Integração do Flex com PHP através do AMFPHP

Eduardo Kraus

FloripaFlex 20/05/2009

Quem sou eu?

- Iniciei minha vida digital aos 15 anos com meu primeiro PC, um 486DX100;
- Iniciei na programação através do C++ com Dev-C++;
- Em 2005 conheci o PHP e larguei tudo e fui atrás dele;
- Em 2007 conheci o Flex e foi paixão a primeira vista.

Porque o Flex é tão bom?

- Você programa para usuário, não para browser;
- Velocidade no desenvolvimento;
- Beleza da aplicação;
- Segurança na aplicação.

Integração via XML

- Maior tráfego de dados;
- Difícil manipulação no Flex;
- http://www.google.com/search?q=flex+xml

```
public function XmlExample() {
  var employees:XML =
    <employees>
      <employee ssn="123-123-1234">
        <name first="John" last="Doe"/>
        <address>
          <street>11 Main St.</street>
          <city>San Francisco</city>
          <state>CA</state>
          <zip>98765</zip>
        </address>
      </employee>
    </employees>;
  trace(employees.employee[0].address.zip);
  trace(employees.employee[1].@ssn);
  trace(employees.employee.name);
  trace(employees..zip[0]);
  trace(employees..@ssn[1]);
  trace(employees..name);
}
```

Fonte: http://livedocs.adobe.com/flex/3/langref/XML.html

Integração via HTTPService

- Tráfego mais rápido que o XML;
- Ainda continua difícil a interpretação no Flex;
- http://www.google.com/search?q=flex+amf

```
private var gateway : RemotingConnection;
private function doLogin(): void{
  gateway = new
RemotingConnection("amfphp/gateway.php");
  gateway.call("classes.doLogin", new
Responder(onResult,
onFault),inUsuario.text,inSenha.text);
}
private function onResult( result:Array) : void{
  Usuario.getInstance().id =
parseInt(result[0]["id"]);
  Usuario.getInstance().nome =
result[0]["nome"].toString();
  Usuario.getInstance().usuario =
result[0]["usuario"].toString();
  Usuario.getInstance().permissao =
parseInt(result[0]["permissao"]);
}
```

Fonte: Projeto antigo

Integração via RemoteObject

- Mesma velocidade de tráfego que o HTTPService com AMFPHP;
- Muito fácil de manipular os dados;
- Muito fácil implementar os chamados para o AMFPHP.

Porque o AMFPHP

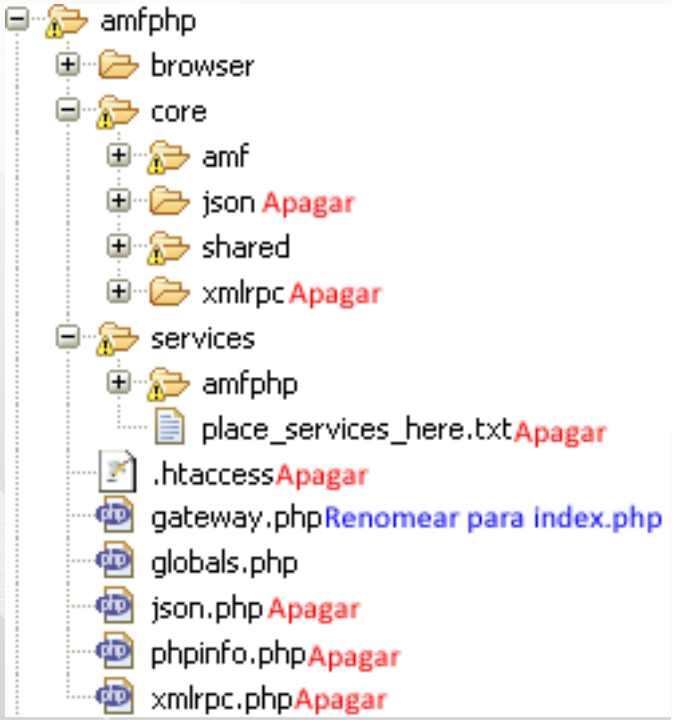
- Muito fácil de utilizar;
- Código aberto;
- Comunidade ampla, facilitando a ajuda;
- Tempo de instalação praticamente zero.

Vamos ao que interessa

O AMFPHP

Baixe-o em http://www.amfphp.org/ descompacte-o na pasta “src” e após isso aconselho a fazer algumas modificações.

Apagar os arquivos e pastas desnecessários e principalmente renomear o gateway.php para index.php

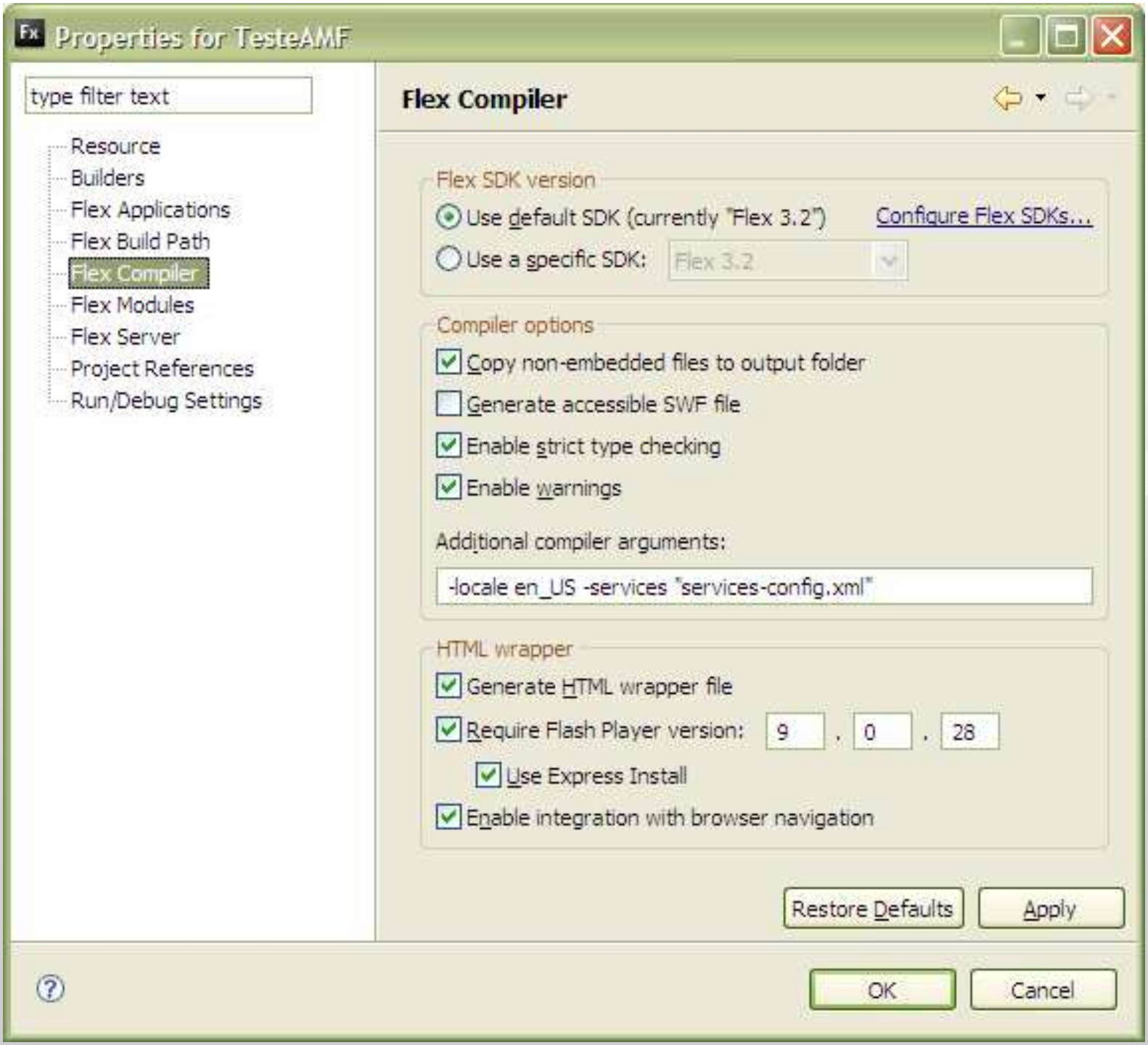


O Arquivo services-config.xml

O arquivo services-config.xml é responsável pela configuração do RemoteObject no Flex Builder.

Este arquivo deve ser passado como argumento na compilação. Para isso você vai a *Proprietes >> Flex Compiler* e em *additional compiler arguments:* e adicione o seguinte

```
-locale en_US -services "services-config.xml"
```


Properties for TesteAMF
type filter text
Resource
Builders
Flex Applications
Flex Build Path
Flex Compiler
Flex Modules
Flex Server
Project References
Run/Debug Settings
Flex Compiler
Flex SDK version
Use default SDK (currently "Flex 3.2")
Configure Flex SDKs...
Use a specific SDK:
Flex 3.2
Compiler options
Copy non-embedded files to output folder
Generate accessible SWF file
Enable strict type checking
Enable warnings
Additional compiler arguments:
-locale en_US -services "services-config.xml"
HTML wrapper
Generate HTML wrapper file
Require Flash Player version:
9
0
28
Use Express Install
Enable integration with browser navigation
Restore Defaults
Apply
OK
Cancel

O Arquivo services-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<services-config>
  <services>
    <service id="amfphp-flashremoting-service"
         class="flex.messaging.services.RemotingService"
         messageTypes="flex.messaging.messages.RemotingMessage">
      <destination id="CanalAmfphp">
        <channels>
          <channel ref="my-amfphp"/>
        </channels>
        <properties>
          <source>*</source>
        </properties>
      </destination>
    </service>
  </services>
  <channels>
    <channel-definition id="my-amfphp"
              class="mx.messaging.channels.AMFChannel">
      <endpoint uri="amfphp/"
            class="flex.messaging.endpoints.AMFEndpoint"/>
    </channel-definition>
  </channels>
</services-config>
```

Destino da conexão

URL para conexão com o AMFPHP

Realizando uma conexão

```
private function listar():void
{
   var remote:RemoteObject = new RemoteObject("CanalAmfphp");
    remote.showBusyCursor = true;
    remote.source = "vo.UsuarioService";
    remote.addEventListener(ResultEvent.RESULT, resultListar)
    remote.addEventListener(FaultEvent.FAULT, ErroListar)
    remote.getOperation("Listar").send();
}
private function ErroListar(e:FaultEvent):void
{
    Alert.show(e.fault.faultString,
                e.fault.faultCode.toString());
}
private function resultListar(e:ResultEvent):void
{
    dgUsuarios.dataProvider = e.result as Array;
}
```

Apresentando os dados

```
<mx:Button x="10" y="10" label="Listar usuários" click="listar()"/>
   <mx:DataGrid x="10" y="40" width="400" id="dgUsuarios"
       itemClick="selecionaUsuario()" height="144">
       <mx:columns>
           <mx:DataGridColumn headerText="Nome completo" dataField="Nome"/>
           <mx:DataGridColumn headerText="E-mail" dataField="Email"/>
           <mx:DataGridColumn headerText="Endereço" dataField="Endereco"/>
       </mx:columns>
   </mx:DataGrid>
   <mx:Label x="10" y="208" text="ID:" width="110" textAlign="right"
  fontWeight="bold"/>
   <mx:Label x="10" y="234" text="Nome completo:" width="110"
  textAlign="right" fontWeight="bold"/>
   <mx:Label x="10" y="260" text="E-mail:" width="110" textAlign="right"
  fontWeight="bold"/>
   <mx:Label x="10" y="286" text="Endereço:" width="110" textAlign="right"
  fontWeight="bold"/>
   <mx:Label x="128" y="208" text="{selectedPerson.ID}"/>
   <mx:Label x="128" y="234" text="{selectedPerson.Nome}"/>
   <mx:Label x="128" y="260" text="{selectedPerson.Email}"/>
   <mx:Label x="126" y="286" text="{selectedPerson.Endereco}"/>
```

As classes Usuario

A classe vo/Usuario.php

```php
class Usuario {
  var $_explicitType = "vo.Usuario";

  var $ID;
  var $Nome;
  var $Email;
  var $Endereco;
}
```

A classe vo/Usuario.as

```actionscript
package vo
{
  [RemoteClass(alias="vo.Usuario")]
  [Bindable]
  public class Usuario
  {
    public var ID:String;
    public var Nome:String;
    public var Email:String;
    public var Endereco:String;
  }
}
```

As classes Usuario

A classe vo/Usuario.php

```php
class Usuario {
  var $_explicitType = "qualquerCoisaAqui";

  var $ID;
  var $Nome;
  var $Email;
  var $Endereco;
}
```

A classe vo/Usuario.as

```actionscript
package vo
{
  [RemoteClass(alias="qualquerCoisaAqui")]
  [Bindable]
  public class Usuario
  {
    public var ID:String;
    public var Nome:String;
    public var Email:String;
    public var Endereco:String;
  }
}
```

A classe UsuarioService

```php
require("Usuario.php");
class UsuarioService
{
  function Listar()
  {
    $retorno = '';

    $retorno[] = new Usuario(1, "Eduardo Kraus", "contato@mxml.com.br",
   "Palhoça SC");
    $retorno[] = new Usuario(2, "Usuarios 2",  "usuario@mxml.com.br",
   "Florianópolis SC");
    $retorno[] = new Usuario(3, "Estagiario",  "estagiario@mxml.com.br", "São
   José SC");
    return $retorno;
  }
}
```

A aplicação


Listar usuários
Nome completo
E-mail
Endereço
ID:
Nome completo:
E-mail:
Endereço:

A aplicação


Listar usuários
Nome completo
E-mail
Endereço
Eduardo Kraus
contato@mxml.com.br
Palhoça SC
Usuarios 2
usuario@mxml.com.br
Florianópolis SC
Estagiario
estagiario@mxml.com
São José SC
ID:
Nome completo:
E-mail:
Endereço:

A aplicação


Listar usuários
Nome completo
E-mail
Endereço
Eduardo Kraus
contato@mxml.com.br
Palhoça SC
Usuarios 2
usuario@mxml.com.br
Florianópolis SC
Estagiario
estagiario@mxml.com
São José SC
ID: 1
Nome completo: Eduardo Kraus
E-mail: contato@mxml.com.br
Endereço: Palhoça SC

```
    [Bindable] private var selectedPerson:Usuario;
    private function listar():void{
      var remote:RemoteObject = new RemoteObject("CanalAmfphp");
      remote.showBusyCursor = true;
      remote.source = "vo.UsuarioService";
      remote.addEventListener(ResultEvent.RESULT, resultListar)
      remote.addEventListener(FaultEvent.FAULT,   ErroListar)
      remote.getOperation("Listar").send();
    }
    private function ErroListar(e:FaultEvent):void{
      Alert.show(e.fault.faultString, e.fault.faultCode.toString());
    }
    private function resultListar(e:ResultEvent):void{
      dgUsuarios.dataProvider = e.result as Array;
    }
    private function selecionaUsuario():void{
      selectedPerson = Usuario(dgUsuarios.selectedItem);
    }
<mx:Button x="10" y="10" label="Listar usuários" click="listar()"/>
<mx:DataGrid x="10" y="40" width="400" id="dgUsuarios"
  itemClick="selecionaUsuario()" height="144">
  <mx:columns>
    <mx:DataGridColumn headerText="Nome completo" dataField="Nome"/>
    <mx:DataGridColumn headerText="E-mail"    dataField="Email"/>
    <mx:DataGridColumn headerText="Endereço"    dataField="Endereco"/>
  </mx:columns>
</mx:DataGrid>
<mx:Label x="128" y="208" text="{selectedPerson.ID}"/>
<mx:Label x="128" y="234" text="{selectedPerson.Nome}"/>
<mx:Label x="128" y="260" text="{selectedPerson.Email}"/>
<mx:Label x="126" y="286" text="{selectedPerson.Endereco}"/>
```

Podemos melhorar a conexão

Criar uma classe global que fará a conexão e em caso de erro apresente em um Alert e em sucesso retorne para a nossa função:

```
public class RemoteObjectAMFPHP extends RemoteObject{
  public function RemoteObjectAMFPHP():void{
    super("CanalAmfphp")
  }
  public function Remoto(_source:String, result:Function):void{
    this.source = _source;
    this.showBusyCursor = true;
    this.addEventListener(ResultEvent.RESULT, result);
    this.addEventListener(FaultEvent.FAULT, Falha);
  }
  private function Falha(e:FaultEvent):void{
    Alert.show(e.fault.faultString, e.fault.faultCode.toString());
  }
}
```

Podemos melhorar a conexão

Deste modo a chamada fica mais simples:

```
private function listar():void{
  var remote:RemoteObjectAMFPHP = new RemoteObjectAMFPHP()
  remote.Remoto("vo.UsuarioService", resultListar);
  remote.getOperation("Listar").send();
}
private function resultListar(e:ResultEvent):void{
  dgUsuarios.dataProvider = e.result as Array;
}
```

Falando de segurança

Nunca estamos seguros o suficiente

Métodos abertos

- Cuidado com métodos que somente serão acessados por usuário logado;
- Cuidado com classes que manipulam banco de dados, nunca as deixe dentro da pasta “services”;
- Cuidado com métodos que apagam arquivos;
- Nunca envie as senhas, mesmo criptografadas, nos métodos que os listam.
- Antes de publicar não esquerça de apagar as pastas amfphp/browser e amfphp/services/amfphp

Sempre o Upload

Quem que não tem um arquivo upload.php com o seguinte conteudo?

```php
<?php
  $strOrigem = $_FILES['Filedata'] ['tmp_name'];
  $strDestino = "../upload/".$_FILES['Filedata']['name'];
  if(!move_uploaded_file ($strOrigem, $strDestino)){
    $erroUp->erro("Erro ao mover");
  }
  else{
    $erroUp->erro("Movido com sucesso");
  }
?>
```

Perguntas

?

Para onde ir agora

- http://blog.mxml.com.br/
- http://www.amfphp.org/
- http://www.adobe.com/devnet/flex/
- http://www.adobe.com/devnet/flex/tourdeflex/
- http://www.adobe.com/devnet/flex/flex_php.html
- http://flex.org/
- http://livedocs.adobe.com/flex/3/langref/
- http://livedocs.adobe.com/flex/3/html/
- http://opensource.adobe.com

Obrigado

eduardokraus@gmail.com

contato@mxml.com.br