# HTML FORMS

PART 2

# TWO PARTS OF FORMS

## HTML form:

The user interface (client-side)

## Server-side processing:

The action (server-side)

send e-mail

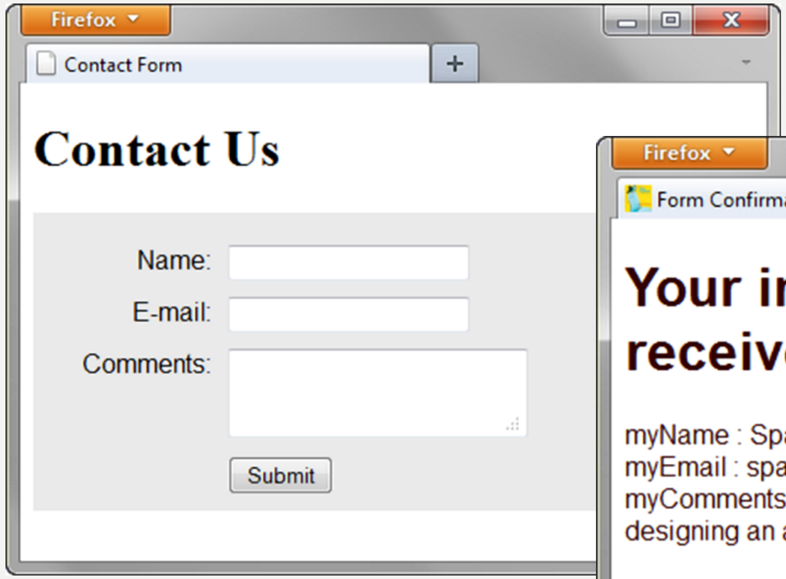
write data to a text file

update a database

performs some other type of processing on the server

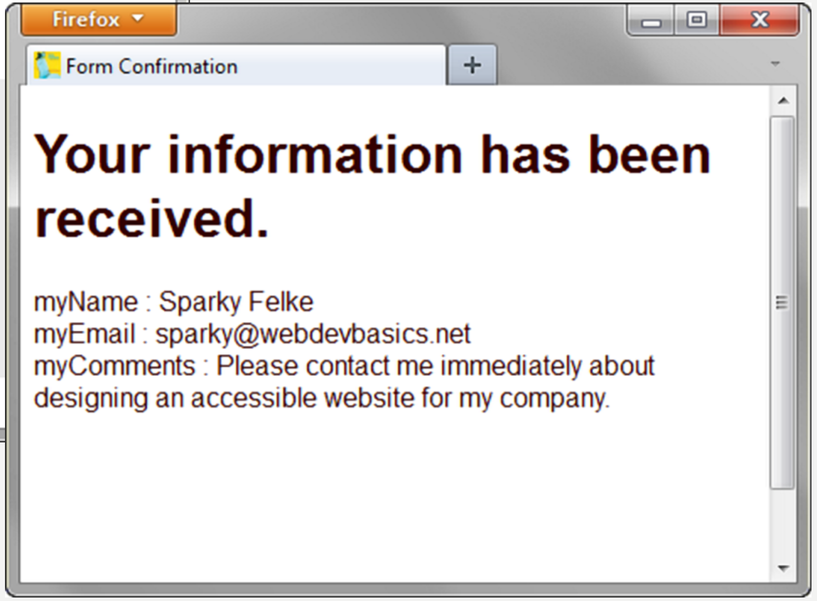
## SENDING INFORMATION TO A SERVER-SIDE SCRIPT

```
<form method="post" action="form-processor.php">
…
</form>
```



Contact Us

| | |
|---|---|
| Name: | |
| E-mail: | |
| Comments: | |
| | Submit |



Your information has been received.

myName : Sparky Felke
myEmail : sparky@webdevbasics.net
myComments : Please contact me immediately about designing an accessible website for my company.

## VARIABLES IN PHP

### Assign a variable using the assignment operator ("=")

- Anything on the right gets inserted into anything on the left
- Anything inserted replaces anything in the variable
- In PHP, all variables start with $
- E.g. $myVariable = "Hello";

### Concatenation

- Add to a variable using the concatenation operator (".=")
- Same as the assignment operator, but instead of replacing, it appends
- E.g.
  $myVariable = "Hello";
  $myVariable .= " Professor!";

## SOME USEFUL PHP COMMANDS

### mail()

- Send mail using the server's mail server software
- Uses four arguments: TO, SUBJECT, BODY, FROM
- E.g.

```
mail("someone@example.com",
        "The subject line",
        "Hello World!",
        "From:<someoneElse@example.com>"
);
```

### echo

- displays the next item on the webpage
- E.g.

```
echo "these words will appear on the webpage"
echo $someMoreWordsInaVariable
```

# COLLECT DATA FROM THE HTML FORM

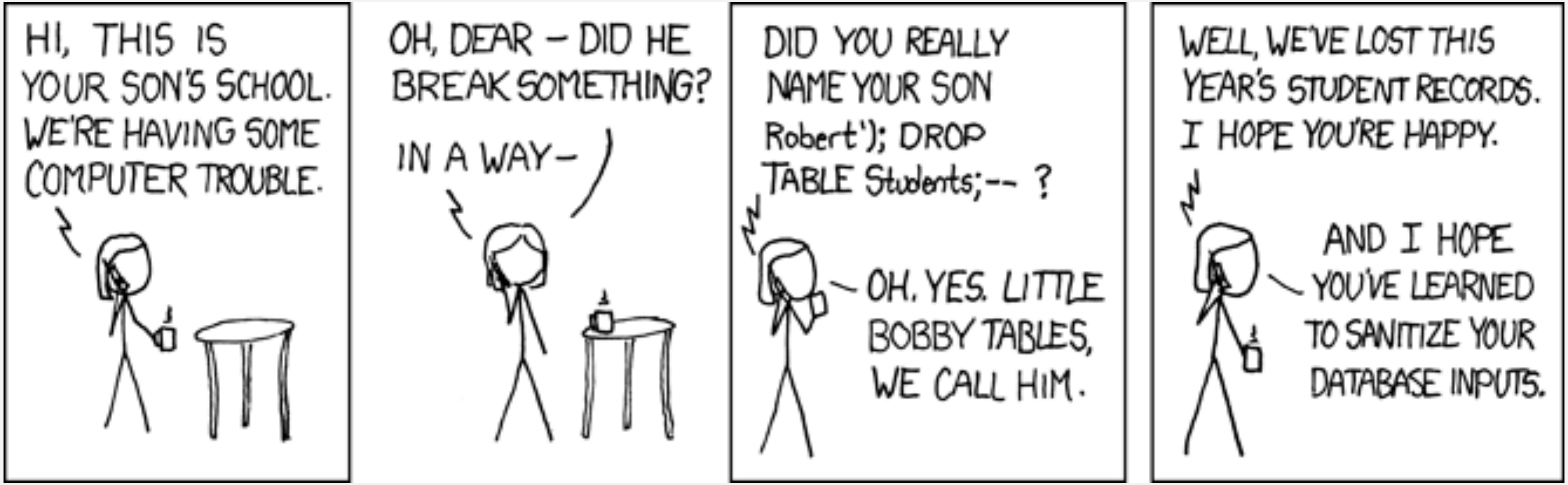
- Superglobals: **`$_POST[]`** or **`$_GET[]`**
- Pull data submitted via an HTML form
  - E.g. `$customerName = $_POST['name'];`

File 1: index.html

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

File 2: welcome.php

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


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

## SCRUB INCOMING DATA FROM HTML FORMS

- Nefarious users can trick servers to run code by entering it into a web form and submitting it
- Protect your web server by scrubbing any text inputs
- Use the PHP trim() and stripslashes() commands

```
$message = Trim(stripslashes($_POST['message']));
```

or (for database connections)…

```
$message = mysqli_real_escape_string($connection, $_POST['message']);
```