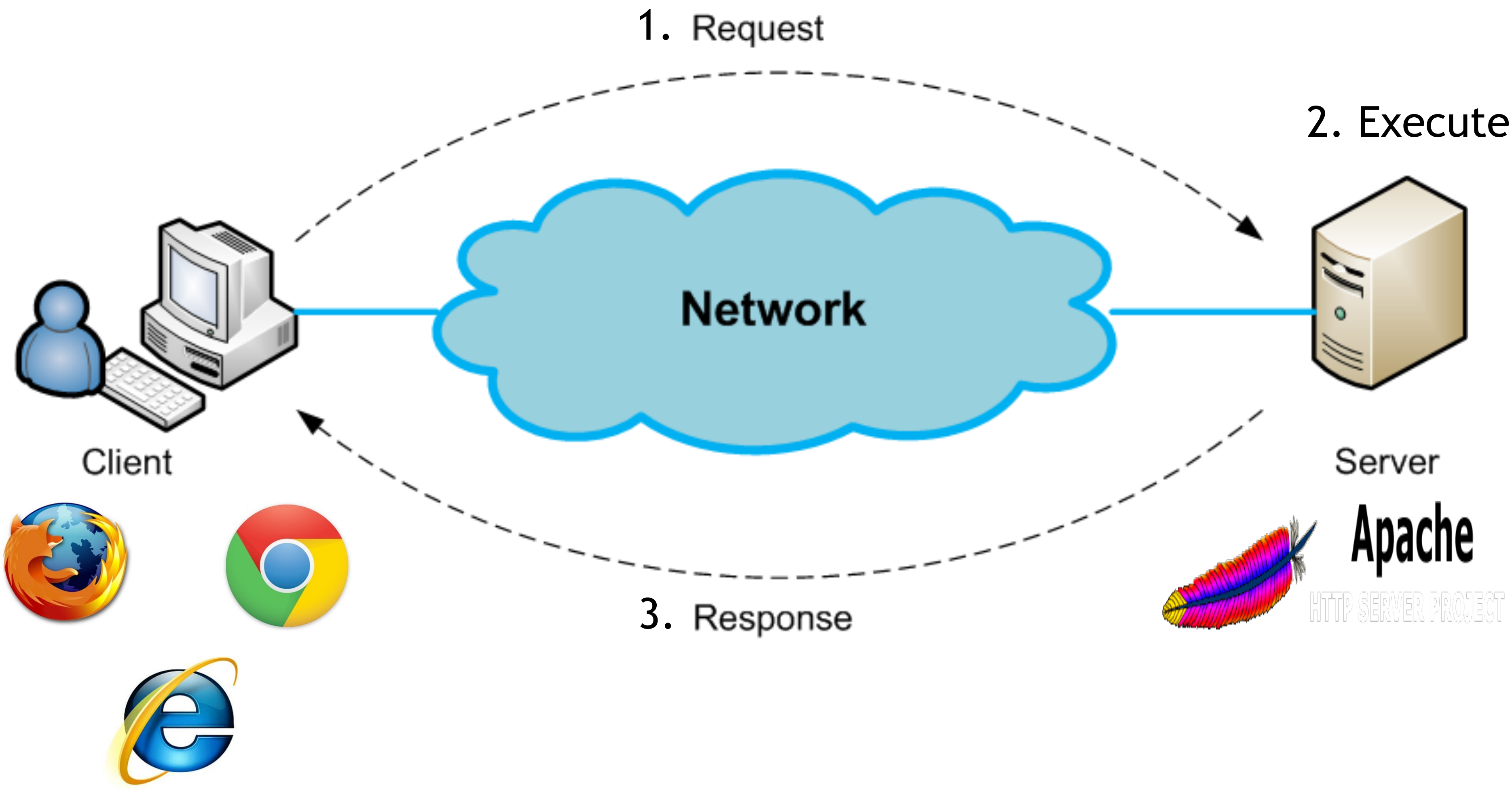
Web Application Development

Server-side script: PHP

Web Client - Server



1. Request

2. Execute

3. Response

Server-side Scripts

- Scripts are programs developed to run at server side. Results of the scripts will be sent to clients
- Examples of server-side scripts
    - JSP: JavaServer Pages
    - ASP: Active Server Pages
    - PHP: Hypertext Preprocessor

PHP

- PHP is a server scripting language for creating dynamic and interactive webpages
- PHP is open-source and platform independent. It runs on various platforms such as Windows, Linux, Mac OS X, etc.
- PHP supports many databases such as MySQL, PostgreSQL, Microsoft SQL, etc.
- **PHP is currently the most popular server-side scripting language**

Installing PHP

- Two main methods to install PHP:
  - Use a Web Host with PHP support
    - https://www.000webhost.com
    - https://www.phpwebhosting.com
  - Install PHP on your own computer
    - Install a web server (e.g. Apache)
    - Install PHP (http://php.net/manual/en/install.php)
    - Install a database (e.g. MySQL)

PHP Development Environment

- XAMPP is the most popular PHP development environment, consisting of:
  - Apache web server
  - PHP interpreter
  - MySQL database server
- XAMPP is an acronym for X (cross-platform), Apache(Web server), MySQL Database, PHP Language and Perl language
- XAMPP can be freely downloaded at:

  https://www.apachefriends.org
- Besides XAMPP, there are other PHP development environments: WAMP, MAMP, LAMP for Windows, Mac and Linux respectively

PHP Syntax

- A PHP script **starts with <?php and ends with ?>**
- A PHP script can be placed at anywhere in the HTML document

```
<html>
  <body>
      <h1>My first PHP page</h1>
      <?php
      echo "Hello World!";
      ?>
  </body>
</html>
```

PHP Syntax

- PHP scripts should be saved with the **.php** file extension (for example, **index.php**)
- PHP scripts must be run on a **PHP-enabled Web server** (whereas HTML pages can be viewed on any computer, directly in a browser)

PHP Syntax

- Whitespace insensitive
- Statements ended by semicolon
- Comments
  - /*….*/ for multiple lines
  - // or # for a line

```
<?php
// this is a comment
echo "Hello World!";
?>
```

PHP Variables

- All variables in PHP are denoted with a **leading dollar sign $**
- Variables can, but do not need, to be declared before assignment
- Variables used before they are assigned have default values

```
<?php
 $txt = "Hello world!";
 $x = 15;
?>
```

PHP Variables

- PHP pre-defined variables are super-global variables which can be accessed from anywhere
- Examples of super-global variables are: $GLOBALS, $_SERVER, $_POST, $_GET, etc.

```
<?php
  echo $_SERVER['SERVER_NAME'];
?>
```

PHP Data Types

- String
- Integer
- Double
- Boolean
- NULL
- Array
- Object

PHP Data Types

- Strings are sequence of characters
  - ‘PHP Class’ or “PHP Class”
- Integers are whole numbers, without a decimal point
  - 2   34   756
- Doubles are floating-point numbers
  - 3.14   675.34
- Booleans have only two possible values:
  - TRUE and FALSE
- NULL is a special type that only has one value:
  - NULL

PHP Data Types

```php
<?php
 // I am a string
 $txt = "Hello world!”;
 // I am an integer
 $x = 15;
 // I am a double
 $y = 15.35;
 // I am a boolean
 $z = true;
 // I am a null
 $t = null;
?>
```

PHP Data Types

- Arrays are named and indexed collections of other values

```
<?php
  // I am an array of strings
  $cars = array(“Volvo","BMW","Toyota");
?>
```

PHP Data Types

- Objects are instances of programmer-defined classes, which can package up both other kinds of values and functions that are specific to the class

```
<?php
    class Car {
        function Car() {
            $this->model = "VW";
        }
    }
    // I am an object
    $myObj = new Car();

    // Display me
    echo $myObj->model;
?>
```

PHP Control Structures

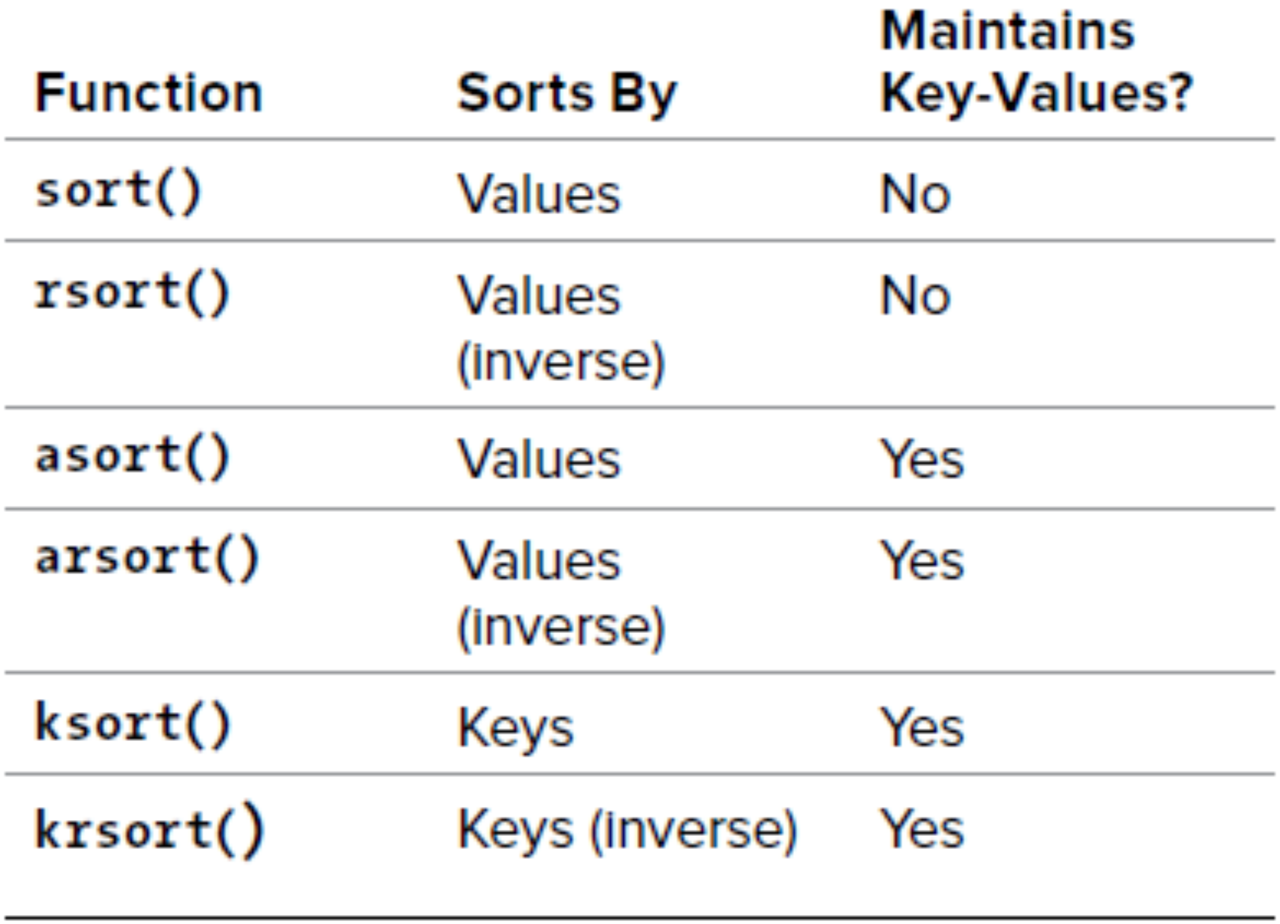
Similar to JavaScript

- if, if else
- for
- while
- switch … case …

Working with Arrays

- $list = array ('apples', 'bananas', ‘oranges');
  - $list[0], $list[1], $list[2]
- $soups = array ('Monday' => 'Clam Chowder', 'Tuesday' => 'White Chicken Chili', 'Wednesday' => ‘Vegetarian’);
  - $soups[‘Monday’], $soups[‘Tuesday’], $soups[‘Wednesday’]

Working with Arrays



Function	Sorts By	Maintains Key-Values?
`sort()`	Values	No
`rsort()`	Values (inverse)	No
`asort()`	Values	Yes
`arsort()`	Values (inverse)	Yes
`ksort()`	Keys	Yes
`krsort()`	Keys (inverse)	Yes

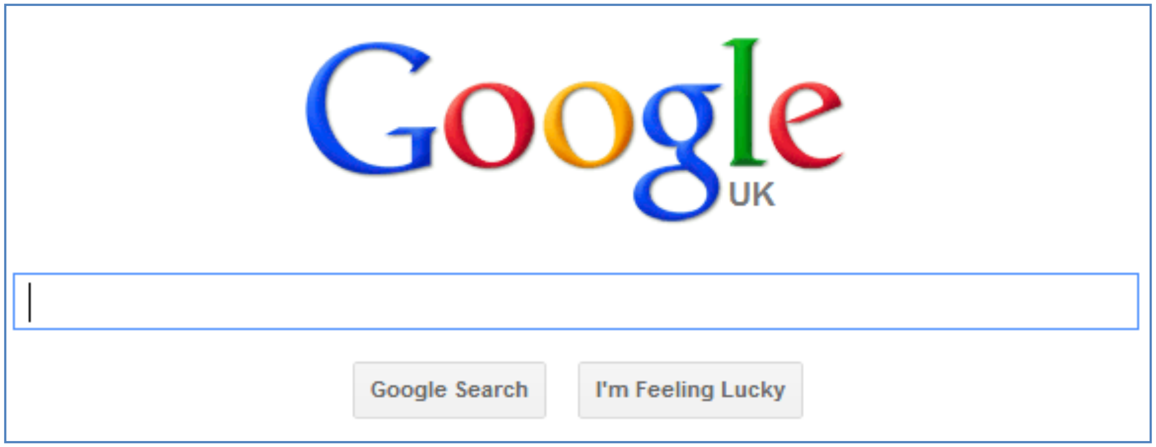
Working with Arrays

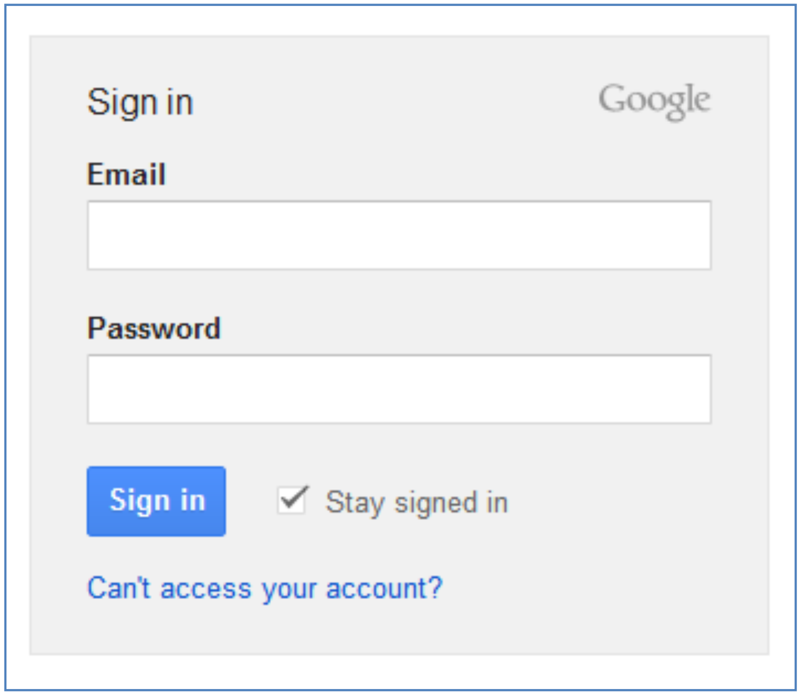
```
<?php
  $cars = array("Volvo", "BMW", "Toyota");
  sort($cars);
?>
```

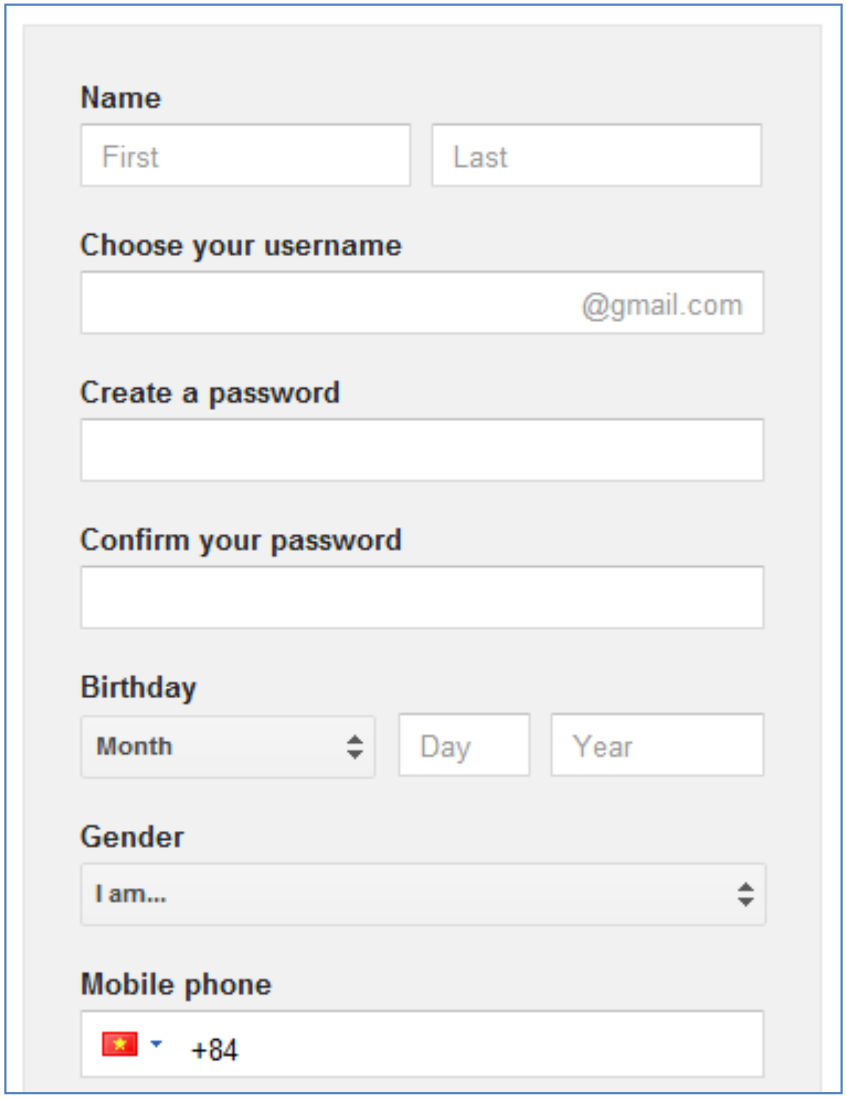
Results:

“BMW”, “Toyota”, “Volvo”

Working with Forms

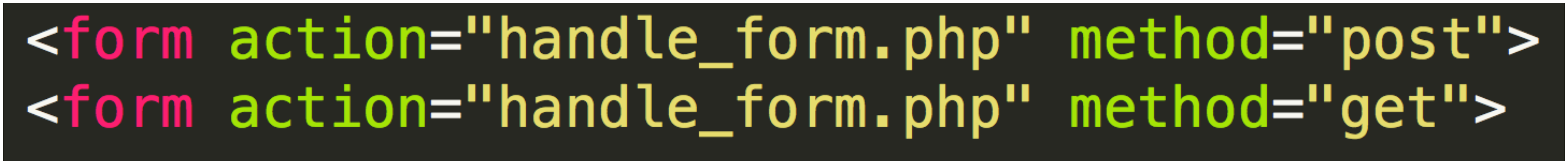






Working with Forms

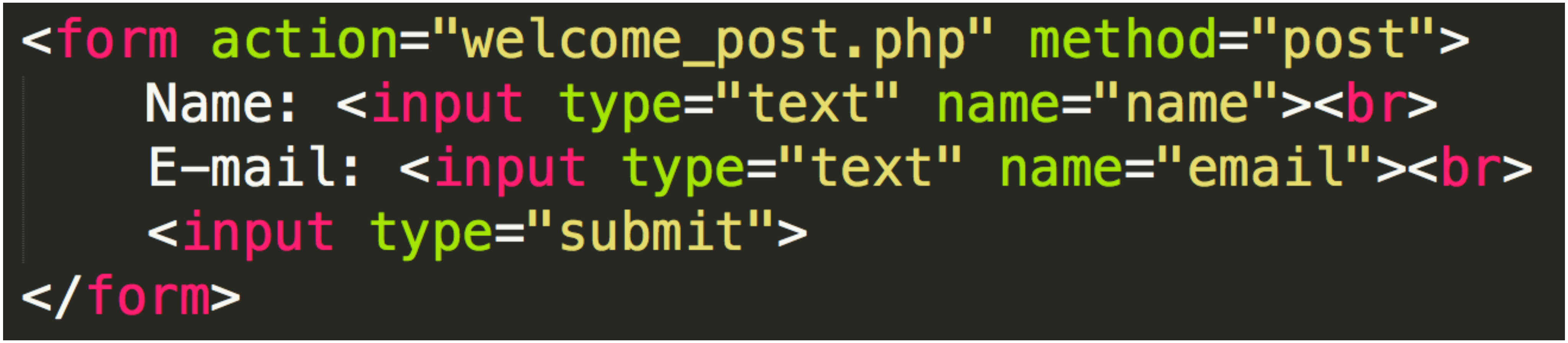
- Define form processing method:



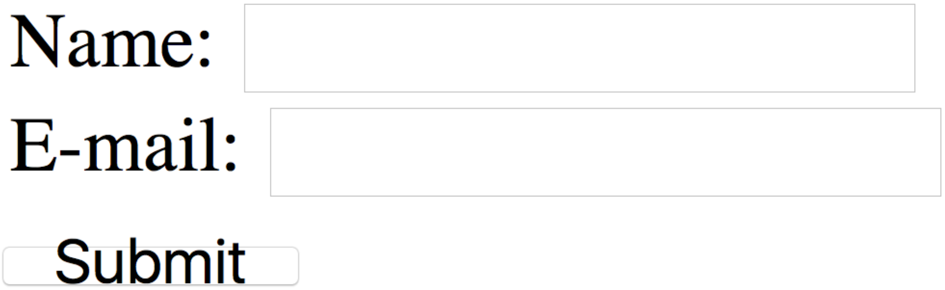
```
<form action="handle_form.php" method="post">
<form action="handle_form.php" method="get">
```

- Receive form data:
  - $_GET['form_element_name']
  - $_POST['form_element_name']

Define form processing method



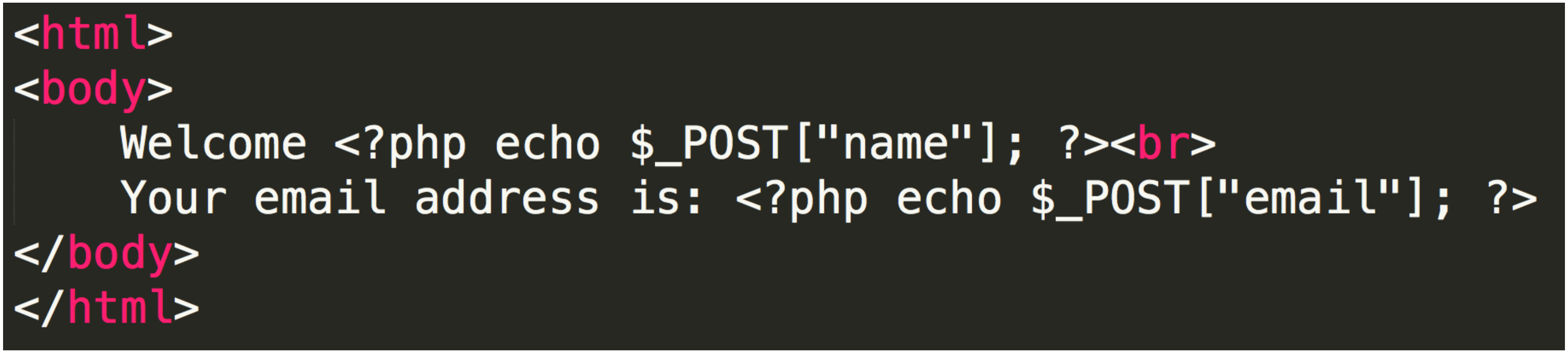
```
<form action="welcome_post.php" method="post">
    Name: <input type="text" name="name"><br>
    E-mail: <input type="text" name="email"><br>
    <input type="submit">
</form>
```



action=“welcome_post.php” method=“post”

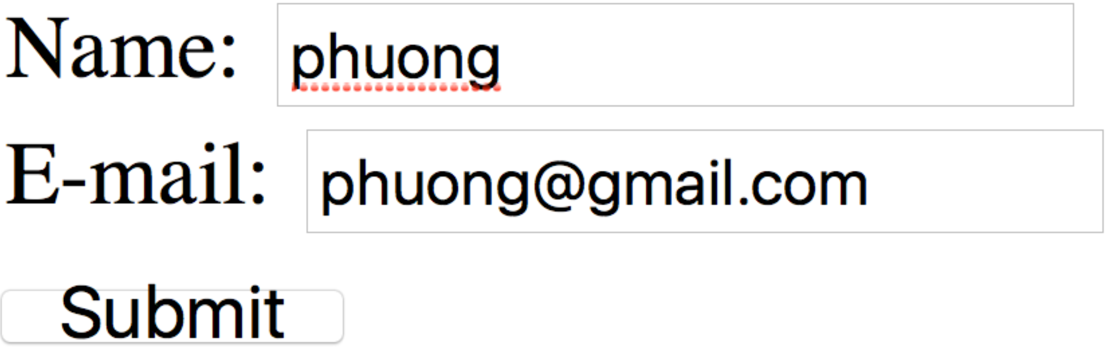
Retrieving form data

The welcome_post.php file contains the following content:



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

User-filled forms:

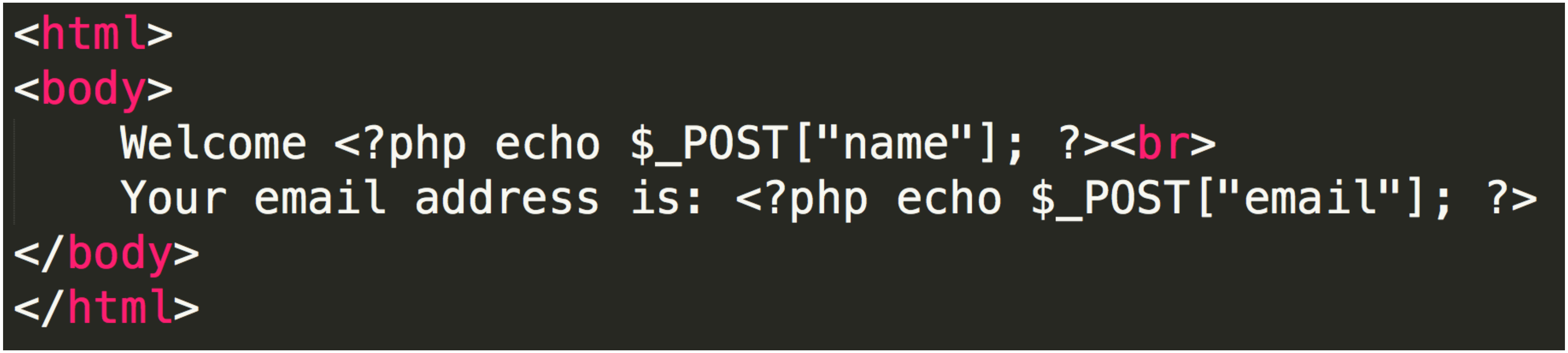


Name: phuong

E-mail: phuong@gmail.com

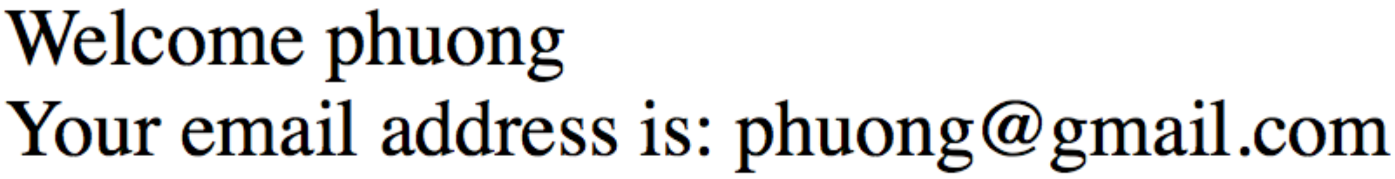
Retrieving form data

The welcome_post.php file contains the following content:



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

Results:



Welcome phuong
Your email address is: phuong@gmail.com