Bioinformatics Web Resources
NCBI / EBI / Uniprot / Pfam

Biol4230    Thurs, Feb 15, 2018
Bill Pearson  wrp@virginia.edu  4-2818  Pinn 6-057

- Recognizing web addresses (URLs)
- NCBI – eutils: esearch/efetch/blast search
  www.ncbi.nlm.nih.gov/books/NBK25501/
- EBI – web services
  www.ebi.ac.uk/Tools/webservices/
- Uniprot ID mapper
  www.uniprot.org/faq/28#id_mapping_examples
- Pfam – using XML data
  pfam.xfam.org/help#tabview=tab10
  xml.etree.ElementTree

To learn more:

- Each of the web resources outlined has a help/FAQ page on downloading content
- Homework, due Monday, Feb. 19

See: fasta.bioch.virginia.edu/biol4230/labs/accessions_hwk5.html

Questions include:
1. Do a text search at the NCBI and download all the human refseq protein accessions for “GSTM”
   - also store protein lengths (see NCBI XML slides)
2. Map each of the Refseq accessions to Uniprot accessions at the Uniprot ID mapping site
   - are all the human proteins present in Uniprot?
   - are the mapped proteins the same length?
   - are the mapped proteins the same identical sequence?
3. Look up the domain content for each of the Uniprot accessions in Pfam
   - For each of the human proteins that can be mapped to Uniprot and Pfam, how many of the proteins have Pfam domains that are less than 50% of the Pfam family model length?
URLs – HTTP requests
(how does the Web know what you want?)

• HTTP GET/PUT:
  http://fasta.bioch.virginia.edu/   web site
  fasta_www2/fasta_www.cgi?rm=select&pgm=fap
  script location/name.cgi
  beginning of arguments: ?arg1=value1
  separator
  arg2=value2

esearch.fcgi
?db=protein&term=GSTM*&rettype=uilist&retmax=1000
?arg1=val1&arg2=val2&arg3=val3 (no spaces)

Information from the NCBI – eutils


• ESearch: responds to a text query with the list of UIDs matching the query in a given database, along with the term translations of the query.
• ESummary: responds to a list of UIDs with the corresponding document summaries.
• EFetch: responds to a list of UIDs with the corresponding data records.
  [reference documentation]
• ELink: responds to a list of UIDs in a given database with either a list of related IDs in the same database or a list of linked IDs in another Entrez database.
• EGQuery: responds to a text query with the number of records matching the query in each Entrez database.

Getting NCBI data using eutils is ALWAYS requires accessons:
1. get a set of accessions with ESearch
2. retrieve the data using the list of accessions (EFetch)
How to find data: NCBI

www.ncbi.nlm.nih.gov/books/NBK25500/
www.ncbi.nlm.nih.gov/books/NBK25497/

**ESearch (text searches)**

eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi

Responds to a text query with the list of matching UIDs in a given database (for later use in ESummary, EFetch or ELink), along with the term translations of the query.

**EPost (UID uploads)**

eutils.ncbi.nlm.nih.gov/entrez/eutils/epost.fcgi

Accepts a list of UIDs from a given database, stores the set on the History Server, and responds with a query key and web environment for the uploaded dataset.

**ESummary (document summary downloads)**

eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi

Responds to a list of UIDs from a given database with the corresponding document summaries.

**EFetch (data record downloads)**

eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi

Responds to a list of UIDs in a given database with the corresponding data records in a specified format.

---

NCBI esearch.fcgi

   <Count>41</Count> <RetMax>41</RetMax> <RetStart>0</RetStart>
   <IdList>
      <Id>NP_001278092.1</Id>
      <Id>NP_001129490.1</Id>
      <Id>NP_758859.1</Id>
      <Id>NP_000843.1</Id>
      <Id>NP_000111.1</Id>
      ...
      <Id>NP_000842.2</Id>
      <Id>NP_000841.1</Id>
      <Id>NP_671489.1</Id>
      <Id>NP_651401.1</Id>
      <Id>NP_714543.1</Id>
   </IdList>
   <TranslationSet><Translation><From>human[organism]</From><To>“Homo sapiens”[Organism]"></Translation>
   <TranslationStack><TermSet><Term>gstm</Term><Field>All Fields</Field><Count>43</Count>...
      <Term>gstmu2[All Fields] OR gstmu3[All Fields]</Term>
   </QueryTranslation></QueryTranslationStack>
</esearchResult>
#!/bin/env python

from urllib2 import urlopen
import re

# import pdb; pdb.set_trace()

# setup URL

s_args = "db=protein&term=GSTM*+AND+human[organism]+" 
"+AND+srcdb_refseq[prop]&idtype=acc&retmax=1000";

acc_lines = urlopen(s_url+s_args).readlines() # get results

acc_re = re.compile(r'<Id>(\w+\.)+</Id>')  # setup RE to get ID’s

acc_ids = [ m.group(1) for l in acc_lines if acc_re.search(l) if m ]

#for id in acc_ids:
#    print id

# now we have a list of acc’s, get the sequences


seq_args = "db=protein&id=",".join(acc_ids) + "+&rettype=fasta"

seq_html = urlopen(seq_url + seq_args).read()

print seq_html


<table>
<thead>
<tr>
<th>Record Type</th>
<th>&amp;rettype</th>
<th>&amp;retmode</th>
</tr>
</thead>
<tbody>
<tr>
<td>text ASN.1</td>
<td>null</td>
<td>asn.1, default</td>
</tr>
<tr>
<td>XML</td>
<td>null</td>
<td>xml</td>
</tr>
<tr>
<td>Gene table</td>
<td>gene_table</td>
<td>text</td>
</tr>
<tr>
<td>text ASN.1</td>
<td>null</td>
<td>text, default</td>
</tr>
<tr>
<td>binary ASN.1</td>
<td>null</td>
<td>asn.1</td>
</tr>
<tr>
<td>Full record in XML</td>
<td>native</td>
<td>xml</td>
</tr>
<tr>
<td>FASTA</td>
<td>fasta</td>
<td>text</td>
</tr>
<tr>
<td>Accession</td>
<td>acc</td>
<td>text</td>
</tr>
<tr>
<td>text ASN.1</td>
<td>null</td>
<td>asn.1, default</td>
</tr>
<tr>
<td>XML</td>
<td>null</td>
<td>xml</td>
</tr>
<tr>
<td>MEDLINE</td>
<td>medline</td>
<td>text</td>
</tr>
<tr>
<td>PMID list</td>
<td>ulist</td>
<td>text</td>
</tr>
<tr>
<td>Abstract</td>
<td>abstract</td>
<td>text</td>
</tr>
<tr>
<td>XML</td>
<td>null</td>
<td>xml, default</td>
</tr>
<tr>
<td>TaxID list</td>
<td>ulist</td>
<td>text or xml</td>
</tr>
</tbody>
</table>
How to find data: EBI/EMBL
www.ebi.ac.uk/Tools/webservices/

Web Services at the EBI

Introduction
The EMBL-EBI provides programmatic access to various data resources and analysis tools via Web Services technologies.

Web Services is an integration and inter-operation technology, to ensure client and server software from various sources will work well together, the technology is built on open standards:

- Representational state transfer (REST): a software architecture style.
- Web Services Description Language (WSDL): a method for describing Web Services and their capabilities.

For the transport layer Web Services utilise common network protocols, generally the Hypertext Transfer Protocol (HTTP).

For an overview of Web Services technologies and short tutorials on using common programming languages and Web Services tool-kits see Introduction to Web Services.

<table>
<thead>
<tr>
<th>Service</th>
<th>Clients</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ArrayExpress</td>
<td>Microarray data searching with ArrayExpress.</td>
<td></td>
</tr>
<tr>
<td>CHEBI Web Services</td>
<td>CHEBI Web Services</td>
<td>Entry retrieval from the CHEBI database.</td>
</tr>
<tr>
<td>CHEMbl Web Services</td>
<td>CHEMbl Web Services</td>
<td>Search data in, and retrieve data from the CHEMbl database.</td>
</tr>
<tr>
<td>EBI-Eye</td>
<td>EBI-eye</td>
<td>Database search using the EBI-eye search engine.</td>
</tr>
<tr>
<td>ENA Browser</td>
<td></td>
<td>Retrieval of sequence and associated records from ENA.</td>
</tr>
<tr>
<td>Gene Expression Atlas API</td>
<td></td>
<td>Enriched database of summary statistics over a curated subset of ArrayExpress Archive.</td>
</tr>
<tr>
<td>MrnService</td>
<td></td>
<td>Database search and data retrieval using BioMart.</td>
</tr>
<tr>
<td>PSICQUIC</td>
<td></td>
<td>Standardised access to molecular interaction databases, including CHEMbl, Reactions and IMAC.</td>
</tr>
<tr>
<td>Rhea</td>
<td></td>
<td>Manually annotated database of chemical reactions.</td>
</tr>
<tr>
<td>SRS</td>
<td></td>
<td>Database search and data retrieval using SRS@EBI.</td>
</tr>
<tr>
<td>UniProt.org</td>
<td></td>
<td>The Universal Protein Resource (UniProt) a comprehensive resource for protein sequence and annotation data.</td>
</tr>
<tr>
<td>WildOwltech (REST)</td>
<td></td>
<td>Identifier based entry retrieval for various up-to-date biological databases.</td>
</tr>
<tr>
<td>WildOwltech (SOAP)</td>
<td></td>
<td>Identifier based entry retrieval for various up-to-date biological databases.</td>
</tr>
</tbody>
</table>
How to find data: EBI/EMBL
www.ebi.ac.uk/Tools/webservices/

<table>
<thead>
<tr>
<th>REST Service</th>
<th>SOAP Service</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FASTA (REST)</td>
<td>FASTA (SOAP)</td>
<td>FAST protein or nucleotide comparison using the FASTA suite. Includes Smith and Waterman local-local (SSSEARCH), global-local (GLSEARCH) and global-global (GGSEARCH) alignment searches.</td>
</tr>
<tr>
<td>FASTM (REST)</td>
<td>FASTM (SOAP)</td>
<td>Peptide fragment searches using the FASTF, FASTM or FASTS programs from the FASTA suite.</td>
</tr>
<tr>
<td>NCBI BLAST (REST)</td>
<td>NCBI BLAST (SOAP)</td>
<td>Compare a sequence with those contained in nucleotide and protein databases using NCBI BLAST.</td>
</tr>
<tr>
<td>PSI-BLAST (REST)</td>
<td>PSI-BLAST (SOAP)</td>
<td>Position Specific Iterative BLAST (PSI-BLAST), guided mode</td>
</tr>
<tr>
<td>PSI-Search (REST)</td>
<td>PSI-Search (SOAP)</td>
<td>Iterative Smith and Waterman using a PSI-BLAST strategy</td>
</tr>
<tr>
<td>WU-BLAST (REST)</td>
<td>WU-BLAST (SOAP)</td>
<td>Compare a novel sequence with those contained in nucleotide and protein databases using WU-BLAST.</td>
</tr>
</tbody>
</table>

From alignments to domains, How to get from RefSeq to Pfam?

- Pfam uses Uniprot Id’s and Uniprot accession numbers:
  >sp|043708|MAA1_HUMAN Maleylacetoacetate isomerase GN=GST21 PE=1 SV=3MQAGP ILYYFRRSCGWRVRALALEKIDYVTVPINLIDRQGQFSDKFDQALNPHQVIPKIDGITYIQSLA IIEYLEDIPTRPRLQGPPRXAVSMHGQLAGIGIQYQMNLYVLSKQGSEMGIWNQAINITOSKHSMNOQYGIYQZQI YCVGEVTMDCLVTPQVANAERFKDLTQPYSTISINRRKLVLLEAPQVREPCQGPQPTRPETRA

- UniProt provides a utility for mapping from other accession numbers to UniProt accessions/ids
  
  http://www.uniprot.org/faq/28#id_mapping_examples
Mapping to/from UniProt accessions
https://www.uniprot.org/help/api_idmapping

<table>
<thead>
<tr>
<th>UniProt</th>
<th>Abbreviation</th>
<th>Direction</th>
<th>UniProtKB AC/ID</th>
<th>ACC+ID</th>
<th>from</th>
</tr>
</thead>
<tbody>
<tr>
<td>UniProtKB AC</td>
<td>ACC</td>
<td>to</td>
<td>DNA</td>
<td>EMBL</td>
<td>both</td>
</tr>
<tr>
<td>UniProtKB ID</td>
<td>ID</td>
<td>to</td>
<td>DNA CDS</td>
<td>EMBL</td>
<td>both</td>
</tr>
<tr>
<td>UniParc</td>
<td>UPARC</td>
<td>both</td>
<td>PIR</td>
<td>PIR</td>
<td>both</td>
</tr>
<tr>
<td>UniRef50</td>
<td>NF50</td>
<td>both</td>
<td>UniGene</td>
<td>UNIGENE ID</td>
<td>both</td>
</tr>
<tr>
<td>UniRef90</td>
<td>NF90</td>
<td>both</td>
<td>Entrez Gene</td>
<td>P_ENTREZGENEID</td>
<td>both</td>
</tr>
<tr>
<td>UniRef100</td>
<td>NF100</td>
<td>both</td>
<td>GI number*</td>
<td>P_GI</td>
<td>both</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IPI</td>
<td>P_IPI</td>
<td>both</td>
</tr>
<tr>
<td>RefSeq</td>
<td>P_REFSEQ_AC</td>
<td>both</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Other sequence databases

<table>
<thead>
<tr>
<th>Name</th>
<th>Abbreviation</th>
<th>Direction</th>
<th>DNA</th>
<th>EMBL</th>
<th>both</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNA</td>
<td>EMBL</td>
<td>both</td>
<td>PIR</td>
<td>PIR</td>
<td>both</td>
</tr>
<tr>
<td>DNA CDS</td>
<td>EMBL</td>
<td>both</td>
<td>PIR</td>
<td>PIR</td>
<td>both</td>
</tr>
<tr>
<td>UniGene</td>
<td>UNIGENE_ID</td>
<td>both</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entrez Gene</td>
<td>P_ENTREZGENEID</td>
<td>both</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GI number*</td>
<td>P_GI</td>
<td>both</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IPI</td>
<td>P_IPI</td>
<td>both</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3D structure databases

<table>
<thead>
<tr>
<th>Name</th>
<th>Abbreviation</th>
<th>Direction</th>
<th>PDB</th>
<th>PDB_ID</th>
<th>both</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDB</td>
<td>PDB_ID</td>
<td>both</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DisProt</td>
<td>DISPROT_ID</td>
<td>both</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HSSP</td>
<td>HSSP_ID</td>
<td>both</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Mapping to/from UniProt

```python
go import urllib, urllib2
import fileinput
url = 'http://www.uniprot.org/mapping/'

# build list of RefSeq protein accessions
refseq_list = []
for acc in fileinput.input():
    acc = acc.strip()
    refseq_list.append(acc)

# tell uniprot mapper what to do
params = {
    'to': 'ACC',
    'from': 'P_REFSEQ_AC',
    'format': 'tab',
    'query': '*',
    'join': refseq_list
}

# params MUST be encoded
data = urllib.urlencode(params)
request = urllib2.Request(url, data)
contact = "wrp@virginia.edu" # set your email address
response = urllib2.urlopen(request)
page = response.read(200000)

print page # note that response includes 'From:', 'To:'
# from_to_lines = page.split("\n") # gives individual lines
```

http://www.uniprot.org/help/programmatic_access#id_mapping_examples
## Mapping to/from UniProt

```
sh uniprot_map.py ref_seq.list
```

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP_001504.2</td>
<td>O43708</td>
</tr>
<tr>
<td>Np_001504.2</td>
<td>A6NNB8</td>
</tr>
<tr>
<td>NP_714543.1</td>
<td>Q7RTV2</td>
</tr>
<tr>
<td>NP_665877.1</td>
<td>O43708</td>
</tr>
<tr>
<td>NP_001503.1</td>
<td>Q15217</td>
</tr>
<tr>
<td>NP_001503.1</td>
<td>Q6P4G1</td>
</tr>
<tr>
<td>Np_001395.1</td>
<td>P26641</td>
</tr>
<tr>
<td>NP_001395.1</td>
<td>Q53YD7</td>
</tr>
<tr>
<td>NP_671488.1</td>
<td>Q8NE79</td>
</tr>
<tr>
<td>NP_665683.1</td>
<td>P08263</td>
</tr>
<tr>
<td>NP_665683.1</td>
<td>Q5ZSC1</td>
</tr>
<tr>
<td>NP_004271.1</td>
<td>O43324</td>
</tr>
<tr>
<td>NP_000844.2</td>
<td>P30711</td>
</tr>
<tr>
<td>NP_000845.1</td>
<td>P30712</td>
</tr>
<tr>
<td>NP_000838.3</td>
<td>Q16772</td>
</tr>
<tr>
<td>NP_006294.2</td>
<td>Q13155</td>
</tr>
</tbody>
</table>

What about Pfam?

[pfam.xfam.org/help#tabview=tab11](pfam.xfam.org/help#tabview=tab11)
What about Pfam?

pfam.xfam.org/help#tabview=tab11

#!/usr/bin/python

from urllib2 import urlopen
import sys

loc = "https://pfam.xfam.org/"
prot_url = "protein?entry="
fam_url = "family?entry="
url = prot_url+xml = "&output=xml"

for acc in sys.argv[1:]:
    # print "====",loc+url+acc
    print sys.stdout.write urlopen(loc+url+acc).read()
Pfam protein annotation: (the web page)

### Summary

**GSTT1_HUMAN**

This is the summary of UniProt entry **GSTT1_HUMAN** (P30711).

<table>
<thead>
<tr>
<th>Description</th>
<th>GSTT1_HUMAN S-transferase theta-1 (EC:2.5.1.18)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source organism</td>
<td>Homo sapiens (Human) (NCBI taxonomy ID:9606)</td>
</tr>
<tr>
<td>Length</td>
<td>240 amino acids</td>
</tr>
</tbody>
</table>

Please note: when we start each new Pfam data release, we take a copy of the Uniprot sequence database. This snapshot of Uniprot forms the basis of the overview that you see here. It is important to note that, although some Uniprot entries may be removed after a Pfam release, these entries will not be removed from Pfam until the next Pfam data release.

### Pfam domains

This image shows the arrangement of the Pfam domains that we found on this sequence. Clicking on a domain will take you to the page describing that Pfam entry. The table below gives the domain boundaries for each of the domains.

<table>
<thead>
<tr>
<th>Source</th>
<th>Domain</th>
<th>Start</th>
<th>End</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pfam A</td>
<td>GST_N</td>
<td>6</td>
<td>76</td>
</tr>
<tr>
<td>Pfam A</td>
<td>GST_C</td>
<td>111</td>
<td>201</td>
</tr>
</tbody>
</table>

What about Pfam?

```python
#!/usr/bin/python

from urllib2 import urlopen
import sys

loc = "https://pfam.xfam.org/"
prot_url = "protein?entry="
fam_url="family?entry="
url = prot_url + fam_url
xml = "&output=xml"

for acc in sys.argv[1:]:
    loc+url+acc

print urlopen(loc+url+acc+xml).read()
```
Dealing with XML

- all we want to do is find:

```
<matches>
  <match accession="PF02798" id="GST_N" type="Pfam-A">  
    <location start="6" end="76" ali_start="17" ali_end="75"  
    hmm_start="15" hmm_end="74" evalue="4.2e-08" bitscore="42.20" />  
  </match>
  <match accession="PF00043" id="GST_C" type="Pfam-A">  
    <location start="111" end="201" ali_start="119" ali_end="200"  
    hmm_start="9" hmm_end="93" evalue="0.00019" bitscore="30.30" />  
  </match>
</matches>
```
Dealing with XML – xml.etree.ElementTree

```python
#!/usr/bin/env python

import pdb; pdb.set_trace()
import xml.etree.ElementTree as ET
from urllib2 import urlopen
import sys
loc = "https://pfam.xfam.org/
url = "protein?entry=
xml = "&output=xml"
for acc in sys.argv[1:] :
    pfam_xml = urlopen(loc+url+acc+xml).read()
    root = ET.fromstring(pfam_xml)
    matches = root.find('.//{{https://pfam.xfam.org/}}matches')
    domains = {}
    if (matches is None): exit()  # possibly no matches
    for child in matches.getchildren():
        domains.update(child.attrib)  # .update adds a dict
        domains.update(child2.attrib)

    print domains['id'],domains['start'],domains['end']
```

Dealing with XML – ElementTree

```python
import xml.etree.ElementTree as ET
from urllib2 import urlopen
import sys

url = "https://pfam.xfam.org/protein?entry="
xml = "&output=xml"
for acc in sys.argv[1:]:
    pfam_xml = urlopen(url+acc+xml).read()
    root = ET.fromstring(pfam_xml)
    matches = root.find('://pfam.xfam.org/}/matches')
    if (matches is None): continue  # sometimes no matches
    domains = {}
    for child in matches:
        for child2 in child:
            domains.update(child.attrib)
            domains.update(child2.attrib)
            print domains['id'],domains['start'],domains['end']
```

Pfam protein annotation:
Pfam Family XML

<xml version="1.0" encoding="UTF-8">
<!-- information on Pfam-A family PF02798 (GST_N), generated: 13:36:46 12-Feb-2015 -->
<pfam xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://pfam.xfam.org/static/documents/schemas/pfam_family.xsd" release="27.0" release_date="2013-03-06">
<entry entry_type="Pfam-A" accession="PF02798" id="GST_N">
<description>
<![CDATA[
Glutathione S-transferase, N-terminal domain
]]>
</description>
<comment>
<![CDATA[
Function: conjugation of reduced glutathione to a variety of targets. Also included in the alignment, but are not GSTs: * β-crystallins from squid. Not known to have GST activity; similarity not previously recognised. * HSP26 family of stress-related proteins, including auxin-regulated proteins in plants and stringent starvation proteins in E. coli. Not known to have GST activity; similarity not previously recognised. The glutathione molecule binds in a cleft between N and C-terminal domains - the catalytically important residues are proposed to reside in the N-terminal domain [1].
]]>
</comment>
<clan_membership clan_acc="CL0172" clan_id="Thioredoxin" />
<go_terms>
<category name="function">
<term go_id="GO:0005515">protein binding</term>
</category>
</go_terms>
<curation_details>
<status>CHANGED</status>
<seed_source>Overington</seed_source>
<previous_id>gluts;</pre>
<num_archs>61</num_archs>
<num_seqs>53</num_seqs>
<num_species>1695</num_species>
<num_structures>674</num_structures>
<percentage_identity>24</percentage_identity>
<av_length>72.80</av_length>
<av_coverage>29.98</av_coverage>
</curation_details>
<hmm_details hmmer_version="3.0" model_version="15" model_length="76">
<build_command>hmmbuild -o /dev/null MMH SEED</build_command>
<search_command>hmmssearch -E 23193494 -e 1000 --cpu 4 MMH pfamseq</search_command>
<cutoffs>
<gathering><sequence>20.8</sequence><domain>20.8</domain></gathering>
<trusted><sequence>20.8</sequence><domain>20.8</domain></trusted>
<noise><sequence>20.8</sequence><domain>20.8</domain></noise>
</cutoffs>
</hmm_details>
</entry>
</pfam>

Pfam Family XML (cont.)

<clan_membership clan_acc="CL0172" clan_id="Thioredoxin" />
<go_terms>
<category name="function">
<term go_id="GO:0005515">protein binding</term>
</category>
</go_terms>
<curation_details>
<status>CHANGED</status>
<seed_source>Overington</seed_source>
<previous_id>gluts;</pre>
<num_archs>61</num_archs>
<num_seqs>53</num_seqs>
<num_species>1695</num_species>
<num_structures>674</num_structures>
<percentage_identity>24</percentage_identity>
<av_length>72.80</av_length>
<av_coverage>29.98</av_coverage>
</curation_details>
<hmm_details hmmer_version="3.0" model_version="15" model_length="76">
<build_command>hmmbuild -o /dev/null MMH SEED</build_command>
<search_command>hmmssearch -E 23193494 -e 1000 --cpu 4 MMH pfamseq</search_command>
<cutoffs>
<gathering><sequence>20.8</sequence><domain>20.8</domain></gathering>
<trusted><sequence>20.8</sequence><domain>20.8</domain></trusted>
<noise><sequence>20.8</sequence><domain>20.8</domain></noise>
</cutoffs>
</hmm_details>
</entry>
</pfam>
import xml.etree.ElementTree as ET
from urllib2 import urlopen
import sys

loc = "https://pfam.xfam.org/
prot_url = "protein/"
fam_url = "family/"
xm = "?output=xml"
url = fam_url

for acc in sys.argv[1:]: # acc is a pfamA_acc, PF01234, not a uniprot_acc
    pfam_xml = urlopen(loc+url+acc+xml).read()
    root = ET.fromstring(pfam_xml)
    entry = root.find(''.https://pfam.xfam.org/)entry')
    details = root.find(''.https://pfam.xfam.org/)hmm_details')

    print entry.attrib['accession'], details.attrib['model_length']

One last XML:
NCBI esummary for protein length


<?xml version="1.0" encoding="UTF-8"?>
<eSummaryResult>
    <DocSum>
        <Id>121735</Id>
        <Item Name="Caption" Type="String">P09488</Item>
        <Item Name="Title" Type="String">RecName: Full=Glutathione S-transferase Mu 1; AltName: Full=GST HB subunit 4; AltName: Full=GST class -mu 1; AltName: Full=GSTM1-1a; AltName: Full=GSTM1a-1a; AltName: Full=GSTM1b-1b; AltName: Full=GTH4</Item>
        <Item Name="Extra" Type="String">gi|121735|sp|P09488.3|GSTM1_HUMAN[121735]</Item>
        <Item Name="GI" Type="Integer">121735</Item>
        <Item Name="CreateDate" Type="String">1989/07/01</Item>
        <Item Name="UpdateDate" Type="String">2016/01/20</Item>
        <Item Name="Flags" Type="Integer">0</Item>
        <Item Name="TaxId" Type="Integer">9606</Item>
        <Item Name="Length" Type="Integer">218</Item>
        <Item Name="Status" Type="String">live</Item>
        <Item Name="AccessionVersion" Type="String">P09488.3</Item>
    </DocSum>
</eSummaryResult>
One last XML: NCBI esummary for protein length

```python
#!/bin/env python
import xml.etree.ElementTree as ET
from urllib2 import urlopen
import sys

def get_summary(acc):
    ncbi_xml = urlopen(loc + prot_db_id + acc).read()
    tree = ET.fromstring(ncbi_xml)
    items = {}
    for item in tree.iter(tag='Item'):
        # no {http://...} because no xmlns:
        items.update({item.attrib['Name']: item.text})
    return items

for acc in sys.argv[1:]:
    info = get_summary(acc)
    # for key in info.keys(): print key, info[key]
    print 'Length (+' + acc + '): ', info['Length']
```

Another last XML: NCBI esearch.fcgi accessions

```xml
<?xml version="1.0" encoding="UTF-8"?>
<eSearchResult>
<Count>40</Count><RetMax>40</RetMax><RetStart>0</RetStart><IdList><Id>NP_001278092.1</Id>
<Id>NP_001129490.1</Id><Id>NP_758859.1</Id><Id>NP_071900.2</Id><Id>NP_000843.1</Id>
... stuff deleted
<Id>NP_000842.2</Id><Id>NP_000841.1</Id><Id>NP_671489.1</Id><Id>NP_714543.1</Id><Id>NP_665531.1</Id><Id>NP_005733.1</Id>
<Id>NP_001278092.1</Id></IdList>
<TranslationSet><Translation><From>human[organism]</From><To>"Homo sapiens"[Organism]</To></Translation></TranslationSet><TranslationStack><TermSet><Term>gstm[All Fields]</Term><Field>All Fields</Field><Count>43</Count></TermSet><TermSet><Term>gstm1[All Fields]</Term><Field>All Fields</Field><Count>2</Count></TermSet><TermSet><Term>gstm1/t1[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm1a[All Fields]</Term><Field>All Fields</Field><Count>2</Count></TermSet><TermSet><Term>gstm1b[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm2[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm3[All Fields]</Term><Field>All Fields</Field><Count>4</Count></TermSet><TermSet><Term>gstm3a[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm3b[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm3c[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm3d[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm4[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm4'[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm5[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm6[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm7[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm8[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm9[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm10[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm11[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstmmu[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstmnu[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm12[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm13[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm14[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm15[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm16[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm17[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet><TermSet><Term>gstm18[All Fields]</Term><Field>All Fields</Field><Count>1</Count></TermSet></TermSet><QueryTranslation>(gstm[All Fields] OR gstm1[All Fields] OR gstm1/t1[All Fields] OR gstm1a[All Fields] OR gstm1b[All Fields] OR gstm2[All Fields] OR gstm3[All Fields] OR gstm3a[All Fields] OR gstm3b[All Fields] OR gstm3c[All Fields] OR gstm3d[All Fields] OR gstm4[All Fields] OR gstm4'[All Fields] OR gstm5[All Fields] OR gstm6[All Fields] OR gstm7[All Fields] OR gstm8[All Fields] OR gstm9[All Fields] OR gstm10[All Fields] OR gstm11[All Fields] OR gstmmu[All Fields] OR gstmnu[All Fields]) AND "Homo sapiens"[Organism] AND srcdb_refseq[prop]"/esearch.fcgi"</QueryTranslation></eSearchResult>
```
Another last XML: NCBI esearch.fcgi accessions

```
#!/bin/env python
import xml.etree.ElementTree as ET
from urllib2 import urlopen
import sys
search_string='GSTM*+AND+human[organism]+AND+srcdb_refseq[prop]'

def get_accs(search_str):
    prot_db_id='db=protein&idtype=acc&retmax=1000&term='
    ncbi_xml = urlopen(loc+prot_db_id+search_str).read()
    tree = ET.fromstring(ncbi_xml)
    acc_list = []
    for item in tree.iter(tag='Id'):
        acc_list.append(item.text)
    return acc_list

acc_list = get_accs(search_string)
for acc in acc_list:
    print acc
```

Bioinformatics Web Resources

- NCBI – eutilities: esearch/efetch/blast search
- Recognizing web addresses (URLs)
- EBI – web services
  [www.ebi.ac.uk/Tools/webservices/](www.ebi.ac.uk/Tools/webservices/)
- Uniprot ID mapper
  [www.uniprot.org/faq/28#id_mapping_examples](www.uniprot.org/faq/28#id_mapping_examples)
- Pfam – using XML data
  [pfam.xfam.org/help#tabview=tab10](pfam.xfam.org/help#tabview=tab10)
  [xml.etree.ElementTree](xml.etree.ElementTree)
Homework, due Monday, Feb 20 (biol4230/hwk5)

Do the exercises and write the programs to answer the questions at:

fasta.bioch.virginia.edu/biol4230/labs/accessions_hwk5.html