UniProt, programmatically

Part of the series: EMBL-EBI, programmatically: take a REST from manual searching

Klemens Pichler
help@uniprot.org
Foundations

• General introduction to RESTful web services:
  • 1\textsuperscript{st} webinar in series \textit{EMBL-EBI, programmatically: take a REST from manual searches}
    • \url{http://www.ebi.ac.uk/training/online/course/embl-ebi-programmatically-take-rest-manual-searches/introduction-programmatic-access}

• General introduction to UniProt (Knowledgebase):
  • \url{http://www.ebi.ac.uk/training/online/course/uniprot-exploring-protein-sequence-and-functional}
  • \url{https://www.youtube.com/user/uniprotvideos}
Contents

• UniProt resources:
  • Which are there?
  • Which can be accessed programmatically?
• Examples:
  1. Querying UniProtKB
  2. Mapping database identifiers
  3. Searching with proteomics peptides
• Summary / Finding help
Resources UniProt provides

UniProtKB
UniProt Knowledgebase

UniRef
Sequence clusters

UniParc
Sequence archive

Proteomes

Supporting data
- Literature citations
- Cross-ref. databases
- Taxonomy
- Diseases
- Subcellular locations
- Keywords

BLAST
SEQUENCE ALIGNMENT
ID MAPPING
PEPTIDE SEARCH
Resources exposed via RESTful URLs on www.uniprot.org

UniProtKB
UniProt Knowledgebase

UniRef
Sequence clusters

UniParc
Sequence archive

Proteomes

Supporting data
- Literature citations
- Cross-ref. databases
- Taxonomy
- Diseases
- Subcellular locations
- Keywords

Example 1

Example 2

ID MAPPING
Additional (large scale) datasets exposed on www.ebi.ac.uk/proteins/api

- Genomic coordinates
- Antigenic regions from HPA
- Protein interactions
- Sequence features
- Natural variants
- Proteomics data
- Protein isoforms
- Taxonomy

Example 3
Example 1 (E1): Querying UniProtKB

- REST endpoint:
  www.uniprot.org/uniprot/ [GET]

- Documentation:
  http://www.uniprot.org/help/api_queries

- Tip:
  - learn parameters and fields using the interactive advanced search on uniprot.org...
  - … and have the URL constructed for you
E1: Querying UniProtKB

![UniProtKB search interface](image)

www.uniprot.org/uniprot/?query=name%3A%22polymerase+alpha%22+AND+taxonomy%3Amus+AND+reviewed%3Ay&sort=score
E1: Querying UniProtKB

This way of learning about fields and parameters is valid for all RESTful services exposed on www.uniprot.org

www.uniprot.org/uniprot/?query=name%3A%22polymerase+alpha%22+AND+taxonomy%3Amus+AND+reviewed%3Ayes&sort=score
E1: Querying UniProtKB programmatically

```
import requests

fullURL = ('http://www.uniprot.org/uniprot/?query=name%3A"polymerase alpha"+'+ 'taxon%3A%5B"Amus"AND+reviewed%3AYes&format=list')
result = requests.get(fullURL)
if result.ok:
    print(result.text)
else:
    print('Something went wrong: ', result.status_code)
```

Output:
```
P33609
P33611
Q61183
P25206
```
E1: Querying UniProtKB programatically

```python
import requests

baseURL = 'http://www.uniprot.org/uniprot/
parameters = '?query=name%3A"polymerase+alpha"+taxonomy%3AUnis+AND+reviewed%3Ayes&format=list'
result = requests.get(baseURL + parameters)
if result.ok:
    print(result.text)
else:
    print('Something went wrong: %d' % result.status_code)
```

Parameters for the request to the server
Ugly! Hard to write by hand!

```text
P33609
P33611
Q81183
P25206
```

baseURL + query = fullURL
E1: Improve parameter handling

import requests

baseURL = 'http://www.uniprot.org/uniprot/
payload = {'query': 'name:"polymerase alpha" AND taxonomy:muS AND reviewed:yes',
'format': 'list'}
result = requests.get(baseURL, params=payload)
if result.ok:
    print(result.text)
else:
    print('Something went wrong: ', result.status_code)

output
P33609
P33611
Q61183
P25206

URL is generated for you!

result.url
'http://www.uniprot.org/uniprot/?query=name%3A%22polymerase+alpha%22+AND+taxonomy%3Amus+AND+reviewed%3Ayes&format=list'
E1: Specify desired data format

```python
import requests

baseUrl = 'http://www.uniprot.org/uniprot/
payload = {'query': 'name:"polymerase alpha" AND taxonomy:mus AND reviewed:yes', 'format': 'txt'}
result = requests.get(baseUrl, params=payload)
if result.ok:
    print(result.text)
else:
    print('Something went wrong: ', result.status_code)
```

We replaced ‘list’ with ‘txt’ to retrieve plain text UniProt entries.

```
<table>
<thead>
<tr>
<th>ID</th>
<th>DPOLA_MOUSE</th>
<th>Reviewed;</th>
<th>1465 AA.</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC</td>
<td>P33609;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DT</td>
<td>01-FEB-1994, integrated into UniProtKB/Swiss-Prot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DT</td>
<td>01-JUN-1994, sequence version 2.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DT</td>
<td>07-JUN-2017, entry version 149.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DE</td>
<td>RecName: Full=DNA polymerase alpha catalytic subunit;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DE</td>
<td>EC=2.7.7.7;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DE</td>
<td>AltName: Full=DNA polymerase alpha catalytic subunit p180;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GN</td>
<td>Name=Polal; Synonyms=Pola;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OS</td>
<td>Mus musculus (Mouse).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OC</td>
<td>Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OC</td>
<td>Mammalia; Eutheria; Euarchontoglires; Glires; Rodentia; Myomorpha;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OC</td>
<td>Muroidea; Muridae; Murinae; Mus; Mus.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OX</td>
<td>NCBI_TaxID=10090;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RN</td>
<td>[1]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RP</td>
<td>NUCLEOTIDE SEQUENCE [MRNA], AND PROTEIN SEQUENCE OF 454-472 AND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RP</td>
<td>1438-1455.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RX</td>
<td>PubMed=8463324;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
# E1: Formats for data retrieval

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>list</td>
<td>A list of accession numbers</td>
<td></td>
</tr>
<tr>
<td>txt</td>
<td>UniProt entries in plain text format</td>
<td></td>
</tr>
<tr>
<td>xml</td>
<td>UniProt entries in XML format</td>
<td></td>
</tr>
<tr>
<td>tab</td>
<td>Selected fields in tabular form</td>
<td>Requires parameter ‘columns’</td>
</tr>
<tr>
<td>fasta</td>
<td>Canonical sequences in FASTA format</td>
<td>Retrieving isoforms as well requires parameter ‘include’: ‘yes’</td>
</tr>
<tr>
<td>gff</td>
<td>Annotated sequence features in GFF format</td>
<td></td>
</tr>
<tr>
<td>RDF</td>
<td>UniProt entries in RDF format</td>
<td></td>
</tr>
</tbody>
</table>

These are data formats for [www.uniprot.org/uniprot/][1] [GET]

Formats vary depending on which REST endpoint is used.

[1]: www.uniprot.org/uniprot/
E1: Using the tabular data format

```python
import requests

baseURL = 'http://www.uniprot.org/uniprot/
payload = {'query': 'name:"DNA polymerase alpha" AND taxonomy:mouse AND reviewed:yes', 
'format': 'tab', 
'columns': 'id,entry_name,reviewed,protein_names,organism,ec,keywords', 
# 'columns': ','.join(['id', 'entry_name', 'reviewed', 'protein_names', 
#    'organism', 'ec', 'keywords']), 
}

result = requests.get(baseURL, params=payload)
if result.ok:
    print(result.text)
else:
    print('Something went wrong: ', result.status_code)
```

Output example:

<table>
<thead>
<tr>
<th>Entry</th>
<th>Entry name</th>
<th>Status</th>
<th>Protein names</th>
<th>Organism</th>
<th>EC number</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>P33609</td>
<td>DPOLA_MOUSE</td>
<td>reviewed</td>
<td>DNA polymerase alpha catalytic subunit (EC 2.7.7.7) (DNA polymerase alpha catalytic subunit p180) Mus musculus (Mouse)</td>
<td>2.7.7.7 4Fe-4S; Acetylation; Complete proteome; Cytoplasm; DNA replication; DNA-binding; DNA-directed DNA polymerase; Direct protein sequencing; Iron; Iron-sulfur; Metal-binding; Nucleotidyltransferase; Nucleus; Phosphoprotein; Reference proteome; Transferase; Zinc; Zinc-finger</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P33611</td>
<td>DPOA2_MOUSE</td>
<td>reviewed</td>
<td>DNA polymerase alpha subunit B (DNA polymerase alpha 70 kDa subunit) Mus musculus (Mouse) Complete proteome; DNA replication; Direct protein sequencing; Nucleus; Phosphoprotein; Reference proteome</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
E2: Using ID mapping / batch retrieval

- REST endpoint: www.uniprot.org/uploadlists/ [GET]

- Documentation:
  - http://www.uniprot.org/help/uploadlists
  - http://www.uniprot.org/help/api_idmapping

- Map UniProt -> other DB identifiers
- Map other DB identifiers -> UniProt
- Retrieve UniProt entries
- Map other DB -> yet another DB
  - other DB -> UniProt -> yet another DB
E2: Using ID mapping / batch retrieval

```python
import requests

def map_retrieve(ids2map, source_fmt='ACC+ID', target_fmt='ACC', output_fmt='list', payload=None):
    baseURL = 'http://www.uniprot.org/uploadlists/
    if hasattr(ids2map, 'pop'):
        ids2map = '.join(ids2map)
    payload = {}
    response = requests.get(baseURL, params=payload)
    if response.ok:
        return response.text
    else:
        response.raise_for_status()"
```

- **IDs we would like to map**
- **Default values**
- **REST endpoint**
- **Optional. If a list is provided it will be concatenated**
- **Required parameters**
- **We return data instead of printing directly.**
E2: Map PDB IDs to UniProt and retrieve TXT

PDB IDs we would like to map

```python
uniprot_acc = map_retrieve(['YCS', '2AC0', '5UN8', '4S2Q'], source_fmt='PDB_ID', output_fmt='txt')
print(uniprot_acc)
```

<table>
<thead>
<tr>
<th>ID</th>
<th>ASPP2_HUMAN</th>
<th>Reviewed;</th>
<th>1128 AA.</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AC</td>
<td>Q13625; B4DG66; Q12892; Q86X75; Q96KQ3;</td>
<td>15-JUL-1998, integrated into UniProtKB/Swiss-Prot.</td>
<td>15-AUG-2003, sequence version 2.</td>
<td>07-JUN-2017, entry version 183.</td>
</tr>
<tr>
<td>DE</td>
<td>RecName: Full=Aptosis-stimulating of p53 protein 2; AltName: Full=Bcl2-binding protein; Short=Bbp; AltName: Full=Renal carcinoma antigen NY-REN-51; AltName: Full=Tumor suppressor p53-binding protein 2; Short=53BP2; Short=p53-binding protein 2; Short=p53BP2;</td>
<td>Name=TP53BP2; Synonyms=ASPP2, BBP;</td>
<td>Homo sapiens (Human).</td>
<td>Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi;</td>
</tr>
</tbody>
</table>
E2: Map PDB IDs to ENSEMBL via UniProt

```python
uniprot_acc = map_retrieve(['1YCS', '2AC0', '5UN8', '4S2Q'], source_fmt='PDB_ID')
print(uniprot_acc)
```

```
Q13625
060502
P04637
q04887
```

No parameter output_fmt as it defaults to ‘list’ anyway
Additional (large scale) datasets exposed on www.ebi.ac.uk/proteins/api

- Genomic coordinates
- Antigenic regions from HPA
- Protein interactions
- Sequence features
- Natural variants
- Proteomics data
- Protein isoforms
- Taxonomy

PEPTIDE SEARCH
E3: Searching for proteomics peptides

Proteins API

Proteins

Proteins inc. isoforms
The Proteins REST API provides access to key biological data from UniProt and data from Large Scale Studies (LSS) mapped to UniProt. The services provide sequence feature annotations from UniProtKB, variation data from UniProtKB and mapped from LSS (1000 Genomes, ExAC and COSMIC), proteomics data mapped from LSS (PeptideAtlas, MaxQB and EPD), antigen sequences mapped from Human Protein Atlas (HPA), proteomes and taxonomy search and retrieval, and reference genome coordinate mappings.

The base URL for the API is www.ebi.ac.uk/proteins/api.

How to cite us: The proteins API: accessing key integrated protein and genome information (doi:10.1093/nar/gkx237)

Proteins

Example on following slides
REST endpoint: [www.ebi.ac.uk/proteins/api/proteomics [GET]]

1. Choose data format (varies depending on endpoint)

2. Provide parameters. We use just ‘peptide’ as an example.

3. Click to test this RESTful service.
E3: Result of sample query in specified format

```
{
  "accession": "P28562",
  "entryName": "DUS1_HUMAN",
  "sequence": "MEVEVGLDAGLRALLGERAACLCLDCLSFFAFNAGHIAGSVNRFSTIVRRRAKGAMGLEHIVPNANEGRRLLAGAYHAYVLLDDERSAALDGA
KRDGTALAGALCREARAAQVFPLKGGEAFSASCPELCSKQTSTMGLSPLSTTSVDPSESGCSSCSTPLYQGGPVEILPFLYLGSAYHASRKD
MLDALGITALINVSANCPNHQEGHYQYSIPPEDNHDJADISSWFNEAIDFIDSIKNAGGRVFVHCQAGISRSATICLAYLMRTNRVKLDEAFEFVKQ
RRSIISPNSFMGQLLQFESQVLAPHCSAEAGSPADVLRDTTTTFNFPFVHSTNASLYLQSPITTSFC",
  "sequenceChecksum": "11BD1D39A9FCD51F",
  "taxid": 9606,
  "features": [
    {
      "type": "PROTEOMICS",
      "begin": "281",
      "end": "289",
    }
  ]
}
```
E3: URL and sample code are provided

**Request URL**

https://www.ebi.ac.uk/proteins/api/proteomics?offset=0&size=100&peptide=rvlslgr%2Cldefefvkr

**Request Sample Code**

```python
import requests, sys

requestURL = "https://www.ebi.ac.uk/proteins/api/proteomics?offset=0&size=100&peptide=rvlslgr%2Cldefefvkr"

r = requests.get(requestURL, headers={ "Accept" : "application/json"})

if not r.ok:
    r.raise_for_status()
    sys.exit()

responseBody = r.text
print(responseBody)
```
E3: Using the sample code

```python
import requests, sys

requestURL = "https://www.ebi.ac.uk/proteins/api/proteomics?offset=0&size=100&peptide=rvlslgr%2C1deafefvk"

r = requests.get(requestURL, headers={ "Accept" : "application/json"})

if not r.ok:
    r.raise_for_status()
    sys.exit()

responseBody = r.text
print(responseBody)

{"accession": "P28562", "entryName": "DUS1_HUMAN", "sequence": "MVMEVGTLDAGGLRALGERAAAQCLLDDCRSFAFNAGHIAGSVNVRFSTIVRRRAKAGMLEHIVPNAELRGRLLAGAYHAVVLLDERSAALDGAKRDTLALLAGAALCREAREAQVFPLGGYEAFSASCPELCSSQSTPMGLSLPLSVDSAESEGSCTCPFLYDQGPVEILFGLGSAHASYRKLMLDGALTIANWSANCNPNFEGHYQKSIPVEDNHKADISWSWNEAIDFISIKNAGGRVFHVCAQGISRSAATCLAILMRTVRKLDFAEFVQQRRIISPFSFMGQLLLFEQSVLAPHCSEAAGSPAMAVLDRGTSTTTVFNPVSVIFVHSTNSALSYLQPITTSPSC", "sequenceChecksum": "11BD1D39A9FCD51F", "taxid": 9606, "features": [{"type": "PROTEOMICS", "begin": "281", "end": "289", "xrefs": [{"name": "Proteomes", "id": "UP000005640", "url": "http://www.uniprot.org"}]}]"
```
E3: Modifications for cleaner parameters

```python
import requests, sys

baseURL = "https://www.ebi.ac.uk/proteins/api/proteomics"

payload = {'peptide': 'rvlslgr,ldeafefvk'}
# If starting from a Python list:
#payload = {'peptide': ','.join(['rvlslgr', 'ldeafefvk'])}

r = requests.get(baseURL, params=payload, headers={'Accept': 'application/json'})

if not r.ok:
    r.raise_for_status()
    sys.exit()

responseBody = r.text
print(responseBody)
```

Values for parameters can now be provided in a nicer way.
Summary / Finding help

- UniProt provides many RESTful service endpoints
- 2 base URLs:
- Explore/learn about [1] via advanced search on uniprot.org
- Explore/learn about [2] via the interactive documentation
- Find help/docs:
  - http://www.uniprot.org/help/programmatic_access
  - http://www.uniprot.org/help/api
  - http://www.ebi.ac.uk.proteins/api
- Questions: help@uniprot.org