

## Metadata gathering using Python & XPath

Install Python & Lxml (these are the versions which we are using): <http://www.python.org/download/releases/2.6.2/> & <http://pypi.python.org/pypi/lxml/2.2.2>

Now use your favorite Python and Xpath development environment (we use the eclipse Java EE IDE <http://www.eclipse.org/downloads/> with PyDev plugin for Python development <http://pydev.sourceforge.net/updates>)

You will most likely want to be able to test your XPath queries as you develop – FirePath (<https://addons.mozilla.org/en-US/firefox/addon/11900/>) an extension of FireFox’s FireBug (<https://addons.mozilla.org/en-US/firefox/addon/1843/>) is one possible tool you could use.

Some helpful references sites to get you started:

Python Tutorial: <http://docs.python.org/tutorial/>

LXML: <http://codespeak.net/lxml/tutorial.html>

XPATH: <http://oreilly.com/catalog/xmlnut/chapter/ch09.html> & <http://www.caucho.com/resin-3.0/xml/xpath.xtp>

REGEX: <http://docs.python.org/library/re.html>

### **Task:**

Write a Python script that goes to the OCC (Office of the Comptroller of the Currency) Interpretations and Actions site: <http://www.occ.gov/topics/licensing/interpretations-and-actions/index-interpretations-and-actions.html> and gathers the following information for the months of March 2010, August 2001 and May 1996:

For each live link in the tables, gather the Letter#, Topic, Date (taken from the Topic) & the href (linked by Letter#). You will need to list them in the result file under their particular Category – for example: in August 2001, letters 911, 912 & 913 should be listed underneath the category “Interpretive Letters”, while letter 109’s category would be “CRA Decisions”. Your source code should use XPath (preferably) and/or Regex to find the Category as well all the other required metadata.

Submit the result file with the gathered data plus your Python source code which harvested it.

Sample python code to start:

```
import re
import urllib2
import urlparse
import lxml.html as html
import lxml.etree as etree
import traceback

START_URL = "http://www.occ.gov/topics/licensing/interpretations-and-actions/index-interpretations-and-actions.html"

if __name__ == '__main__':
    """
    This example code gets html from the top level web page, converts it to an xml tree,
    and uses xpath to select and print the relevant elements
    """
    find_links = etree.XPath("//div [@id='rxbbody']//a")
    try:
        # get html url and convert it to an etree using the most excellent lxml library
        (http://codespeak.net/lxml)
        rawhtml = urllib2.urlopen(START_URL).read()
        xmlTree = etree.fromstring(rawhtml, parser=html.HTMLParser(recover=True, remove_comments=True))

        # find relevant links and print them to stdout
        for link in find_links(xmlTree):
            href = urlparse.urljoin(START_URL, link.get("href"))
            print link.text, href
    except:
        print("Exception while parsing: %s\n"%traceback.format_exc())
```