**DOM Xml**

1. **¿Qué es DOM Xml?:** Para poder hablar de DOM Xml primero debemos hablar de Xml, estas siglas significan “Extensible Markup Language”, es un Lenguaje similar a HTML, con la diferencia que el HTML está diseñado para mostrar información, mientras que el Xml es lo que almacena la información. A diferencia de HTML en Xml no hay etiquetas predefinidas, sino que el usuario define las etiquetas ya que al final del día solo es la información que el usuario define y es posible agregarle nueva información, modificar la información o incluso remover la información de una versión anterior del archivo Xml y esto no afectaría la funcionalidad de nuestra aplicación. Pero justo como todas las cosas el Xml también tiene un conjunto de reglas sobre cómo se debe de manipular, en este caso es el DOM, “Document Object Model”, y es una plataforma donde sin importar el Lenguaje en el que se haya programado cuya función es definir un estándar para accesar y actualizar el contenido, la estructura y el estilo de un documento de forma dinámica.

A continuación, vemos un ejemplo de un archivo XML estructurado usando la plataforma DOM:

<juego>

<titulo>MegaMan Zero 3</titulo>

<consola>Gameboy Advance</consola>

<review>El major juego de MegaMan de la vida! <3</review>

</juego>

1. **Uso DOM Xml en Python:** Debido a la naturaleza del curso debemos aprender a utilizar Xml en Python, para esto Python incluye el paquete de Xml que necesita que exista almenos un parser de Xml instalado en nuestro Python, afortunadamente Python ya nos incluye uno llamado “Expat” y esto incluye varios submódulos para poder manejar los archivos XML que son:
   * [xml.etree.ElementTree](https://docs.python.org/3/library/xml.etree.elementtree.html#module-xml.etree.ElementTree)
   * [xml.dom](https://docs.python.org/3/library/xml.dom.html#module-xml.dom):
   * [xml.dom.minidom](https://docs.python.org/3/library/xml.dom.minidom.html#module-xml.dom.minidom)
   * [xml.dom.pulldom](https://docs.python.org/3/library/xml.dom.pulldom.html#module-xml.dom.pulldom)
   * [xml.sax](https://docs.python.org/3/library/xml.sax.html#module-xml.sax)
   * [xml.parsers.expat](https://docs.python.org/3/library/pyexpat.html#module-xml.parsers.expat)

Nos centrarémos en los principales, siendo estos:

* + [xml.etree.ElementTree](https://docs.python.org/3/library/xml.etree.elementtree.html#module-xml.etree.ElementTree): Nos permite crear un documento de XML bien estructurado usando las reglas del DOM y nos permite hacerlo mediante sus métodos por ejemplo

Si usamos el archivo XML:

<?xml version="1.0"?>

**<data>**

**<curso** name="IPC2"**>**

**<creditos>**5**</creditos>**

**<semestre>**4**</semestre>**

**<codigo>**0771**</codigo>**

**</curso>**

**<curso** name="Lenguajes de Programacion"**>**

**<creditos>**5**</creditos>**

**<semestre>**4**</semestre>**

**<codigo>**0796**</codigo>**

**</curso>**

**<curso** name="Practicas Iniciales"**>**

**<creditos>**0**</creditos>**

**<semestre>**4**</semestre>**

**<codigo>**2025**</codigo>**

**</curso>**

**</data>**

Podemos importar su información usando el código:

**import** **xml.etree.ElementTree** **as** **ET**

tree = ET.parse('cursos\_sistemas.xml')

root = tree.getroot()

* [xml.dom.minidom](https://docs.python.org/3/library/xml.dom.minidom.html#module-xml.dom.minidom): Que nos trae una versión más simple que el DOM completo pero mientras que es más fácil manejarla en Python sí se debe de tener experiencia usando DOM, de lo contrario es mejor que el usuario utilice el Módulo de Element Tree.

Por ejemplo, para guardar un archivo XML usando Minidom tendríamos que hacerlo así:

**from** **xml.dom.minidom** **import** parse, parseString

dom1 = parse('c:**\\**temp**\\**Ejemplo.xml') *# parse an XML file by name*

datasource = open('c:**\\**temp**\\**Ejemplo.xml')

dom2 = parse(datasource) *# parse an open file*

dom3 = parseString('<Titulo>Ejemplo Chido ;v</Titulo>')

1. **XPath Python:** XPath es un un estándar para identificar y navegar en los nodos de un documento XML, XPath es una abreviación de “XML Path Language”, en Python para poder usarlo necesitamos del paquete de XML para Python y de su módulo Element Tree, que incluye métodos tales como “find” y “find all” para encontrar los elementos que necesitemos dentro de la estructura del XML por ejemplo:

Si tenemos esta estructura:

<Catalogo>

<Libros>

<Libro id="1" precio="7.95">

<Titulo>Do Androids Dream of Electric Sheep?</Titulo>

<Autor>Philip K. Dick</Autor>

</Libro>

<Libro id="5" precio="5.95">

<Titulo>The Colour of Magic</Titulo>

<Autor>Terry Pratchett</Autor>

</Libro>

<Libro id="7" precio="6.95">

<Titulo>The Eye of The World</Titulo>

<Autor>Robert Jordan</Autor>

</Libro>

</Libros>

</Catalogo>

Entonces podemos usar el siguiente código para obtener todos los libros:

Import xml.etree.cElementTree as ET

Tree = ET.parse(‘Ejemplo.xml’)

Tree.findall(‘Libros/Libro’)

O si queremos encontrar un solo libro usando el mismo archivo XML el Código sería el siguiente:

Import xml.etree.cElementTree as ET

Tree = ET.parse(‘Ejemplo.xml’)

Tree.findall(“Libros/Libro[Titulo=’The Colour of Magic’]”)