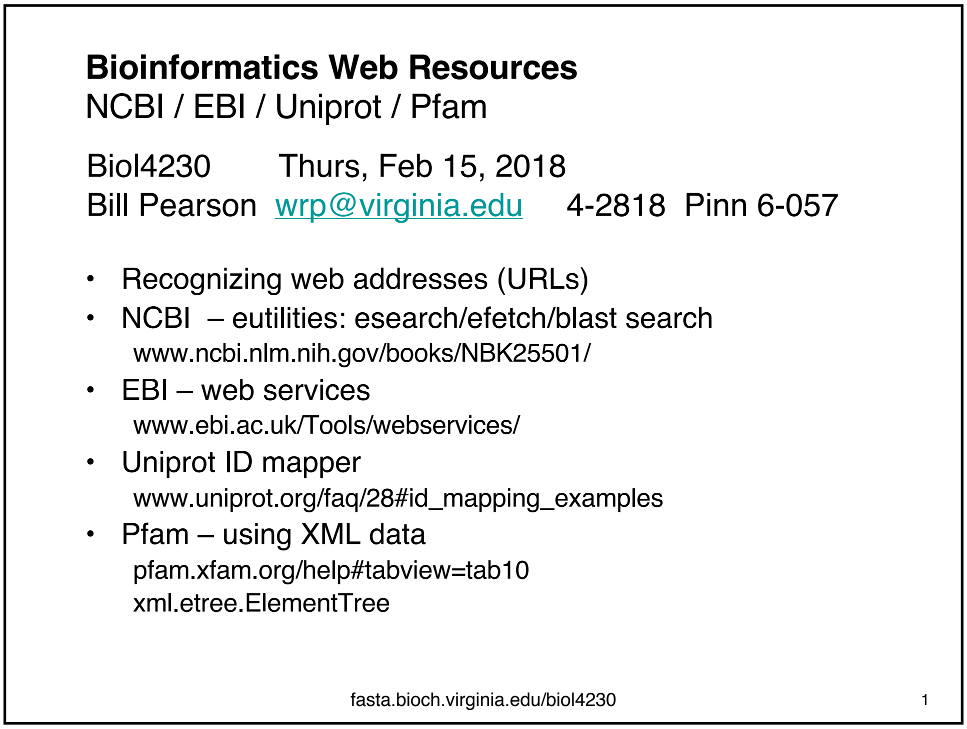
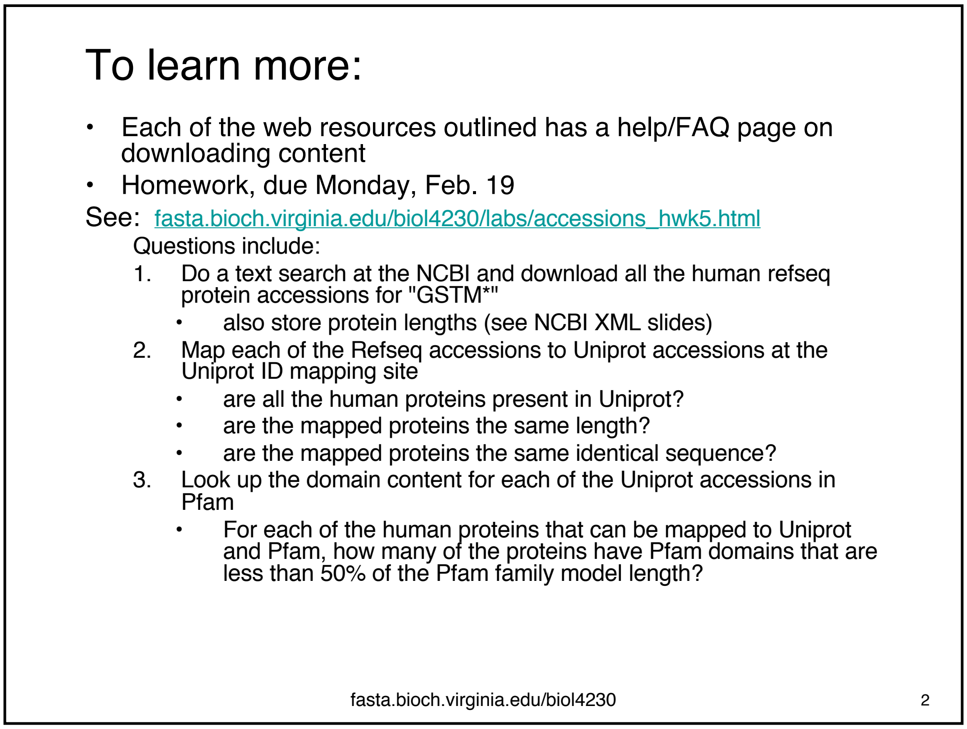


Bioinformatics Web Resources
NCBI / EBI / Uniprot / Pfam

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

- Recognizing web addresses (URLs)
- NCBI – eutilities: 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



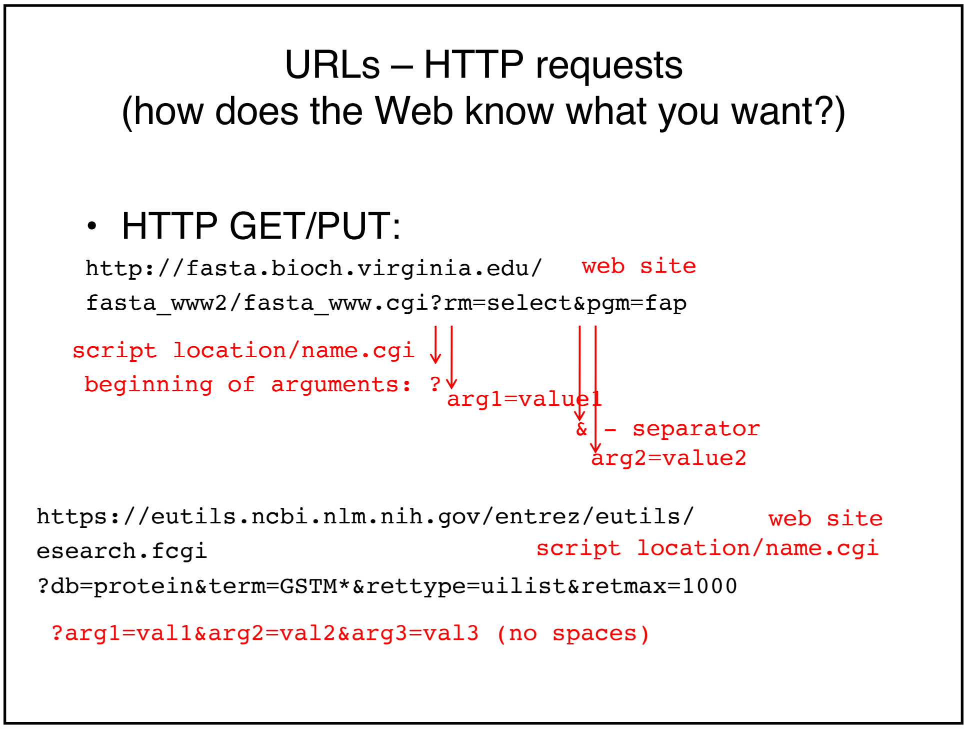
To learn more:

- 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?

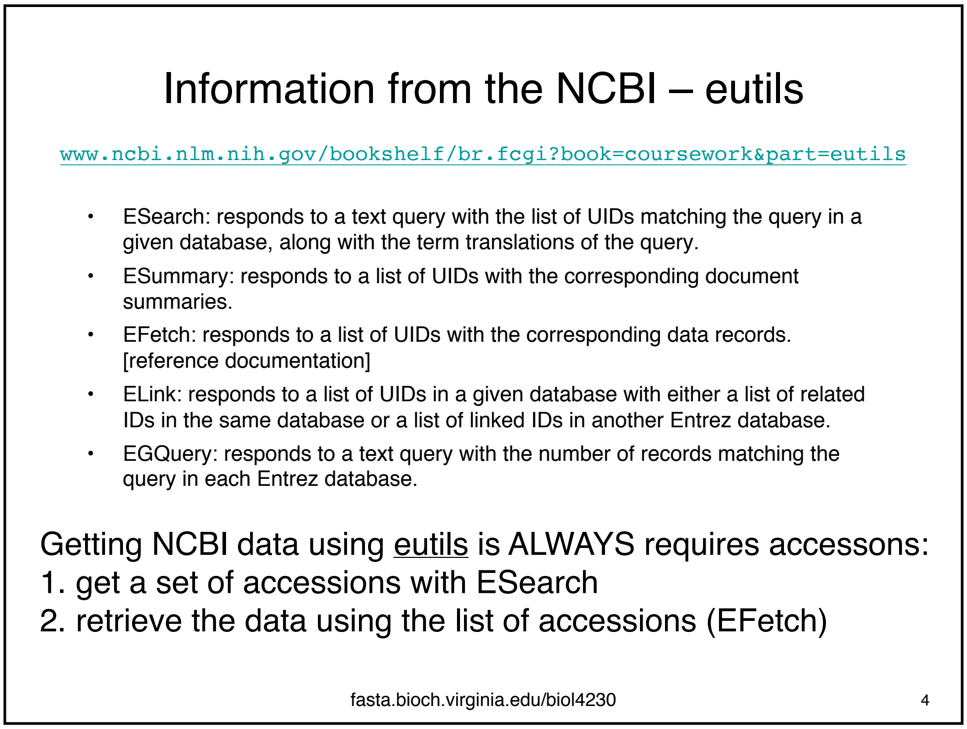


URLs – HTTP requests (how does the Web know what you want?)

- 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
```

```
https://eutils.ncbi.nlm.nih.gov/entrez/eutils/      web site
esearch.fcgi                        script location/name.cgi
?db=protein&term=GSTM*&rettype=uilist&retmax=1000
 ?arg1=val1&arg2=val2&arg3=val3 (no spaces)
```



Information from the NCBI – eutils

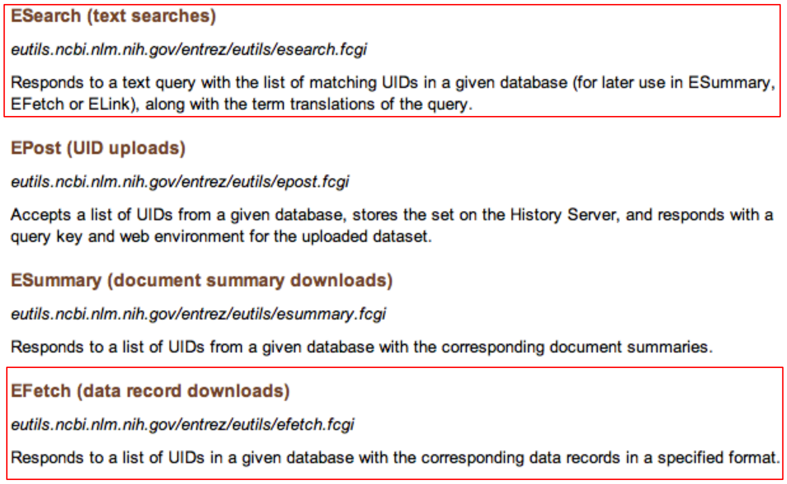
www.ncbi.nlm.nih.gov/bookshelf/br.fcgi?book=coursework&part=eutils

- 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

```
curl
'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=protein&term=GSTM*+AND+human\[organism\
]+AND+srcdb_refseq\[prop\]&idtype=acc&retmax=10000'
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE eSearchResult PUBLIC "-//NLM//DTD esearch 20060628//EN"
"http://eutils.ncbi.nlm.nih.gov/eutils/dtd/20060628/esearch.dtd">
<eSearchResult>
<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_071900.2</Id>
           <Id>NP_000843.1</Id>
           <Id>NP_000111.1</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_666533.1</Id>
           <Id>NP_000552.2</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>
... stuff deleted
</TranslationStack><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 gstmu2[All Fields] OR gstmu3[All Fields]) AND "Homo sapiens"[Organism] AND
srcdb_refseq[prop]</QueryTranslation>
</eSearchResult>
```

urllib2/urlopen at the NCBI

```
#!/bin/env python

from urllib2 import urlopen
import re
#import pdb; pdb.set_trace()
# setup URL
s_url = "https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?";
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 for m in [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_url = "https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?"
seq_args = "db=protein&id="+ ",".join(acc_ids)  + "&rettype=fasta"

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

print seq_html
```

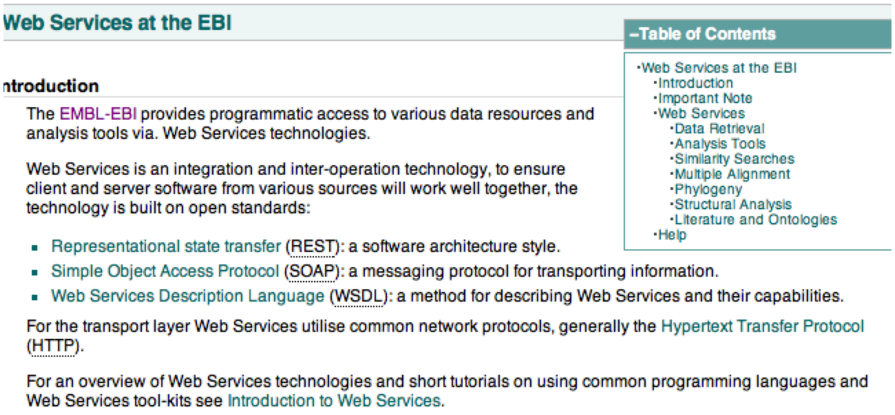
NCBI &retmode, &rettype

https://www.ncbi.nlm.nih.gov/books/NBK25499/table/chapter4.T._valid_values_of__retmode_and/

Record Type	&rettype	&retmode
	db = gene	
text ASN.1	null	asn.1, default
XML	null	xml
Gene table	gene_table	text
	db = nuccore, nucest, nucgss, protein or popset	
text ASN.1	null	text, default
binary ASN.1	null	asn.1
Full record in XML	native	xml
FASTA	fasta	text
Accession	acc	text
	db = pubmed	
text ASN.1	null	asn.1, default
XML	null	xml
MEDLINE	medline	text
PMID list	uilist	text
Abstract	abstract	text
	db = taxonomy	
XML	null	xml, default
TaxID list	uilist	text or xml

How to find data: EBI/EMBL

www.ebi.ac.uk/Tools/webservices/



Web Services at the EBI

ntroduction

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:

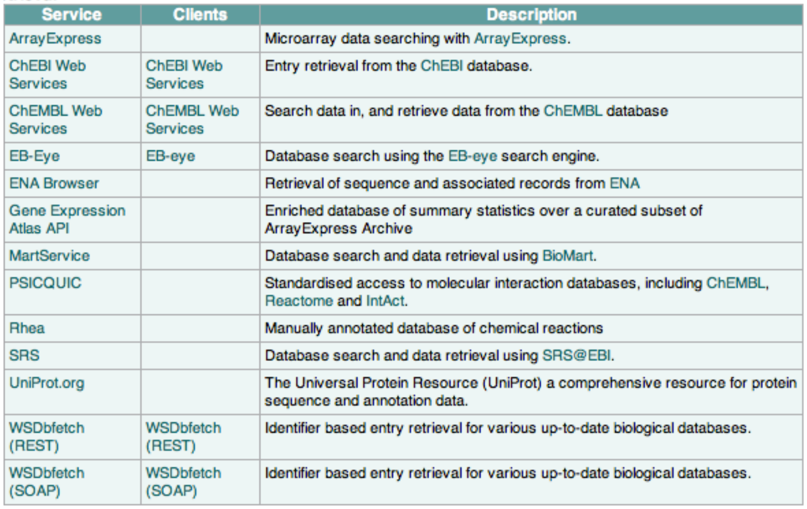
- Representational state transfer (REST): a software architecture style.
- Simple Object Access Protocol (SOAP): a messaging protocol for transporting information.
- Web Services Description Language (WSDL): a method for describing Web Services and their capabilities.

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.

How to find data: EBI/EMBL

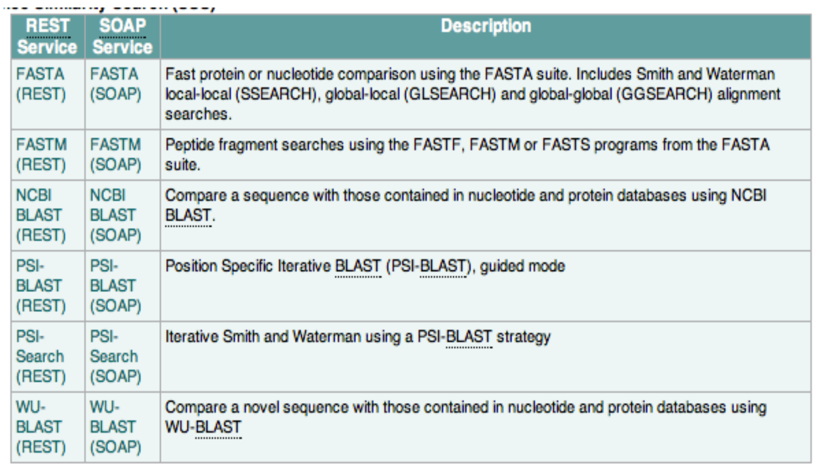
www.ebi.ac.uk/Tools/webservices/



Service	Clients	Description
ArrayExpress		Microarray data searching with ArrayExpress.
ChEBI Web Services	ChEBI Web Services	Entry retrieval from the ChEBI database.
ChEMBL Web Services	ChEMBL Web Services	Search data in, and retrieve data from the ChEMBL database
EB-Eye	EB-eye	Database search using the EB-eye search engine.
ENA Browser		Retrieval of sequence and associated records from ENA
Gene Expression Atlas API		Enriched database of summary statistics over a curated subset of ArrayExpress Archive
MartService		Database search and data retrieval using BioMart.
PSICQUIC		Standardised access to molecular interaction databases, including ChEMBL, Reactome and IntAct.
Rhea		Manually annotated database of chemical reactions
SRS		Database search and data retrieval using SRS@EBI.
UniProt.org		The Universal Protein Resource (UniProt) a comprehensive resource for protein sequence and annotation data.
WSDbfetch (REST)	WSDbfetch (REST)	Identifier based entry retrieval for various up-to-date biological databases.
WSDbfetch (SOAP)	WSDbfetch (SOAP)	Identifier based entry retrieval for various up-to-date biological databases.

How to find data: EBI/EMBL

www.ebi.ac.uk/Tools/webservices/



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

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

- Pfam uses Uniprot Id's and Uniprot accession numbers:

```
    |acc   |id
>sp|O43708|MAAI_HUMAN Maleylacetoacetate isomerase GN=GSTZ1 PE=1
SV=3MQAGKPILYSYFRSSCSWRVRIALALKGIDYKTVPINLIKDRGQQFSKDFQALNPMKQVPTLKIDGITIHQSLA
IIEYLEEMRPTPRLLPQDPKKRASVRMISDLIAGGIQPLQNLSVLKQVGEEMQLTWAQNAITCGFNALEQILQSTAGI
YCVGDEVTMADLCLVPQVANAERFKVDLTPYPTISSINKRLLVLEAFQVSHPCRQPDTPTELRA
```

- 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

Name	Abbreviation	Direction	Name	Abbreviation	Direction
UniProt			**Other sequence databases**		
UniProtKB AC/ID	ACC+ID	from	DNA	EMBL_ID	both
UniProtKB AC	ACC	to	DNA CDS	EMBL	both
UniProtKB ID	ID	to	PIR	PIR	both
UniParc	UPARC	both	UniGene	UNIGENE_ID	both
UniRef50	NF50	both	Entrez Gene	P_ENTREZGENEID	both
UniRef90	NF90	both	GI number*	P_GI	both
UniRef100	NF100	both	IPI	P_IPI	both
			RefSeq	P_REFSEQ_AC	both
			3D structure databases		
			PDB	PDB_ID	both
			DisProt	DISPROT_ID	both
			HSSP	HSSP_ID	both

Mapping to/ from UniProt

```
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
request.add_header('User-Agent','Python %s' % contact)
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
```

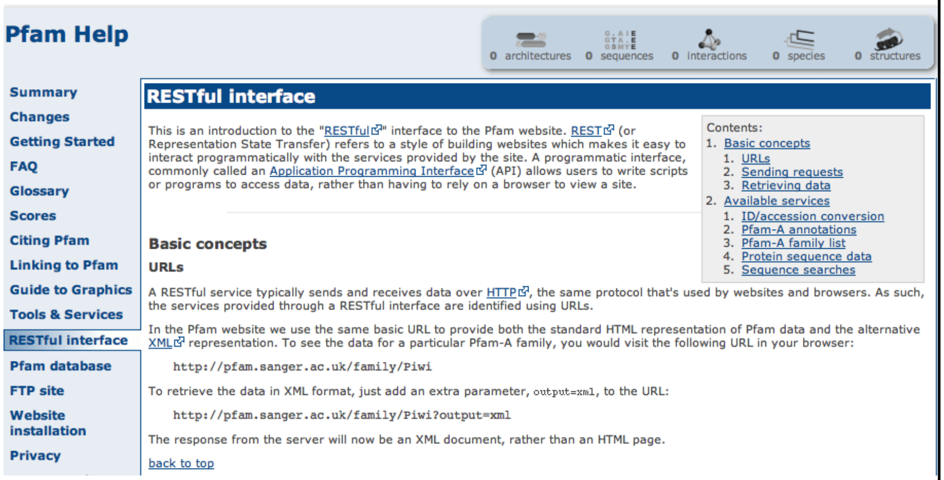
http://www.uniprot.org/help/programmatic_access#id_mapping_examples

Mapping to/from UniProt

```
sh% uniprot_map.py ref_seq.list
                From    To
         NP_001504.2    O43708
         Np_001504.2    A6NNB8
         NP_714543.1    Q7RTV2
         NP_665877.1    O43708
         NP_001503.1    O15217
         NP_001503.1    Q6P4G1
         NP_001395.1    P26641
         NP_001395.1    Q53YD7
         NP_671488.1    Q8NE79
         NP_665683.1    P08263
         NP_665683.1    Q5SZC1
         NP_004271.1    O43324
         NP_000844.2    P30711
         NP_000845.1    P30712
         NP_000838.3    Q16772
         NP_006294.2    Q13155
```

What about Pfam?

pfam.xfam.org/help#tabview=tab11



RESTful interface

This is an introduction to the "RESTful" interface to the Pfam website. REST (or Representation State Transfer) refers to a style of building websites which makes it easy to interact programmatically with the services provided by the site. A programmatic interface, commonly called an Application Programming Interface (API) allows users to write scripts or programs to access data, rather than having to rely on a browser to view a site.

Contents:

1. Basic concepts
   1. URLs
   2. Sending requests
   3. Retrieving data
2. Available services
   1. ID/accession conversion
   2. Pfam-A annotations
   3. Pfam-A family list
   4. Protein sequence data
   5. Sequence searches

Basic concepts

URLs

A RESTful service typically sends and receives data over HTTP, the same protocol that's used by websites and browsers. As such, the services provided through a RESTful interface are identified using URLs.

In the Pfam website we use the same basic URL to provide both the standard HTML representation of Pfam data and the alternative XML representation. To see the data for a particular Pfam-A family, you would visit the following URL in your browser:

```
http://pfam.sanger.ac.uk/family/Piwi
```

To retrieve the data in XML format, just add an extra parameter, `output=xml`, to the URL:

```
http://pfam.sanger.ac.uk/family/Piwi?output=xml
```

The response from the server will now be an XML document, rather than an HTML page.

back to top

What about Pfam?

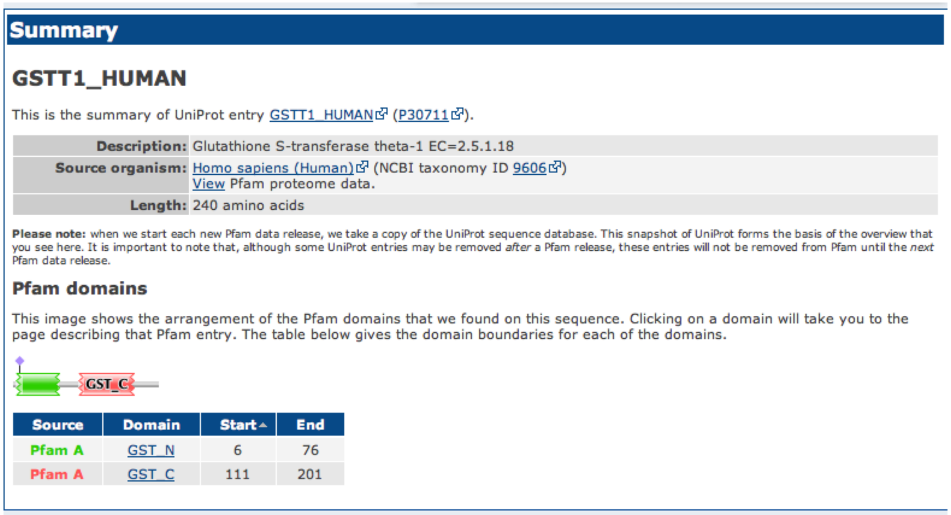
pfam.xfam.org/help#tabview=tab11

```
#!/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 urlopen(loc+url+acc).read()
```

Pfam protein annotation: (the web page)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
  <head>
   <title>Pfam:
Protein: GSTT1_HUMAN (P30711)
</title>
    <meta name="verify-v1" content="GjV+z5lf7mSCShhAOJZh1UW8J+iiCgWmbxIFg2GkG0Q=" />
<meta name="verify-v1" content="FA9AR+bh3BmS05vcSp0mbiAB80DgELEAkFvu4q9ViC8=" />

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="Description" content="Pfam is a large collection of protein families, represented by
     multiple sequence alignments and hidden Markov models (HMMs)" />


<!-- ====================================================================== -->
<!-- make the site RSS feed discoverable -->

<link href="http://xfam.wordpress.com/tag/pfam/feed/"
      rel="alternate"
      type="application/rss+xml"
      title="Pfam News" />

<!-- ====================================================================== -->
<!-- third-party javascript libraries

     we are now loading third-party libraries from remote sites. We get
     prototype and scriptaculous from googleapis and the YUI components
     for tree viewing directly from yahoo
-->
```

Pfam protein annotation: (the web page)



Summary

GSTT1_HUMAN

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

Description:	Glutathione S-transferase theta-1 EC=2.5.1.18
Source organism:	Homo sapiens (Human) (NCBI taxonomy ID 9606) View Pfam proteome data.
Length:	240 amino acids

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.

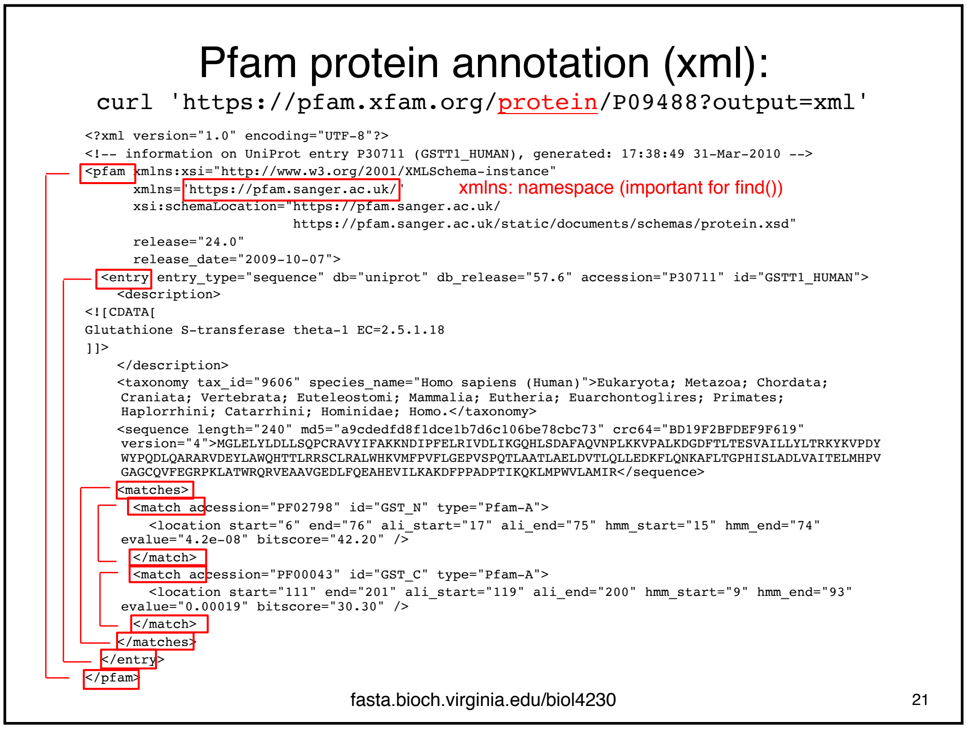
Pfam domains

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.

Source	Domain	Start▲	End
Pfam A	GST_N	6	76
Pfam A	GST_C	111	201

What about Pfam?

```
#!/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 urlopen(loc+url+acc+xml).read()
```

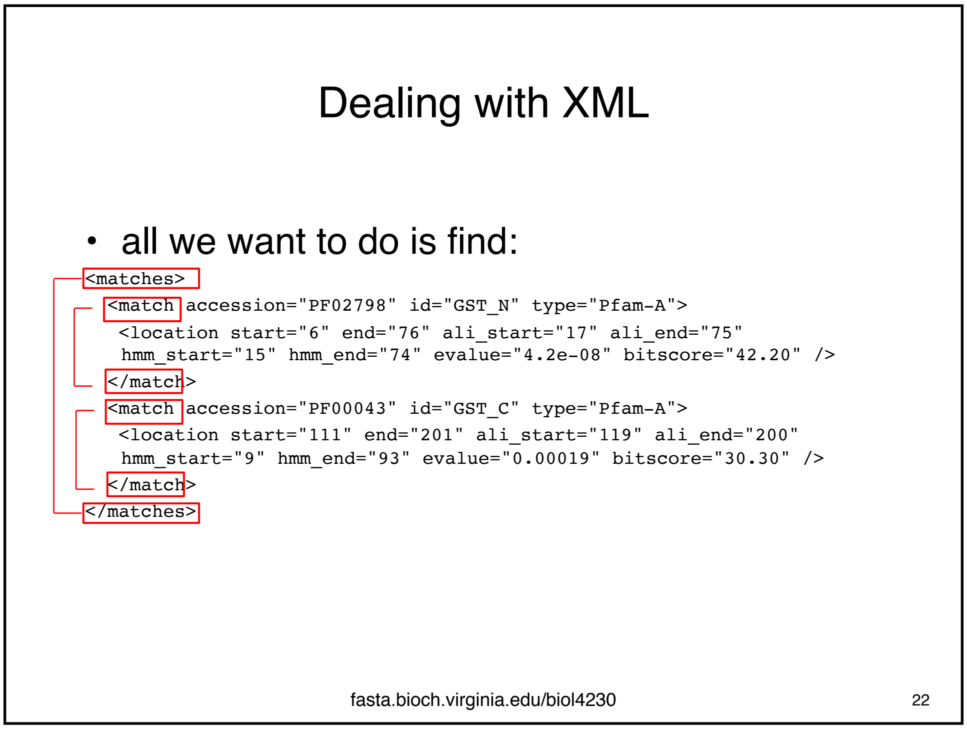


Pfam protein annotation (xml):

```
curl 'https://pfam.xfam.org/protein/P09488?output=xml'
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- information on UniProt entry P30711 (GSTT1_HUMAN), generated: 17:38:49 31-Mar-2010 -->
<pfam xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns='https://pfam.sanger.ac.uk/'
      xsi:schemaLocation="https://pfam.sanger.ac.uk/
                         https://pfam.sanger.ac.uk/static/documents/schemas/protein.xsd"
      release="24.0"
      release_date="2009-10-07">
  <entry entry_type="sequence" db="uniprot" db_release="57.6" accession="P30711" id="GSTT1_HUMAN">
    <description>
<![CDATA[
Glutathione S-transferase theta-1 EC=2.5.1.18
]]>
    </description>
    <taxonomy tax_id="9606" species_name="Homo sapiens (Human)">Eukaryota; Metazoa; Chordata;
     Craniata; Vertebrata; Euteleostomi; Mammalia; Eutheria; Euarchontoglires; Primates;
     Haplorrhini; Catarrhini; Hominidae; Homo.</taxonomy>
    <sequence length="240" md5="a9cdedfd8f1dce1b7d6c106be78cbc73" crc64="BD19F2BFDEF9F619"
     version="4">MGLELYLDLLSQPCRAVYIFAKKNDIPFELRIVDLIKGQHLSDAFAQVNPLKKVPALKDGDFTLTESVAILLYLTRKYKVPDY
     WYPQDLQARARVDEYLAWQHTTLRRSCLRALWHKVMFPVFLGEPVSPQTLAATLAELDVTLQLLEDKFLQNKAFLTGPHISLADLVAITELMHPV
     GAGCQVFEGRPKLATWRQRVEAAVGEDLFQEAHEVILKAKDFPPADPTIKQKLMPWVLAMIR</sequence>
    <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>
  </entry>
</pfam>
```

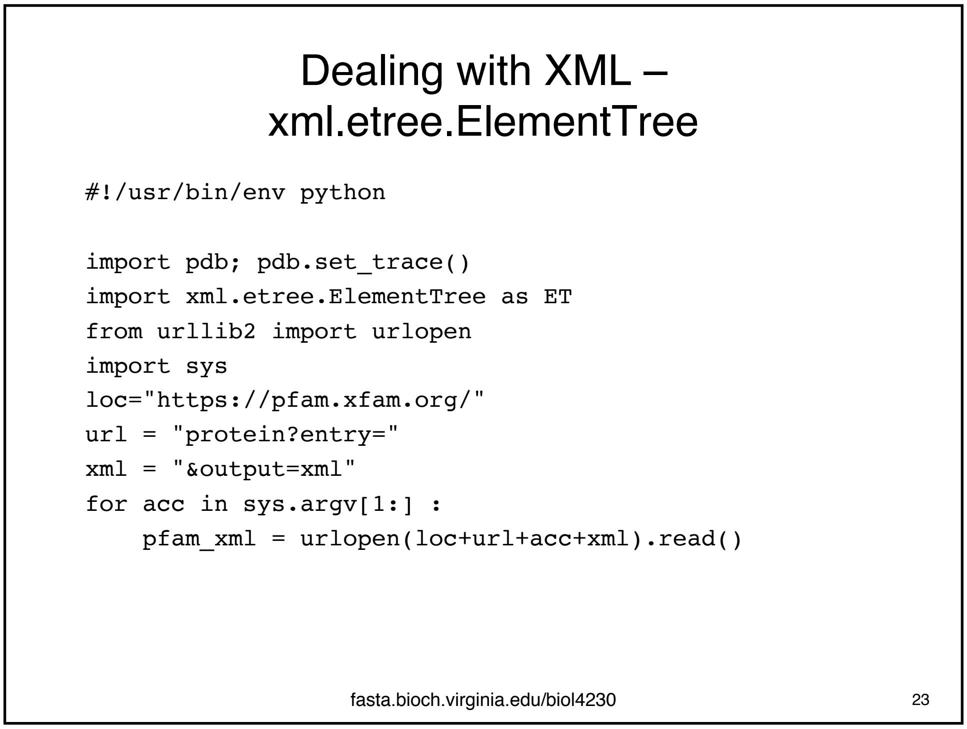
xmlns: namespace (important for find())



Dealing with XML

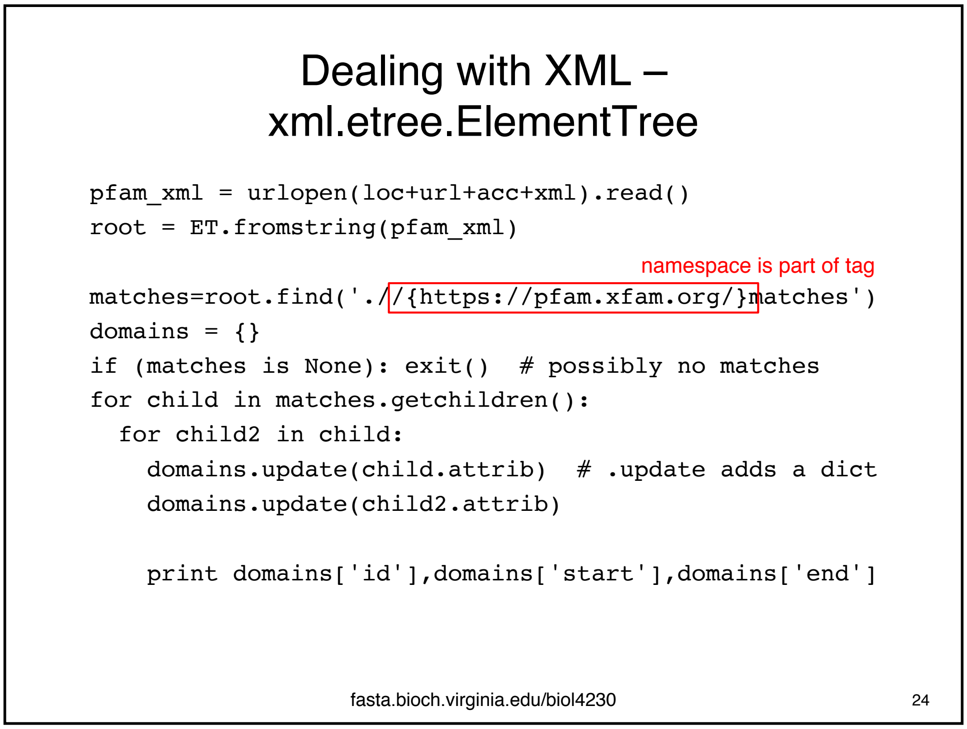
- all we want to do is find:

```
<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

```
#!/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()
```



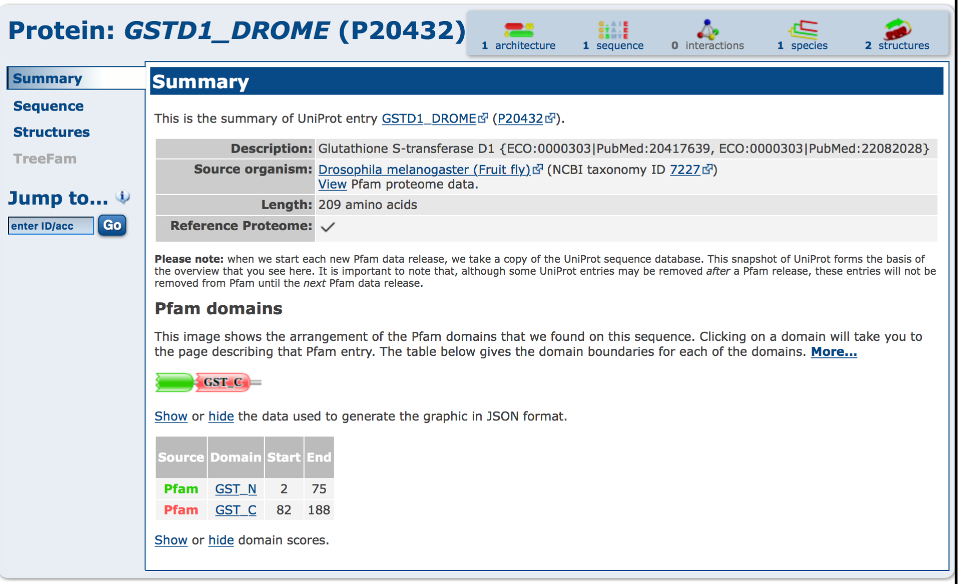
Dealing with XML – xml.etree.ElementTree

```
pfam_xml = urlopen(loc+url+acc+xml).read()
root = ET.fromstring(pfam_xml)
```

namespace is part of tag

```
matches=root.find('.//{https://pfam.xfam.org/}matches')
domains = {}
if (matches is None): exit()  # possibly no matches
for child in matches.getchildren():
  for child2 in child:
    domains.update(child.attrib)  # .update adds a dict
    domains.update(child2.attrib)

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

Pfam protein annotation:



Protein: *GSTD1_DROME* (P20432)

Summary

This is the summary of UniProt entry GSTD1_DROME (P20432).

Description:	Glutathione S-transferase D1 {ECO:0000303|PubMed:20417639, ECO:0000303|PubMed:22082028}
Source organism:	Drosophila melanogaster (Fruit fly) (NCBI taxonomy ID 7227) View Pfam proteome data.
Length:	209 amino acids
Reference Proteome:	✓

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. **More...**

Show or hide the data used to generate the graphic in JSON format.

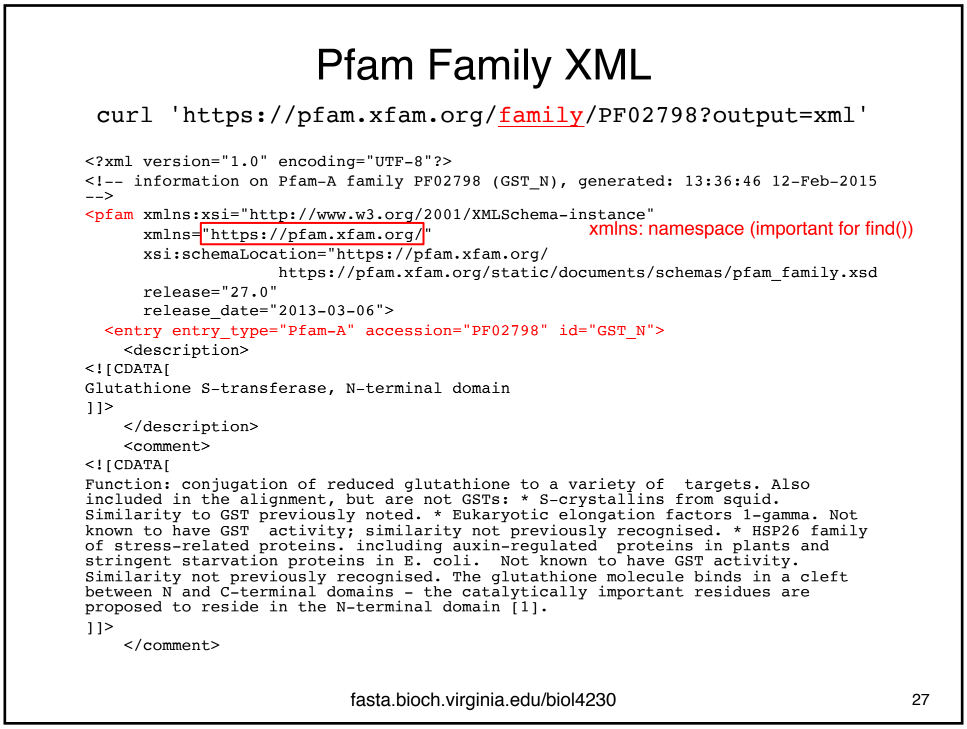
Source	Domain	Start	End
Pfam	GST_N	2	75
Pfam	GST_C	82	188

Show or hide domain scores.

Dealing with XML – ElementTree

```
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('.//{https://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_xml.py gstd1_human
GST_N   6       76
GST_C   111     201
```

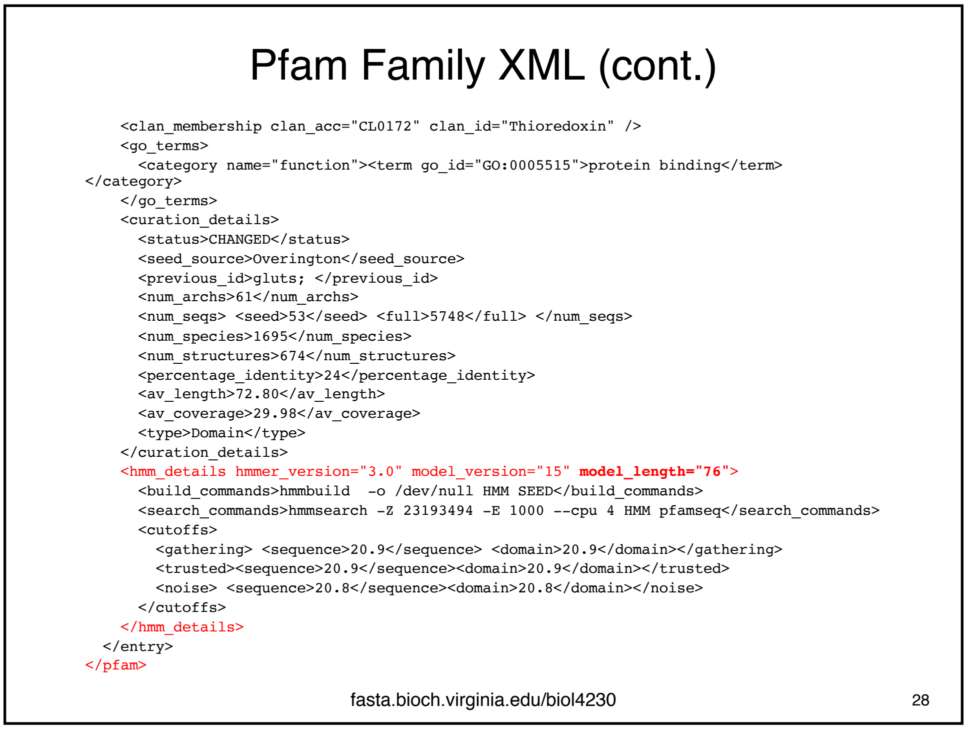


Pfam Family XML

```
curl 'https://pfam.xfam.org/family/PF02798?output=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"
      xmlns="https://pfam.xfam.org/"
      xsi:schemaLocation="https://pfam.xfam.org/
                    https://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: * S-crystallins from squid.
Similarity to GST previously noted. * Eukaryotic elongation factors 1-gamma. 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>
```

xmlns: namespace (important for find())

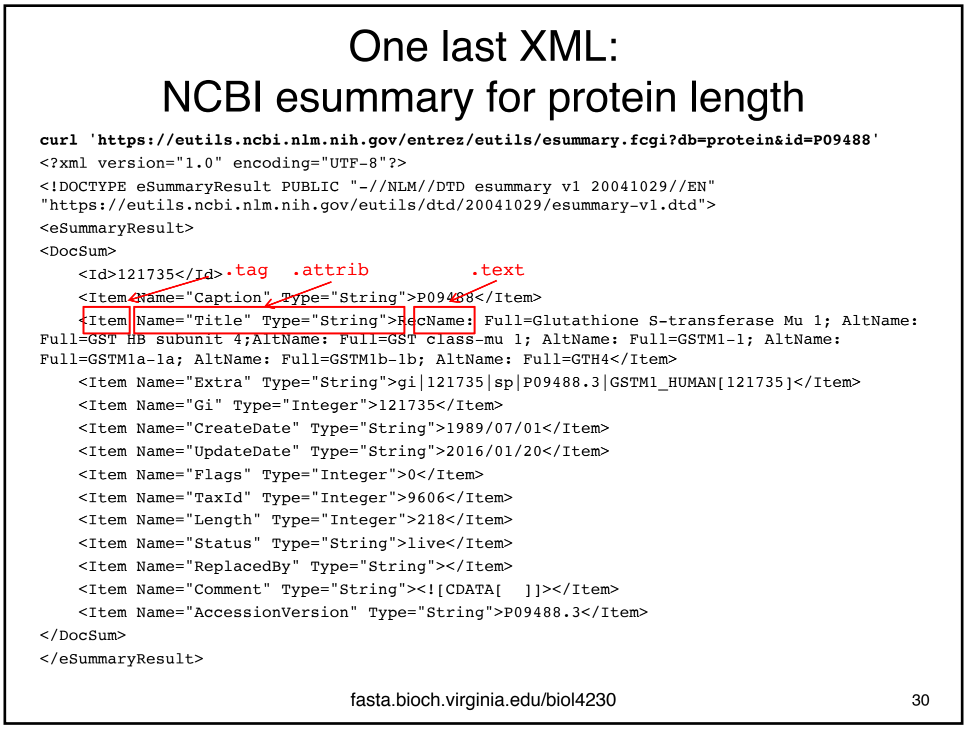


Pfam Family XML (cont.)

```
    <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; </previous_id>
      <num_archs>61</num_archs>
      <num_seqs> <seed>53</seed> <full>5748</full> </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>
      <type>Domain</type>
    </curation_details>
    <hmm_details hmmer_version="3.0" model_version="15" model_length="76">
      <build_commands>hmmbuild  -o /dev/null HMM SEED</build_commands>
      <search_commands>hmmsearch -Z 23193494 -E 1000 --cpu 4 HMM pfamseq</search_commands>
      <cutoffs>
        <gathering> <sequence>20.9</sequence> <domain>20.9</domain></gathering>
        <trusted><sequence>20.9</sequence><domain>20.9</domain></trusted>
        <noise> <sequence>20.8</sequence><domain>20.8</domain></noise>
      </cutoffs>
    </hmm_details>
  </entry>
</pfam>
```

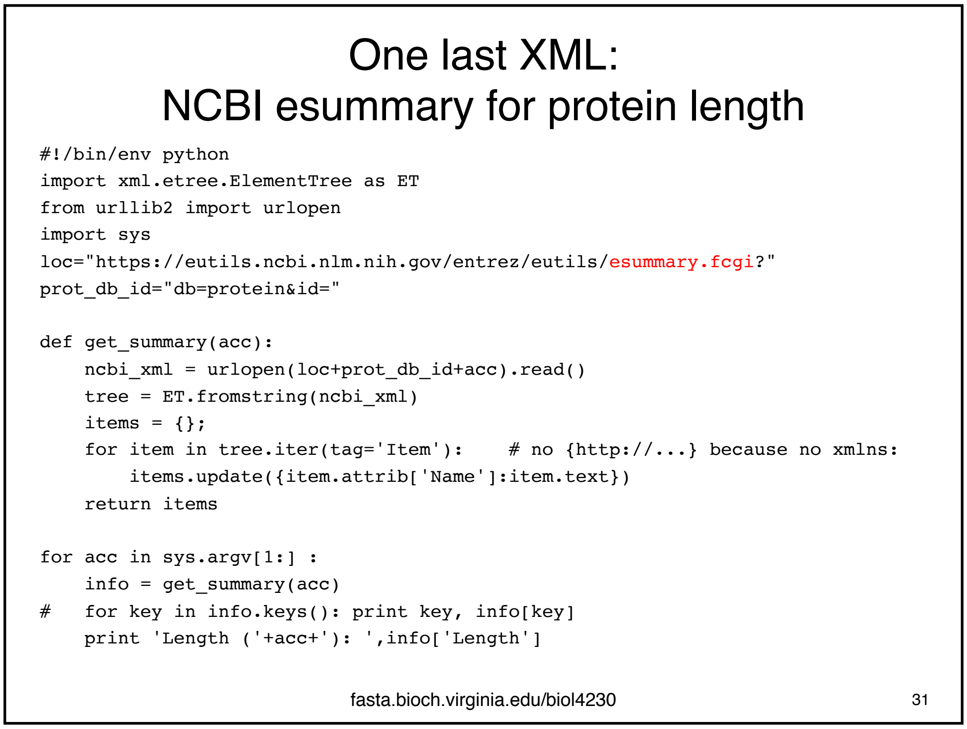
Pfam family XML (xml.etree.ElementTree)

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

loc="https://pfam.xfam.org/"
prot_url="protein/"
fam_url="family/"
xml = "?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']
```



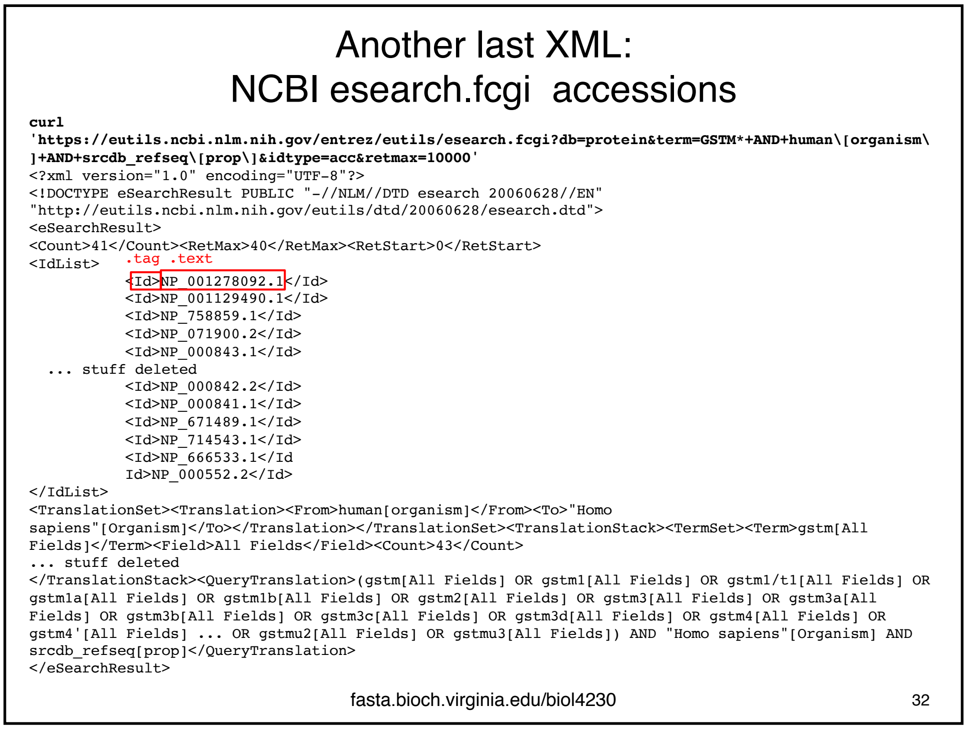
One last XML: NCBI esummary for protein length

```
curl 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi?db=protein&id=P09488'
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE eSummaryResult PUBLIC "-//NLM//DTD esummary v1 20041029//EN"
"https://eutils.ncbi.nlm.nih.gov/eutils/dtd/20041029/esummary-v1.dtd">
<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=GSTM1-1; 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="ReplacedBy" Type="String"></Item>
    <Item Name="Comment" Type="String"><![CDATA[  ]]></Item>
    <Item Name="AccessionVersion" Type="String">P09488.3</Item>
</DocSum>
</eSummaryResult>
```



One last XML: NCBI esummary for protein length

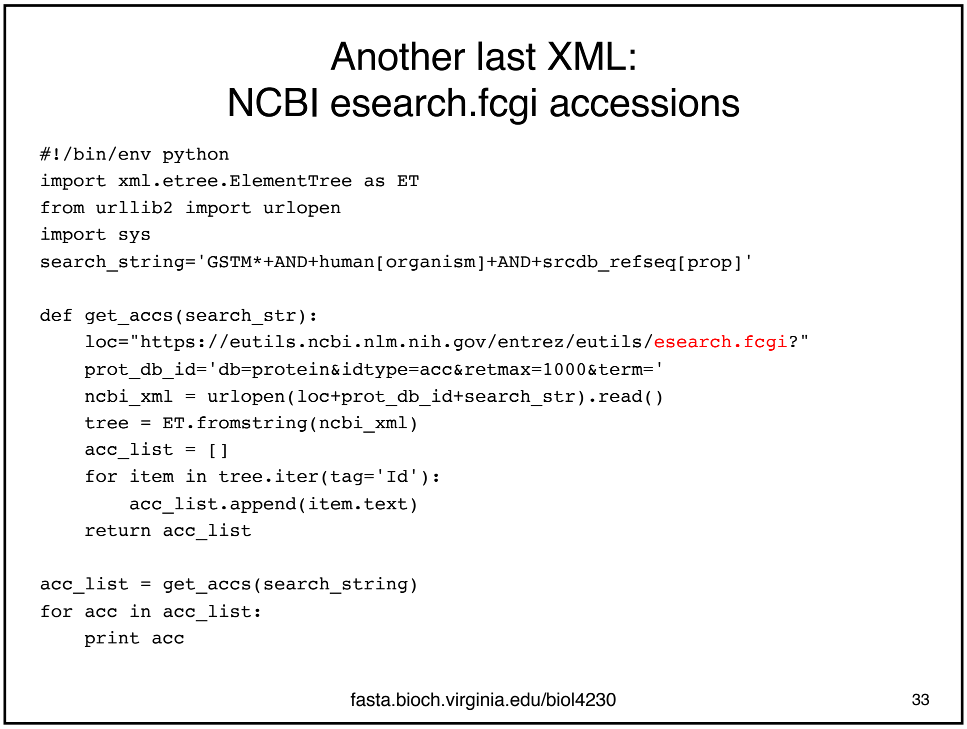
```
#!/bin/env python
import xml.etree.ElementTree as ET
from urllib2 import urlopen
import sys
loc="https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi?"
prot_db_id="db=protein&id="

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

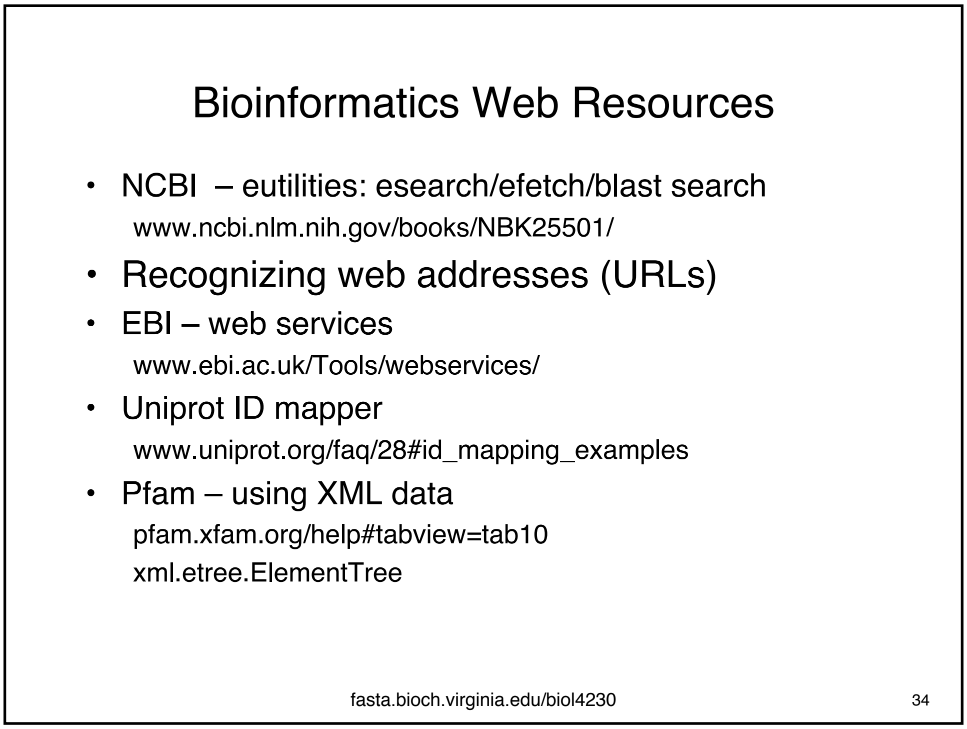
curl 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=protein&term=GSTM*+AND+human\[organism\]+AND+srcdb_refseq\[prop\]&idtype=acc&retmax=10000'

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE eSearchResult PUBLIC "-//NLM//DTD esearch 20060628//EN"
"http://eutils.ncbi.nlm.nih.gov/eutils/dtd/20060628/esearch.dtd">
<eSearchResult>
<Count>41</Count><RetMax>40</RetMax><RetStart>0</RetStart>
<IdList>    .tag .text
            <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_666533.1</Id
            Id>NP_000552.2</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>
... stuff deleted
</TranslationStack><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 gstmu2[All Fields] OR gstmu3[All Fields]) AND "Homo sapiens"[Organism] AND
srcdb_refseq[prop]</QueryTranslation>
</eSearchResult>
```



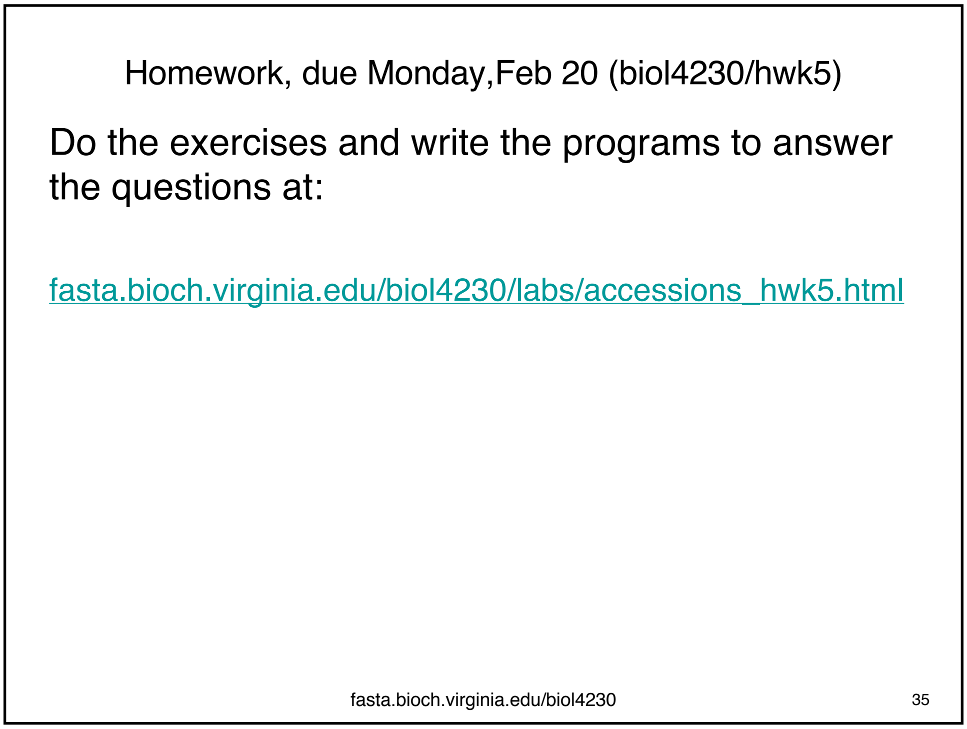
Another last XML:
NCBI esearch.fcgi accessions

```
#!/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):
    loc="https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?"
    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
  www.ncbi.nlm.nih.gov/books/NBK25501/
- Recognizing web addresses (URLs)
- 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



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