



# 企业级Flex开发框架对比

RIAMeeting.com

李文磊 郭少瑞

2010/4/23



## ADOBE® FLASH® PLATFORM



# 常见的开发框架

- Cairngorm(Adobe 官方)
- SpringActionscript ( Spring 官方 )
- Parsley
- PureMVC
- Mate
- Swiz

# 常见的开发框架

- Cairngorm(Adobe 官方)



- 轻量级架构框架，包含RIA应用中常见问题的解决模式
- 多个设计模式结合汇聚成一个比单一模式更有效的集合体
- 基于社区开发过程中遇到问题汇集而成的解决方案，帮助开发者高效解决问题

# 常见的开发框架

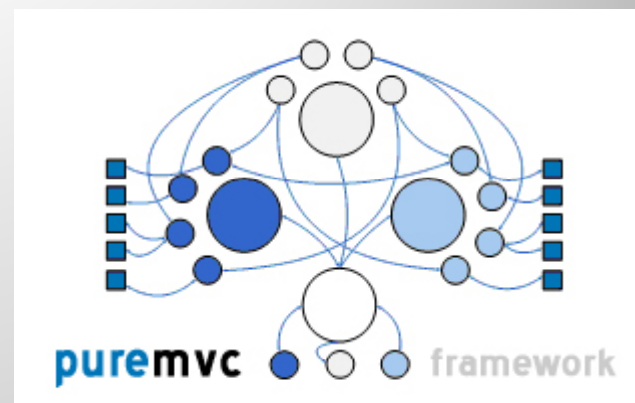
- SpringActionscript (Spring 官方)



- Spring官方提供的IoC容器
- 支持比较全面的IoC特性（相对其他IoC容器）
- 提供与Actionscript特性相关的一些特别解决方案（事件托管，远程数据加载）
- 强大的扩展性(自定义标签等)

# 常见的开发框架

- PureMVC



- MVC 结构化框架
- 支持组件重用度高
- 支持Flex Module

# 常见的开发框架

- Parsley



- Adobe 官方咨询团队推荐的IoC 容器
- Adobe Cairngorm3 的IoC容器实现
- 消息托管机制
- 视图动态绑定
- Flex Module支持
- 本地化支持
- 框架扩展

# Parsley背景

- Parsley 是 Spicefactory 旗下基于Actionscript3的开发结构框架
- SpiceFactory是大名鼎鼎的Powerflasher(FDT作者)的一个开源项目，其中包括服务端以及客户端的一些Flash/Flex/AIR/Java相关的一些项目
- Adobe咨询团队推荐的一个IoC实现

# Parsley特点

- IoC容器
- 消息框架(消息托管)

# Parsley—IoC容器-配置兼容多种形式

- 可以在Flex中直接使用MXML进行配置
- 可以使用XML进行配置
- 可以使用Actionscript代码进行配置
  
- 多种配置方式可以在最后整合到一起。没有任何区别和限制。

# Parsley—IoC容器 Metadata配置

Metadata

**[Inject]**

```
public var service:LoginService
```

这个标签在IoC容器中可以实现与Java版Spring的Autowire类似的功能。

**[Inject(id="mainLoginService")]**

也可以指定id来注入

# Parsley—IoC容器 MXML配置

```
<mx:Object
  xmlns:mx="http://www.adobe.com/2006/mxml"
  xmlns="http://www.spicefactory.org/parsley">

  <Object id="userManager" type="{UserManager}"/>
  <Object id="loginService" type="{LoginServiceImpl}">

    <Object type="{LoginAction}">
      <ConstructorArgs>
        <ObjectRef idRef="userManager"/>
      </ConstructorArgs>
      <Property name="service" idRef="loginService"/>
    </Object>
```

# Parsley—IoC容器 XML配置

```
<objects xmlns="http://www.spicefactory.org/parsley">  
  <object id="loginService"  
    type="com.bookstore.services.LoginServiceImpl">  
    <property name="timeout" value="3000"/>  
  </object>  
  
  <object id="userManager"  
    type="com.bookstore.services.UserManager"/>  
  
  <object type="com.bookstore.actions.LoginAction">  
    <constructor-args>  
      <object-ref id-ref="userManager"/>  
    </constructor-args>  
    <property name="service" id-ref="loginService"/>  
  </object>  
</objects>
```

# Parsley—事件消息机制

其他框架一般情况下处理全局事件方式：

单例全局事件播发( Cairngorm2)

```
CairngormEventDispatcher.getInstance().dispatchEvent( loginEvent );
```

或

```
loginEvent.dispatch();
```

( 前提：loginEvent是一个CairngormEvent的子类 )

# Parsley—事件消息机制

## Parsley的事件消息机制

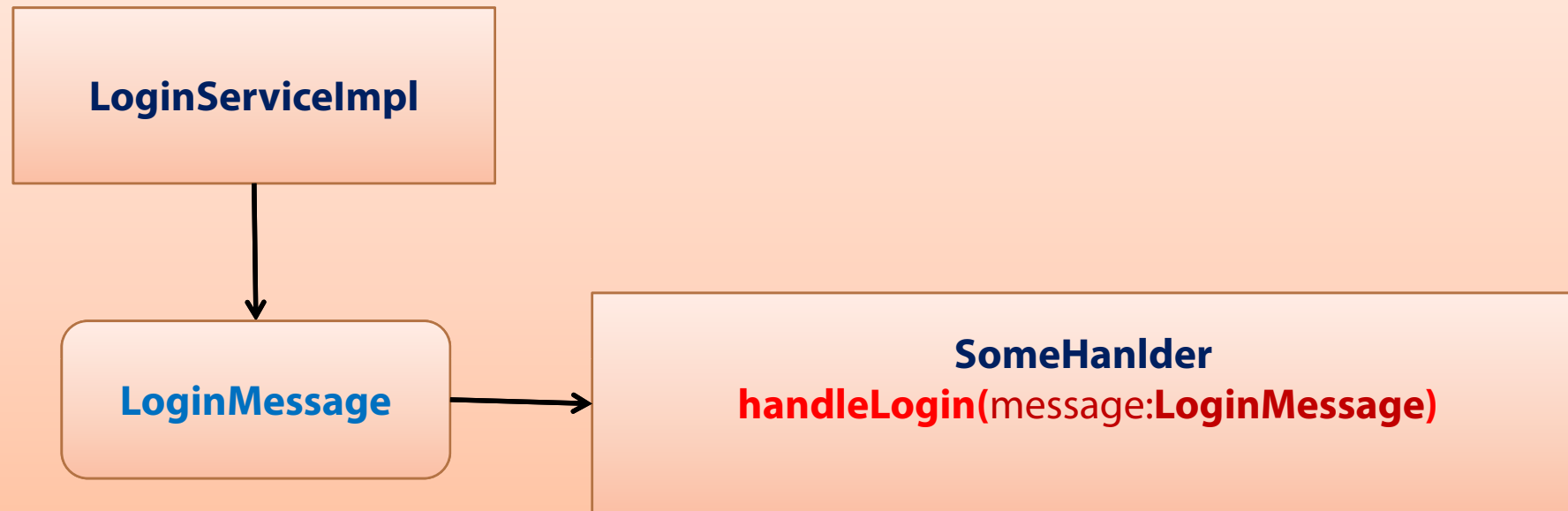
1. 组件不必依赖甚至不必“知道”框架的存在
2. 功能强大并且灵活

Parsley是一个消息托管容器

# Parsley—消息机制概览

消息派发 → 框架监听此消息 → 转给系统中注册的消息处理器

## Paseley 框架



# Parsley—事件消息机制

Parsley派发消息的三种方式：

1. 托管的事件
2. 注入消息派发器
3. 使用消息API

# Parsley—托管的事件(Metadata配置)

```
[Event(name="loginSuccess",type="com.bookstore.events.LoginEvent")]
```

```
[Event(name="loginFailed",type="com.bookstore.events.LoginEvent")]
```

```
[Event(name="stateChange",type="flash.events.Event")]
```

```
[ManagedEvents("loginSuccess,loginFailure")]
```

```
public class LoginServiceImpl extends EventDispatcher implements  
LoginService {
```

```
[...]
```

```
private function handleLoginResult (user:User) : void
```

```
{
```

```
dispatchEvent(new LoginEvent("loginSuccess", user)); }
```

```
}
```

# Parsley—托管的事件（MXML以及XML）

```
<Object type="{LoginServiceImpl}">
```

```
  <ManagedEvents
```

```
    names="['loginSuccess','loginFailure']"/>
```

```
</Object>
```

---

```
<object
```

```
  type="com.bookstore.services.LoginServiceImpl">
```

```
  <managed-events
```

```
    names="loginSuccess,loginFailure"/>
```

```
</object>
```

# Parsley—注入消息派发器

某些情况下：

- 不希望继承EventDispatcher
- 类已经继承自某个其他类，无法再继承EventDispatcher

可以通过注入消息派发器来完成这个工作

消息派发器是一个方法(function)，这个方法可以被注入到一个类中。

# Parsley—注入消息派发器

```
public class LoginServiceImpl implements LoginService
{
    [MessageDispatcher]
    public var dispatcher:Function;
    [...]
    private function handleLoginResult (user:User) :
    void{
        dispatcher(new
LoginMessage(user));
    }
}
```

# Parsley—注入消息派发器(通过XML配置)

```
public class LoginServiceImpl implements LoginService {  
    [MessageDispatcher]  
    public var dispatcher:Function;  
    [...]  
    private function handleLoginResult (user:User) : void{  
        dispatcher(new LoginMessage(user));  
    }  
}
```

# Parsley—注入消息派发器(通过XML配置)

```
<object type="com.bookstore.services.LoginServiceImpl">  
  <message-dispatcher property="dispatcher"/>  
</object>
```

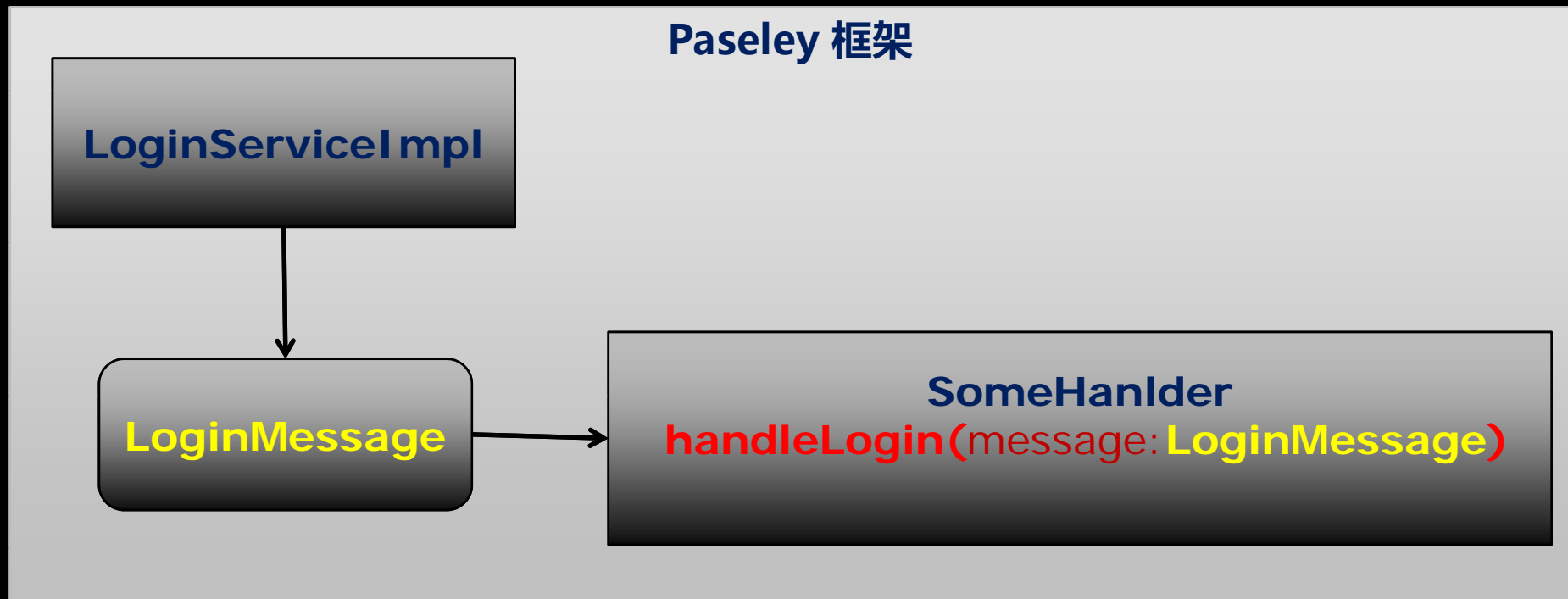
# Parsley—消息处理器(Message handler)

接收并且处理消息

Parsley可以自动根据**消息类型**匹配消息接收器：

**[MessageHandler]**

```
public function handleLogin (message:LoginMessage) : void {
```



# Parsley—消息处理器配置

## Metadata方式：

[MessageHandler]

```
public function handleLogin (message:LoginMessage) : void {
```

# Parsley—消息处理器配置

XML配置方式：

```
<object type="com.bookstore.actions.LoginAction">  
    <message-handler method="handleLogin" />  
</object>
```

# Parsley—消息绑定(Message Bindings)

消息绑定与Flex的消息绑定有一定区别：

- A. Flex的绑定仅仅能在Flex框架下使用，不可以在其他Actionscript项目中使用
- B. Flex绑定必须在源代码中设置并且指定，而且要在编译时应用。

# Parsley—消息绑定

## Metadata配置

```
[MessageBinding(messageProperty="user",type="com.bookstore.events.LoginMessage")]
```

```
public var user:User;
```

# Parsley—消息绑定

## XML配置

```
<object type="com.bookstore.services.LoginServiceImpl">  
  <message-binding  
    target-property="user"  
    message-property="user"  
    type="com.bookstore.events.LoginMessage" />  
</object>
```

# Parsley—消息其他功能

- 消息拦截
- 错误处理
- 异步消息

.....

详细功能可以见文档

<http://www.spicefactory.org/parsley/docs/2.2/manual/messaging.php>

# 更多相关内容可以参考RIAMeeting

RIA视频教程下载和播放工具(Beta 1.35)

视频列表 下载任务管理(2) 播放器 设置 帮助

Flash - EmployeeDirectory/src/EmployeeDirectory.mxml - Flash Builder

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
3   xmlns:s="library://ns.adobe.com/flex/spark"
4   xmlns:mx="library://ns.adobe.com/flex/halo" minWidth="1024" minHeight="768">
5   <fx:Declarations>
6     <!-- Place non-visual elements (e.g., services, value objects) here -->
7   </fx:Declarations>
8 </s:Application>
9
```

视图最大化。

- 在此文件中代码的第一行是XML声明，定义此文件为一个XML文件。
- 请记住MXML是基于XML的，在Flex框架中它主要用于可视化布局使用的声明性语言。
- XML声明必须是在文件中代码的第一行，不能包含在任何字符之前，甚至是空白。
- 请注意，如果我在此代码前添加一个空格然后保存文件，编辑器选项卡会显示一个红色的X图标，会显示在一行代码中有错误。**
- 要查看错误的详细信息，我双击编辑器标签就能看到问题视图中存在的错误信息。
- 我删除了额外的空格（白）并保存该文件。
- 在编辑器选项卡上的错误指示消失了。在MXML文件中的下一个元素是一个Application的MXML标签块，这是所有的MXML应用程序所必需的标签。

01:17 07:01

参与对这个视频的评论

RIAMeeting翻译了一整套来自Adobe的Flex4教学视频，请访问[www.riameeting.com](http://www.riameeting.com)使用RIAMediaPlayer下载工具和视频播放器，下载这些视频到本地观看



