

```
p.abstract{
    color: #c0c0c0;
    text-align: justify;
    font-size: 90%;
}
```

List of properties setted in the declarations

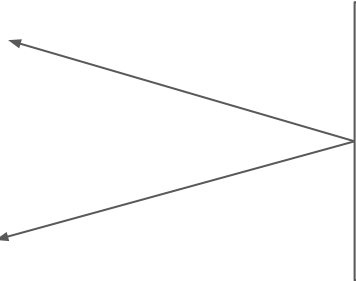
- <https://bit.ly/3iEDZV8>

Cascade

CSS are a cascade of style sheets because properties are combined, calculated, inherited and overwritten in cascade, i.e. according to the hierarchy and the priority of the declarations

E.g.

```
div{  
    font-size: 16px;  
}  
div p{  
    font-size: 50%;  
}
```



Font size of paragraphs is set to 8px, because the size of the parent element of **p**, which is **div**, is set to 16px

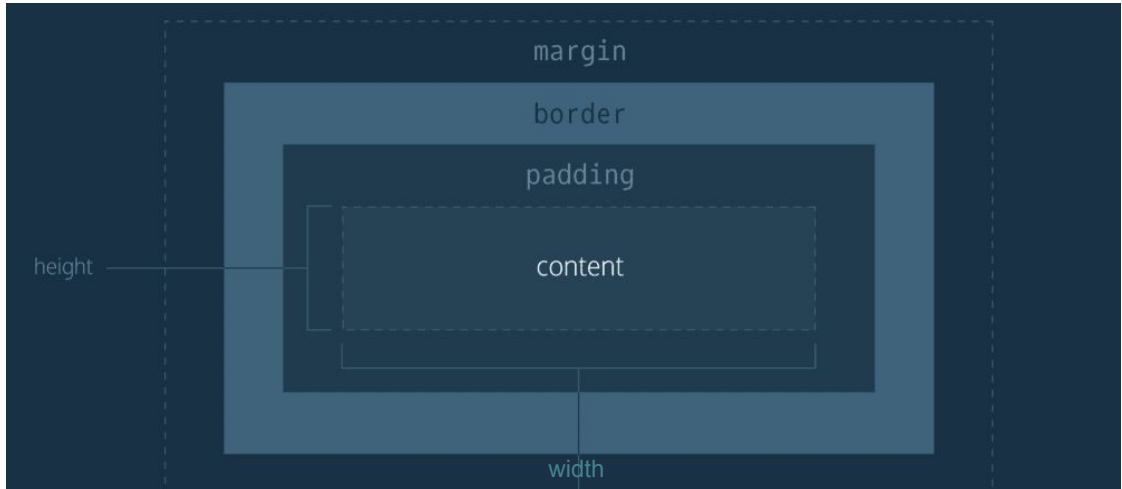
Element box

Each element **content** is located in a **box** surrounded by settable **padding**, **border** and **margin**

padding is a space between **content** and **border**

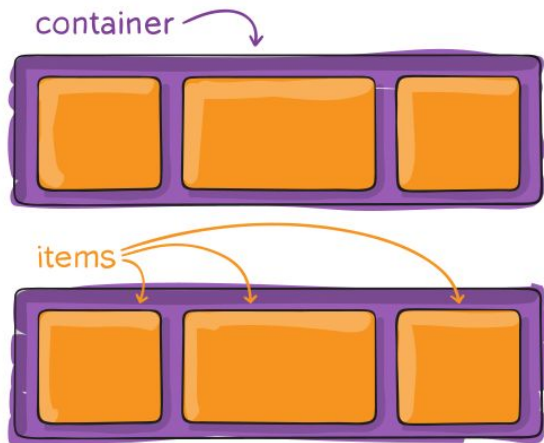
border can be invisible or render by several types of lines

margin is a space beyond the **border**



Complex layouts

New media devices (tablets, smartphones, etc.) need a complex and dynamic management of web page layouts: **flex display** for containers and **flex properties** for content elements have been created for this purpose



```
div.container{  
  display: flex;  
}  
p.item{  
  flex: auto;  
  width: 33%;  
  text-align: justify;  
  margin: 1%;  
}
```

A complete guide to flexbox

- <https://bit.ly/2BCpj8e>

XML

From HTML to XML

HTML is a markup language oriented to the **presentation** and **interlinking** of documents on the web. Its expressivity is limited but optimized for the **visualization** of textual and graphic content on the web.

The evolution of the web requires more expressivity: **XML**, the eXtensible Markup Language, was created as a **metalanguage** oriented to **data representation** (instead of **data presentation**).

SGML (the common ancestor)

The **Standard Generalized Markup Language (SGML)** is the common ancestor of **HTML** and **XML**.

In the previous slides we introduced only the XHTML serialization for HTML, in order to have HTML documents that are also XML well-formed documents. But HTML can be serialized also to be just an SGML well-formed documents, without being an XML well-formed documents. For this reason, do not surprise if HTML documents can contain empty tags like **
** instead of **
** or empty attributes like **selected** instead of **selected="true"**. We will continue using only XHTML!

XML Elements

The **prolog** contains the XML declaration, optional comment lines, optional processing instructions, and an optional document type declaration.

The XML **declaration** indicates that the document is written in XML and specifies the version of XML used.

The encoding attribute identifies the character set used in the document.

The standalone attribute indicates whether the document contains any references to external files.

Comments in the prolog provide additional information about what a document will be used for and how it was created.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<!--
  This document contains data on SJB Pet Boutique
  holiday specials
  Filename: sjbpct.xml
  Author:   Patricia Dean
  Date:    9/18/2017
-->
<products>
  <product>
    <productName>Dog Shirt Gift Basket</productName>
    <manufacturer>SJB Pet Boutique</manufacturer>
    <description>Something for every day of the week</description>
    <price>35.99</price>
    <price>26.79</price>
    <productItems>1200, 1201, 1202, 1203, 1204, 1205, 1206</productItems>
  </product>
</products>
<!-- generated by the finance department -->
```

The **document body** contains the document content in a hierarchical tree structure.

Found after the document body, the optional **epilog** contains any final comment lines and processing instructions.

DTDs
schemas

XML is used to create markup languages (tag sets, vocabularies, XML applications).

XML is **extensible** because it does not provide any default element but rules to build XML documents

- ◆ XML **prolog** (*declaration*)
- ◆ XML **body** (*node tree*)
- ◆ XML **epilog** (*final comments*)

XML Elements

```
<element>content</element>
```

opening tag: `<element>`

closing tag: `</element>`

```
<element/> (empty element)
```

```
<element>
```

```
  <element>some text</element>
```

(nested element)

```
</element>
```

As in HTML, an **empty element** has no content. An element can contain text or other (nested) elements.

XML element content

- structural: only other elements
- mixed: other elements and text
- textual: only text

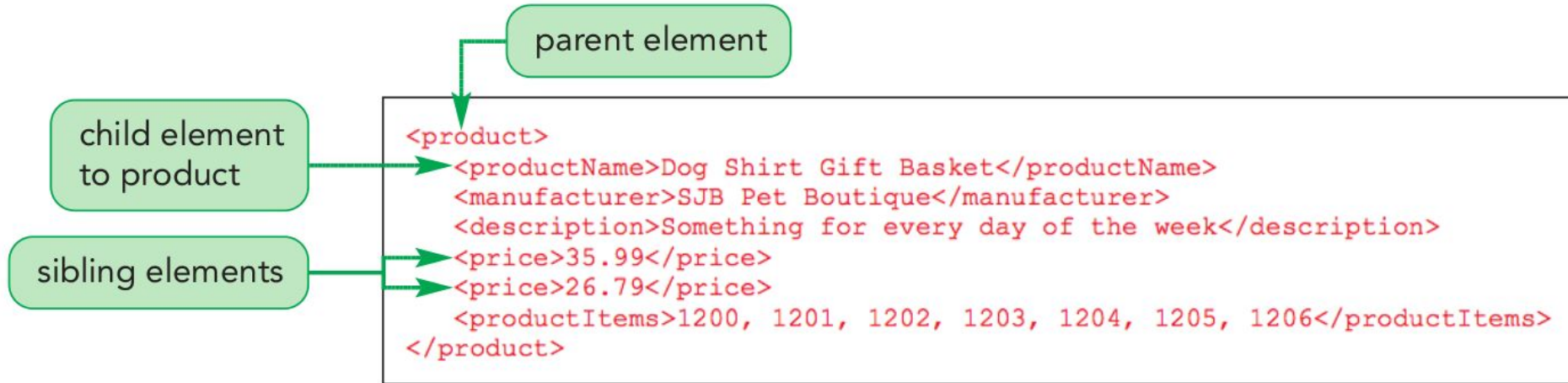
Opening and closing tags must have the same name. Tags can be used more than one time.

A **vocabulary** is constituted by a tag set

A nested element (child) is contained inside another element (parent).

Sibling element are hierarchically equivalent and positionally sequential.

XML Elements



XML attributes

```
<element attribute="value">
    [...]
</element>

<element attribute="value" />

<element  attribute="value",
          attribute2="value2" />
```

The order of the attributes is not relevant.

XML elements may have one or more attributes.

An attribute describe a property of the element.

An attribute has two components: name and value.

The value is a string and must be always surrounded by quotation marks.

XML schema

XML is a metalanguage. Any specific XML language is described by an XML schema, which define the vocabulary of the language (the tag set) and the syntactic structure of the language (content model)

The schema may be created *a priori*, to validate all the documents defined by the schema, or *a posteriori* from non validated but well-formed documents. Some applications (e.g. oXigen can be used to extract a schema from well-formed documents

Well-formed and valid documents

Well-formed documents

General XML syntactic rules are respected (e.g. the rules to open and close tags)

Any XML document must be well formed: no syntactic errors are allowed

Valid documents

The constraints of the XML schema are respected

Any valid document must be also well-formed

SVG, MathML and other popular XML schemas

Many XML schemas have been created for specific purposes, e.g. SVG (graphics), MathML (mathematics), docbook (publishing)

```
<svg height="100"
      width="100">
  <circle cx="50"
          cy="50"
          r="40"
          stroke="black"
          stroke-width="3"
          fill="red" />
</svg>
```

```
<mrow>
  <msup>
    <mfenced>
      <mrow>
        <mi>a</mi>
        <mo>+</mo>
        <mi>b</mi>
      </mrow>
    </mfenced>
    <mn>2</mn>
  </msup>
</mrow>
```

```
<book xml:id="simple_book"
      xmlns="http://docbook.org/ns/docbook" version="5.0">
  <title>Very simple book</title>
  <chapter xml:id="chapter_1">
    <title>Chapter 1</title>
    <para>Some text</para>
  </chapter>
  <chapter xml:id="chapter_2">
    <title>Chapter 2</title>
    <para>Other text</para>
  </chapter>
</book>
```

A minimalist HTML doc vs a minimalist TEI-XML

```
<html>
  <head>
    <meta charset="UTF-8" />
    <title>HTML doc</title>
  </head>
  <body>
    <div>
      <h1></h1>
      <p>Textual content</p>
    </div>
  </body>
</html>
```

```
<?xml version="1" encoding="utf-8" ?>
<!DOCTYPE TEI SYSTEM "tei-all.dtd">
<TEI>
  <teiHeader>
    <fileDesc>
      <!-- mandatory data -->
    </fileDesc>
  </teiHeader>
  <text>
    <body>
      <div>
        <head></head>
        <p>
          Textual content
        </p>
      </div>
    </body>
  </text>
</TEI>
```

The header in HTML and TEI-XML

```
<head>
  <title>Decameron</title>
  <meta name="author"
    content="G. Boccaccio">
  <meta name="description"
    content="VeDPH Project">
  <meta name="keywords"
    content="italian literature">
  <meta charset="UTF-8">
  <link rel="stylesheet"
    href="style.css">
  <script
    src="app.js"></script>
  <meta name="application-name"
    content="Decameron
    digital edition">
</head>
```

```
<?xml version="1" encoding="utf-8"?>
<teiHeader>
  <fileDesc>
    <titleStmt>
      <title>Decameron digitale</title>
      <author xml:id="gb">G. Boccaccio</author>
      <respStmt>
        <resp>[...]</resp>
        <name>[...]</name>
      </respStmt>
    </titleStmt>
    <noteStmt>[...]</noteStmt>
    <publicationStmt>[...]</publicationStmt>
    <sourceDesc><bibl>Decameron</bibl></sourceDesc>
  </fileDesc>
  <encodingDesc><projectDesc><p>VeDPH Project</p>
</projectDesc></encodingDesc>
  <profileDesc></profileDesc>
  <revisionDesc></revisionDesc>
</teiHeader>
```

Conclusion

HTML, CSS and XML are technologies in evolution and the general trend is the max separation of concern between data **structure**, **semantics**, **representation** and **presentation**

New versions of HTML are more and more open to semantics (i.e. data representation vs data presentation) and customization (i.e. component embedding and tag set extension). Next HTML specification tags like `<app-my-project>...</app-my-project>` can be embedded in an HTML document, in order to provide the language with custom expressivity