Using GET and POST with Forms

Have a question? Discuss this lecture in the week forums.

Interactive Transcript

Search Transcript English

0:08
So we just talked about how the GET request sends data in. Now we're going to talk about the POST request. And we'll draw our forms a little differently. But ultimately, it's a different form of the HTTP request, the data comes in via POST. It doesn't come in on the URL. And we'll talk all about that. So now we're going to show how we get POST data in. So, the simplest thing is you tell the form tag, please when I hit the submit button, when I hit the submit button send this data into POST via $_POST. And we'll talk about what that means kind of in a low level technical bit. And it just comes in, in the $_POST array instead of the $_GET array. So in this case, the GET is empty, there are no parameters. That's the nice thing about POST, is there are no parameters, your URLs don't look ugly, the data comes in and it magically comes in this $_POST array, not the $_GET array. So that's how we do it.

0:57
It's important to know when to use GET and know when to use POST. When you do GET the parameters are posted on the URL and users see them. The nice things is if you want them to bookmark those parameters, they can because the things that gets bookmarked when you drag a URL on your browser is the whole URL including the parameters. The POST you don't see the parameters. And so that sometimes that's exactly what you want. But they're actually sent, they're just not sent on the URL. So if we're to look at the request response cycle this is HTTP, right? We're looking for the browsers talking to the server. So in the browser and remember if we went to port 80 and we sent this GET command, the sort of an abbreviated version of it. So when you're doing a GET sending data, it says get form1.php and then send this GET parameter. And then there are some things and then you put in the blank line and then it sends it. If you're sending a POST request, it's a different HTTP verb. These things are called verbs. There's a GET verb. There's a POST verb. There's a PUT. There's a DELETE. And there's a HEAD. And there's other ones. So the POST verb says, we're going to send data differently and we don't put the parameters here. We still are sending a key of guess with 42, and there's some data. And we say this is a particular kind of data, says, this is URL encoding the data. And then it has a blank line and then it actually sends it down here. And this is why on a POST, you don't see it. It's part of a connection, this is a connection to the server, send, send, send, send, send. Send a blank line, send the data. And then that gets copied into the $_POST array. Okay, so that's the $_POST array.

2:34
So again, what we just were watching is what's coming across here. It's doing a POST. Apache sees the post, it pulls the data out and sticks it into the $_POST array for us. And then the first line of our code starts up. Now, you can choose whether or not to do GET or POST, and as a developer you're responsible for choosing the right thing. So, first off, POST is to be used when you're creating or modifying data.

3:00
Get is to be used if you're searching. So, for example, if you are selling something online and you had a catalog and you typed in a part number and you hit search, it would be in your best interest to use GET. And that's because if they bookmarked the part, you'd actually have the part number right there in the URL. If on the other hand you're taking money out of a bank and your saying I'd like to take $100 out and you hit the button. The last thing you want is that withdraw $100 to be right up on the URL because then if they bookmark that, every time they hit it they're going to withdraw $100. So, that's why we use POST when data is being changed. Also, web spiders will follow GET URLs but not generally POST URLs. And so web spiders know that if you type stuff into a form that says it's going to POST and you hit the submit something might happen. So the last thing you would like to do is you like to have a blog and you have a delete button and that's a GET request. And then web search engine goes through and tries to read all your blog post and ends up deleting them by mistake. The nerd word that we use for this is that GET URLs are idempotent, meaning that you're supposed to, if you hit the same URL, you should get the same thing back. The example of in a catalog a part number equals, should get the same part number. Now, maybe there's less in inventory, it doesn't have to be identical page. It just means the same thing, right? So, part number is, this is our skateboard, and there's 4 in stock and then, you hit refresh, it should be the same skateboard but there might be 40 in stock, right? So, that's what it means by the same thing. It's not identically HTML, it's the same thing. Another reason why you might want to use POST is there is a length limit that's unknown on GET parameters. Different browser, depending on your browser, depending on your web server.

4:45
It's not exactly a number, it's not exactly known, it's not exactly the same. But, there is a limit and so you just don't want to run into it. So we tend to like the URLs themselves to not get very long. If you're going to send a paragraph text, it would be really bad to send that on a GET request, right? But a few parameters you don't have to worry like if you have x and y and something else and a part number or whatever. But you just don't want it too big. So up next we're going to talk a little bit about how we can create different input pipes in the form.

Downloads

Lecture Video mp4

Subtitles (English) WebVTT

Transcript (English) txt

Would you like to help us translate the transcript and subtitles into additional languages?