

Chapter 37: Using Flash Remoting Update

You can use Flash Remoting Update to create Rich Internet Applications by using ColdFusion with Adobe Flex.

Contents

About Flash Remoting Update	688
Using Flash Remoting Update	688

About Flash Remoting Update

Flash Remoting Update lets you create rich Internet applications using Adobe Flex Builder 2, with the advanced data retrieval features of ColdFusion, such as the `cfpop`, `cfldap`, and `cfquery` tags. In addition, you can use Flash Remoting Update to create Flash Forms and Flash applications that contain features, such as server call backs and customized user interface.

You can use Flash Remoting Update with all configurations of ColdFusion (server, multiserver, and J2EE) on all the platforms that ColdFusion supports.

To use Flash Remoting Update, you must have the following installed:

- Flex 2 SDK or later, Flex Builder 2 or later, or the version of LiveCycle Data Services that is included as an optional component of ColdFusion 8.
- Flash Player 8.5 or later

***Note:** Flex 3, and Flex Builder 3 were released after ColdFusion 8, and can be used with ColdFusion 8. The MXML compiler is included in the free Flex SDK. An open source version of the Flex SDK, including the MXML compiler, is available at the [Adobe Open Source site](#). The full Flex SDK (including parts that are not open source) is available from the [Flex SDK Downloads site](#).*

Using Flash Remoting Update

To use the Flash Remoting Update you do the following:

- 1 [Configure Flex Compilation](#)
- 2 [Specify a CFC](#)
- 3 [Use the CFC](#)
- 4 [Compile and Run the application](#)

Configure Flex Compilation

You use Flex Builder or the Flex SDK to compile Flex applications into SWF files. To use the Flash Remoting Update, these programs must use the ColdFusion `services-config.xml` file when compiling the MXML. If you have installed ColdFusion with LiveCycle Data Services, and are requesting MXML from the ColdFusion web root (that is, if the Adobe Flex compiler module for J2EE application servers compiles the MXML on the ColdFusion system), this is done automatically. Otherwise, you must configure Flex Builder to use the ColdFusion configuration file, or specify the file when you use the SDK to compile your application (as described in [Compile and Run the application](#)).

Configure Flex Builder 2 to use the ColdFusion configuration file

When you use the Flex Builder Project Setup Wizard and select ColdFusion as the server type, the wizard configures Flex Builder to use the `services-config.xml` file for you. Use the following steps to configure your project:

- 1 Select **File > New > Flex Project** to open the New Flex Project Wizard, and enter the appropriate information in the first sections of the Create a Flex project page.
- 2 Select one of the radio buttons, as follows:
 - Select **ColdFusion Flash Remoting** to compile in Flex Builder.
 - If you installed LiveCycle Data Services with ColdFusion and want to use messaging or data management, select **Flex Data Services**.
- 3 If you select Flex Data services, select whether to compile the application locally in Flex Builder or on the application server where the page is viewed. Do not select to compile code that you will deploy on the server; this option is for development purposes only.
- 4 Click **Next** and complete creating the project, then click **Finish**.

If you select **Basic** on the first Create a Flex Project page, and decide later to compile the application for use with ColdFusion, you must configure Flex Builder manually, as follows:

- 1 Select **Project > Properties**.
- 2 Select **Flex Compiler** in the right pane of the Properties dialog.
- 3 In the **Additional Compiler arguments** add `-services=` followed by the absolute path to the `services-config.xml` file in the local ColdFusion installation. For example, on a Windows system with a default ColdFusion stand-alone installation, specify the following argument string.

```
-services=C:/ColdFusion8/wwwroot/WEB-INF/flex/services-config.xml
```

Configure Flex Builder 3 to use the ColdFusion configuration file

When you use the Flex Builder Project Setup Wizard and select ColdFusion as the server type, the wizard configures Flex Builder to use the `services-config.xml` file for you. Use the following steps to configure your project:

- 1 Select **File > New > Flex Project** to open the New Flex Project Wizard, and enter the appropriate information in the first sections of the Create a Flex project page.
- 2 In the **Server technology** section of the Create a Flex project page, select **ColdFusion** as the Application server type, and select **Use remote object access service**.
- 3 Select one of the radio buttons, as follows:
 - Select **ColdFusion Flash Remoting** to compile in Flex Builder.
 - If you installed LiveCycle Data Services with ColdFusion, and want to compile the application on the server, select **LiveCycle Data services** in Flex Builder 3.

4 Click Next to open the Configure ColdFusion page, and enter the required information. If you selected LiveCycle Data services in step 3, you can select to compile the application locally or on the server. You should select to compile on the server only when you are developing your application, for convenience. Do not select to compile on the server code that you will deploy, because the MXML page will not be compiled to a SWF file until the user requests it, and the compiler does not create an HTML wrapper page.

5 Click Finish to complete the configuration.

If you do not specify ColdFusion in the Server technology section of the Create a Flex project page, and decide later to compile the application for use with ColdFusion, you must configure Flex Builder manually, as follows:

1 Select Project > Properties.

2 Select Flex Compiler in the right pane of the Properties dialog.

3 In the Additional Compiler arguments add `-services=` followed by the absolute path to the `services-config.xml` file in the local ColdFusion installation. For example, on a Windows system with a default ColdFusion stand-alone installation, specify the following argument string.

```
-services=C:/ColdFusion8/wwwroot/WEB-INF/flex/services-config.xml
```

Specify a CFC

To specify a CFC to connect to, you do one of the following:

- Specify the dot-delimited path from the web root to the CFC in the MXML.
- Create a named resource for the CFC. This is similar to registering a data source; you then use the resource name in your XML.

Specify the CFC in the MXML

To specify the CFC in your MXML, use code such as the following:

```
<mx:RemoteObject  
  id="myCfc"  
  destination="ColdFusion"  
  source="myApplication.components.User"/>
```

The destination `ColdFusion` is preconfigured in the `services-config.xml`, if you are not using LiveCycle Data Services, or in `remoting-config.xml`, if you are using LiveCycle Data services. The default source value for this destination is the wildcard, `*`.

You do not have to use the `ColdFusion` destination if you have configured other valid destinations in the configuration file. In this case, the destination definition must specify `*` as the value of its `source` element. If you specify a source other than `*` in `services-config.xml` or `remoting-config.xml`, that source definition overrides the source specified in the MXML. For details of defining a destination, see [Create a named resource for a CFC](#).

Create a named resource for a CFC

1 Edit the `WEB-INF/flex/services-config.xml` or `remoting.config.xml` file by adding a destination entry for the CFC, for example:

```
<destination id="CustomID">  
  <channels>  
    <channel ref="my-cfamf"/>  
  </channels>  
  <properties>  
    <source>dot_path_to_CFC</source>  
    <!-- Define the resolution rules and access level of the cfc being invoked -->  
    <access>
```

```

<!-- Specify whether to use the ColdFusion mappings to find CFCs.
      By default only CFC files under the CF webroot can be found. -->
<use-mappings>>false</use-mappings>
<!-- Allow "public and remote" or just "remote" methods to be invoked. -->
<method-access-level>remote</method-access-level>
</access>
<property-case>
  <!-- cfc property names -->
  <force-cfc-lowercase>>false</force-cfc-lowercase>
  <!-- Query column names -->
  <force-query-lowercase>>false</force-query-lowercase>
  <!-- struct keys -->
  <force-struct-lowercase>>false</force-struct-lowercase>
</property-case>
</properties>
</destination>

```

The `source` attribute specifies the dot notation to the CFC from the web root (the classpath to the CFC).

2 Restart the ColdFusion server.

The following table describes the XML attributes:

Element	Description
<code>destination id</code>	The <code>destination</code> attribute that the MXML <code>mx:RemoteObject</code> tag must specify to access the CFC.
<code>channels</code>	A container for one or more child <code>channel</code> attributes specifying the AMF channels to use to access the ColdFusion server.
<code>source</code>	The dot-delimited file path to the CFC, from the <code>cfWebRoot</code> , or, if the <code>use-mappings</code> property is <code>true</code> , an entry in the ColdFusion Administrator Mappings page.
<code>access</code>	Properties that control how the CFC is accessed on the ColdFusion server.
<code>use-mappings</code>	A Boolean value specifying whether the source attribute can be relative to (start with) a ColdFusion mapping.
<code>method-access-level</code>	Specifies the <code>access</code> attribute values a CFC must have for ColdFusion to respond to the request. The following values are valid: <ul style="list-style-type: none"> <code>remote</code> Flex can access only functions that specify remote access. (the default) <code>public</code> Flex can access functions that specify both remote or public access.
<code>property-case</code>	Contains properties that specify whether ColdFusion forces property names, column names, or structure keys to all lowercase before it returns the data. (All three child properties default to <code>false</code> .) You do not normally need to use this attribute or its children, because ColdFusion automatically maps between ColdFusion data types and ActionScript data types. Note that the mapping can be successful, only if ActionScript class has careful calling conventions, explicit property definitions, and a <code>RemoteClass</code> definition.

Use the CFC

Use the CFC in your application:

- 1 In the MXML file, you use the `<mx:RemoteObject>` tag to connect to your CFC named resource. With this connection you can call any remote method on the CFC.
- 2 If you created a destination for the CFC in the `services-config.xml` or `remoting.config.xml` file, specify the destination name in the `mx:RemoteObject` tag; for example:

```
<mx:RemoteObject
```

```

    id="a_named_reference_to_use_in_mxml"
    destination="CustomID"
    result="my_CFC_handler(event)"/>

```

If you did not create a destination for the CFC, specify the ColdFusion destination and the CFC path in the `mx:RemoteObject` tag; for example:

```

<mx:RemoteObject
    id="myCfc"
    destination="ColdFusion"
    source="myApplication.components.User"/>

```

3 Call a CFC method, for example, as the following example shows:

```

<mx:Button label="reload" click="my_CFC.getUsers()"/>

```

In this example, when a user presses a button, the Click event calls the CFC method `getUsers`.

4 Specify a handler for the returned result of the CFC method call for the `<mx:RemoteObject>` tag, as the following example shows.

```

private function my_CFC_handler( event:ResultEvent )
{
    // Show alert with the value that is returned from the CFC.
    mx.controls.Alert.show(ObjectUtil.toString(event.result));
}

```

Compile and Run the application

You can compile and run your application from Flex Builder or without using Flex Builder. The techniques you can use also depend on whether you have installed LiveCycle Data Services with ColdFusion.

Compile and run the application using Flex Builder

To compile and run an application using Flex builder, make sure that Flex Builder is configured as described in [“Configure Flex Compilation” on page 689](#). Compile your application normally to create a SWF file. When you configure your Flex Builder project you can specify the location in which to put it. By default, Flex Builder attempts to put the SWF file and an HTML wrapper page under the web root. You can then run the application as appropriate, for example, by requesting the HTML wrapper for the SWF file in a browser.

If you have installed LiveCycle Data Services with ColdFusion 8 and configured your Flex Builder project to compile the application on the server when the page is viewed, the MXML file is located on the sever under the web root. Request the MXML page in your browser and the data services compiler will create a SWF, if one has not yet been created, and display it in browser. An HTML wrapper will be returned to the browser that loads the resulting SWF.

Compile and Run the application without Flex Builder

To compile the application directly using the SDK, you must set the Flex compiler to use the ColdFusion `services-config.xml` file. Do this by adding to the `mxmlc` command line `-services=` followed by the absolute path to the `services-config.xml` file in the local ColdFusion installation. For example, on a Windows system with a default ColdFusion stand-alone installation, specify the following argument string.

```

-services=C:/ColdFusion8/wwwroot/WEB-INF/flex/services-config.xml

```

You can then run the compiled SWF normally.

If you have installed Flex Data Services in ColdFusion 7.0.2 or LiveCycle Data Services in ColdFusion 8, you can also compile and run your application directly from the MXML file. However, you should do this only during development, as a convenience. To compile and run directly from the MXML file, put your MXML application under the ColdFusion web root and specify the MXML file URL in your browser. The data services compiler will create a SWF (if one has not yet been created) and display it in browser. An HTML wrapper will be returned to the browser that loads the resulting SWF.