Digital Approaches to Historical Inquiries
4th Class
Gifts to federal employees from foreign sources

• Title of Nobility Clause (U.S. Constitution, Article I, Statute 9):
  “[N]o Person holding any Office of Profit or Trust under them, shall, without the Consent of the Congress, accept of any present, Emolument, Office, or Title, of any kind whatever, from any King, Prince, or foreign State.”

• Foreign Gifts and Decorations Act (1966; 5 USC 7342)

• Reporting system for gifts over certain minimum threshold.
  • Reported by the Office of Protocol (Department of State) in the Federal Register
What makes Historical Data(sets) special?

- Proxy
  - Dissimilar sources
- Fragmented
  - Sparse
- Interdependent (with time)
  - Multi-scalar/geo-locations unknown

A Möbius strip (wikimedia)

Magritte (1928/1929)

Fragment of P52
(earliest record of New Testament; wikimedia)
Proxies, Phenomena and Abstractions
Gift Diplomacy

Heuristic Process

Gift Diplomacy → abstraction → political entities interact; goods are transacted → representation → Gifts to Federal Employees Register
Gift Diplomacy

abstraction

Dissimilar Sources

political entities interact; goods are transacted

representation

Dissimilar Sources
Fragmented
Interdependent

• Temporal:
  • Phenomenon takes place in time
    • Datestamp?
    • What system?
    • Contextual?
  • Phenomenon takes place over time
    • How to trace the phenomenon longitudinally?
    • Snapshots?
    • Dynamic?
    • Multi-levelled?

• Geographically

• Linguistically

• Other phenomena
  • Diplomatic Gift Giving and Economy or Identity or Popularity or…
Markup languages

• Describe data, but “don’t do anything” (not a programming language)

• Many different types:
  • HTML
  • Tex
  • Wiki-markup

• XML
  • eXtensible: can be extended (not a single pre-defined markup language)
  • Markup: from marking up a document, with <tags>…</tags>
  • Language: both human and machine readable, based in the Standard Generalized Markup Language (ISO 8879:1986)

• Authored by the World Wide Web Consortium, learn all about XML in their tutorial.

• XML is a widely used format for data interchange
  • Comparable to JSON, but not the same
XML

eexample adapted from Python’s ElementTree documentation

```xml
<?xml version="1.0" encoding="UTF-8"?>
<data>
    <country name="Liechtenstein">
        <rank>1</rank>
        <year>2008</year>
        <gdppc>141100</gdppc>
    </country>
    <country name="Singapore">
        <rank>4</rank>
        <year>2011</year>
        <gdppc>59900</gdppc>
    </country>
    <country name="Panama">
        <rank>68</rank>
        <year>2011</year>
        <gdppc>13600</gdppc>
    </country>
</data>
```

xml prolog: `<?xml version="1.0" encoding="UTF-8"?>`

tag: `<data>`

element: `<data> ... </data>`

attribute: `name`

attribute value: `Liechtenstein`

text: `1`
Step - 1: learn the basics of Python

• This how-to pre-supposes you understand the basics of Python, such as:
  • What variables are
    • What variable types are (e.g. integer, string)
  • What functions are
  • How to make for-loops
  • What an if-statement is
  • What a dictionary is in Python

• If you don’t, you can still do this how-to, but you may not understand (some of) the code and why we use it.

• There are a LOT of Python tutorials out there
  • Learn Python!
    • Learn Python 3!
  • I learned Python from Dr. Chuck.
    • www.py4e.com
Step 0: download the how-to folder, extract it to your workfolder, and access your files folder with Anaconda.

- You can find the files for this tutorial at www.shoresoftime.com/dighist in the schedule for the 4th class.
- Unzip them to a folder of your choice in your P: drive (tip: keep the path short)
- Open Anaconda console and navigate to the location

```bash
p:  #navigate to your p: drive
cd  #access folder
dir # see contents of folder
cd .. # go up one folder
cls # clear console screen
python helloworld.py # run a python program in Anaconda
```

- Tip: anything following # in Python is a “comment”, it will be ignored by the computer when reading through your program.
- Test it yourself:
  - Run helloworld.py
Step 1: access the xml-tree on the web

```python
import urllib.request
url = 'http://www.shoresoftime.com/ObamaGifts.xml'
response = urllib.request.urlopen(url)
```

• All these steps can be found in the how-to folder, e.g. step1.py

• Try it yourself:
  • Open the .py file for editing
  • Write a function that will allow you to print the contents of response to the Anaconda console to see what you downloaded
    • hint: the python command to print to console is `print()`
  • Run the program

• NB Python or Anaconda does not have a built in virus scanner, so make sure, before you download anything, you can trust its source.
Step 2: save the xml-tree

```python
import urllib.request

url = 'http://www.shoresoftime.com/ObamaGifts.xml'
response = urllib.request.urlopen(url)
contents = response.read()
file = open("ObamaGifts.xml", 'wb')
file.write(contents)
file.close()
```

• Run the program
• NB this is a very long route to take to save a xml file, we could also have navigated to the page and simply saved it via the browser. Still, it is good to have a bit of practice for when you will need to automate webpage accessing and downloading.
Step 3: Open the tree

```python
import xml.etree.ElementTree as etree
with open('ObamaGifts.xml', 'rb') as xml_file:
    tree = etree.parse(xml_file)
root = tree.getroot()
```
Step 4: Look inside the tree

import xml.etree.ElementTree as etree
with open('ObamaGifts.xml', 'rb') as xml_file:
    tree = etree.parse(xml_file)

root = tree.getroot()

for child in root:
    print (child.tag)

• Run program

• Try it yourself:
  • Print the attribute of the GIFTS child
    • Hint: the abbreviation used here is attrib
import xml.etree.ElementTree as etree
with open('ObamaGifts.xml', 'rb') as xml_file:
    tree = etree.parse(xml_file)
root = tree.getroot()
for element in tree.iter():
    print (element.tag)

• Run program
• Try it yourself:
  • Print the tag and the text of the elements
    • Hint: you can separate variables (or strings) to print with a ,
      e.g. print (x, y)
Step 6: Cut the tree in pieces (it won’t hurt)

```python
import xml.etree.ElementTree as etree
with open('ObamaGifts.xml', 'rb') as xml_file:
    tree = etree.parse(xml_file)

root = tree.getroot()

for child in root:
    print (child.tag)
    for attribute, value in child.attrib.items():
        if value != "2016": continue
    for subchild in child.iter():
        print (subchild.tag, subchild.text)
```

• Run program
• Try it yourself:
  • Change the != operator to ==, what happens?
import xml.etree.ElementTree as etree
with open('ObamaGifts.xml', 'rb') as xml_file:
    tree = etree.parse(xml_file)

root = tree.getroot()
giftevent = 0
gifts = root.findall("./GIFT")

for gift in gifts:
    if "Barack Obama" in gift[0].text: giftevent = giftevent + 1

    giftevent = str(giftevent)

print("Barack Obama had " + giftevent + " foreign gift parties during his presidency.")

• Run program
• Try it yourself:
  • How many gift events did Hillary have during the same time?
    • Tip: she is the only Hillary in this database
Assignment III

- Remember: these federal employees are not even supposed to accept gifts from foreign sources!
- End goal of this assignment is to have an exhaustive list of all the reasons these people have for still accepting gifts.
- Aim is to be able to extract information from an XML file as well as find solution to a programming question online.
- Spend 2 hours on this
- By next Tuesday, hand in, via a direct message to me in Slack:
  - your solution/the current output of your script (even if it is an error)
  - your script (even if it does not function)
  - a short overview of how you got to where you did (resources used, alternative solutions tried, etc.)
See you next week!