

# Populating a Control from an HTTPService Request

## Create an assets folder

1. Unzip the **adobeODTAssets.zip** to the **C:\adobeFlexTraining\AdobeODT\src** folder.

You should see that an **assets** folder is created with an image and an XML file.

## Create a Remote object

2. Remove the current `roomList` **Array** code.
3. Within the **Script** block after the current import statement, import the `ArrayCollection` Class.

```
import mx.collections.ArrayCollection;
```

4. Create a private variable named `roomList` datatyped as `ArrayCollection`.

```
private var roomList:ArrayCollection;
```

5. Bind the `roomList` variable by adding the `Bindable` metadata tag.

```
[Bindable] private var roomList:ArrayCollection;
```

6. After the **Script** block, add an `HTTPService` tag. Add the following properties:

```
id = "rooms"
```

```
url = "assets/roomList.xml"
```

```
<mx:HTTPService id="rooms" url="assets/roomList.xml" />
```

## Trigger the HTTPService call with a system event

7. Locate the beginning **Application** tag. Add a `creationComplete` property with the `init()` method for the value.

```
creationComplete="init()"
```

## Create the system event handler

8. Before the end of the **Script** block, create a private function named `init()`. It takes no parameters and returns void.

```
private function init():void{  
}
```

9. Within the function invoke the `send()` method on the rooms **HTTPService** object.

```
rooms.send();
```

## Create an HTTPService fault event and handler

10. Locate the HTTPService tag. Add a fault event that is handled by the `httpFaultHandler` method. Pass the event object as the only parameter.

```
fault="httpFaultHandler(event)"
```

11. After the last import statement in the **Script** block, import the `FaultEvent` class.

```
import mx.rpc.events.FaultEvent;
```

12. Also import the `Alert` class.

```
import mx.controls.Alert;
```

13. Within the **Script** block, add a private function named `httpFaultHandler` that takes an event parameter datatyped as `FaultEvent`. It returns void.

```
private function httpFaultHandler(event:FaultEvent):void{  
}
```

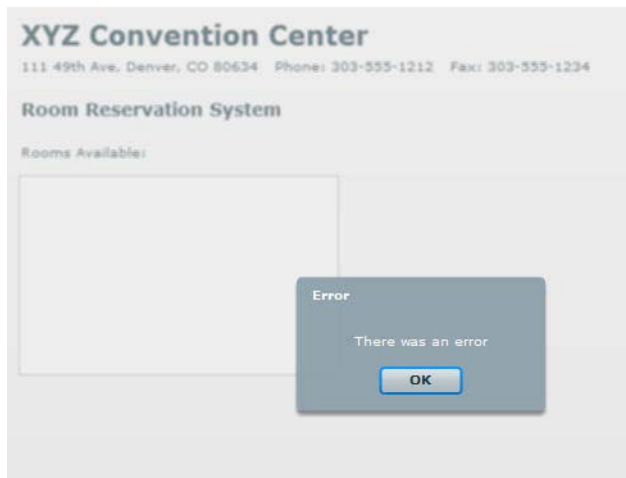
14. Within the function invoke the `show()` method on the `Alert` class. Pass two parameters: "There was a problem" and "Error".

```
Alert.show("There was a problem","Error");
```

15. Locate the **HTTPService** call, change the `url` value to `assets/roomList.xml`.

16. Save and run the file.

You should see an alert message pop up.



## Create an HTTPService result event and handler

17. Locate the **HTTPService** tag.

18. Change the `url` value back to `assets/roomList.xml`.

19. Add a result event that is handled by an `httpResultHandler` method. Pass `event` as the only parameter.

```
result="httpResultHandler(event)"
```

20. After the last `import` statement in the **Script** block, import the `ResultEvent` class.

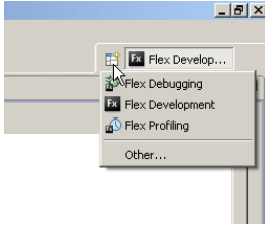
```
import mx.rpc.events.ResultEvent;
```

21. Before the end of the **Script** block, create a private function named `httpResultHandler` that takes an event parameter datatyped as `ResultEvent`. The function returns `void`.

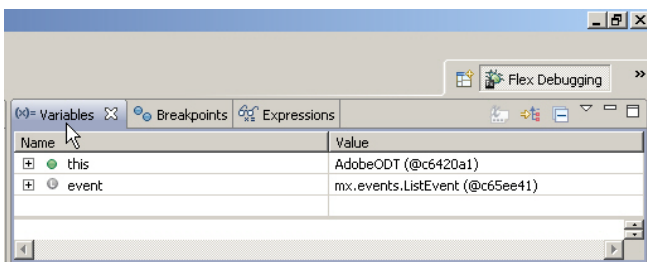
```
private function httpResultHandler(event:ResultEvent):void{  
  
}
```

22. Add a **break point** to the line with the closing curly brace by double clicking the marker bar next to the line number associated with the brace.

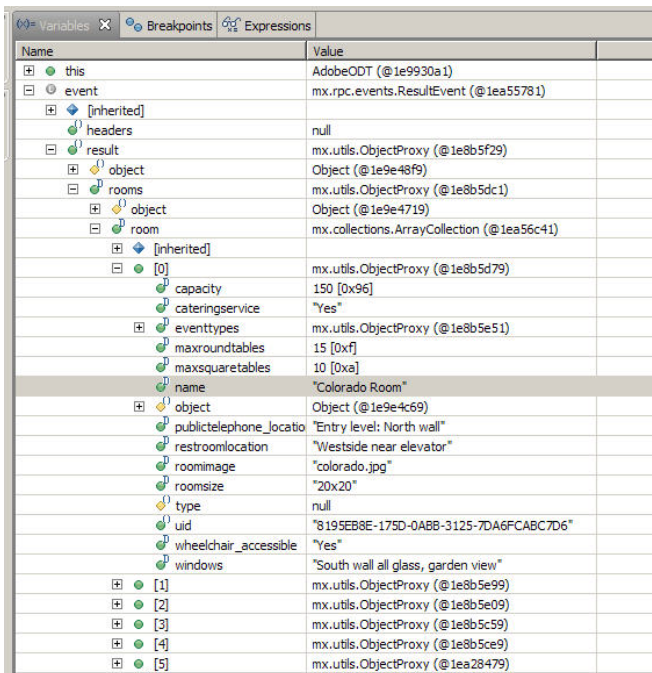
23. Switch to the **Flex Debugging** perspective.



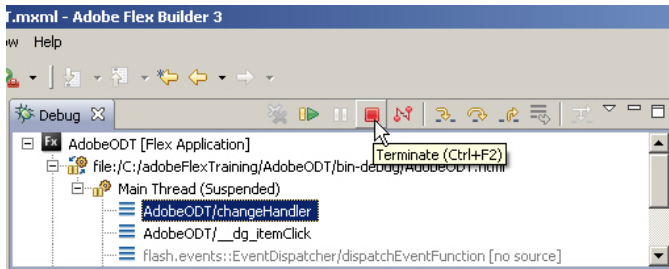
24. Save and run the file.
25. The **creationComplete** system event will trigger the **HTTPService** call and return you to Flex Builder.
26. Double click the **Variables** view to expand it.



27. Expand the **event** variable, and then expand the **result** variable.
28. Expand the **rooms** variable and then expand the **room** variable to see the **ArrayCollection**. Expand an index to see the variables associated with each room.



29. Stop the debugging session by clicking the red square.



30. Return to the **Flex Development** perspective.
31. Within the **httpResultHandler** function assign `event.result.rooms.room` to `roomList`.

```
roomList = event.result.rooms.room;
```

32. Save the file and run.

You should see `[object Object]` in the **List** control.



33. Locate the **List** control and add a `labelField` property with a value of `name`. This value references the `name` variable in the **ArrayCollection**.

```
<mx:List x="10" y="134" id="dg" width="250"
  itemClick="changeHandler(event)"
  dataProvider="{roomList}"
  labelField="name"></mx:List>
```

34. Save and run the file.

You should see six rooms in the **List** control.



**XYZ Convention Center**  
111 49th Ave, Denver, CO 80634 Phone: 303-555-1212 Fax: 303-555-1234

**Room Reservation System**

Rooms Available:

- Colorado Room
- Mile High Room
- Bronco Room
- Greeley Room
- Boulder Room
- Denver Room