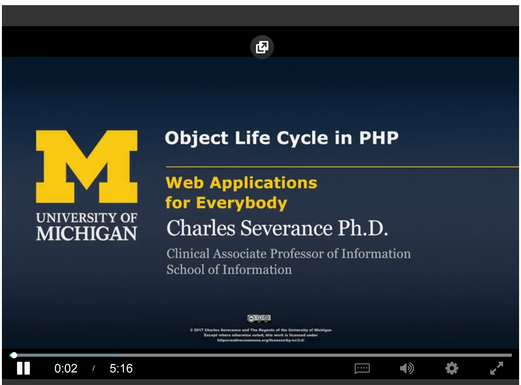
Course Materials

Lecture Content

Bonus Materials



Object Life Cycle in PHP

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

Interactive Transcript

English

0:09
Okay, so one of the things I talked about in that previous section was how cool the PHP object model is. In particular, because it's aware of sort of things like session and the kinds of stuff we use in web applications. But now I want to talk about sort of the classic things when we talk about object life cycle, about the definition of the template, the creation and initialization of the objects, and then the destruction of those objects. And we have special methods that we get called. And we saw that in some of the documentation. The constructor and the destructor, that is a bit of code that you as the object builder get to say, you know what, call me when you're setting me up or calling me when you're throwing this thing away. We use constructors all the time so that we can get our data into the right state when we've set things up. And the destructor is used less often. So like I said, the constructor is there to get us started and set up some primary instance variables so they have the right values. As we'll talk about in a bit, you can sometimes have these hidden variables. They're non-public and the constructor can set those values up to what they're supposed to be when the thing first starts. So here's a little sample, we're not going to do anything intelligent in there, we're just going to kind of show how this constructor and destructor start working. And so, we're going to have a class, our template, PartyAnimal. And it's going to have a method called __construct, it's a specially named method. And then one called __destruct. And all we're saying is, call us back when you're creating us. And then call us back when you're destroying us. So let's run the code then, right? So we print out One and we say, let's say we make a new PartyAnimal. And that says, run the constructor as that moment of creation is happening. It's before the assignment statement comes back with $x. So we say we're constructed. Then we say Two and then we're constructed again. So we're running the second one. So we're creating $y now and it constructs it. It makes the template, then calls the constructor, and then give us back the object. That's the order in which it happens. And then we say Three, and then we say Something, which just runs something, right? And so now we're at the end of the program. So we did all these things, and now we're running the end of the program. Except that this last line of the program. And so it's going to clean all of these variables up. So $x is going to go away and $y is going to go away. But before $x and $y go away, PHP calls us back for $x and gets rid of that and then calls us back to get rid of $y when there's no code writing whatsoever. Because at the end of a request response cycle, when this file is all done, then it's going to garbage collect and get rid of all these things. And so it's going to carefully call us, call our destructor.

2:39
And like I said, the constructor's used a lot, the destructor, PHP is actually more consistent about calling the destructor than other languages. Because we know when we're done with the request in PHP. And so you tend not to see destructors in other object oriented languages. Because the timing of the destructor is not as predictable as it is in PHP.

3:00
So again, object and instance.

3:03
Each instances has its own distinct variables, and you could have many instances of the same class. And so we've been doing this all along. And so let's just talk about something where we're going to set a variable in the constructor that's going to control the behavior of this application. So now we're going to pass a constructor parameter in. And that just becomes a parameter into the __construct method. And we're going to do this to set up this Hello translator.

3:30
And we're going to tell it what languages that we want to translate. And so we're going to pass that in. And PHP is going to build a Hello object. It's going to put a variable in here and we're going to have a method in here. And now this is sort of empty and now it's going to call the constructor. And the constructor's going to copy this variable that it came in through lang into the instance. So 'es' is going to be in here. And then, after the constructor is done, then this assignment finishes. And so we have variable $hi now, which $hi points now at this object that's been constructed from the template. Except we put some data in, in the constructor. We're using $this to save this one, versus $lang is the one that came in. $this is the one that's actually in the object.

4:15
So that's what happens in this line. Constructor runs, we copy a little bit of data from the parameter to the constructor into the object itself. And now we say hi->greet, which means we're just going to find the greet code within $hi. That's this code right here, greet, we're going to run it. $this points to this instance. And we can see what the current language of the current object is. If it's French, we're going to do one thing. If it's Spanish, we're going to return 'Hola'. And we're going to return 'Hello' if it's none of the above. And so in this case, because we got 'es' in this particular instance, we're returning 'Hola'. And so that's what prints out here. Okay, so we're passing stuff in, in the constructor to kind of personalize, or unique-ize, that particular object in the right way. So that's the constructor that's called at the moment of creation. You create it, you construct it, and then you give it back to us. And then we get to use it.

5:07
The next thing we'll talk about is inheritance. And that's how we take the capabilities of one object and inherit them when we make a new object.

Downloads

Lecture Video mp4
Subtitles (English) WebVTT
Transcript (English) txt

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