Flex in a Week, Flex 4.5

Video 2.11: Generating value objects using Flash Builder data services

In the last exercise, you manually created a class, named Employee, to create value objects.

This wasn't such a hard task, but you can save some time by taking advantage of the Flash Builder data services tools.

In this video, you will learn how to have Flash Builder generate a value object class for you.

To have Flash Builder create a value object, you must first create a service using the Data Services view.

I am clicking the Connect to Data/Service link which opens the dialog to select the service type.

You can generate value objects from any of these data services.

In this video, I will use a web service.

I am selecting the WSDL option and clicking Next.

I am adding the absolute path (http://www.adobetes.com/f45iaw100/remoteData/ex2_09.cfc?wsdl) to the WSDL document in the WSDL URI field.

Notice that when I entered the path, these fields were automatically filled out.

The service name represents the identity of the service within Flash Builder.

The service package is where the Flash Builder generated service files will be placed.

The data type package is where the generated value object files will be placed.

I am clicking Next.

Flash Builder introspects the WSDL and displays the operations I can import.

I am unchecking updateEMPLOYEES, leaving getAllEMPLOYEES, and clicking Finish.

In the Data/Services view I can see my service, Ex2_09, and the operation I selected.

In the Package Explorer view, there is a new directory named services.ex2_09.

When I expand the directory, you can see all the files that configure the data service.

_Super_Ex2_09.as, is an internal class that has the service connection logic.

You should not modify it because Flash Builder will overwrite the file if you regenerate the service.

You can change the Ex2_09.as file, since Flash Builder will not overwrite it.

Back in the Data/Services view, note that the service is returning an Object data type.

I am going to change this so it will return the Employee data type and generate the value object class for me.

I am right clicking on the getAllEMPLOYEES operation and selecting Configure Return Type.

I am going to allow Flash Builder to auto-detect the return type if there is one.

In this case I do not need to provide authentication to access the operation so I am clicking Next.
We already know that the WSDL is returning generic objects so I will assign a new data type named Employee.

It will be an array of Employee objects.

I am clicking Finish.

You can see that the return data type is now Employee.

The brackets mean that an Array of Employee objects will be return with this operation.

In the Package Explorer view, you see that a new directory was created, named valueObjects.

It contains three new files.

The _Super_EMPLOYEE.as and _EMPLOYEEEntityMetadata.as are internal class files that have logic for properties and metadata.

Do not modify these files.

The file, Employee.as is a public class file that you can customize by overriding methods.

This file will not be overwritten if the value object is regenerated.
This is the main application file for this video.

You can see that I am using the MXML WebService component to access the data instead of the data service that was generated by Flash Builder.

I am doing this to illustrate that you can use the generated value object without having to use the generated service.

I am adding a breakpoint on the result handler function and clicking on the Debug button.

You can see that the data is an ArrayCollection of generic objects.

Remember that ColdFusion is a case-insensitive technology, so it converts all the properties of the object to uppercase letters.

As in the previous exercise, in order to create a typed model, you must loop over the retrieved data and use them to populate value object properties.

I will place the returned data in this employees class property.

But first, I need to create all of the value objects.

In the result handler, I am creating a local variable named employeeData that I am data typing to the ArrayCollection class.

For its value, I am assigning the event.result property as an ArrayCollection instance.

After this assignment, I am using content assist to insert a code template creating a for each loop by typing CTRL+Space and for e.
I am modifying the generated for each loop so that it loops over each item in the employeeData ArrayCollection using the iterant emp, data typed to the Object class.

This code will loop over the employeeData object and reference each item of the data in the array as emp on every iteration of the loop.

As I loop over each item in the employeeData ArrayCollection, I want to convert the data from a generic object into a typed data object.

Explicitly, I want to create instances of the Employee value object.

Above the loop, I'm creating a local variable named employee, data typed to the Employee class.

Note that Flash Builder imports the class forme.

Within the for each loop, I am instantiating the employee variable using the new keyword and calling its constructor.

Next, I am populating the FIRSTNAME property for the instance with the emp.FIRSTNAME value.

Remember that employee is the typed Employee object that I am creating, while emp is the untyped data from the server.

To save time, I am pasting all of the other properties.

Currently, this code just creates a new employee object on every instance of the loop.

Now, I need to take that employee object and populate the employees ArrayCollection class property with it.

As the last line in the for loop, I am typing employees and then a period.

Then I am using the addItem() method of the ArrayCollection class to add the new employee value object to the collection.

I am making sure that my breakpoint is still defined on the closing brace of the result handler.

When I debug the application and drill down in the application objects to find the employees object, you can see that it is populated with an ArrayCollection of typed data based on Flash Builder's generated value object.

For your next step, work through the exercise titled "Using the Data Services wizard to connect to a service".