

Accessing NCBI's Entrez databases

using Biopython's Bio.Entrez module

Antonio B.

Health Engineering, Advanced Programming in Bioinformatics
University of Málaga

April 18, 2017



You can find everything about how to use Bio.Entrez on
[http://biopython.org/DIST/docs/tutorial/Tutorial.
html#htoc110](http://biopython.org/DIST/docs/tutorial/Tutorial.html#htoc110)

NCBI's Guidelines

If the NCBI finds you are abusing their systems, they can **AND** will ban your access!

Gold rule: Make no more than three requests every seconds (relaxed from at most one request every three seconds in early 2009).

Use the optional email parameter so the NCBI can contact you if there is a problem.

e.g.

```
1 • from Bio import Entrez
2 • Entrez.email = "antonio.b@uma.es"  # !!
```

NCBI's Guidelines

Also, if you are using BioPython within some larger software suite, use the tool parameter to specify this.

e.g.

```
1 • from Bio import Entrez
2 • Entrez.tool = "MyLocalScript"
```

...If you plan to download lots of data, consider other options!!

Searching the Entrez databases

```
1 • handle = Entrez.esearch(db="nucleotide", term="
    Mycoplasma leachii complete genome")
2 • record = Entrez.read(handle) # to parse the XML
    data
3 • print(record)
4 {'Count': '5', 'RetMax': '5', 'RetStart': '0', '
    IdList': ['313664890', '526641971', '526641931',
    '526641891', '312949189'], 'TranslationSet':
    [...], 'TranslationStack': [...], '
    QueryTranslation': '("Mycoplasma leachii PG50"[
    Organism] OR Mycoplasma leachii pg50[All Fields
    ]) AND clone[All Fields] AND complete[All
    Fields] AND genome[All Fields]'}

```

Searching the Entrez databases

record is an object, so you can use:

```
1 • print(record["IdList"])
2 ['313664890', '526641971', '526641931', '526641891',
   '312949189']
```

We will use EFetch to obtain more information for each of these entries IDs.

Downloading full records from Entrez

```
1 • handle = Entrez.efetch(db="nucleotide", id="
    313664890", rettype="gb", retmode="text")
2 • print handle.read()
```

The result is:

```
LOCUS      NC_014751          1008951 bp    DNA      circular CON 30-JUL-2015
DEFINITION Mycoplasma leachii PG50 clone MU clone A8, complete genome.
ACCESSION  NC_014751
VERSION    NC_014751.1
DBLINK     BioProject: PRJNA224116
...
```

The arguments `retmode="text"` and `rettype="gb"` let us download this record in the GenBank format.

retmode

Retrieval mode. This parameter specifies the data format of the records returned, such as plain text, HTML or XML.

rettype

Retrieval type. This parameter specifies the record view returned, such as Abstract or MEDLINE from PubMed, or GenPept or FASTA from protein.

Demo code

Function to get a list of id's from file.

```
1 • def get_list(filename):
2 •     idList = []
3 •     with open(filename, 'r') as f:
4 •         for line in f:
5 •             idList.append(line.strip())    # a
        newline character (\n) is left at the end of
        the string
6 •     return idList
```

Demo code

Function to retrieve a record from NCBI using his id number.

```
1 • def get_record_from_id(id_code):  
2 •     handle = Entrez.efetch(db="nucleotide", id=  
    id_code, rettype="fasta", retmode="text")  
    # this time we use rettype="fasta"  
3 •     record = handle.read()  
4 •     handle.close()  
5 •     return record
```

Demo code

Main program.

```
1 • if __name__ == "__main__":  
2 •     idList = get_list('idList.txt')  
3  
4 •     for id_code in idList:  
5 •         record = get_record_from_id(id_code)  
6 •         with open(id_code+".fasta", "w") as  
           output_fasta:  
7 •             output_fasta.write(record)
```

Demo code (2)

Better main program.

```
1 • if __name__ == "__main__":  
2 •     idList = get_list('idList.txt')  
3  
4 •     ids = ",".join(idList)  
5 •     records = get_record_from_id(ids)  
6  
7 •     with open("records.fasta","w") as output_fasta:  
8 •         output_fasta.write(records)
```

Now you can use `Bio.SeqIO` module to parse the multifasta file.

...When you make a request with EFetch your list of IDs, the database etc, are all turned into a long URL sent to the server. If your list of IDs is long, this URL gets long, and long URLs can break.

Sol:

We can use EPost to upload a list of identifiers, which starts a new history session. You then download the records with EFetch by referring to the session (instead of the identifiers).

Using the history and WebEnv

```
1 • handle = Entrez.epost("pubmed", id=ids)
2 • print(handle.read())
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE ePostResult PUBLIC "-//NLM//DTD epost 20090526//EN" "https://eutils.ncbi.nlm.nih..."><ePostResult>
  <QueryKey>1</QueryKey>
  <WebEnv>NCID_1_69751425_130.14.18.34_9001_1490028628_783976133_0MetA0_S_MegaStore_F_1</WebEnv>
</ePostResult>
```

```
1 • session = Entrez.read(handle)
2 • webenv = session["WebEnv"]
3 • query_key = session["QueryKey"]
4 • print(webenv + ';' + query_key)
5 NCID_1_53970299_..._0MetA0_S_MegaStore_F_1;1
```

Demo code (3)

The best main program.

```
1  • if __name__ == "__main__":
2  •     idList = get_list('idList.txt')
3  •     ids = ",".join(idList)
4
5  •     handle = Entrez.epost("nucleotide", id=ids)
6  •     session = Entrez.read(handle)
7
8  •     webenv = session["WebEnv"]
9  •     query_key = session["QueryKey"]
10
11  •     records = get_record_from_id(ids, webenv,
    query_key)
12
13  •     with open("records.fasta", "w") as output_fasta:
14  •         output_fasta.write(records)
```

LOONEY TUNES



"That's all Folks!"

Source from Bitbucket.

You can download the code used in this example (along with this slides) from:

https://github.com/benhid/demo_entrez_biopython.git

Prerequisites:

1 BioPython 1.68

```
conda install -c anaconda biopython=1.68
```

2 Python 3.x

3 Internet connection!!

You can set a proxy within Python at the start of your code using `os.environ`