Creating and Validating Forms

Unit - IV

4.1 GUI Components

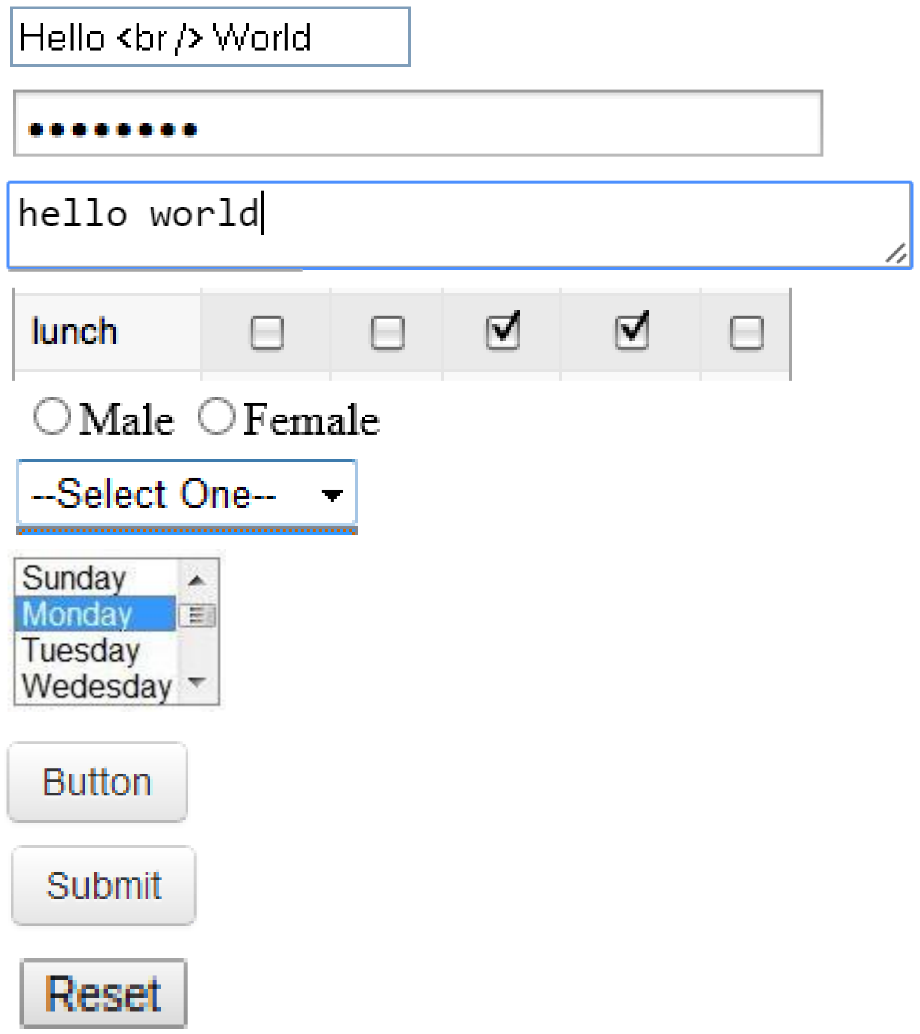
- HTML controls like textbox, textarea, check boxes, radio buttons etc enclosed in HTML form are used to collect data from user.

- When the user fills out the form and clicks submit button, the form data is sent for processing to a PHP file.

- There are two methods "*get*" and "*post*" commonly used to send data from HTML controls to PHP script on server.

4.1 GUI Components

- About 10 elements are commonly found within <form/> elements.
  - Text Box
  - Password
  - Text Area
  - Check Box
  - Radio Button
  - Drop Down List
  - List Box
  - Standard Button
  - Submit Button
  - Reset Button



4.1 GUI Components

- <form> tag is used to create an HTML form for user input.
- Name: Specifies the name of a form.
- Action: URL
  specifies where to send the form data when a form is submitted.
- Target: Specifies where to display the response that is received after submitting the form.
  _blank , _self , _parent , _top
- Method: Specifies the HTTP method to use when sending form data.
  Get :
  Post :

4.1 GUI Components

- Get:
  - It is the default method when submitting a form.
  - When Get method is used, the submitted form data will be visible in the web page address bar.
  - It appends form data into the URL in name=value pair separated by &.
  - The length of a URL is limited.
  - Never use Get to send sensitive data.

4.1 GUI Components

- Get:

GUI.html file

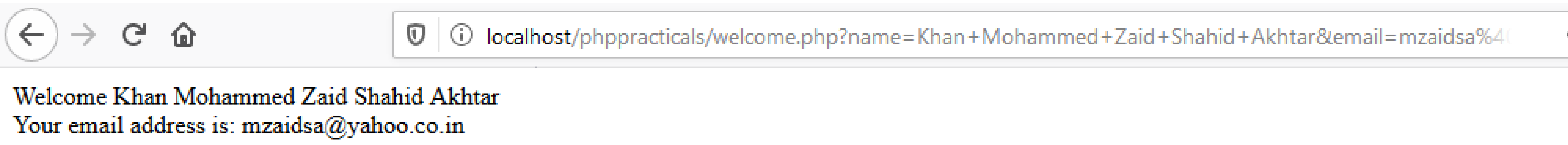
```
<html> <body>
	<form action="welcome.php" method="get">
	Name: <input type="text" name="name"><br>
	E-mail: <input type="text" name="email"><br>
	<input type="submit">
	</form>
</body> </html>
```

Welcome.php file

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

4.1 GUI Components

- Get:


localhost/phppracticals/welcome.php?name=Khan+Mohammed+Zaid+Shahid+Akhtar&email=mzaidsa%4

```
Welcome Khan Mohammed Zaid Shahid Akhtar
Your email address is: mzaidsa@yahoo.co.in
```

4.1 GUI Components

- Post:
  - Post method is used if the form contains personal or sensitive information.
  - The POST method does not display the submitted form data in the page address field.
  - POST has no size limitations, and can be used to send large amounts of data.
  - Form submissions with POST cannot be bookmarked.

4.1 GUI Components

- Post:

GUI.html file

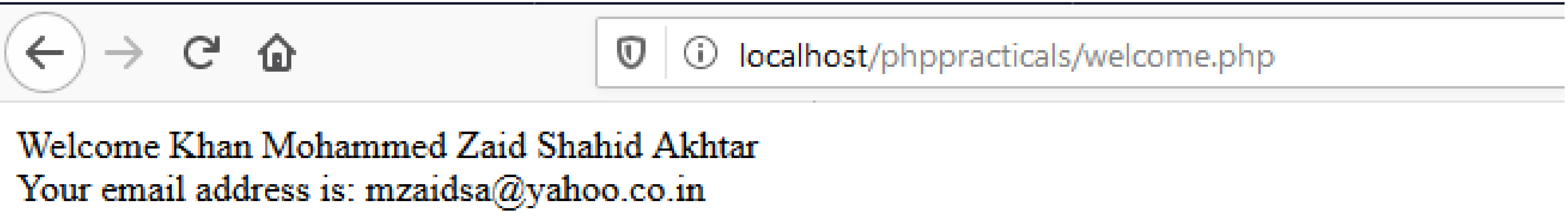
```
<html> <body>
	<form action="welcome.php"
  method="post">
	Name: <input type="text"
  name="name"><br>
	E-mail: <input type="text"
  name="email"><br>
	<input type="submit">
	</form>
</body> </html>
```

Welcome.php file

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

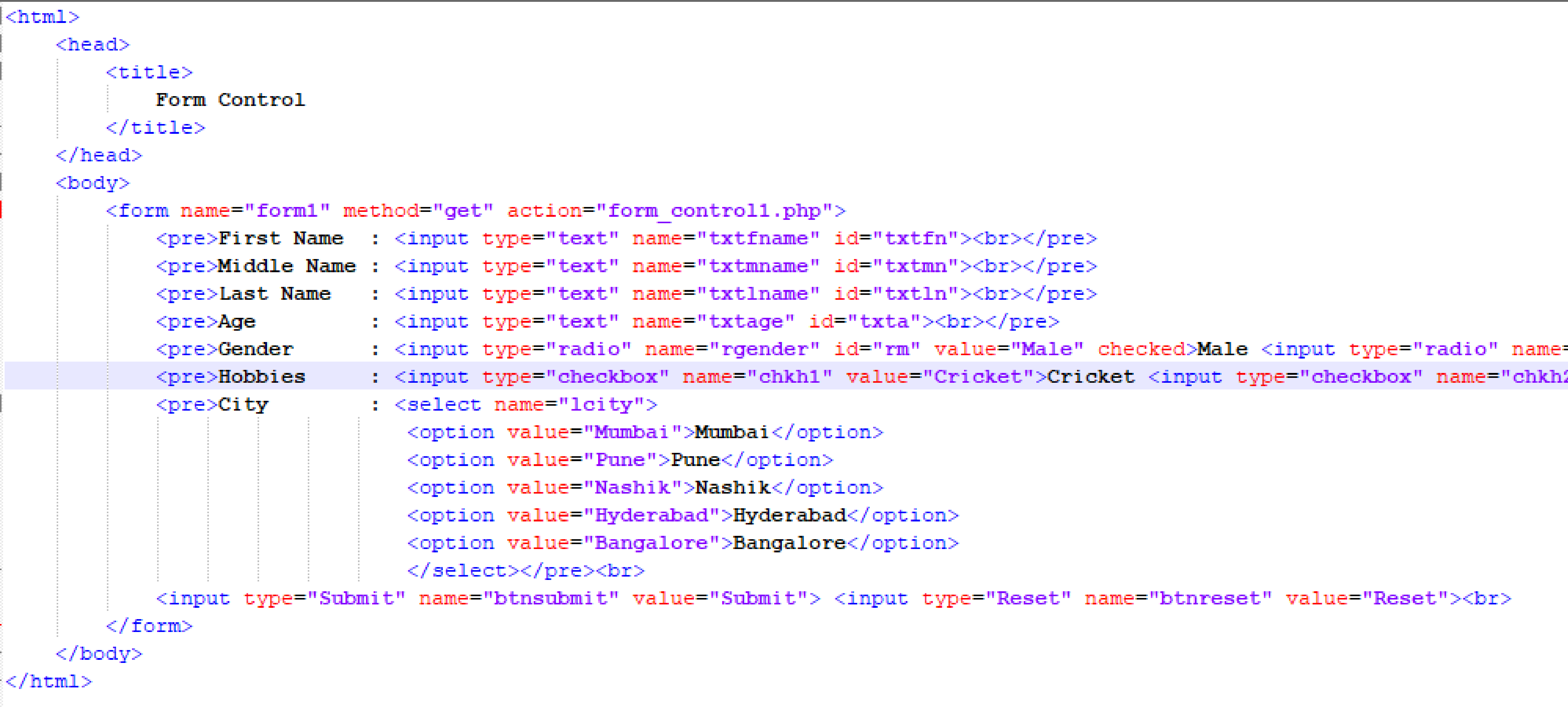
4.1 GUI Components

- Post:



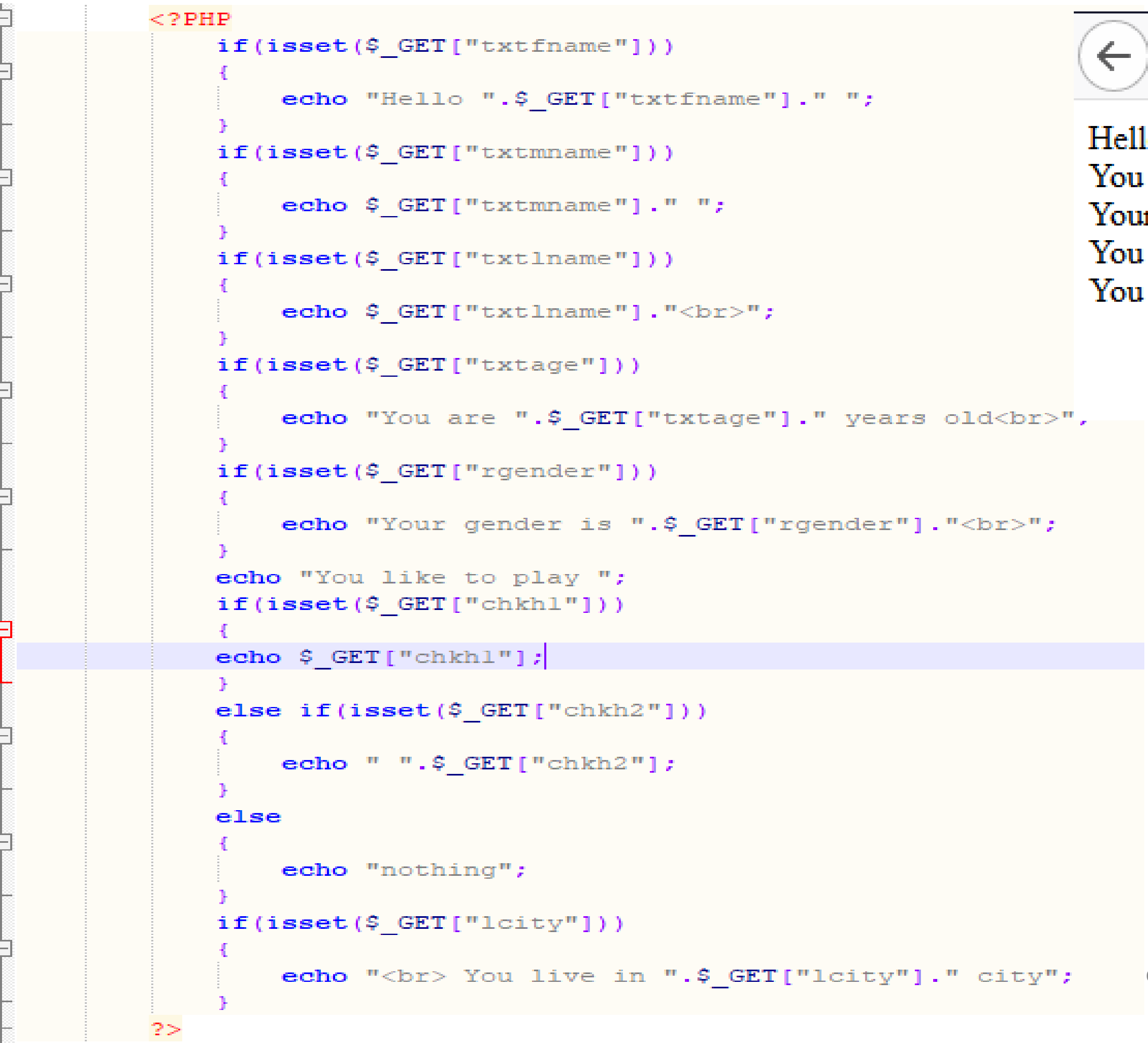
```
Welcome Khan Mohammed Zaid Shahid Akhtar
Your email address is: mzaidsa@yahoo.co.in
```

4.2 Form Controls

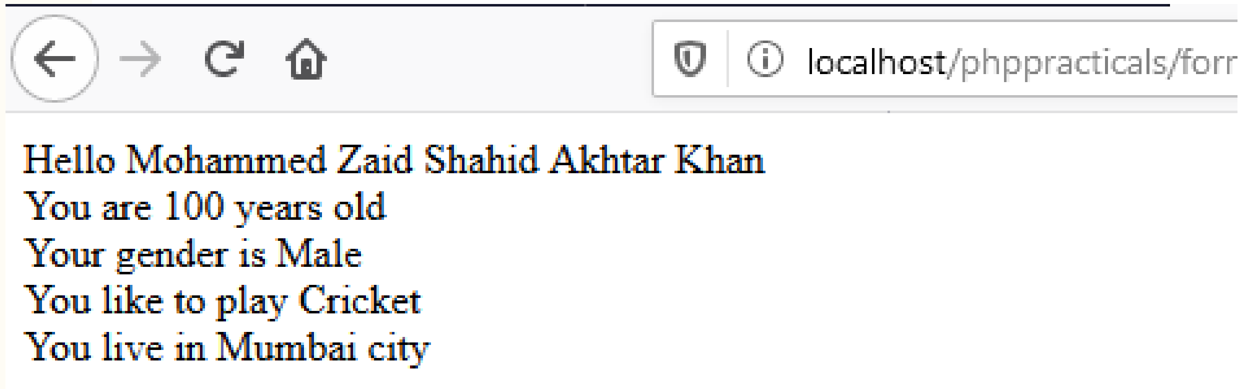


```
<html>
    <head>
        <title>
            Form Control
        </title>
    </head>
    <body>
        <form name="form1" method="get" action="form_control1.php">
            <pre>First Name  : <input type="text" name="txtfname" id="txtfn"><br></pre>
            <pre>Middle Name : <input type="text" name="txtmname" id="txtmn"><br></pre>
            <pre>Last Name   : <input type="text" name="txtlname" id="txtln"><br></pre>
            <pre>Age         : <input type="text" name="txtage" id="txta"><br></pre>
            <pre>Gender      : <input type="radio" name="rgender" id="rm" value="Male" checked>Male <input type="radio" name=
            <pre>Hobbies     : <input type="checkbox" name="chkh1" value="Cricket">Cricket <input type="checkbox" name="chkh
            <pre>City        : <select name="lcity">
                                <option value="Mumbai">Mumbai</option>
                                <option value="Pune">Pune</option>
                                <option value="Nashik">Nashik</option>
                                <option value="Hyderabad">Hyderabad</option>
                                <option value="Bangalore">Bangalore</option>
                                </select></pre><br>
            <input type="Submit" name="btnsubmit" value="Submit"> <input type="Reset" name="btnreset" value="Reset"><br>
        </form>
    </body>
</html>
```

4.2 Form Controls



```
<?PHP
    if(isset($_GET["txtfname"]))
    {
        echo "Hello ".$_GET["txtfname"]." ";
    }
    if(isset($_GET["txtmname"]))
    {
        echo $_GET["txtmname"]." ";
    }
    if(isset($_GET["txtlname"]))
    {
        echo $_GET["txtlname"]."<br>";
    }
    if(isset($_GET["txtage"]))
    {
        echo "You are ".$_GET["txtage"]." years old<br>",
    }
    if(isset($_GET["rgender"]))
    {
        echo "Your gender is ".$_GET["rgender"]."<br>";
    }
    echo "You like to play ";
    if(isset($_GET["chkh1"]))
    {
    echo $_GET["chkh1"];
    }
    else if(isset($_GET["chkh2"]))
    {
        echo " ".$_GET["chkh2"];
    }
    else
    {
        echo "nothing";
    }
    if(isset($_GET["lcity"]))
    {
        echo "<br> You live in ".$_GET["lcity"]." city";
    }
?>
```



localhost/phppracticals/forr

```
Hello Mohammed Zaid Shahid Akhtar Khan
You are 100 years old
Your gender is Male
You like to play Cricket
You live in Mumbai city
```

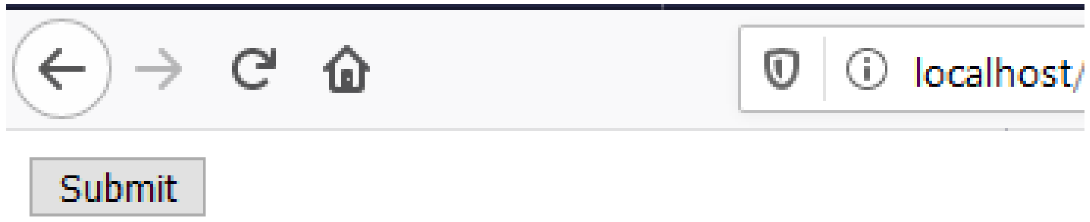
4.2 Form Controls

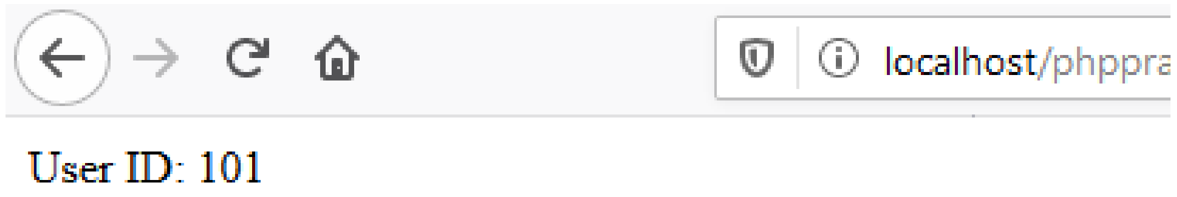
- Hidden Controls:
  - Hidden controls are used to store the data in a webpage that user cant see.
  - Hidden controls will be included in <form> element of a web page what will be used to store data that will not be visible to the user, which will be sent to PHP script available on the server.
  - Data on the server will be fetched by any one of the method (GET or POST).

4.2 Form Controls

- Hidden Controls:

```
<form name="form1" method="post"
  action="hidden_control1.php">
                <input type="hidden"
                name="userid" value="101">
                <input type="submit" value="Subn
        </form>


<?php
  if(isset($_POST["userid"]))
  {
        echo "User ID: ".$_POST["userid"];
  }
?>
```





```
User ID: 101
```

4.3 Working with multiple forms

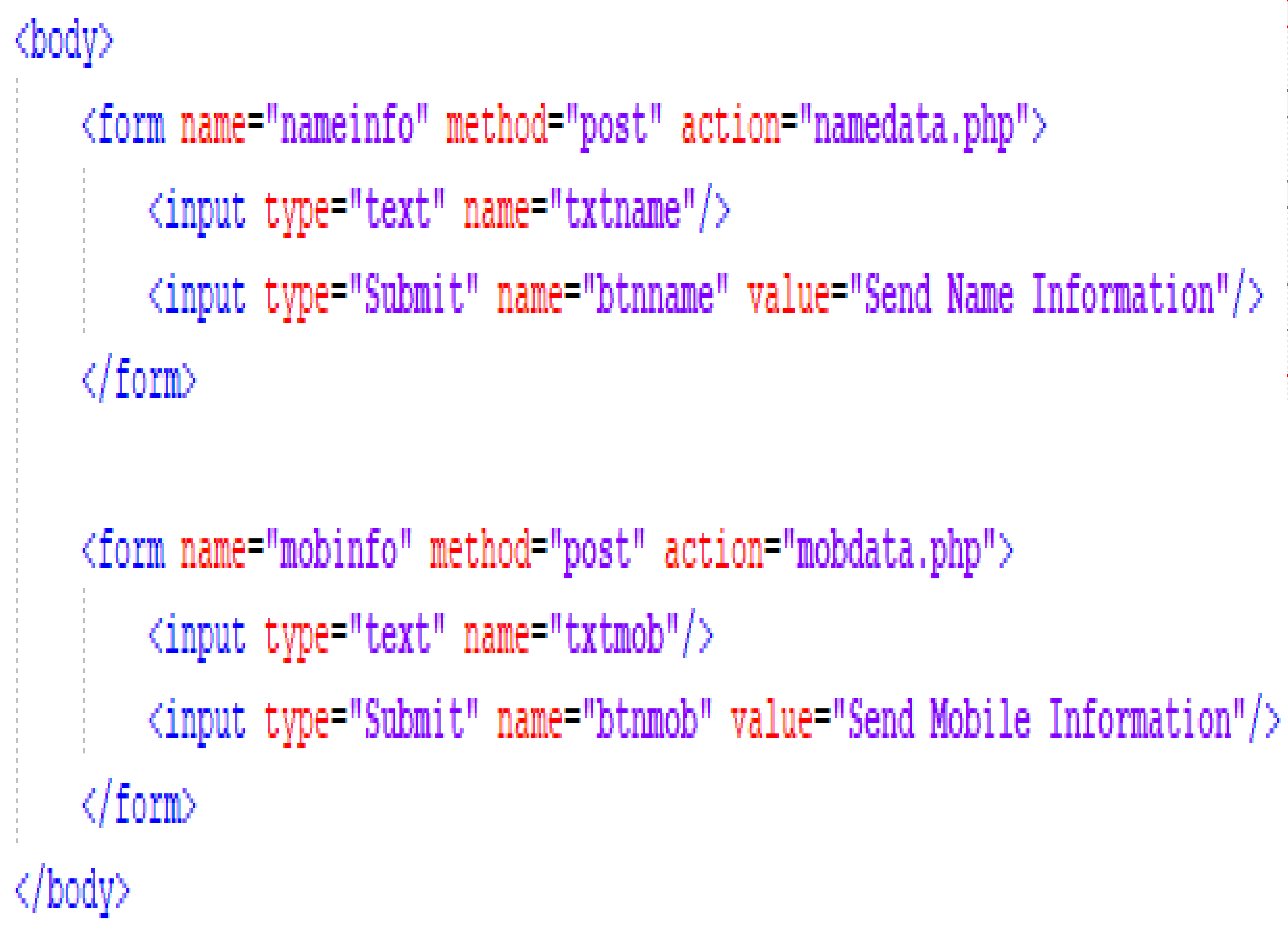
- A web page with multiple forms:

A web page with multiple forms can be processed in 2 ways:

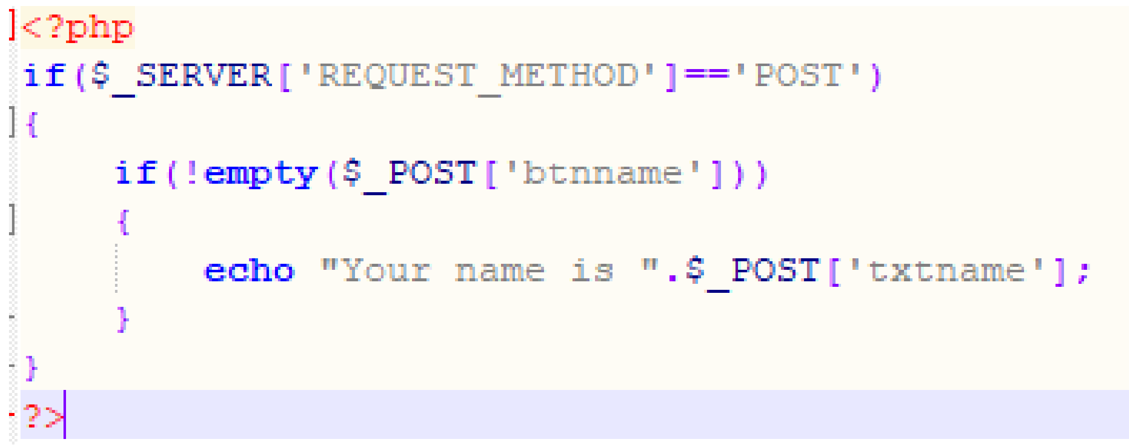
1. Posting each form to different PHP script file for processing:
2. Posting all forms to single PHP script file for processing:

4.3 Working with multiple forms

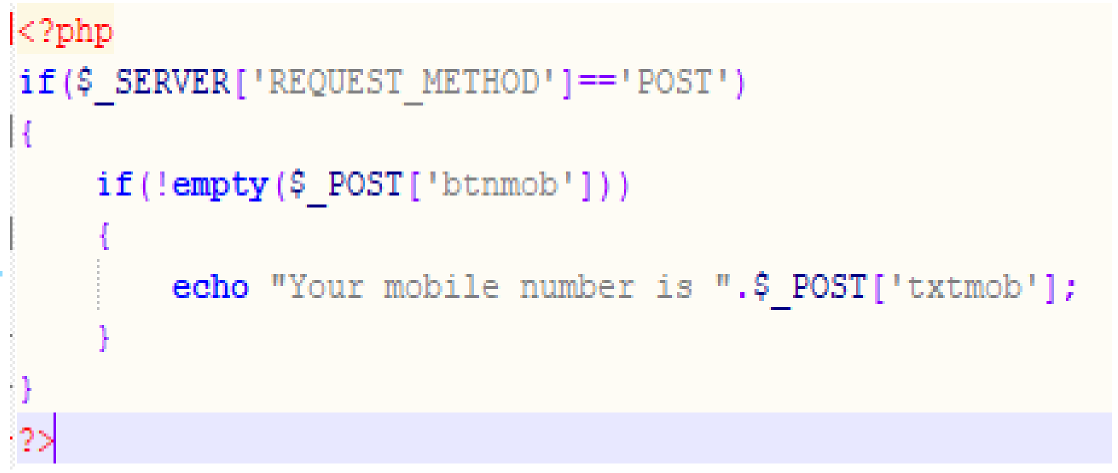
- Posting each form to different PHP script file for processing:



```
<body>
    <form name="nameinfo" method="post" action="namedata.php">
        <input type="text" name="txtname"/>
        <input type="Submit" name="btnname" value="Send Name Information"/>
    </form>

    <form name="mobinfo" method="post" action="mobdata.php">
        <input type="text" name="txtmob"/>
        <input type="Submit" name="btnmob" value="Send Mobile Information"/>
    </form>
</body>
```



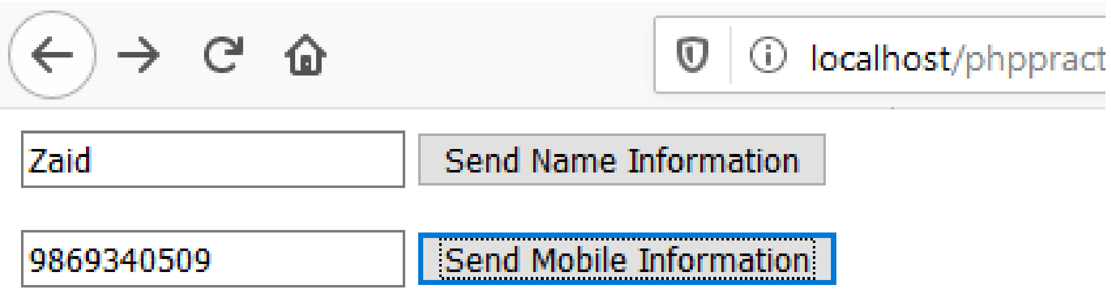
```
<?php
if($_SERVER['REQUEST_METHOD']=='POST')
{
    if(!empty($_POST['btnname']))
    {
        echo "Your name is ".$_POST['txtname'];
    }
}
?>
```

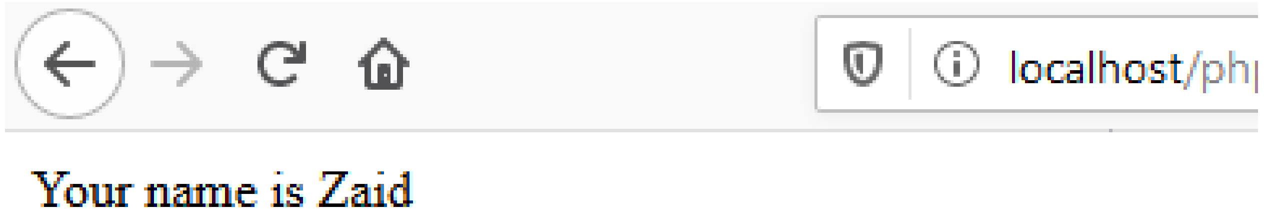


```
<?php
if($_SERVER['REQUEST_METHOD']=='POST')
{
    if(!empty($_POST['btnmob']))
    {
        echo "Your mobile number is ".$_POST['txtmob'];
    }
}
?>
```

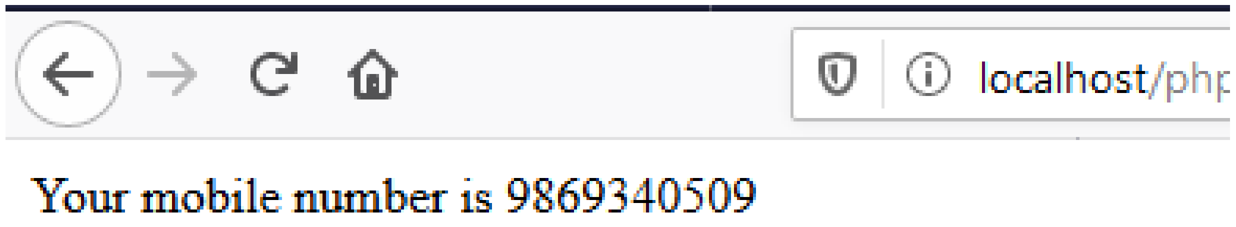
4.3 Working with multiple forms

- Posting each form to different PHP script file for processing:





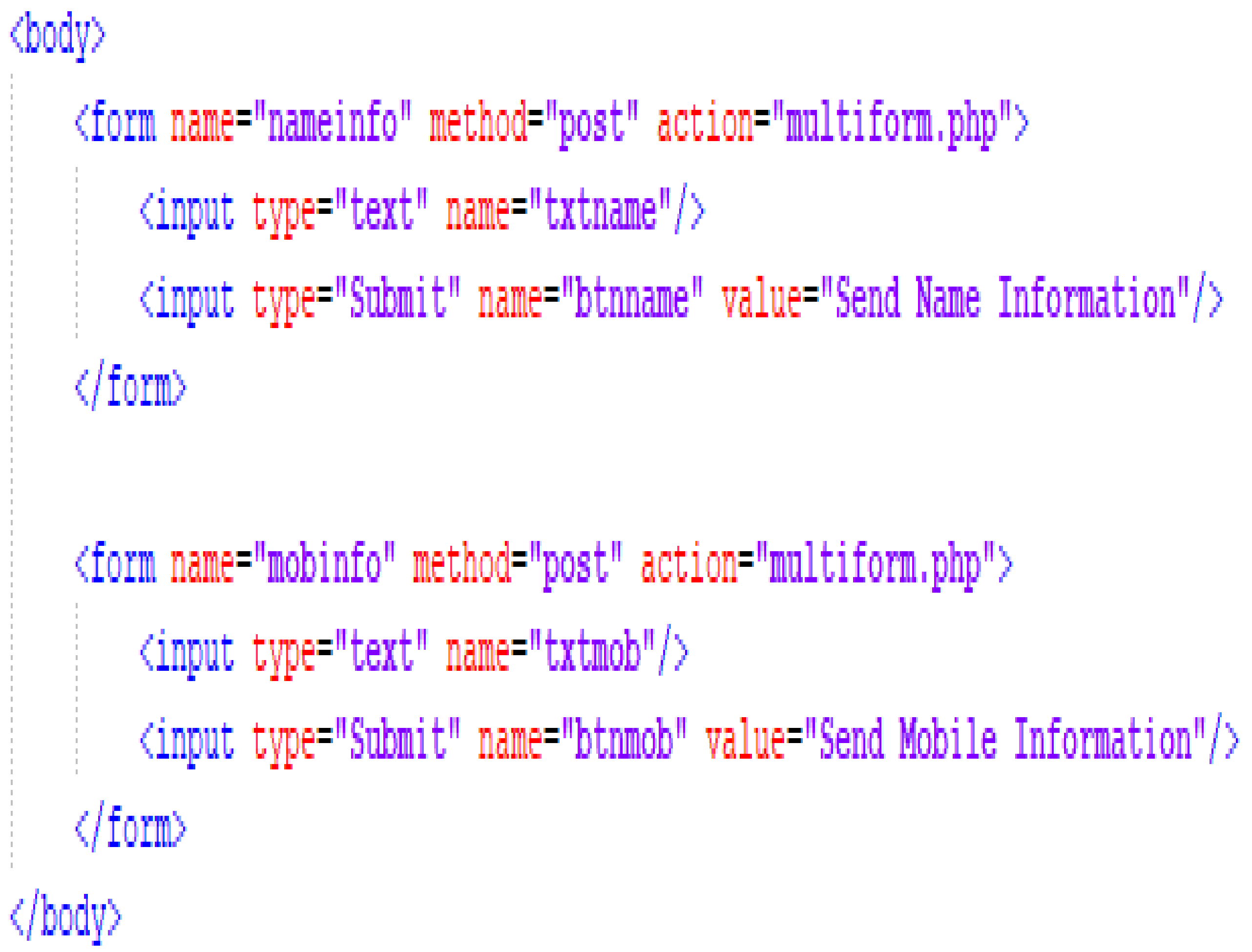
Your name is Zaid



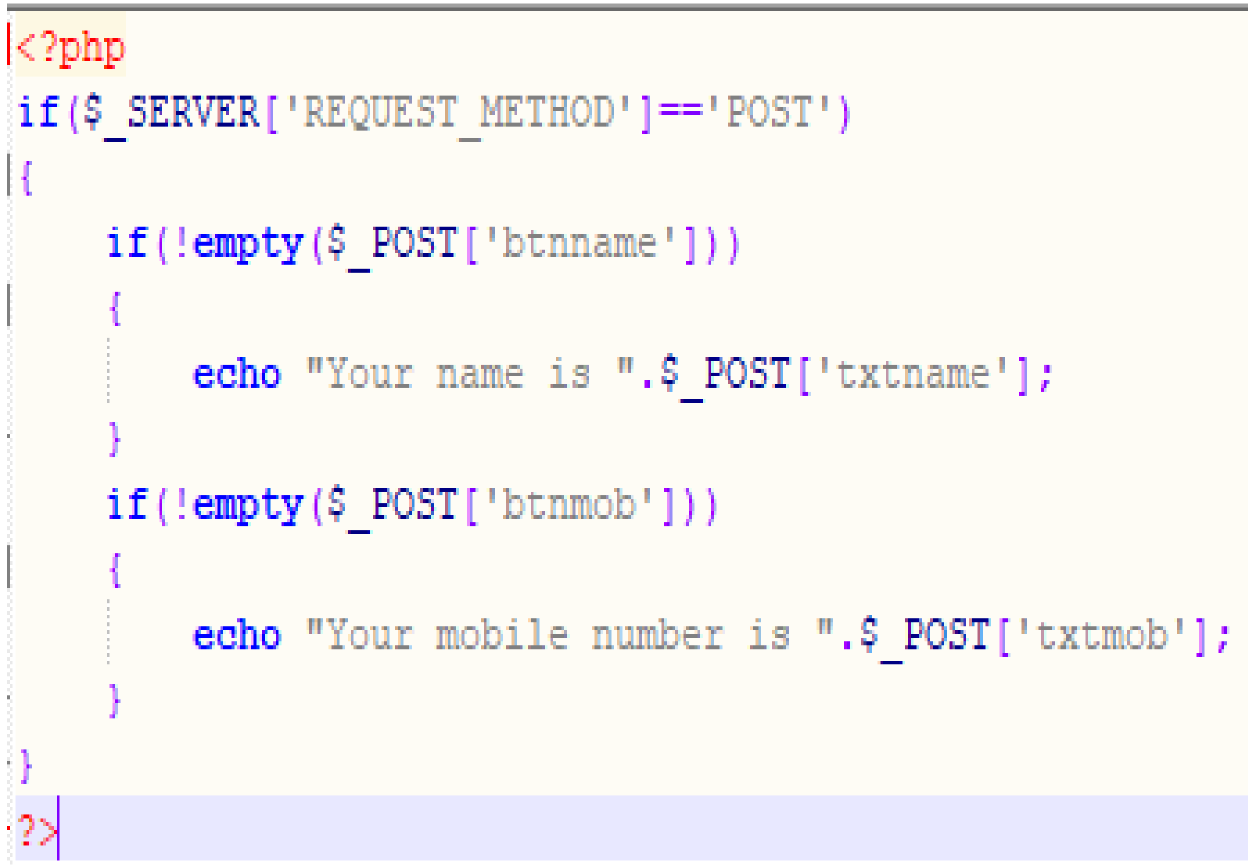
Your mobile number is 9869340509

4.3 Working with multiple forms

- Posting all forms to a single PHP script file for processing:



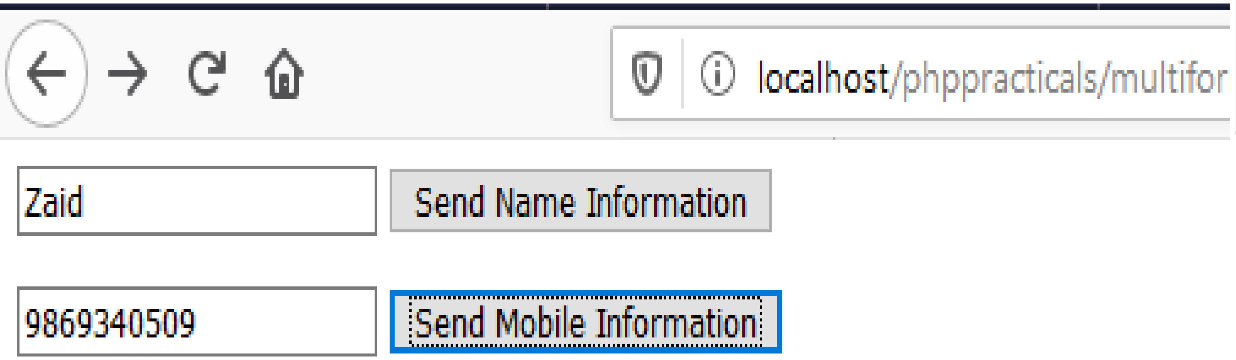
```
<body>
    <form name="nameinfo" method="post" action="multiform.php">
        <input type="text" name="txtname"/>
        <input type="Submit" name="btnname" value="Send Name Information"/>
    </form>

    <form name="mobinfo" method="post" action="multiform.php">
        <input type="text" name="txtmob"/>
        <input type="Submit" name="btnmob" value="Send Mobile Information"/>
    </form>
</body>
```



```
<?php
if($_SERVER['REQUEST_METHOD']=='POST')
{
    if(!empty($_POST['btnname']))
    {
        echo "Your name is ".$_POST['txtname'];
    }
    if(!empty($_POST['btnmob']))
    {
        echo "Your mobile number is ".$_POST['txtmob'];
    }
}
?>
```

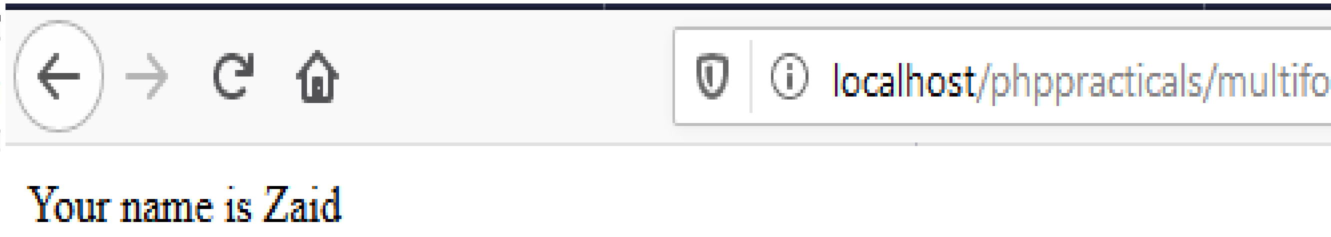
4.3 Working with multiple forms

- Posting all forms to a single PHP script file for processing:



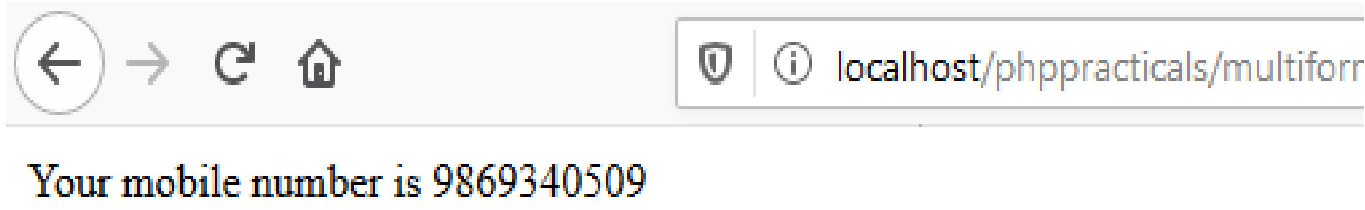
localhost/phppracticals/multifor

Zaid	Send Name Information
9869340509	Send Mobile Information



localhost/phppracticals/multifo

Your name is Zaid

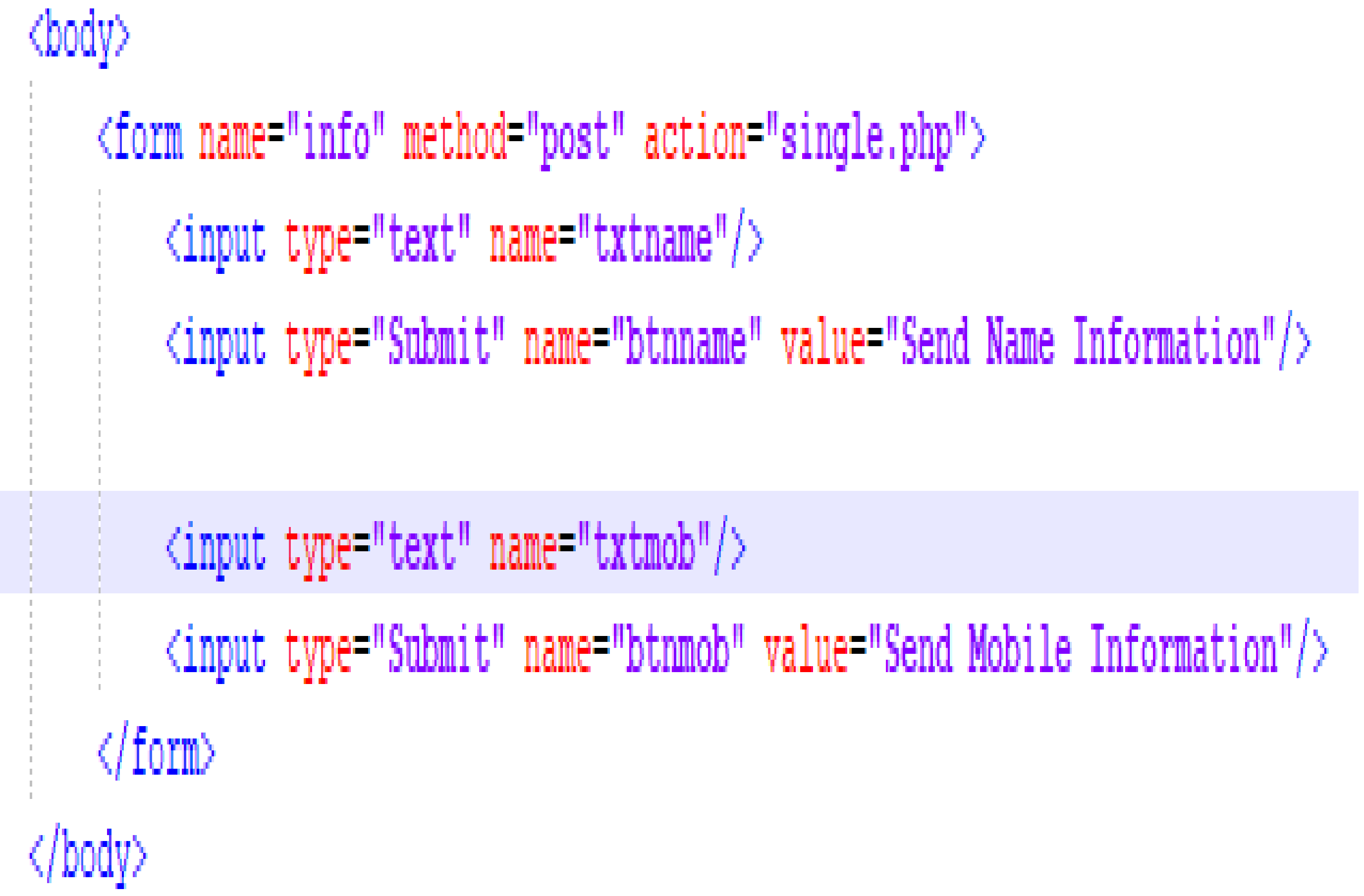


localhost/phppracticals/multiforr

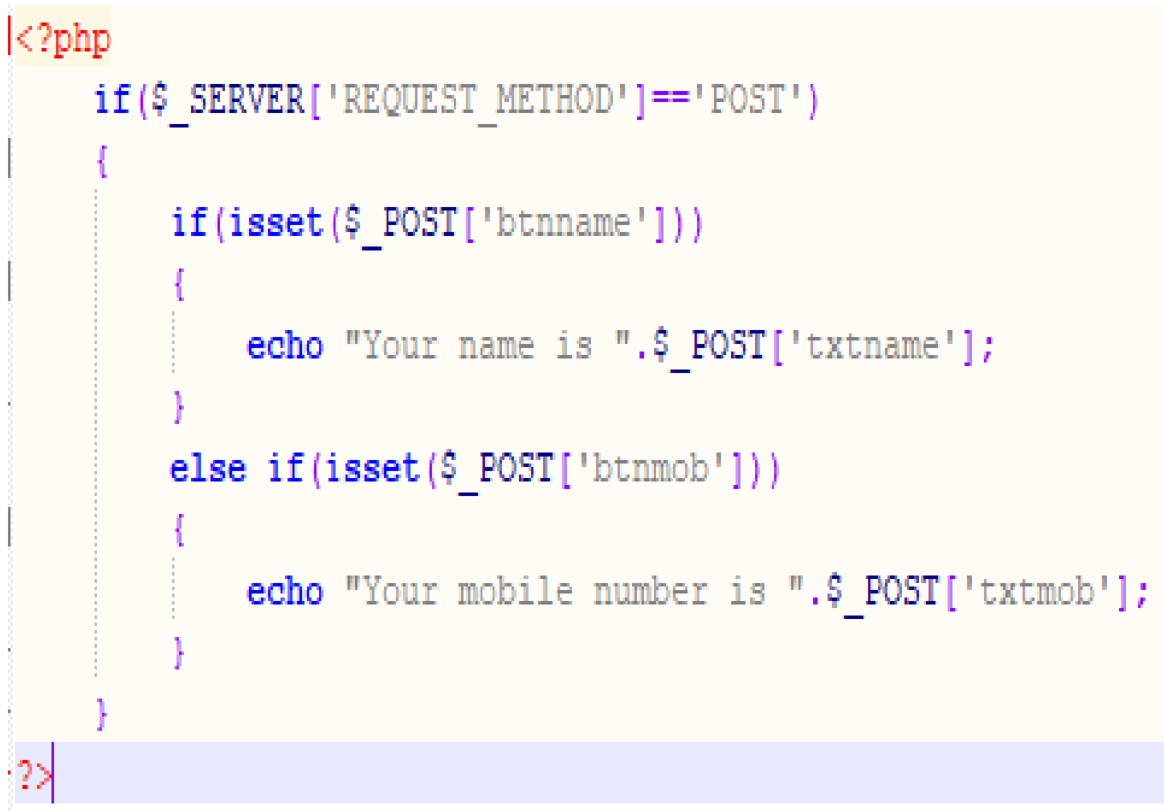
Your mobile number is 9869340509

4.3 Working with multiple forms

- Single form with multiple submit buttons:



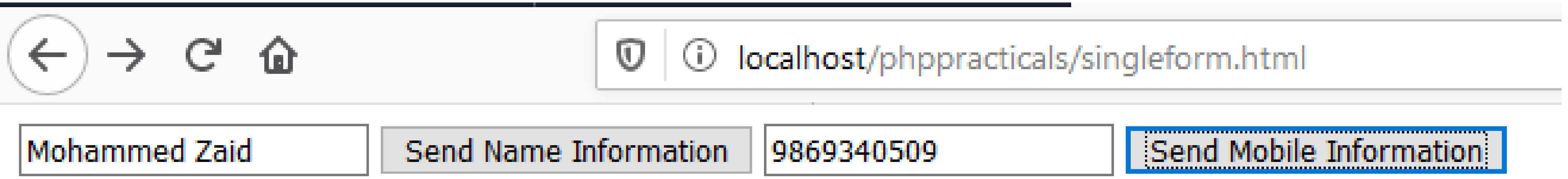
```
<body>
    <form name="info" method="post" action="single.php">
        <input type="text" name="txtname"/>
        <input type="Submit" name="btnname" value="Send Name Information"/>


        <input type="text" name="txtmob"/>
        <input type="Submit" name="btnmob" value="Send Mobile Information"/>
    </form>
</body>
```

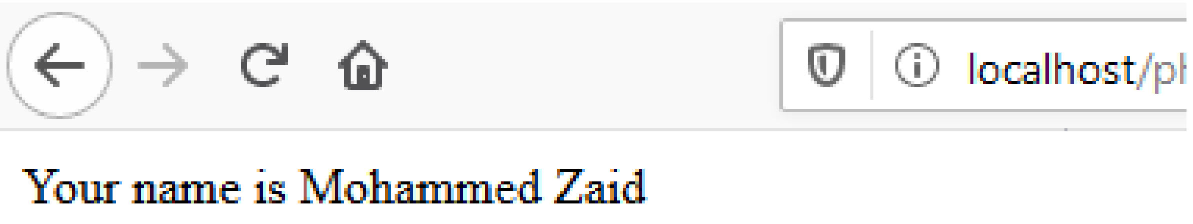


```
<?php
    if($_SERVER['REQUEST_METHOD']=='POST')
    {
        if(isset($_POST['btnname']))
        {
            echo "Your name is ".$_POST['txtname'];
        }
        else if(isset($_POST['btnmob']))
        {
            echo "Your mobile number is ".$_POST['txtmob'];
        }
    }
?>
```

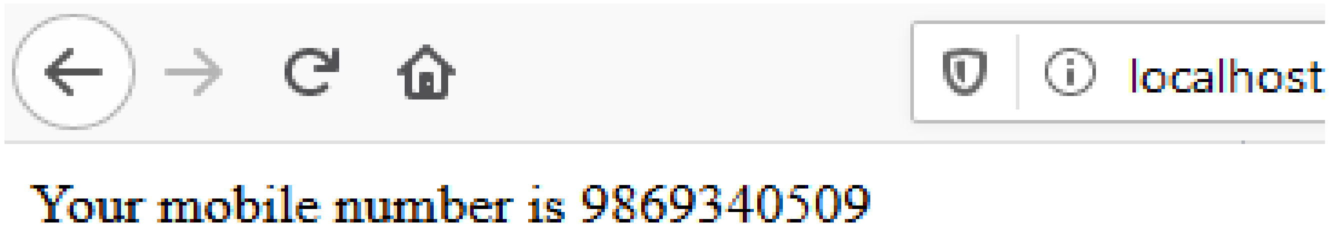
4.3 Working with multiple forms

- Single form with multiple submit buttons:





Your name is Mohammed Zaid



Your mobile number is 9869340509

4.4 Web Page Validation

- PHP provides some inbuilt functions using these functions that input data can be validated.

- empty( ):
  - It ensures that text field is not blank.
  - This function accepts a variable as an argument and returns TRUE when the text field is submitted with *empty string*, *zero*, *NULL* or *FALSE* value.
- is_numeric( ):
  - It ensures that data entered in a text field is a numeric value.
  - Function accepts a variable as an argument and returns TRUE when the text field is submitted with *numeric value*.

Regular Expression

- A regular expression is an object that describes a pattern of characters.

- Regular expressions are used to perform pattern-matching and "search-and-replace" functions on text.

Syntax:

/pattern/modifiers;

- E.g.: var patt = /w3schools/i

Where:	w3schools/i	is a regular expression
	w3schools	is a pattern to be used in search
	i	is a modifier (modifies search to be case-insensitive)

Regular Expression

- Modifiers:
  - g : performs a global match, case-sensitive(find all matches rather than stopping after the first match)
  - i : performs case-insensitive matching and returns the first occurrence.
  - m: By default, all matching or search operation is done as case sensitive and on single line. To perform search or matching on text containing new line character (\n) use modifier (*m*). Performs multiline matching.

Regular Expression

- Brackets: they are used to find a range of characters.
  - [abc] : find any character between the bracket. i.e. between a, b and c.
  - [^abc]   : find any character NOT between the brackets.
  - [0-9]: Find any character between the brackets (any digit).
  - (x|y): Find any of the alternatives specified.

Regular Expression

- Metacharacters: Characters with special meaning.
  - . : Find any single character, except newline or line terminator.
  - \w : Find a word character.
  - \W: Find a Non word character.
  - \d : Find a digit.
  - \D : Find a Non digit character.
  - \s : Find a white space character.
  - \S : Find a Non white space character.
  - \b : Find a match at the beginning of a word: \bDICE
    or at the end of the word: DICE\b

Regular Expression

- Quantifiers:
  - n+ : Matches any string that contains at least one n. (1 or more)
  - n* : Matches any string that contains zero or more occurrences of n.
  - n?: Matches any string that contains zero or one occurrences of n.
  - n{X} : Matches any string that contains a sequence of X n’s.
  - n{X,Y} : Matches any string that contains a sequence of X to Y n’s.
  - n{X,} : Matches any string that contains a sequence of at least X n’s.
  - n$: Matches any string with n at the end of it.
  - ^n : Matches any string with n at the beginning of it.

4.4 Web Page Validation

- preg_match( ):
  - This function return whether a match was found in a string or not. Returns 1 if the pattern was found and 0 if not.

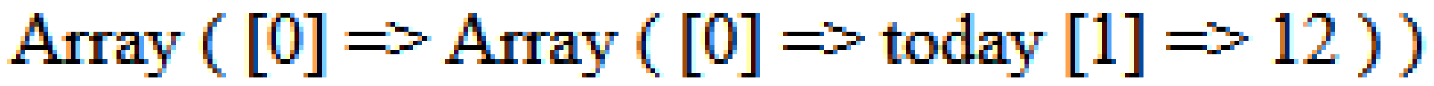
Syntax: preg_match(pattern, input, matches, flags, offset)

Pattern: Required, contains a regular expression
Input: Required, the string in which the search will be performed.
Matches: Optional. The variable used in this parameter will be populated with an array containing all of the matches that were found
Flags: Optional. A set of options that change how the matches array is structured:
Offset: Optional. Defaults to 0. Indicates how far into the string to begin searching. The preg_match() function will not find matches that occur before the position given in this parameter

4.4 Web Page Validation

- preg_match( ):

```
<?php
$str = "How are you today";
$pattern = "/Today/i";
preg_match($pattern, $str, $matches, PREG_OFFSET_CAPTURE);
print_r($matches);
?>
```



```
Array ( [0] => Array ( [0] => today [1] => 12 ) )
```

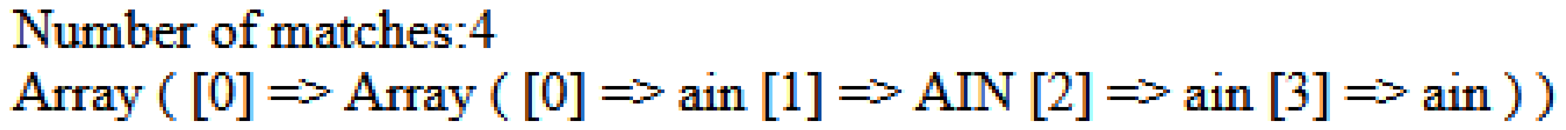
4.4 Web Page Validation

- preg_match_all( ):
  - Returns the number of times the pattern was found in the string, which may also be 0

Syntax: preg_match_all(pattern, input, matches, flags, offset)

E.g.:

```
<?php
$str = "The rain in SPAIN falls mainly on the plains.";
$pattern = "/ain/i";
if(preg_match_all($pattern, $str, $matches)) {
    echo "Number of matches:".preg_match_all($pattern, $str, $matches)."<br>";
  print_r($matches);
}
?>
```



```
Number of matches:4
Array ( [0] => Array ( [0] => ain [1] => AIN [2] => ain [3] => ain ) )
```

4.4 Web Page Validation

- preg_replace( ):
  - Returns a new string where matched patterns have been replaced with another string

Syntax: preg_replace(pattern, replacements, input, limit, count)

Pattern: Required, contains a regular expression

Replacements: Required. A replacement string or an array of replacement strings

Input: Required, the string in which the search will be performed.

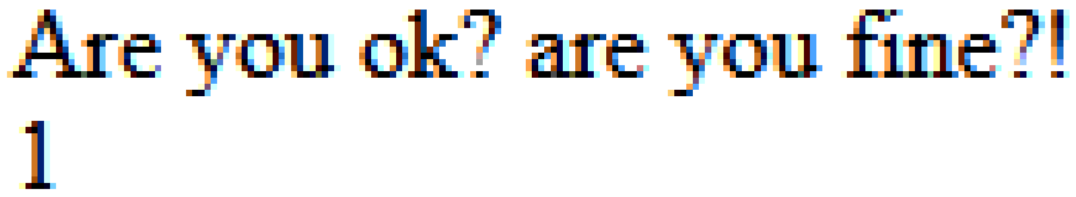
Limit: Optional. Defaults to -1, meaning unlimited. Sets a limit to how many replacements can be done in each string.

Count: Optional, Indicates how many replacements were performed.

4.4 Web Page Validation

- preg_replace( ):

```
<?php
$str = 'are you ok? are you fine?!';
$pattern = '/are/i';
echo preg_replace($pattern, 'Are', $str,1,$x);
echo "<br>".$x
?>
```



```
Are you ok? are you fine?!
1
```

4.5 Cookies

- Use of cookies:
  - A cookie is a small file that the server embeds on the user's computer.
  - Each time the same computer requests a page with a browser, it will send the cookie too.
  - A cookie is often used to identify a user (sessions).
  - Hence, securing a cookie effectively means securing a user's identity.

4.5 Cookies

- Attributes of cookies:
  - Name & Value
  - Secure
  - Domain
  - Path
  - HTTPOnly
  - Expires

4.5 Cookies

- Attributes of cookies: Name & Value

  - Name specifies the name of the cookie. It is a required attribute.

  - Value specifies the value of the cookie. It is an option attribute.

4.5 Cookies

- Attributes of cookies: Secure
  - It specifies whether or not the cookie should only be transmitted over a secure HTTPS connection.
  - *TRUE* indicates that the cookie will only be set if a secure connection exists.
  - Default is *FALSE*.
  - It is an optional attribute.

4.5 Cookies

- Attributes of cookies: Domain
  - It specifies the domain for which the cookie is valid and can be submitted with every request for this domain or its sub domain.
  - If this attribute is not specified, then the host name of the originating server is used as the default value.
  - It is an optional attribute.

4.5 Cookies

- Attributes of cookies: Domain
  - E.g. if a cookie is set by an application at app.mydomain.com with no domain attribute set, then the cookie will be resubmitted for all the subsequent requests for app.mydomain.com & its subdomain but not to otherapp.mydomain.com
  - If domain attribute is set to mydomain.com, then cookie would be sent to all requests for app.mydomain.com and mydomain.com.

4.5 Cookies

- Attributes of cookies: Path
  - signifies the URL or path for which the cookie is valid.
  - If set to "/", the cookie will be available within the entire domain. If set to "/php/", the cookie will only be available within the php directory and all sub-directories of php.
  - The default value is the current directory that the cookie is being set in.
  - It is an optional attribute.

4.5 Cookies

- Attributes of cookies: HTTPOnly
  - If set to TRUE the cookie will be accessible only through the HTTP protocol (the cookie will not be accessible by scripting languages).
  - This setting can help to reduce identity theft through XSS (Cross-site Scripting) attacks.
  - Default is FALSE.
  - This is an optional attribute.

4.5 Cookies

- Attributes of cookies: Expires
  - Specifies when the cookie expires.
  - If this parameter is omitted or set to 0, the cookie will expire at the end of the session (when the browser closes).
  - Default is 0.
  - It is an optional attribute.

4.5 Cookies

- Creating a Cookie:
  - A cookie can be created using setcookie( ) function.

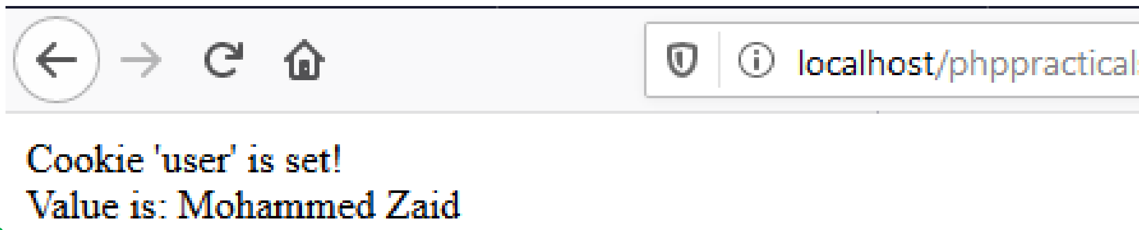
    Syntax:

    setcookie(name, value, expire, path, domain, secure, httponly);
  - Only name parameter is required.
  - All other parameters are optional.
  - The setcookie() function must appear BEFORE the <html> tag.

4.5 Cookies

- Creating a Cookie:

```
<?php $cookie_name = "user";
$cookie_value = "Mohammed Zaid";
setcookie($cookie_name, $cookie_value,
time() + (86400 * 30), "/"); // 86400 = 1 day ?>
<html>
<body>
<?php if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}?>
</body></html>
```



```
Cookie 'user' is set!
Value is: Mohammed Zaid
```

4.5 Cookies

- Modifying a Cookie:

  - To modify a cookie, just set (again) the cookie using the setcookie( ) function.
  - The value of the cookie is automatically URLencoded when sending the cookie.
  - It is automatically decoded when received.
  - To prevent URLencoding, use setrawcookie() instead.

4.5 Cookies

- Deleting a Cookie:

  - To delete a cookie, use the setcookie function with an expiration date in the past.

  - E.g. time( ) – 3600
    set expiry date with current time – value.

4.6 Session

- Use of Session:
  - Session is a way to store information (in variables) to be used across multiple web pages.
  - Unlike a cookie, the information is not stored on the user's computer.
  - By default session variables last until the user closes the browser.
  - Session variables hold information about one single user, and are available to all pages in one application.

4.6 Session

- Start of Session:
  - Session_start( ) function can be used to start a session.
  - Session variables are set with the PHP global variable: $_SESSION
  - The session_start() function must be the very first thing in your document. Before any HTML tags.

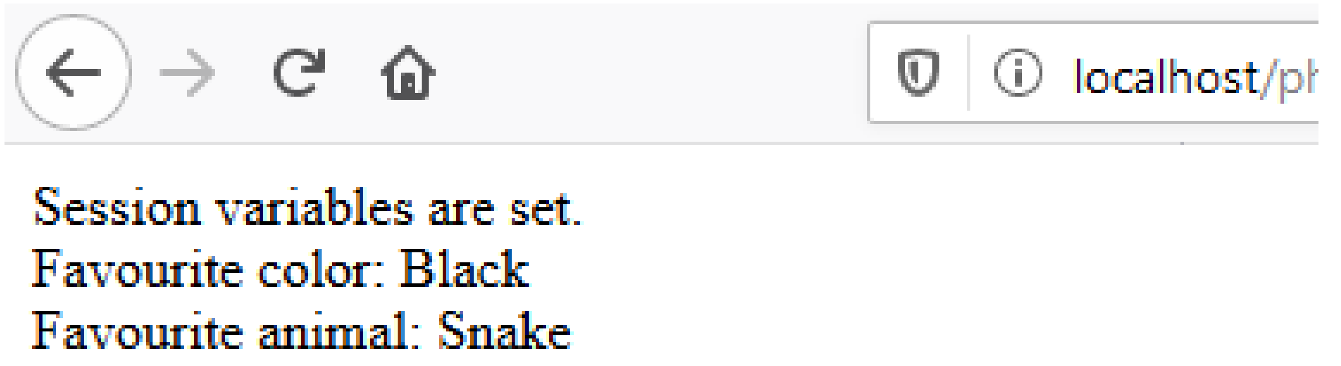
4.6 Session

- Start of Session:
  - To remove all global session variables, use *session_unset( )* and function.
  - To destroy the session, use *session_destroy( ) function.*

4.6 Session

- Start of Session:

```
<?php
session_start(); ?>
<html> <body>
<?php
$_SESSION["favcolor"] = "Black";
$_SESSION["favanimal"] = "Snake";
echo "Session variables are set.";
echo "<br>Favourite color: ".$_SESSION["favcolor"]."<br>Favourite animal: ".$_SESSION["favanimal"];
session_unset();
session_destroy(); ?>
</body> </html>
```



```
Session variables are set.
Favourite color: Black
Favourite animal: Snake
```

4.7 Sending E-mail

- mail( ):
  - mail( ) function allows user to send emails directly from a script.

Syntax:

mail(to,subject,message,headers,parameters);

Where:

To : Required. Specifies the receiver/s of the email.
Subject: Required. Specifies the subject of the email. It cannot contain any newline characters.
Message: Required. Defines the message to be sent. Each line should be separated with (\n). Line should not exceed 70 characters.
Headers: Optional. Specifies additional headers, Cc and Bcc.
Parameters: Optional. Specifies an additional parameter to the sendmail program.