Form Processing in PHP

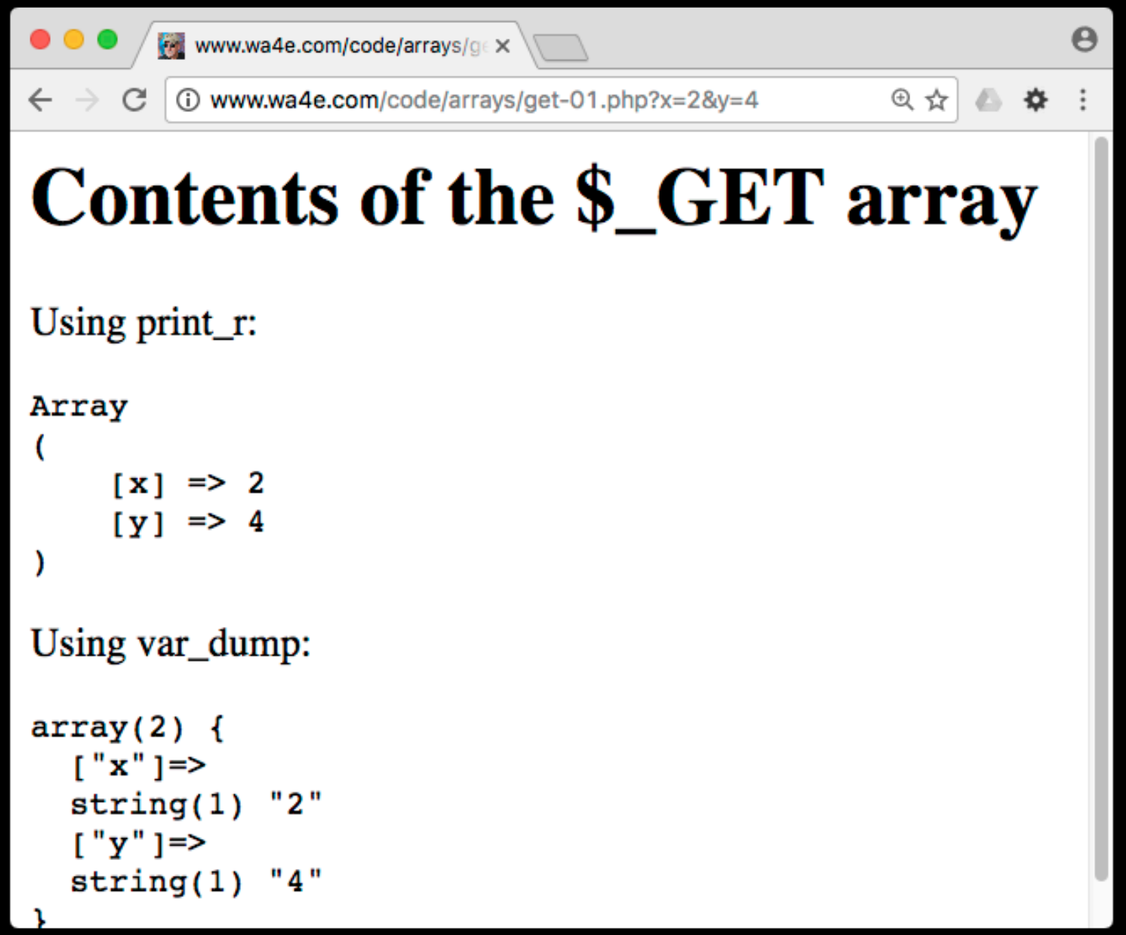
Dr. Charles Severance
www.wa4e.com

http://www.wa4e.com/code/forms
http://www.wa4e.com/code/forms.zip

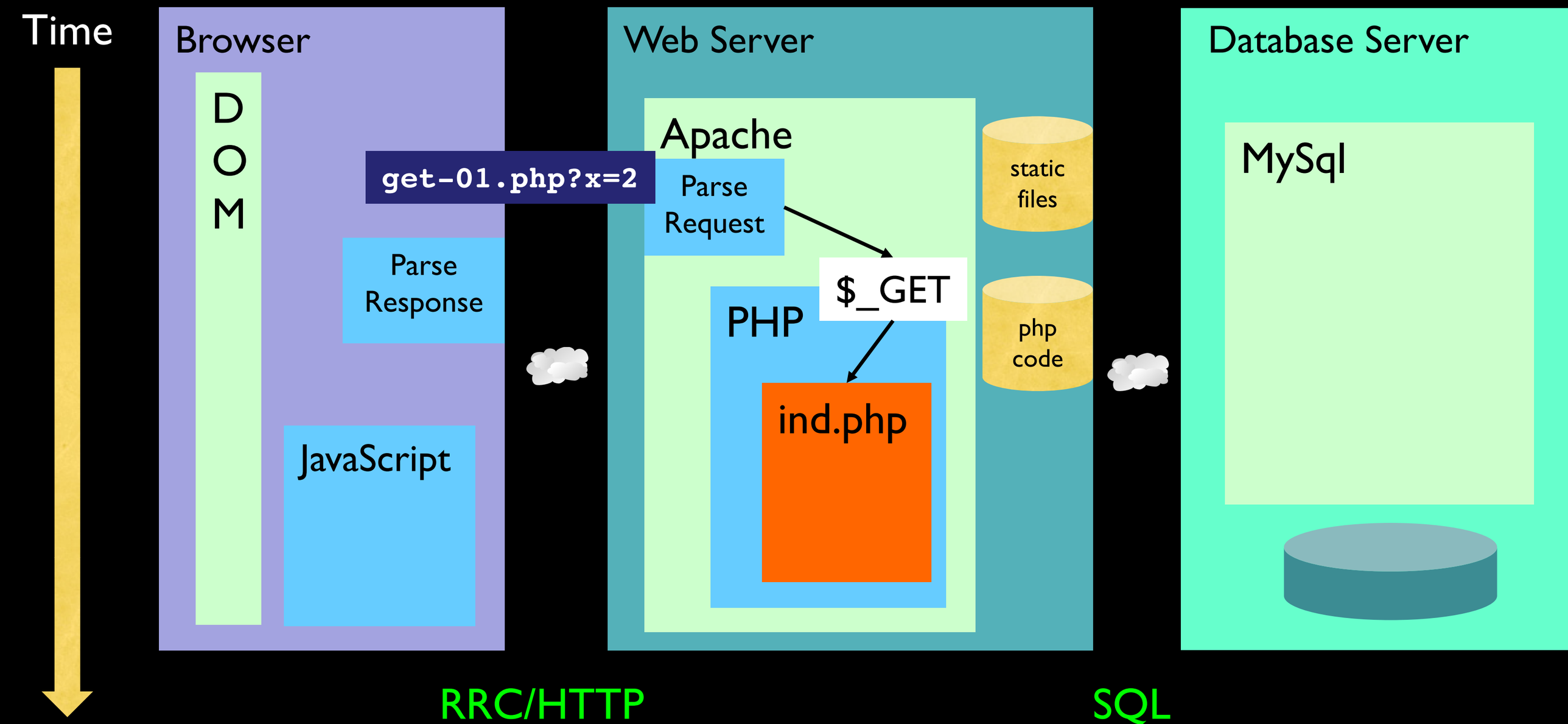
PHP Global Variables

- Part of the goal of PHP is to make interacting with HTTP and HTML as easy as possible.
- PHP processes the incoming HTTP request based on the protocol specifications and drops the data into various super global variables (usually arrays).

```php
<h1>Contents of the $_GET array</h1>
<p>Using print_r:</p>
<pre>
<?php
  print_r($_GET);
?>
</pre>
<p>Using var_dump:</p>
<pre>
<?php
  var_dump($_GET);
?>
</pre>
```

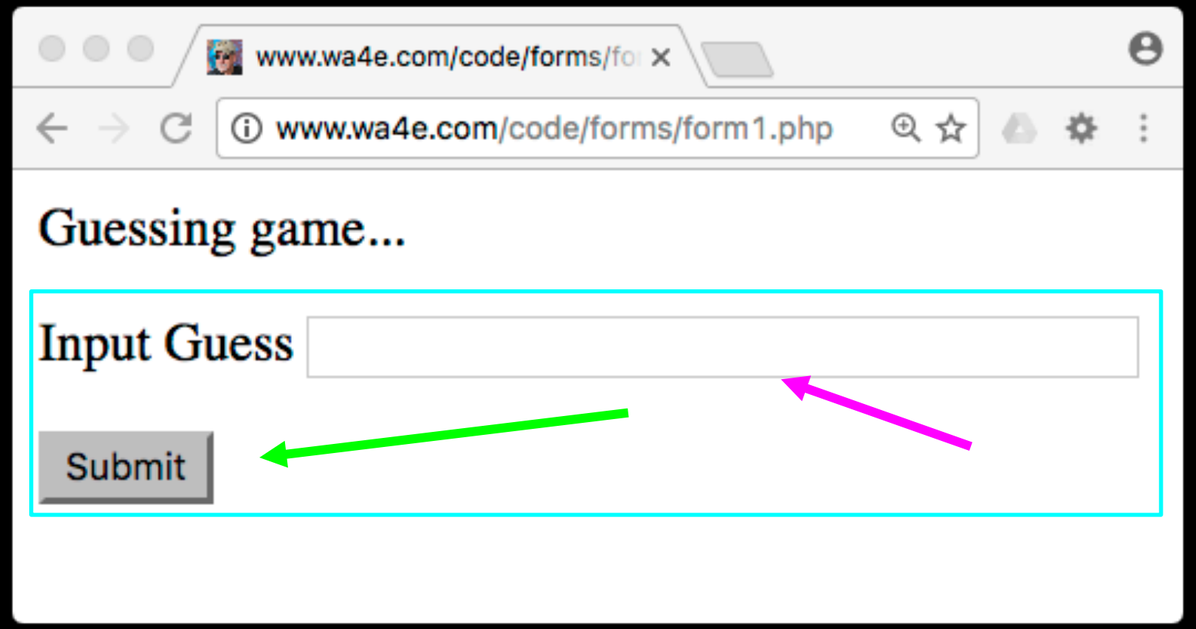


http://www.wa4e.com/code/arrays/get-01.php


Time
Browser
DOM
Parse Response
JavaScript
get-01.php?x=2
Web Server
Apache
Parse Request
$_GET
PHP
ind.php
static files
php code
Database Server
MySql
RRC/HTTP
SQL

Forms – User Input / Action

```
<p>Guessing game...</p>
<form>
    <p><label for="guess">Input Guess</label>
    <input type="text" name="guess" id="guess"/></p>
    <input type="submit"/>
</form>
```



http://www.wa4e.com/code/forms/form1.php

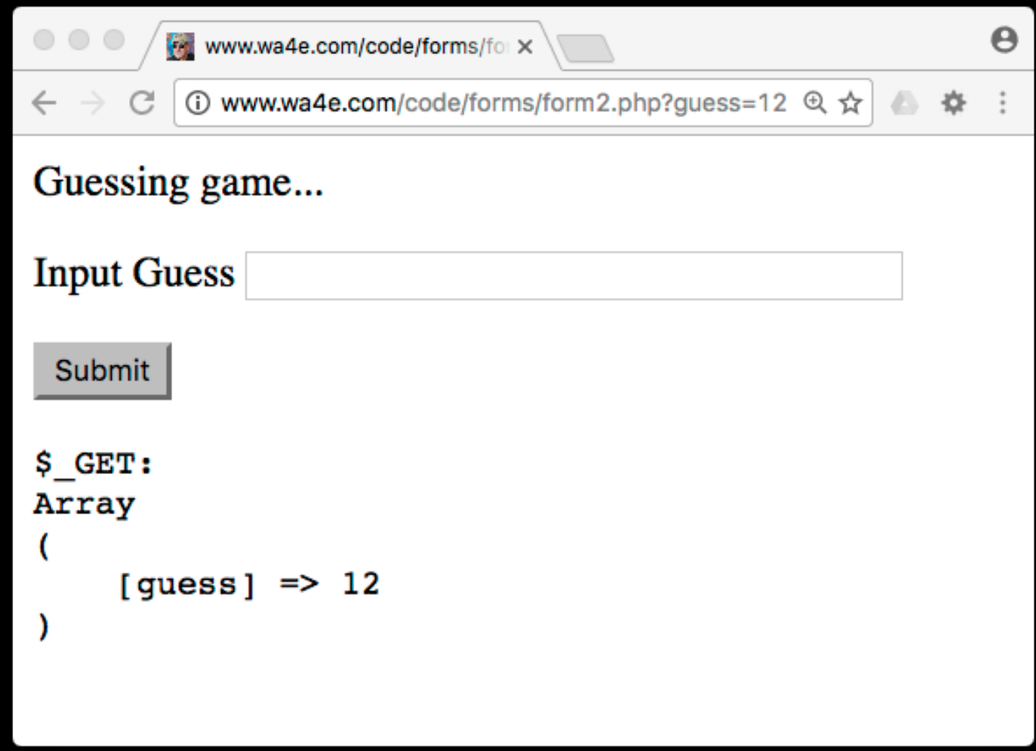
form1.php

Forms Submit Data

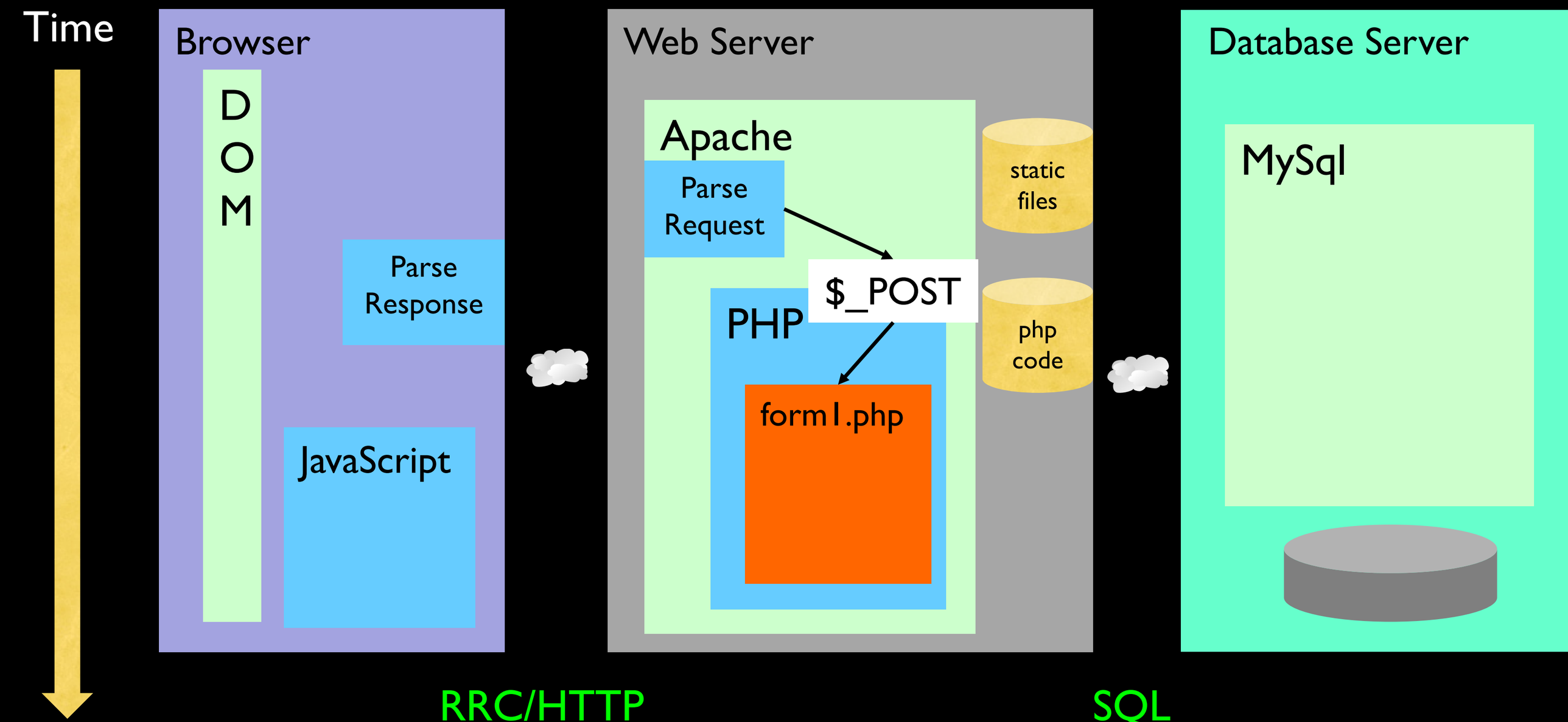
```
<p>Guessing game...</p>
<form>
    <p><label for="guess">Input Guess</label>
    <input type="text" name="guess" id="guess"/></p>
    <input type="submit"/>
</form>
```

form2.php

```
<p>Guessing game...</p>
<form>
    <p><label for="guess">Input Guess</label>
    <input type="text" name="guess" id="guess"/></p>
    <input type="submit"/>
</form>
<pre>
$_GET:
<?php
    print_r($_GET);
?>
</pre>
```

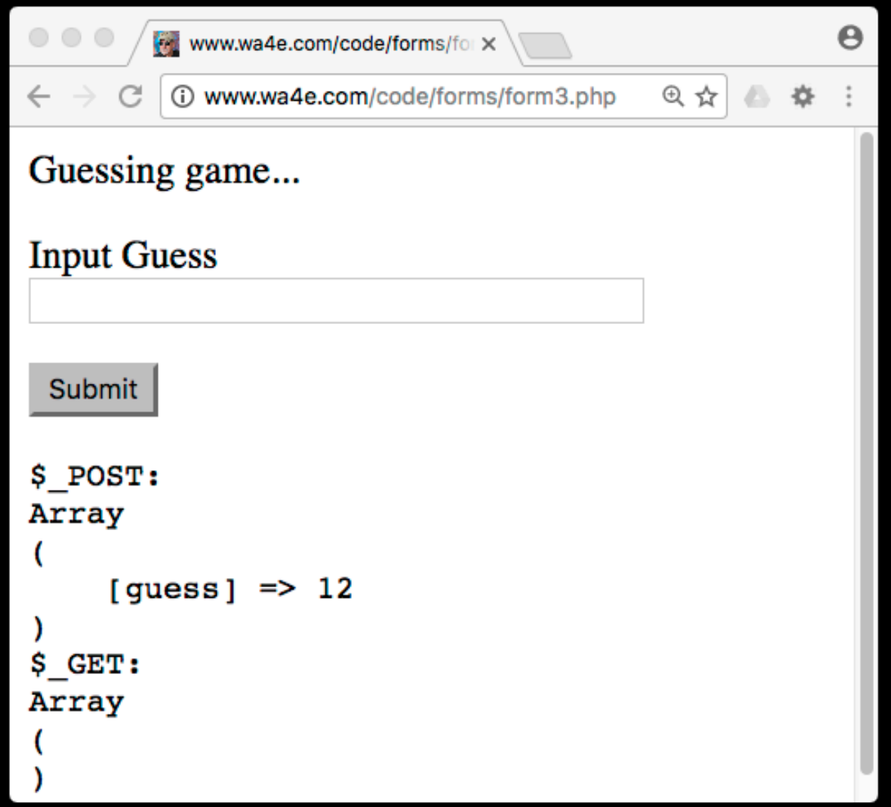


Using GET and POST with Forms


Time
Browser
DOM
Parse Response
JavaScript
Web Server
Apache
Parse Request
$_POST
PHP
form1.php
static files
php code
Database Server
MySql
RRC/HTTP
SQL

form3.php

```
<p>Guessing game...</p>
<form method="post">
    <p><label for="guess">Input Guess</label>
    <input type="text" name="guess" size="40" id="guess"/></p>
    <input type="submit"/>
</form>
<pre>
$_POST:
<?php
    print_r($_POST);
?>
$_GET:
<?php
    print_r($_GET);
?>
</pre>
```

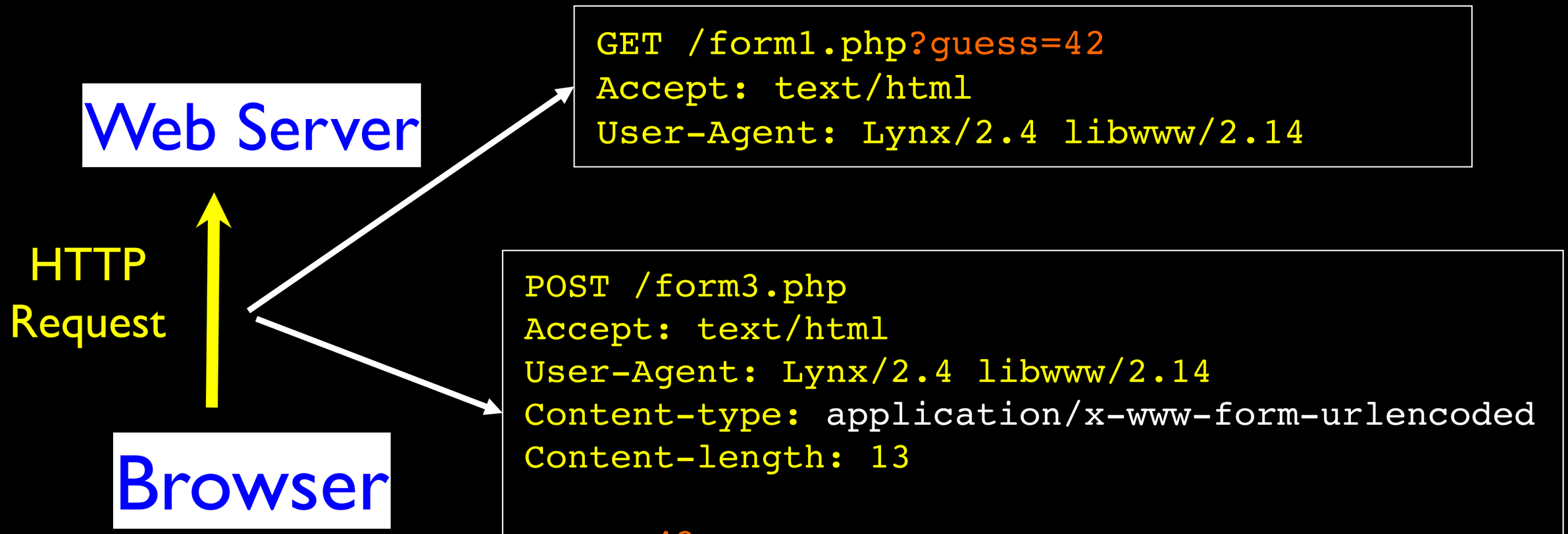


Forms GET vs. POST

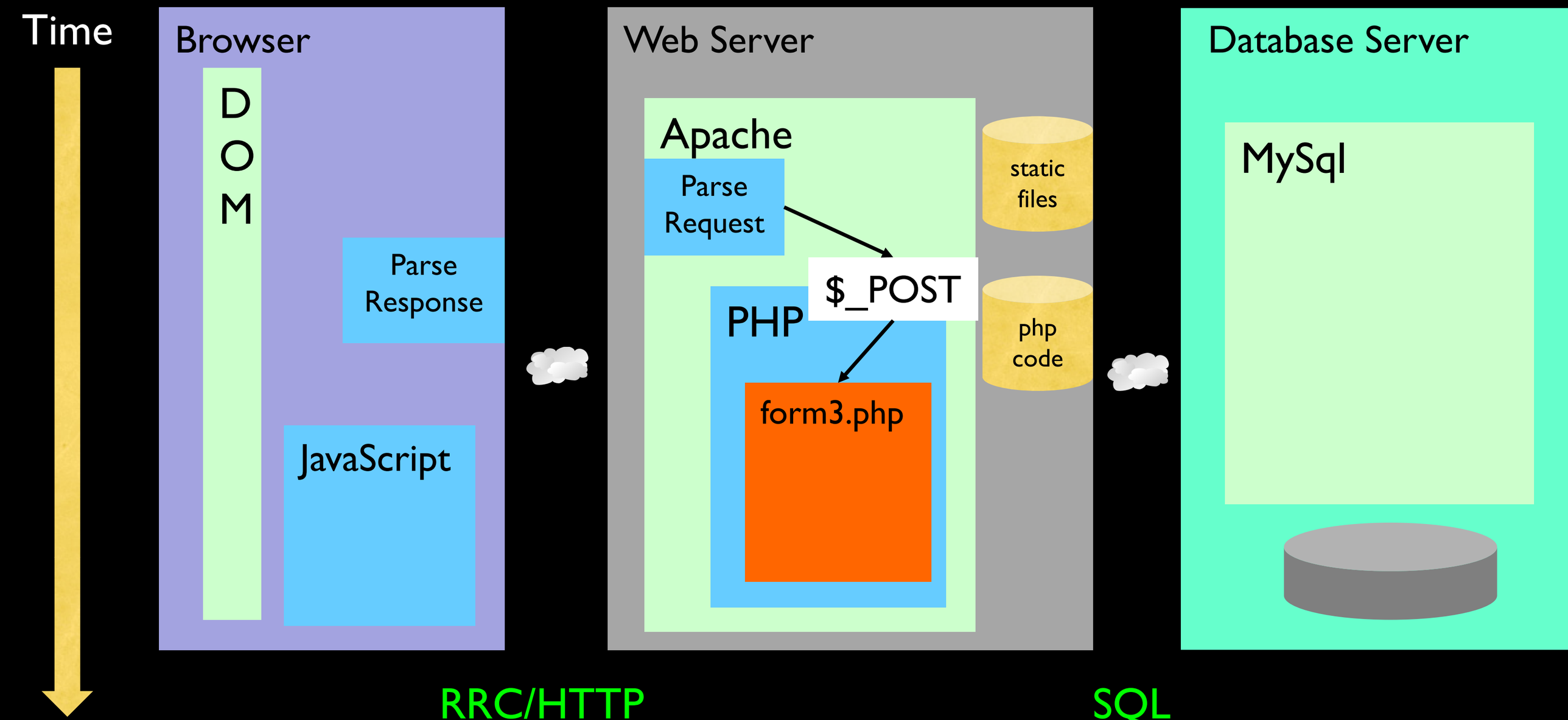
Two ways the browser can send parameters to the web server

- GET - Parameters are placed on the URL which is retrieved.
- POST - The URL is retrieved and parameters are appended to the request in the the HTTP connection.

Passing Parameters to The Server




```
<input type="text" name="guess" id="yourid" />
```


Time
Browser
DOM
Parse Response
JavaScript
Web Server
Apache
Parse Request
$_POST
PHP
form3.php
static files
php code
Database Server
MySql
RRC/HTTP
SQL

Rules of the POST/GET Choice

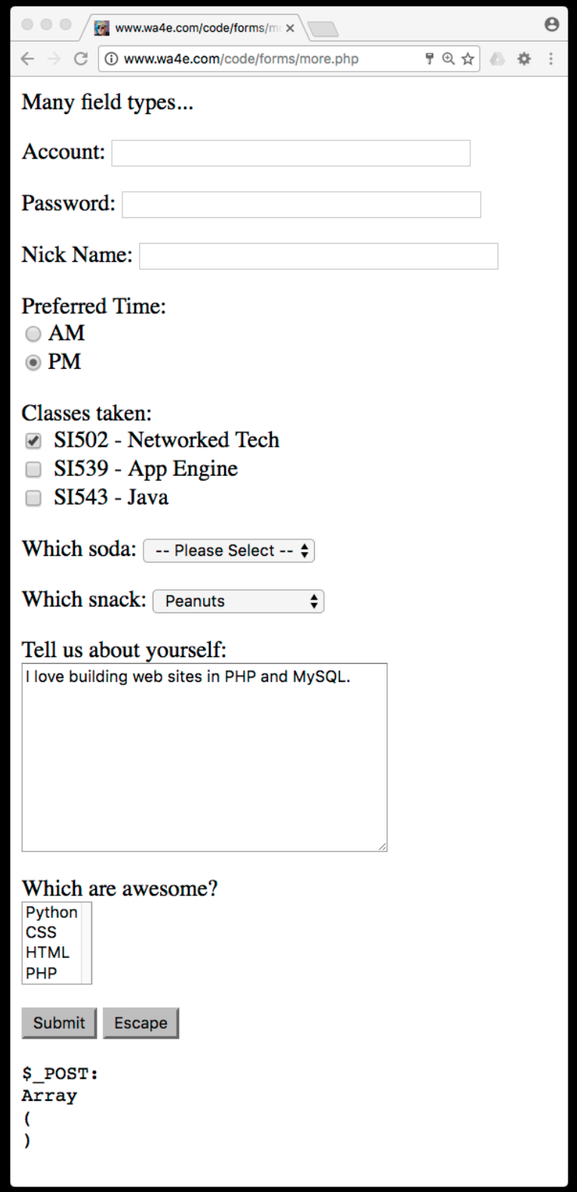
- POST is used when data is being created or modified.
- GET is used when your are reading or searching things.
- Web search spiders will follow GET URLs but generally not POST URLs.
- GET URLs should be “idempotent” - the same URL should give the “same thing” each time you access it.
- GET has an upper limit of the number of bytes of parameters and values (think about 2K).

HTML Input Types

Other Input Types

- Text
- Password
- Radio Button
- Check Box
- Select / Drop-Down
- Textarea

http://www.wa4e.com/code/forms/more.php



more.php

```
<p>Many field types...</p>
<form method="post" action="more.php">
   <p><label for="inp01">Account:</label>
   <input type="text" name="account" id="inp01" size="40" ></p>
   <p><label for="inp02">Password:</label>
   <input type="password" name="pw" id="inp02" size="40" ></p>
   <p><label for="inp03">Nick Name:</label>
   <input type="text" name="nick" id="inp03" size="40" ></p>
```


```
$_POST:
Array
(
    [account] => Beth
    [pw] => 12345
    [nick] => BK
    [when] => pm
  ...
)
```

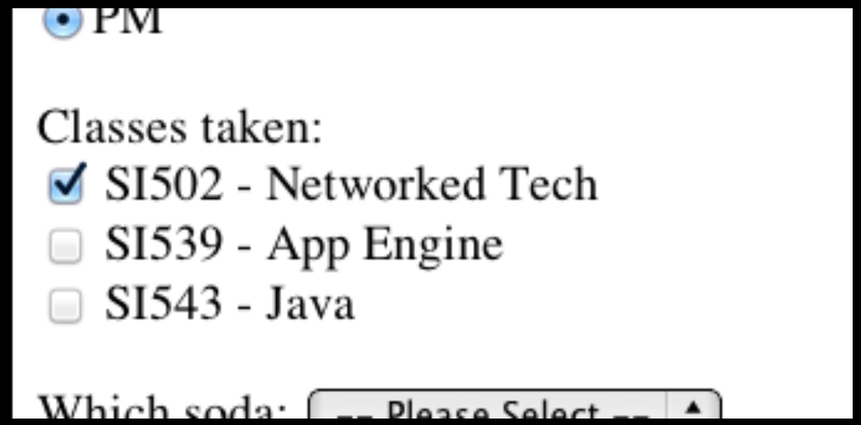
more.php

```
<p>Preferred Time:<br/>
 <input type="radio" name="when" value="am">AM<br>
 <input type="radio" name="when" value="pm" checked>PM</p>
```


```
$_POST:
Array(
   ...
      [nick] => BK
      [when] => pm
      [class] => si502
   ...
)
```

```
<p>Classes taken:<br/>
 <input type="checkbox" name="class1" value="si502" checked>
     SI502 - Networked Tech<br>
 <input type="checkbox" name="class2" value="si539">
     SI539 - App Engine<br>
 <input type="checkbox" name="class3">
     SI543 - Java<br>    </p>
```

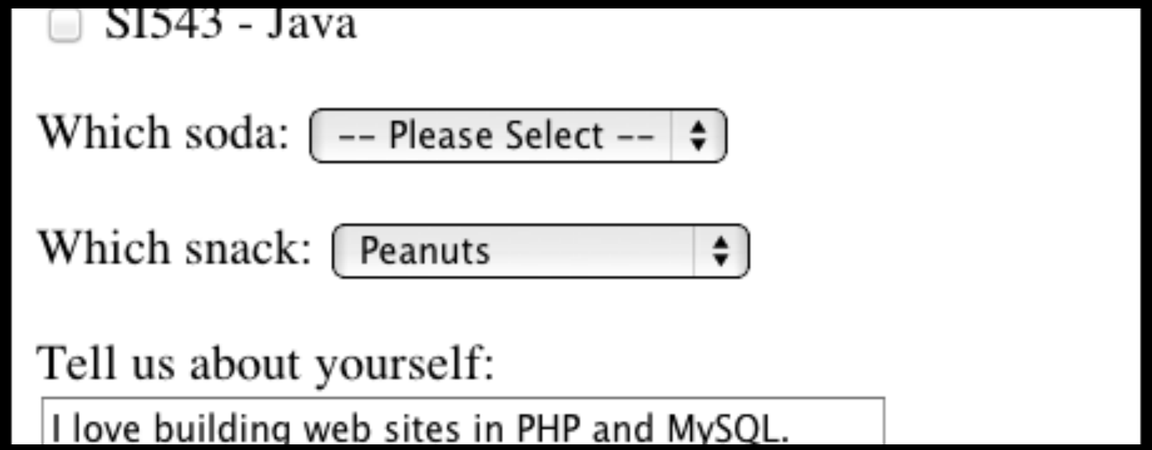
```
$_POST:
Array(
   ...
     [when] => pm
     [class1] => si502
     [soda] => 0
   ...
)
```




```
$_POST:
Array(
   ...
     [when] => pm
     [class3] => on
     [soda] => 0
   ...
)
```

more.php

```
<p><label for="inp06">Which soda:
 <select name="soda" id="inp06">
   <option value="0">-- Please Select --</option>
   <option value="1">Coke</option>
   <option value="2">Pepsi</option>
   <option value="3">Mountain Dew</option>
   <option value="4">Orange Juice</option>
   <option value="5">Lemonade</option>
 </select>
</p>
```

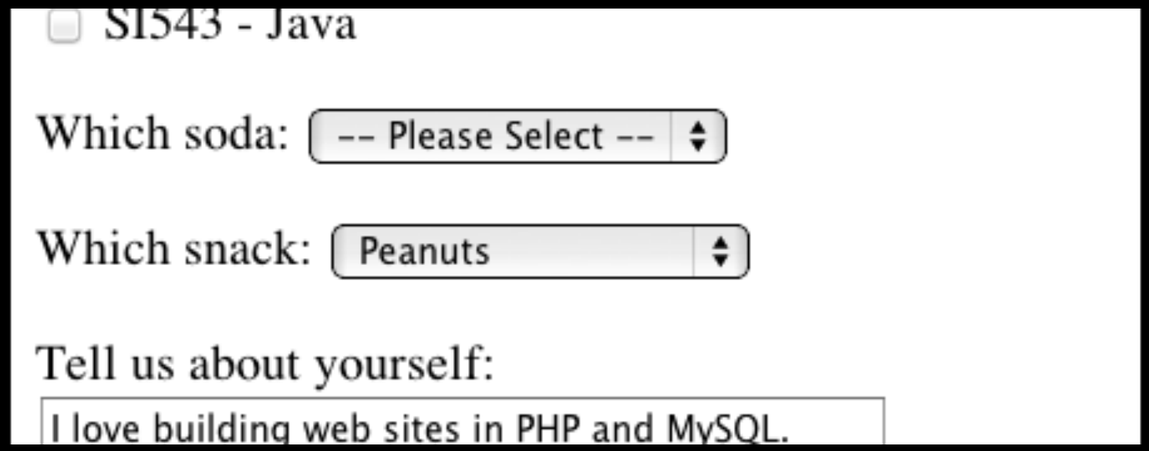



```
$_POST:
Array(
   ...
     [class] => si502
     [soda] => 0
     [snack] => peanuts
   ...
)
```

The values can be any string, but numbers are used quite often.

more.php

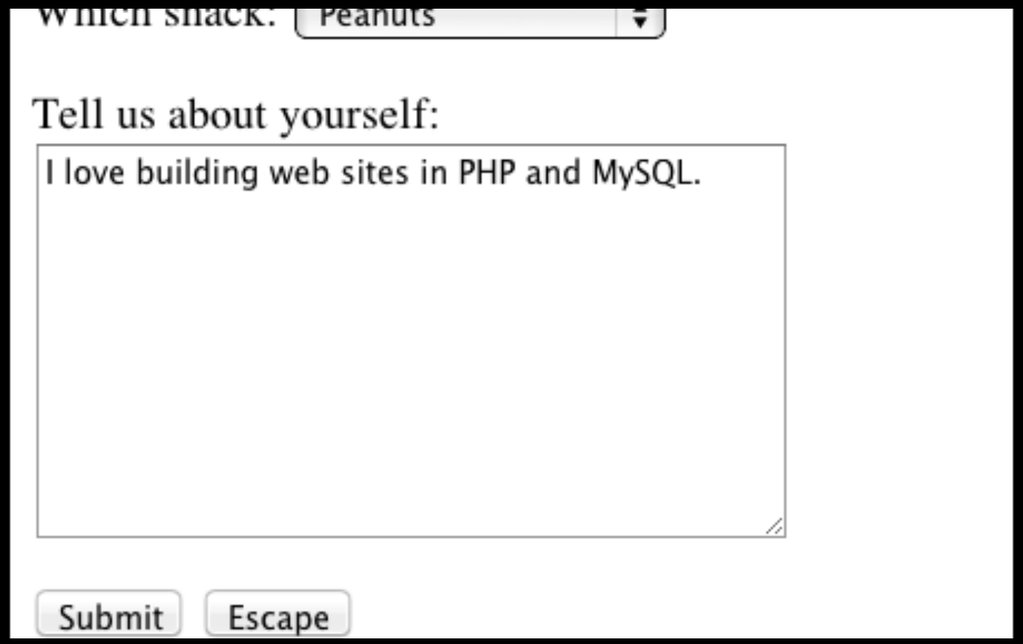
```
<p><label for="inp07">Which snack:
  <select name="snack" id="inp07">
    <option value="">-- Please Select --</option>
    <option value="chips">Chips</option>
    <option value="peanuts" selected>Peanuts</option>
    <option value="cookie">Cookie</option>
  </select>
</p>
```




```
$_POST:
Array(
    ...
      [class] => si502
      [soda] => 0
      [snack] => peanuts
    ...
)
```

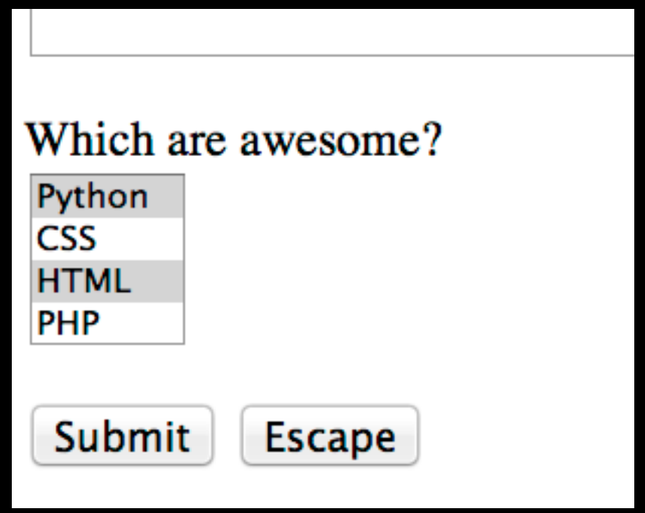
more.php

```
<p><label for="inp08">Tell us about yourself:<br/>
  <textarea rows="10" cols="40" id="inp08" name="about">
    I love building web sites in PHP and MySQL.
  </textarea>
</p>
```




```
$_POST:
Array(
   ...
      [about] => I love
building web sites in
PHP and MySQL.
      [dopost] => Submit
   ...
)
```

```
<p><label for="inp09">Which are awesome?<br/>
<select multiple="multiple" name="code[]" id="inp09">
  <option value="python">Python</option>
  <option value="css">CSS</option>
  <option value="html">HTML</option>
  <option value="php">PHP</option>
</select>
```

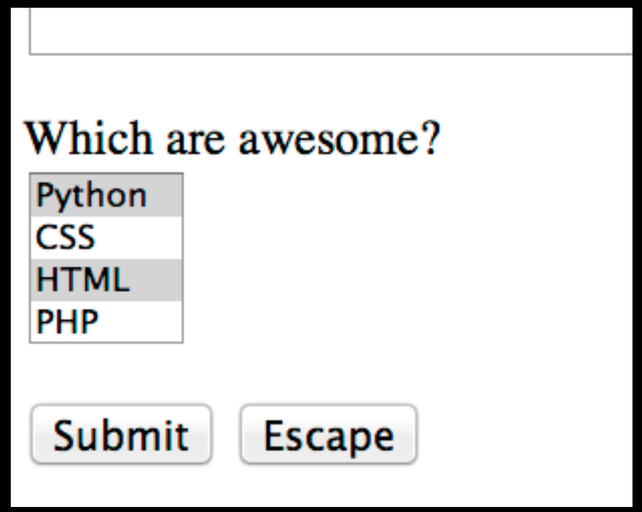


more.php

```
$_POST:
Array(
   ...
      [code] => Array
          (
              [0] => css
              [1] => html
          )
      [dopost] => Submit
   ...
)
```

more.php

```
<p>
<input type="submit" name="dopost" value="Submit"/>
<input type="button"
  onclick="location.href='http://www.wa4e.com/'; return false;"
  value="Escape">
</p>
```




```
$_POST:
Array(
   ...
     [dopost] => Submit
   ...
)
```

On submit input types, the text is both in the UI and in $_POST so we tend to look for the key, not the value.

HTML 5 Input Types

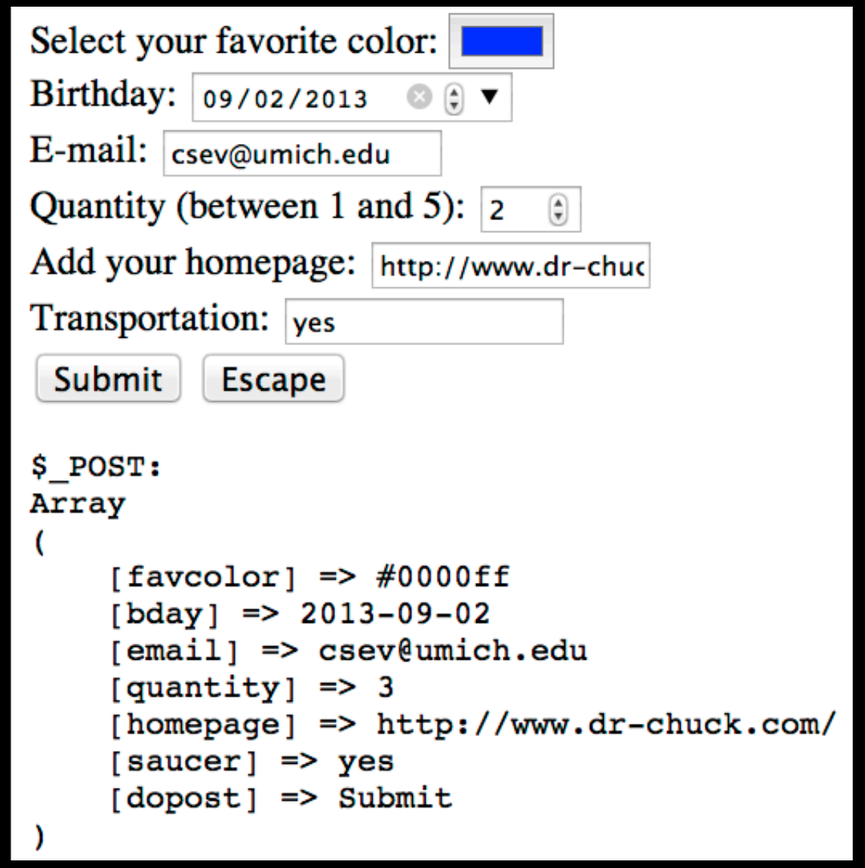
HTML5 Input Types

- HTML5 defines new input types
- Not all browsers support all input types
- They fall back to type="text"
- http://www.w3schools.com/html/html5_form_input_types.asp

```
Select your favorite color:
<input type="color" name="favcolor" value="#0000ff"><br/>
Birthday:
<input type="date" name="bday" value="2013-09-02"><br/>
E-mail:
<input type="email" name="email"><br/>
Quantity (between 1 and 5):
<input type="number" name="quantity"
   min="1" max="5"><br/>
Add your homepage:
<input type="url" name="homepage"><br>
Transportation:
<input type="flying" name="saucer"><br>
```

Validation happens when you press submit.

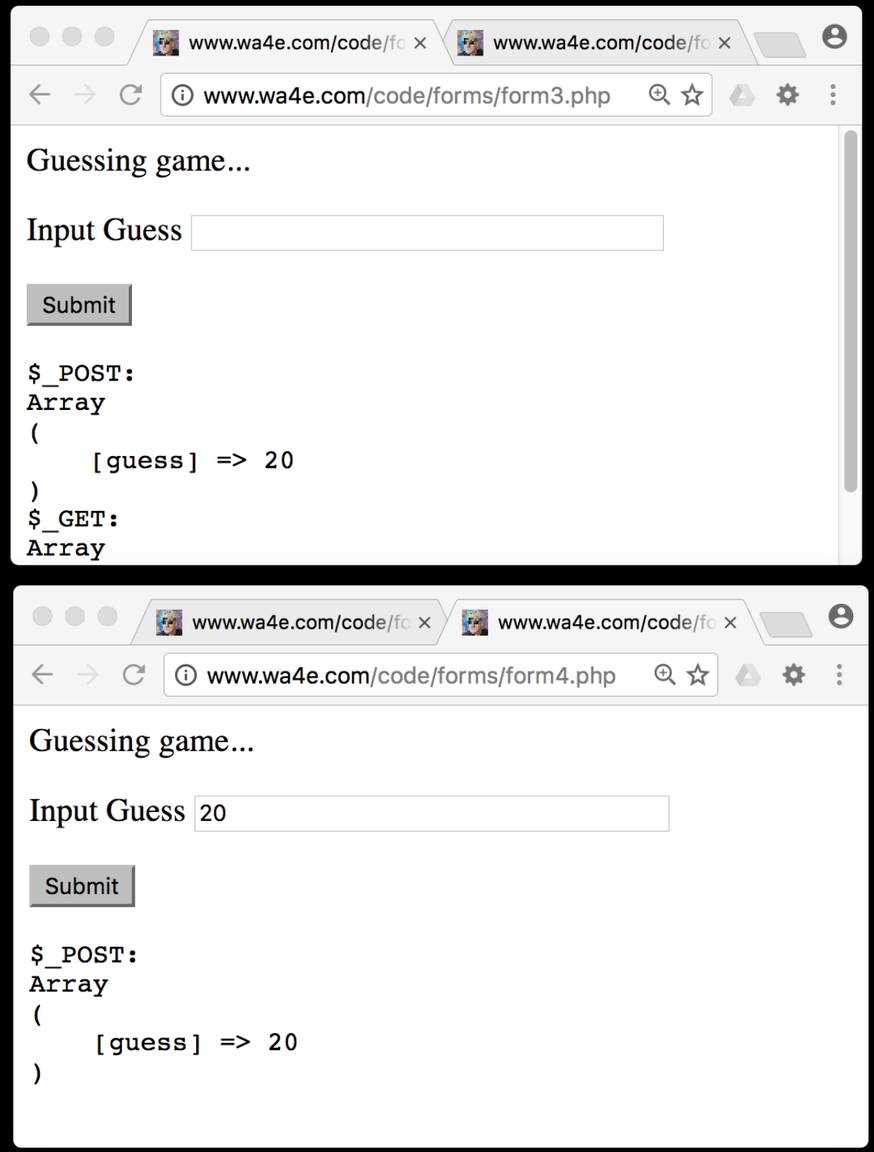
http://www.wa4e.com/code/forms/html5.php



Processing Form Data and HTML Injection

Persisting Form Data

- When we submit forms and there is an error, we just expect that the data will remain in the form when the page is redisplayed.
- The application needs to make sure to put the previous values back into the form.



```
<?php
    $oldguess = isset($_POST['guess']) ? $_POST['guess'] : '';
?>
<p>Guessing game...</p>
<form method="post">
    <p><label for="guess">Input Guess</label>
    <input type="text" name="guess" id="guess"
      size="40" value="<?= $oldguess ?>"/></p>
    <input type="submit"/>
</form>
```

"Persisting" Form Data Across Requests

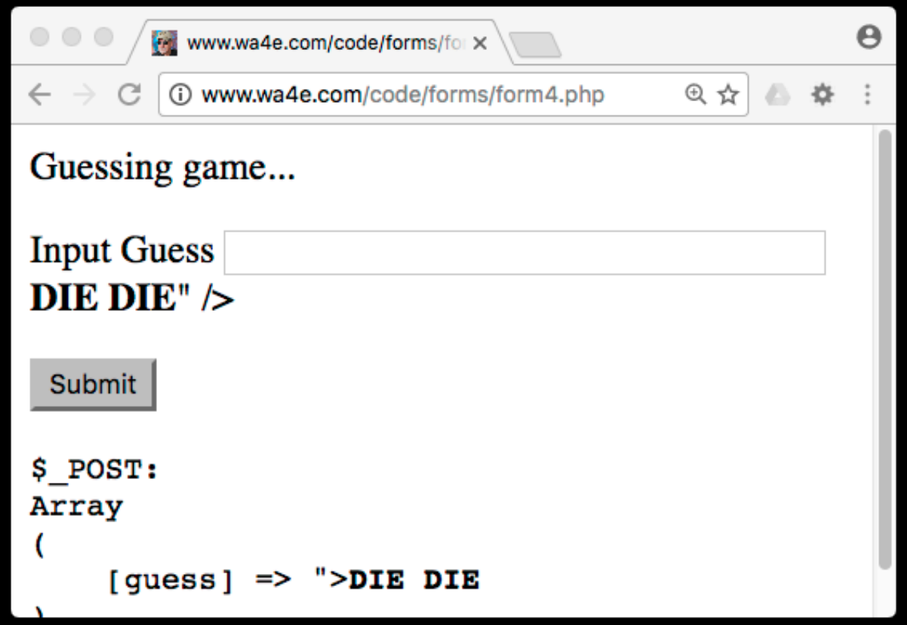
form4.php

```
<?= $oldguess ?>
<?php echo($oldguess); ?>
```

Review: Ternary Operation

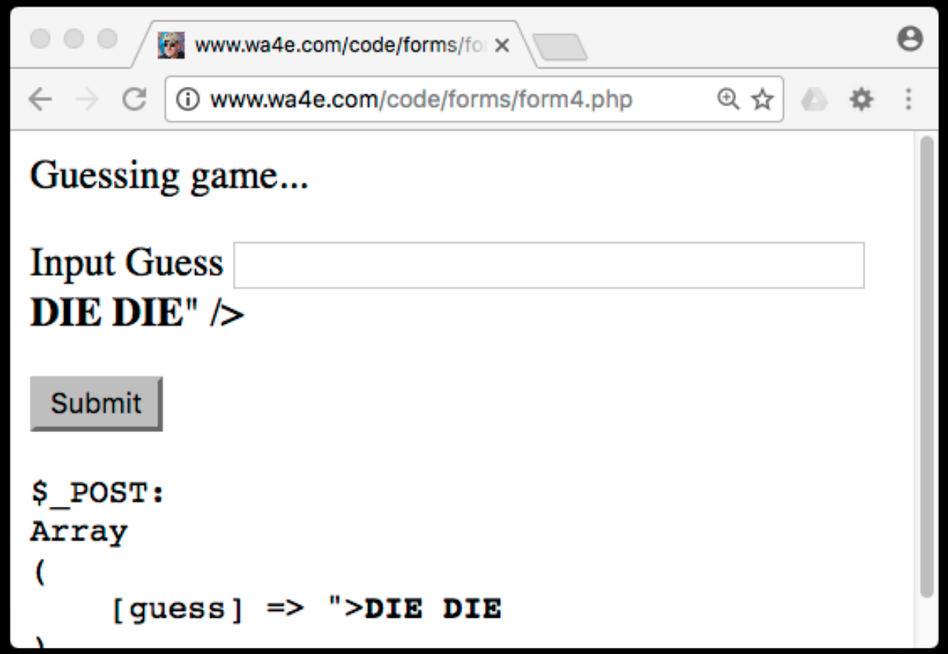
Hygiene Alert!

What happens when we use an HTML character in a form field value?



form4.php

```
<form method="post">
    <p><label for="guess">Input Guess</label>
    <input type="text" name="guess" id="guess
      size="40" "value=""><b>DIE DIE</b>" /></p>
    <input type="submit"/>
</form>
```

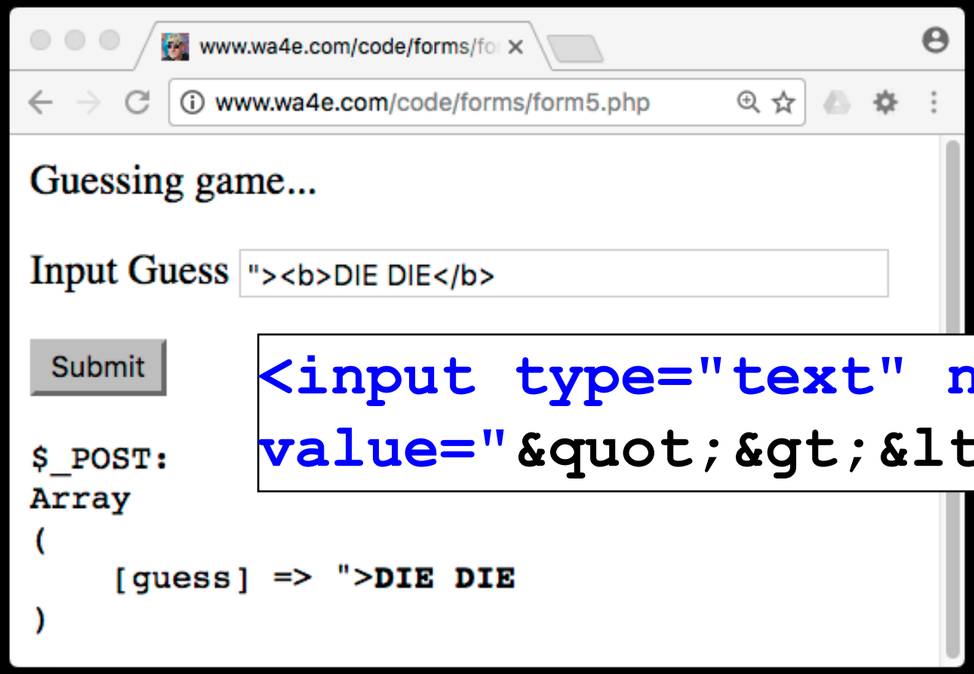


To The Rescue: htmlentities()

```
<form method="post">
  <p><label for="guess">Input Guess</label>
  <input type="text" name="guess" id="guess"
   size="40" value="<?= htmlentities($oldguess) ?>"/></p>
   <input type="submit"/>
</form>
```

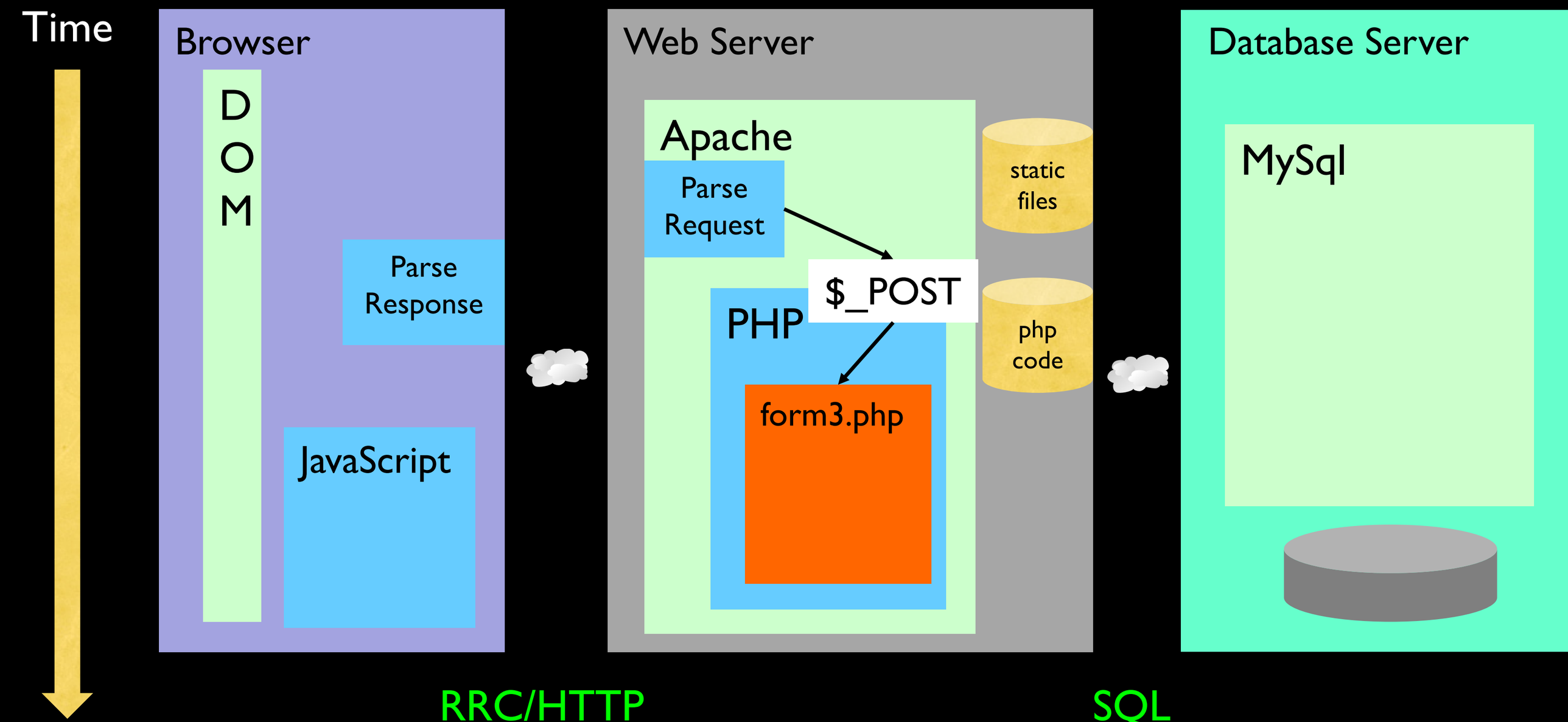
form5.php

```
<form method="post">
  <p><label for="guess">Input Guess</label>
  <input type="text" name="guess" id="guess"
   size="40" value="<?= htmlentities($oldguess) ?>"/></p>
   <input type="submit"/>
</form>
```




```
<input type="text" name="guess" id="guess"
value="&amp;quot;&amp;gt;&amp;lt;b&amp;gt;DIE DIE&amp;lt;/b&amp;gt;"   /></p>
```

Guessing Game


Time
Browser
DOM
Parse Response
JavaScript
Web Server
Apache
Parse Request
$_POST
PHP
form3.php
static files
php code
Database Server
MySql
RRC/HTTP
SQL

Incoming Data Validation

Making sure all user data is present and the correct format before proceeding

- Non-empty strlen($var) > 0
- A number is_numeric($var)
- An email address strpos($var, '@') > 0
- Or filter_var($var, FILTER_VALIDATE_EMAIL) !== false
- ....

http://www.wa4e.com/code/forms/guess.php?guess=7

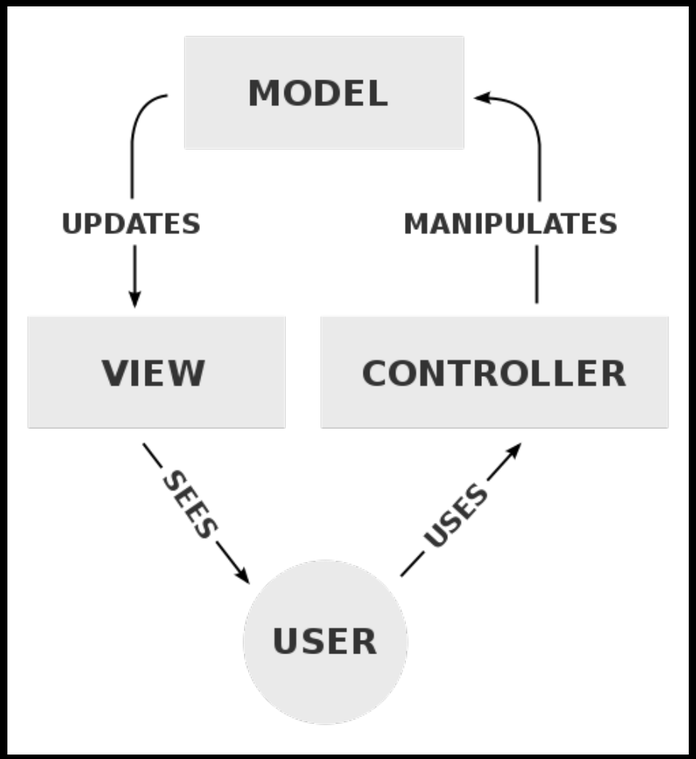
```
<html>
<head>
<title>Guessing Game for Charles Severance</title>
</head>
<body>
<h1>Welcome to my guessing game</h1>
<p>
<?php
  if ( ! isset($_GET['guess']) ) {
    echo("Missing guess parameter");
  } else if ( strlen($_GET['guess']) < 1 ) {
    echo("Your guess is too short");
  } else if ( ! is_numeric($_GET['guess']) ) {
    echo("Your guess is not a number");
  } else if ( $_GET['guess'] < 42 ) {
    echo("Your guess is too low");
  } else if ( $_GET['guess'] > 42 ) {
    echo("Your guess is too high");
  } else {
    echo("Congratulations - You are right");
  }
?>
</p>
</body>
</html>
```

http://www.wa4e.com/code/forms/guess.php?guess=200

Model View Controller (MVC)

Model-View-Controller

- A model that defines the elements of a web application and how they interact
- View – Produces output
- Model – Handles data
- Controller – Orchestration / Routing



https://en.wikipedia.org/wiki/Model-view-controller

Pattern: Processing POST Data

- Many patterns for handling POST data
- No “rules”, just “suggestions”

What about frameworks?

```php
<?php
    $guess = '';
    $message = false;
    if ( isset($_POST['guess']) ) {
        // Trick for integer / numeric parameters
        $guess = $_POST['guess'] + 0;
        if ( $guess == 42 ) {
            $message = "Great job!";
        } else if ( $guess < 42 ) {
            $message = "Too low";
        } else {
            $message = "Too high...";
        }
    }
?>
```

```php
<html>
<head>
    <title>A Guessing game</title>
</head>
<body style="font-family: sans-serif;">
<p>Guessing game...</p>
<?php
   if ( $message !== false ) {
       echo("<p>$message</p>\n");
    }
?>
<form method="post">
   <p><label for="guess">Input Guess</label>
   <input type="text" name="guess" id="guess" size="40"
 <?php   echo 'value="' . htmlentities($guess) . '"';
?>
   /></p>
   <input type="submit"/>
</form>
</body>
```

Completely process incoming data (if any) - produce no output

---

Produce the page output

guess_mvc.php

Model

View

Context

Controller

```
<?php
    $oldguess = '';
    $message = false;
    if ( isset($_POST['guess']) ) {
        // Trick for integer / numeric parameters
        $oldguess = $_POST['guess'] + 0;
        if ( $oldguess == 42 ) {
            $message = "Great job!";
        } else if ( $oldguess < 42 ) {
            $message = "Too low";
        } else {
            $message = "Too high...";
        }
    }
?>
<html>
<head>
    <title>A Guessing game</title>
</head>
<body style="font-family: sans-serif;">
<p>Guessing game...</p>
<?php
   if ( $message !== false ) {
        echo("<p>$message</p>\n");
    }
?>
<form method="post">
   <p><label for="guess">Input Guess</label>
   <input type="text" name="guess" id="guess" size="40"
     value="<?= htmlentities($oldguess) ?>"/></p>
   <input type="submit"/>
</form>
</body>
```

guess_mvc.php

No HTML

No Database

```
<?php
    $oldguess = '';
    $message = false;
    if ( isset($_POST['guess']) ) {
        // Trick for integer / numeric parameters
        $oldguess = $_POST['guess'] + 0;
        if ( $oldguess == 42 ) {
            $message = "Great job!";
        } else if ( $oldguess < 42 ) {
            $message = "Too low";
        } else {
            $message = "Too high...";
        }
    }
?>
<html>
<head>
    <title>A Guessing game</title>
</head>
<body style="font-family: sans-serif;">
<p>Guessing game...</p>
<?php
   if ( $message !== false ) {
        echo("<p>$message</p>\n");
    }
?>
<form method="post">
   <p><label for="guess">Input Guess</label>
   <input type="text" name="guess" id="guess" size="40"
     value="<?= htmlentities($oldguess) ?>"/></p>
   <input type="submit"/>
</form>
</body>
```

Context

Controller

guess_mvc.php

```
<?php
    $guess = '';
    $message = false;
    if ( isset($_POST['guess']) ) {
        // Trick for integer / numeric parameters
        $guess = $_POST['guess'] + 0;
        if ( $guess == 42 ) {
            $message = "Great job!";
        } else if ( $guess < 42 ) {
            $message = "Too low";
        } else {
            $message = "Too high...";
        }
    }
?>
<html>
<head>
    <title>A Guessing game</title>
</head>
<body style="font-family: sans-serif;">
<p>Guessing game...</p>
<?php
   if ( $message !== false ) {
        echo("<p>$message</p>\n");
    }
?>
<form method="post">
   <p><label for="guess">Input Guess</label>
   <input type="text" name="guess" id="guess" size="40"
     value="<?= htmlentities($oldguess) ?></p>
   <input type="submit"/>   <input type="submit"/>
</form>
</body>
```

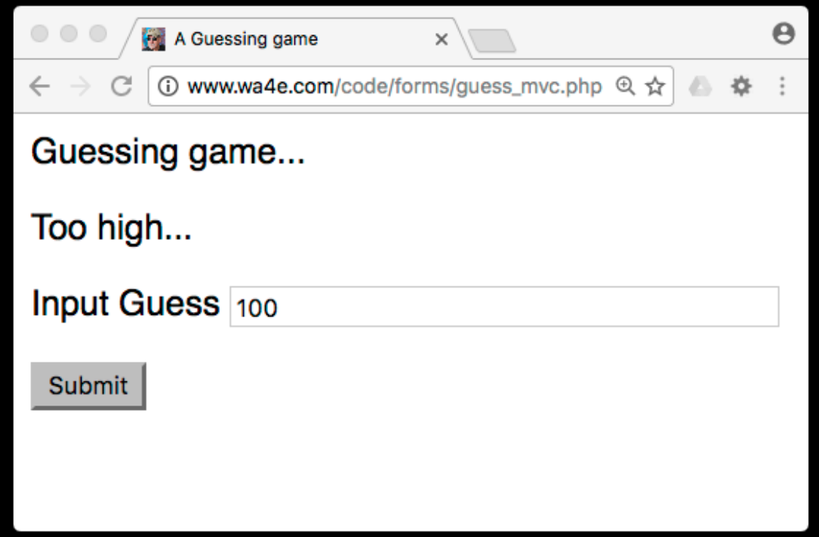
```
<?php
    $oldguess = '';
    $message = false;
    if ( isset($_POST['guess']) ) {
        // Nifty trick
        $oldguess = $_POST['guess'] + 0;
        if ( $oldguess == 42 ) {
            $message = "Great job!";
        } else if ( $oldguess < 42 ) {
            $message = "Too low";
        } else {
            $message = "Too high...";
        }
    }
?>
<html> ...
```

guess_mvc.php

```php
<?php
    $guess = '';
    $message = false;
    if ( isset($_POST['guess']) ) {
        // Trick for integer / numeric parameters
        $guess = $_POST['guess'] + 0;
        if ( $guess == 42 ) {
            $message = "Great job!";
        } else if ( $guess < 42 ) {
            $message = "Too low";
        } else {
            $message = "Too high...";
        }
    }
?>
<html>
<head>
    <title>A Guessing game</title>
</head>
<body style="font-family: sans-serif;">
<p>Guessing game...</p>
<?php
   if ( $message !== false ) {
        echo("<p>$message</p>\n");
    }
?>
<form method="post">
   <p><label for="guess">Input Guess</label>
   <input type="text" name="guess" id="guess" size="40"
     value="<?= htmlentities($oldguess) ?></p>
   <input type="submit"/>   <input type="submit"/>
</form>
</body>
```

```php
...
?>
<html>
<head>
    <title>A Guessing game</title>
</head>
<body style="font-family: sans-serif;">
<p>Guessing game...</p>
<?php
   if ( $message !== false )  {
        echo("<p>$message</p>\n");
    }
?>
<form method="post">
 <p><label for="guess">Input Guess</label>
   <input type="text" name="guess" id="guess"
size="40"
     value="<?= htmlentities($oldguess) ?>"></p>
   <input type="submit"/>
</form>
</body>
```

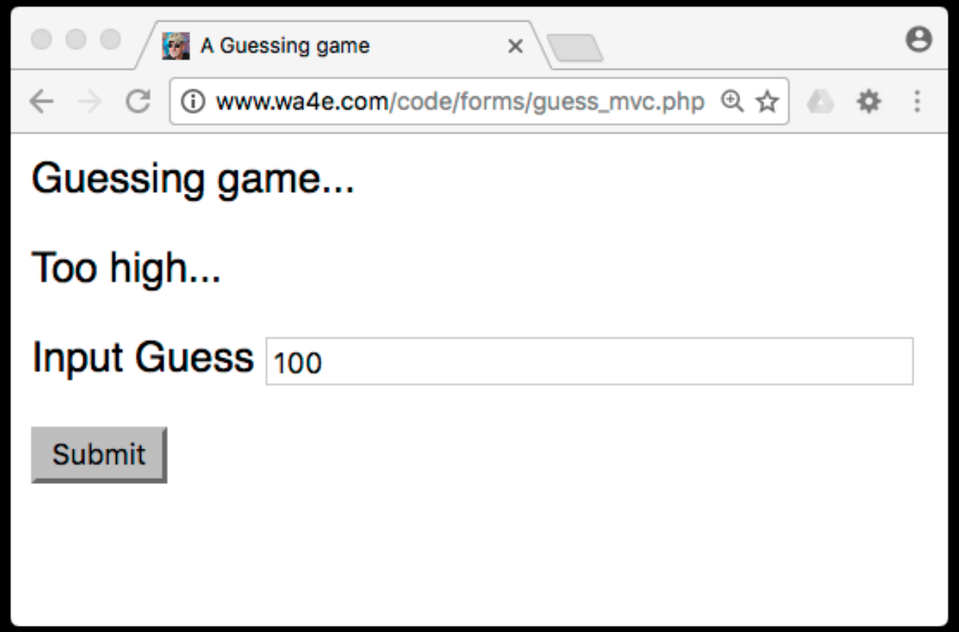
```php
<?php
    $oldguess = '';
    $message = false;
    if ( isset($_POST['guess']) ) {
        // Nifty trick
        $oldguess = $_POST['guess'] + 0;
        if ( $oldguess == 42 ) {
            $message = "Great job!";
        } else if ( $oldguess < 42 ) {
            $message = "Too low";
        } else {
            $message = "Too high...";
        }
    }
?>
<html> ...
```



guess_mvc.php

Note: This code is a little sloppy in terms of its data validation.

```
<html>
<head>
    <title>A Guessing game</title>
</head>
<body style="font-family: sans-serif;">
<p>Guessing game...</p>
<?php
   if ( $message !== false )  {
        echo("<p>$message</p>\n");
    }
?>
<form method="post">
 <p><label for="guess">Input Guess</label>
   <input type="text" name="guess" id="guess" size="40"
     value="<?= htmlentities($oldguess) ?>"></p>
   <input type="submit"/>
</form>
</body>
```



guess_mvc.php

Summary

- Forms, $_GET and $_POST
- Form fields
- New form fields in HTML5
- Sanitizing HTML
- Data Validation
- Model-View-Controller

Acknowledgements / Contributions

Initial Development: Charles Severance, University of Michigan School of Information

Insert new Contributors and Translators here including names and dates

Continue new Contributors and Translators here