COMP 307

Principles of Web Development

Lecture 13

Unit 4 – Servers

XAMPP * Apache * PHP

Class Outline

- About XAMPP
- Backend programming with PHP
- Synchronous communication with REST
  - (not to be confused with asynchronous communication)

Readings

- Internet and World Wide Web textbook
  - Chapter: 4.11, 23
- Full Stack Developer
  - Chapter 3
- Internet Resources
  - https://www.w3schools.com/php/default.asp
  - https://www.w3schools.com/html/html_forms.asp
  - https://www.tutorialspoint.com/html/html_forms.htm

What is XAMPP & Apache

XAMPP * Apache * PHP

About Flexible Applications

The ability to do the same thing in many different ways.

Apache:

- Queue packets
- Monitor packets by port #
- Route payload to host program belonging to port # using a local IP address
- Supports REST

Notice that Apache makes no restrictions on host programs, other than they are communicable by a local IP address. The host program then handles the data.

Flexibility comes in the many types of host programs: languages, services, databases.

What is XAMPP?

A collection of server programs:

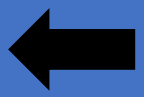
Apache – server (ports 80, 443)
MySQL – database (MariaDB) (ports 156)
FileZilla – FTP (ports 20, 21, 115)
Mercury – email server (ports 25, 110)
Tomcat – Java servlets ----------(below) through Apache
PHP – server programming language
Perl – server text processing scripting language

X – extra

- It provides everything you need to run a general-purpose website
- In this course we will focus on Apache and MariaDB and PHP

COMP 307
Principles of Web Development

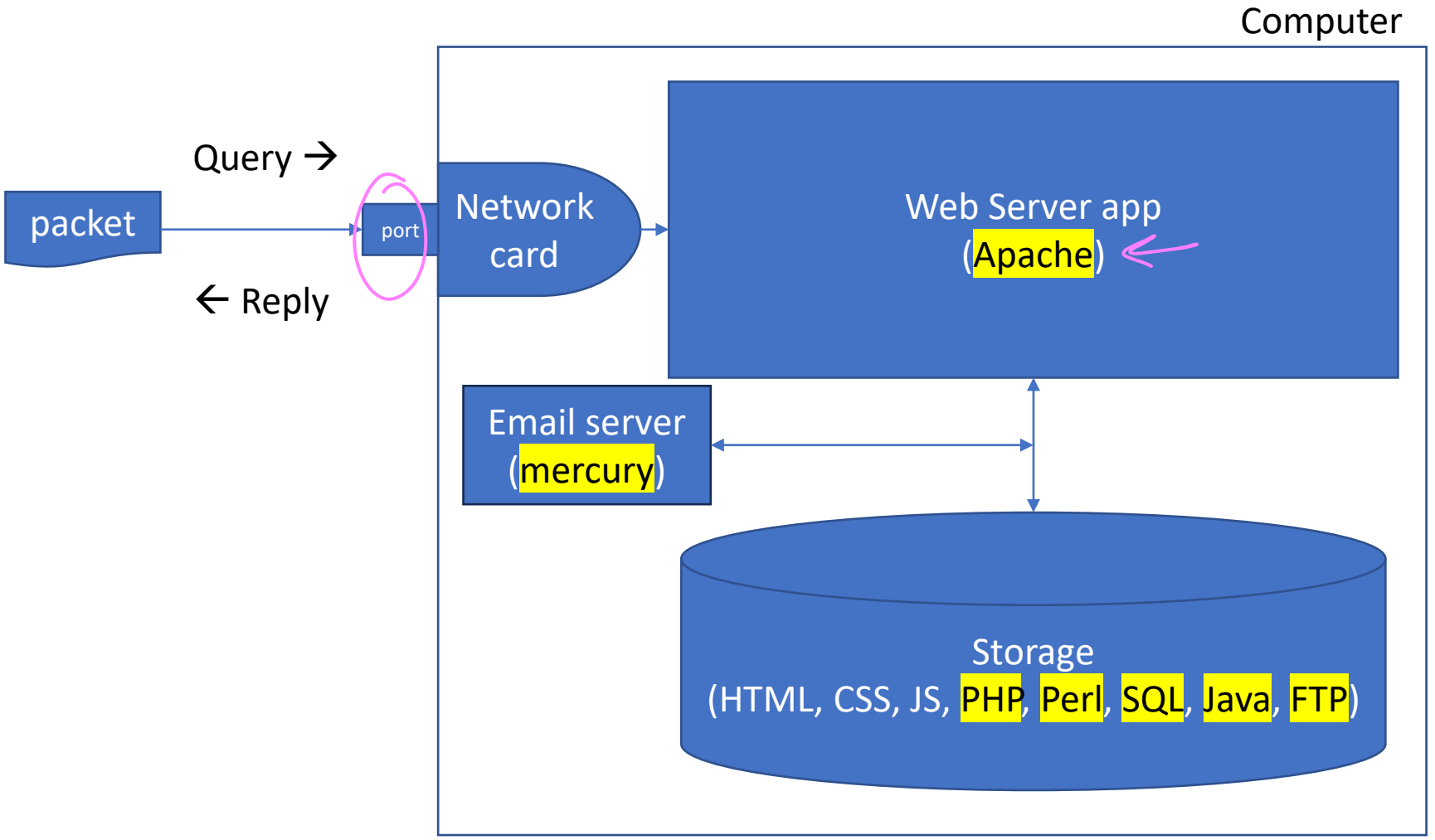
To browser for rendering



Contents

XAMPP Server

Apache POV



REST

Definition:

- REpresentational State Transfer
- We “represent” a query with a string
- We encode a data structure (“state”) and append it to the string
- And we send the string (“transfer”) somewhere to get processed

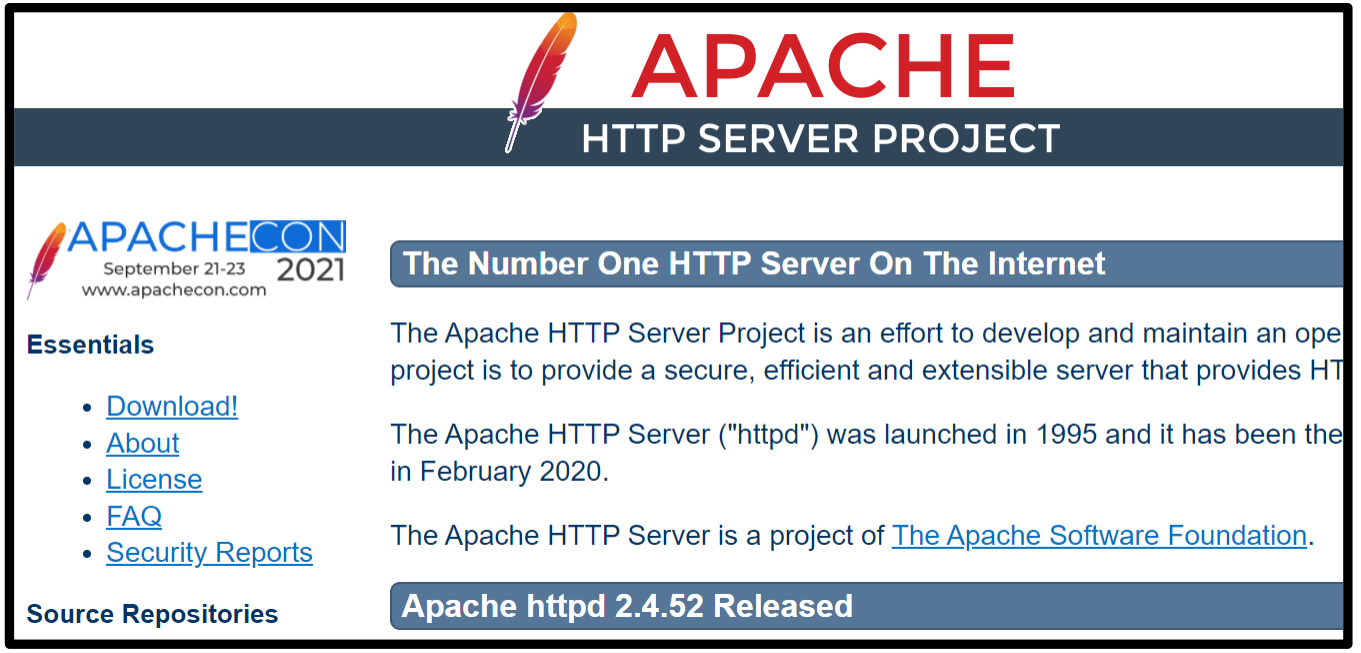
REST Methods

HTTP Method	CRUD	Collection Resource (e.g. /users)	Single Resouce (e.g. /users/123)
POST	Create	201 (Created), 'Location' header with link to /users/{id} containing new ID	Avoid using POST on a single resource
GET	Read	200 (OK), list of users. Use pagination, sorting, and filtering to navigate big lists	200 (OK), single user. 404 (Not Found), if ID not found or invalid
PUT	Update/Replace	405 (Method not allowed), unless you want to update every resource in the entire collection of resource	200 (OK) or 204 (No Content). Use 404 (Not Found), if ID is not found or invalid
PATCH	Partial Update/Modify	405 (Method not allowed), unless you want to modify the collection itself	200 (OK) or 204 (No Content). Use 404 (Not Found), if ID is not found or invalid
DELETE	Delete	405 (Method not allowed), unless you want to delete the whole collection — use with caution	200 (OK). 404 (Not Found), if ID not found or invalid

COMP 307
Principles of Web Development

Contents

https://www.apache.org/



https://httpd.apache.org/

An Apache Integrated Stack

Apache Friends Download Add-ons Hosting Community About Search.. Search EN

XAMPP Apache + MariaDB + PHP + Perl

What is XAMPP?

XAMPP is the most popular PHP development environment

XAMPP is a completely free, easy to install Apache distribution containing MariaDB, PHP, and Perl. The XAMPP open source package has been set up to be incredibly easy to install and to use.

XAMPP

Download
Click here for other versions

XAMPP for **Windows** 8.1.2 (PHP 8.1.2)

XAMPP for **Linux** 8.1.2 (PHP 8.1.2)

XAMPP for **OS X** 8.1.2 (PHP 8.1.2)

https://www.apachefriends.org/index.html

A nice packaged server with basic tools. Very popular.
DOWNLOAD IT!

XAMPP Demo

Apache – server
MySQL – database (MariaDB)
FileZilla – FTP
Mercury – email server
Tomcat – Java servlets

Apache Configuration

The **httpd.conf** file:

- Take your time to look at it. The file contains helpful comments.
- Default HTML file expected in every subdirectory and root: `index.html`
- Important config statements:
  - `Listen 127.0.0.1:80` `#for localhost socket`
  - `DocumentRoot "/path"` `#website root`

https://httpd.apache.org/docs/2.4/configuring.html

leaving all default is enough for us

Apache Supports

- Synchronous communication
  - User cannot use webpage while waiting for reply
    - E.g., login page, switch to a new webpage
  - E.g., CGI (Common Gateway Interface)
- Asynchronous communication* later
  - User can continue to interact with current webpage
    - E.g., used to call an API for data, not a new page.
    - Often used to update a portion of the webpage, not used to replace the entire page.
  - E.g., AJAX (**A**synch **J**S **A**nd **X**ML)

What is synchronous communication?

Definition:

- The browser sends a request to the server,
- Browser then waits for the answer.
- The server processes the request and returns a reply to the browser
- The browser only after it receives the reply starts to operate once again.
  - Browser screen clears
  - New complete webpage is displayed
  - User can interact with page

replaced w/ data coming back from server

COMP 307
Principles of Web Development

Browser Communication

Query string

```
<html>
<body>
        <form action="post.php" method="post">
                Name:<input type="text" name="name"></br>
                E-mail:<input type="text" name="email"></br>
                <input type="submit">
        </form>
</body>
</html>
```

browser

URL to server program

REST command

Like <script> but runs on server

```
<html>
<body>
        Welcome <?php echo $_POST["name"]; ?><br>
        Your email address is: <?php echo $_POST["email"]; ?>
</body>
</html>
```

server

Built-in access to query string

Server HTML file contains PHP code

Only the HTML/CSS/JS output to browser

embedded php to html

Contents

Example

HTML

```
<form  action="URL/prog"  method=”GET”>

        <textarea  name="feedback" rows=5 cols=20>
                My thoughts so far
        </textarea>

        <br />

        <input  type="submit"  name="button"  value="Send">

</form>
```

Spaces not permitted

packet

```
http://URL/prog?feedback=My+thoughts+so+far&button=Send
```

Notice how the HTML maps into the packet (ie. CGI string format).
Note: the packet is what is sent to the server after SUBMIT is pressed.

The CGI String

Common Gateway Interface syntax:

"URL/path/program?var=val&var2=val2"

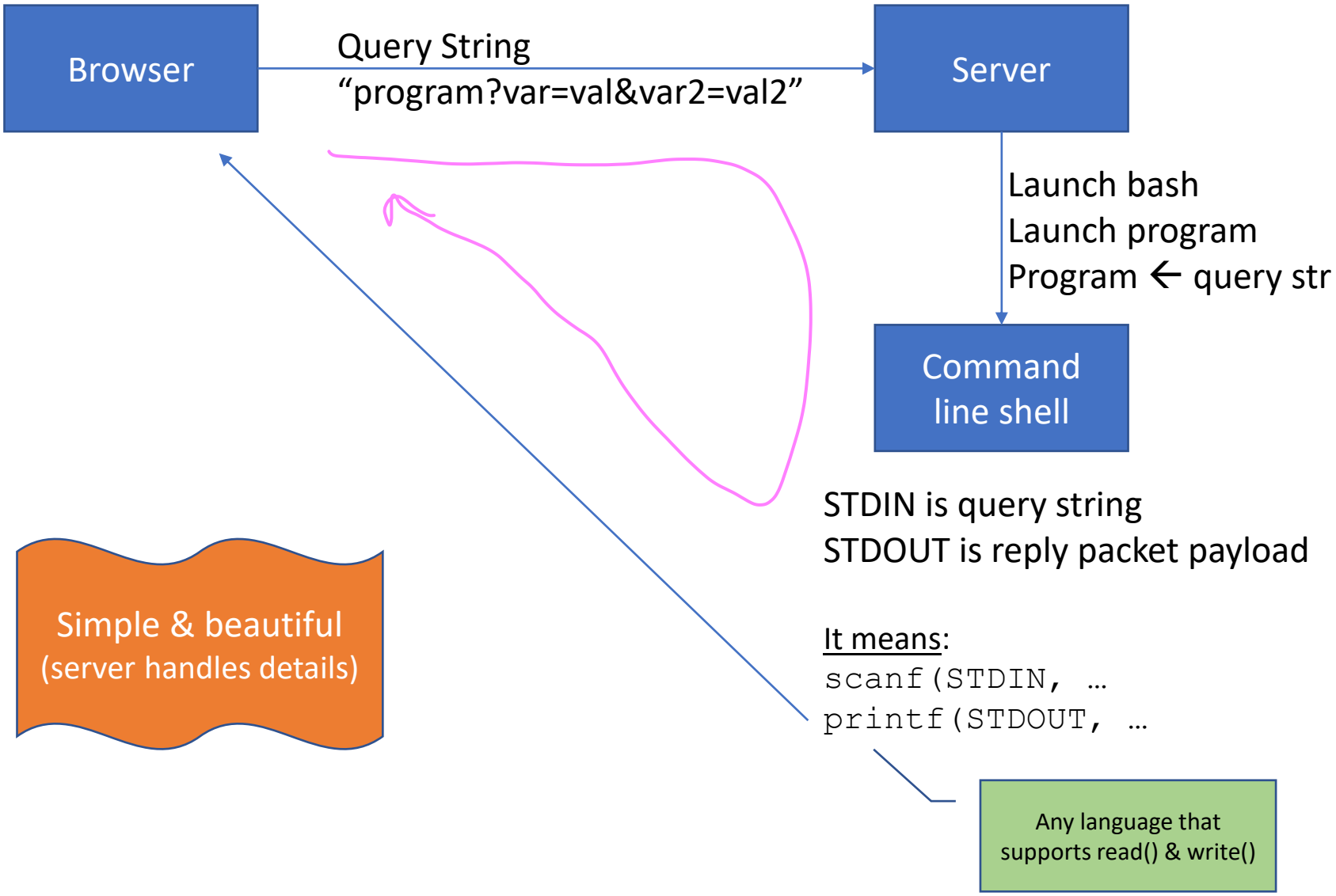
Special symbols

Example:

"http://server.com/public_html/my.py?age=20"

Rule: the CGI string must be a contiguous series of simple characters plus CGI symbols. Simple characters means a-z,A-Z,0-9 some ASCII symbols.

Basic CGI Communication



CGI Formatting Rules

- Format
  - Must be a single string (a vector) without spaces (ampersand separated)
- Reserved symbols
  - ? - indicates beginning of query string
  - = - separation symbol between variable and value
  - & - separation symbol between query tuples
  - + - the replacement character for space
  - % - ASCII escape code
- ASCII Characters
  - %12 - Insert the ASCII code into the string with %## to specify special ASCII characters (other languages) or to use reserved symbols

CGI Formatting Rules

- Example
  - “My name is Béba”
  - “My+name+is+B%30ba”

Introduction to PHP

XAMPP * Apache * PHP

What's nice about PHP?

- You program it like JavaScript directly within the HTML
- But you can write proprietary code and data in PHP because it is only executed on the server. The browser only receives the outputs from PHP. The PHP code is not downloaded to the browser.
- A single `filename.php` can have: HTML, CSS, JS and PHP all together.

Example.php

Example
(original code)

```
<html>
        <head> .....  </head>
        <body>
                ... html and css stuff ....
                ... java script stuff ....

                <?php
                        echo "hello world<br/>";
                        phpInfo();
                ?>

                ... more html css and JS ...
        </body>
</html>
```

Notice how code is added into HTML doc

Before sending the HTML to the client, the PHP is removed by executing it

The purpose of executing it is to generate more web page content from databases, CSV files, other HTML files, etc.

Example

(after server executed – what browser sees)

```
<html>
        <head> .....  </head>
        <body>
                ... html and css stuff ....
                ... java script stuff ....

                hello world<br/>
                PHP Statistics:
                ……

                ... more html css and JS ...
        </body>
</html>
```

Notice how packet contains TEXT is added into HTML without PHP

Comments and Expressions

```
<!DOCTYPE html>
<html>
<body>

<?php
// This is a single-line comment

# This is also a single-line comment

/*
This is a multiple-lines comment block
that spans over multiple lines
*/

// You can also use comments to leave out
// parts of a code line
$x = 5 /* + 15 */ + 5;
echo $x;
?>

</body>
</html>
```

Use of $ like in Bash

Strings

```
•echo strlen("Hello world!"); // outputs 12
•echo str_word_count("Hello world!"); //  outputs 2
•echo strrev("Hello world!"); // outputs !dlrow olleH
•echo strpos("Hello world!", "world"); // outputs 6
•echo str_replace("world", "Dolly", "Hello world!");
                        // outputs Hello Dolly!
```

If Statements

```
<?php
  $t = date("H");

  if ($t < "20") {
    echo "Have a good day!";
  }
?>
```

```
<?php
  $t = date("H");

  if ($t < "10") {
    echo "Have a good morning!";
  } elseif ($t < "20") {
    echo "Have a good day!";
  } else {
    echo "Have a good night!";
  }
?>
```

Loops

Notice we output HTML always

```
$x = 1;

while($x <= 5) {
    echo "The number is: $x <br>";
    $x++;
}
```

```
$x = 1;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
```

```
for ($x = 0; $x <= 10; $x++) {
    echo "The number is: $x <br>";
}
```

- like bash and C mixed

File Processing

```
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");

// Output one line until end-of-file

while(!feof($myfile)) {
  echo fgets($myfile) . "<br>";
}

fclose($myfile);
```

exit command returns a string not int

Does this remind you of C?

HTTP Commands

```
// Prevent page caching on browser

<?php
// Date in the past
header("Expires: Mon, 26 Jul 1997 05:00:00 GMT");
header("Cache-Control: no-cache");
header("Pragma: no-cache");
?>

// Prompt user to save PDF file
<?php

header("Content-type:application/pdf");

// It will be called downloaded.pdf
header("Content-Disposition:attachment;filename='downloaded.pdf'");

// The PDF source is in original.pdf
readfile("original.pdf");
?>
```

PHP Environment Variables

- $GLOBAL
  - Used to access global variables from anywhere in the PHP script
- $_SERVER
  - Holds information about headers, paths, and script locations
- $_REQUEST
  - Used to collect data after submitting an HTML form
- $_POST
  - Used to collect form data after submitting an HTML form. Also used to pass variables
- $_GET
  - Collect data sent in the URL

DEMO XAMPP POST

`HTDOS/example/post.html`

`HTDOS/example/post.php`

Example PHP Programs

```
// https://www.phptutorial.net/php-tutorial/php-csv/
// Writing to a CSV file

<?php
$data = [
          ['Symbol', 'Company', 'Price'],
          ['GOOG', 'Google Inc.', '800'],
          ['AAPL', 'Apple Inc.', '500'],
          ['AMZN', 'Amazon.com Inc.', '250'],
          ['YHOO', 'Yahoo! Inc.', '250'],
          ['FB', 'Facebook, Inc.', '30'],
];

$filename = 'stock.csv';

// open csv file for writing
$f = fopen($filename, 'w');
if ($f === false) {
          die('Error opening the file ' . $filename);
}

// write each row at a time to a file
foreach ($data as $row) {
          fputcsv($f, $row);
}

// close the file
fclose($f);

?>
```

Example PHP Programs

```
// https://www.phptutorial.net/php-tutorial/php-csv/
// Reading from a CSV file

<?php

$filename = './stock.csv';
$data = [];

// open the file
$f = fopen($filename, 'r');
if ($f === false) {
        die('Cannot open the file ' . $filename);
}

// read each line in CSV file at a time
while (($row = fgetcsv($f)) !== false) {
        $data[] = $row;
}

// close the file
fclose($f);

?>
```

Example PHP Programs

```
// https://www.phptutorial.net/php-tutorial/php-csv/
// Processing UNICODE

fputs($f, (chr(0xEF) . chr(0xBB) . chr(0xBF))); // support unicode
```

Prepare for Next Class

- Assignments
  - Mini 5 due next class
- Labs
  - Lab C (XAMPP and PHP)
- On your own
  - Install XAMPP on your personal computer (laptop)
  - Create an HTML page with a <form> tag that calls a PHP program to displays the fields passed to it. The PHP output will overwrite the webpage on the browser.