XML

Project objective

- Learn the basics of programming in **XML**
- Understanding that **it's** a branding language
- Understanding that **it's** a metalanguage
- Know the main differences between **XML** and **JSON**

Theory

Which is a brand language

Markup

It's our markers defines the rules for encoding a document
The format it uses can read as many machines as humans
They are symbols that give information which the document is added and improves the meaning

Which is a metalanguage

It's used to use a different language. To the language that the object language is talking about.
It serves to describe the characteristics of a particular document
It is not a programming language tags are used to save and organize the data
It's extensible we can create our own tags
our own language

Know the main differences between XML and JSON

The difference between XML and JSON is that XML is a meta-language/markup language and JSON is a lightweight data exchange. That is, XML syntax is specifically designed to not have inherent semantics. Particular element names mean nothing until a particular processing application processes them in a particular way. In contrast, JSON syntax has a specific semantics embedded in things between things is an object, things between [] is an array, and so on.

A JSON parser, therefore, knows exactly what each JSON document means. An XML parser only knows how to separate markup from data. To deal with the meaning of an XML document, you must write additional code.
identifies the parts

Defines five cases where XML is used

- To display dynamic data in HTML. With JavaScript you can read an external XML file and update the contents of the data in a web page.
- Both computer systems and databases contain information in incompatible formats.

- XML data is stored in plain text format, which enables an independent way to store data, and facilitates the creation of data that can be shared by different applications.
- Upgrading to new systems (hardware or software platforms) takes a long time. Large amounts of data must be converted and incompatible data is often lost.
- XML data is stored in text format. This makes it easy to expand or upgrade to new information systems, new applications, or new browsers without data loss.
- Store an application's settings, such as style templates.
- To use standard XML-based protocols such as Simple Object Access Protocol (SOAP), for remote communications between applications.

What does AJAX relate to XML

AJAX is an acronym for Asynchronous Javascript and XML, i.e. Javascript and Asynchronous XML.
indicates how the parties relate to tags
is placed between symbols <> a closure </>

Syntax

Statement

may or may not go is optional it is recommended to be the first element
It is sensitive to capital and minuscula
starts with xml and then the Encoding version specifies the type of encoding by the characters

Tags

It's a structure element
It is known as xml node or elements
The name is placed between <>
It is sensitive in mayuscula and minuscula
It must have open and close if no content is called empty tag
In case of simple it is placed as follows <miElement/>

Root Element

You only have one root element
It's the highest level
the other element are nested to the
We can think of the xml document as a tree

Attribute

Specifies a property that has the element
Used as name/Value pair
you can have more than one attribute
is sensitive to mayusand and minuscule
The attribute name is placed normal and the value is enclosed in quotation marks

Reference

Add Text or Information in the Document

They always start with & and end with ;
There are two types of entity renference
contains a name between the beginning and the end
&amp
Character reference
contains a number followed by a number
the number is a Unicode code for the character
&#65

Text

space, tabs and line breaks between attributes are ignored
Some characters are reserved for syntax and cannot be used directly so a replacement must be used
These characters are known as predefined character entities

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

Number of character entities
We use this code

	Decimal	Hexadecimal;
<	&#60;	&#x3C;
>	&#63;	&#x3E;
'	&#39;	&#x26;
&	&#38;	&#x27;
"	&#34;	&#x22;

Significant spaces
If space is part of the text
insignificativo
space is ignored is insignificant

Xml document
has two parts
prolog document
It's always going to go the top of the document
It goes before the root element
Contains the xml declaration
element document
It's for a hierarchical.
each section has a specific purpose
sections are created with elements that in turn can contain other elements

Attribute TYPES

Chain
Tokenized
id uniquely identify the element
idref One-id reference

IdrefS Reference of all element ids
Entity attribute represents an external entity in the document
Entities The attribute represents external entities in the documents
NMTOKEN has restrictions on the data can be part of the attribute

Listed predefined values in your statement
NotationType, will be referenced to an annotation declared in the document
Enumeration, defines a list of values that the attribute can have

Comments are placed this way
<!-- Second Plate-->

CDATA is character data
It is taken as direct text and the parser ignores <> & what is inside is not considered inside a Markup
We can use them to place character entities directly
<! [CDATA[
Text
]]>
Nesting is not allowed

PI processing instructions
allows the document to have Instructions for Applications
it's not part of the document data
It can contain any data except ?>
These are steps to the app
<?target instruction?>
target - identifies the application to which the instruction is given
instruction - the information it processes to the app

<?xml-stylesheet href="miestilos.css" type="text/css"?>
indicates that you are going to the browser
href and type are the instructions

Validation
Validation is carried out in two ways
Well-formed XML well-formed document
valid XML document -valid document

Well-trained
If DTD is not used it should be used correctly &, <,>,', "
In nesting the internal tag must be closed before the external tag
all tags should have a close
It has to have a root element
Validated
If it is well formed and has a DTD we say that it is valid

DTD
document type Declaration
Check the vocabulary

Check the validity of the structure with the rules of a particular XML language
You can specify the document in another document
The outsiders use
<! DOCTYPE curso System "curso.dtd">
Started
<! DOCTYPE
Element - It serves us to tell the parser from which element is the root for the parse
DTD identifier es el path de un archivo URL
[] There are the statements that are known as internal subsets
<! DOCTYPE element "identifier">
XSD
They are used to describe

LISTA DE TAREAS A REALIZAR

Task	Priority	Hours	Difficulty
Documentation	High	1,00	High
Organization	High	1,00	High
Pre-search for information	Normal	1,00	Normal
Repository creation	Low	0,15	Low
Documenting Project	Normal	1,00	Normal
Restaurant.xml	Normal	1,00	normal
CREATION README	Low	1,00	Low
Testing / Correction Errors	High	0,30	High

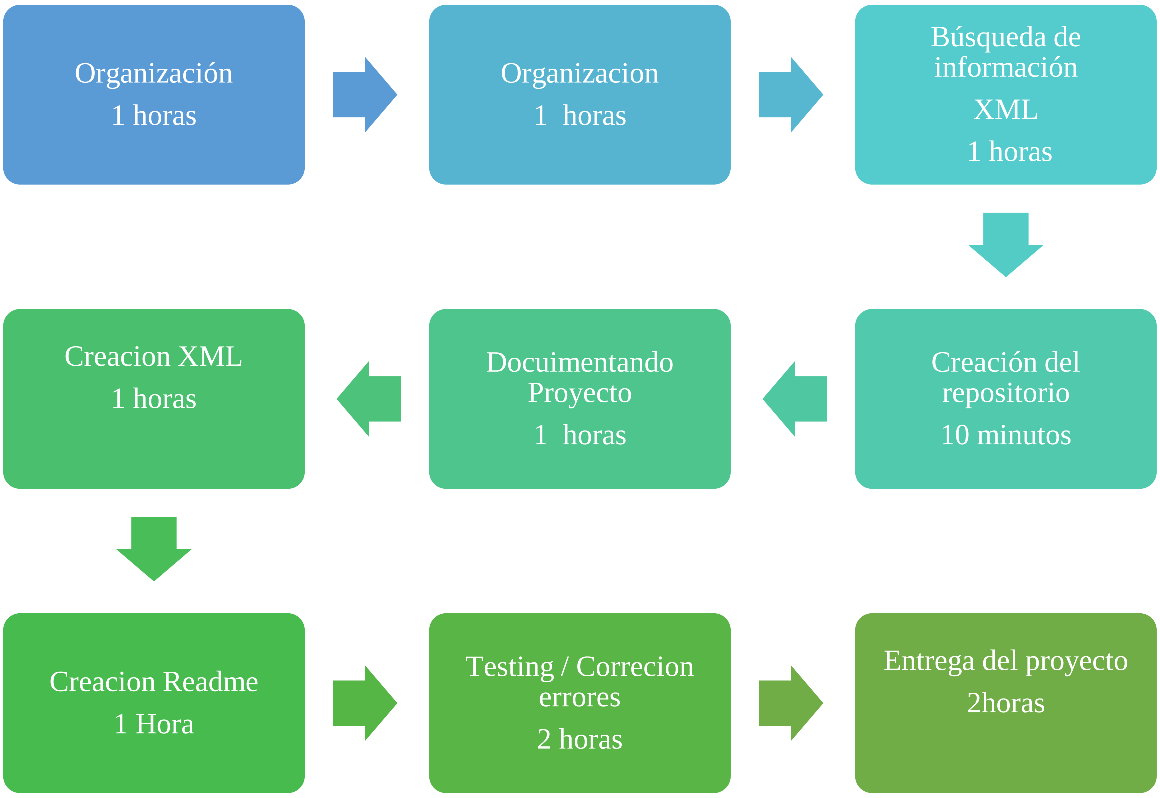
We plan to finish the project in 8 Hours

In which we give 8 more hours to anticipate incidents in the project

Incidence Record detected during the project

- No

Project Calendar Tracking



Risk Documentation

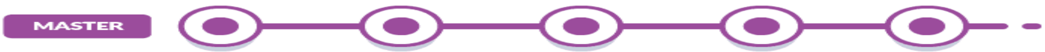
- Loss or damage of work material.

GIT WORKFLOW documentation

- Creating Https://github.com/robertfox11/XmlBasic.git Git Hub
- We make commits of the structure of the main page.
- Chance of it occurring 80%
- Project impact 60%
- Possible alternative (mitigation) Ask colleagues for help
- Chance of it occurring 30%
- Project impact 60%

- Possible alternative (mitigation) Ask colleagues for help
- Not easily finding information related to the project
- Chance of it occurring 30%
- Project impact 60%
- Alternative alternative (mitigation)
- Ask colleagues for help

From the realization of the structure, work continued only on the
"master" branch, through the Workflow "Gitflow".
But information --> https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow



Project tooling

Different tools were used in the development of the project. They are as follows:

1. ***git: A powerful version control system*** that helps track changes in the work tree.
2. ***Visual Studio Code: A code editor*** optimized for creating and debugging modern web applications.

Git workflow

All commits will be inserted into the master branch, following a personal criterion of loading only snapshots that are functional and working correctly, not counting minor errors. There are no other branches, as it would slow down the development process.
On the other hand, confirmation messages end with their primary purpose indicated in square brackets— for example.

- We'll use a restaurant.xml to create the tags
- mainly related to documentation, ***[documentation]***.

File structure

Project files will be organized as follows:

Record of lessons learned.

- The statement of the document