DZone Refcardz

brought to you by... Adobe

CONTENTS INCLUDE:

Adobe® Flash® Builder™ 4.5 for PHP

Adobe/Zend Tools for the PHP Professional

By Marco Tabini

ABOUT THIS REFCARD

Over the past few years, Adobe Flash has grown to encompass a family of technologies whose abilities go well beyond the browser. Today, Flash is a platform that can be used to build and deliver speedy, highly functional applications capable of running on a variety of different systems, from desktop OSs to mobile devices.

Thanks to the joint efforts of Adobe and Zend and with the help of many members of an enthusiastic community, Flash and PHP enjoy a synergistic relationship, which means that developers who know both languages find themselves empowered to build complex, vertically integrated applications that go well beyond the realm of the Web.

Flash Builder 4.5 for PHP is a new offering from Adobe and Zend that combines the two companies' knowledge in Flash and PHP into a single development environment that makes building integrated applications easy and fast.

Hot Tip You can download a 60-day trial copy of Flash Builder 4.5 for PHP from the Adobe Website at http://www.adobe.com/go/flashbuilder.

KEY FEATURES

Flash Builder 4.5 for PHP is built on Eclipse, an IDE technology that relies on Java. As a result, nearly identical versions of the application are available for both Windows and OS X.

Support for Flash and Flex

As part of the Flash Builder family, the IDE provides you with all the power of Adobe's best-of-breed Rich Internet Application development features. These include the ability to write applications that can be deployed on the Web, on the desktop, and on several mobile platforms.

You can find out more about Adobe Flex at http://www.adobe.com/products/flex/ and about Adobe AIR at http://www.adobe.com/products/air/.

Support for Data Exchange

Because it is built specifically with Flash in mind, one of Flash Builder for PHP's best features is its ability to interact with remote data services.

Through its data connection and service manipulation wizards, Flash Builder for PHP makes it easy to discover services based on most Internet standards like AMF, XML-RPC, REST, and SOAP. It then creates object proxies for them, which makes them accessible to your code as if they were local Flash objects.

Best of all, it can marshal the data exchanged between your application and the services it consumes in a mostly automated fashion, reducing your time to market and improving the overall stability of your applications.

Hot Tip You can find out more about the Action Message Format—the optimized data interchange format built natively into Flash—and its integration within Zend's products at http://framework.zend.com/manual/en/zend.amf.html.

Support for Mobile Applications

As part of its support for multiple platforms, Flash Builder for PHP is also capable of building applications that can run on multiple platforms, including Apple's iOS, Google's Android, and even Research in Motion's BlackBerry Tablet OS.

Integration with PHP

Because Flash Builder 4.5 for PHP incorporates the functionality of Zend Studio, it features a rich environment for the production of complex and highly interactive PHP-based applications.

The built-in debugger allows you to step through your Rich Internet Applications end to end, maintaining complete control over both the frontend and backend portions of your code.

Thus, you can initiate an action on the client side, watch it work its way through the Flash frontend, examine the data it exchanges with the backend, and debug your PHP code, all through the Flash Builder IDE.

Integration with Zend Technologies

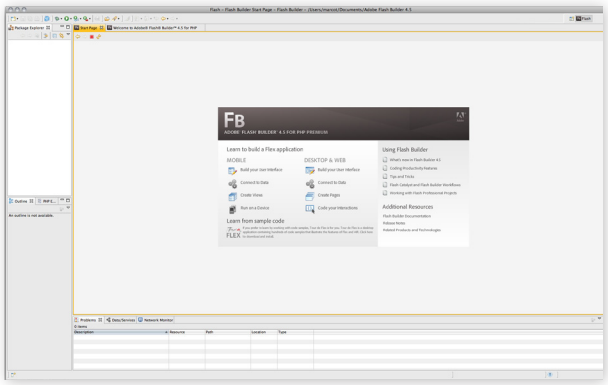
Flash Builder 4.5 for PHP is designed to work equally well across any PHP-centric environment, with support for strong debugging, code management, and cross-integration with Adobe's Flash platform.

However, Adobe's latest offering for PHP is also fully integrated with several Zend technologies, including Zend Framework, Zend Server, and Zend Studio. This allows you to take advantage of advanced PHP coding techniques and capabilities that increase your productivity and the quality of your code.

Note: *This Refcard assumes that you are familiar with both PHP and Adobe Flex concepts, including AMF, MXML and ActionScript.*

A TOUR OF FLASH BUILDER 4.5

Users of an Eclipse-based IDE will find Flash Builder for PHP's interface, pictured below, extremely familiar:



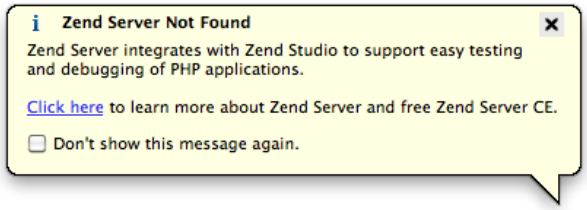
The left portion of the screen is dedicated to your project's workspace. Here, you can explore the various files and packages that make up the projects loaded into Flash Builder for PHP.

The central space gives you access to the contents of your files, the help system, and various consoles that can be used for error checking, data manipulation, and debugging.

Setting Up

Before you start using Flash Builder for PHP, it's a good idea to set up your environment so that it can take full advantage of the product's capabilities.

When you first start Flash Builder for PHP, it will notify that you do not have Zend Server installed:



Zend Server is a product developed by Zend that combines a business-grade version of PHP with several deployment and debugging facilities designed to make the management of PHP applications easier.

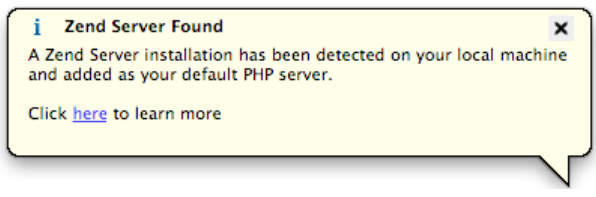
Although not necessary, installing Zend Server makes it easier to access some of Flash Builder 4.5 for PHP's more advanced functionality by providing a simplified installation mechanism and making the deployment of Websites easier.

Together with the server components, Zend Server also installs a copy of Zend Framework, Zend's own application development framework.

Zend Framework, one of the most popular and widely used frameworks in the PHP market, features built-in support for AMF communication, which is essential to building advanced services that Flash applications can easily consume.

Flash Builder for PHP relies on some features of Zend Framework to introspect your code and provide support for the automated creation and discovery of services you create.

The next time you launch Flash Builder for PHP, it will also automatically detect Zend Server and make it your default PHP server:



What if you do not have Zend Server or use a different deployment environment or framework? No problem! Although Zend Server makes getting up and running quick and easy, and Zend Framework is needed to enable some of the AMF data exchange functionality, you do not need to rely on either product for your Websites. Flash Builder 4.5 for PHP will work equally with applications written using other frameworks or deployed on any kind of server.

Building Your First Application

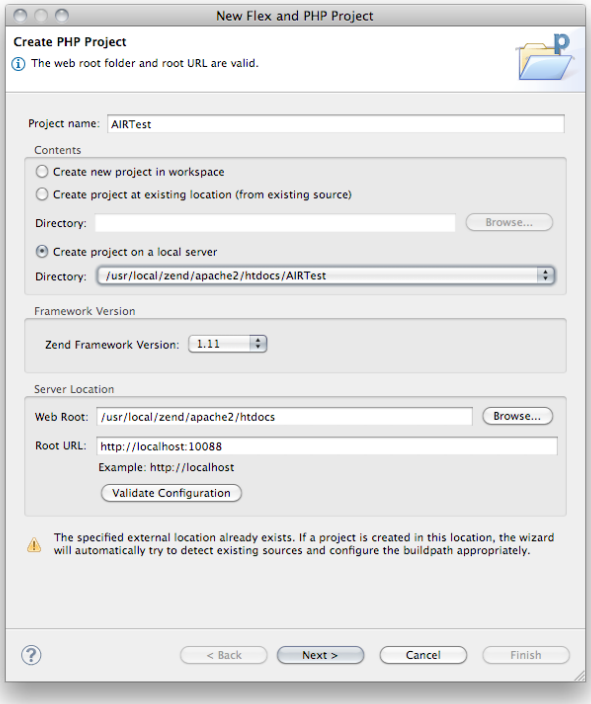
Before you can dive into creating code for your Web application, you need to figure out where Zend Server expects the root of your site to be located.

The exact location depends on the operating system on which your copy of Zend Server is running; you can find more information on figuring out this path in the Zend Server FAQ at http://www.zend.com/products/server/faq#faq1.

The creation of a new project can take many routes, depending on the kind of application you intend to build.

For example, you can create a Flash-only project for either desktop or mobile deployment or a PHP project that takes advantage of Zend Studio and Zend Framework for easy deployment.

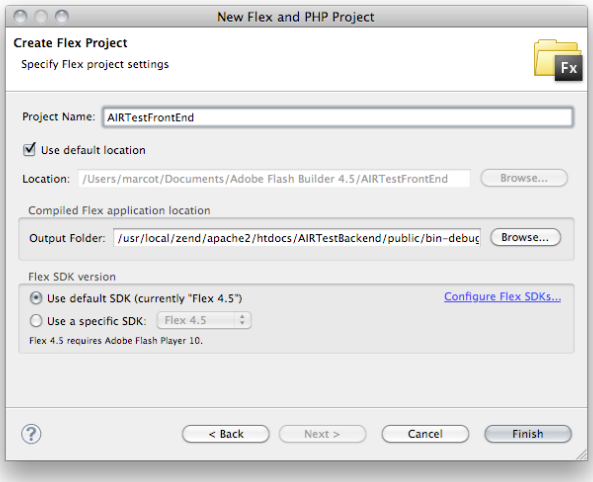
Where Flash Builder for PHP really shines, however, is in the creation of projects that integrate both Flash and PHP. Let's take a look at how those are created and managed. You start by selecting New > Flex and PHP Project from the File menu; this will bring up a dialog box like the following:



If you have already installed and configured Zend Server or another PHP server, most of this information will be filled in automatically for you. You only have to worry about coming up with a unique name for the PHP portion of your project.

Hot Tip Don't forget to click on the "Validate Configuration" button before proceeding. Flash Builder for PHP will ensure that your Web site's root is reachable and properly configured.

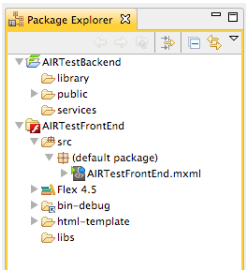
Next, you will need to specify the metadata required to create the Flex side of your project:



Flash Builder for PHP will fill out more of your required fields here, too.

You'll notice, however, that I've had to use a different name; this is because the two separate PHP and Flex projects are created and are linked to each other.

Once this last step is completed, Flash Builder for PHP will add the two projects to the Package Explorer:



SETTING UP YOUR PROJECT

Now that you have created your projects, you will need to perform a final step to make sure that Flash Builder for PHP can interact with it properly.

As mentioned earlier in this document, Flash Builder for PHP requires a copy of Zend Framework to understand how your backend is structured and build AMF proxies for any of your services.

Since Zend Server comes with a built-in copy of Zend Framework, you need to make a small change to your project so that the Flash Builder for PHP engine knows where to look for it.

In the build-debug directory of your Web project, change the amf_config.ini file so that the zend_path setting points to the location where Zend Framework is installed.

The template that Flash Builder for PHP generates when you create your project contains several helpful hints on the default location where Zend Server installs Zend Framework. For example, on a Mac, you will normally set up your INI setting like this:

```
<zend_path =/user/apache/PHPFrameworks/ZendFramework/library
```

Hot Tip If you skip this step, Flash Builder for PHP will attempt to download and install another copy of Zend Framework for you when you start using the AMF data connector. If you use Zend Server, this is not desirable, because it leads to duplication and makes deployment more difficult.

CREATING A DATA SERVICE

Now that everything is set up, we can move on to building our application.

In a real-world project, you will typically want to design your systems so that they can interact with the user through multiple interfaces—for example, you could build a traditional Web site or Flash-based application for general access, and then use AIR for mobile usage.

Thus, if you think of your overall application as using a model-view-controller architecture, PHP always provides your model, while your controller and view can be provided by different technologies, depending on the particular platform you're targeting.

It's very advantageous to adopt this development from the very start of your project for two reasons. First, it separates the various layers of your application neatly, which makes the duplication of code less likely and its reusability much higher. Second, it simplifies the process of building each frontend by ensuring that all the business logic is stashed away in a single place.

For our example, we will start by creating a simple data service in PHP and make our Flex application interact with it.

Thanks to the way Zend Framework works, this is a very simple process that starts with the creation of a new file, which we'll call clock.php, in the services directory of your Web project. To get our example started, we'll create a very simple service that returns the time of day:

```
<?php

class Clock {

            function clock() {
                        return date("Y/m/d H:i:s");
            }

}
```

As you can see, there is nothing special about this short snippet of code. Our AIRTest class simply features a function, called clock(), that returns the current date and time formatted as a string. Remember that, if you are using a recent version of PHP, you need to set your timezone properly or the call to the date function will issue a warning and cause your service calls to fail.

Interestingly, this is all we need to do to create a data service! The only real requirement is that the name of the file in which we have stored our code and the name of the class it contains match. Clearly, this means that we can only store one class per file.

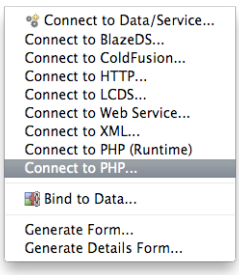
Hot Tip This is obviously a contrived example, kept simple for the sake of clarity. A real-life application would require a proper security and infrastructure to be built around your data services to ensure proper access control.

CONNECTING YOUR FLASH FRONTEND TO PHP

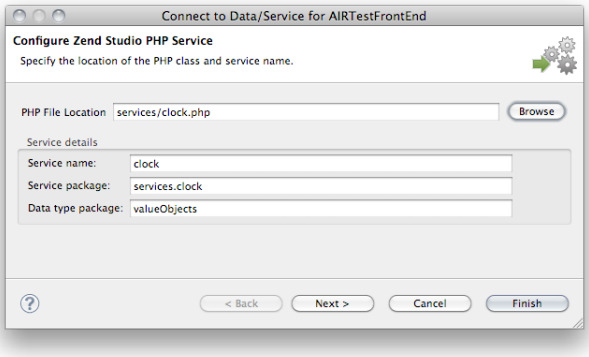
We can now move on to the frontend, where Flash Builder for PHP will take care of most of the hard work for us.

The first step is to make our Flash application aware of the data service's existence. To do so, we use a feature in Flash Builder for PHP that helps us connect to and introspect PHP services.

From the Data menu, select "Connect to PHP…"



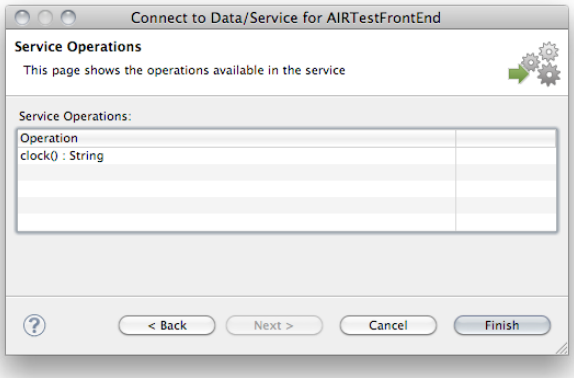
This pulls up the Data Connection Wizard, which guides you through the process of creating ActionScript classes for connecting to the PHP service and value objects. The first step consists of telling Flash Builder for PHP where your data service is located:



As you can see, all you really need to do is point the wizard to the location of your file (and, because you set the site location when you created the project, it already knows its root directory).

The wizard is capable of automatically identifying the class contained in your file and building a corresponding Flash package inside your frontend project.

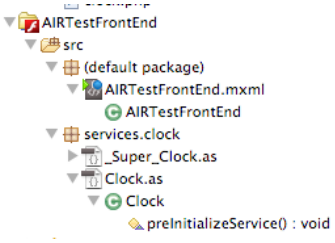
Clicking on Next will bring up a list of available method calls inside your service class:



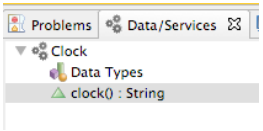
In this case, we only have the clock method, which returns a string.

Hot Tip Flash Builder for PHP relies on Zend Framework's introspection framework to use your code's docblocks and annotation tags to determine the data types associated with the parameters and return the values of your method. You can find more information at http://framework.zend.com/manual/en/zend.reflection.reference.

You can now complete the wizard by clicking the Finish button. This will cause the Data Connection Wizard to create a number of classes inside your frontend project that automatically marshal calls and the exchange of data between Flash and PHP:



The service is also available in the Data/Services panel:



From here, you can create service calls directly in your code. Select the main MXML file of your frontend project, then right click on the clock method in the Data/Services panel, and choose "Generate Service Call" from the resulting pop-up menu.

This causes an instance of your service call to appear in your code:

```
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
 xmlns:s="library://ns.adobe.com/flex/spark"
 xmlns:mx="library://ns.adobe.com/flex/mx"
 xmlns:clock="services.clock.*"
 minWidth="955" minHeight="600">

<fx:Script>
 <![CDATA[
  import mx.controls.Alert;

  protected function clock2():void
  {
   clockResult.token = clock.clock();
  }
 ]]>
</fx:Script>

<fx:Declarations>
 <s:CallResponder id="clockResult"/>
 <clock:Clock id="clock"
  fault="Alert.show(event.fault.faultString + '\n' +
  event.fault.faultDetail)"
  showBusyCursor="true"/>

 <!—
  Place non-visual elements (e.g., services, value objects) here
 -->

</fx:Declarations>
</s:Application>
```

As you can see, our service call is implemented as a custom class, which is imported and instantiated into our MXML file. The instance of the clock service is then connected to an instance of CallResponder, which is responsible for marshaling the actual service call in an asynchronous fashion.

We can now bind controls to our service in such a way that we can execute the service call and display its results to the user.

Hot Tip By default, Flash Builder for PHP creates a method that calls your data services on launch by binding to the Application's creationComplete event. html.

Let's start with the execution process; we'll pick a button and drop it somewhere on our application's canvas. In MXML, it will look something like this:

```
<s:Button id="button" x="529" y="485" label="Get Time" />
```

Next, we can switch to Design Mode, where we see a visual representation of our code. To cause the button to call our service when it is clicked, we just need to drag the clock method from the Data/Services panel onto the button.

This will cause our code to change as such:

```
<s:Button id="button" x="551" y="521" label="Button"
 click="button_clickHandler(event)"/>
```

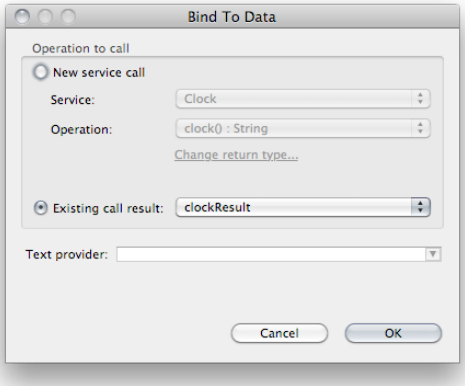
The Data Connector also added a method that is called whenever the button is clicked:

```
protected function button_clickHandler(event:MouseEvent):void
{
 clockResult.token = clock.clock();
}
```

Our final step is to set up a way for the application to display the service's return value. We can add a label to our canvas, which will look similar to this:

```
<s:Label id="label2" x="501" y="445"
 text=""/>
```

We can now bind the service call using the same process that we used for the button. From the Design view, just drag the clock method onto the label. Flash Builder for PHP will show a dialog box that gives you the option of creating a new service call or using an existing one:



Since we have already created a service call, we simply point to that one and click on the OK button; the label is now bound to the responder of our service. You can test the project by instantiating it in your browser (select "Run" from the Run menu) and clicking on the button. You should see something like this:

As you can see, the service is working perfectly—and we didn't even have to write a single line of code!

DEBUGGING

One of the most interesting features of Flash Builder for PHP is end-to-end debugging.

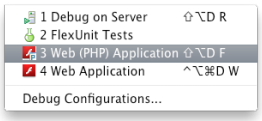
Whereas both the traditional versions of Flash Builder and Zend Studio allow you to debug your Flash and PHP individually, with Flash Builder for PHP you can actually debug your code all the way from the frontend to the backend in a single operation.

Once you start building complex applications that involve multiple data services and subsystems, this feature will become extremely valuable by improving your efficiency and productivity.

Like many of Flash Builder for PHP's other features, end-to-end debugging "just works" once you have configured your project properly.

Hot Tip If your site runs on a server technology other than Zend Server, you can still access the debugging functionality of Adobe Flash Builder for PHP 4.5 by installing Zend Debugger, which can be downloaded from http://www.zend.com/en/products/studio/downloads

In most cases, this means ensuring that Flash Builder knows that your project should be debugged as a PHP-powered Rich Internet Application. This can be accomplished by right-clicking on your front-end project in the Package Explorer, then select "Web (PHP) Application" from the Debug As menu:

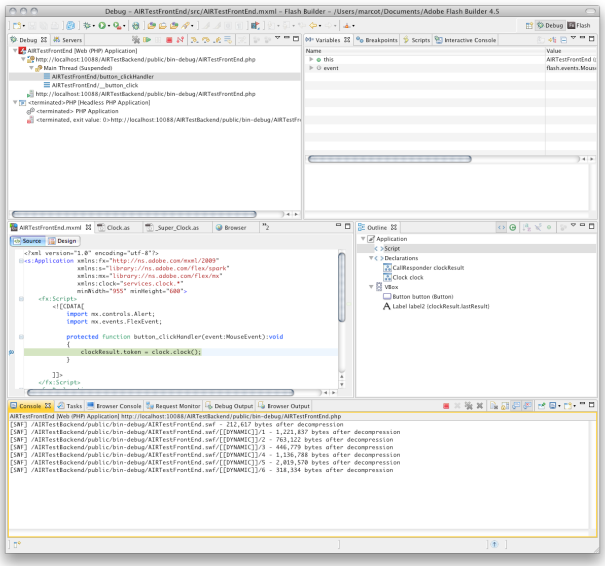


At this point, you can start the debug process by launching the frontend project by selecting "Debug" from the Run menu. This will bring up your browser and load the SWF file associated with your application.

You can now start interacting with your Web-based application as you normally would; you can observe your code as it is executed by setting a breakpoint. The easiest way to do so is to double-click in the gutter next to the line where you want the execution to pause. A breakpoint is represented by a blue dot:

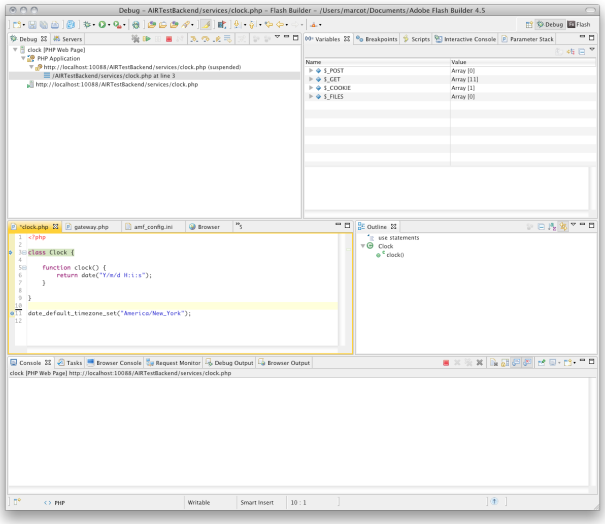
```
protected function button_clickHandler(event:MouseEvent):void
{
    clockResult.token = clock.clock();
}
```

When the execution reaches this point (in our case because the user has clicked on the button in our sample application), Flash Builder for PHP will regain focus and switch to the Debug perspective for Flash, which looks like this:



As you can see, two new panels have appeared at the top of the window. On the left is the Debug panel, where you can see the call stack that has led to the current execution point. On the right-hand side are several panels that let you interact with breakpoints, perform live operations against an interactive console, and observe various values in your application in real time.

If you now set a breakpoint in your backend, Flash Builder for PHP will follow the execution of code from Flash to PHP, allowing you to move from one side of your application the other in a single sweep:



As you can see, the Debug perspective on the PHP side is very similar to its counterpart on the frontend. You also have full access to all your PHP runtime information, including the various superglobals.

Once you have stepped through the PHP portion of your code execution, control reverts to the Flash side, where you can continue to follow the debugging process until control is yielded back to the user.

Hot Tip You can switch from the Debugging perspective back to the Flash or PHP perspective by clicking the appropriate button on the top-right corner of Flash Builder for PHP's window.

WHERE TO GO FROM HERE

Space constraints have made it possible for us to only explore a small portion of the functionality that Flash Builder for PHP provides. There are many other areas of the product that can significantly increase your productivity—both independently as a PHP or Flash/Flex developer and jointly as the two sides of a Rich Internet Application.

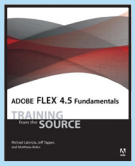
You can find more information about the Flash Builder family and its integration with PHP at http://www.adobe.com/go/flashbuilder.

ABOUT THE AUTHOR

Marco Tabini is the co-founder of "php|architect": http://phparch.com/ and CEO of "Blue Parabola, LLC": http://blueparabola.com/, a consulting firm that specializes in PHP-related knowledge products.

He is a frequent speakers at PHP conferences worldwide, including php|tek, the Dutch PHP Conference, Enterprise LAMP Camp and many more. Marco is also the author of two books on computer programming and the "long-running exit(0)" column on php|architect. Marco blogs regularly at "The Accidental Businessman": http://blog.tabini.ca and is a frequent collaborator at "Macworld": http://macworld.com/

RECOMMENDED BOOK



Adobe Flex: Training from the Source is the best-selling and most trusted resource for learning about Adobe Flex. Written by a team of authors with practical experience as consultants, mentors and developers of courseware, this book/CD uses project-based tutorials, and is designed to teach beginning Flex developers the details of building and architecting real-world rich internet applications using Flash Builder incorporating MXML and ActionScript 3.0. The book includes a CD that contains all the files used in the lessons, plus completed projects for comparison. This latest edition includes complete coverage of new Flex 4.5 features, such as new enhancements to the Spark architecture and component set. It will also show you how to take advantage of the improvements to core Flex infrastructure for large application development.

DZone

DZone communities deliver over 6 million pages each month to more than 3.3 million software developers, architects and decision makers. DZone offers something for everyone, including news, tutorials, cheat sheets, blogs, feature articles, source code and more. **"DZone is a developer's dream,"** says PC Magazine.

DZone, Inc.
140 Preston Executive Dr.
Suite 100
Cary, NC 27513

888.678.0399
919.678.0300

Refcardz Feedback Welcome
refcardz@dzone.com

Sponsorship Opportunities
sales@dzone.com

ISBN-13: 978-1-936502-44-8
ISBN-10: 1-936502-44-5
50795
9 781936 502448
$7.95

Version 1.0