Julio José Orellana Ruíz ----- 201908120

Actividad 2

DOM XML para Python

El Modelo de Objetos del Documento, o «DOM», es un lenguaje API del Consorcio World Wide Web (W3C) para acceder y modificar documentos XML.

Una implementación del DOM presenta los documento 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. 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.

Una vez que tengas un objeto del documento del DOM, puedes acceder a las partes de tu documento XML a través de sus propiedades y métodos. Estas propiedades están definidas en la especificación del DOM; está porción del manual describe la interpretación de la especificación en Python.

XPath XML para Python

La biblioteca ElementTree incluye herramientas para analizar XML usando APIs basadas en eventos y documentos, buscando documentos analizados con expresiones XPath, y creando nuevos o modificando documentos existentes.

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

```
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.

Ejemplos de uso de Xpath:

Ejemplo No.1 de xPath:

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]")

Como vemos nos permite 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 parser xml.

notas: las expresiones entre corchetes deben ser precedidas por un nombre de etiqueta, un asterisco u otro comodín. Las referencias a position deben ser precedidas por una etiqueta xml válida.

Ejemplo de uso de Xpath dentro de una sentencia xml:

<xpath expr="//field[@name]='is_done'" position="before">

<field name="date_deadline" />

</xpath>

Ejemplo No.2 XML DOM:

Documento xml para analizar

```
<?xml version="1.0"?>
- <empresa>
  - <empleado id="1">
      <nombre>José Ernesto</nombre>
      <username>jose</username>
      <password>321423</password>
    </empleado>
  - <empleado id="2">
      <nombre>Daniel Pérez</nombre>
      <username>dperez</username>
      <password>433543</password>
    </empleado>
</empresa>
```

Imprimiremos en la consola el nombre del empleado con ID = 1

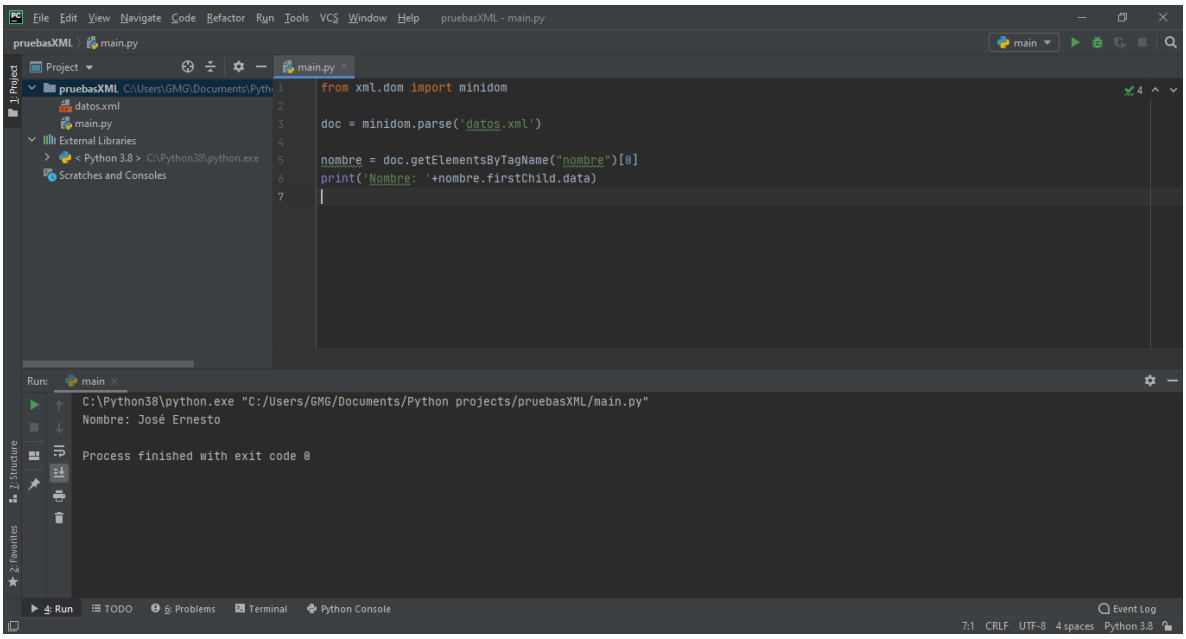
Solucion en codigo:

```
from xml.dom import minidom

doc = minidom.parse('datos.xml')

nombre = doc.getElementsByTagName("nombre")[0]
print('Nombre: '+nombre.firstChild.data)
```

solución en consola:



Ejemplo No.3 XML DOM:

Documento xml para analizar

```
<?xml version="1.0"?>
- <empresa>
  - <empleado id="1">
      <nombre>José Ernesto</nombre>
      <username>jose</username>
      <password>321423</password>
    </empleado>
  - <empleado id="2">
      <nombre>Daniel Pérez</nombre>
      <username>dperez</username>
      <password>433543</password>
    </empleado>
</empresa>
```

Imprimiremos todos los datos del XML y los etiquetaremos según su categoría

Solución en código:

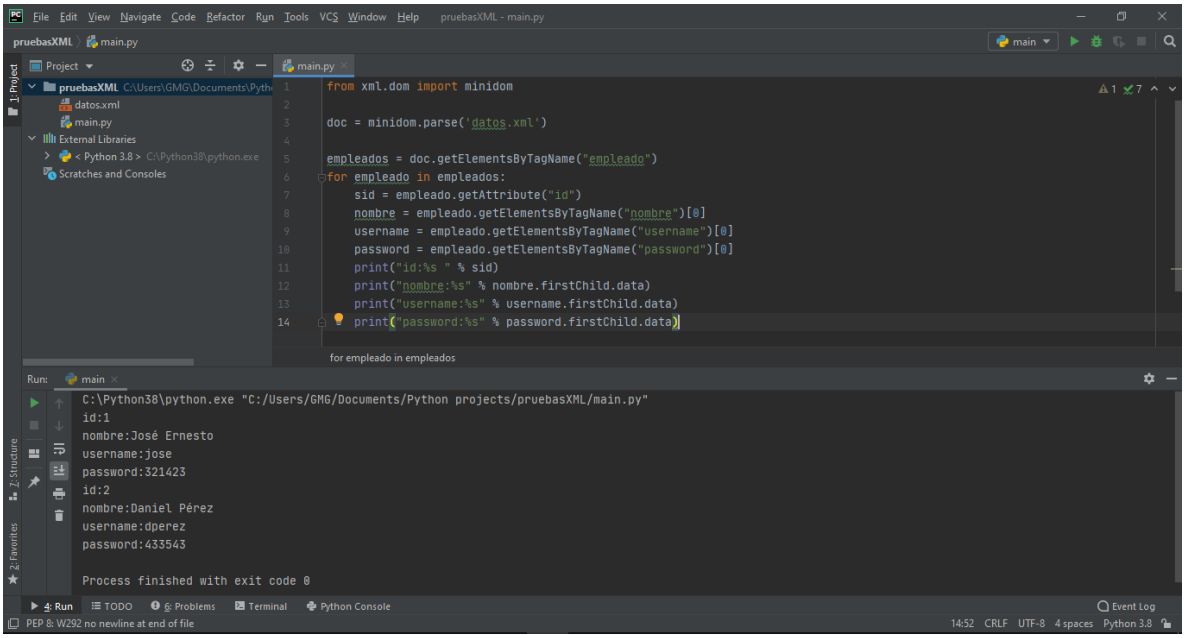
```
from xml.dom import minidom

doc = minidom.parse('datos.xml')

empleados = doc.getElementsByTagName("empleado")
for empleado in empleados:
    sid = empleado.getAttribute("id")
    nombre = empleado.getElementsByTagName("nombre")[0]
    username = empleado.getElementsByTagName("username")[0]
```

```
    password = empleado.getElementsByTagName("password")[0]
    print("id:%s " % sid)
    print("nombre:%s" % nombre.firstChild.data)
    print("username:%s" % username.firstChild.data)
    print("password:%s" % password.firstChild.data)
```

solucion mostrada en consola:



Ejemplo No. 4 XML Xpath

Documento xml a analizar:

```
<?xml version="1.0"?>
- <Catalog>
  - <Books>
    - <Book price="7.95" id="1">
        <Title>Do Androids Dream of Electric Sheep?</Title>
        <Author>Philip K. Dick</Author>
      </Book>
    - <Book price="5.95" id="5">
        <Title>The Colour of Magic</Title>
        <Author>Terry Pratchett</Author>
      </Book>
    - <Book price="6.95" id="7">
        <Title>The Eye of The World</Title>
        <Author>Robert Jordan</Author>
      </Book>
    </Books>
</Catalog>
```

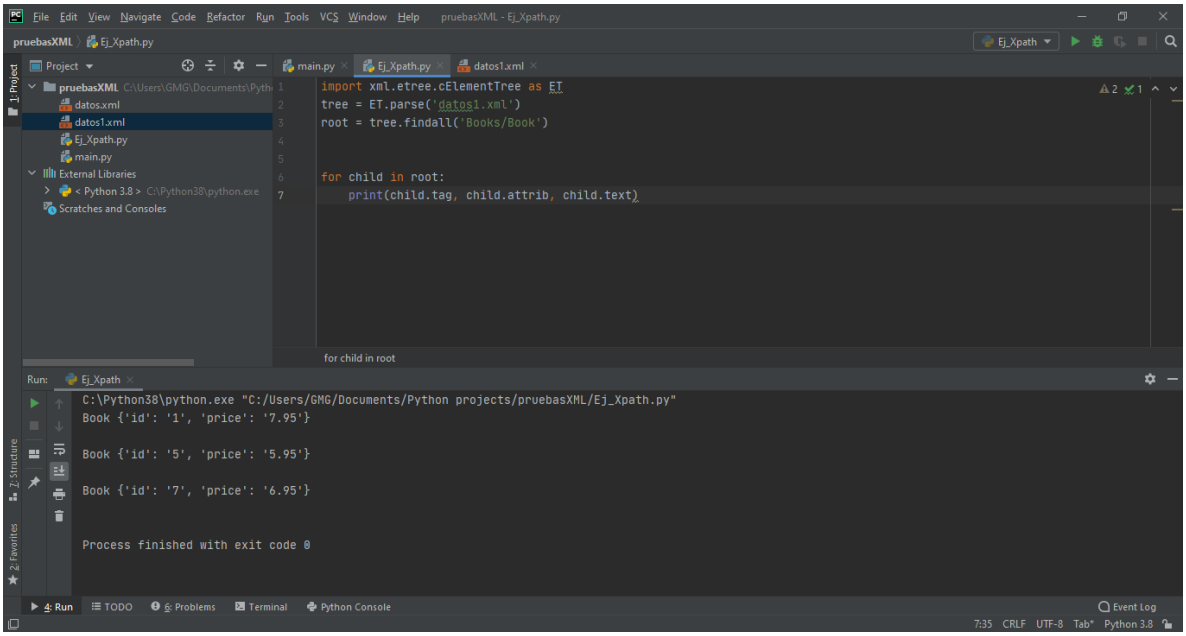
Se imprimirán todos los atributos y el nombre del elemento que pertenecerán, para este ejemplo se mostrara **Book**.

Solución en codigo:

```
import xml.etree.cElementTree as ET
tree = ET.parse('datos1.xml')
root = tree.findall('Books/Book')


for child in root:
   print(child.tag, child.attrib, child.text)
```

Solución en consola:



Ejemplo No. 5 XML Xpath

Documento xml a analizar:

```
<?xml version="1.0"?>
- <Catalog>
  - <Books>
    - <Book price="7.95" id="1">
        <Title>Do Androids Dream of Electric Sheep?</Title>
        <Author>Philip K. Dick</Author>
      </Book>
    - <Book price="5.95" id="5">
        <Title>The Colour of Magic</Title>
        <Author>Terry Pratchett</Author>
      </Book>
    - <Book price="6.95" id="7">
        <Title>The Eye of The World</Title>
        <Author>Robert Jordan</Author>
      </Book>
  </Books>
</Catalog>
```

Se imprimirán todos los elementos de una etiqueta BOOK en especial, el cual es: **The colour of magic** y lo mostraregos con su respectiva etiqueta.

Solución codigo:

```
import xml.etree.cElementTree as ET
tree = ET.parse('datos1.xml')
root = tree.find("Books/Book[Title='The Colour of Magic']")


for child in root:
   print(child.tag+":",  child.text)
```

Solución en consola