INFO-664-01 Programming For Cultural Heritage

XML

XML

- Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

- Standards that use XML

  - http://www.loc.gov/standards/mods/

  - http://www.loc.gov/ead/

XML

- There are more XML documents in the world than any other format, more than 65 billon, what are they?

- XML is popular in cultural heritage. A lot of people use Extensible Stylesheet Language Transformations (XSLT: https://en.wikipedia.org/wiki/XSLT) to modify and convert XML data.

- But we are going to use Python.

XML

- Tags or Elements:

  - `<title>Document [fair copy of the Declaration of Independence]</title>`

XML

- Attributes
  - `<dateCreated encoding="w3cdtf">1776-07-04</dateCreated>`

XML

- Nesting

```xml
<name type="personal" usage="primary">

    <namePart>United States. Congress, Continental, 1775–1789</namePart>

    <role>

        <roleTerm authority="marcrelator" type="code" valueURI="http://id.loc.gov/vocabulary/relators/cre">
            cre
        </roleTerm>

        <roleTerm authority="marcrelator" type="text" valueURI="http://id.loc.gov/vocabulary/relators/cre">
            Creator
        </roleTerm>
    </role>

</name>
```

XML - Python Access

```
<dateCreated encoding="w3cdtf">1776-07-04</dateCreated>
```

XML - Python Access

```
<mods:dateCreated>1776-07-04</dateCreated>
```

Prefix on the element tag

XML - Python Access

```
#Here we are importing the ElementTree Class from the xml.etree module
import xml.etree.ElementTree as etree

#ask the xml module to load the xml file and parse it
tree = etree.parse('class.xml')

#return the root xml element and store it into the root variable
root = tree.getroot()

```

```
#using a for loop we can loop through the root element
for a_element in root:

    print(a_element.tag)
    print(a_element.attrib)
    print(a_element.text)

```

XML - Python Access

```
29  #using a for loop we can loop through the root element
30  for a_element in root:
31
32      print(a_element.tag)
33      print(a_element.attrib)
34      print(a_element.text)
35
```

XML - Exercise

- http://digitalcollections.nypl.org/items/510d47e3-d9ee-a3d9-e040-e00a18064a99
- declaration.xml

- Print the text of the <abstract> element

- Print all of the <namePart> of the documents

```
1
2  #instead of the etree class we are loading 3 other classes, subelement and element and ElementTree
3  from xml.etree.ElementTree import Element, SubElement, ElementTree
4
```

We are going to use the Element and SubElement to create XML elements and write it to file with ElementTree

XML - Writing

We define a root element

```
4
5  #this is the root element, we are calling it rootelement, for no reason
6  root = Element('rootelement')
7
```

XML - Writing

And add a child element to it

```
 9  #make a child of root, and addd it to the root element
10  child = SubElement(root, 'childchild')
```

XML - Writing

Now you can set attributes and text values

```
14  child.text = 'This child contains text.'
15
16  #now set an attribute
17  child.set("type","firstchild")
```

```
21  #write it out to file
22  ElementTree(root).write("text.xml")
```

Write it to file.

XML - Challenge - Option 1

- http://legacy.www.nypl.org/research/chss/spe/rbk/soldiers/index.html

- Problem: We have a CSV file dump of this boutique website, we need to convert it into a machine readable XML to put into the finding aid. http://archives.nypl.org/mss/19877

- http://www.loc.gov/ead/tglib/elements/index-element.html

- Your Mission: Turn the CSV into EAD <index> XML

XML - Challenge!

0 = id
1 = last name
2 = first name
3 = other name
4 = unit number
5 = unit state
6 = unit type
7 = unit letter
8 = rank
9 = additional info
10 = alt info
11 = year
12 = box number
13 = folder number

The column layout of mss_soldiers.csv

XML - Challenge!

What the output of the XML file should have in it, things in the [brackets] are meta-comments.

```
<index>

    <indexentry>

        <name> [solider name here]            </name>
        <note> [all of the extra data here]   </note>
        <ref>  [the box and the folder here] </ref>

    </indexentry>

    [repeat indexentry elements for each name]

</index>
```

XML - Challenge - Option 2

- Use the CSV from last week to build a HTML page.
- Loop through your CSV and build an HTML document for some of the data.