UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS Y SISTEMAS
LABORATORIO DE IPC 2

ACTIVIDAD	2	FECHA	06/02/2021
NOMBRE			CARNET
WILMER ESTUARDO VASQUEZ RAXON			201800678

xml.dom

El Modelo de Objetos del Documento, o «DOM» por sus siglas en inglés, es un lenguaje API del Consorcio *World Wide Web* (W3C) para acceder y modificar documentos XML. Una implementación del DOM presenta los documentos XML como un árbol, o permite al código cliente construir dichas estructuras desde cero para luego darles acceso a la estructura a través de un conjunto de objetos que implementaron interfaces conocidas.

El DOM es extremadamente útil para aplicaciones de acceso directo. SAX sólo te permite la vista de una parte del documento a la vez. Si estás mirando un elemento SAX, no tienes acceso a otro. Si estás viendo un nodo de texto, no tienes acceso al elemento contendor. Cuando desarrollas una aplicación SAX, necesitas registrar la posición de tu programa en el documento en algún lado de tu código. SAX no lo hace automáticamente.

Algunas aplicaciones son imposibles en un modelo orientado a eventos sin acceso a un árbol. Por supuesto que puedes construir algún tipo de árbol por tu cuenta en eventos SAX, pero el DOM te evita escribir ese código. El DOM es una representación de árbol estándar para datos XML.

El Modelo de Objetos del Documento es definido por el *W3C* en fases, o «niveles» en su terminología. El mapeado de Python de la API está basado en la recomendación del DOM nivel 2.

Las aplicaciones DOM típicamente empiezan al diseccionar (*parse*) el XML en un DOM. Cómo esto funciona no está incluido en el DOM nivel 1, y el nivel 2 provee mejoras limitadas. Existe una clase objeto llamada DOMImplementation que da acceso a métodos de creación de Document, pero de ninguna forma da acceso a los constructores (*builders*) de *reader*/*parser*/*Document* de una forma independiente a la implementación. No hay una forma clara para acceder a estos métodos sin un objeto Document existente.

En Python, cada implementación del DOM proporcionará una función getDOMImplementation(). El DOM de nivel 3 añade una especificación para Cargar (*Load*)/Guardar (*Store*), que define una interfaz al lector (*reader*), pero no está disponible aún en la librería estándar de Python.

Interfaz	Sección	Propósito
`DOMImplementation`	Objetos DOMImplementation	Interfaz para las implementaciones subyacentes.
`Node`	Objetos Nodo	Interfaz base para la mayoría de objetos en un documento.
`NodeList`	Objetos NodeList	Interfaz para una secuencia de nodos.
`DocumentType`	Objetos DocumentType	Información acerca de la declaraciones necesarias para procesar un documento.
`Document`	Objetos Documento	Objeto que representa un documento entero.
`Element`	Objetos Elemento	Nodos elemento en la jerarquía del documento.
`Attr`	Objetos Atributo	Nodos de los valores de los atributos en los elementos nodo.
`Comment`	Objetos Comentario	Representación de los comentarios en el documento fuente.
`Text`	Objetos Texto y CDATASection	Nodos con contenido textual del documento.
`ProcessingInstruction`	Objetos ProcessingInstruction	Representación de instrucción del procesamiento.

Xpath.dom

Xpath es un módulo que es parte de la librería xml.etree.ElementTree por lo general la misma se importa de la siguiente manera:

```
1 | import xml.etree.ElementTree as ET
```

Xpath provee una serie de expresiones para localizar elementos en un árbol, su finalidad es proporcionar un conjunto de sintaxis, por lo que debido a su limitado alcance no se considera un motor en sí mismo.

EJEMPLOS:
import xml.etree.ElementTree as ET root = ET.fromstring(docxml)
#1 Elementos de nivel superior root.findall(".")

#2 Todos los hijos de vecino o nietos de país en el nivel superior root.findall("./país/ vecino")

#3 Nodos xml con nombre='Guatemala' que sean hijos de 'año' root.findall(".//año/..[@nombre='Guatemala']")

#4 Nodos 'años' que son hijos de etiquetas xml cuyo nombre='Guatemala' root.findall(".//*[@nombre='Guatemala']/año")

#5 Todos los nodos 'vecino' que son el segundo hijo de su padre root.findall(".//vecino[2]")