

Creating a PHP Service



- ▶ To develop server side files in Flex Builder you have a few project setup choices.
- ▶ Flex Builder provides wizards to create projects for different server types:
 - Other/None - This is the basic Flex Project
 - PHP
 - LiveCycle Data Services
 - ColdFusion
 - ASP.NET
 - J2EE
- ▶ To create a Flex Project with a server type of PHP
 - Create New Flex Project
 - Enter a project name
 - Fill out the wizard properties

Understanding the project type

- ▶ This type of Flex Project is not really that different
- ▶ The wizard helps define the following project property values:
 - Property Location
 - Application Type
 - Server Technology
 - Server Location
 - Compiled Flex application location
- ▶ The location of the project does not have to be the same as the output folder
 - The output folder is typically below your web root
 - The project location is typically not below the web root to hide any source files.

Note: The Flex Project with server type PHP is good for beginning applications that do not have complex deployment requirements. You should make any PHP calls from the RPC services in your application relative to the project root.

Walkthrough 4: Creating a PHP Service Class



Steps

In this walkthrough, you will perform the following tasks:

- ▶ Create a php class inside a Flex Project with server type PHP
- ▶ Run the php file from the web server

1. Make sure your web server is running by going to **http://localhost/** in a web browser.
2. You should see a XAMPP log and text.

Create the Flex Project

3. Select **File > New > Flex Project**
4. Set the **Project Name** property equal to `PHPServices`.
5. Set the **Project Location** set to `c:\max2007\ra101h\services`.
6. Set the **Application Type** to **Web Application**.
7. Select **PHP** from the **Application server type** drop down menu.
8. Click the **Next** button.
9. Set the **Web Root** property equal to `c:\xampplite\htdocs`.
10. Set the **Root URL** property equal to `http://localhost/`.
11. Note the **Compiled Flex application location**.
 - Output folder: `C:\xampplite\htdocs\PHPServices-debug`

Create the PHP class

12. Select the **src** folder.
13. Select **File > New > File**
14. Set the **File Name** property equal to `WT4.php`.

15. Open the file and input the following code

```
<?php
header('Content-type: text/xml');
$xml = '<?xmlversion="1.0" encoding="utf-8"?>';
$xml .= '<results>';
$xml .= '<restaurant> <name>Lumiere</name>
        <city>West Newton</city></restaurant>';
$xml .= '<restaurant><name>Baraka Cafe</name>
        <city>Cambridge</city></restaurant>';
$xml .= '</results>';
echo $xml;
?>
```

16. Open up a browser and run the PHP script.

- <http://localhost/PHPServices-debug/WT4.php>

Demonstration 4: Database Driven Services



Steps

In this demonstration, your instructor will perform the following tasks:

- ▶ Explain the use of Zend Framework to provide database abstraction and REST services.

1. Run the PHP service located at.
 - <http://localhost/services/restaurant/?method=getRestaurantList>
2. View the structure of `index.php` and the Zend REST server class.

Retrieving Data from a Remote Service



- ▶ Use the `<mx:HTTPService>` tag to retrieve remote data from a REST-ful web service
- ▶ To use the `HTTPService` class
 - Supply a `url` location
 - Call a remote method on some event
- ▶ To retrieve data at application startup
 - `load` event of `HTTPService` class itself
 - `creationComplete` event of `Application`
 - User event during application interaction

Using Returned Results

- ▶ Results returned in the `lastResult` property of the `HTTPService` call
- ▶ Use the returned data as `httpServiceInstance.lastResult`
- ▶ Bind the returned result to appropriate controls
- ▶ Later in the session will use a `result` event to handle results
 - A better practice

Walkthrough 5: Populating a ComboBox from HTTP Service Data



Steps

In this walkthrough, you will perform the following tasks:

- ▶ Retrieve restaurant information use an `<mx:HTTPService>` object
- ▶ Populate a ComboBox with data returned from the web service

1. Browse to the URL
`http://localhost/services/restaurant/?method=getRestaurantList`.
2. After you see the list of restaurants, copy the URL.

Create the http service and call a method

3. Open **WT5.mxml**.
4. Insert an `<mx:HTTPService>` tag with an id of `restaurantService`.
5. Set the `url` property equal to the URL you copied in step 2.
6. Set the `resultFormat` property equal to `e4x`.
7. Use the `creationComplete` event of the Application component to call the http service.

```
creationComplete="restaurantService.send()"
```

Use the results in to populate a ComboBox

8. Insert an `<mx:ComboBox>` tag.
9. Bind the `dataProvider` property to the `lastResult` property of the method called from the web service.

```
dataProvider="{restaurantService.  
lastResult.restaurant}"
```

10. Run the application.
You should see the ComboBox populated with `<restaurant>...</restaurant>` since you did not state which property of the xml object should be displayed.
11. In the ComboBox, set the `labelField` property equal to `name`.

```
labelField="name"
```

12. Run the application.
You should see the restaurant names now appear in the ComboBox.

Walkthrough 6: Using a result Event with a HTTPService object



Steps

In this walkthrough, you will perform the following tasks:

- ▶ Add a result event to a HTTPService object
- ▶ Create a handler for the event

1. Open WT6.mxml.

Create the <mx:Script> block and a variable

2. Just below the <mx:Application> tag insert an <mx:Script> block.
3. Create a bindable, private variable named `restaurants` data typed as **XMLListCollection**.

```
[Bindable]
private var restaurants:XMLListCollection;
```

Note: When you data typed the variable Flex Builder 3 should have automatically imported the XMLListCollection class.

Add a result event to the <mx:HTTPService> tag

4. Add a result event to the <mx:HTTPService> tag and call the handler `resultHandler()`. Pass the event object as a parameter.

```
result="resultHandler(event) "
```

Create the event handler

5. At the bottom of the Script block, create the skeleton for a private function named `resultHandler()`, data typed as **void**. The function should accept a parameter named `event` data typed as **ResultEvent**.

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

Note: When you data typed the parameter Flex Builder 3 should have automatically imported the ResultEvent class.

6. In the function, assign `restaurants` the value of `event.result.restaurant` as a new `XMLListCollection`.

```
restaurants=new
XMLListCollection(event.result.restaurant);
```

7. Be sure your function appears as follows.

```
private function resultHandler(
    event:ResultEvent):void
{
    restaurants= new
XMLListCollection(event.result.restaurant);
}
```

Use the results

8. Change the binding for the `dataProvider` of the `ComboBox` to simply `restaurants`.
9. Run the application.
You should see the `ComboBox` is still correctly populated.

Display an Image

10. Set the `ComboBox`'s `id` property to `cmbRestaurants`.
11. Insert an `<mx:Image>` tag.
12. Bind the `source` property to the `image` property of the currently selected `ComboBox` item.

```
source="{ 'http://localhost/services/assets/ima
ges/' + cmbRestaurant.selectedItem.image"
```

13. Run the application.
You should see a `Image` below the `ComboBox`.

Sending Data to a Remote Service



- ▶ Use the `<mx:HTTPService>` tag to send remote data to a REST-ful web service
- ▶ To use the `HTTPService` class
 - Supply a `url` location
 - Call a remote method on some event
- ▶ To send data when the submit button is clicked
 - Pass arguments to a remote service when the `send` method is called

Walkthrough 7: Send a data to the database with HTTP Service



Steps

In this walkthrough, you will perform the following tasks:

- ▶ View the restaurant reviews with data returned from a web service
- ▶ Submit a restaurant review using an `<mx:HTTPService>` object

Create the `getReviewList` http service

1. Open `WT7.mxml`.
2. Insert an `<mx:HTTPService>` tag with an `id` of `getReviewService`.
3. Set the `url` property equal to `http://localhost/services/restaurant/?method=getReviewList`.
4. Set the `resultFormat` property equal to `e4x`.
5. Use the `change` event of the `cmbRestaurants` component to call the http service.

```
change="getReviews()"
```

6. Create the `getReviews()` function inside the `<mx:Script>` block

```
private function getReviews():void
{
    var id:int =
        cmbRestaurants.selectedItem.restaurant_id;
    var args:Object = {restaurantID:id};
    getReviewService.send(args);
}
```

Use the results to populate a List

7. Insert an `<mx:HBox>` tag around the `<mx:Image>` tag.
8. Insert an `<mx>List>` tag inside the `<mx:HBox>` and below the `<mx:Image>` tag.
9. Bind the `dataProvider` property to the `lastResult` property of the method called from the web service.

```
dataProvider="{getReviewService.lastResult.review}"
```

10. In the ComboBox, set the `labelField` property equal to `review_text`.

```
labelField="review_text"
```

11. Run the application.
You should see the reviews next to the image names after selecting the Baraka Cafe.

Create a input Form

12. Insert an `<mx:Form>` tag.
13. Insert an `<mx:FormItem>` tag inside the `<mx:Form>` tag.
14. Set the `label` property equal to `Review Text`.
15. Insert an `<mx:TextArea>` tag inside the `<mx:FormItem>` tag.
16. Set the `id` property equal to `txtReview`.
17. Insert an `<mx:Button>` tag inside the `<mx:Form>` tag.
18. Set the `label` property equal to `Submit`.
19. Use the `click` event of the `<mx:Button>` component to call the `http` service.

```
click="sendReview()"
```

20. Create the `sendReview()` function inside the `<mx:Script>` block

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

Send the results with a HTTPService object

21. Insert an `<mx:HTTPService>` tag with an `id` of `sendReviewService`.
22. Set the `url` property equal to `http://localhost/services/restaurant/?method=createReview`.
23. Set the `resultFormat` property equal to `e4x`.
24. Use the `result` event of the `<mx:HTTPService>` component to refresh the review list.

```
result="getReviews()"
```

25. Make the HTTPService call inside sendReview,. Add the following code to the sendReview function.

```
private function sendReview():void
{
    var id:int =
        cmbRestaurants.selectedItem.restaurant_id;
    var args:Object =
        {restaurantID:id,text:txtReview.text};
    sendReviewService.send(args);
}
```

26. Bind the dataProvider property to the lastResult property of the method called from the web service.

```
dataProvider="{restaurantService.
    lastResult.restaurant}"
```

27. Run the application.

You should be able to submit a review for a restaurant and view the new review in the list.

Setting Up the Environment

Installing Flex Builder 3 and PHP plugin

1. Obtain Flex Builder 3.

Note: The following instructions were created for Flex Builder 3 Beta 2. There might be difference in the final release of Flex Builder 3.

2. Install Flex Builder 3 accepting all of the default options.
 - The trial period on Flex Builder 3 is 25 days.
 - You can extend the trial period with a valid Flex Builder 2 serial number.
3. Install PHP Eclipse plugin.
 - Start Flex Builder 3.
 - Select **Help > Software Updates > Find and Install...**
 - Check the "**Search for new features to install**" Radio button, and then click Next
 - Click the "**New Remote Site...**" button and fill in the following information:
 - Name: PHPEclipse Update Site
 - URL: <http://phpeclipse.sourceforge.net/update/releases>
 - Click the OK button
 - With the new "**PHPEclipse Update Site**" checkbox checked click the Finish button.
 - In the Search Results dialog window, open up the PHPEclipse Update Site option tree and select "**PHPEclipse 1.1.8**". Then click the Next button.
 - Select the "**I accept the terms...**" radio button, and click the Next button.
 - Now you are in the Installation dialog window, make sure PHPEclipse is selected and click the Finish button.
 - It will start downloading the plugin and the prompt with a Feature Verification dialog window. Click the "**Install All**" button.
 - When finished installing, it will prompt you to restart Flex Builder 3 for the changes to take affect. Click the Yes button to restart Flex Builder 3.

Installing XAMPP project (Apache, PHP, and MySQL)

1. Download XAMPP Lite 1.6.3a ([xampplite-win32-1.6.3a.zip](http://www.apachefriends.org/en/xampp-win32-1.6.3a.zip)).
 - <http://www.apachefriends.org/en/xampp-windows.html#646>
2. Unzip all the files where you downloaded the project. You should see a xampplite-win32-1.6.3a folder.

3. Navigate into the xampplite-win32-1.6.3a folder and copy the "xampplite" folder to the root of your drive (ie. C:\), it should look like C:\xampplite.
4. Run C:\xampplite\xampp_start.exe
 - Note: Leave the command prompt up, to shutdown close the command prompt or run C:\xampplite\xampp_stop.exe

Installing Zend Framework

1. Download Zend Framework
 - <http://framework.zend.com/download>

Note: Zend Framework version 1.0 was used as of this writing, but later version should work fine

2. Unzip Zend Framework download in C:\xampplite\

Installing session files

1. Unzip ra101h.zip to C:\max2007\.

Setting the workspace in Flex Builder 3

1. Open Flex Builder.
2. Select **File > Switch Workspaces...**
3. Set the workspace to C:\max2007\ra101h_workspace.
4. You should now see the Navigator view with the following projects.
 - Session
 - SessionSolutions
 - PHPServices

Setting up the database

1. Open up a command prompt clicking **Start > Run** then typing **cmd** and then clicking **Ok**.
2. Create the database and import the default values by running the commands below:
 - >cd c:\xampplite\mysql\bin
 - >mysql -u root
 - mysql>create database restaurant;
 - mysql>exit
 - >mysql restaurant -u root < c:\xampplite\htdocs\services\restaurant.sql
3. Test the data is correct:
 - >mysql restaurant -u root
 - mysql>select * from restaurant limit 5;
 - You should see 5 records returned to the console

- mysql>exit
- >exit

Testing the installation

1. Start Apache and MySQL by running **C:\xampplite\xampp_start.exe**
2. Open Flex Builder.
3. Select the SessionSolutions project in the Navigator view.
4. Click the Run button.

You should be able to see a list of restaurants in a ComboBox.