ADOBE FLEX INTEGRADO COM JAVA

Fabiano Silva de Carvalho
Luiz Mauro de Pádua Silveira
Ana Carolina do Prado
Jéssica Ellen da Silva Ferreira Jacinto
Rogério Ferreira da Silva
Sadraque Dias Soares

RESUMO

Este artigo foi desenvolvido com o objetivo de elicitar e aglomerar informações a fim de compor e disseminar conhecimento sobre a linguagem de programação Java e o framework Flex. A linguagem de programação Flex veio como auxilio ao Java, trazendo inovações a fim de tornar o desenvolvimento de softwares muito mais ágil e o produtivo. O Flex caracteriza – se em um framework web baseado em MVC (Padrão de Projeto) que combina os principais frameworks utilizados na plataforma Java e respeita o “paradigma” Convention – over – configuration (programação por convenção). A seguir será realizada uma breve apresentação do assunto de forma a esclarecer conceitos e teorias com o objetivo de enfocar na inovação provocada por essas tecnologias.

Palavras – chave: Dinâmica; Framework; Integração; Flex; Java;

1. INTRODUÇÃO

Os Computadores são maquinas que agem por meio de instruções, nós seres humanos deveram programá-los de acordo com o resultado que desejarmos. O mercado de hoje não tem de esperar tanto para soluções de problemas simples e frequentes, o mercado quer produtividade e qualidade. Para suprir estas necessidades à quantidade e diversidade de linguagens de programação estão crescendo bastante no mundo inteiro, a cada ano as empresas deparam com desenvolvedores apresentando linguagens e tecnologias novas para seus gerentes. Quando e qual caminho escolher?

2. O QUE É FLEX

Flex é o nome de um conjunto de tecnologias, lançada em Março de 2004 pelo Adobe, feito para aplicação rica para a internet baseadas na plataforma do Flash. Os arquivos desenvolvidos em Flex possuem a extensão MXML, a qual é uma linguagem de Marcação. Para visualizar uma aplicação MXMl o usuário faz uma requisição pelo seu servidor, o servidor recebe a requisição, o servidor do Flex compila o arquivo MXML para SWF(formato Flash – Binário), este será rodado no Flash Player que será na maquina do cliente e visualizado no navegador.

3. O ADOBE BLAZE DS

O BlazeDS é um produto Open Source (Licença LGPL v3) que corresponde à tecnologia JAVA server-side que dá suporte tanto para o Remoting assim como ao Messaging de objetos trocados entre o Java e o Flex/Flash. Com o BlazeDS você pode gerar vários tipos de canais de conexão, em detaque é o streaming sobre o protocolo HTTP, que antes era provido por alguns protocolos como o RMTP da Adobe/Macromedia e o MMS da Microsoft.

Outro destaque muito importante para toda a comunidade Flex/Flash mundial, é que o serviço de data-push também foi disponibilizado de graça, para quem não

conhece, neste longo artigo quero apresentar um exemplo de implementação desta poderosa tecnologia. Para aqueles que conhecem padrões de projeto, é algo semelhante ao Pattern Observer.

3. COMUNICAÇÃO FLEX E JAVA

O Flex é executado a partir de uma máquina virtual, logo o programador apenas se preocupa em desenvolver a interface não em programar compatibilidade entre browsers. Você programa sua interface totalmente orientada a objetos, isso visa reuso de componentes, desenvolvimento de módulos e afins. Há várias maneiras de comunicar o Java com o Flex, mas em destaque é que você pode trocar objetos Java/Flex por um protocolo que provê compactação e transferência binária, este é o AMF.

Exemplo de Comunicação no Java Flex

```
<?xml version="1.0" encoding="UTF-8" ?>
- <service id="remoting-service" class="flex.messaging.services.RemotingService">
- <adapters>
<adapter-definition id="java-object"
class="flex.messaging.services.remoting.adapters.JavaAdapter" default="true" />
</adapters>
- <default-channels>
<channel ref="my-amf" />
</default-channels>
- <destination id="modulo">
- <properties>
<source>agenda.controle.ControleSeguranca</source>
</properties>
</destination>
</service>
```

```
<mx:RemoteObject
id="operacao"
result="event.token.resultHandler(event)"
fault="event.token.faultHandler(event)"
destination="modulo"
/>
```

Id Corresponde a identificação do objeto
Result Corresponde a resultado positivo do processo efetuado com sucesso.
Fault Corresponde a resultado negativo do processo não efetuado.
Destination Corresponde a configuração feita no arquivo acima.

HTTPService

Veja um exemplo de URL relativa (o arquivo dados.xml localizado no mesmo diretório do arquivo SWF):

```
<mx:HTTPService id="modulo"
url="dados.xml"
</mx:HTTPService>
```

O simples fato de se criar um objeto HTTPService não faz uma requisição automática para carregar os dados solicitados. Antes, é preciso efetuar uma chamada ao seu método.

4. OS TIPOS DE DADOS, VARIÁVEIS E CONSTANTES.

Existem os seguintes tipos de variáveis que flex aceitam que são eles: booleanos, inteiros, ponto flutuante, strings, arrays e objetos. Tipos nativos: Boolean, int, String, Void. Tipos complexos: Object, Array, Vector, XmlList, Function, RegExp.

Talvez a grande diferença entre as funções no AS3 e Java seja em como defini-las.

No Java, você pode definir qualquer quantidade de funções em um arquivo. Em AS3 você pode definir somente uma função em cada arquivo, e o nome da função deve ser o mesmo do arquivo. Por exemplo, se você criar uma função chamada doSomeMath(), você deve criar um arquivo chamado doSomeMath.as. Ao definir funções você pode usar packages. Assim, quando você precisar criar um grupo de funções utilitárias, se você não quiser criar um grupo de arquivos, você pode criar uma classe única e defini-los como métodos estáticos.

5. ESCOPO DE VARIÁVEIS

Tendo visto como variáveis, funções e classes trabalham no Flex e no AS3, chegou a hora de falar de escopo de variáveis. No Java, basicamente, você tem dois escopos: global (variáveis definidas no nível de arquivo) e local (variáveis definidas dentro de funções). No Flex há cinco escopos possíveis: dentro da função, em um método de uma instancia, em um método estático, em uma classe e no escopo global. Somando-se estes modificadores de acesso (public/private/proteced/interno) as coisas ficam um pouco mais complicadas em relação ao Java. Escopos podem ser aninhados, neste caso as variáveis / funções / membros do escopo podem ser aninhadas. Por exemplo, quando você declarar uma função anônima dentro de uma outra função, em AS3 todas as variáveis definidas na função mais externa estarão disponíveis na funções mais interna.

6. NAMESPACES

Se você está procurando um conceito equivalente em AS3 para os namespaces do PHP, você deve ler sobre a seção Packages, porque os Packages AS3 são semelhantes aos namespaces do PHP. No ActionScript, namespaces tem um significado diferente. Vamos ver para que os namespaces sejam usados, e então dar alguns exemplos:

 Prevenir conflitos de nomes (você pode criar vários métodos com o mesmo nome na mesma classe, cada um em um namespace diferente);

 Marcar variáveis e métodos através de frameworks/programas para possuírem uma visibilidade customizada (por exemplo, Flex usa o namespace chamado mx_internal; usando o namespace ao invés de private ou protected, torna possível a utilização destes métodos através de qualquer pacote ou classe a partir da estrutura do Flex. Ao mesmo tempo, desenvolvedores são alertados que estes membros não devem ser usados externamente, pois eles podem mudar);
 Implementar permissões de controle de acesso baseado em uma classe;
 Criar uma classe que pode mudar seu comportamento baseado no namespace específico selecionado;

7. DATABIND, METADA TAGS

DataBind é outra funcionalidade do Flex que torna a vida do desenvolvedor bem mais fácil, e ao mesmo temo reduz a quantidade de linhas de código. DataBin d é uma forma elegante de ligar os dados de um modelo a uma camada de layout e automaticamente alterar os dados tanto no objeto quanto no layout. Já que o Flex é usado para criar interfaces e aplicações, componentes Flex normalmente devem exibir uma grande quantidade de dados. Quando os dados são modificados, mesmo em tempo real, normalmente você deseja exibir os dados alterados mais recentes e são os dados antigos. Utilizando dataBind você consegue isso com muita facilidade. DataBind ligam a propriedade de um objeto (chamado de source) para a propriedade de outro objeto (chamado de destination), então sempre que a origem muda, o destino é automaticamente atualizado. O metadados Bindable na variável labelValue marca ele como uma fonte de dados. Em seguida, eu usei “, -” para o atributo text de um Label para marcar a propriedade como destino de uma ligação (binding). Feito o binding, a qualquer hora que a variável labelValue for alterada, o Label atualiza a propriedade text para refletir a mudança. Eu poderia usar a mesma variável para muitos Labels ou TextInputs, e todos eles poderiam ser atualizados para refletir o novo valor. Há também uma sintaxe MXML:

```
<mx:Binding source=”labelValue” destination=”myLabel.text”/>
```

8. CONCLUSÃO

As variáveis do negocio alteram a cada momento, conforme as circunstâncias surgem diferentes problemas. Para auxiliar as soluções com eficiência e eficácia é necessário identificar qual a ferramenta certa para cada ocasião. Flex não é só mais uma linguagem de programação, é diferenciada, traz resultados rápidos, aumenta a produtividade. Juntos Flex e Java tornam – se uma poderosa e excelente ferramenta para programação de sistemas vem acompanhada de tudo do que já havia no mercado, adicionando uma grande quantidade de recursos, não deixa a desejar quando comparada às linguagens consolidadas no mercado do desenvolvimento de software.

REFERÊNCIAS

http://www.adobe.com/products/flashbuilder/
http://www.flex.etc.br/