Python-Working with XML

Working with XML in Python

Python's `xml.etree.ElementTree` module provides a simple and easy-to-use API for parsing and generating XML files.

Example: Parsing an XML File

Suppose we have the following XML file, `data.xml`:

```
 <?xml version="1.0" encoding="UTF-8"?>
<catalog>
    <book id="bk101">
        <author>John Smith</author>
        <title>XML for Dummies</title>
        <genre>Computer</genre>
        <price>39.95</price>
        <publish_date>2000-10-01</publish_date>
        <description>The best book on XML.</description>
    </book>
    <book id="bk102">
        <author>Jane Doe</author>
        <title>XSLT for Dummies</title>
        <genre>Computer</genre>
        <price>29.95</price>
        <publish_date>2001-05-01</publish_date>
        <description>A great book on XSLT.</description>
    </book>
</catalog>
```

We can parse this file using the following Python code:

```
 import xml.etree.ElementTree as ET

# Parse the XML file
tree = ET.parse('data.xml')
root = tree.getroot()

# Print out all the book elements
for book in root.findall('book'):
    print(f"Book Title: {book.find('title').text}")
    print(f"Author: {book.find('author').text}")
    print(f"Genre: {book.find('genre').text}")
    print("")

# Accessing attributes
print(root.find('book')['id'])  # Output: bk101

# Modifying elements
root[0].find('price').text = '49.95'
tree.write('modified_data.xml')
```

In this example, we:

1. Import the `xml.etree.ElementTree` module and assign it a shorter alias, `ET`.
2. Parse the XML file using `ET.parse()`, which returns an `ElementTree` object.
3. Get the root element of the parsed tree using `tree.getroot()`.
4. Iterate over all the `book` elements in the tree using `findall()` and print out their title, author, and genre.
5. Access attributes using square brackets (`[]`) on a node (e.g., `root.find('book')['id']`).
6. Modify an element's text content using dot notation (e.g., `root[0].find('price').text = '49.95'`).

This is just a brief introduction to working with XML in Python. The `xml.etree.ElementTree` module provides many more features and methods for parsing, generating, and manipulating XML data.

Advice

- Use the `ET.parse()` method to parse an XML file.
- Use dot notation (e.g., `tree.root`) or square brackets (`[]`) to access nodes and attributes.
- Use `findall()`, `findtext()`, and other methods to query the tree for specific elements.
- Be mindful of XML namespace issues when working with external files.

Additional Resources

- Python documentation: xml.etree.ElementTree
- Real Python tutorial: Working with XML in Python

Curated by Brajesh Kumar