Flex in a Week, Flex 4.5
Video 3.04_http: Making an HTTPService request with parameters

In the first and second Days of training, you learned how to request data from the server to use in your application.

In this video, you will learn how to send data to the server using the Flex HTTPService component.

You will also handle the confirmation message from the server that returns the record id for the data that was inserted into the database.

In earlier videos, you retrieved data from the server and bound it for display to the Employee DropDownList control of the Employee Portal: Vehicle Request Form.

In this exercise you will send the data entered into this form to the server so that it can be inserted into a database.

When I select an employee from the DropDownList control, this pre-fills the TextInput control for the employee's office phone number.

I'm adding a mobile phone number and then selecting the pickup and return dates from the DateChooser controls, .

When I submit the form to the server, all of the data is sent, the server submits the data to the database, and then the record id is returned to the Flex application, which displays it in this Alert dialog box.

This is the main application starter file.

All of this code should be familiar to you since you built it over the last three days of training.

The HTTPService object, retrieves the employees data from a remote server and then loops over it to convert it from an ArrayCollection of generic objects into an ArrayCollection of Employee value objects based on this Employee class in the valueObjects package.

On each iteration of the loop, the employee instance is added to the employees class property, which is bound to the employees property of the VehicleRequestForm custom component, which is the view in our MVC implementation.

In this component, the data is bound to the DropDownList control.

That is how the data is coming into the custom component.

Now let's review the data flow as it leaves the custom component in an event object to be handled by the main application.

On the form's submitButton click, the handler instantiates the VehicleRequest class to create a value object that contains all of the data from the form.

This data is then passed as an argument into a custom event, which packages it in the event object.

This VehicleRequestEvent object is dispatched to the VehicleRequestForm instance in the main application.

I am Control + clicking on its registered event handler so that you can see that the application currently just shows an alert message that the form was submitted.

I have adding a breakpoint to the closing curly brace of this handler and am now debugging the application.

In the browser, I am filling out the form and then submitting it.

The form data is dispatched from the custom component to the main application and then handled by the event handler that hits the breakpoint.

When prompted, I am switching to the Debugging perspective and drilling down in the event object to see the form data.

Now that I have the form data in the main application, I can send it to the server.

In the Declarations block, I already have one HTTPService object, but I want to create a second instance to send data back to the server.

I am typing HTTP and pressing CTRL+Space twice to show code templates in the content assist tool and selecting the HTTPService object code template.

To the new HTTPService object, I am modifying the id property so its value is vehicleService and I am changing the url value (http://www.adobetes.com/f45iaw100/remoteData/addVehicleRequest.cfm) to point to a ColdFusion page that will accept the data.

Remember that, although we are using a ColdFusion server for this training series, you can use any backend technology – like Java, JSP, .NET or PHP – that can accept the data in an HTTP operation.

The Flex code is the same regardless of the backend technology.

I am using Flash Builder content assist to create a result handler in the result event.

Lastly, I am deleting the fault, resultFormat, and showBusyCursor properties from the HTTPService object.

Most server-side processes usually define the type of parameters that they accept and the type of data that will be returned.

There are two ways to send parameters on an HTTPService request:

Parameter binding and explicit parameter passing.
In parameter binding, you can directly bind the values from UI controls or data models directly to the parameters.

This method of sending data is only available if you have created the HTTPService object in MXML and not in ActionScript.

The advantage of using this method is that you can apply validators to the parameter values before submitting them to the service.

To implement this method, simply create a request property and pass one or more parameters as XML nodes.

You then actually send the data to the server using the service object's send() method.

In this code, you can see the request property defined as a property of the HTTPService object.

You can pass one or more values to the server by directly binding them in XML nodes.

Since this service is named employeeService, you send the data to the server on some user or system event by referencing the employeeService.send() method.

The second method for passing data in HTTPService calls is explicit parameter passing.

You provide the parameters for the service in the service's send() method as you would provide arguments to a function.

The downside of this method is that you cannot use automatic Flex data validation, which you will learn about later in this Day of training.

This first example passes the employeeID parameter, with a value of 4, as a name/value data pair.

Alternatively, you can first create an object and then pass that object in the send() method as shown in this second example.

This is the way that I will send the data to the server for the Company Vehicle Request Form.

In the custom event handler for this custom component, I am adding the vehicleService.send() method.

You just saw in the debugging session that the event object contains the VehicleRequestData object with all the form data.

I am adding event.vehicleRequestData to the send() method and then saving the file.

I am selecting the Network Monitor view and maximizing it so that I will be able to see the audit trail more clearly.

I am enabling the tool and then running the application.

You can see that the first HTTPService call retrieves the data for display in the DropDownList control.

I am selecting an employee from the DropDownList and selecting a pickup and return date from the DateChooser control. I am leaving the mobile phone field blank and submitting the form.

The application displays a submission Alert message that confirms that I hit the submit buttons, but I also get a runtime error.

When I return to Flash Builder, you can see that a second service request is made for the POST operation, but there is an error.

I am opening the Response tab and switching to Raw view.

The first line tells me that I have experienced the 500 Internal Server Error.

As I scroll down, I see a message that says "Field 'VEHICLEREQUESTS.MOBILEPHONE' cannot be a zero-length string.

I am minimizing the Network Monitor view and returning to my code.

I am going to add a fault handler to show an Alert message when the Mobile Phone field is left blank.

To the vehicleService HTTPService object, I am adding a fault event and using the content assist tool (CTRL+Space) to generate the fault handler.

I am locating the generated fault handler in the Script block and deleting the generated code stub.

Within the function, I am adding the Alert.show() method to display the event.fault.faultString value as the text in the Alert pop-up.

I am giving the Alert message a title of Fault Information for POST, saving the file and running the application.

Now when I fill out everything, but the Mobile Phone field in the form and submit it, I see an alert message.

I want to display an alert message with the record id when the data is returned from the server successfully.

So, in the result handler, I am typing if and pressing CTRL+Space twice to show code templates in the content assist tool. I am selecting the template for the if statement.

I am setting the if statement to evaluate the event.result.vehicleData.vehicleRequest.id value.

Within the if statement, I'm adding an Alert.show() method that states "The request was submitted. The record id is " and then concatenating the event.result.vehicleData.vehicleRequest.id.

I am saving the file, running the application and then filling out the form.

I’m going to select an employee from the DropDownList, put some string data into Mobile Phone field and then select some dates.

When I submit the form, I see the record id displayed.

For your next step, work through the exercise titled “Passing data to the server with the HTTPService class”.