**Course Materials** 

**Lecture Content** 

Welcome to the Course

Object Oriented Concepts

Creating Objects in PHP

Object Life Cycle in PHP

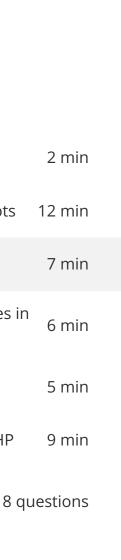
Object Inheritance in PHP

**Practice Quiz:** 

PHP Objects

Object Oriented Libraries in

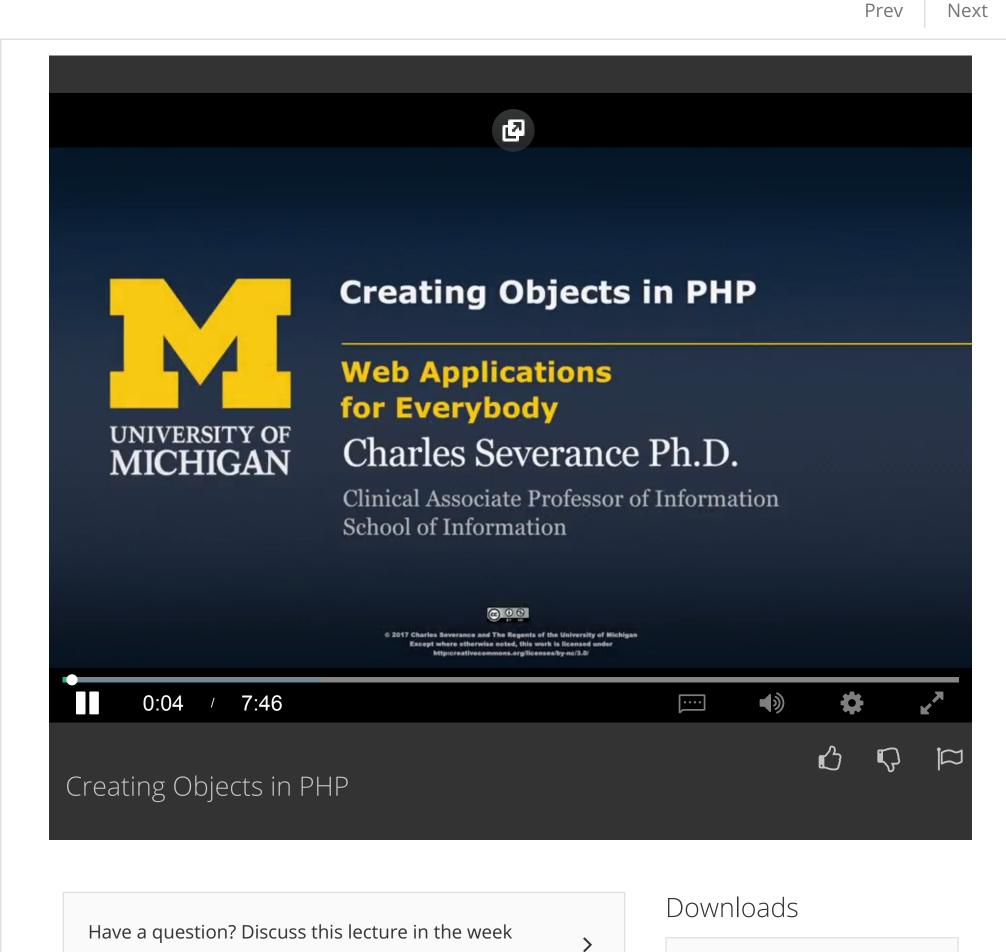
◀ Back to Week 1



**X** Lessons

**Bonus Materials** 

PHP



Interactive Transcript English -0:08

forums.

So, while I'm not going to have you build your own objects, I think it's really instructive to take a look at the syntax used to define them to help understand how they work. Again, my goal is that you are going to look at objects and you're going to use them. Read the documentation to make use of them. But let's just take a look at how you might build such a thing. But before we do that, let's kind of go back to PHP4 and the concept of data structures. I mentioned that one of the things that people love about PHP is how they're using arrays, especially key value arrays. You can sort of make data structures. And you might have just not even realize how awesome you are, that you're sort of making clever data structures by making arrays with keys and values that have conventions. And so, let's say, we're going to deal with people's names. And sometimes we have a full name and sometimes we have a first name and last name, right? We decide to call the last name family name because it's culturally incorrect to call it last name because family name in some cultures comes first and given name comes second. So, that's the cool thing we're going to do. So, we're going to have some objects. Some objects are going to have a full name. Some are going to have a family and given name. And then, every object is going to have a room. Now, that's not really an object, it's just kind of using object terminology. So, we've created data structures. And the problem is that, we don't want to print people's names out. And so, we are going to want to write some code. And so, we're going to have to write a function called dealing with the two variations of these objects, which are really just arrays. We're going to look through this, because we're going to print these things out so much and there's going to be thousands of these things, right? Right now, we got two of them, but there'll be thousands, and they'll all be different. And, why don't we write a bit of code that handles the fact that sometimes, we use the full name and sometimes the given name. So, let's write a thing that takes this as parameter, person as parameter, checks to see a full name is set, if so, it returns the full name. And, if indeed Instead, the family name is set and the given name is set, we're going to concatenate the given name and the family name. There might be something more complex to say which culture you're in and then concatenate the given name and the family name. And, or family name and given name, in some order. So, assume there are some non-trivial complexity in here. So, there's code. We don't want to just do it over and over. And, now we can pass Chuck in. Let's get print Chuck's person's name and let's print Colleen's person name. All this code runs, outcomes Colleens van Lent and Chuck Severance the right way. And, we have these sort of variations, right? And, that's the non homage or any way to do it, right? You just make an array and maybe give yourself a little support routine to save yourself re-using this code. But now, if we were to recast this as Object Oriented pattern, what we would do, is we would create a template. And say, persons look this way. Okay. And so, we are going to use class. class is a new keyword that we're seeing, class is kind of like function. We're defining a function. So, we're going to make a template. Now, this code doesn't run. It's parsed and looked at but, the side effect of this is, there's going to be added a new Person which is a template. It's not really code, it just grabbed all this and it put it in. And then, what we're going to do is, we're gonna put some variables in it. So, I said that an object is data plus code. So, we're going to say there's a attribute of \$fullname and attribute a \$givenname and attribute of \$familyname and \$room. And, there are

variables in every Person object. And, we're going to start

function that is part of every Person object. And, it looks like

\$this. Okay. \$this, is a predefined constant that you can only

use inside of a method that's inside of a class. Okay. And,

before, or I had one object which was \$now, and another

right? What happens is, this code get\_name() is part of both

of these, right? So, get\_name(), is really in both of these. But,

when we run it in this one, it needs to look at the data that's

inside of itself. And, when we run it inside this one, it needs

to look inside data inside this and so on. So, what happens

always points to the one that we're in right now. So, it's like

is, it's as if there's a parameter, called \$this. And \$this,

objects, thousands of Person objects, sort of floating

around. We're still in the class. But we'll have one called

Chuck and we'll have one called Colleen. And we'll make a

could be thousands of these things. This always points to

the one that you're in, right? So, when you're executing

code in this one, it points to the data in this one. When

you say the one that's in this instance or this object. So,

you're executing code in this one, it points to that one. So,

that's really important. If you know about Python, there was

know what I mean, we're not talking Python right now. Most

Java uses this with no dollar sign and that's where this came

this concept of self, in Python. It's the same thing. But, you

objects or languages have some way of doing this. I think

from. I think it was borrowed from Java, if I'm not

that's just a template. It's a shape that we can then

mistaken. Okay. So, here comes our class. It's got four

pieces of data and one method. One bit of code in it. And,

instance. So, then we run. We're going to like, let's make

one of these things. So, we take the little template, we make

an actual class. It has four pieces of data and one function

inside of it. And then, we're going to put that in \$chuck. So

minus>. But, it's really just an operator that says, go and

find full name, which is like this. And, put Chuck Severance

in here. So fill it, Chuck Severance in there. And then, the

room is going to be in 44 blah blah North quad. And

then, we're going to make another one. So, we're going to

have four pieces of data in it and one function in it. And,

we're going to say the familyname is van Lent and we're

going to say the givenname is Colleen. And, we're going to

give her that 34 North quad, right? Okay. So now, we have,

and that's going to be this guy. So we filled it up, we build it

and then we filled it up with stuff, right? So, that creates this

object. And now, I'm going to run the get\_name() function,

inside of \$chuck. Which is this function right here. It's

identical to this one here, but except that, the \$this is

pointing up to that data. So, when this code runs, \$this

fullname is this one. And the room etcetera, is this one right

here. And so, it runs. We do the get\_name() and says, oh,

well this one, fullname is not false so, we're going to return

Chuck Severance. Then, it goes back. Code comes back and

then, runs this. Let's go find the \$colleen object, which is

is going to point at this data so, it runs. And in this case,

fullname is false but familyname is not. And so, it returns

this code. And so, it returns the code, print that out and out

comes Colleen van Lent. I hope you understand it. Go back

namespaced. And, \$this is a powerful concept. The \$this is a

powerful concept. Okay. So, now that we talked about

making one and you sort of see some new syntax, we're

documentation, as if you're going to have to use objects.

and watch it again. It's really just code and data,

going to talk a little bit more about reading

Okay.

right here, and run the get\_name() code there. Except, this

take this template again and make another one. It's going to

\$chuck points to this one. And, we can access the data

items with this little arrow operator, which is

bunch but we could be making them in a loop and so there

this one. So, there could be thousands of

what is \$this? Well, if you remember the picture I drew

object which is \$nextWeek. Now and then nextWeek,

every other function. You can even have parameters which

them to false in the beginning. And then, every Person

object will also have a built in function. And so, this is a

we'll see in a bit, except that it uses this weird little

Lecture Video mp4

Subtitles (English) WebVTT Transcript (English) txt Would you like to help us translate the transcript and subtitles into additional languages?