Sometimes you need to verify a method on the test that is returning the right type. So let's take a look at this CustomerController class in the fundamentals folder. This is the simplified example of a controller in ASP.NET MVC applications. If you don't have experience with ASP.NET MVC it doesn't matter. So let's see what is happening here. You have this class called ActionResult, and this class has two derivatives, you have NotFound which is derived from ActionResult, and also Ok.

So, in our controller class, you have this method, GetCustomer, gets an ID. If ID is 0 it returns a new instance of the NotFound class.

Otherwise, it returns a new instance of the Okay class.

So because both these classes derive from action result, the return type of this method is ActionResult. So, how many tests do we need here, we need two tests, because we have two execution paths, if ID is 0, we want to make sure that the object that this method returns is a NotFound object, otherwise, it's an Ok object, so, back in our unit testing project, I'm going to add a new test class, called CustomerControllerTest.

Apply to TestFixture attribute with a Test method. So, the method is GetCustomer, one scenario is IdIsZero, should return ReturnNotFound. And of course the other scenario is public void GetCustomer IdIsNotZero, it should ReturnOk. Now, in this lecture, I'm just going to implement one of these test methods, and I leave the other two as an exercise. So we create controller object a new CustomerController, and call GetCustomer, and pass 0 as the argument.

We get the result, now we want to Assert that this result is a NotFound object. In NUnit there are two ways to write such Assertions. One way is like this: Assert That result Is TypeOf, and this is a generate method, so we supply the generate argument, in this case, Not Found. This is the Assertion that you would use most of the time. We have another way to write this Assertion, so let me duplicate this line, instead of the type of method, we use, Instance Of. So, what is the difference? This InstanceOf method, means the result and NotFound object, are one of it's derivatives, whereas TypeOf, ensures that the result is exactly NotFound object. So, this means the result should be a NotFound object, and this Assertion means the result should be a NotFound object. or one of its derivatives. In this particular implementation, I'm going to use the first Assertion, because I want to make sure the result is exactly a Not Found object. But be aware of this other way to write this Assertion, maybe you will need these in some particular scenario in your application. So I'm going to comment this out because we don't need it.