Web Programming

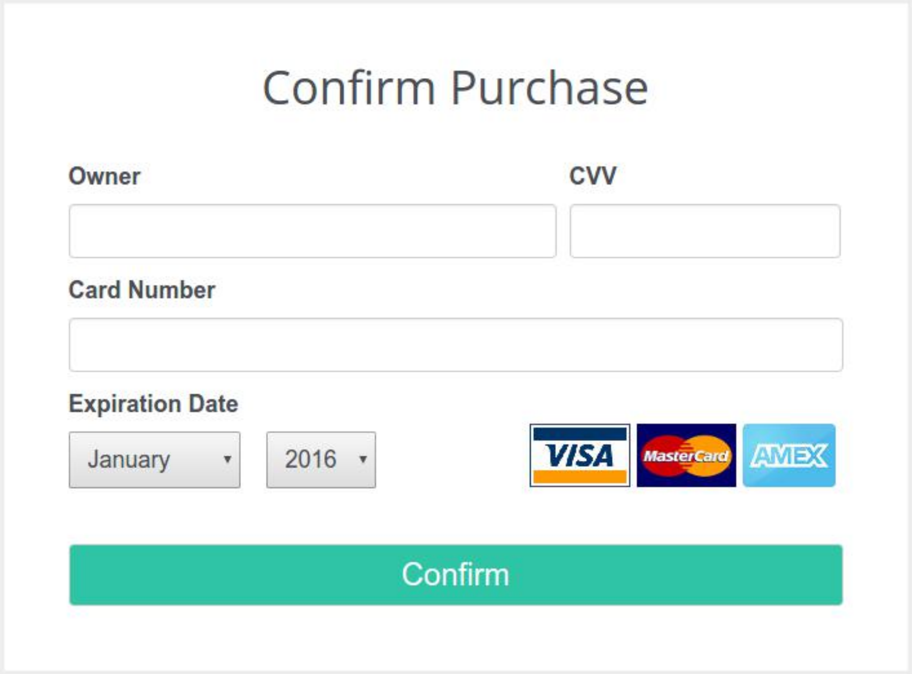
#8 Forms Handling

Herman Kabetta, M.T.

hermanka.github.io

FORMS Handling

PHP also provides the capability to handle input from users through HTML forms in a straightforward manner



PHP Global Variables - Superglobals

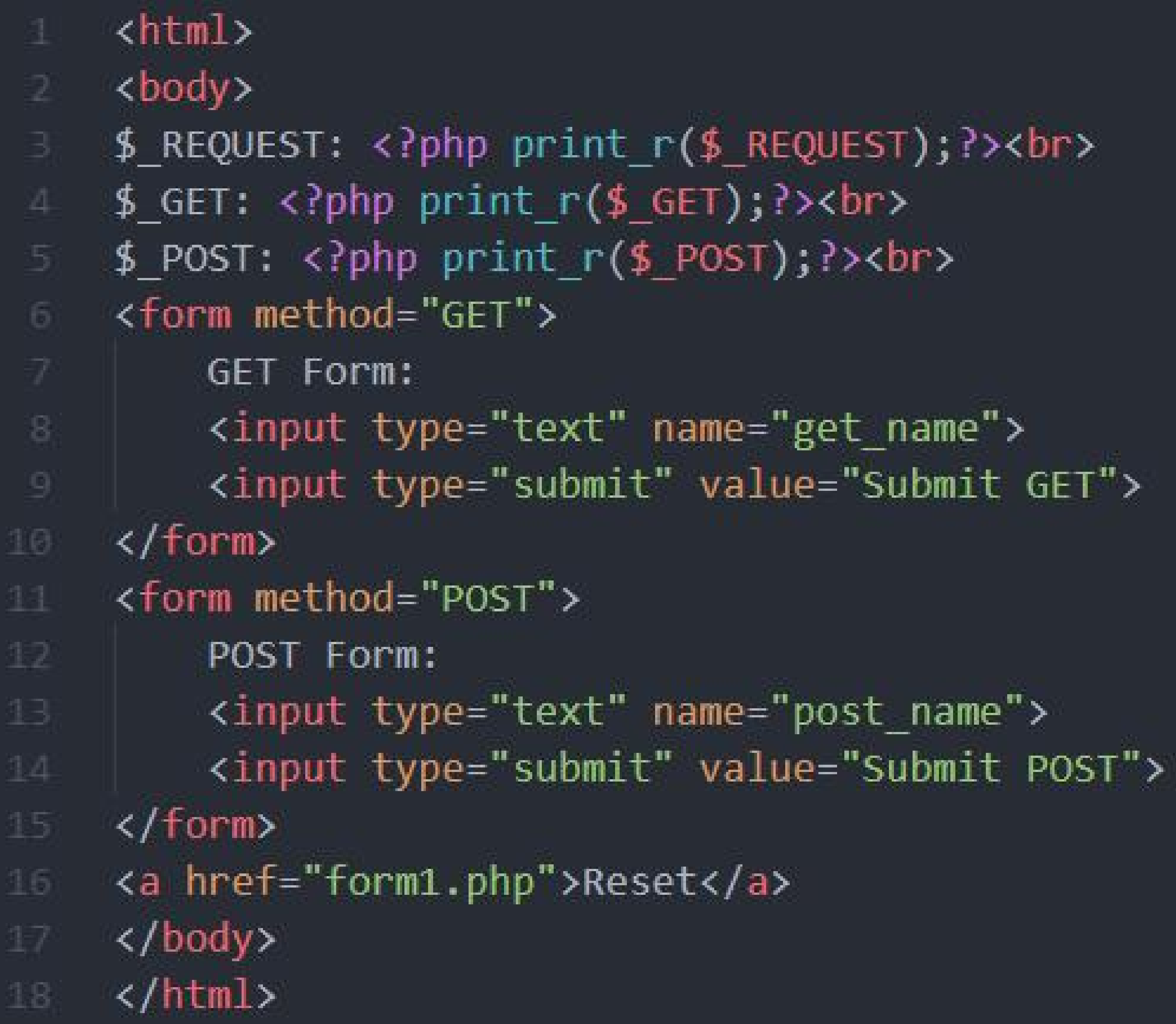
Superglobals were introduced in PHP 4.1.0, and are built-in variables that are always available in all scopes.

- `$GLOBALS`
- `$_SERVER`
- `$_REQUEST`
- `$_POST`
- `$_GET`
- `$_FILES`
- `$_ENV`
- `$_COOKIE`
- `$_SESSION`

Superglobals in Forms Handling

- **$_GET** - this contains an associative array of variables passed to the current script using query parameters in the URL
- **$_POST** - this contains an associative array of variables passed to the current script using a form submitted using the “POST” method
- **$_REQUEST** - this contains the contents of **$_GET**, **$_POST**, and **$_COOKIE**

Superglobals in Forms Handling



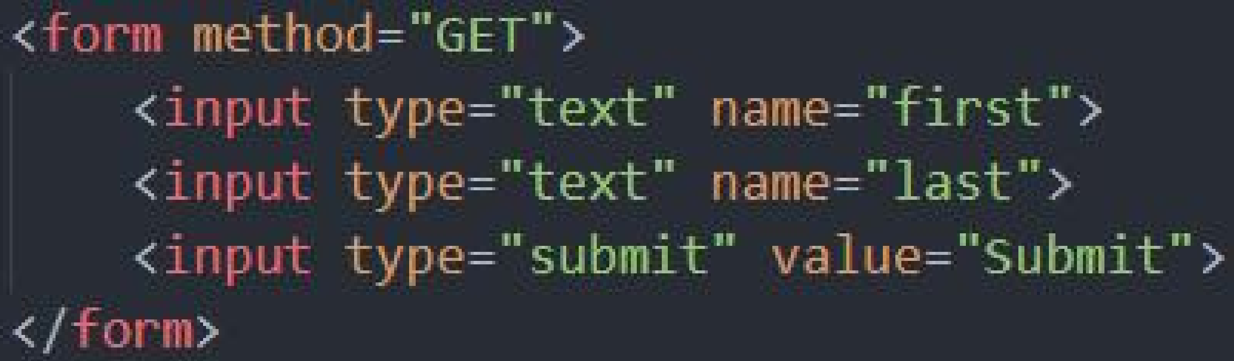
```
<html>
<body>
$_REQUEST: <?php print_r($_REQUEST);?><br>
$_GET: <?php print_r($_GET);?><br>
$_POST: <?php print_r($_POST);?><br>
<form method="GET">
    GET Form:
    <input type="text" name="get_name">
    <input type="submit" value="Submit GET">
</form>
<form method="POST">
    POST Form:
    <input type="text" name="post_name">
    <input type="submit" value="Submit POST">
</form>
<a href="form1.php">Reset</a>
</body>
</html>
```

GET Form Handling

- In HTML, setting a form’s method attribute to "get" specifies that you would like the form to be submitted using the GET method.
- When using this method, the form entries are passed as parameters in a URL query string.

http://localhost/form1.php?first=ellen&last=richards

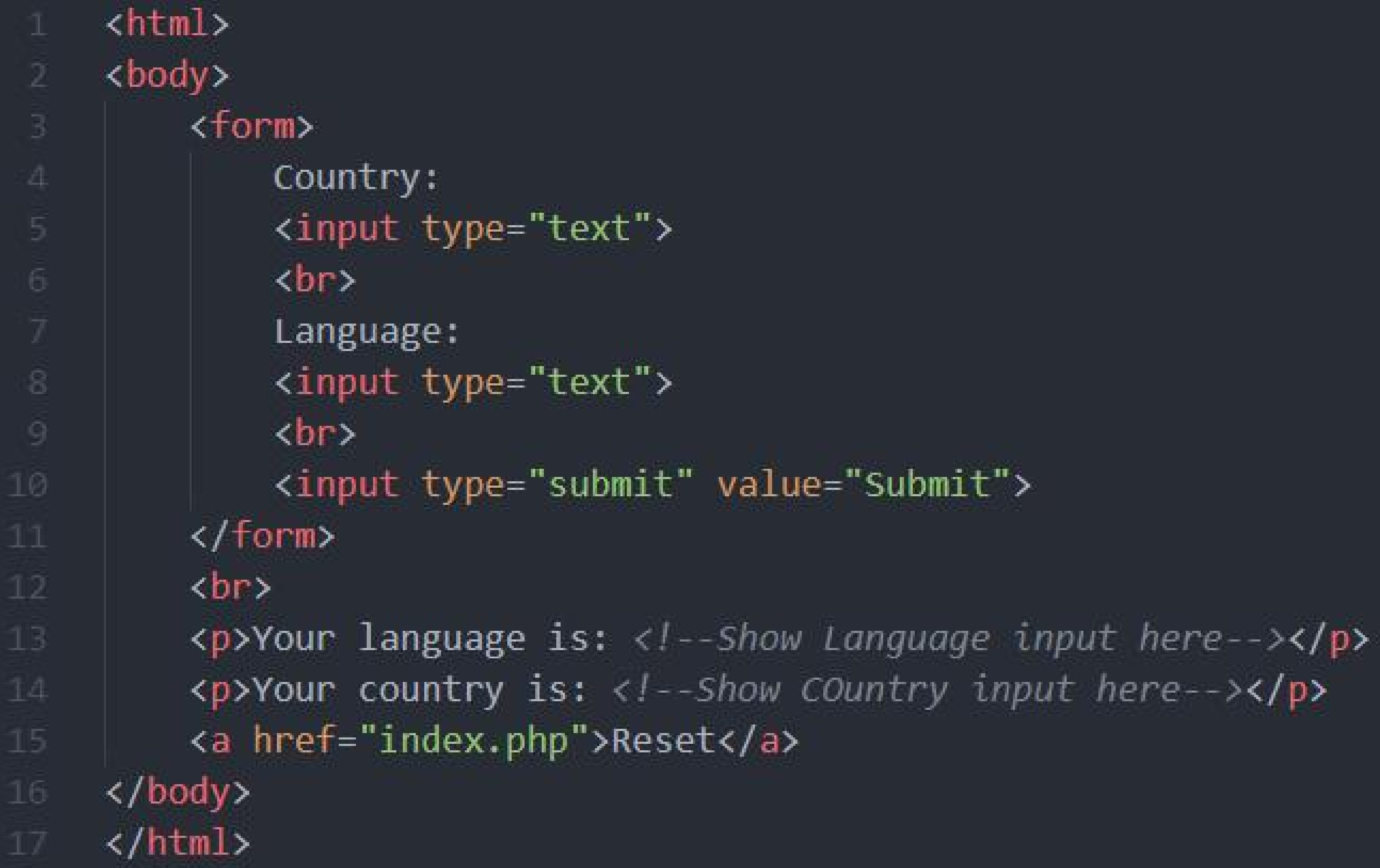
- The parameter names (first and last) come from the name attribute of each form input.



```
<form method="GET">
    <input type="text" name="first">
    <input type="text" name="last">
    <input type="submit" value="Submit">
</form>
```

GET Form Handling

Fixes code below with GET Method



```html
<html>
<body>
    <form>
        Country:
        <input type="text">
        <br>
        Language:
        <input type="text">
        <br>
        <input type="submit" value="Submit">
    </form>
    <br>
    <p>Your language is: <!--Show Language input here--></p>
    <p>Your country is: <!--Show COuntry input here--></p>
    <a href="index.php">Reset</a>
</body>
</html>
```

POST Form Handling

- When using POST to submit forms, you will not see the URL change.
- The form data is sent using the headers of the HTTP request instead of URL parameters.

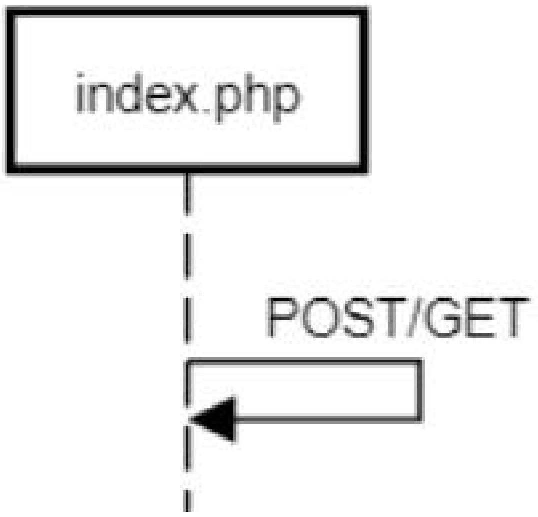
POST Form Handling

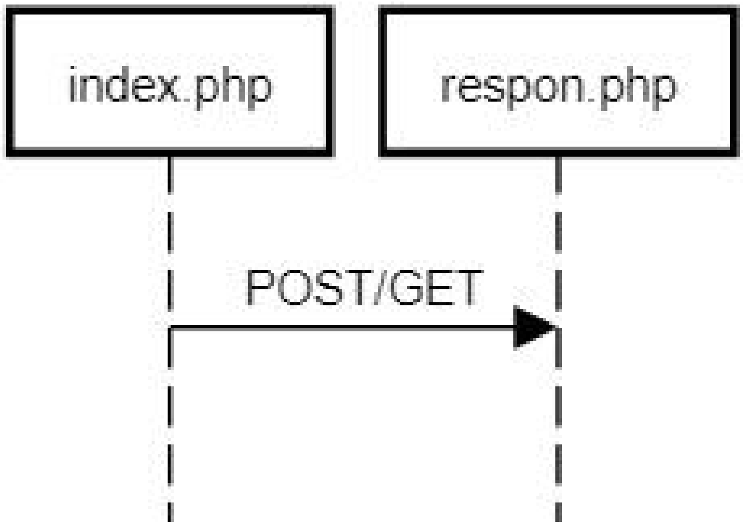
Fixes code below with POST Method

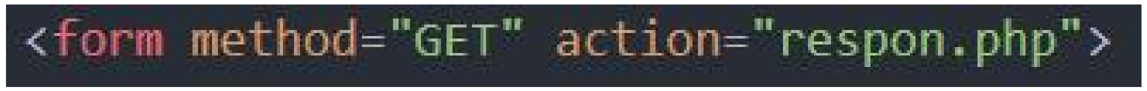
```
<html>
<body>
    <form>
        Favorite Color:
        <input type="text">
        <br>
        Favorite Food:
        <input type="text">
        <br>
        <input type="submit" value="Submit">
    </form>
    <br>
    <p>Best food is: <!--Show Food input here--></p>
    <p>Best color is: <!--Show Color input here--></p>
    <a href="index.php">Reset</a>
</body>
</html>
```

Action Attribute

If you would like to have the user navigate to a new URL and handle the form input there, you can specify the URL in the form’s action attribute.



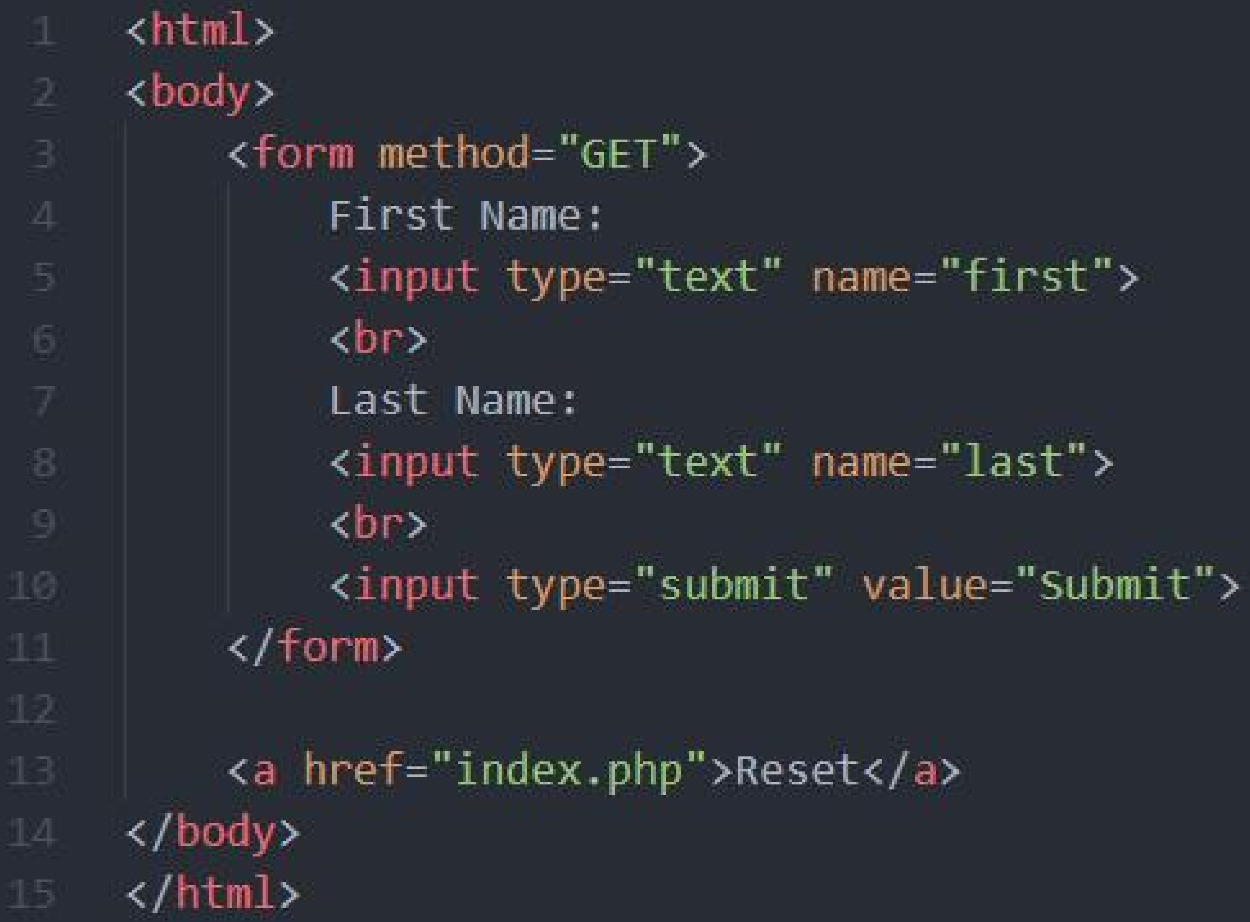




```
<form method="GET" action="respon.php">
```

Action Attribute

index.php

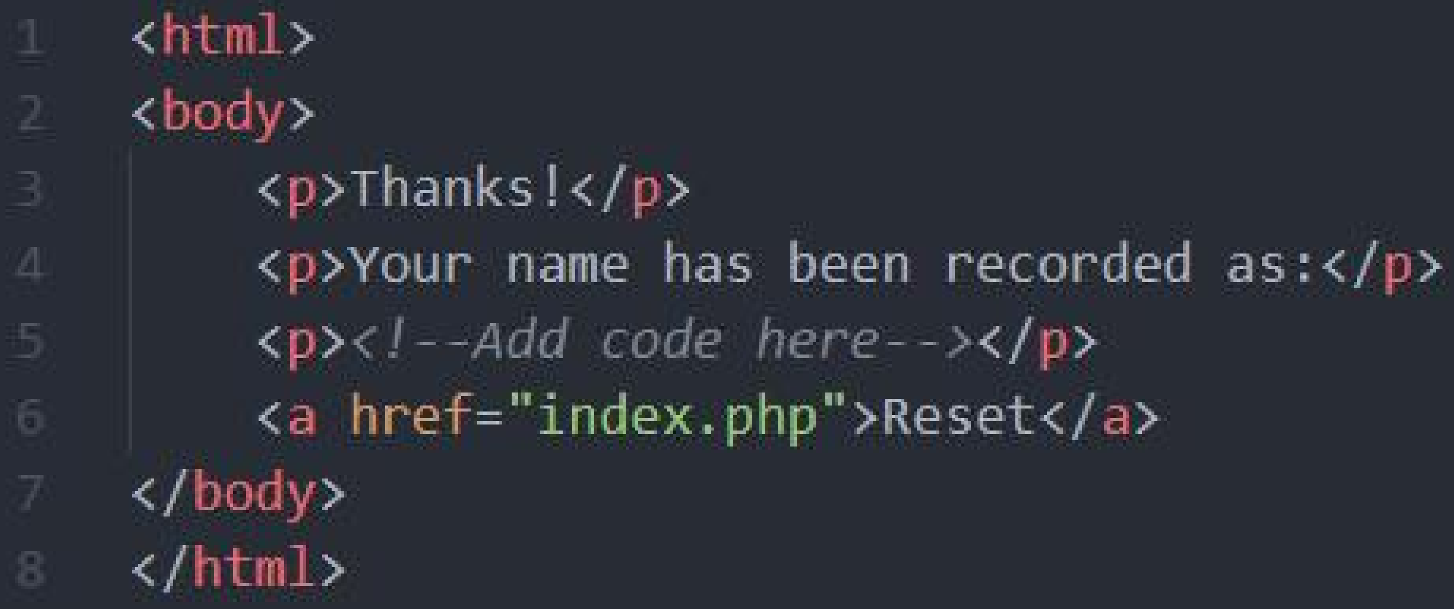


```
<html>
<body>
    <form method="GET">
        First Name:
        <input type="text" name="first">
        <br>
        Last Name:
        <input type="text" name="last">
        <br>
        <input type="submit" value="Submit">
    </form>

    <a href="index.php">Reset</a>
</body>
</html>
```

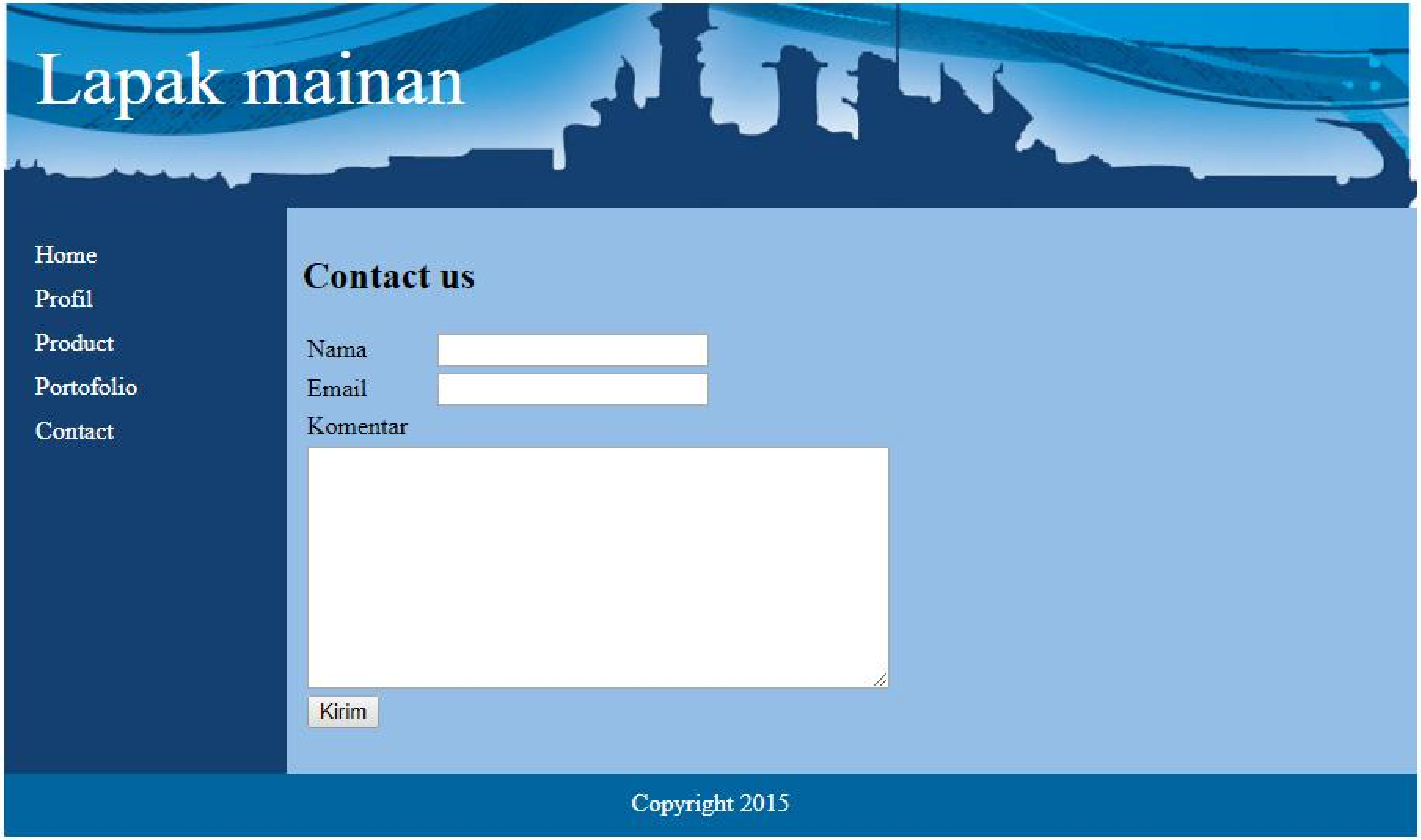
Action Attribute

respon.php



```html
<html>
<body>
    <p>Thanks!</p>
    <p>Your name has been recorded as:</p>
    <p><!--Add code here--></p>
    <a href="index.php">Reset</a>
</body>
</html>
```

Homework


Lapak mainan

- Home
- Profil
- Product
- Portofolio
- Contact

Contact us

Nama
Email
Komentar

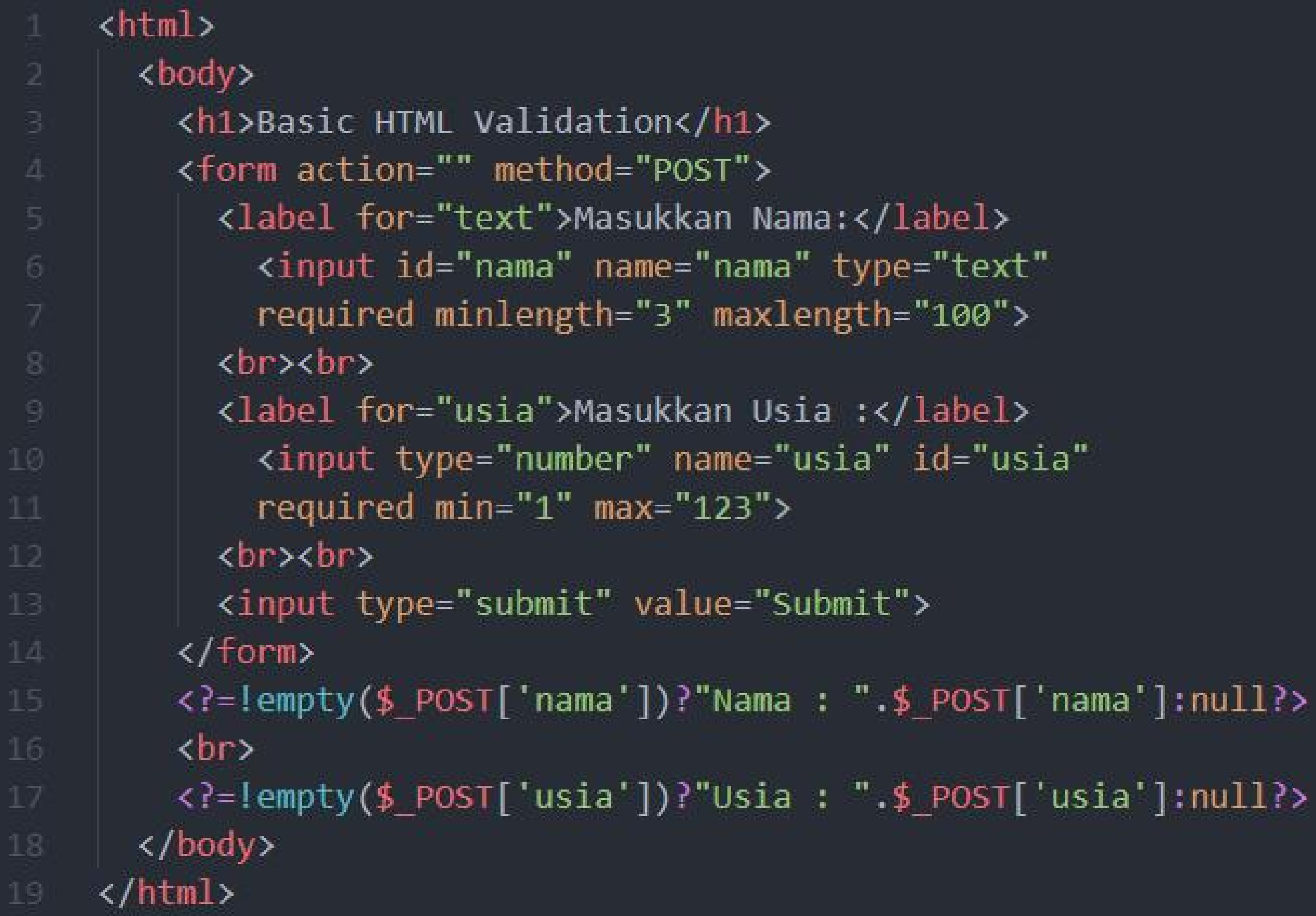
Copyright 2015

contact.php --> show user input in page

FORM Validation

- Make sure the information that will be stored in the database is accurate.
- Make sure operations that depend on the data to work.
- Keep our site secure.

Client-Side Validation



```
<html>
  <body>
    <h1>Basic HTML Validation</h1>
    <form action="" method="POST">
      <label for="text">Masukkan Nama:</label>
        <input id="nama" name="nama" type="text"
        required minlength="3" maxlength="100">
      <br><br>
      <label for="usia">Masukkan Usia :</label>
        <input type="number" name="usia" id="usia"
        required min="1" max="123">
      <br><br>
      <input type="submit" value="Submit">
    </form>
    <?=!empty($_POST['nama'])?"Nama : ".$_POST['nama']:null?>
    <br>
    <?=!empty($_POST['usia'])?"Usia : ".$_POST['usia']:null?>
  </body>
</html>
```

Server-Side Validation

- Front-end validations are easy to bypass, a malicious user can simply turn off JavaScript on their browser.
- Potential for middleman attacks in which data is changed after the request is submitted by a user but before it arrives at the server.

Server-Side Validation - Exercise

Rewrite code in the previous slide and add a server-side validation.

Basic Data Sanitizing

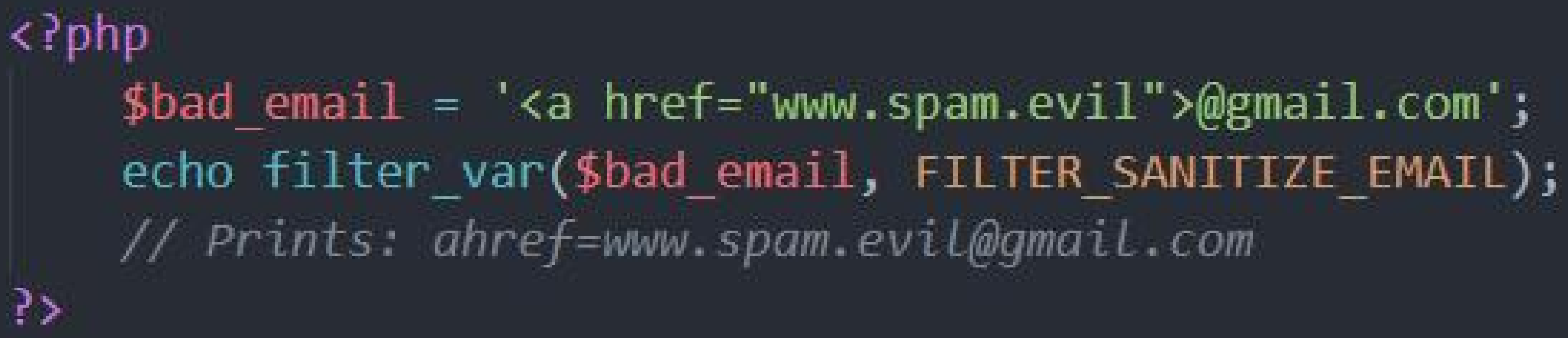
- Transform input data into a safe and standardized format.
- Use `htmlspecialchars()`
- e.g. `htmlspecialchars($_POST['nama'])`

Basic Data Sanitizing - exercise

Rewrite code in the previous slide and sanitize every input from user using `htmlspecialchars()`.

Basic Sanitization with filter_var()

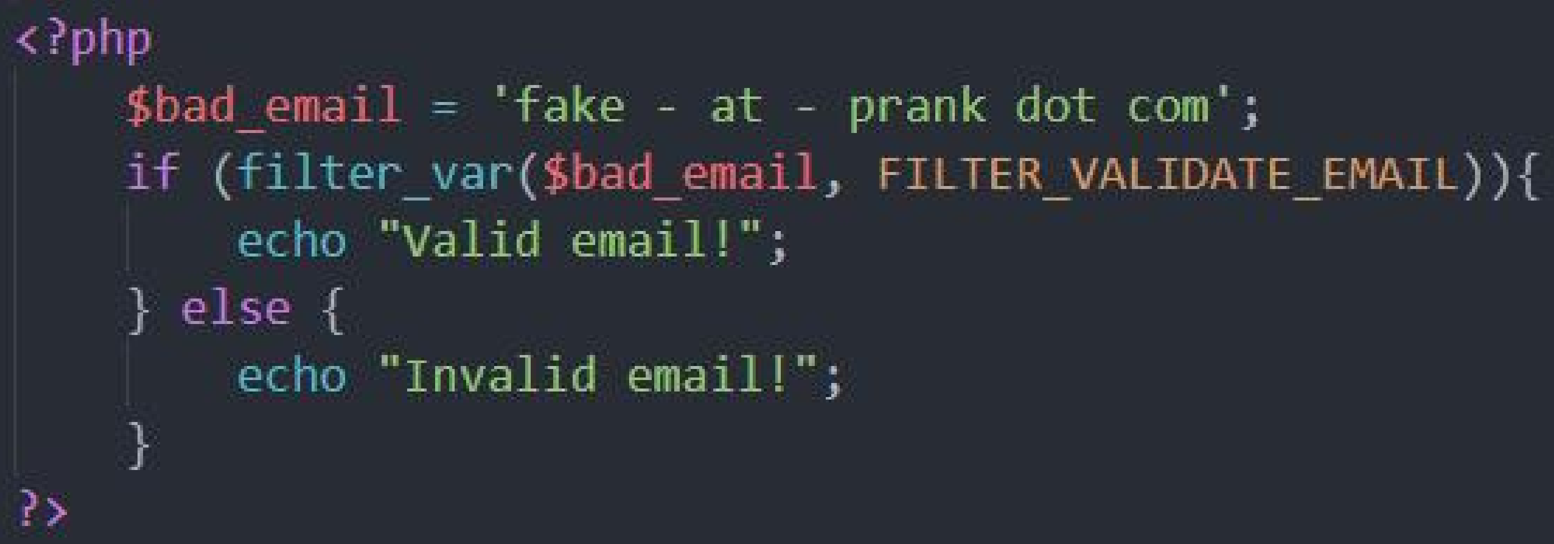
- This function operates on a variable and passes it through a “filter” that produces the desired outcome.
- As its first argument, `filter_var()` takes a variable. As its second, it takes an ID representing the type of filtering that should be performed.
- https://www.php.net/manual/en/filter.filters.sanitize



```php
<?php
    $bad_email = '<a href="www.spam.evil">@gmail.com';
    echo filter_var($bad_email, FILTER_SANITIZE_EMAIL);
    // Prints: ahref=www.spam.evil@gmail.com
?>
```

Basic Validation with filter_var()

- Validation make a False return.
- As its first argument, `filter_var()` takes a variable. As its second, it takes an ID representing the type of filtering that should be performed.
- https://www.php.net/manual/en/filter.filters.validate.php



```php
<?php
    $bad_email = 'fake - at - prank dot com';
    if (filter_var($bad_email, FILTER_VALIDATE_EMAIL)){
        echo "Valid email!";
    } else {
        echo "Invalid email!";
    }
?>
```