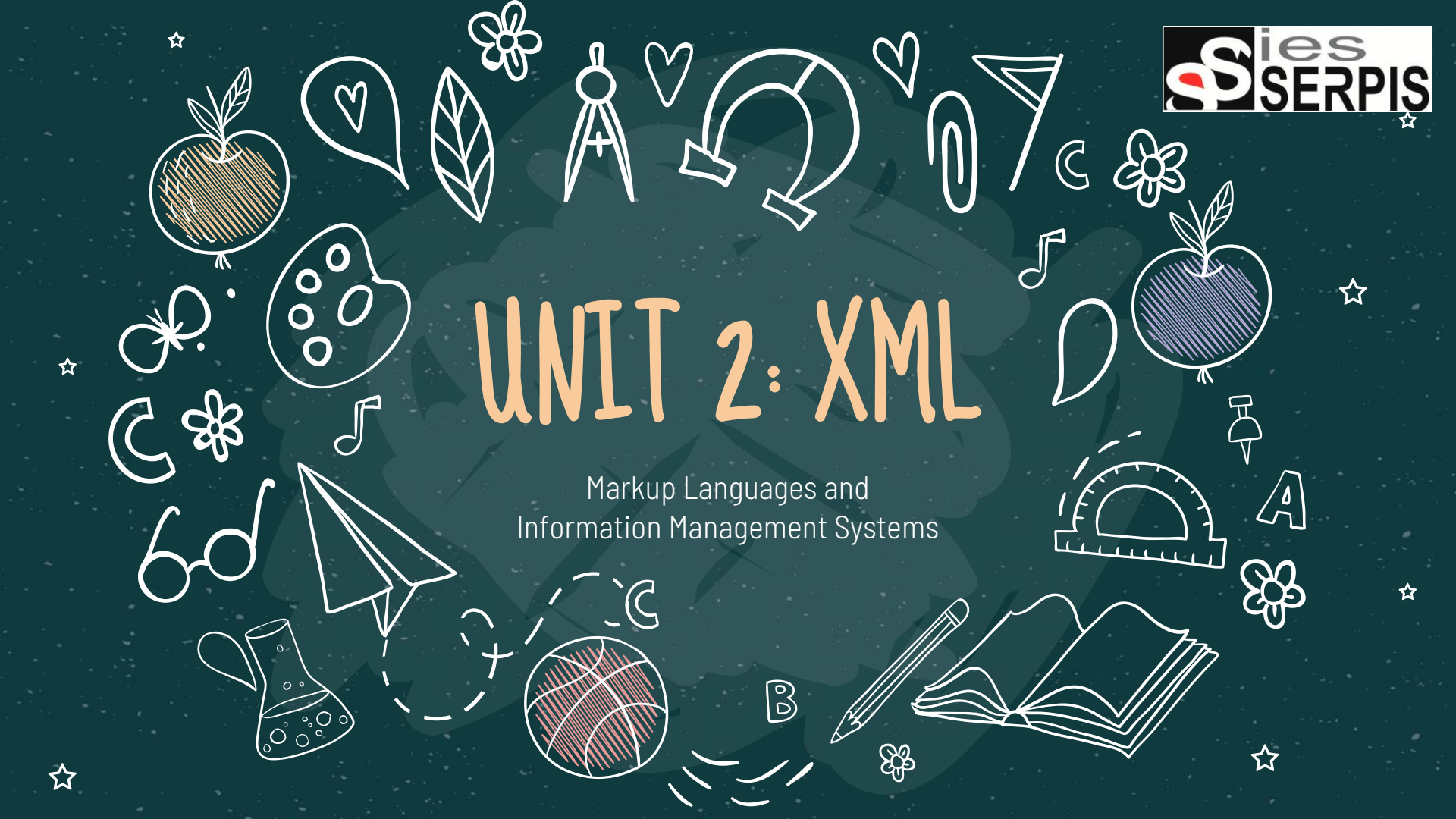

ies SERPIS
UNIT 2: XML
Markup Languages and
Information Management Systems

01

INTRODUCTION

02

CHARACTERISTICS

03

STRUCTURE

04

VALID vs. WELL-FORMED

05

COMPONENTS

01. INTRO-DUCTION

eXtensible Markup Language

XML means *lenguaje de marcas extensible*

INTRODUCTION

XML is a W3C specification, defined as a *general purpose markup language*.

HTML has shortcomings regarding information treatment.

INTRODUCTION

BIRTH

In late 90s, W3C was under pressure to expand HTML and to include data storage and transport capabilities.

W3C decided not to expand it, but set some guidelines so anyone could create extensions: markup languages adapted to their needs.

INTRODUCTION

- This decision from W3C is coherent with their philosophy:
  - Separate information from representation.
  - We wil reiterate this when the time comes to study HTML and CSS
- XML defines a common structure and syntax for data storing, managing
  - to promote compatibility
  - to enable treatment with the same tools

INTRODUCTION

<xml />

EXTENSIBLE

Allows you to create extensions: derived formats that meet XML criteria

METALANGUAGE

It is a lenguage that allows you to create markup languages, used to store data in a readable form

INTRODUCTION

XHTML

- Version of HTML compliant with XML rules (2000).
- W3C unsuccessfully tried to set it as the standard language for web page creation, replacing HTML.
- They finally resigned and focused on HTML5 development

INTRODUCTION

01 Usage of XML is very widespread

02 It has become a basic tool for information Exchange and storage in many computer areas

03 The main reason is the use of plain text to store information

04 Compatibility with any software and hardware

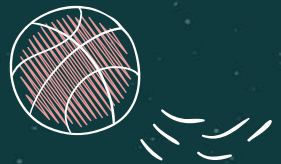
INTRODUCTION

Areas where XML is useful

- Web site maintenance.
- Information exchange among enterprises (B2B).
- Uploading to and downloading from databases.
- Content syndication (RSS).
- e-commerce applications
- Scientific applications, with markup for mathematical expressions, chemical formulation...
- e-books, with markup for authoring statement, rights and legal stuff...
- Smartphones and other small electronic devices, with optimized marked languages
- ...

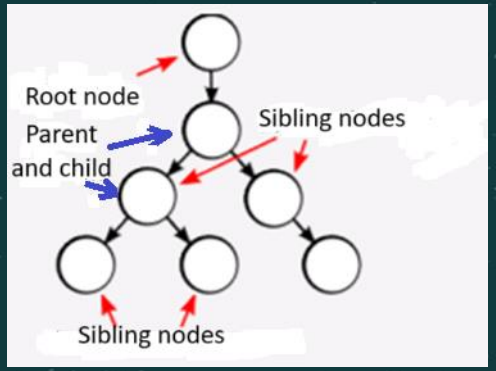
CHARACTERISTICS

02.



CHARACTERISTICS

- XML is a markup language (tagged).
- Basic syntax:
  - Opening <xxx> and closing </xxx> tags, for each element.
  - Also, empty tags <xxx/> (no content, opening + closing merged).
- Names for the tags: developer's decision.
  - No restrictions.
- Tree structure:
  - Tag nesting.
  - Father-son and sibling relationships.
  - Information within elements: possible in attributes or as content.



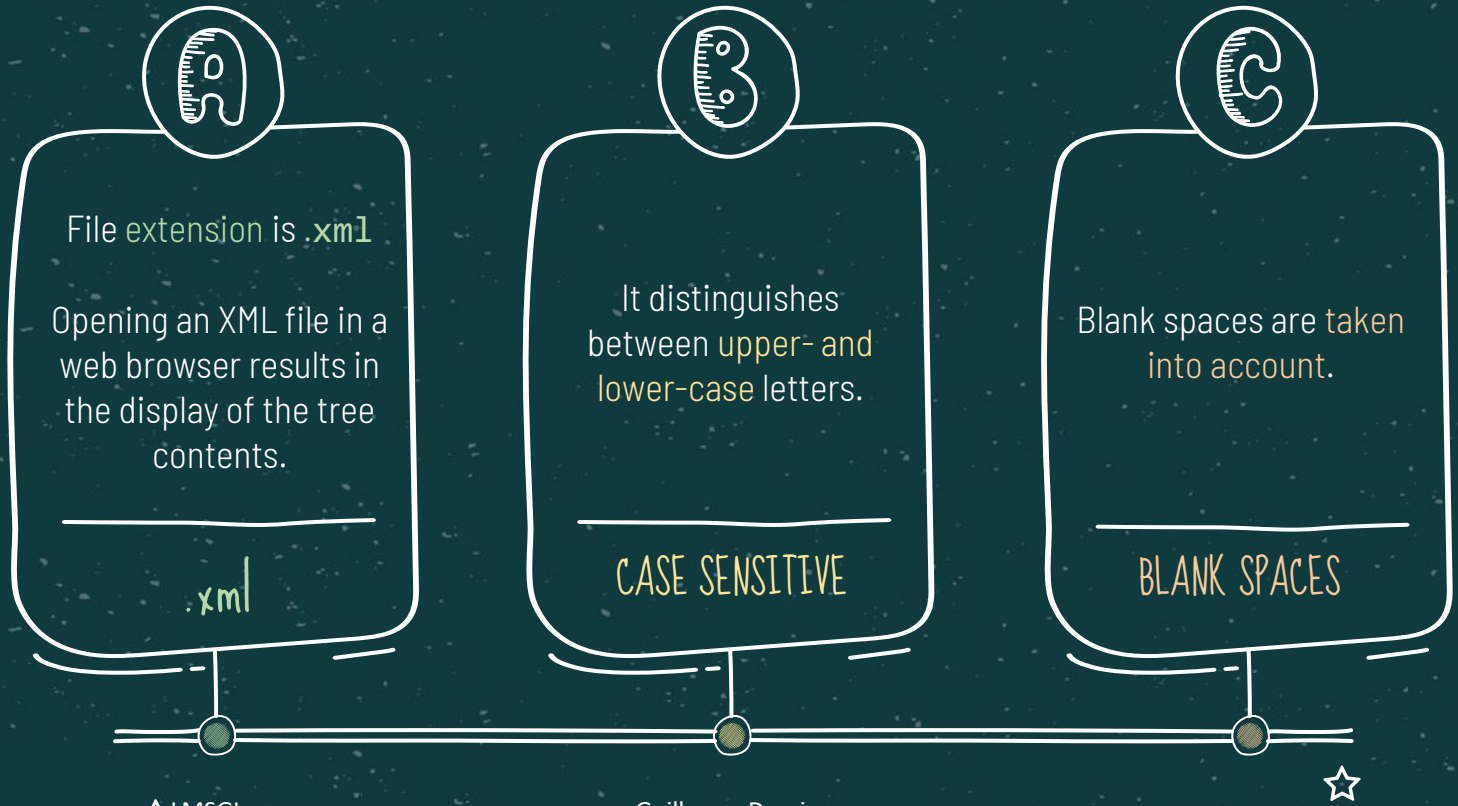
CHARACTERISTICS

example

```
<?xml version="1.0" encoding="UTF-8"?>
<contactlist>
  <contact>
    <name>Guillermo Domingo</name>
    <phone1>961206105</phone1>
    <phone2/>
    <email>g.domingomartinez@edu.gva.es</email>
  </contact>
</contactlist>
```

empty tag

CHARACTERISTICS

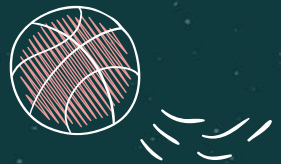


CHARACTERISTICS

XML	HTML
Contains data	Presents data
Does nothing	Used to create web pages
Non-predefined tag	Predefined tags
Extensible: data can increase or decrease, but applications keep working	

STRUCTURE

03.



STRUCTURE

example

```
<?xml version="1.0" encoding="UTF-8"?>
<contactlist>
  <contact>
    <name>Guillermo Domingo</name>
    <phone>961205925</phone>
    <email>g.domingomartinez@edu.gva.es</email>
  </contact>
</contactlist>
```

Root element opening tag

The contact list has one contact

Root element closing tag

Definition of the XML version and character set

The contact has three child elements

STRUCTURE

XML documents have
two distinct parts

STRUCTURE

The **prologue** contains:

- A first line indicating
  - that it is an XML file.
  - the XML version used.
  - (*optional*) the character set (`encoding`).
  - (*optional*) an indicator of the autonomy of the document (`standalone`).
    - `no`: (*default*) the document depends on definitions in an external file
    - `yes`: the document is self-contained
- It might also contain, among other things:
  - the declaration of the type of the document (DTD).
  - a link to a style sheet (CSS o XSLT).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE contactlist SYSTEM "contacts.dtd">
<?xml-stylesheet type="text/css"
 href="style.css"?>
```

STRUCTURE

The body:

- Contains the stored information.
  - Structured using tags.
- Tags are defined by the developer.
  - Illustrative names of the info contained should be used.
- Must start with the root element opening tag and end with the corresponding closing one.
  - Cannot be empty.
- Any element may contain:
  - other elements.
  - attributes.
  - text.
  - entities.
  - comments.

```
<element1>
  <element2 attrib="value">Text1</element2>
  <element2 attrib="value2">Text2</element2>

  <element3>
    <element4 attribute="v4"/>
  </element3>
  <!--Comment->

</element1>
```

STRUCTURE

Types of tags

Opening or starting

<xxx>

Closing or ending

</xxx>

Empty

<xxx/>
(equivalent to <xxx></xxx>)

Tags must be closed in the reverse order of opening

STRUCTURE

Attributes

- Complementary information about an element.
- Defined also by the developer.
- `attributeName = "value"` pairs
  - Value in (single or double) quotation marks mandatorily.
- Never in closing tags.
- Cannot be repeated in the same tag.

```
<contact group="family">
   ...
</contact>
```

STRUCTURE

Elements

STRUCTURE

Character entities

- Escape sequences for special characters.
- They begin with & and end with ;
- Five of the have aliases.
- For the remaining symbols: number format with #
  - Example: ± is ±
  - See values in table (warning: hex values)

&amp;	&
&lt;	<
&gt;	>
&quot;	"
&apos;	'

STRUCTURE

CDATA

- Means *Character Data*.
- Syntax:

  `<![CDATA[content]]>`

- The content thus delimited will be interpreted as a character string, and not as tags or other references.
  - exception: `]]>` → Cannot be nested.

```
<units>
  <unit number="2">XML</unit>
</units>
<!-- Element unit contains XML -->


<units>
  <![CDATA[<unit number="2">XML</unit>]]>
</units>
<!-- Element units contains all blue text -->
```

STRUCTURE

- First, identify what elements will the file contain.
- There are usually several possibilities.
- The chosen solution must consider the needs of the applications or DBs that will use the documents.

CREATING XML

STRUCTURE

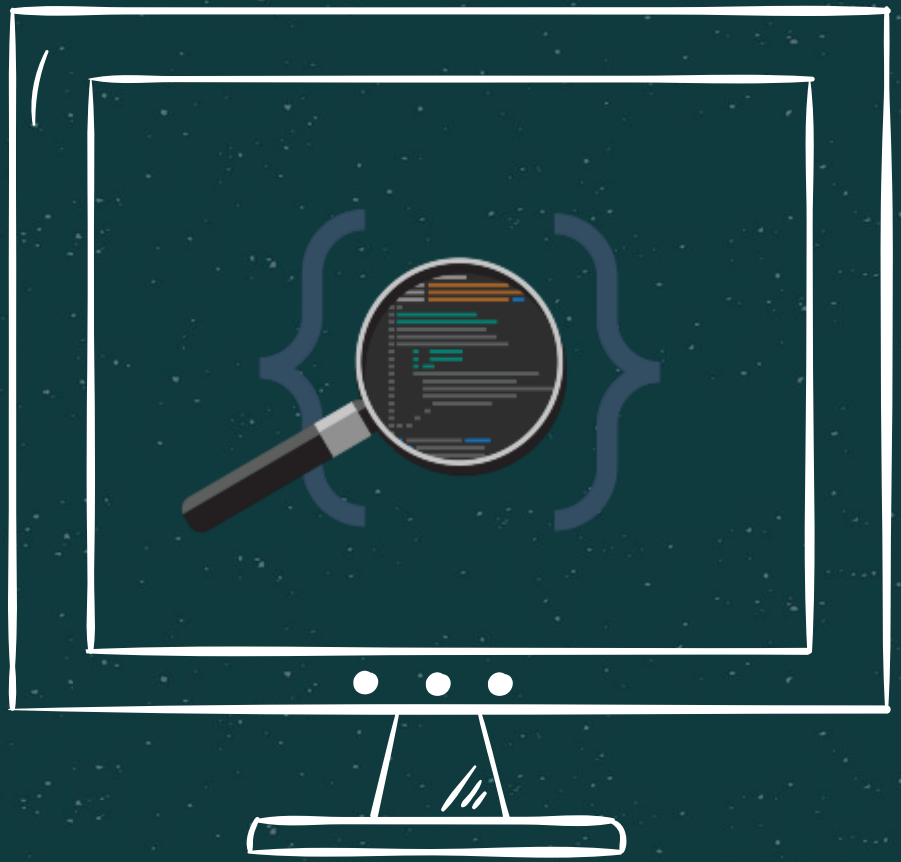
examples

```xml
<!-- Example 1 -->
<?xml version="1.0" encoding="UTF-8"?>
<name>Guillermo Domingo Martínez</name>

<!-- Example 2 -->
<?xml version="1.0" encoding="UTF-8"?>
<name>
  <first_name>Guillermo</first_name>
  <surname>Domingo Martínez</surname>
</name>

<!-- Example 3 -->
<?xml version="1.0" encoding="UTF-8"?>
<name>
  <first_name>Guillermo</first_name>
  <surname1>Domingo</surname1>
  <surname2>Martínez</surname2>
</name>
```

PRÁCTICA 2.1



STRUCTURE

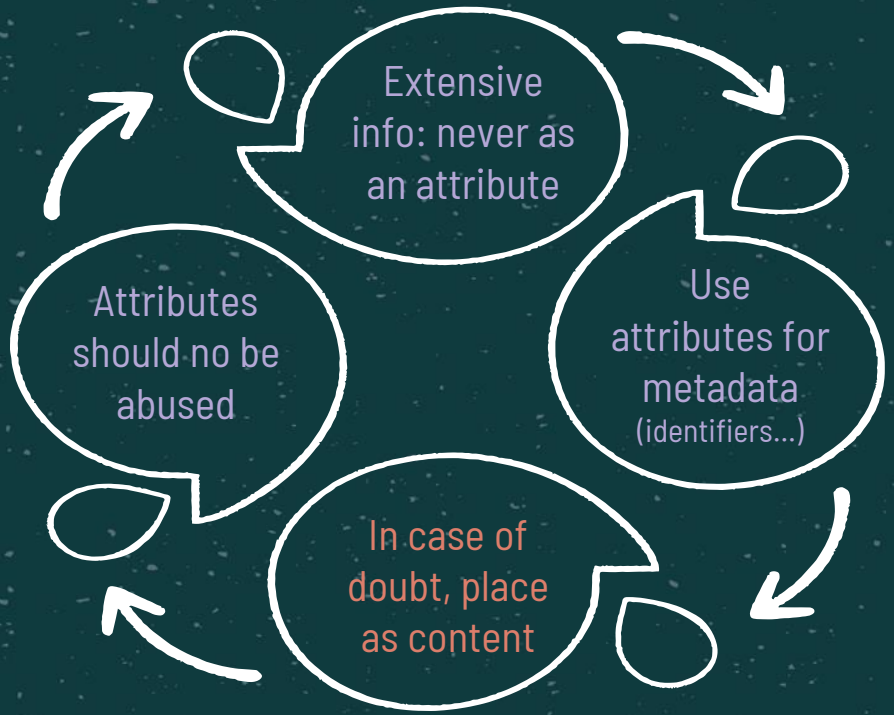
```
<slide>
  <title>Use content or use attributes?
  </title>
            ...
</slide>


<slide
  title="Use content or use attributes?">
            ...
</slide>
```

DILEMMA:

When to save data as content and when as an attribute value?

STRUCTURE



This aspect is not regulated and is left to the developer's discretion.

However, there is a good practices agreement.

Please note that attributes cannot be nested and are hard to expand.

STRUCTURE

example

```
<bookstore>
  <book id="345">
    <author>Rowling, Joanne Kathleen</author>
    <title>Fantastic Animals and Where To Find
Them</title>
    <genre>Fantasy</genre>
  </book>
  <book id="346">
    <author>King, Stephen</author>
    <title>The Shining</title>
    <genre>Horror</genre>
  </book>
</bookstore>
```

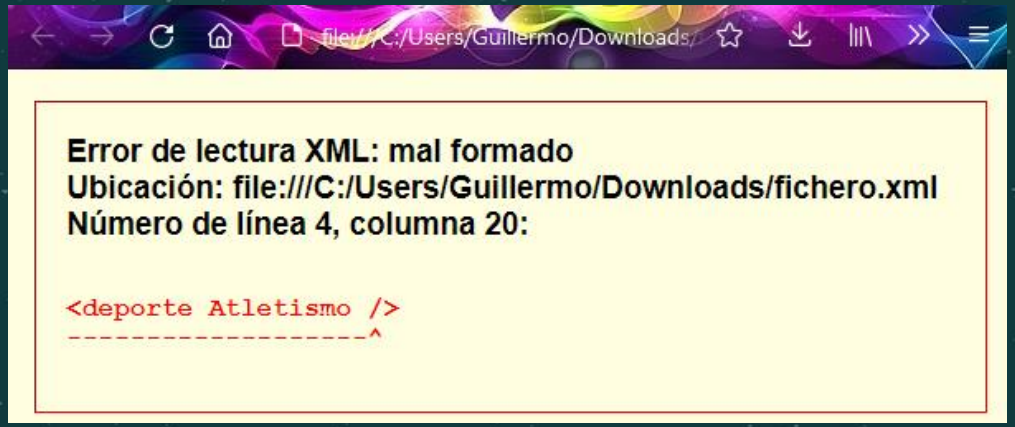
04. VALID vs. WELL-FORMED

VALID vs. WELL-FORMED

An XML document is *well-formed* if it does not have syntactical errors

- No unclosed elements.
- Checks:
  - case sensitivity.
  - blank spaces.
- Tags: closed in the reverse order of opening.
- Unique root node: first to open and last to close.
- Attributes, only in opening tags.
  - Separated by blanks.
  - No duplicates in the same tag.
- Attributes must always have values in quotes.
- Names: alphanumeric, '-', '_', '.'
  - First character: letter or '_'
  - Never begin with 'xml'.
- XML declaration is the first line and is well typed.
- Comments, well typed.
- Entities, well typed.

VALID vs. WELL-FORMED



BADLY FORMED

- Web browsers try to display HTML documents with syntactical errors.
- In contrast, W3C established that applications must not process XML files with errors.
  - A badly-formed file will halt the application.

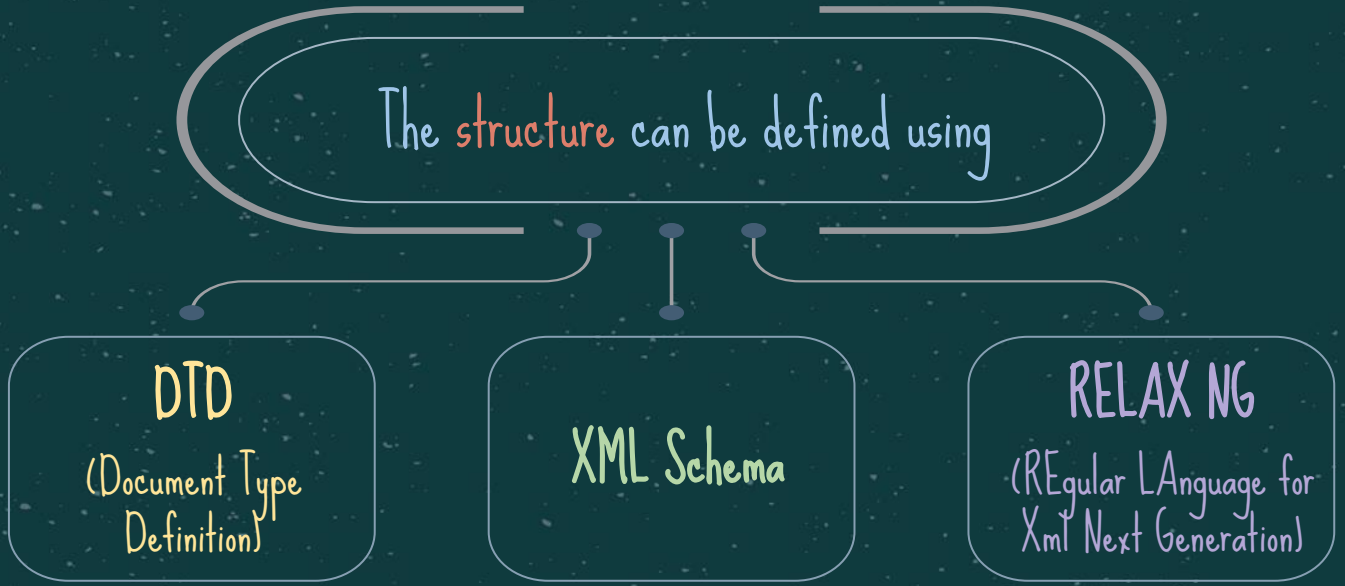
VALID vs. WELL-FORMED

An XML document is *valid* when:

Is well-formed

Complies with the rules established regarding its structure.

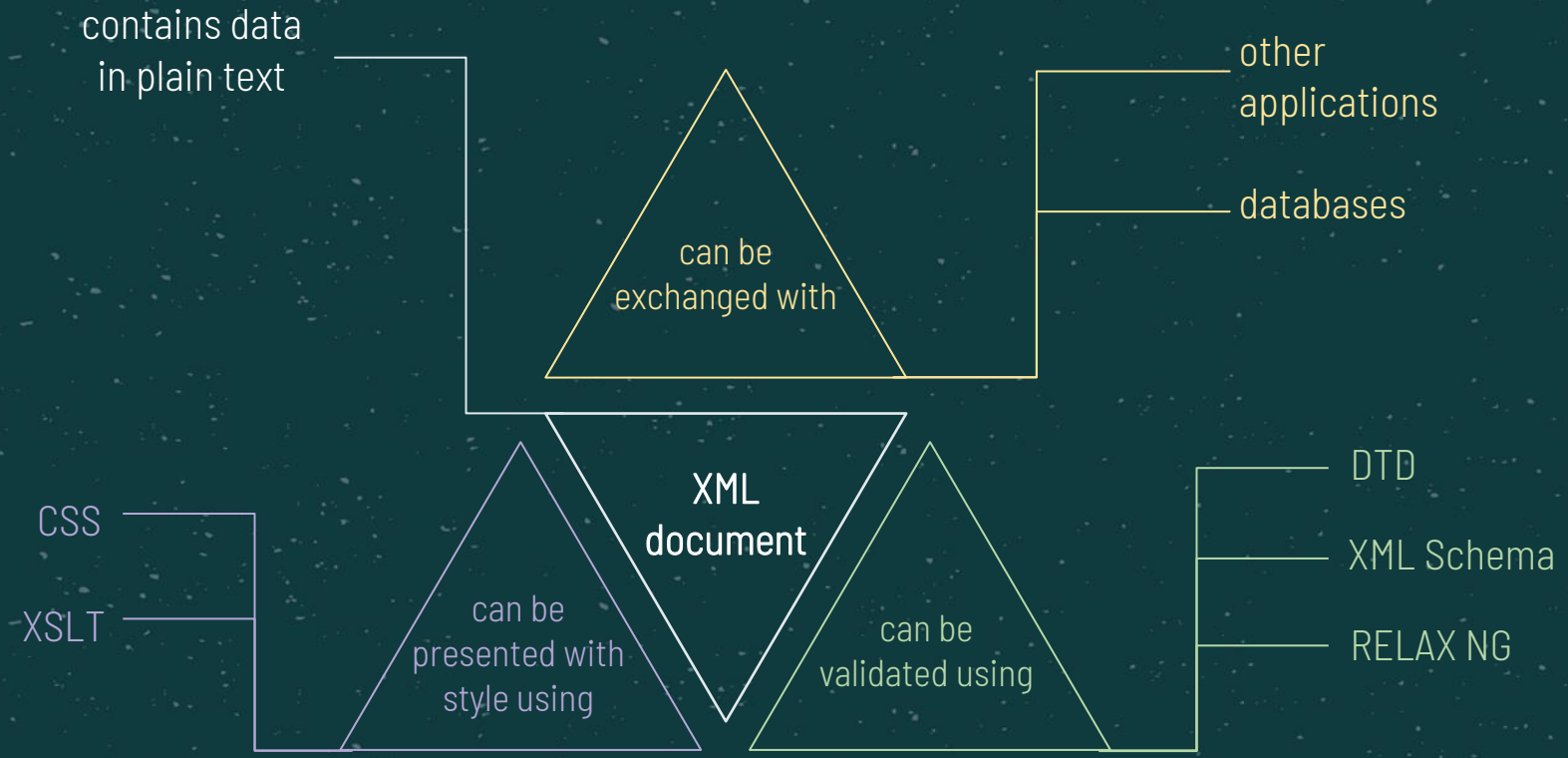
VALID vs. WELL-FORMED



PRÁCTICA 2.2

05. COMPONENTS

COMPONENTS


contains data
in plain text
other
applications
databases
can be
exchanged with
XML
document
CSS
XSLT
can be
presented with
style using
can be
validated using
DTD
XML Schema
RELAX NG

DTD
(Document Type Definition)

- Defines:
  - **what** can be used (elements, attributes, entities and annotations) to build an XML document.
  - **how** to use it.
- Allows to check the validity of the document.
- Created using a special syntax.

COMPONENTS

XML Schema

- Same purpose as DTD.
  - Plus, it allows to indicate data types for elements.
- Created using XML syntax.
- Greater complexity tan DTD.

RELAX NG

(REgular LAnguage for Xml Next Gen)

- Same purpose as DTD.
- Syntax can be:
  - XML-like.
  - compact.
- Simpler.

COMPONENTS

CSS

- Allows to apply style to the presentation of the information contained in XML documents.
- Same usage as with HTML
  - Element selection based only in the tree structure of the document.

XSL

(eXtensible Stylesheet Language)

- Designed to meet the styling needs of XML.
- Allows to:
  - transform the document into another (*XSL Transformations*)
  - apply style to the document using Formatting Objects (XSL-FO)

COMPONENTS

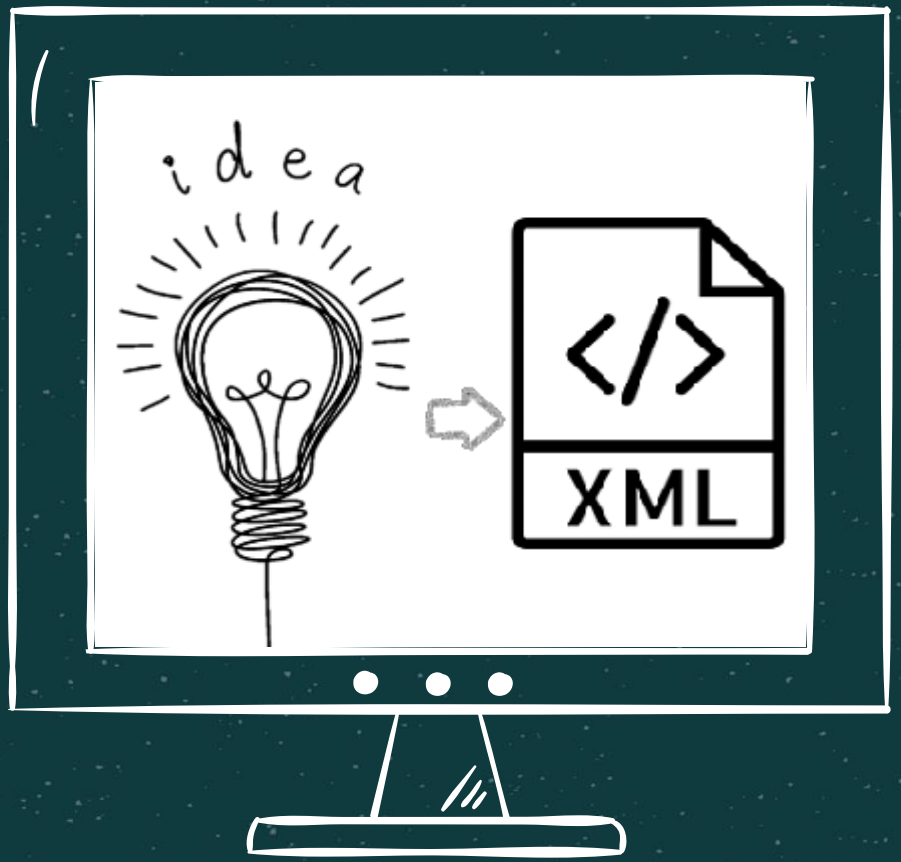
CSS

- Is a style sheet markup language
- Can be used both with HTML and XML.

XSL

- Is a programming language
  - Greater power.
- Used only with XML.

PRÁCTICA 2.3



THANKS!

Do you have any questions?

g.domingomartinez@edu.gva.com