Brayan Alexander Mejia Barrientos
201900576
06/02/2021

Creación y lectura en lenguaje XML

Hoy en día se usa en el mundo de las integraciones (conversaciones entre aplicaciones) es un intermediario entre dos lenguajes, es muy utilizado en las aplicaciones móviles hoy en día en los smartphone o iphone guarda el login utilizado charsprefence es un archivo XML se queda almacenado en el teléfono para guardar cuentas.

EJEMPLO XML:

```
<root>
    <doc>
       <modo1 name = "nodo">Texto 0 </nodo1><nodo2 atributo = "manzana">texto 0</nodo2>
    </doc>
    <doc>
       <nodo1 name ="nodo">Texto 1</nodo1><nodo2 atributo = "manzana">texto 1</nodo2>
    </doc>
       <nodo1 name = "nodo">Texto 2 </nodo1><nodo2 atributo = "manzana">texto 2 </nodo2>
    </doc>
    <doc>
       <nodo1 name = "nodo">Texto 3 </nodo1><nodo2 atributo ="manzana">Texto 3</nodo2>
    </doc>
     <doc>
       <modo1 name = "nodo">Texto 4 </nodo1><nodo2 atributo = "manzana">texto 4</nodo2>
    </doc>
    <doc>
       <nodo1 name ="nodo">Texto 5</nodo1><nodo2 atributo = "manzana">texto 5</nodo2>
    </doc>
       <nodo1 name = "nodo">Texto 6 </nodo1><nodo2 atributo = "manzana">texto 6</nodo2>
    </doc>
    <doc>
       <nodo1 name = "nodo">Texto 7 </nodo1><nodo2 atributo ="manzana">Texto 7</nodo2>
    </doc>
```

Creación de un XML Python:

Las estructuras xml se crean a partir de el esquema de una árbol o ramificación

Y se utilizan ciertas librerías de Python como en el siguiente ejemplo:

```python
import xml.etree.cElementTree as ET

ruta = "C:\\Users\\braya_000\\Desktop\\XML\\" #ruta de la carpeta destino

root  = ET.Element("root")

i = 0
for numero in range(1,10):

    doc = ET.SubElement(root, "doc")
    ET.SubElement(doc,"nodo1", name="nodo").text = f"Texto nodo1 {i}"
    ET.SubElement(doc,"nodo2", atributo="manzana").text =f"Texto nodo2 {i}"

    i+=1

archivo = ET.ElementTree(root)
archivo.write(ruta+"newfile.xml")
```

Lectura de un XML Python:

Para poder leer un XML es necesario usar la librería dom y se debe de realizar un parseo para poder leer ese codigo obligatoriamete tiene que estar ramificado correctamente en formato XML

```python
from xml.dom import minidom

ruta = "C:\\Users\\braya_000\\Desktop\\XML\\ejemplo.xml"
print(ruta)
xml = minidom.parse(ruta)


docs = xml.getElementsByTagName("doc")


for doc in docs:
    nodo1 = doc.getElemntsByTagName("nodo1")[0]
    nodo2 = doc.getElemntsByTagName("nodo2")[0]
    print(f"nodo1={nodo.firstChild.data} | nodo1 ={nodo.firstChild.data}")
```

DOM (Document Object Model):

Características:

- Carga en memoria TODO el documento XML
- Estructura de árbol de Nodos
- Recorrido y navegación del árbol creado

Tipos de Nodos (hasta 12 tipos):

- Node (todos los objetos del árbol)
- Elements(etiquetas<>)
- Attr (ataributos) (Dentro de los nodos)
- Text(el texto entre dos etiquetas)
- Comment(comentarios del fichero)

Ventajas y desventajas sobre SAX:

Ventajas sobre SAX	Desventajas Sobre SAX
• Permite acceso directo a los elementos del documento • Permite crear y modificar los documentos sax	• Limitado por la cantidad de memoria

Ejemplos de XML DOM Python:

```
from xml.dom import minidom

ruta = "C:\\Users\\braya_000\\Desktop\\XML\\ejemplo.xml"
print(ruta)
xml = minidom.parse(ruta)


docs = xml.getElementsByTagName("doc")


for doc in docs:
    nodo1 = doc.getElemntsByTagName("nodo1")[0]
    nodo2 = doc.getElemntsByTagName("nodo2")[0]
    print(f"nodo1={nodo.firstChild.data} | nodo1 ={nodo.firstChild.data}")

```

```xml
<library>
    <book isbn ="50517387">
        <title>Core PHP</title>
        <date>December 2020</date>
        <author>Daniel Cohel</author>
    </book>
    <book isbn ="12356754">
        <title>JavaScript in Action</title>
        <date>Jun 2019</date>
        <author>Javier </author>
    </book>
    <book isbn ="98131520">
        <title>JQuery in short</title>
        <date>May 2018</date>
        <author>Jose Luis</author>
    </book>
    <book isbn ="191878201">
        <title>Core PHP</title>
        <date>December 2020</date>
        <author>Natalia </author>
    </book>

</library>
```

XML PATH

Características:

- Xpath es un lenguaje para localizar nodos en un documento de árbol
- Siete Xpath Nodos
- Root node
- Elemnt node
- Text node
- Attribute node
- Comment node
- Processing-instrucción node

Ejemplo:

```xml
<?xml version = "1.0" standalone = "no"?>
<people>
    <husband employed = "Yes">
        <name>Mark</name>
        <age>45</age>
         <wife>
             <wname>Janet</wname>
             <age>&marklong</age>
         </wife>
    </husband>
    <husband employed = "NO">
        <name>Matt</name>
        <age>42</age>
         <wife>
             <wname>Annie</wname>
             <age>&marklong:</age>
         </wife>
    </husband>
</people>
```

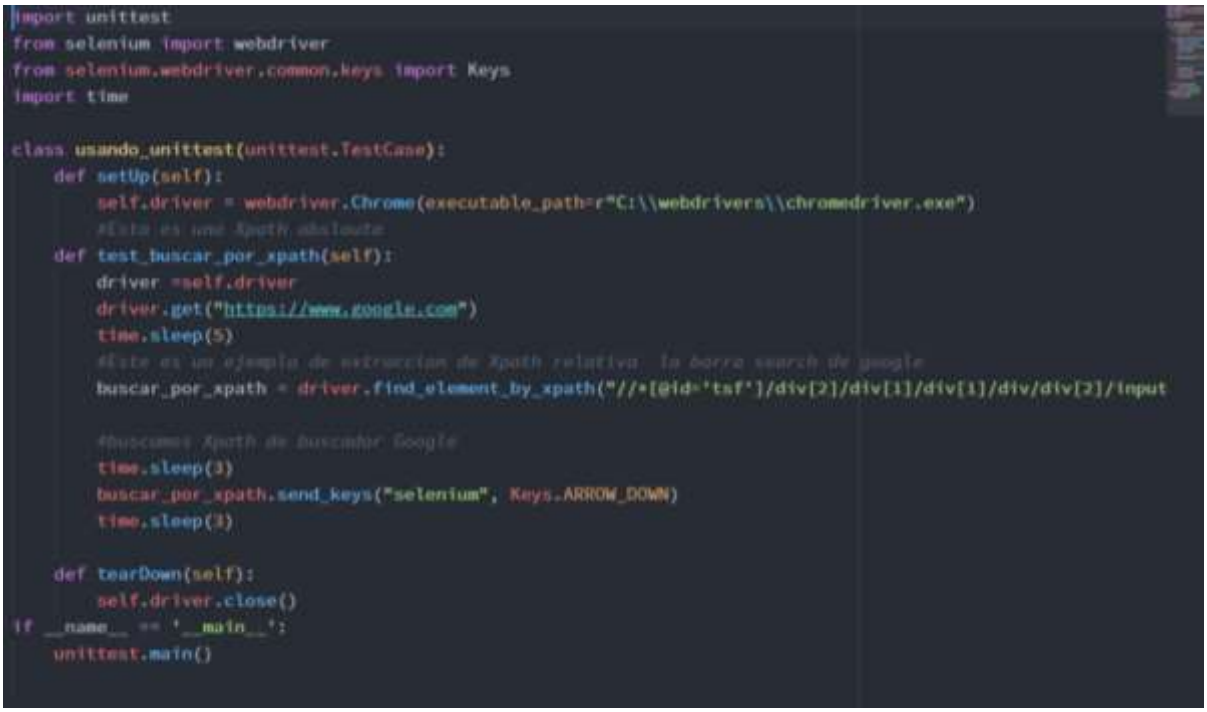
Python + Xpath = Extra Parsing Power

En automatizaciones se usa mucho el conceto de Xpath. Existen dos formas de buscar un Xpath Relativo y el Xpath absoluto .Una ruta o dirección de fichero contiene una estructura XML pero nosotros lo vemos como una ruta es una estructura con un árbol árbol de carpetas

XPATH: es una estructura de objetos.

XPATH relativo: Es la estructura por nodos nodos a a cada nivel del árbol y se utiliza el ultimo nivel es decir //ultima rama es la mas utilizada debido a que es fácil de cambiar a diferencia de la Xpath Absoluta. en el futuro cualquiera de los elementos web cuando se agregue o elimine puede cambiar su ruta Xpath absoluta entonces es recomendable usar Xpath relativas

XPATH Absoluto: Es la dirección exacta de un objeto esta se utiliza si y solo si sabemos que la ubicación no se cambiara. Utiliza la ruta completa desde el elemento raíz hasta el elemento deseado.



```
import unittest
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time

class usando_unittest(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome(executable_path=r"C:\\webdrivers\\chromedriver.exe")
        #Esta es una Xpath absoluta
    def test_buscar_por_xpath(self):
        driver =self.driver
        driver.get("https://www.google.com")
        time.sleep(5)
        #Este es un ejemplo de extraccion de Xpath relativa  la barra search de google
        buscar_por_xpath = driver.find_element_by_xpath("//*[@id='tsf']/div[2]/div[1]/div[1]/div/div[2]/input

        #buscamos Xpath de buscador Google
        time.sleep(3)
        buscar_por_xpath.send_keys("selenium", Keys.ARROW_DOWN)
        time.sleep(3)

    def tearDown(self):
        self.driver.close()
if __name__ == '__main__':
    unittest.main()
```