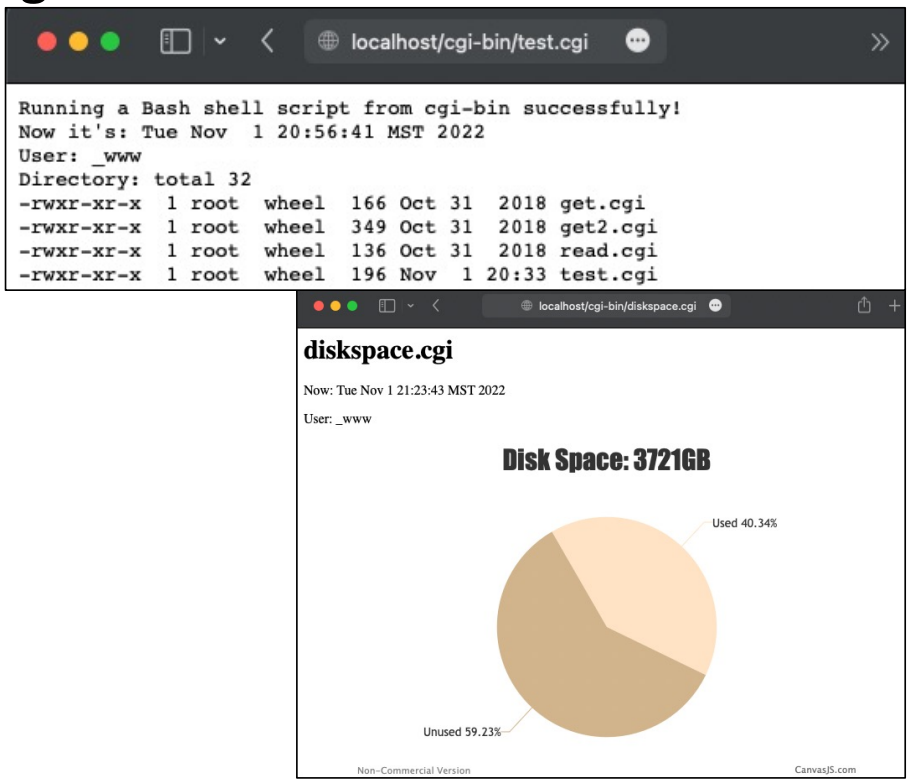
Lecture 19

408/508 *Computational Techniques for Linguists*

Today's Topics

- Last time we discussed running programs on the webserver itself.
  - two examples: one `text/plain`,
  - one `text/html`
  - both from the default `cgi-bin` directory

1. Today, running a program from inside your home directory

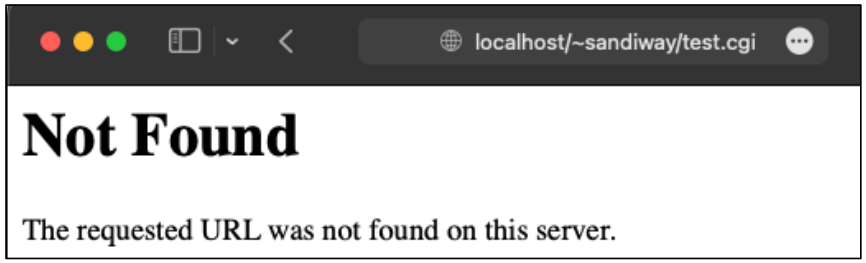


Apache2 `cgi-bin` directories

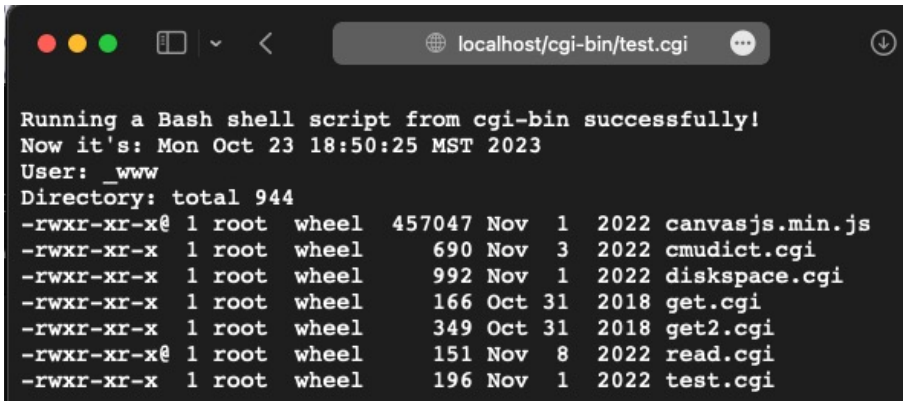
- Default `cgi-bin` directory
  - `http://localhost/cgi-bin/`*`test.cgi`*
  - macOS: `/Library/WebServer/CGI-Executables/`
  - Ubuntu: `/usr/lib/cgi-bin/`
- User directories
  - `http://localhost/`*`~username`*`/`*`test.cgi`*
  - macOS: `~/Sites`
  - Ubuntu: `~/public_html`
  - *Users are not permitted access to the default cgi-bin directory*
  - We can permit cgi-bin scripts in **a designated user directory**

Apache2 Webserver on macOS

- cgi-bin not working from user directory!



- cgi-bin working from default directory!



- Note:
  - http://localhost/~user/test.cgi should map to ~user/Sites/test.cgi
  - By default, it's turned off (*for security*)
  - Even if httpd is running … check with `ps –ax | grep httpd`

Apache2 Webserver on macOS

- To run programs in `~/Sites`, i.e. outside of `/Library/WebServer/CGI-Executables`
  - modify the Apache2 httpd configuration file:
  - `sudo nano /etc/apache2/httpd.conf`
  - `# To use CGI scripts outside of ScriptAliased directories:`
- After modifying:
  - `sudo apachectl stop`
  - `sudo apachectl start`

Apache2 Webserver on macOS

- `/etc/apache2/httpd.conf`

```
176#LoadModule dav_fs_module libexec/apache2/mod_dav_fs.so
177#LoadModule dav_lock_module libexec/apache2/mod_dav_lock.so
178#LoadModule vhost_alias_module libexec/apache2/mod_vhost_alias.so
179LoadModule negotiation_module libexec/apache2/mod_negotiation.so
180LoadModule dir_module libexec/apache2/mod_dir.so
181#LoadModule imagemap_module libexec/apache2/mod_imagemap.so
182#LoadModule actions_module libexec/apache2/mod_actions.so
183#LoadModule speling_module libexec/apache2/mod_speling.so
184#LoadModule userdir_module libexec/apache2/mod_userdir.so
185LoadModule alias_module libexec/apache2/mod_alias.so
186#LoadModule rewrite_module libexec/apache2/mod_rewrite.so
187#PHP was deprecated in macOS 11 and removed from macOS 12
188#LoadModule perl_module libexec/apache2/mod_perl.so
189LoadModule hfs_apple_module libexec/apache2/mod_hfs_apple.so
```

  - *relevant directive is UserDir*

Apache2 Webserver on macOS

- `/etc/apache2/httpd.conf`

```
438
439     #
440     # AddHandler allows you to map certain file extensions to "handlers":
441     # actions unrelated to filetype. These can be either built into the server
442     # or added with the Action directive (see below)
443     #
444     # To use CGI scripts outside of ScriptAliased directories:
445     # (You will also need to add "ExecCGI" to the "Options" directive.)
446     #
447     AddHandler cgi-script .cgi
448
```

uncomment

  - *invokes the cgi-script handler for all files of type .cgi*
    - after this, all our scripts should have extension .cgi

Apache2 Webserver on macOS

- `/etc/apache2/httpd.conf`

```
501# Supplemental configuration¶
502#¶
503# The configuration files in the /private/etc/apache2/extra/ directory can be ¶
504# included to add extra features or to modify the default configuration of ¶
505# the server, or you may simply copy their contents here and change as ¶
506# necessary.¶
507¶
508# Server-pool management (MPM specific)¶
509Include /private/etc/apache2/extra/httpd-mpm.conf¶
510¶
511# Multi-language error messages¶
512#Include /private/etc/apache2/extra/httpd-multilang-errordoc.conf¶
513¶
514# Fancy directory listings¶
515Include /private/etc/apache2/extra/httpd-autoindex.conf¶
516¶
517# Language settings¶
518#Include /private/etc/apache2/extra/httpd-languages.conf¶
519¶
520# User home directories¶
521#Include /private/etc/apache2/extra/httpd-userdir.conf¶
```

uncomment

Apache2 Webserver on macOS

- `/etc/apache2/extra/httpd-userdir.conf`

```
# Settings for user home directories
#
# Required module: mod_authz_core, mod_authz_host, mod_userdir

#
# UserDir: The name of the directory that is appended onto a user's home
# directory if a ~user request is received.  Note that you must also set
# the default access control for these directories, as in the example below.
#
UserDir Sites

#
# Control access to UserDir directories.  The following is an example
# for a site where these directories are restricted to read-only.
#
#Include /private/etc/apache2/users/*.conf
<IfModule bonjour_module>
        RegisterUserSite customized-users
</IfModule>
```

uncomment

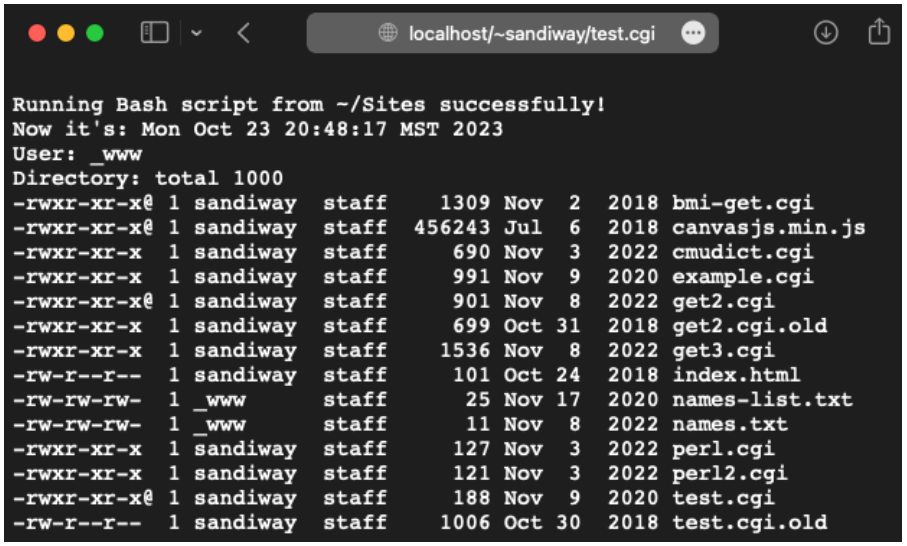
Apache2 Webserver on macOS

- `/etc/apache2/users/username.conf`

```
<Directory "/Users/sandiway/Sites/">
        AllowOverride All
        Options Indexes MultiViews FollowSymLinks ExecCGI
        Require all granted
</Directory>
```

  - *create this file for your username if it doesn't exist*
    - *must have ExecCGI there!*

Apache2 Webserver on macOS



check permissions must be executable
`-rwxr-xr-x`

- ~*username*/Sites/test.cgi

```
1 #!/bin/bash
2 echo "Content-Type: text/plain"
3 echo
4 echo "Running Bash script from ~/Sites successfully!"
5 echo -n "Now it's: "
6 date
7 echo -n "User: "
8 whoami
9 echo -n "Directory: "
10 ls -l
11 exit 0
```

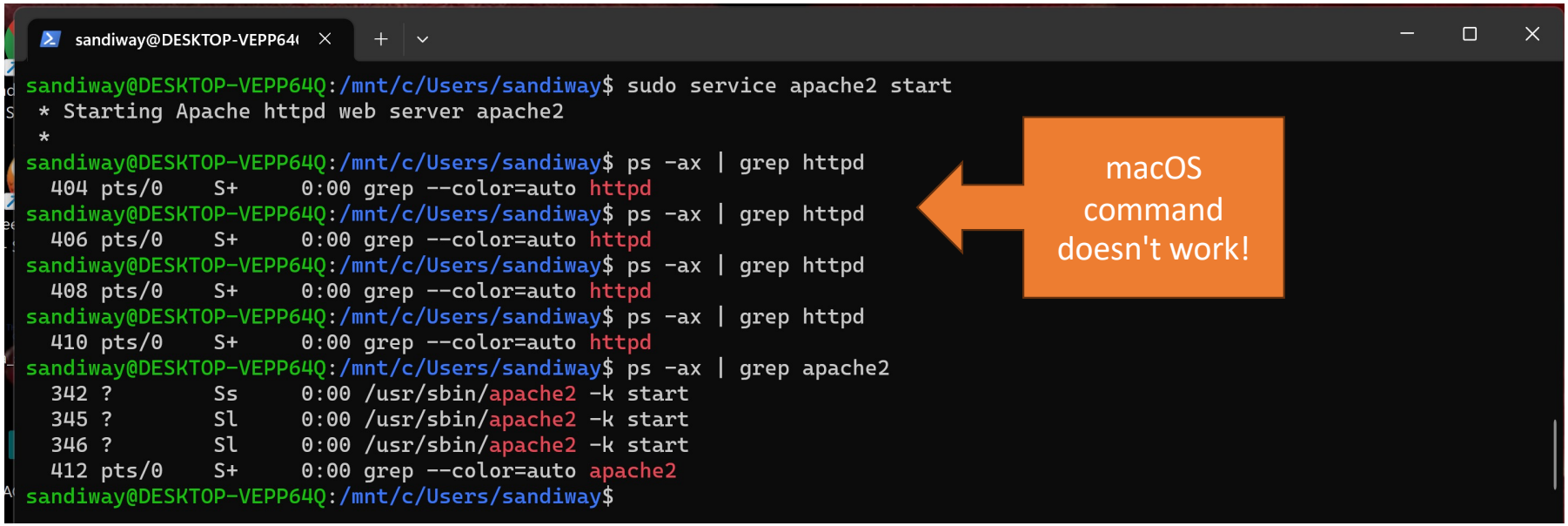
Same as test.cgi on course website except for echo here

Note the format of the first 3 lines:

1. `#!/bin/bash` (*use bash to run this*)
2. `Content-type: text/html` (*echo*)
3. (blank line) (*echo*)

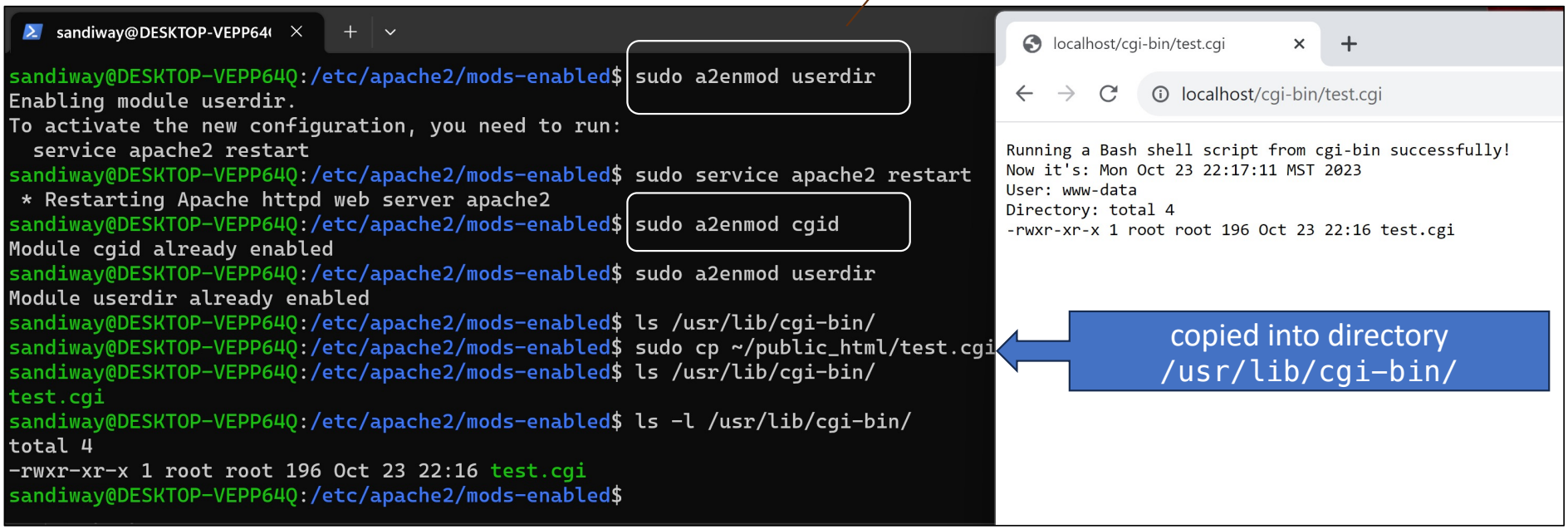
Apache Webserver on Ubuntu

- `Windows Powershell: wsl (or ubuntu)`
- `sudo service apache2 start`
- To see if apache2 is running `ps –ax | grep apache2`:



Apache Webserver on Ubuntu

- Last time (a2enmod for Ubuntu version):



Apache Webserver on Ubuntu

- https://manpages.ubuntu.com/manpages/jammy/en/man8/a2enmod.8.html

NAME

a2enmod, a2dismod – enable or disable an apache2 module

SYNOPSIS

a2enmod [-q|--quiet] [-m|--maintmode] [module]

a2dismod [-q|--quiet] [-f|--force] [-m|--maintmode] [-p|--purge] [module]

DESCRIPTION

This manual page documents briefly the **a2enmod** and **a2dismod** commands.

a2enmod is a script that enables the specified module within the **apache2** configuration. It does this by creating symlinks within **/etc/apache2/mods-enabled**. Likewise, **a2dismod** disables a module by removing those symlinks. It is not an error to enable a module which is already enabled, or to disable one which is already disabled.

Note that many modules have, in addition to a .load file, an associated .conf file. Enabling the module puts the configuration directives in the .conf file as directives into the main server context of **apache2**.

Apache2 Webserver on Ubuntu

By default, `cgi-bin` is not enabled for `~/public_html` (`~/Sites` on macOS)

- https://httpd.apache.org/docs/2.4/howto/cgi.html
- **CGI outside of ScriptAlias directories**
- CGI programs are often restricted to ScriptAlias'ed directories for security reasons. In this way, administrators can tightly control who is allowed to use CGI programs. However, if the proper security precautions are taken, there is no reason why CGI programs cannot be run from arbitrary directories.
- For example, you may wish to let users have web content in their home directories with the UserDir directive.

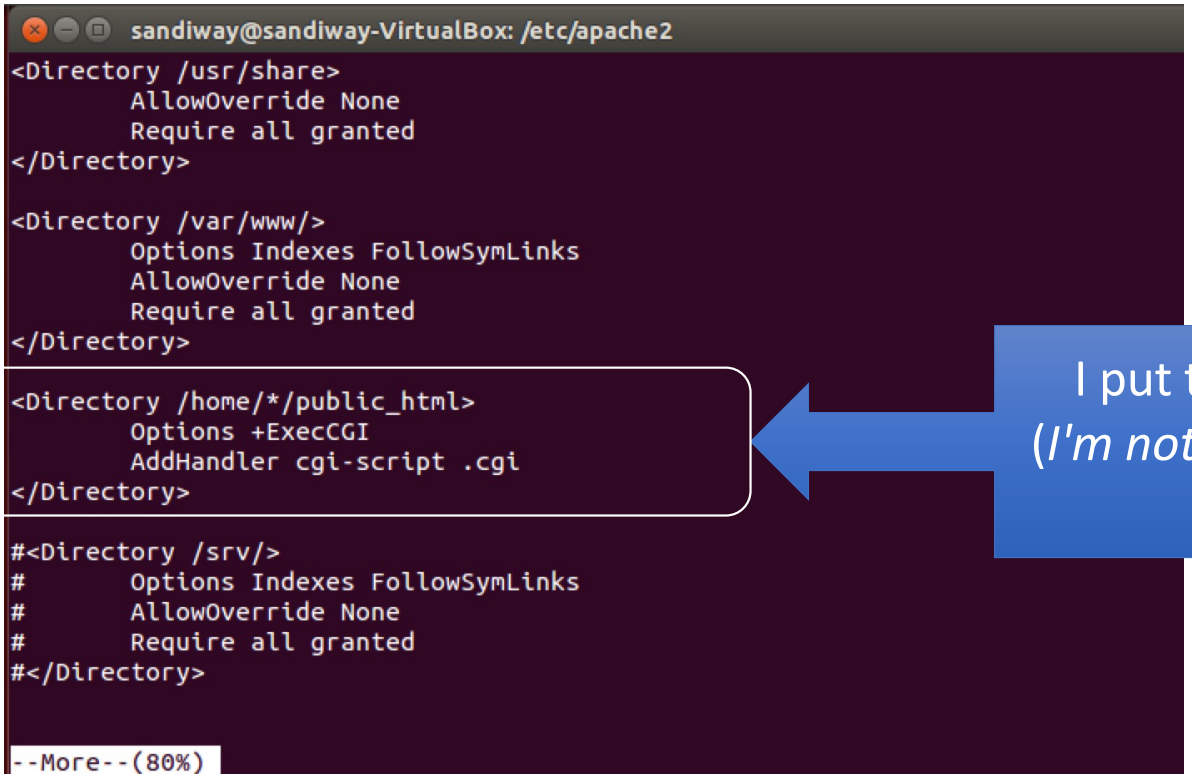
`mkdir ~/public_html` (if directory doesn't exist)
~ should be `/home/`*`username`* (`cd ~` *should take you there*)
`/mnt/c/Users/`*`username`* (`default on WSL` *is not your Ubuntu home!*)

Apache Webserver on Ubuntu

- From https://httpd.apache.org/docs/current/howto/cgi.html
  - add these lines to `/etc/apache2/apache.conf`
    - `<Directory "/home/*/public_html">`
    - `    Options +ExecCGI`
    - `    AddHandler cgi-script .cgi`
    - `</Directory>`
  - and restart apache2 after the modifications:
    - `sudo service apache2 restart`

Apache Webserver on Ubuntu

- `sudo nano /etc/apache2/apache2.conf`



I put the declarations here
(*I'm not sure it's the best place though…*)

Apache Webserver on Ubuntu

- `sudo nano /etc/apache2/mods-available/userdir.conf`

```
<IfModule mod_userdir.c>
        UserDir public_html
        UserDir disabled root

        <Directory /home/*/public_html>
                AllowOverride FileInfo AuthConfig Limit Indexes
                Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
                <Limit GET POST OPTIONS>
                        Require all granted
                </Limit>
                <LimitExcept GET POST OPTIONS>
                        Require all denied
                </LimitExcept>
        </Directory>
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

not sure this exactly is necessary, but it works!

Apache Webserver on Ubuntu

- ~*sandiway*/public_html/test.cgi

A worked example: `CMUDict`

- Suppose:
  - we want to run the *CMU Pronouncing Dictionary* (CMUDict) – *see next slide*
  - assume it's already installed in Perl (*a programming language*)
- **How?**
  1. assume script is executable (`chmod 755`)
  2. 1st line of script, e.g. `#!/usr/bin/perl` tells the shell what program to use
  3. Then remaining lines of the script will be Perl code
  4. `which perl` gives the absolute PATH for Perl as, e.g., `/usr/bin/perl`
  - *same for Python and any other programming languages etc*.
- **Debugging:**
  - `tail /var/log/apache2/error_log`
  - `tail /var/log/apache2/access_log`

A worked example: `CMUDict`

- `cmudict.cgi` on course website

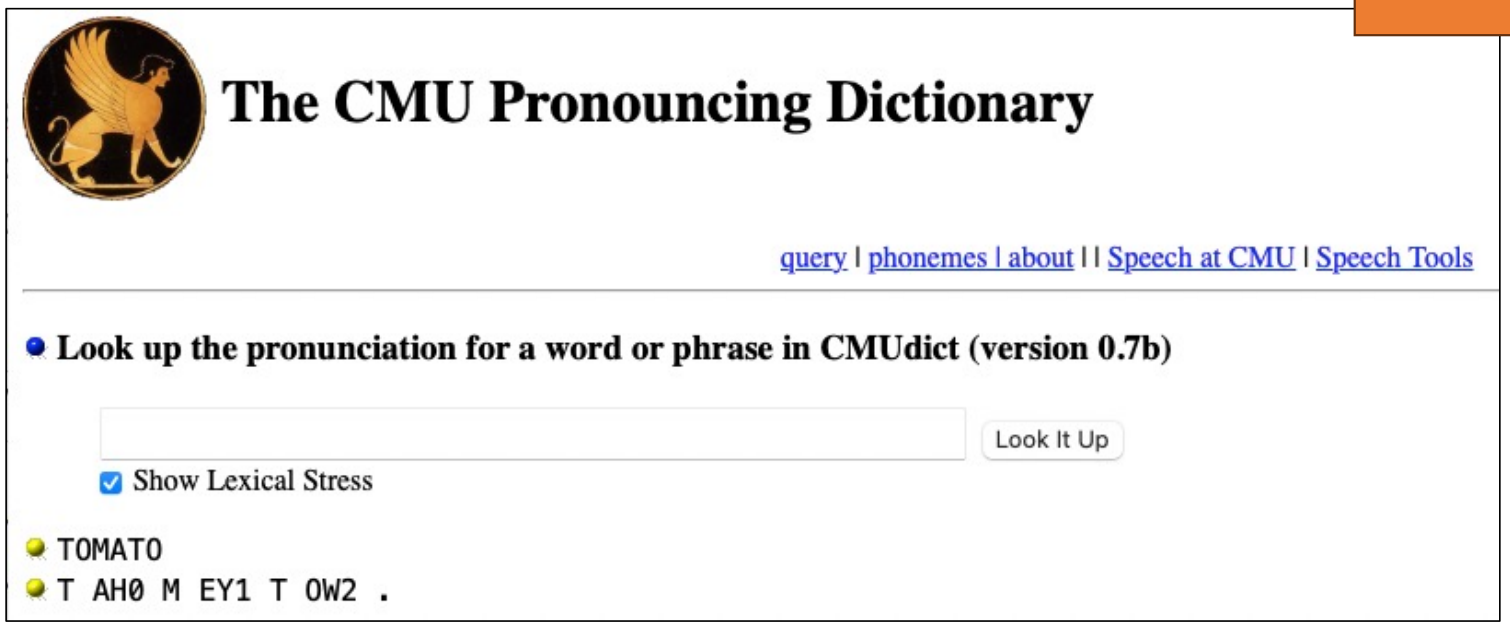
```
 1#!/usr/bin/perl¶
 2use Lingua::EN::CMUDict;¶
 3my $obj = new Lingua::EN::CMUDict;¶
 4my $string = $ENV{QUERY_STRING};¶
 5my $word =  $string ? $string : $ARGV[0]; # remove word=¶
 6my $n = $obj->number_of_syllables($word);¶
 7print "Content-type: text/html; charset=utf-8\n\n";¶
 8print '<html><head><style>div {font-size: x-large}</style></head>';¶
 9print '<body><h1>Using CMUDict from Perl</h1><div>';¶
10if ($n) {¶
11  print "<em>$word</em> has $n syllable(s).</div>";¶
12  my $pron = $obj->get_word($word);¶
13  print "$pron<br>";¶
14} else {¶
15  print "<em>$word</em> not in cmudict</div>";¶
16}¶
17print 'Arpabet: <a href="https://en.wikipedia.org/wiki/ARPABET#Symbols">symbol table</a>';¶
18print "</body></html>\n"¶
```

This is Perl code, not Bash!

A worked example: `CMUDict`

- http://www.speech.cs.cmu.edu/cgi-bin/cmudict

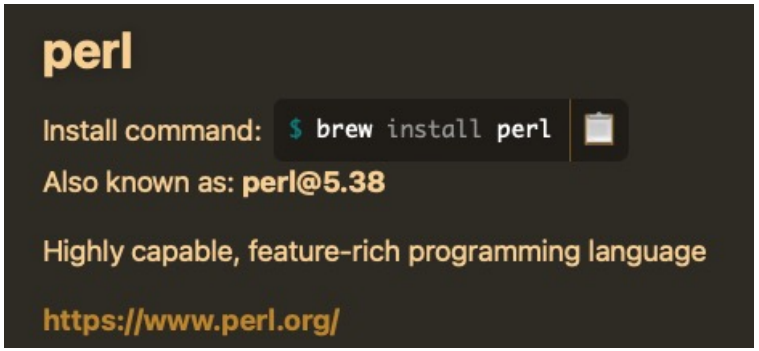
Notice it's a cgi-bin!



A worked example: `CMUDict`

- You must have Perl installed:
  - macOS: install homebrew from http://brew.sh
  - then `brew install perl`
- Installing CMU Dictionary in Perl:
  - `which cpan`
  - `/opt/homebrew/bin/cpan`

  cpan is the Perl library installer
  - `sudo cpan Lingua::EN::CMUDict`



A worked example: `CMUDict`

- Windows 11 PowerShell:
  - `wsl` to start Ubuntu

```
$ which perl
/usr/bin/perl
$ which cpan
/usr/bin/cpan
$ sudo cpan Lingua::EN::CMUDict
$ cd ~
$ mkdir public_html
$ cd public_html
$ cp /mnt/c/Users/username/cmudict.cgi .
(check permissions for read and execute)
```

A worked example: `CMUDict`

```
#!/opt/homebrew/bin/perl
use Lingua::EN::CMUDict;
my $obj = new Lingua::EN::CMUDict;
my $string = $ENV{QUERY_STRING};
my $word =  $string ? $string : $ARGV[0];
my $n = $obj->number_of_syllables($word);
print "Content-type: text/html; charset=utf-8\n\n";
print '<html><head><style>div {font-size: x-large}</style></head>';
print '<body><h1>Using CMUDict from Perl</h1><div>';
if ($n) {
  print "<em>$word</em> has $n syllable(s).</div>";
  my $pron = $obj->get_word($word);
  print "$pron<br>";
} else {
  print "<em>$word</em> not in cmudict</div>";
}
print 'Arpabet: <a href="https://en.wikipedia.org/wiki/ARPABET#Symbols">symbol table</a>';
print "</body></html>\n"
```

usually /usr/bin/perl or /opt/local/bin/perl

word is in $QUERY_STRING
or on command line $ARGV[0]

text/html followed by a *blank line*

A worked example: `CMUDict`

- Command line usage (*assuming I'm in directory* ~/Sites *or in* ~/public_html):

```
./cmudict.cgi tomato
Content-type: text/html; charset=utf-8

<html><head><style>div {font-size: x-large}</style></head><body><h1>Using
CMUDict from Perl</h1><div><em>tomato</em> has 3 syllable(s).</div>T AH0 - M EY1
- T OW2<br>Arpabet: <a
href="https://en.wikipedia.org/wiki/ARPABET#Symbols">sym
table</a></body></html>
```

I added ?tomato in manually
(*cheat for now*)
define a <form> next time!

A worked example: `CMUDict`

Symbols [edit]

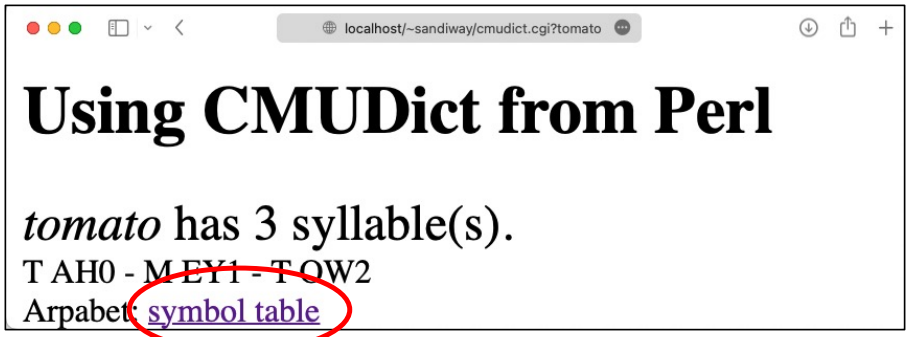
Stress is indicated by a digit immediately following a vowel. Auxiliary symbols are identical in 1- and 2-letter codes. In 2-letter notation, segments are separated by a space.

Vowels[2]

ARPABET		IPA	Example(s)
1-letter	2-letter		
a	AA	ɑ~ɒ	balm, bot (with father–bother merger)
@	AE	æ	bat
A	AH	ʌ	butt
c	AO	ɔ	caught, story
W	AW	aʊ	bout
x	AX	ə	comma
—	AXR[3]	ɚ	letter, forward
Y	AY	aɪ	bite
E	EH	ɛ	bet
R	ER	ɝ	bird, foreword
e	EY	eɪ	bait
I	IH	ɪ	bit
X	IX	ɨ	roses, rabbit
i	IY	i	beat
o	OW	oʊ	boat
O	OY	ɔɪ	boy
U	UH	ʊ	book
u	UW	u	boot
—	UX[3]	ʉ	dude

Consonants[2]

ARPABET		IPA	Example
1-letter	2-letter		
b	B	b	buy
C	CH	tʃ	China
d	D	d	die
D	DH	ð	thy
F	DX	ɾ	butter
L	EL	l̩	bottle
M	EM	m̩	rhythm
N	EN	n̩	button
f	F	f	fight
g	G	g	guy
h	HH *or* H[3]	h	high
J	JH	dʒ	jive
k	K	k	kite
l	L	l	lie
m	M	m	my
n	N	n	nigh
G	NX *or* NG[3]	ŋ	sing
—	NX[3]	ɾ̃	winner
p	P	p	pie



A worked example: `CMUDict`

https://metacpan.org/pod/Lingua::EN::CMUDict

Lingua::EN::CMUDict - Perl extension for utilizing the CMU dictionary file

SYNOPSIS

```
use Lingua::EN::CMUDict;
my $obj = new Lingua::EN::CMUDict;
print $obj->number_of_syllables("test");
```

DESCRIPTION

This version of the CMU Pronouncing dictionary was generated from the original dictionary and designed to syllabify it. The paper *On the Syllabification of Phonemes* by Susan Bartlett, Grzegorz Kondrak and Colin Cherry (NAACL-HLT 2009) covers the methods used to generate the dictionary.

EXPORT

None by default.

METHODS

new(cmudict=>*file*)

Creates a new object, populating it with the cmusyldict db file. If the cmudict argument is passed with a filename as the argument, that file is used. If you do not use that argument, the default cmusyldict db file installed with the module is used.

rhymes(word)

In the case of an array being returned, returns all rhymes to the given word. In a scalar context, returns a single rhyme.

number_of_syllables(word)

Returns the number of syllables in the word. Many pluralities do not add syllable counts and are therefore not in the original database. This code tries to be intelligent by looking for those and returning the number of syllables. Also, if a sentence is passed in, returns the number of syllables in the sentence. Doesn't currently deal with punctuation very well.

get_word(word)

Returns the pronunciation for the word with syllable boundaries.

SEE ALSO

Lingua::EN::Phoneme -- another way of accessing the CMU Pronunciation dictionary.

AUTHOR

Leigh Metcalf, <leigh@fprime.net>

COPYRIGHT AND LICENSE

A worked example: `CMUDict`

- Actual entry in CMUDict looks like this:

```
1. TOMATO T AH0 M EY1 T OW2
2. TOMATO(1) T AH0 M AA1 T OW2
3. TOMATOE T AH0 M EY1 T OW0
4. TOMATOE(1) T AH0 M AA1 T OW0
5. TOMATOES T AH0 M EY1 T OW0 Z
6. TOMATOES(1) T AH0 M AA1 T OW0 Z
7. TOMATOS T AH0 M EY1 T OW2 Z
8. TOMATOS(1) T AH0 M AA1 T OW2 Z
```

A shame the code doesn't retrieve all available pronounciations!