

Python Tutorial

Day 5: The Web

Today

- a very brief overview of CGI
- HTML production
- basic CGI libraries
- William's quick-and-dirty stateful form generator

Proviso, Disclaimer, etc.

- for real web programming, **pick an existing web development framework**
- a little extra work to learn
- but will have everything you want — easy database integration, session management, user-friendly state persistence, probably page templating, &c.

CGI

- simply a way to run a program on a server
- whatever that program prints is sent back to browser
- data sent to CGI program from an HTML form block
- that data gets to the program via an environment variable
- good CGI libraries make it trivial to get at form data

Producing HTML

- there is no standard library for this
- each program or framework has its own templating language
- usually template systems allow embedded code
- we'll stick with simple HTML generation by scratch
- don't write giant scary HTML libraries — hide the details in CSS

CGI & HTML

- CGI scripts must print out a header, so that the server and browser both know how to interpret the output
- cookies complicate matters, but for now, use this:

```
print "Content-Type: text/html"    # HTML is following
print                               # blank line, end of headers

# Next, proper HTML:
print "<!DOCTYPE html PUBLIC ...."
```

CGI Paradigm Decision

- there are two basic ways to have a CGI script operate
- the form may call the CGI from a separate HTML web page
- or the CGI script itself may do both — print the form and process the form — by keeping track of whether the fields are full (process) or empty (print form)

CGI from HTML

- here's the HTML:

```
<form method="POST"
      enctype="multipart/form-data"
      action="http://system.biostat.wisc.edu/cgi-bin/pytut/adder">
Enter some numbers to add: <br />
<i>X</i> <input type="text" name="X"> <br />
<i>Y</i> <input type="text" name="Y"> <br />
<input type="submit">
</form>
```

```
import cgi
# .... print headers
form = cgi.FieldStorage()
if not (form.has_key("X") and form.has_key("Y")):
    print "<h1>Error</h1>"
    print "<p>Please fill out both fields.</p>"
else:
    try:
        X = float(form["X"].value) # all CGI values strings
        Y = float(form["Y"].value)
        print "<p>The answer is %s.</p>" % (X + Y)
    except ValueError:
        print "<h1>Error</h1>"
        print "<p>Both fields must be numbers.</p>"
# .... print footers
```

Debugging note

- once the header is printed, program errors are silent to the browser
- so, you may need to check the `httpd` logs to discover the error
- you can use the `cgitb` (CGI Trace-Back) library for development
- turn OFF when the program goes live

```
import cgibt  
cgitb.enable()
```

William's CGI Library

- similar to other web development libraries, only much simpler
- similar in design to some, but not most, libraries
- designed to run as a CGI from the start (i.e., the same program prints the form as well as processes it)
- used for several BCG CGIs

- again, for big, complex apps, pick an existing framework
- but it is good for quick, basic forms
- at the moment, it has no idea about CSS; I should probably fix that

wsalib

- you build up a form with different input components
- the component classes are assembled into a list, and sent to a form class, `CGIForm`
- `CGIForm` knows if a form has been filled in yet (via hidden input flags)
- have a single if/else block, one just prints the form, one processes it
- the `CGI` class holds trivial HTML formatting

```

from wsalib import *

v_x = CGITextInput("X", text="X value:")
v_y = CGITextInput("Y", text="Y value:")
v_sub = CGISubmit("submit")

cgi = CGI(title="The Incredible Adding Machine Mark II")
form = CGIForm([v_x, v_y, v_sub])

cgi.header()
if form.filled_in():
    try:
        X = float(form.data['X'].value)
        Y = float(form.data['Y'].value)
        print "<p>The answer is %s.</p>" % (X + Y)
    except ValueError:
        print "<h1>Error</h1>"
        print "<p>Both fields must be numbers.</p>"
else:
    print form
cgi.footer()

```

Other Components

- `CGIText("html here")` - just spits out HTML, for mixing text in with the form
- `CGISelect` - drop-down menu of options

```
v_files = CGISelect('file_instructions',  
    text=""Should we delete the files, or get more  
    information from you first?""",  
    options=('delete', 'contact me'))
```

- `CGISubmit("Submit")` - submit button
- `CGISubmitAndReset("Submit")` - submit and reset buttons
- `CGIPasswdInput(fieldname)` - text field in which all typing shows up as stars
- `CGITextArea(fieldname)` - multi-line input
- The entry field classes all have a default method which sets a pre-filled default

CGIForm methods

- `empty_required_fields(['name', 'email'])` - returns list of named fields missing data
- `marked_fields(missing, marker="*** ")` - prints out the form with fields named in `missing` marked with the marker string

```
if form.filled_in():
    missing = form.empty_required_fields(['user', 'sender', 'data'])
    if missing:
        print form.marked_fields(missing, marker="***")
        print "<b>You must fill out the fields marked with the stars.<b>"
```

CGIForm methods

- `mail(from_addr, to_addr_list, subject)`

```
if form.filled_in():
    missing = form.empty_required_fields(['requester_email', 'login_name',
                                         'disable_date', 'delete_date'])
    if missing:
        print "You must fill out all fields."
        print form.marked_fields(missing)
    else:
        fromaddr = form.data["requester_email"].value
        goneuser = form.data["login_name"].value
        mailok = form.mail('system@biostat.wisc.edu',
                          [goneuser, 'sysreq@biostat.wisc.edu'],
                          "Account deletion request: %s" % goneuser)
        if mailok:
            print "Thanks for filling out this form. We will send you email"
            print "when the account has been deactivated."
else:
    print form
```

The End

- I've covered all but the most exotic Python
- when I wrote `wsalib` Python hadn't yet become a giant web language
- **find libraries at python.org** - let me know what you want installed
- I'm happy to answer Python questions, but for big libraries please do read over the docs first