PHP Programming Assignment 2

1. On what basis are elements in forms mapped into the $_POST, $_GET, or $_REQUEST super global arrays?

   ANSWER:

   The HTML attribute “name” is used to map elements into the superglobals. For example, <input type='text' id='foo' name='bar' /> would appear in $_REQUEST using ‘bar’ as the index.

2. What is distinct about a hidden field as opposed to any other text field? How does data get into a hidden field? What are they commonly used for?

   ANSWER:

   A hidden field in a form is not displayed to the user. The data gets into the hidden field programmatically from the server side using the “value” attribute of an input tag. They are commonly used to pass around the IDs of objects that the receiving page needs to look up in order to carry out its job. For example, if you are working on an “edit” action for an entity, you would display the human-relevant data in a form, but have a hidden field for the ID so that the receiving script can update the database entity using the ID.

3. Explain the difference between the ‘==’ and the ‘===’ operators. Given an example of two variables ($a and $b) such that $a == $b is true, but $a === $b is false.

   ANSWER:

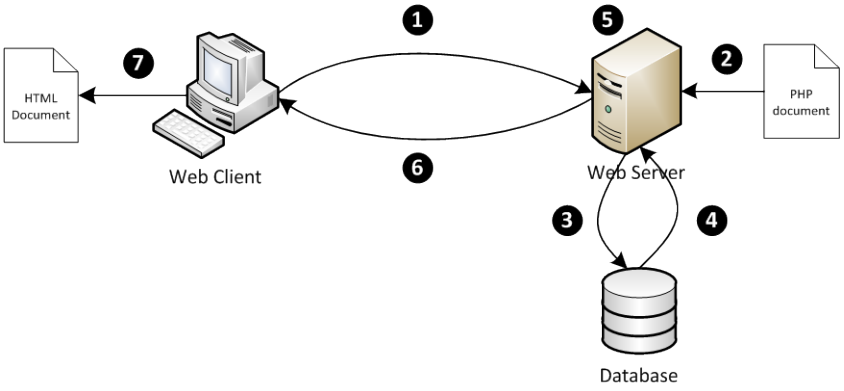
   The == operator employs type juggling to do the comparison. However, the === operator not only compares values, but also types (i.e. no juggling is permitted). For example, when $a=0 and $b='0', then $a==$b will be true, but $a===$b is false because the types don’t match.

1. In a web application, how is the execution of JavaScript fundamentally different than PHP?

   ANSWER:

   JavaScript is executed in the browser on the user’s machine. On the other hand, PHP executes on the server and produces the HTML and JavaScript that is sent to the user’s machine.

2. List and explain the steps in the request/response cycle in a PHP application that accesses a database.



ANSWER:

1) The browser sends an HTTP request as a URL to the server
2) The server loads the appropriate PHP file and begins to interpret it
3) The PHP file contains code that queries the database
4) The database responds with a result set
5) The contents of the result set are inserted into the HTML document
6) An HTTP reply containing the HTML document is sent back to the browser
7) The browser renders the HTML document

3. How are the GET and POST methods of a web form different in how data is transmitted to the server?

ANSWER:

In a GET method, parameters from a web form are encoded as part of the URL (for example http://mycompany.com/app/somePage.php?name=George&city=Columbus indicates that two parameters – name and city – are given). In a POST method, the parameters are put inside the body of the HTTP request and are not visible in URL bar of the browser.

4. How are single-quote strings different from double-quote strings in PHP?

ANSWER:

Single-quote strings do not perform variable interpolation or handle escape sequences. Double-quote strings are parsed and handle variable interpolation and escape sequences.

5. **[10 points]** Write a solution to the "FizzBuzz" problem. The FizzBuzz problem is a classic programming interview problem in which a range of numbers is examined to locate those numbers divisible by 3 and 5. If the number is divisible by 3, then print "Fizz." If the number is divisible by 5 then print "Buzz." If it is divisible by both, then print "FizzBuzz." Otherwise, just print the number.

You should have an `index.html` file and a `fizzbuzz.php` file. The `index.html` has a form that submits data to the `fizzbuzz.php` file. You may optionally have a style file as well, referenced from both. If the user does not submit a start and stop number for the range (inclusive), then assume that start is 1 and stop is 100. Otherwise, use the specified range.

You may use either GET or POST for your form processing. Screen shots of one sample solution are below. You should try to get your solution to look as close to this output as possible.

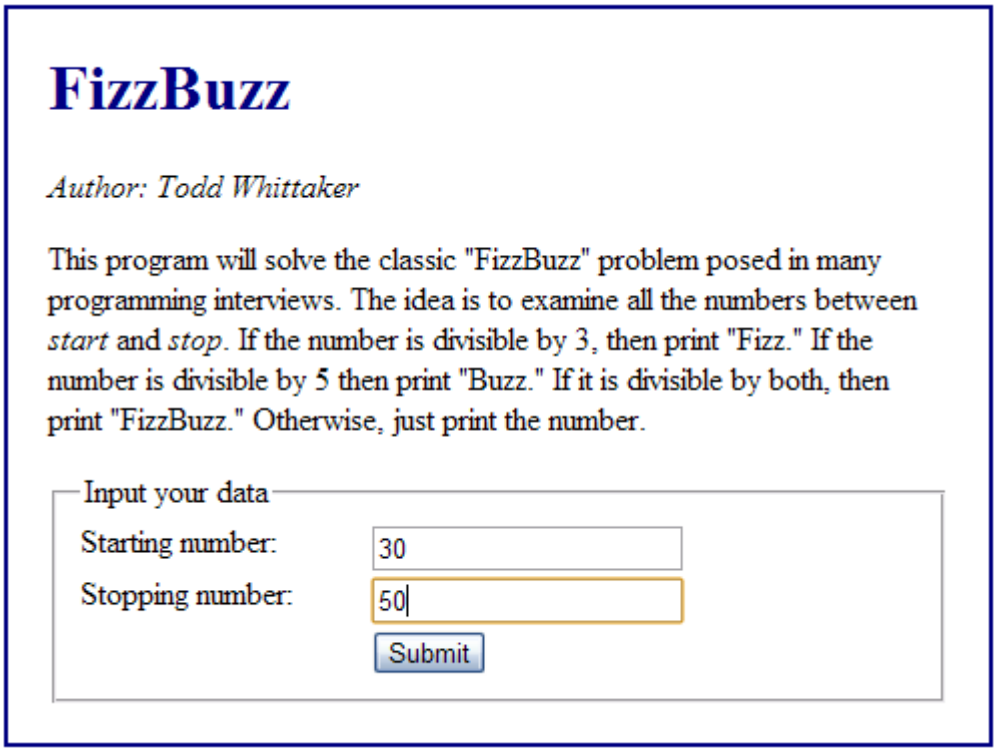


Figure 1: The index.html file with sample inputs.

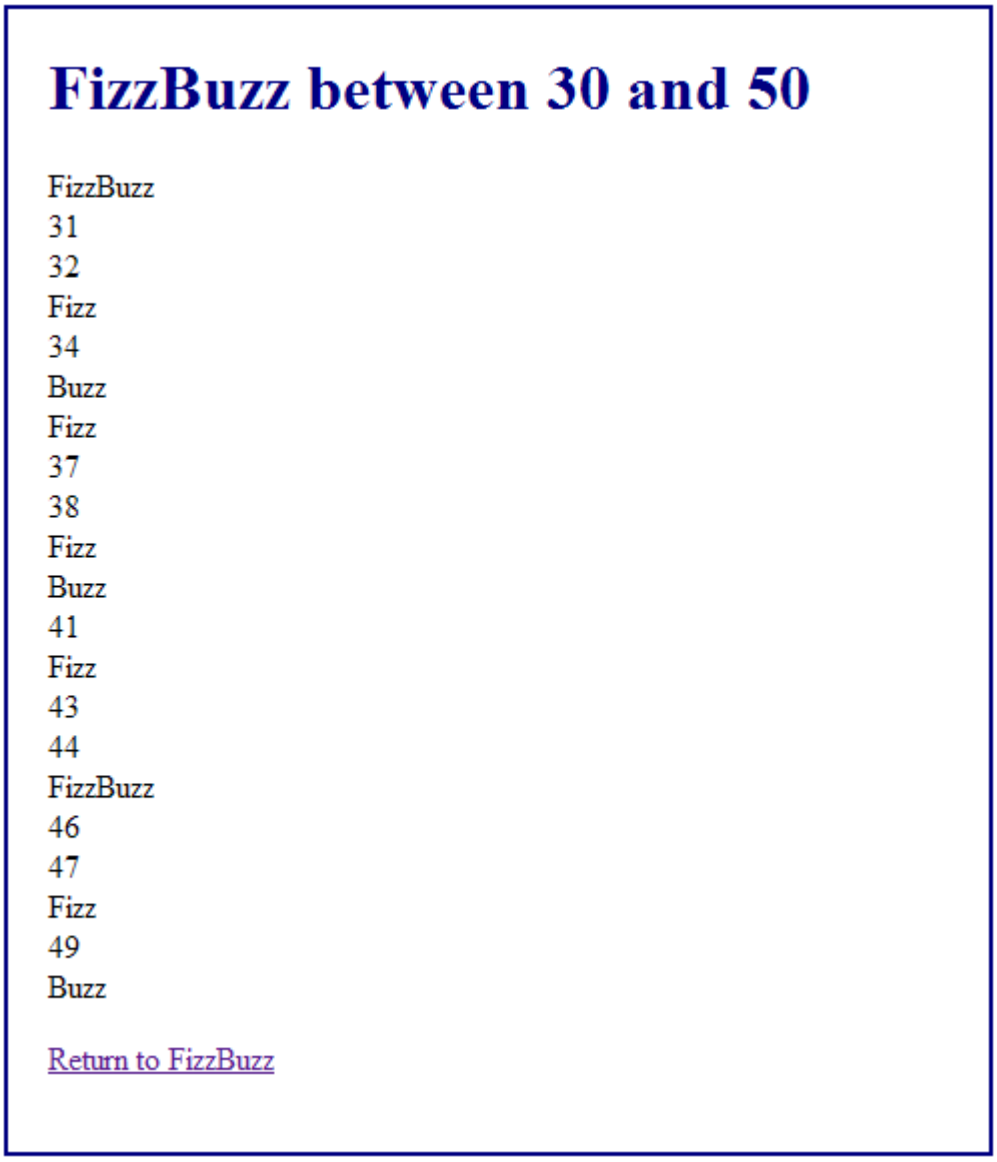


Figure 2: Output from the fizzbuzz.php file.

ANSWER:

Index.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<!--
    Author  : Todd A. Whittaker
    Version : January 15, 2012
    Description: This file is the main page that lets someone enter
        numbers to solve the FizzBuzz Problem. The idea is to
        examine all the numbers between start and stop (inclusive)
        If the number is divisible by 3, then print "Fizz."  If the
        number is divisible by 5 then print "Buzz."  If it is
```

```
        divisible by both, then print "FizzBuzz."  Otherwise, just
        print the number.
-->
<html>
    <head>
        <title>FizzBuzz</title>
        <link rel="stylesheet" href="style.css" />
    </head>
    <body>
        <div id="content">
            <h1>FizzBuzz</h1>
            <p>
                <em>Author: Todd Whittaker</em>
            </p>
            <p>
                This program will solve the classic "FizzBuzz" problem
                posed in many programming interviews.  The idea is to
                examine all the numbers between <em>start</em> and
                <em>stop</em> (inclusive). If the number is divisible
                by 3, then print "Fizz."  If the number is divisible
                by 5 then print "Buzz."  If it is divisible by both,
                then print "FizzBuzz."  Otherwise, just print the
                number.
            </p>
            <form action="fizzbuzz.php" method="post">
                <fieldset>
                    <legend>
                        Input your data
                    </legend>
                    <label for="start">Starting number:</label>
                    <input type="text" id="start" name="start" />
                    <br />
                    <label for="stop">Stopping number:</label>
                    <input type="text" id="stop" name="stop" />
                    <br />
                    <label> </label>
                    <input type="submit" value="Submit" />
                    <br />
                </fieldset>
            </form>
        </div>
    </body>
</html>
```

Style.css

```css
/**
 * Author  : Todd Whittaker
 * Version : January 15, 2012
 */
#content {
      width: 450px;
      margin: 0 auto;
      padding: 0px 20px 20px;
      background: white;
      border: 2px solid navy;
}
h1 {
      color: navy;
}
label {
      width: 8em;
      padding-right: 1em;
      float: left;
}
```

Fizzbuzz.php

```php
<?php
/**
 * Author  : Todd Whittaker
 * Version : January 15, 2012
 * Description: This file is the solution to the FizzBuzz Problem.
 *     The idea is to examine all the numbers between start and
 *     stop (inclusive) If the number is divisible by 3, then
 *     print "Fizz."  If the number is divisible by 5 then print
 *     "Buzz."  If it is divisible by both, then print "FizzBuzz."
 *     Otherwise, just print the number.
 */

/**
 * Returns a value in a safe way from the _GET or _POST arrays.
 * Values are escaped using htmlentities, and attempts to retrieve
 * a non-existent key will result in returning a default value
 * specified as a parameter.
 * @param $key the key index to use in referencing _GET or _POST
 * @param $default the value to return in the event that the key
 *     doesn't exist in either array, or is the empty string.
 * @return the parameter at the index or the default if not found
 */
function safeParam($key, $default = null) {
    if (isset($_POST[$key]) && $_POST[$key] != "") {
```

```
        return htmlentities($_POST[$key]);
    } else if (isset($_GET[$key]) && $_GET[$key] != "") {
        return htmlentities($_GET[$key]);
    } else {
        return $default;
    }
}

/**
 * Produce a string with the answer to FizzBuzz.
 * @param $start The starting number
 * @param $stop The ending number
 * @return a big string with the HTML solution to FizzBuzz
 */
function fizzbuzz($start, $stop) {
    $result = "";

    # look at each number between start and stop
    for ($i = $start; $i <= $stop; $i++) {

        # is it divisible by 3 or 5?
        if (($i % 3 == 0) || ($i % 5 == 0)) {
            if ($i % 3 == 0) {
                $result = $result . "Fizz";
            }
            if ($i % 5 == 0) {
                $result = $result . "Buzz";
            }
        } else {
            # nope.
            $result = $result . $i;
        }
        $result = $result . "<br />\n";
    }
    return $result;
}

/**
 * Makes sure that the parameter given is both numeric and 1 or
greater.
 * @param $num the number to test
 * @param $default what to return if the test fails.
 * @return either the number given, or 1 if it fails the positive test
 */
function ensurePositive($num, $default=1) {
    if (!is_numeric($num) || $num <= 0) {
```

```
        return $default;
    }
    return $num;
}

# do some rudimentary error checking
$start = ensurePositive(safeParam('start'), 1);
$stop = ensurePositive(safeParam('stop'), 100);

# swap if out of order
if ($stop < $start) {
    $temp = $stop;
    $stop = $start;
    $start = $temp;
}

# solve the problem
$output = fizzbuzz($start, $stop);
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
    <head>
        <title>FizzBuzz between <?php echo $start ?> and <?php echo
$stop ?></title>
        <link rel="stylesheet" href="style.css" />
    </head>
    <body>
        <div id="content">
            <h1>FizzBuzz between <?php echo $start ?> and <?php echo
$stop ?></h1>
            <p>
                <?php echo $output ?>
            </p>
            <p>
                <a href="index.html">Return to FizzBuzz</a>
            </p>
        </div>
    </body>
</html>
```