Server-Side Web Programming: CGI (Part 1)

Robert M. Dondero, Ph.D.
Princeton University

Objectives

- We will cover…
  - Common Gateway Interface (CGI) programming
  - The HTTP GET method
  - The HTTP POST method

Agenda

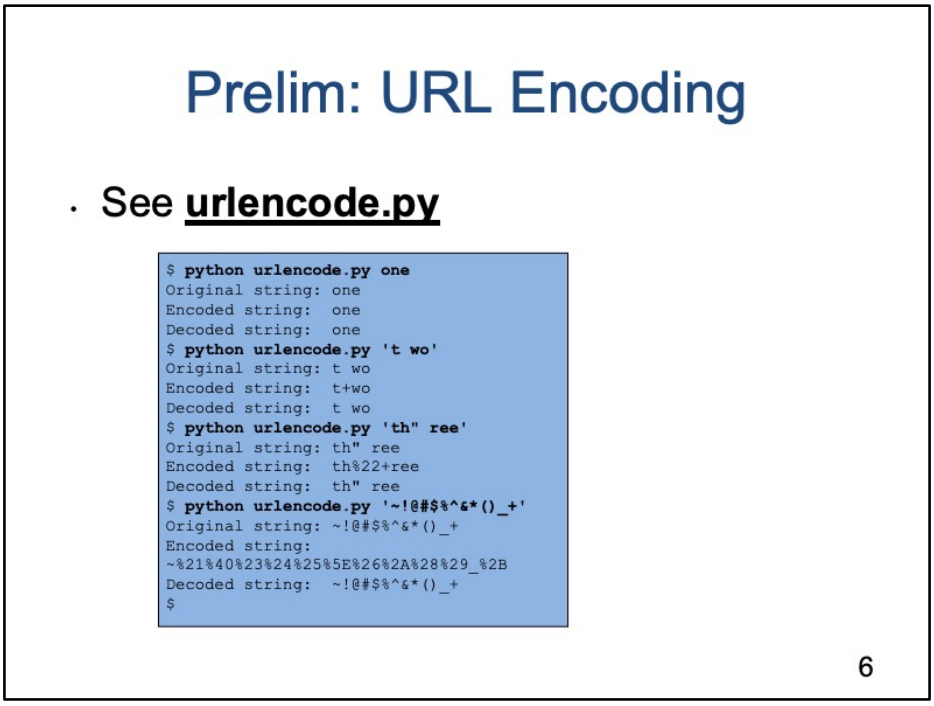
- **Preliminary: URL encoding**
- CGI programming
- CGI using the HTTP GET method
- CGI using the HTTP POST method

Prelim: URL Encoding

- ***URL encoding***
  - A way of encoding a string such that it may appear within a URL
  - Each space is encoded as +
  - Each special character (", ', =, &, etc.) is encoded as:
    - `%nn` where `nn` are hex digits as per UTF-8, or
    - `%nn%nn`, or
    - `%nn%nn%nn`, or
    - `%nn%nn%nn%nn`

Prelim: URL Encoding

- Python support for URL encoding
  - In urllib.parse
    - `quote_plus()`
      - Converts to URL encoding
    - `unquote_plus()`
      - Converts from URL encoding



Aside: URL Encoding

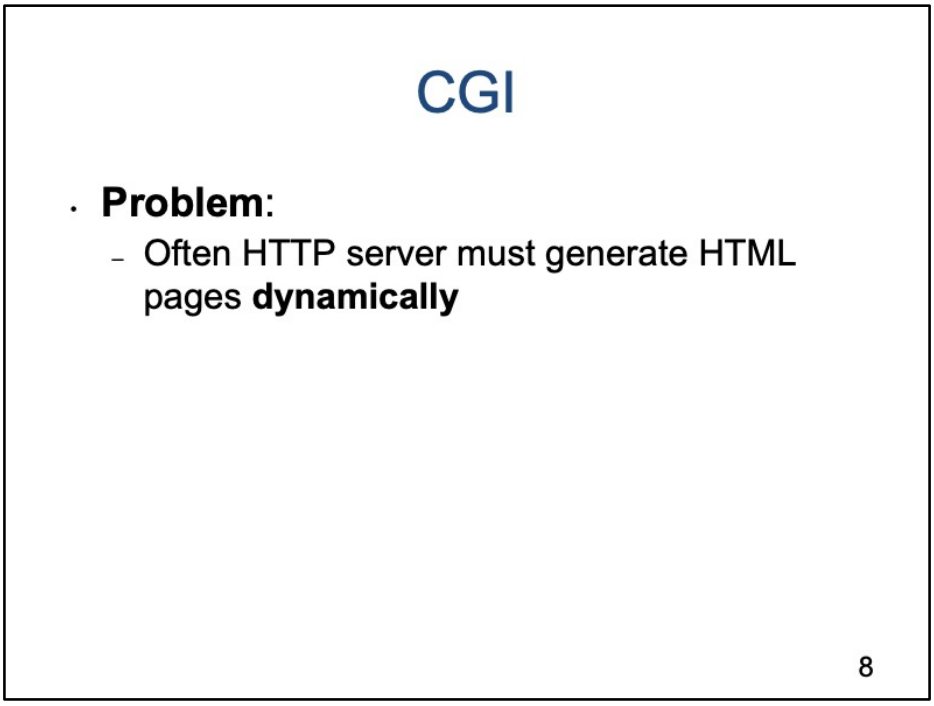
[see slide]

Code notes:

Accepts original_str as a command-line arg
URL-encodes it to yield encoded_str
URL-decodes it to yield decoded_str, which should be the same as original_str
Writes all three to stdout

Agenda

- Preliminary: URL encoding
- **CGI**
- CGI using the HTTP GET method
- CGI using the HTTP POST method



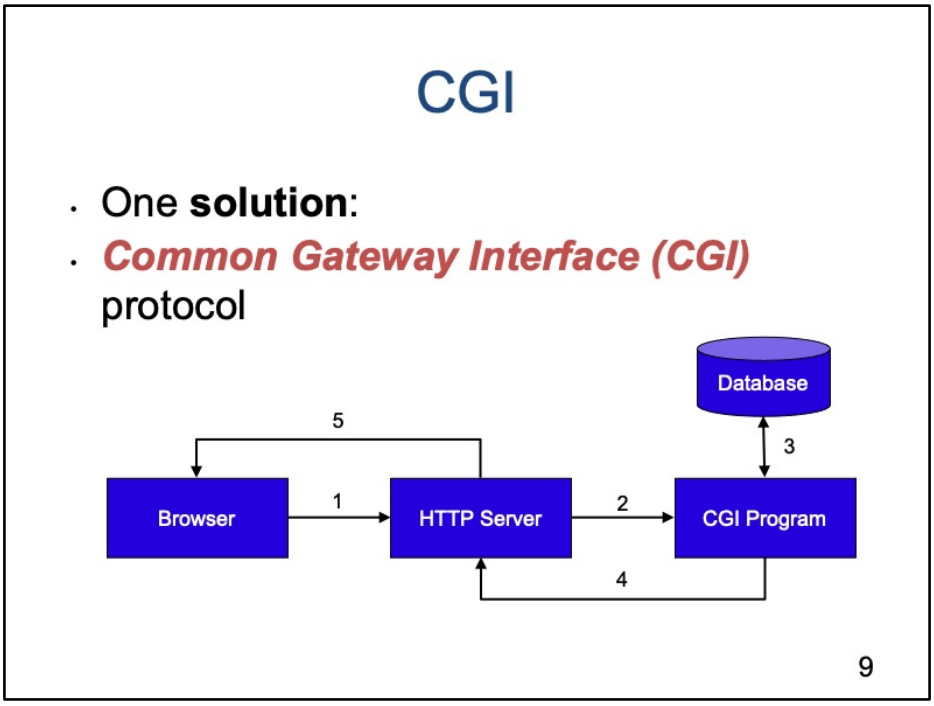
CGI

Problem

Often it's insufficient for the HTTP server to deliver **static** HTML pages

Often the HTTP server must generate HTML pages **dynamically**

Based upon arguments supplied by client



CGI

[see slide]

CGI is a 5 step process:
(1) Browser sends HTTP request to HTTP Server
(2) HTTP server doesn't fetch a file; instead it executes a program
(3) CGI program (often) accesses data in a database; generates a HTTP response
(4) CGI program sends HTTP response to HTTP Server
(5) HTTP server sends HTTP response to Browser

CGI

- HTTP methods
  - HTTP defines multiple “methods”
  - The two most common:
    - GET
    - POST

Agenda

- Preliminary: URL encoding
- CGI programming
- **CGI using the HTTP GET method**
- CGI using the HTTP POST method

CGI Using GET

- **Question**: How does user command browser to communicate with HTTP server using CGI with the **GET** method?

CGI Using GET

- **Answer 1**: Enter a URL at top of browser

```
http://host:port/somepgm.py?name1=value1&name2=value2
```

The default protocol is http
There is no default host
The default port is 80

CGI Using GET

- **Answer 2**: Click on a page link

```
<a href="http://host:port/somepgm.py?name1=value1&name2=value2>
   ...
</a>
```

The default protocol is http
The default host is the host that delivered **this** page
The default port is the port from which **this** page was delivered

CGI Using GET

- **Answer 3**: Submit a form

User enters value1 and value2 in input elements

```
<form action="http://host:port/somepgm.py" method="get">
   <input type="text" name="name1" value="init1">
   <input type="text" name="name2" value="init2">
   <input type="submit">
</form>
```

The default protocol is http
The default host is the host that delivered **this** page
The default port is the port from which **this** page was delivered

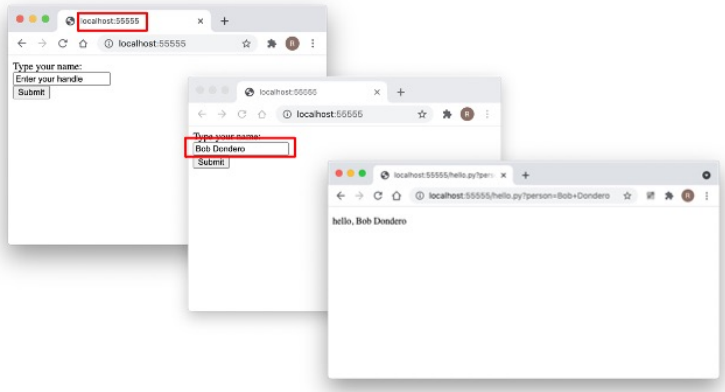
CGI Using GET

- See **HelloPythonGet** app

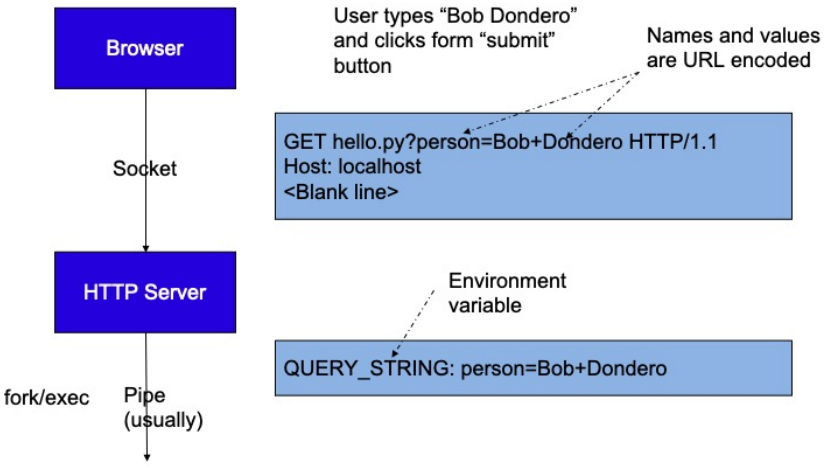
```
$ python runserver.py 55555
Serving HTTP on 0.0.0.0 port 55555 (http://0.0.0.0:55555/) ...
```

CGI Using GET

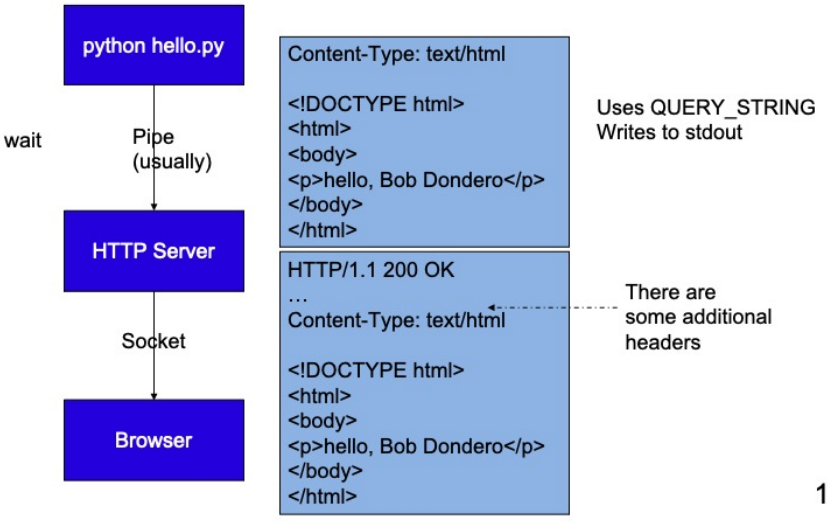
- See **HelloPythonGet** app

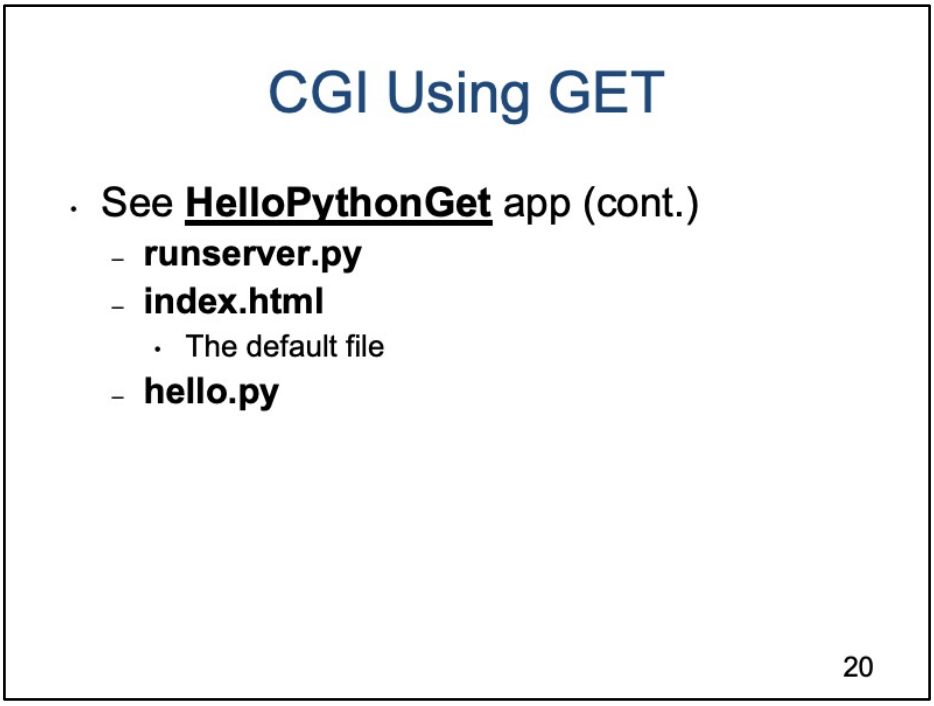


CGI Using GET



CGI Using GET





CGI Using GET

[see slide]

Code notes: runserver.py

Launches simplehttpserver.py with the --cgi argument

When simplehttpserver.py sees a request for a .py file…

It interprets that request as a CGI request

I'll create a runserver.py file for every web app

So I (and you) need not remember how to launch the HTTP server

Code notes: index.html

Defines a form

Submitting the form will send a GET request

simplehttpserver.py considers index.html to be the default file

If HTTP request doesn't name a file, the file defaults to index.html

Code notes: hello.py

Uses Python environ variable (a dict object) to get the value of the QUERY_STRING environment variable

Fetches name/value pairs from QUERY_STRING environment variable

Composes HTTP response
Writes it to stdout

Agenda

- Preliminary: URL encoding
- CGI
- CGI using the HTTP GET method
- **CGI using the HTTP POST method**

CGI Using POST

- **Question**: How does user command browser to communicate with HTTP server using CGI with the **POST** method?

CGI Using POST

- **Answer**: Submit a form

User enters value1 and value2 in input elements

```
<form action="http://host:port/somepgm.py" method="post">
   <input type="text" name="name1" value="init1">
   <input type="text" name="name2" value="init2">
   <input type="submit">
</form>
```

The default protocol is http
The default host is the host that delivered **this** page
The default port is the port from which **this** page was delivered

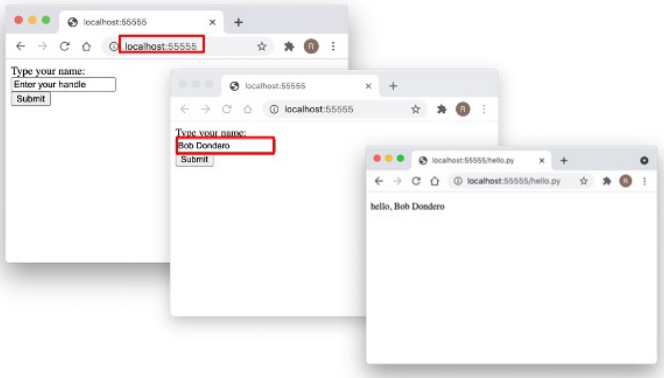
CGI Using POST

- See **HelloPythonPost** app

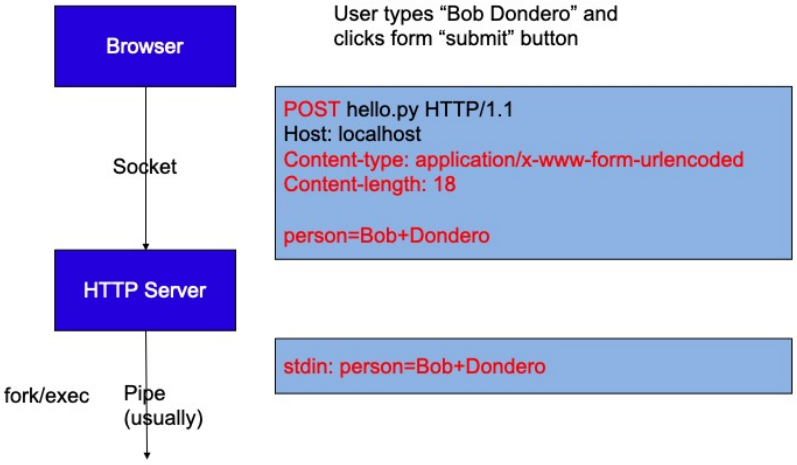
```
$ python runserver.py 55555
Serving HTTP on 0.0.0.0 port 55555 (http://0.0.0.0:55555/) ...
```

CGI Using POST

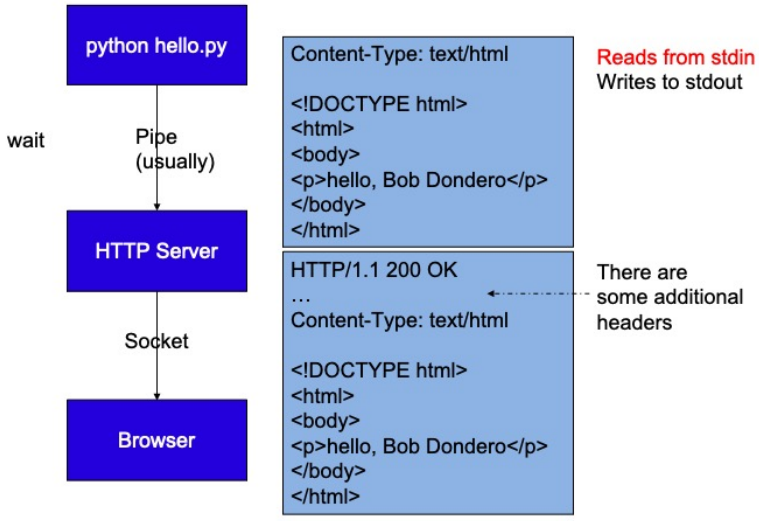
- See **HelloPythonPost** app

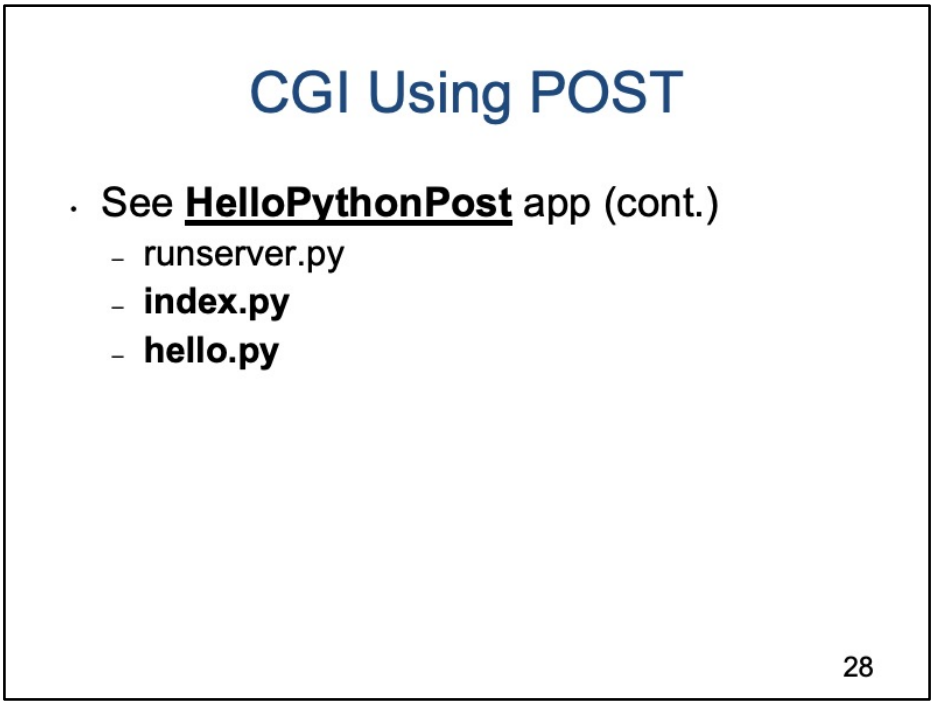


CGI Using POST



CGI Using POST





CGI Using POST

[see slide]

Code notes: index.py
Form will submit a POST request

Code notes: hello.py
Fetches name=value pairs from its stdin, not from an env var

Summary

- We have covered:
  - Common Gateway Interface (CGI) programming
  - The HTTP GET method
  - The HTTP POST method