**Object-oriented programming syntax**

In this video, you'll see what a class and object look like in Python. In the next section, you'll have the chance to play around with the code. Finally, you'll write your own class.

**Function versus method**

*In the video above, at 1:44, the dialogue mistakenly calls* ***init*** *a function rather than a method. Why is* ***init*** *not a function?*

A function and a method look very similar. They both use the def keyword. They also have inputs and return outputs. The difference is that a method is inside of a class whereas a function is outside of a class.

**What is self?**

If you instantiate two objects, how does Python differentiate between these two objects?

shirt\_one = Shirt('red', 'S', 'short-sleeve', 15)

shirt\_two = Shirt('yellow', 'M', 'long-sleeve', 20)

That's where self comes into play. If you call the change\_price method on shirt\_one, how does Python know to change the price of shirt\_one and not of shirt\_two?

shirt\_one.change\_price(12)

Behind the scenes, Python is calling the change\_price method:

def change\_price(self, new\_price):

self.price = new\_price

Self tells Python where to look in the computer's memory for the shirt\_one object. Then, Python changes the price of the shirt\_one object. When you call the change\_price method, shirt\_one.change\_price(12), self is implicitly passed in.

The word self is just a convention. You could actually use any other name as long as you are consisten, but you should use self to avoid confusing people.