

Introduction to CGI Programming

1 Python Scripts in Browsers

- Apache and LAMP
- dynamic web pages

2 Internet Basics

- HTTP: `GET` and `POST` methods
- `FORM` to accept user data
- server-side scripting

3 Interactive Web Pages

- accepting user input
- greatest common divisor on the web

MCS 275 Lecture 21
Programming Tools and File Management
Jan Verschelde, 27 February 2017

Introduction to CGI Programming

1 Python Scripts in Browsers

- Apache and LAMP
- dynamic web pages

2 Internet Basics

- HTTP: GET and POST methods
- FORM to accept user data
- server-side scripting

3 Interactive Web Pages

- accepting user input
- greatest common divisor on the web

the Web Server Apache

the A in LAMP

The combination of web server, scripting language, and database is often referred to as the LAMP system.

LAMP stands for

L is Linux, the operating system;

A is Apache, the web server;

M is MySQL, the database;

P is Python, the scripting language.

Observe that all four are open source software.

Apache makes a cute pun on “a patchy web server”, but its name is in honor of the Native American Apache tribe.

Its web site is at `http://www.apache.org`.

Running Apache

on a Mac OS X

Apache is platform independent.
We will demonstrate on a Mac OS X.

- 1 Apache is already installed on Mac OS X, launch Safari with `http://localhost/` to verify.
- 2 To enable web sharing, select Sharing from the System Preferences.
- 3 Instead of `public_html`, the `Sites` directory is where Mac users store their web pages.
- 4 Instead of `/var/www/cgi-bin`, CGI scripts are in `/Library/WebServer/CGI-Executables`.
CGI = Common Gateway Interface, is set of protocols through which applications interact with web servers.

Using `localhost` we remain working offline.

more instructions are needed since Yosemite

To check the version of Apache:

```
$ httpd -v
Server version: Apache/2.4.23 (Unix)
Server built:   Aug  8 2016 16:31:34
$
```

To launch Apache, type `sudo /usr/sbin/apachectl graceful` in a Terminal window.

Pointing the browser to `localhost` (or `127.0.0.1`) shows It works!.

To check the configuration of Apache, type

```
apachectl configtest
```

at the command prompt.

serving web pages from user directories

To serve web pages from user directories, in the file

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

uncomment the line that contains

```
Include /private/etc/apache2/users/*.conf
```

In `/etc/apache2/httpd.conf` **uncomment the lines**

```
LoadModule userdir_module libexec/apache2/mod_userdir.so
# User home directories
Include /private/etc/apache2/extra/httpd-userdir.conf
```

and also the line

```
LoadModule cgi_module libexec/apache2/mod_cgi.so
```

configuration of users

In the folder `/etc/apache2/users`,
if `<user>` is your user name, in the file `<user>.conf`,
then add the line

```
Require all granted
```

inside the block

```
<Directory "/Users/<user>/Sites/">  
  Options Indexes MultiViews  
  AllowOverride None  
  Require all granted  
</Directory>
```

Running the Web Server Apache

on Windows ...

Apache is platform independent.

On a Windows 2000 Dell Latitude Laptop:

- 1 Download binaries of Apache webserver from `http://www.apache.org`.
- 2 Login as Administrator and install Apache binary.
- 3 Connect to `http://127.0.0.1:8080/`
- 4 **html files** are in `c:/Program Files/Apache " + "Software Foundation/Apache2.2/htdocs"`
- 5 **scripts** are in `"c:/Program Files/Apache " + "Software Foundation/Apache2.2/cgi-bin"`

Important: `localhost = 127.0.0.1:8080`.

Introduction to CGI Programming

1 Python Scripts in Browsers

- Apache and LAMP
- dynamic web pages

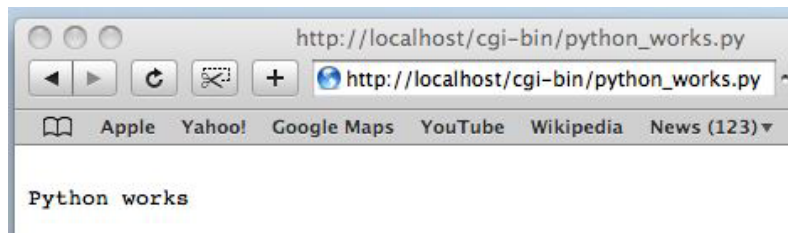
2 Internet Basics

- HTTP: GET and POST methods
- FORM to accept user data
- server-side scripting

3 Interactive Web Pages

- accepting user input
- greatest common divisor on the web

Python Works! – our first CGI script

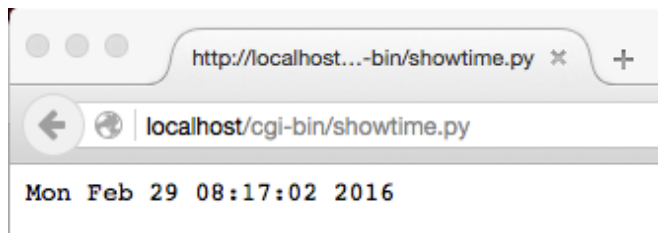


The Python script `python_works.py`:

```
#!/usr/bin/python
"""
Place this in /Library/WebServer/CGI-Executables.
"""
print("Content-Type: text/plain\n\n")
print("Python works")
```

The first line is the location of the Python interpreter.

dynamic web pages



```
#!/usr/bin/python
"""
Displays the current time in a browser window.
"""
import time
print("Content-Type: text/plain\n")
print(time.ctime(time.time()))
```

The time gets updated every time the page refreshes.

Introduction to CGI Programming

1 Python Scripts in Browsers

- Apache and LAMP
- dynamic web pages

2 Internet Basics

- **HTTP: GET and POST methods**
- FORM to accept user data
- server-side scripting

3 Interactive Web Pages

- accepting user input
- greatest common divisor on the web

Dynamic Web Pages

HTTP: GET and POST methods

HTTP (HyperText Transfer Protocol) determines request-response communication between web browser and web server.

Methods of HTTP:

GET method is a request for a static resource, such as an HTML page.

Just typing the URL of the requested web page invokes the **GET** method.

POST method is a request for a dynamic resource, with input parameters of the request contained within the body of the request.

The **GET** and **POST** methods are most commonly used.

Elements of HTML

HyperText Markup Language

Commonly used elements in HTML documents:

HTML `<HTML>` marks start of HTML document,
and `</HTML>` marks the end.

HEAD specifies header information of a document.

TITLE specifies title of the document.

BODY contains the body text of the document.

FONT used to alter font size and color of text.

H1 to display headings of type 1,
other heading elements are **H2** and **H3**.

P defines a paragraph.

OL ordered list, unordered list is **UL**.

LI list element in ordered or unordered list.

Learn **HTML** by looking at source of web pages.

Introduction to CGI Programming

1 Python Scripts in Browsers

- Apache and LAMP
- dynamic web pages

2 Internet Basics

- HTTP: GET and POST methods
- **FORM to accept user data**
- server-side scripting

3 Interactive Web Pages

- accepting user input
- greatest common divisor on the web

Accepting Data from Users

To accept data from users,
three elements are generally used:

`FORM` contains all the code related to a form.

Its purpose is to accept user input in a systematic and structured manner.

`INPUT` specifies the code used to create the form controls that accept user input.

`SELECT` used to display lists in a form.

Designing and creating interactive web pages
is similar to GUI design.

the FORM element

A form is a collection of text boxes, radio buttons, check boxes, and buttons.

Two attributes of a form:

`METHOD` is GET or POST.

`ACTION` is typically used to specify the code that will process the the input data.

Syntax:

```
<FORM METHOD="GET_or_POST" ACTION="file_name">  
    code_of_the_form  
</FORM>
```

the `INPUT` element

The `INPUT` element is specified inside a `FORM` element.

The `INPUT` elements consists of controls, such as text boxes, buttons, radio buttons, and check boxes.

Each of these controls can have attributes:

`TYPE` specifies type of control to accept user input.

`NAME` specifies name of a control, for identification.

`VALUE` holds value entered by user, or default.

Five types of control:

- (1) submit button;
- (2) text boxes;
- (3) radio buttons;
- (4) check boxes;
- (5) combo boxes.

Introduction to CGI Programming

1 Python Scripts in Browsers

- Apache and LAMP
- dynamic web pages

2 Internet Basics

- HTTP: GET and POST methods
- FORM to accept user data
- **server-side scripting**

3 Interactive Web Pages

- accepting user input
- greatest common divisor on the web

Client-Side / Server-Side scripting

We distinguish between

- Client-Side Scripting: processed by browser
advantage: saves time of the server
- Server-Side Scripting: processed by server
needed for synchronization, modification of data
server time is then *the* time

Python is a powerful server-side scripting language.
The `cgi` module has to be imported,
in order to communicate the data from client.

Introduction to CGI Programming

1 Python Scripts in Browsers

- Apache and LAMP
- dynamic web pages

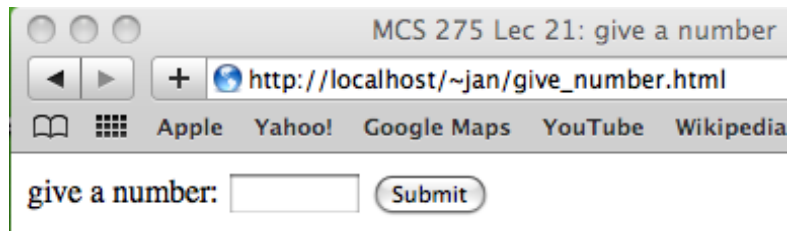
2 Internet Basics

- HTTP: GET and POST methods
- FORM to accept user data
- server-side scripting

3 Interactive Web Pages

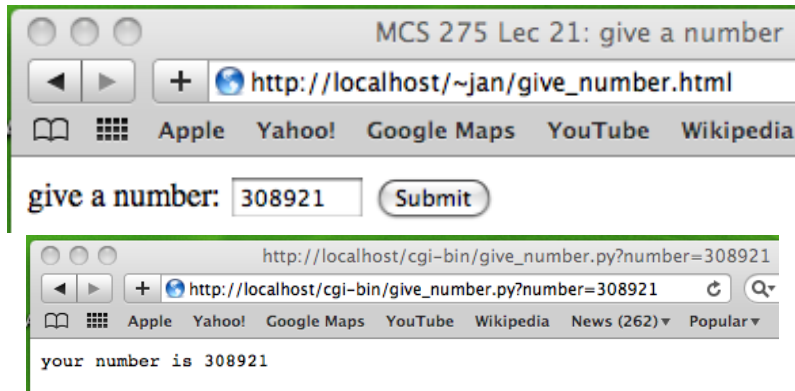
- accepting user input
- greatest common divisor on the web

Prompting for a Number



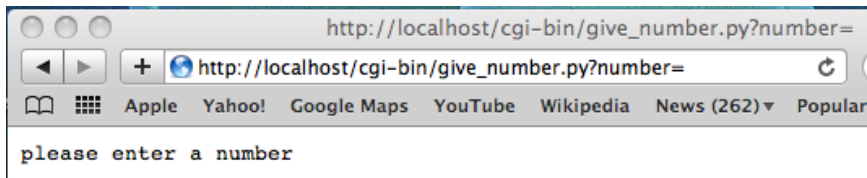
- 1 The displayed web page uses a `form` element.
- 2 The `form` contains two `input` elements
- 3 After the user hits `submit`, a Python script will run.

the User enters a Number



Observe the URL: it contains the data.

the User enters Nothing



The user receives an error message.

Form and Input Elements

```
<HTML>
<HEAD>
<TITLE> MCS 275 Lec 21: give a number </TITLE>
</HEAD>
<BODY>

<FORM action="http://localhost/cgi-bin/give_number.py">

give a number:

<INPUT type="text" name="number" size = "8">

<INPUT type="submit">

</FORM>
</BODY>
</HTML>
```

the action of python

```
#!/usr/bin/python
"""
Accepts a number from a form.
"""
import cgi
form = cgi.FieldStorage()
print("Content-Type: text/plain\n")
try:
    n = form['number'].value
    print("your number is " + n)
except KeyError:
    print("please enter a number")
```

Introduction to CGI Programming

1 Python Scripts in Browsers

- Apache and LAMP
- dynamic web pages

2 Internet Basics

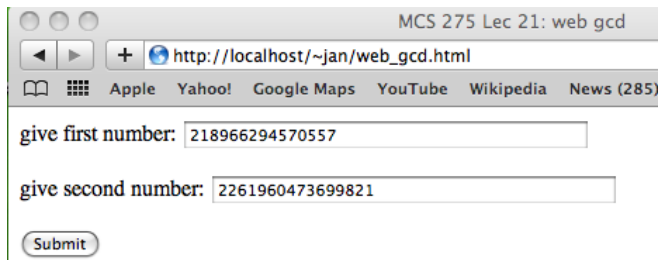
- HTTP: GET and POST methods
- FORM to accept user data
- server-side scripting

3 Interactive Web Pages

- accepting user input
- greatest common divisor on the web

GCD on the Web

web computing of the greatest common divisor



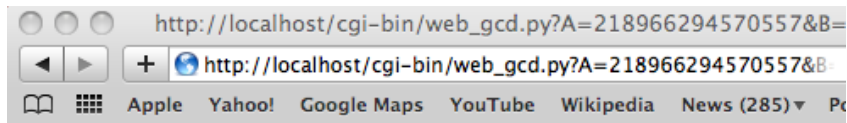
A screenshot of a web browser window. The title bar reads "MCS 275 Lec 21: web gcd". The address bar shows "http://localhost/~jan/web_gcd.html". The browser's search bar contains "Apple", "Yahoo!", "Google Maps", "YouTube", "Wikipedia", and "News (285)". The main content area features two input fields: "give first number:" with the value "218966294570557" and "give second number:" with the value "2261960473699821". Below the input fields is a "Submit" button.

The html code extends naturally.

When the user submits the numbers, their greatest common divisor will be computed and displayed.

Output of Web GCD

Euclid does web computing



```
your first number is 218966294570557
your second number is 2261960473699821
gcd(218966294570557,2261960473699821) = 59228567
```

Which computer does the computation?

The HTML Code

```
<HTML>
<HEAD>
<TITLE> MCS 275 Lec 21: web gcd </TITLE>
</HEAD>
<BODY>

<FORM action="http://localhost/cgi-bin/web_gcd.py">
<P> give first number:
<input type="text" name="A" size = "40"> </P>
<P>give second number:
<input type="text" name="B" size = "40"> </P>
<P> <input type="submit"> </P>
</FORM>

</BODY>
</HTML>
```

The Python Script

```
#!/usr/bin/python
"""
Script to compute the greatest common divisor
of two numbers entered via a form.
"""
import cgi
form = cgi.FieldStorage()
print("Content-Type: text/plain\n")
try:
    x = form['A'].value
    print("your first number is " + x)
except KeyError:
    print("please enter a first number")
try:
    y = form['B'].value
    print("your second number is " + y)
except KeyError:
    print("please enter a second number")
```

The Python Script continued

```
def gcd(alpha, beta):
    """
    Returns the greatest common divisor
    of the numbers alpha and beta.
    """
    rest = alpha % beta
    if rest == 0:
        return beta
    else:
        return gcd(beta, rest)

ix = int(x)
iy = int(y)
print("gcd(%d,%d) = %d" % (ix, iy, gcd(ix, iy)))
```


Summary + Assignments

Exercises:

- 1 Make your own web page, using `people.uic.edu`.
- 2 Verify if Apache is installed on your computer.
Install Apache if necessary.
- 3 Design a web interface to convert pounds into kilograms and kilograms into pounds.
- 4 Take the script `facnum.py` of Lecture 1 and write a web interface for it.