PHP Form Handling

The PHP $_GET and $_POST variables are used to retrieve information from forms, like user input.

The most important thing to notice when dealing with HTML forms and PHP is that any form element in an HTML page will automatically be available to your PHP scripts.

Form Example

The example HTML page above contains two input fields and a submit button. When the user fills in this form and click on the submit button, the form data is sent to the "welcome.php" file.

```
<html>
<body>

<form action="welcome.php" method="post">
Name: <input type="text" name="name" />
Age: <input type="text" name="age" />
<input type="submit" />
</form>

</body>
</html>
```

Form Example

The "welcome.php" file looks like this:

```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?>.<br />
You are <?php echo $_POST["age"]; ?> years old.

</body>
</html>
```

A sample output of the above script may be:

```
Welcome John.
You are 28 years old.
```

The PHP $_GET and $_POST variables will be fully explained in the next lecture.

Form Validation

User input should be validated whenever possible. Client side validation is faster, and will reduce server load.

However, any site that gets enough traffic to worry about server resources, may also need to worry about site security. You should always use server side validation if the form accesses a database.

A good way to validate a form on the server is to post the form to itself, instead of jumping to a different page. The user will then get the error messages on the same page as the form. This makes it easier to discover the error.