A large red square with a white border, centered on a white background. Inside the square, the text "Form Processing in PHP" is written in white, bold, sans-serif font, centered horizontally and vertically.

Form Processing in PHP

Forms

Forms are special components which allow your site visitors to supply various information on the HTML page.

We have previously talked about creating HTML forms.

Forms typically include some input elements to gather information.

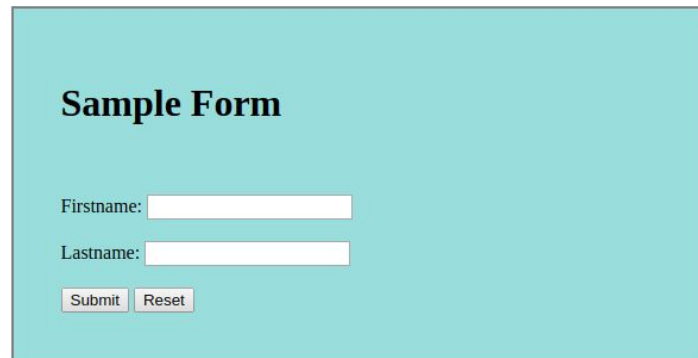
We have also talked about client-side form validation using javascript.

This time we are going to talk about processing forms at server side using PHP.

Forms

index.html

```
<form name="myForm" id="myForm">
  <br><br>
  <span id="firstname">Firstname: </span>
  <input type="text" name="firstname"><br><br>
  <span id="lastname">Lastname: </span>
  <input type="text" name="lastname"><br><br>
  <input type="submit" name="submitBtn">
  <input type="reset" name="resetBtn">
</form>
```



Sample Form

Firstname:

Lastname:

Forms

HTML forms can have many attributes. Two of the most important attributes for HTML forms that we haven't talked about so far are:

action: The action parameter tells the browser what script/site must be called when the visitor presses the submit button.

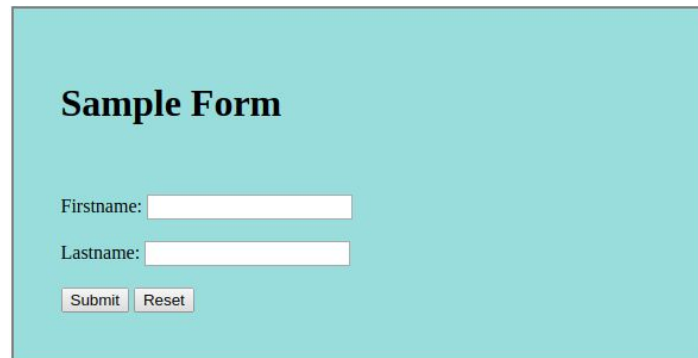
method: The method parameter tells the browser by which method to send the user submitted data to the web-server. The parameter values are either **POST** or **GET**.

Forms

index.html

```
<form name="myForm" id="myForm"
action="process.php" method="GET">
  <br><br>
  <span id="firstname">Firstname: </span>
  <input type="text" name="firstname"><br><br>
  <span id="lastname">Lastname: </span>
  <input type="text" name="lastname"><br><br>
  <input type="submit" name="submitBtn">
  <input type="reset" name="resetBtn">
</form>
```

We are telling the browser to send form data to "process.php" file by using "get" method, upon form submission.



Sample Form

Firstname:

Lastname:

Accessing Form Data in PHP using GET

when the user enters information in the form and clicks the submit button, the form data will be sent to the PHP file specified by the action attribute

When GET method is used, form data will be saved in an array called `$_GET`

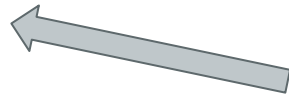
`$_GET` is an associative array, with the form field names used as the array keys and form field values stored as values

`$_GET['firstname']`



This array element holds the value for firstname field in the form

`$_GET['lastname']`



This array element holds the value for lastname field in the form

Using Form Data in PHP

Once you know how to access form data in PHP, you can use it just like any other variable

In this example, we write a PHP function that welcomes the user to our web site.

process.php

```
<?php
function welcome($fname, $lname)
{
    echo '<h1>Welcome ' . $fname . ' ' . $lname . '!</h1>';
}

welcome($_GET['firstname'], $_GET['lastname'])
?>
```

Did you notice the URL changed?!

Great! Now we have an HTML form that sends data back to our server for processing. And we have a separate PHP script that reads that data and does some simple stuff with it.

[Check it out here!](#)

Once you submit the form, you see a welcome message on a new page.

But did you notice the URL changed once you submit the form?

Why did the URL change?

There are 2 reasons for this:

1. We had our PHP script in a separate file from our form fields
2. We are using GET method to send data back to the server

We could have used a single PHP file to show our HTML form and process it!

Let's see how

PHP and HTML in the Same File

index2.php

```
<?php
if ($_GET['firstname'] != null)
{
    function welcome($fname, $lname)
    {
        echo '<h1>Welcome ' . $fname . ' ' . $lname . '!</h1>';
    }

    welcome($_GET['firstname'], $_GET['lastname']);
}
else {
?>
<!doctype html>
<html>
<head>
<title>Form Validation</title>
```

PHP and HTML in the Same File

index2.php continued

```
<link rel="stylesheet" href="css/style.css">
</head>
<body>
<div id="container">
<h1>Sample Form</h1>
<form name="myForm" id="myForm" action="index2.php" method="get">
  <br><br>
  <span id="firstname">Firstname: </span>
  <input type="text" name="firstname"><br><br>
  <span id="lastname">Lastname: </span>
  <input type="text" name="lastname"><br><br>
  <input type="submit" name="submitBtn"><input type="reset" name="resetBtn">
</form>
</div>
</body>
</html>
<?php
}
?>
```

method="GET"

When using method="get" in HTML forms, all variable names and values are displayed in the URL.

This method should not be used when sending passwords or other sensitive information!

However, because the variables are displayed in the URL, it is possible to bookmark the page. This can be useful in some cases.

The get method is not suitable for large variable values; the value cannot exceed 100 chars.

method="POST"

The built-in `$_POST` array is used to collect values from a form sent with `method="post"`.

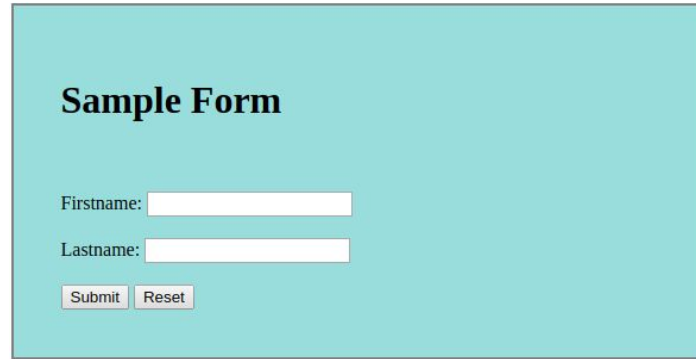
Information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send.

Form with method="POST"

index3.html

```
<form name="myForm" id="myForm"
action="process2.php" method="POST">
  <br><br>
  <span id="firstname">Firstname: </span>
  <input type="text" name="firstname"><br><br>
  <span id="lastname">Lastname: </span>
  <input type="text" name="lastname"><br><br>
  <input type="submit" name="submitBtn">
  <input type="reset" name="resetBtn">
</form>
```

We are telling the browser to send form data to "process.php" file by using "post" method, upon form submission.



The image shows a screenshot of a web browser displaying a form titled "Sample Form". The form is set against a light blue background. It contains two text input fields: "Firstname:" followed by an empty text box, and "Lastname:" followed by another empty text box. Below these fields are two buttons: "Submit" and "Reset".

Accessing Form Data in PHP using POST

when the user enters information in the form and clicks the submit button, the form data will be sent to the PHP file specified by the action attribute

When POST method is used, form data will be saved in an array called `$_POST`

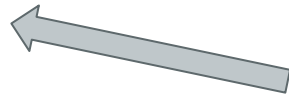
`$_POST` is an associative array, with the form field names used as the array keys and form field values stored as values

`$_POST['firstname']`



This array element holds the value for firstname field in the form

`$_POST['lastname']`



This array element holds the value for lastname field in the form

Using Form Data in PHP

Once you know how to access "POST" form data in PHP, you can use it just like any other variable

In this example, we write a PHP function that welcomes the user to our web site.

process2.php

```
<?php
function welcome($fname, $lname)
{
    echo '<h1>Welcome ' . $fname . ' ' . $lname . '!</h1>';
}

welcome($_POST['firstname'], $_POST['lastname'])
?>
```


Check the URL

Great! Now we have an HTML form that sends data back to our server for processing using POST method. And we have a separate PHP script that reads that data and does some simple stuff with it.

[Check it out here!](#)

Once you submit the form, you see a welcome message on a new page.

But this time, unlike when we used GET method, nothing is attached to the URL.

Everything is invisible to the user.

GET vs POST

We showed that form data can be submitted to the server, using POST or GET methods

With GET, form data will be attached to the URL and is visible to the user

With POST, data is hidden from the user.

But regardless of the differences, both methods can be used to achieve the same functionality while processing forms.

From now on, we are going to use POST method, but everything we discuss also applies to GET method.

Form Processing in PHP

Now that we know the basics of form processing in PHP, let's create a real example of how this can be used.

You can download the complete example from [here](#).

Remember that, this example includes PHP, so you need a webserver with PHP

If you have WAMP or MAMP installed, untar the file and copy the contents in your www or public_html folder.