Jorge Antonio Pérez Ordóñez 201900810

Actividad 2

DOM xml

El Document Object Model, o "DOM", es una librería que se utiliza para acceder y modificar documentos XML. Una implementación DOM presenta un documento XML como una estructura de árbol o permite que el código del cliente construya dicha estructura desde cero. Luego da acceso a la estructura a través de un conjunto de objetos que proporcionan interfaces bien conocidas.

El DOM es extremadamente útil para aplicaciones de acceso aleatorio. SAX solo le permite ver un fragmento del documento a la vez. Si está mirando un elemento SAX, no tiene acceso a otro. Si está mirando un nodo de texto, no tiene acceso a un elemento contenedor. Cuando escribe una aplicación SAX, necesita realizar un seguimiento de la posición de su programa en el documento en algún lugar de su propio código. SAX no lo hace por ti. Además, si necesita mirar hacia adelante en el documento XML, simplemente no tiene suerte.

Algunas aplicaciones son simplemente imposibles en un modelo impulsado por eventos sin acceso a un árbol. Por supuesto, podría construir usted mismo algún tipo de árbol en eventos SAX, pero el DOM le permite evitar escribir ese código. El DOM es una representación de árbol estándar para datos XML.

xPath xml

xml.etree.ElementTree es una librería de Python en la cual se encuentra el módulo Xpath 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 si mismo. Su función es extraer fácilmente partes del xml haciendo referencia a su ubicación nodal representada a forma de path, lo que nos hace una sintaxis familiarmente sencilla a la hora de construir un paser xml.

```
import xml.etree.ElementTree as ET

root = ET.fromstring(docxml)

# Elementos de nivel superior
root.findall(".")

# todos los hijos de neighbor o nietos de country en el nivel superior
root.findall("./country/neighbor")

# Nodos xml con name='Singapore' que sean hijos de 'year'
root.findall(".//year/..[@name='Singapore']")

# nodos 'year' que son hijos de etiquetas xml cuyo name='Singapore'
root.findall(".//*[@name='Singapore']/year")

# todos los nodos 'neighbor'que son el segundo hijo de su padre
root.findall(".//neighbor[2]")
```