ricoschneider
webanwendungen

Flex and PHP

Communication between Flex and PHP with amfphp

You can download the current version of this documentation here
http://www.schneider-webanwendungen.de/index.php?action=doShowSupport

1 Download Amfphp

Download the amfphp project files from the sourceforge download page http://sourceforge.net/projects/amfphp/files/amfphp/amfphp%201.9%20beta2/amfphp-1.9.beta.20080120.zip/download and store the file amfphp-1.9.beta.20080120.zip on your desktop.

2 Amfphp Installation

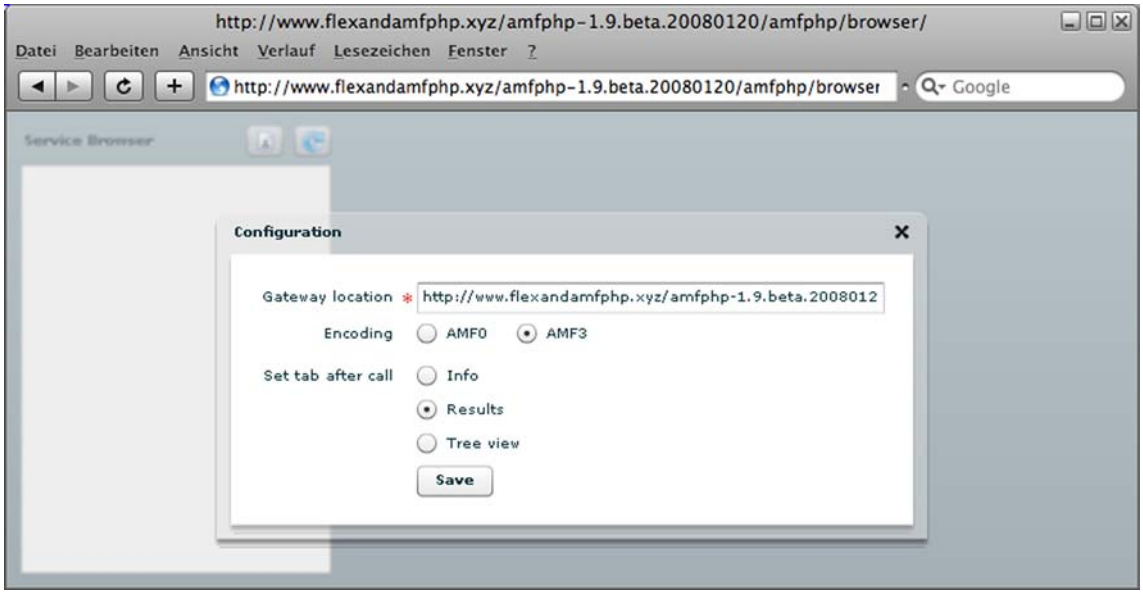
I'm using the local test url http://www.flexandamfphp.xyz. My test url targets the document root folder D:/Web/EasyPhp/www/flexandamfphp. For universal usage and an independent documentation the following local test url is http://localhost and the document root folder is named DOCROOT.

2.1 Setup Amfphp

Open the downloaded zip archive amfphp-1.9.beta.20080120.zip and extract the content directly into your document root folder.

2.2 Test the Amfphp Setup

Amfphp has a service browser included. You can open the service browser with the url http://localhost/amfphp-1.9.beta.20080120/amfphp/browser. When you see the following configuration panel in the middle of the web page, your amfphp setup was successfully installed.



Please select the radio button "Tree view" and press the „Save" Button.

3 PHP Service "helloUser" (Part 1)

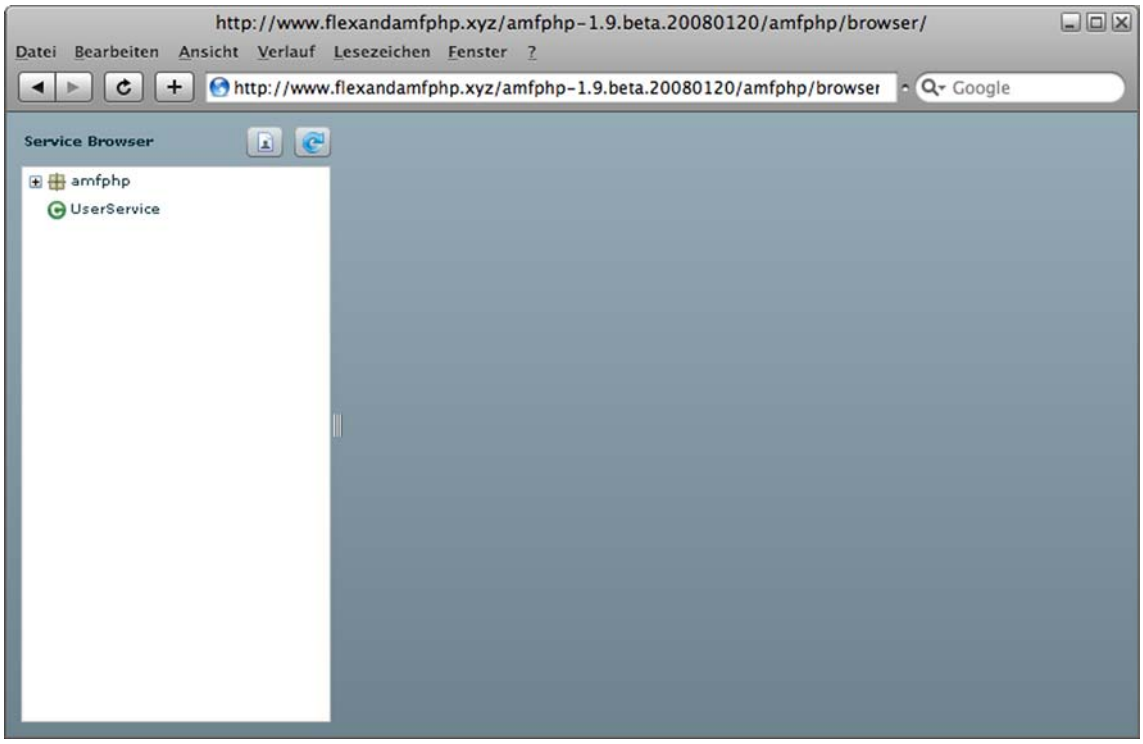
3.1 Create the PHP function "helloUser"

Please create a new php file "UserService.php" into the folder DOCROOT/amfphp-1.9.beta.20080120/amfphp/services. Add the following file content in the file UserService.php.

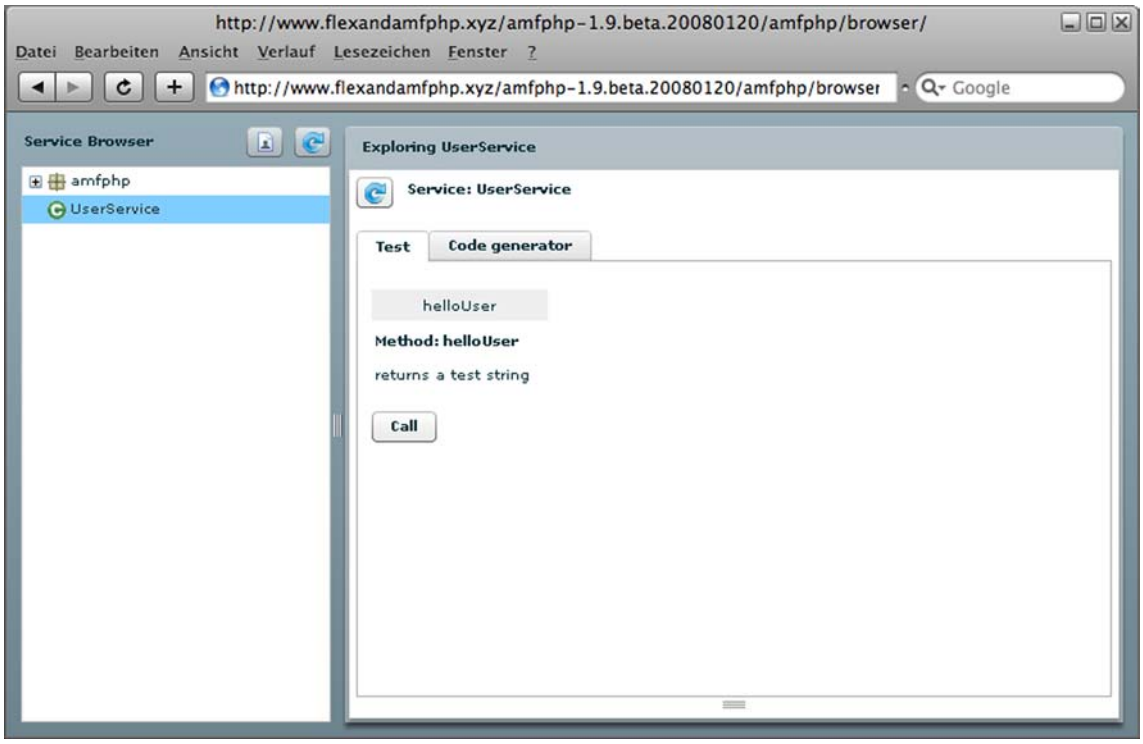
```
<?php

class UserService {

  function UserService() {}

  /**
   *   returns a test string
   *   @access public
   *   @return string  $result     test string
   */
  public function helloUser() {
    return "Hello Test User!";
  }

}

?>
```

3.2 Test PHP Service “helloUser” in the Service Browser

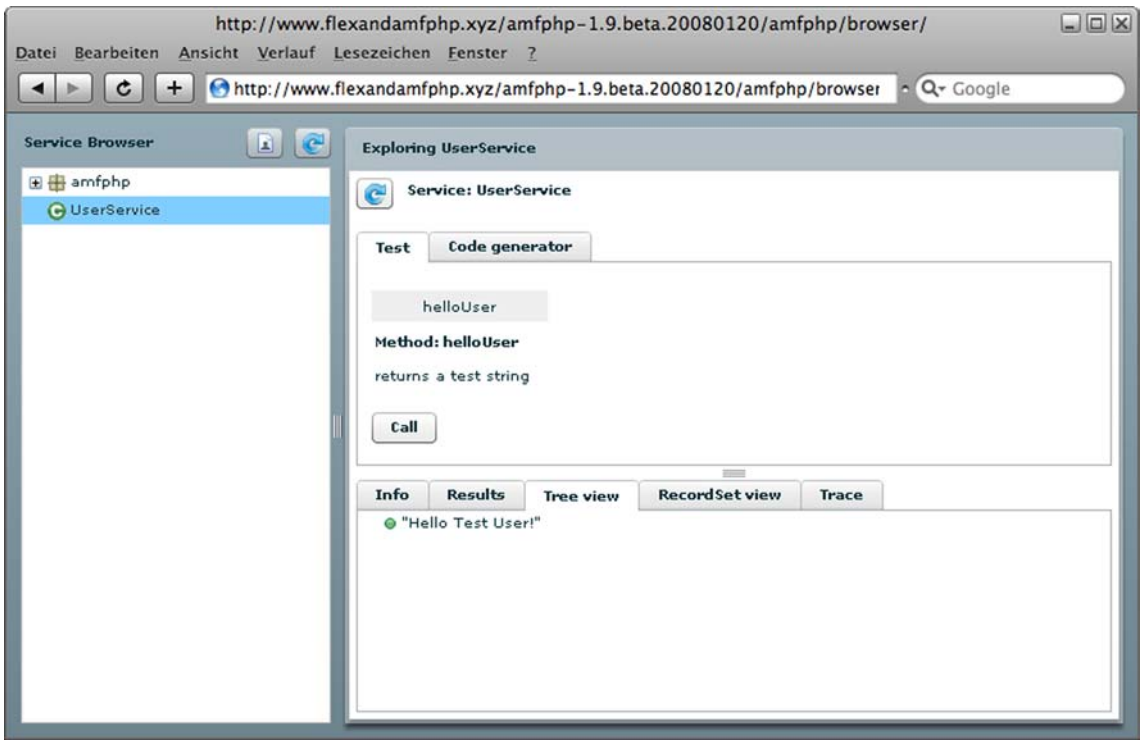
Please refresh your browser view or call the url http://localhost/amfphp-1.9.beta.20080120/amfphp/browser. Now, you see our new php service “UserService” on the left side in the service browser window.



Please select the item “UserService” on the left side. You will see the function “helloUser” and the function description “returns a test string” in the right panel.

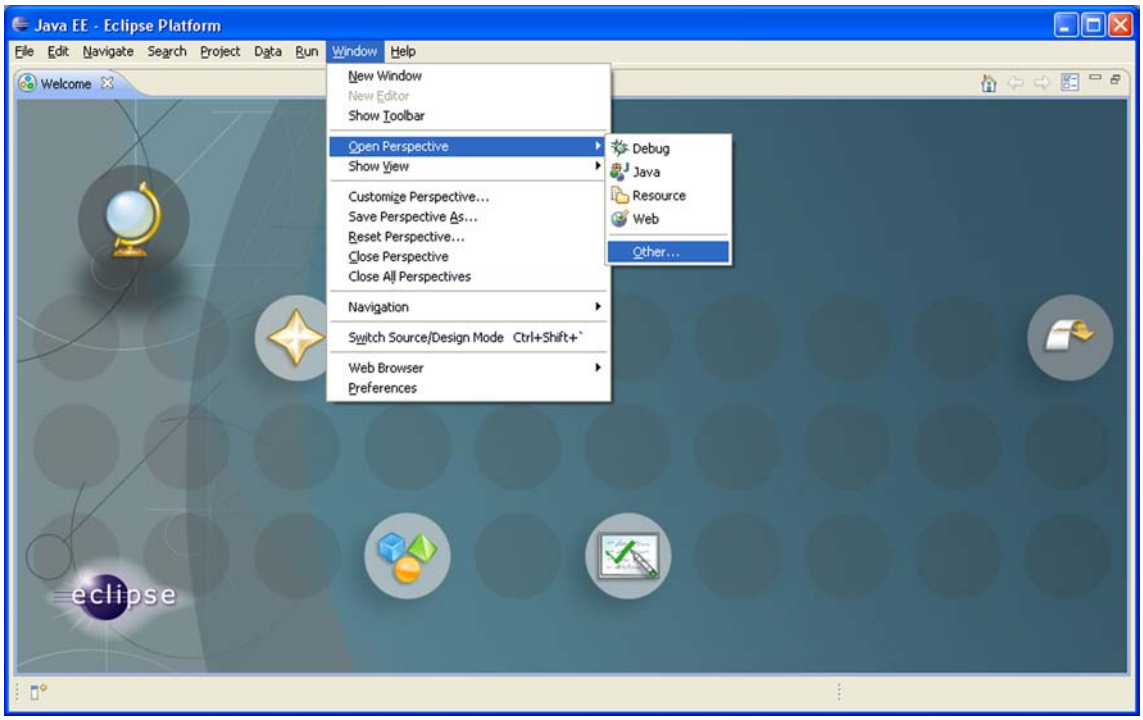


When you press the “Call” button under the function description, the PHP function “helloUser” is called and you’ll see the return string “Hello Test User!” into the Tree view panel.

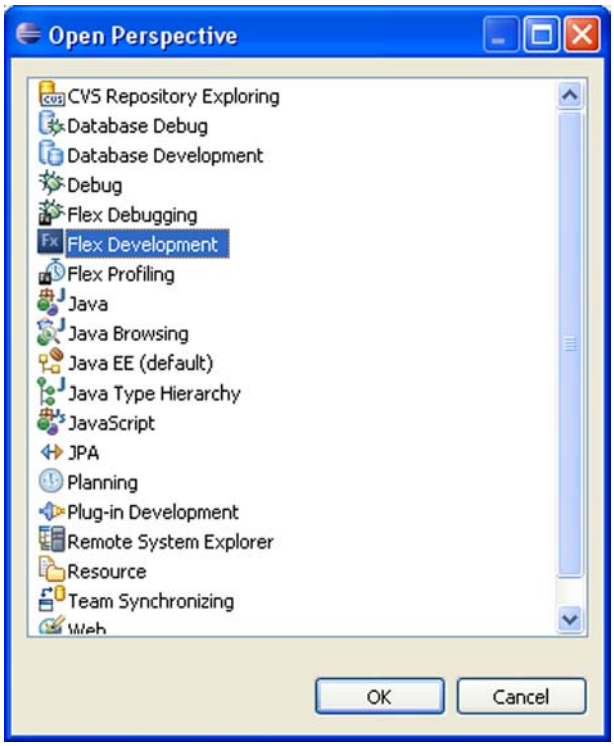


3.3 Create the Flex Code for the PHP Service “helloUser”

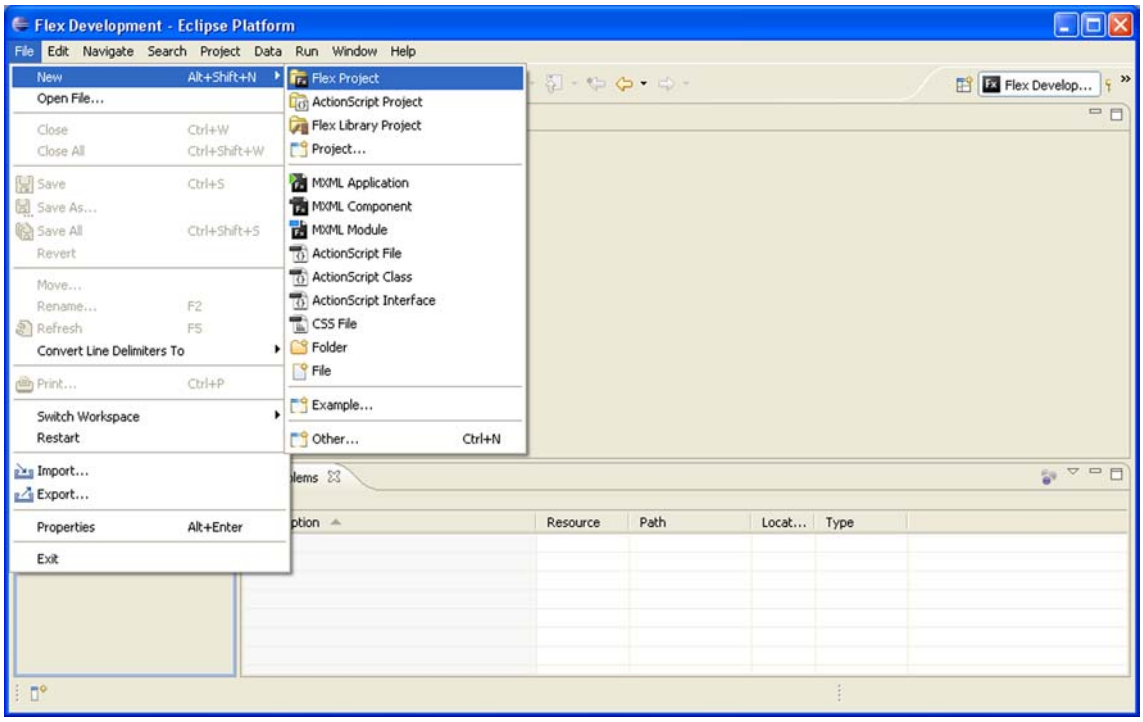
Please open your Flex IDE, I’m using the Flex Builder 3 IDE. Please open the “Flex Development” perspective with menu Window → Open Perspective → Other ...



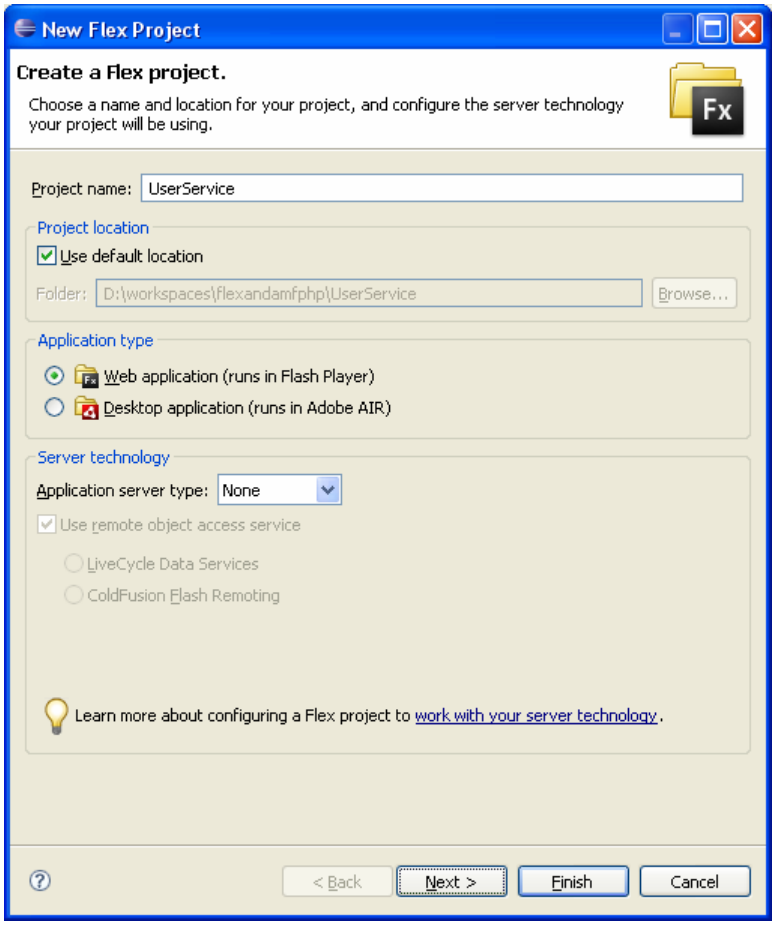
In the window “Open Perspective” select the item “Flex Development” and press the “OK” button.



Crete a new flex project with File → New → Flex Project



In the window “New Flex Project” insert the project name UserService and press the “Finish” button.



Please open the mxml file “UserService.mxml” and insert a Button and a Label component between the Application-Tags.

```
<mx:Button id="btnSend" label="Send" x="10" y="20" />
<mx:Label id="lblResult" x="10" y="110" text="" />
```

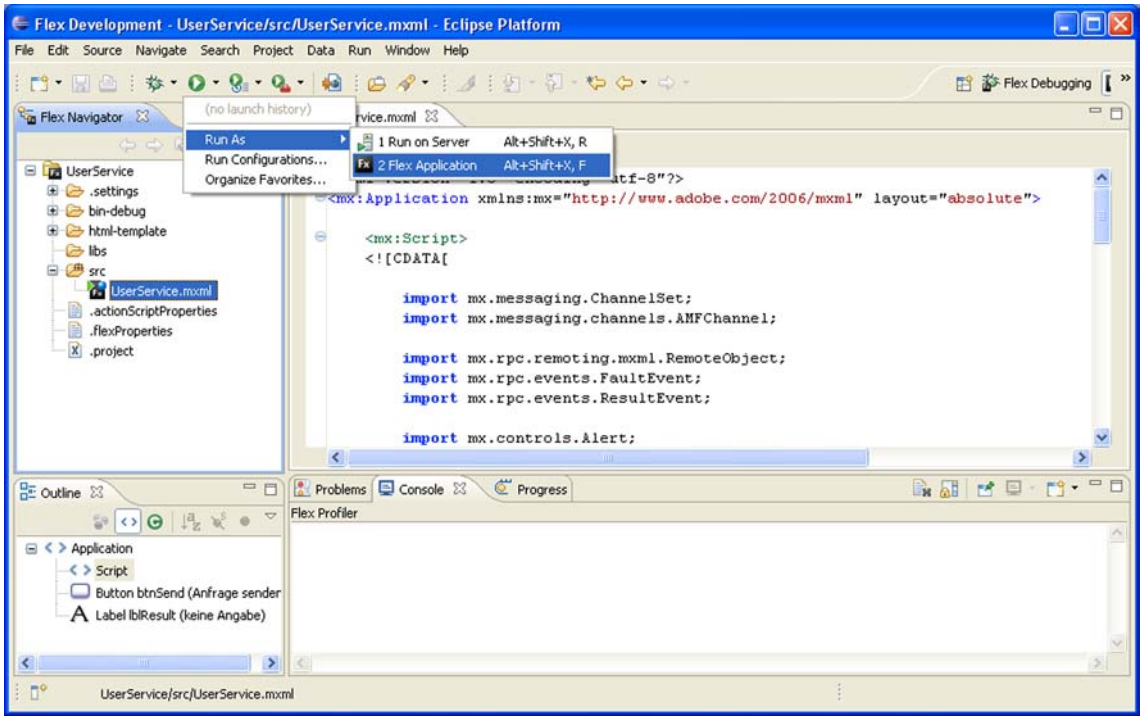
Please copy the following ActionScript-Code between the first Application-Tag and the Button-Tag with the id “btnSend”. It may be possible that you have to change the url from the private variable _gateway.

```
<mx:Script>
<![CDATA[

import mx.messaging.ChannelSet;
import mx.messaging.channels.AMFChannel;
import mx.rpc.remoting.mxml.RemoteObject;
import mx.rpc.events.FaultEvent;
import mx.rpc.events.ResultEvent;
import mx.controls.Alert;

private var _gateway:String =
  "http://www.flexandamfphp.xyz/amfphp-1.9.beta.20080120/amfphp/gateway.php";

private var _phpServiceClass:String = "UserService";

private var _amfService:RemoteObject = null;

private function onBtnSendClicked():void {

       var channel:AMFChannel = new AMFChannel( "my_amfphp", _gateway );
       var channelSet:ChannelSet = new ChannelSet();
       channelSet.addChannel( channel );

       _amfService = new RemoteObject();
       _amfService.destination = "amfphp";
       _amfService.showBusyCursor = true;
       _amfService.channelSet = channelSet;

       _amfService.source = _phpServiceClass;
       _amfService.addEventListener( ResultEvent.RESULT, onSuccess );
       _amfService.addEventListener( FaultEvent.FAULT, onFault );
       _amfService.getOperation( "helloUser" ).send();
}

private function onSuccess( event:ResultEvent ):void {
       destroy();
       lblResult.text = event.result.toString();
}

private function onFault( event:FaultEvent ):void {
       destroy();
       Alert.show(
         event.fault.faultDetail + ", " +
         event.fault.faultString, "Fehler"
       );
}

private function destroy():void {
       _amfService.removeEventListener( ResultEvent.RESULT, onSuccess );
       _amfService.removeEventListener( FaultEvent.FAULT, onFault );
       _amfService = null;
}

]]>
</mx:Script>
```

Please add the click event handler on the button component “btnSend”. The click event calls the `onBtnSendClicked` function.

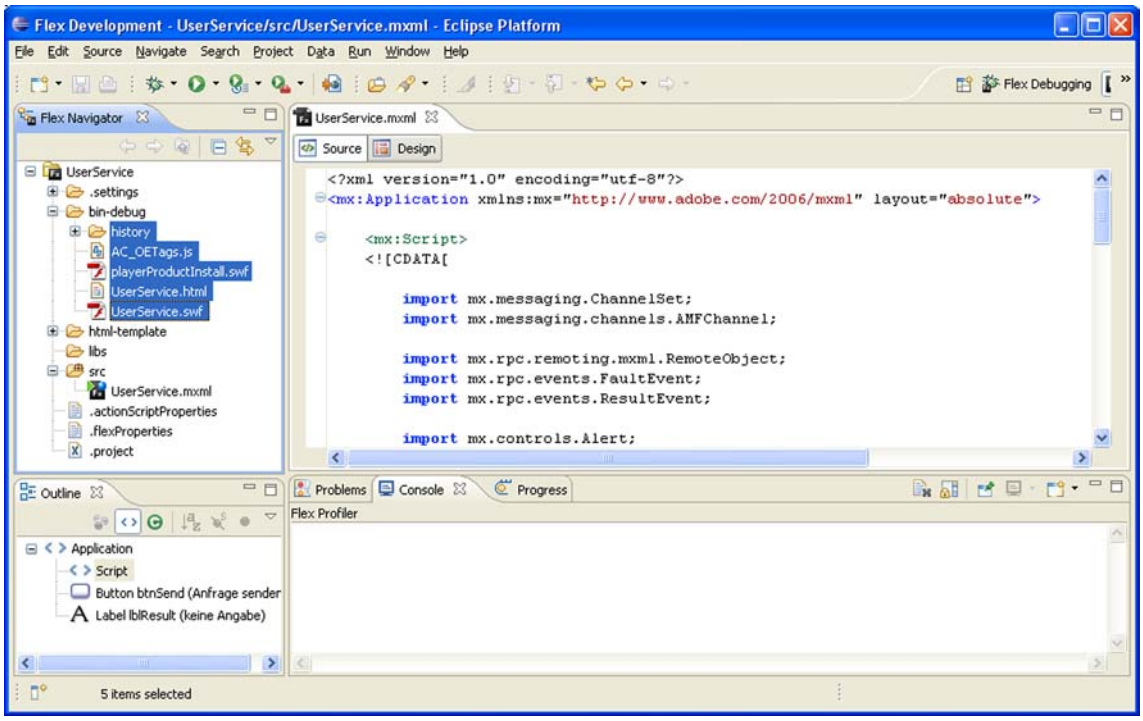
```
<mx:Button
  id="btnSend" label="Send" x="10" y="20" click="onBtnSendClicked();"
/>
```

Now you can run the flex project. Press the black triangle right from the “Run” icon. Then select “Run As” and the item “2 Flex Application”.

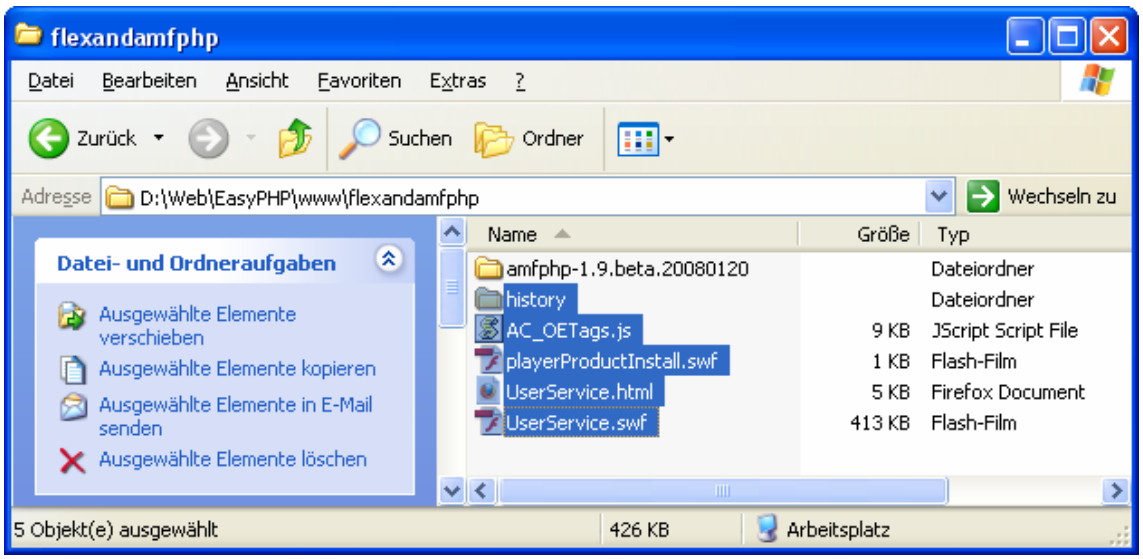


Now the flex application opens your default browser and you should see the swf file with the “Send” button on the stage. When you press the “Send” button you’ll see the message “Hello Test User!” from the PHP service.

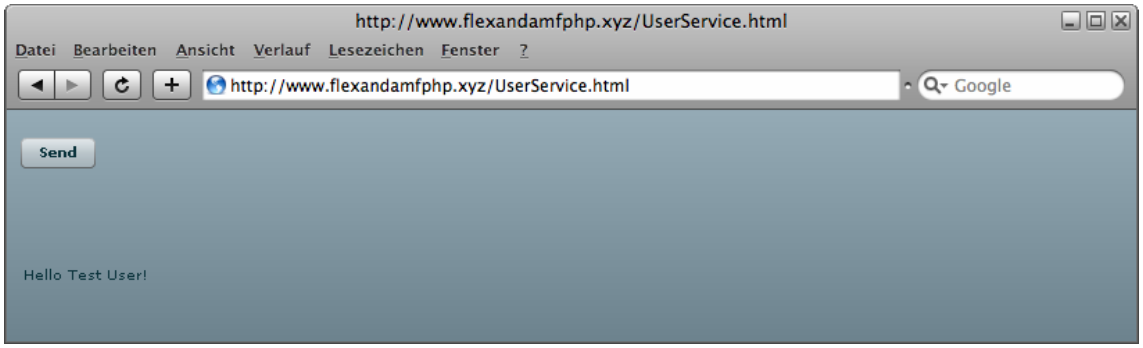
You can also copy the content from the bin-debug folder to your DOCROOT folder and open the browser with the url http://localhost/UserService.html.



The DOCROOT folder looks now like the following screen.



Now you can test the flex application UserService with the url http://localhost/UserService.html.



4 PHP Service “calculate” (Part 2)

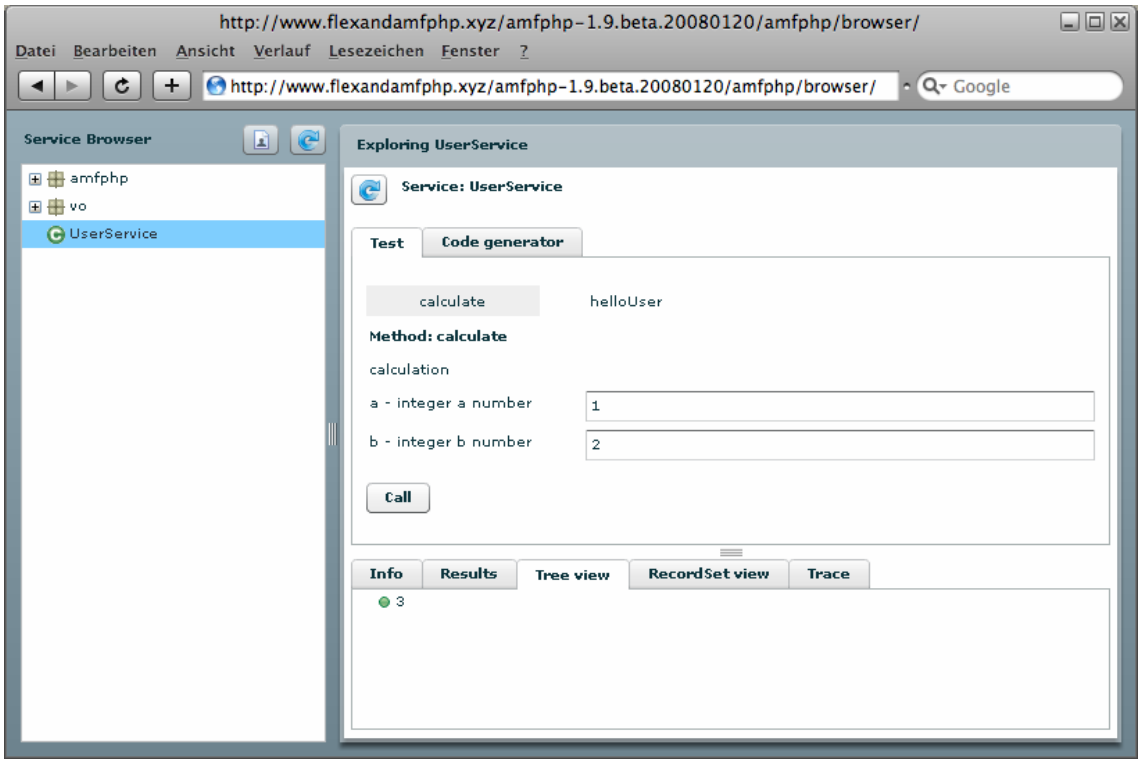
4.1 Create the PHP function “calculate”

Open the PHP file UserService.php and copy the following code under the function “helloUser” from the first part.

```
/**
 *   calculation
 *   @access public
 *   @param  integer $a          number
 *   @param  integer $b          number
 *   @return integer $result     $a and $b
 */
public function calculate( $a, $b ) {
  return $a + $b;
}
```

4.2 Test PHP Service “calculate” in the Service Browser

Please open the service browser with the url http://localhost/amfphp-1.9.beta.20080120/amfphp/browser. Then select the function “calculate” in the right panel. Now you can test the function “calculate” with the two parameters a and b.



4.3 Create the Flex Code for the PHP Service “calculate”

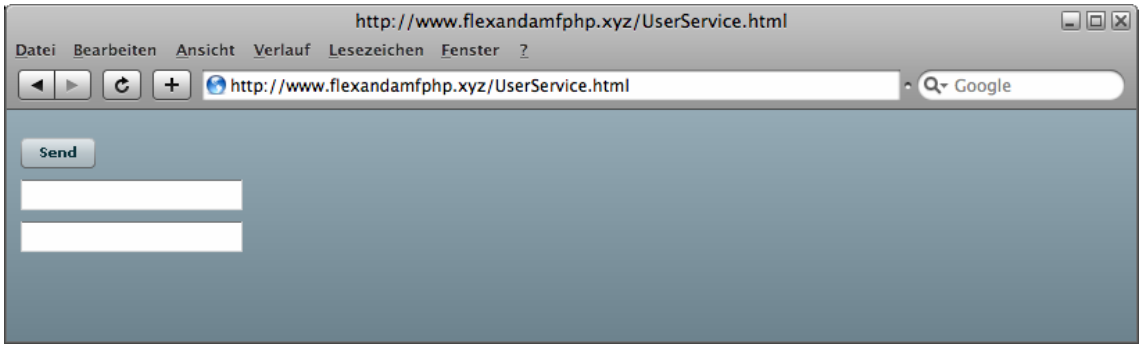
Insert two TextInput Components over the Button Component btnSend from part one for the two parameters a and b.

```
<mx:TextInput id="tiNumber1" x="10" y="50" />
<mx:TextInput id="tiNumber2" x="10" y="80" />
```

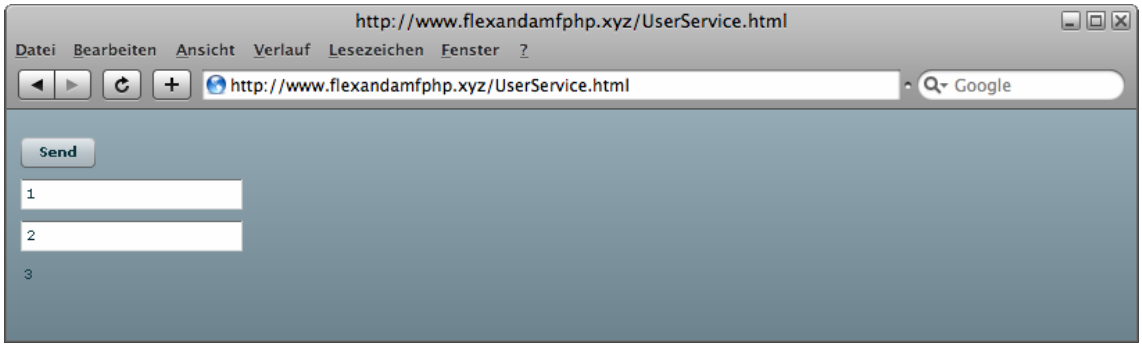
Now we want to call the PHP function “calculate” with the two TextInput parameters and we have to change the getOperation function into:

```
_amfService.getOperation( "calculate" ).send(
  Number(tiNumber1.text), Number(tiNumber2.text)
);
```

When you run the flex project you’ll see the following screen with the two TextInput components.



After writing two numbers into the TextInput Components and pressing the “Send” button, you’ll see the calculation result.



5 PHP Service “getUser” (Part 3)

5.1 MySQL-Database Settings

In this part of the documentation we need the following MySQL-Database connection settings. May be you have to use your own DB-HOST parameter.

DB-Setting	Value
DB-HOST	localhost
DB-Name	db_flex_and_php
DB-Table	tbl_user
DB-User	u_flex_and_php
DB-Password	pwd123456
DB-CHARACTER-SET	utf8
DB-COLLATION	utf8_general_ci

Please open your database administration tool and run the following five MySQL commands:

```
CREATE DATABASE `db_flex_and_php` DEFAULT CHARACTER SET utf8 COLLATE
utf8_general_ci;
```

```
CREATE TABLE `tbl_user` (
  `user_id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `name` VARCHAR( 100 ) NOT NULL,
  `street` VARCHAR( 100 ) NOT NULL,
  `zip` CHAR( 10 ) NOT NULL,
  `city` VARCHAR( 100 ) NOT NULL,
  `message` TEXT NOT NULL
) ENGINE = MYISAM;
```

```
INSERT INTO `tbl_user`
  ( `user_id` , `name` , `street` , `zip` , `city` , `message` ) VALUES
  ( NULL , 'Rico Schneider', '4th Avenue 321', '112233', 'New York', ''),
  ( NULL , 'Kevin Lynch', '5th Avenue 123', '112244', 'Washington D.C.', '');
```

```
GRANT USAGE ON * . * TO 'u_flex_and_php'@'localhost' IDENTIFIED BY 'pwd123456'
WITH MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0 MAX_UPDATES_PER_HOUR 0;
```

```
GRANT SELECT, INSERT, UPDATE, DELETE ON `db_flex_and_php` .* TO
'u_flex_and_php'@'localhost';
```

After the MySQL database and user setup, we can create the next PHP service.

5.2 Create the PHP file “db.inc.php”

Please create the folder “include” in the DOCROOT folder. Open the include folder and create a new PHP file “db.inc.php”. Copy the following PHP code in the file “db.inc.php”.

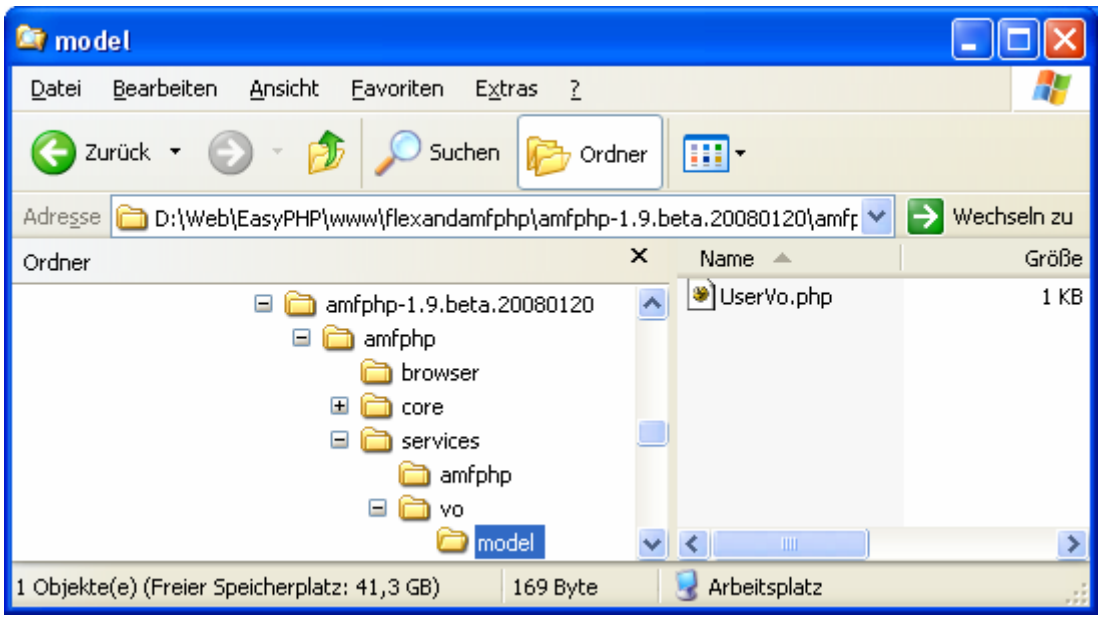
```
<?php

  define("dbUser", "u_flex_and_php");
  define("dbPwd", "pwd123456");
  define("dbHost", "localhost");
  define("dbName", "db_flex_and_php");

?>
```

Please check the PHP constants and change the values if it’s necessary.

5.3 Create the PHP file “UserVo.php”

Please open your services folder. The services folder is the folder with your PHP service UserService.php inside. Create the folder “vo” in the services folder. Open the vo folder and create the folder “model” inside. Open the model folder and create a new PHP file “UserVo.php”. Copy the following PHP code in the file “UserVo.php”.

```
<?php

class UserVo {

  var $user_id;
  var $name;
  var $street;
  var $zip;
  var $city;
  var $message;

  var $_explicitType = "model.UserVo";
}

?>
```



5.4 Create the PHP function “getUser”

Open the PHP file UserService.php and copy the following code under the function calculate from part two.

```
/**
 *   get a user object list
 *   @access public
 *   @return array   $result     UserVo list
 */
public function getUser() {

  $dbQuery = "SELECT * FROM tbl_user";

  $link = mysql_connect( dbHost, dbUser, dbPwd ) or
    error_log( "db connection error - " . mysql_error() );

  mysql_select_db( dbName ) or error_log( "db selection error" );

  $dbResult = mysql_query( $dbQuery ) or
    error_log( "sql query request error - " . mysql_error() );

  require_once( "vo/model/UserVo.php" );

  $result = array();

  while ( $line = mysql_fetch_array( $dbResult, MYSQL_ASSOC ) ) {
    $vo = new UserVo();
    $vo->user_id = (int) $line["user_id"];
    $vo->name = $line["name"];
    $vo->street = $line["street"];
    $vo->zip = $line["zip"];
    $vo->city = $line["city"];
    $vo->message = $line["message"];
    $result[] = $vo;
  }

  mysql_free_result( $dbResult );
  mysql_close( $link );

  return $result;
}
```

Please copy the following line

```
require_once( "../../../include/db.inc.php" );
```

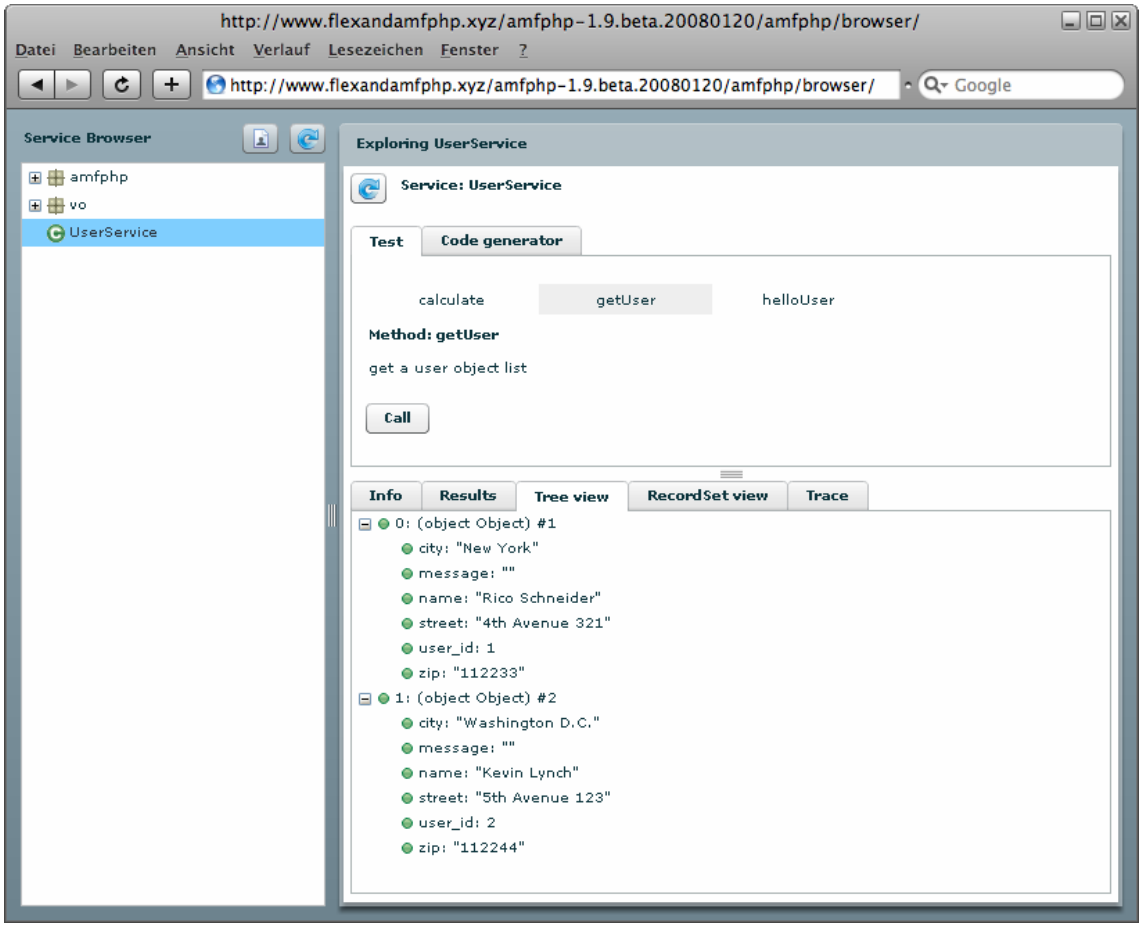
over the class definition

```
class UserService {
```

in the file UserService.php.

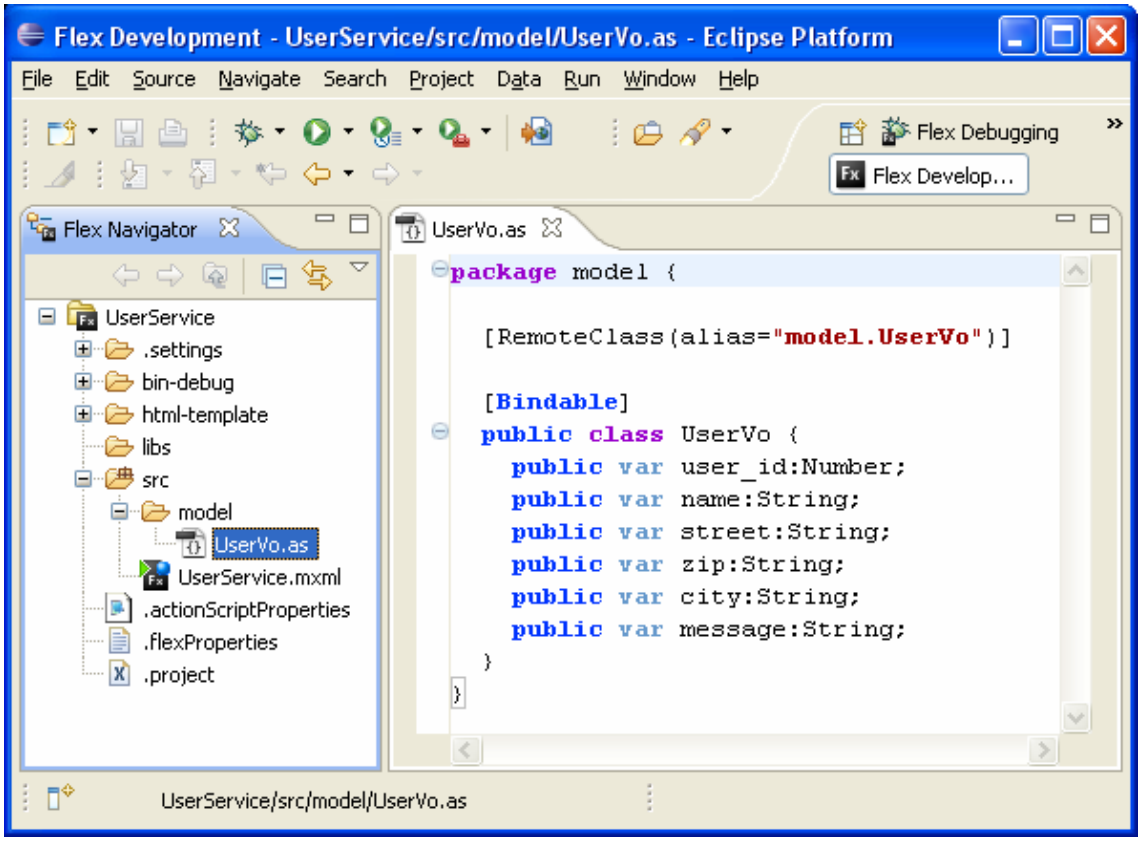
5.5 Test PHP Service “getUser” in the Service Browser

Please open the service browser with the url http://localhost/amfphp-1.9.beta.20080120/amfphp/browser. Please select the function getUser in the right panel. Now you can test the function getUser.



5.6 Create the Flex Code for the PHP Service “getUser”

First of all, we need the corresponding ActionScript value object to the PHP value object. Create a new ActionScript class UserVo. The ActionScript class file has to be in the new model folder into the src folder.



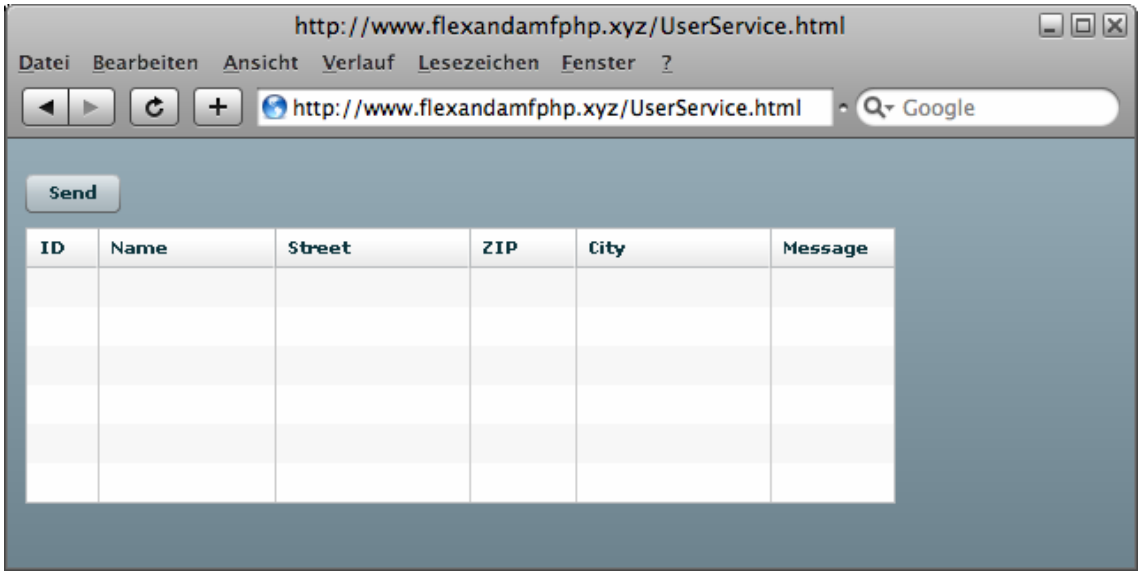
Replace the Script block in the file UserService.mxml with the code from the next page. Please notice that you have to check the private _gateway variable again.

```
<mx:Script>
<![CDATA[
import model.UserVo;
import mx.messaging.ChannelSet;
import mx.messaging.channels.AMFChannel;
import mx.rpc.remoting.mxml.RemoteObject;
import mx.rpc.events.FaultEvent;
import mx.rpc.events.ResultEvent;
import mx.controls.Alert;

private var _gateway:String =
  "http://www.flexandamfphp.xyz/amfphp-1.9.beta.20080120/amfphp/gateway.php";

private var _phpServiceClass:String = "UserService";
private var _amfService:RemoteObject = null;

[Bindable]
private var _userList:Array;

private function onBtnSendClicked():void {

       var channel:AMFChannel = new AMFChannel( "my_amfphp", _gateway );
       var channelSet:ChannelSet = new ChannelSet();
       channelSet.addChannel( channel );

       _amfService = new RemoteObject();
       _amfService.destination = "amfphp";
       _amfService.showBusyCursor = true;
       _amfService.channelSet = channelSet;

       _amfService.source = _phpServiceClass;
       _amfService.addEventListener( ResultEvent.RESULT, onSuccess );
       _amfService.addEventListener( FaultEvent.FAULT, onFault );
       _amfService.getOperation( "getUser" ).send();
}

private function onSuccess( event:ResultEvent ):void {
       destroy();
       _userList = new Array();
       for( var i:uint = 0; i < event.result.length; i++ ) {
              _userList[i] = event.result[i] as UserVo;
       }
}

private function onFault( event:FaultEvent ):void {
       destroy();
       Alert.show(
         event.fault.faultDetail + ", " +
         event.fault.faultString, "Fehler"
       );

private function destroy():void {
       _amfService.removeEventListener( ResultEvent.RESULT, onSuccess );
       _amfService.removeEventListener( FaultEvent.FAULT, onFault );
       _amfService = null;
}

]]>
</mx:Script>
```

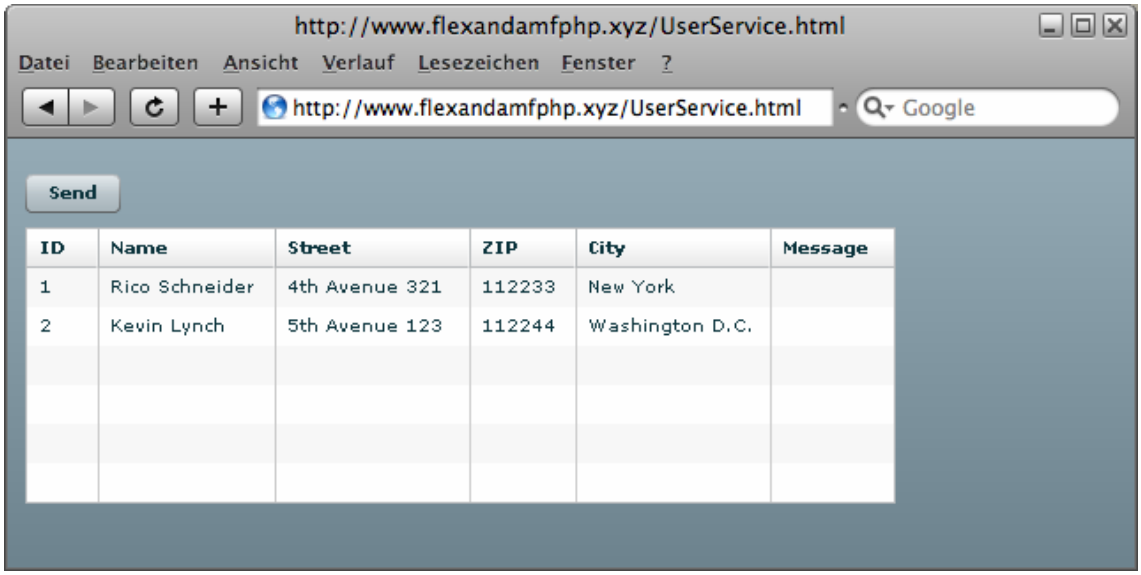
Replace the MXML code after the Script block.

```
<mx:Button
  id="btnSend" label="Send" x="10" y="20" click="onBtnSendClicked();"
/>

<mx:DataGrid x="10" y="50" id="dgUser" dataProvider="{_userList}">
<mx:columns>
<mx:DataGridColumn headerText="ID" dataField="user_id" width="40" />
<mx:DataGridColumn headerText="Name" dataField="name" width="100" />
<mx:DataGridColumn headerText="Street" dataField="street" width="110" />
<mx:DataGridColumn headerText="ZIP" dataField="zip" width="60" />
<mx:DataGridColumn headerText="City" dataField="city" width="110" />
<mx:DataGridColumn headerText="Message" dataField="message" width="70" />
</mx:columns>
</mx:DataGrid>
```

When you run the flex project you'll see the following screen.



After pressing the Send button you'll see the following screen.



6 PHP Service “storeUser” (Part 4)

In the last part of the documentation I’ve added the function storeUser.

7 ValueObjects-Path Settings

The ValueObjects-Path setting is in the folder http://localhost/amfphp-1.9.beta.20080120/amfphp. The file gateway.php includes the file globals.php. You can change the ValueObjects path setting in the file globals.php in line 13.

File	Line	Code-Snippet
globals.php	13	`$voPath = "services/vo/";`
gateway.php	122	`$gateway->setClassMappingsPath($voPath);`

8 Setup Amfphp Production Version

- Delete the folder browser
  The service browser was designed as a debugging tool and has no access controls. Never place the service browser on a production server.

- Delete the amfphp folder in the services folder

- Change the line 106 in the file gateway.php in the folder amfphp from
  `define(”PRODUCTION_SERVER”, false);`
  to
  `define(”PRODUCTION_SERVER”, true);`

- Delete the file phpinfo.php in the folder amfphp

- Delete the file place_services_here.txt in the services folder

9 Source Files

You can download all the source files from the url http://blog.schneider-webanwendungen.de/?p=55.