Lecture 21

408/508 *Computational Techniques for Linguists*

Today's Topics

- `cgi-bin` with Python on *CMU Pronouncing Dictionary*
  - (*as promised last time*)
- LIVE PROGRAMMING TODAY!

1. `cgi-bin` works with any programming language!
2. `#!/path/to/python3` (1st line: may vary on your computer)
3. `which python3` (reports executable path)
   - `/Library/Frameworks/Python.framework/Versions/3.14/bin/python3`
4. form submission (client `.html` -> server `cgi-bin` program)

Scheduling announcement

from Dec 2nd, I'll be away for research talks in Japan



Scheduling announcement

- this means for class scheduling:
  - no lectures in December (3rd and 8th)
  - Nov 26th is *Thanksgiving*
- last days of 408/508 lectures will be Nov 24th and Dec 1st
  - after that date, you should be working on your term project!
  - next week, I'll ask for project proposals

Last time: `CMUDict`

```
~$ python3
Python 3.14.7 (v3.14.7:823f0323ee6, Aug  5 2026, 07:07:01) [Clang 21.0.0 (clang-
2100.1.1.101)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> from nltk.corpus import cmudict
>>> d = cmudict.dict()
>>> d['tomato']
[['T', 'AH0', 'M', 'EY1', 'T', 'OW2'], ['T', 'AH0', 'M', 'AA1', 'T', 'OW2']]
>>> d['trump']
[['T', 'R', 'AH1', 'M', 'P']]
>>> d['obama']
[['OW2', 'B', 'AA1', 'M', 'AH0']]
>>> [[ph for ph in p if ph[-1].isdigit()] for p in d['obama']]
[['OW2', 'AA1', 'AH0']]
>>> [[ph for ph in p if ph[-1].isdigit()] for p in d['tomato']]
[['AH0', 'EY1', 'OW2'], ['AH0', 'AA1', 'OW2']]
>>> [len([ph for ph in p if ph[-1].isdigit()]) for p in d['tomato']]
[3, 3]
>>>
```

macOS: ~/Sites/cmudictpy.cgi

[for Linux, use location: `~/public_html/cmudictpy.cgi`]

- File `cmudictpy.cgi` is on the course website (*a Python program*)
- Assume we've enabled the userdir module in Apache2 (*last lecture*)
- `chmod a+x cmudictpy.cgi` (permits execution of file for all)

cmudictpy.cgi

```
#!/Library/Frameworks/Python.framework/Versions/3.14/bin/python3
print("Content-type: text/plain\n")
import os
import sys
import nltk
# assume we didn't install nltk_data in the common area:
#   macOS: /usr/local/share/nltk_data
#   Linux: /usr/share/nltk_data
nltk.data.path.append('/Users/sandiway/nltk_data')
from nltk.corpus import cmudict
d = cmudict.dict()
# works for word from:
#  command line (argv[1]) or
#  environment variable QUERY_STRING
if len(sys.argv) > 1:
    word = sys.argv[1]
else:
    word = os.getenv("QUERY_STRING", "")
if word:
    print(f"CMUdict lookup for: {word}")
    if word in d:
        syls = [len([ph for ph in p if ph[-1].isdigit()]) for p in d[word]]
        print(f"Number of syllables: {syls}")
    else:
        print(f"Word {word} not found!")
else:
    print("No word supplied!")
```

cmudictpy.cgi

cmudict is part of nltk_data!

For central installation on a multi-user machine, do the following from an administrator account.

Run the Python interpreter and type the commands:

```
>>> import nltk
>>> nltk.download()
```

A new window should open, showing the NLTK Downloader. Click on the File menu and select Change Download Directory. For central installation, set this to `C:\nltk_data` (Windows), `/usr/local/share/nltk_data` (Mac), or `/usr/share/nltk_data` (Unix).

```
>>> import nltk
>>> print(nltk.data.path)
['/Users/sandiway/nltk_data',
'/Library/Frameworks/Python.framework/Versions/3.14
/nltk_data',
'/Library/Frameworks/Python.framework/Versions/3.14
/share/nltk_data',
'/Library/Frameworks/Python.framework/Versions/3.14
/lib/nltk_data', '/usr/share/nltk_data',
'/usr/local/share/nltk_data', '/usr/lib/nltk_data',
'/usr/local/lib/nltk_data']
```

- **Tricky**: we'll do this from the user side – user _www (*not the sudo side*)
  - `nltk.data.path.append('/Users/sandiway/nltk_data')`

cmudictpy.cgi

running as user _www, it won't see my ~/nltk_data!

```
$ tail /var/log/apache2/error_log  (Linux: error.log)
```

- [cgi:error] [pid 39328] [client ::1:61062] AH01215: stderr from /Users/sandiway/Sites/cmudictpy.cgi: - '/Library/Frameworks/Python.framework/Versions/3.14/nltk_data'

Not installed there!
(I installed nltk_data without sudo in ~/nltk_data)

cmudictpy.cgi

sys.argv

The list of command line arguments passed to a Python script. `argv[0]` is the script name (it is operating system dependent whether this is a full pathname or not). If the command was executed using the `-c` command line option to the interpreter, `argv[0]` is set to the string `'-c'`. If no script name was passed to the Python interpreter, `argv[0]` is the empty string.

- Let's try it on the command line first!
  - `import sys`
  - `#  command line (argv[1])`
- Terminal:

```
Sites$ ./cmudictpy.cgi tomato
Content-type: text/plain


CMUdict lookup for: tomato
Number of syllables: [3, 3]
```

```
Sites$ ./cmudictpy.cgi tomo
Content-type: text/plain

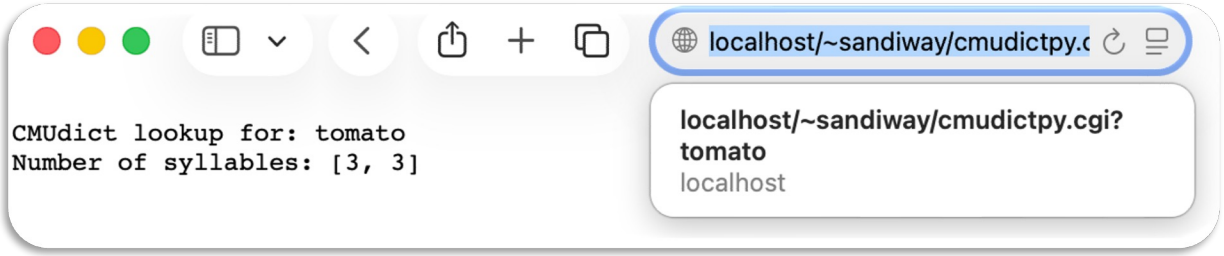

CMUdict lookup for: tomo
Word tomo not found!
```

```
Sites$ ./cmudictpy.cgi
Content-type:
text/plain


No word supplied!
```

cmudictpy.cgi

- Let's try it on the browser too!
  - URL: `http://localhost/~username/cmudictpy.cgi?word`



  - later, we will use a form instead of putting the word directly into the URL string

html version of `cmudictpy.cgi`

Python CMUDict

CMUdict lookup for: *tomato*

Pronunciation: [['T', 'AH0', 'M', 'EY1', 'T', 'OW2'], ['T', 'AH0', 'M', 'AA1', 'T', 'OW2']]

Number of syllables: [3, 3]

Arpabet: symbol table

Python CMUDict

No word supplied!

Python CMUDict

CMUdict lookup for: *putin*

Word putin not found!

Arpabet: symbol table

```
https://en.wikipedia.org/wiki/ARPABET#Symbols
```

html version of `cmudictpy.cgi`

- Add Python code to emit html code directly, e.g.:

```
print("Content-type: text/html; charset=utf-8\n")
print("<html><head></head><body><h1>Python CMUDict</h1>")
…
 print(f"<p>CMUdict lookup for: <i>{word}</i></p>")
…
 print(f'<p>Pronunciation: {ps}</p>')
…
print(f"<p>Number of syllables: {syls}</p>")
…
 print(f"<span style=\"color:red\">Word {word} not
found!</span>")
```

html version of `cmudictpy.cgi`

- Simple version here: case folding not done.

Python CMUDict

CMUdict lookup for: *tomato*

Pronunciation: [['T', 'AH0', 'M', 'EY1', 'T', 'OW2'], ['T', 'AH0', 'M', 'AA1', 'T', 'OW2']]

Number of syllables: [3, 3]

Arpabet: symbol table

Python CMUDict

CMUdict lookup for: *Tomato*

Word Tomato not found!

Arpabet: symbol table

Using a <form>

- Instead of editing the URL string, we will use:
  - `<form action="filename.cgi" method="GET">`
  - plus `<input name="word" type="text">` elements
  - plus a `<input type="submit">` element
- it will send the form to the environment variable QUERY_STRING:
  - `import os`
  - `word = os.getenv("QUERY_STRING", "")` `word=tomato`

`os.`**`getenv`**`(`*`key, default=None`*`)`

Return the value of the environment variable *key* as a string if it exists, or *default* if it doesn't. *key* is a string. Note that since `getenv()` uses `os.environ`, the mapping of `getenv()` is similarly also captured on import, and the function may not reflect future environment changes.

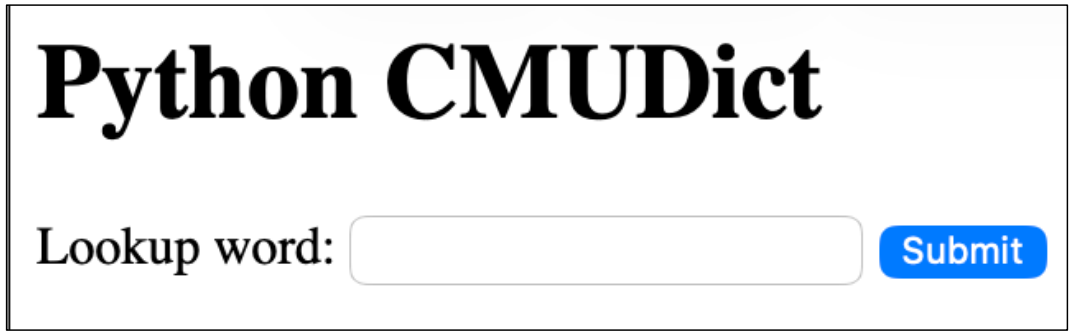
Using a <form>

`cmudict.html`

place it in the same directory `~/Sites` (`~/public_html`)

accessed as `http://localhost/~username/cmudict.html`

Browser:




```
<form action="filename.cgi" method="GET">
```

`cmudictpy.cgi`

QUERY_STRING

`word=tomato`

QUERY_STRING: word=(.*) (regex)

```
# works for word from:
#  command line (argv[1]) or
#  environment variable QUERY_STRING
if len(sys.argv) > 1:
    word = sys.argv[1]
else:
    string = os.getenv("QUERY_STRING", "")
    match = re.match(r"word=(.*)", string)
    if match:
        word = match.group(1)
    else:
        word = ""
```

QUERY_STRING: word=(.*) (regex)

re.**match**(*pattern*, *string*, *flags=0*)

> *Soft deprecated since version 3.15:* `match()` has been soft deprecated in favor of the alternate `prefixmatch()` name of this API which is more explicitly descriptive. Use it to better express intent. The norm in other languages and regular expression implementations is to use the term *match* to refer to the behavior of what Python has always called `search()`. See prefixmatch() vs. match().

word=tomato

```
match = re.match(r"word=(.*)", string)
    if match:
        word = match.group(1)
    else:
        word = ""
```