

What's new in Flex 4.

What Developers (not  
marketing folk) really  
need to know.

Duane Nickull  
Sr. Technology Evangelist



Adobe

*Duanes World TV.com*



***"Wer nicht fragt, bleibt dumm!"***

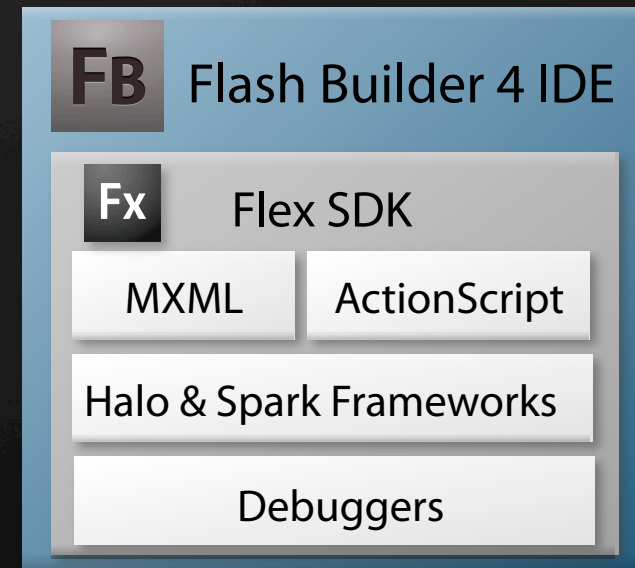


# ADOBE FLEX<sup>®</sup> 4

A highly productive, **free open source framework** for building expressive web applications that deploy consistently on all major browsers and on the desktop with Adobe AIR

# Understanding AIR, Flex and Flash Builder

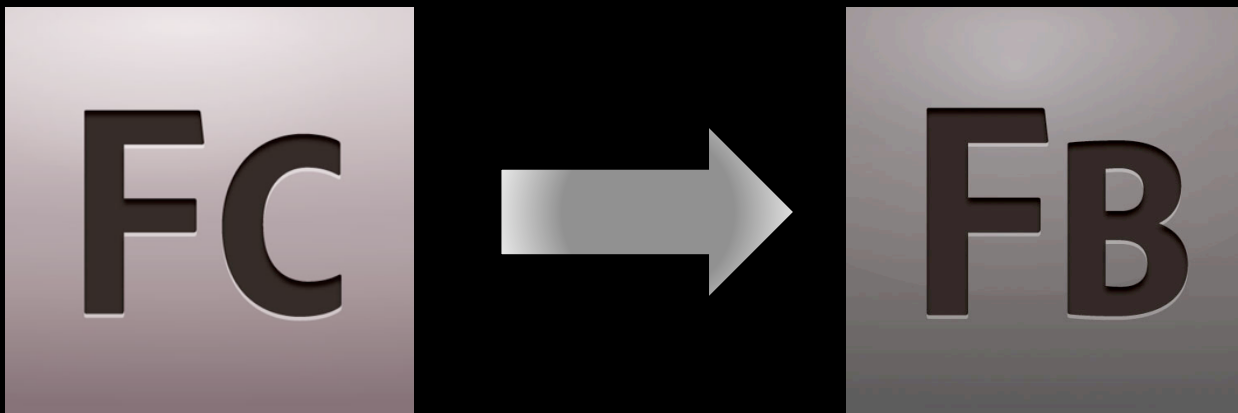
- 2 languages
  - MXML (actually a library of ActionScript)
  - ActionScript 3
- Multiple Components
  - Halo (<mx:Button>)
  - Spark (<s:Button>)
  - Fxg (<fxg:component>)
- Compilers
- Debuggers
- Rich Component Library



- Flash Builder IDE
  - Eclipse Plugin or turn-key install
  - Accelerates Design
  - Design view and code view

## Flex 4 Language changes

- Halo - `xmlns:mx="library://ns.adobe.com/flex/halo"`
  - Included in previous releases of Flex
- Spark - `xmlns:s="library://ns.adobe.com/flex/spark">`
  - New architecture for skinning and have other advantages/components
- Flash XML Graphics – for workflow hand off between Flash Catalyst and Flash Builder 4
  - `xmlns:fx=http://ns.adobe.com/mxml/2009`
  - Support for Flash Catalyst (Thermo) Exports



# Adobe® Flash™ Builder™

- Next evolution of “Flex Builder 3”
  - Uses the Flex Framework v.4 (Halo & Spark)



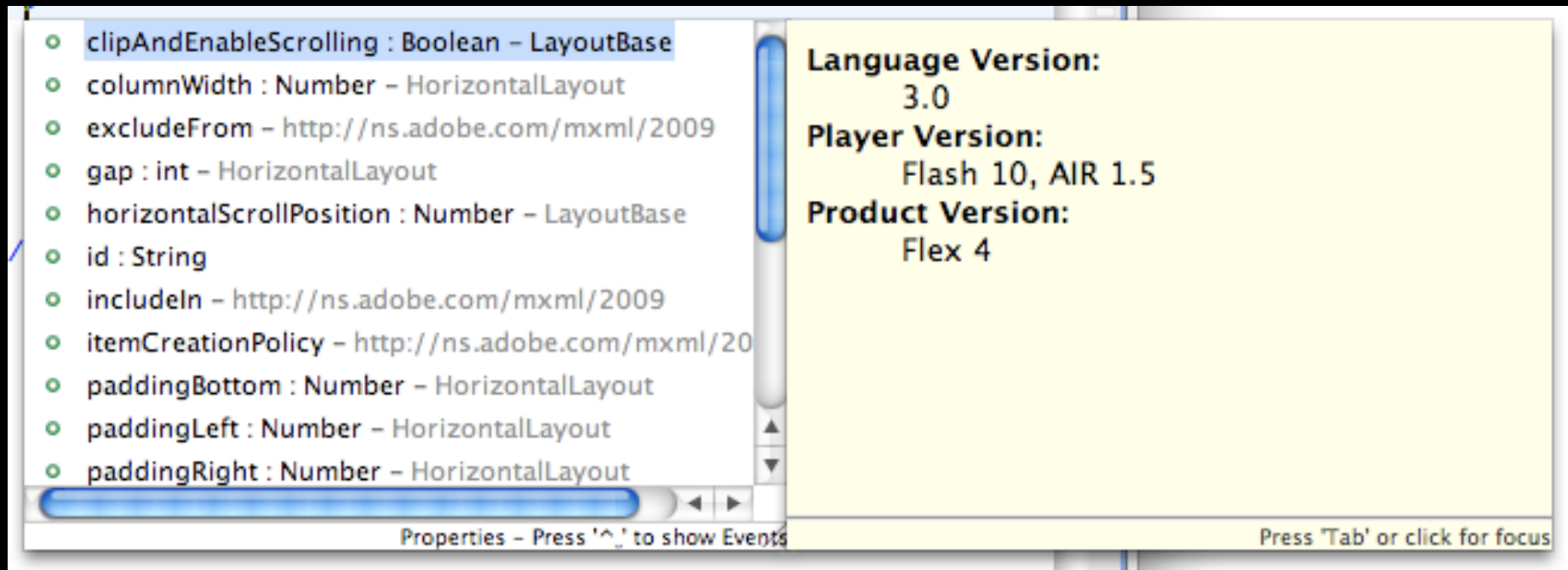
```
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
  xmlns:mx="library://ns.adobe.com/flex/halo"
  xmlns:s="library://ns.adobe.com/flex/spark">
  <s:layout>
    <s:VerticalLayout/>
  </s:layout>

  <s:Panel title="My Application">
    <s:SimpleText text="Hello World"
      fontWeight="bold" fontSize="24"/>
  </s:Panel>
</s:Application>
```



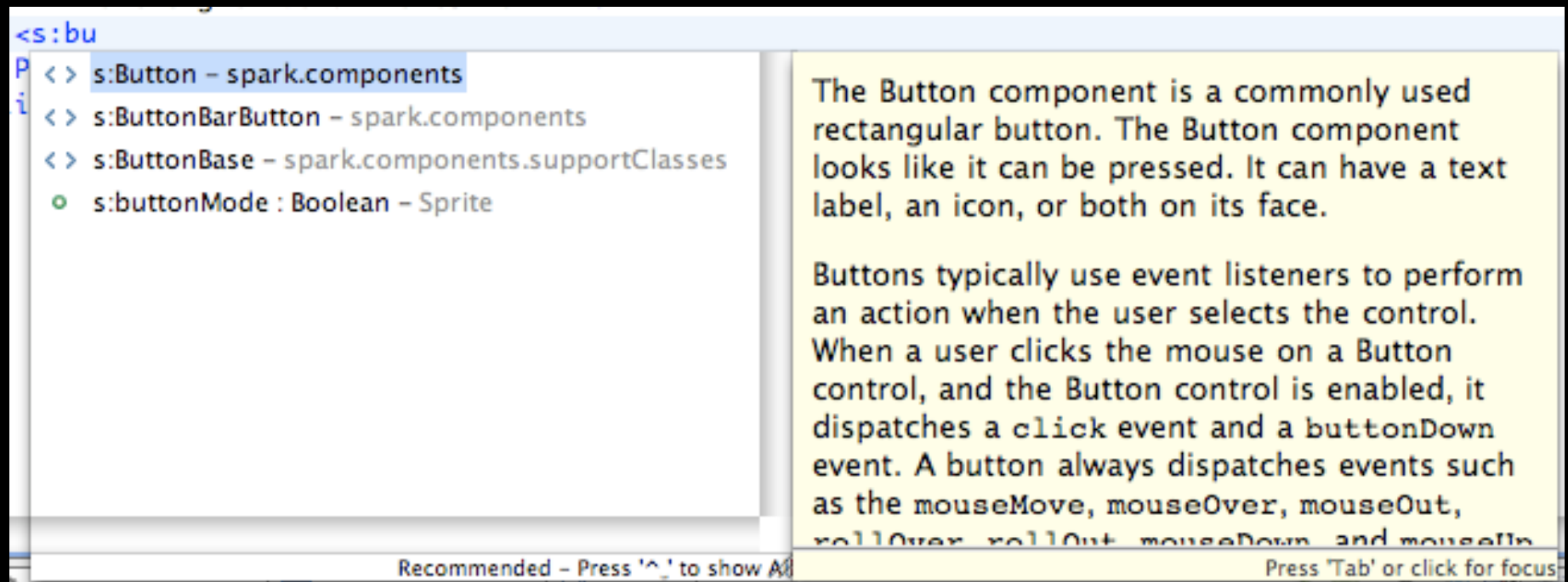
# Adobe® Flash™ Builder™ - new Features

- **Advanced Code Hinting**
- Content assist in Class Wizard, MXML Component Wizard
- Open Type filters Recommended types



# Adobe® Flash™ Builder™ - new Features

- Advanced Code Hinting
- Content assist in Class Wizard, MXML Component Wizard
- Open Type filters Recommended types






# Adobe® Flash™ Builder™ - new Features

```
<s:layout>
    <s:HorizontalLayout paddingTop="50" />
    <s:VerticalLayout />
</s:layout>

<s:Panel title="
    <s:SimpleText
        fontWeig
</s:Panel>
Application>
```

 spark.components.SkinnableContainer.layout(value:LayoutBase):void

The layout object for this container. This object is responsible for the measurement and layout of the visual elements in the container.

The layout object for this container. This object is responsible for the measurement and layout of the visual elements in the container.

← →

## Styles and Themes

- You modify Flex components via style properties.
- Some inherited by children from their parent containers, and across style types and classes.
- Define style once => apply to set of controls or single type.
- Override properties for each control at a local, component, or global level.
- Style property mutation depends on namespace:
  - components in the Halo packages (mx.controls.\*, mx.containers.\*) take one set of styles.
  - Components in the Spark packages (spark.components.\*, spark.containers.\*) allow different set of styles.
- Properties such as x, y, width, and height are properties, not styles, of the UIComponent class, and therefore cannot be set in CSS

# Styles Code Sample

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
  xmlns:mx="library://ns.adobe.com/flex/halo" xmlns:s="library://
ns.adobe.com/flex/spark">
  <fx:Style>
    @namespace mx "library://ns.adobe.com/flex/halo";

    .myFontStyle {
      fontSize: 15;
      color: #9933FF;
    }

    mx|Button {
      fontStyle: italic;
    }
  </fx:Style>
  <!-- ALTERNATIVE: fx:Style source="../assets/SimpleTypeSelector.css"/-->
  <mx:Button id="myButton" styleName="myFontStyle" label="Style Me"/>

</mx:Application>
```

Click Me

## Global Styles and Themes

- The `style_name` can be the literal global, a type selector (example: `_Button`) or a class selector that you define in either the `<fx:Style>` tag or an external style sheet.
- Global styles apply to every object that does not explicitly override them

# Using the StyleManager class for over-riding values

```
<fx:Style>
    .myStyle{
        color: red;
    }
</fx:Style>
<fx:Script><![CDATA[
    import mx.styles.StyleManager;

    public function initApp(e:Event):void {
        /* Type selector; applies to all Buttons and subclasses of Button. */
        StyleManager.getStyleDeclaration("mx.controls.Button").setStyle("fontSize",24);
        /* Class selector; applies to controls using the style
           named myStyle. Note that class selectors must be prefixed
           with a period. */
        StyleManager.getStyleDeclaration(".myStyle").setStyle("color",0xCC66CC);

        /* Global style: applies to all controls. */
        StyleManager.getStyleDeclaration("global").setStyle("fontStyle","italic");
    }
]]></fx:Script>

<mx:Button id="myButton" label="Click Me" styleName="myStyle"/>

<mx:Label id="myLabel" text="This is a label." styleName="myStyle"/>
```

Click Me

This is a label.

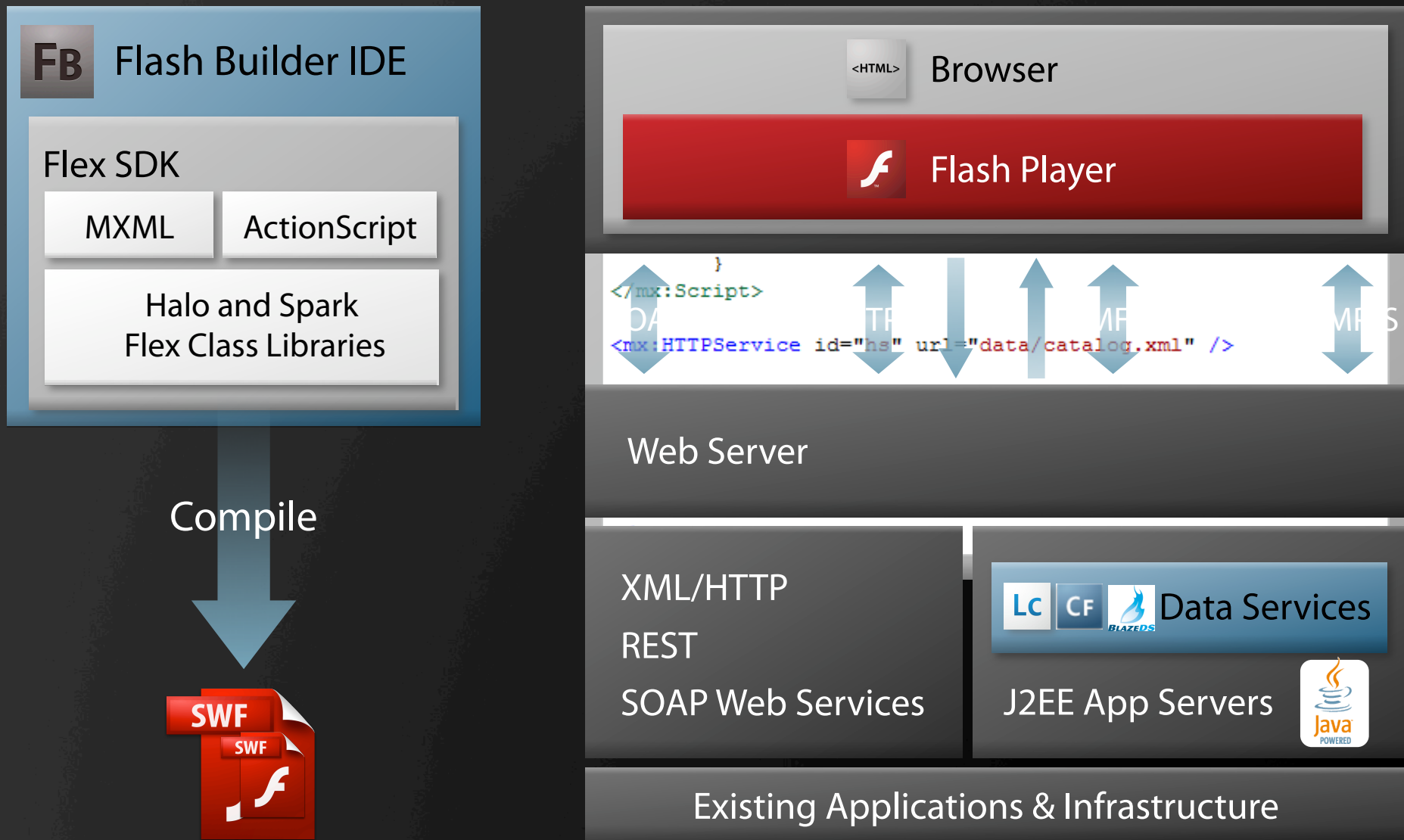
# Skinning

- Skinning is the process of changing the appearance of a component by modifying or replacing its visual elements.
- These elements can be made up of bitmap images, SWF files, or class files that contain drawing methods that define vector images.
- Skins can define the entire appearance, or only a part of the appearance, of a component in various states.
  - Example, a Button control has eight possible states, and eight associated skin properties:

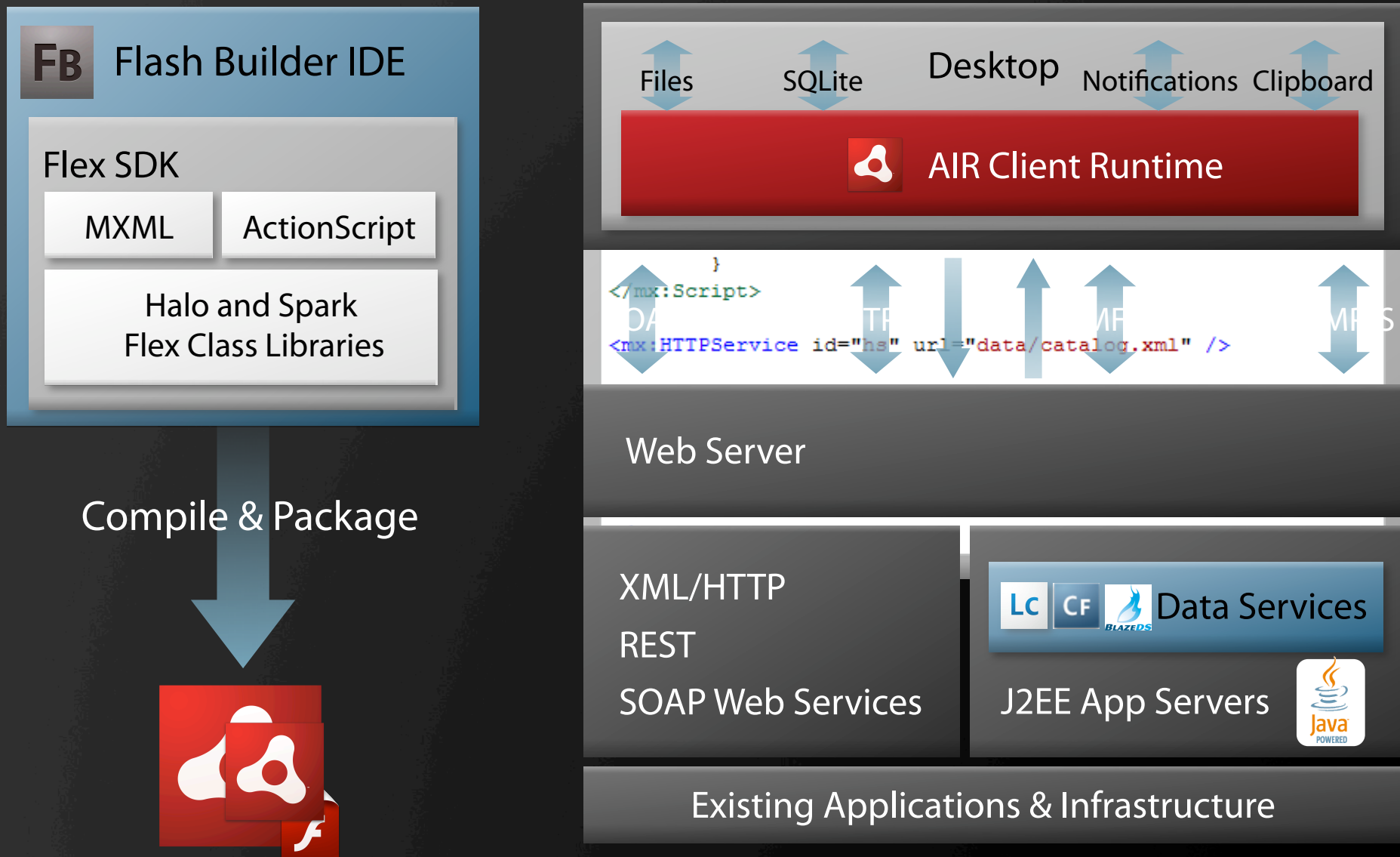


State	Skin Property	Default Skin Classes
Down	downSkin	mx.skins.halo.ButtonSkin
Over	overSkin	mx.skins.halo.ButtonSkin
Up	upSkin	mx.skins.halo.ButtonSkin
Disabled	disabledSkin	mx.skins.halo.ButtonSkin
selectedDisabled	selectedDisabledSkin	mx.skins.halo.ButtonSkin
selectedDown	selectedDownSkin	mx.skins.halo.ButtonSkin
SelectedOver	selectedOverSkin	mx.skins.halo.ButtonSkin
SelectedUp	SelectedUpSkin	mx.skins.halo.ButtonSkin

# How Flex Works in the Browser

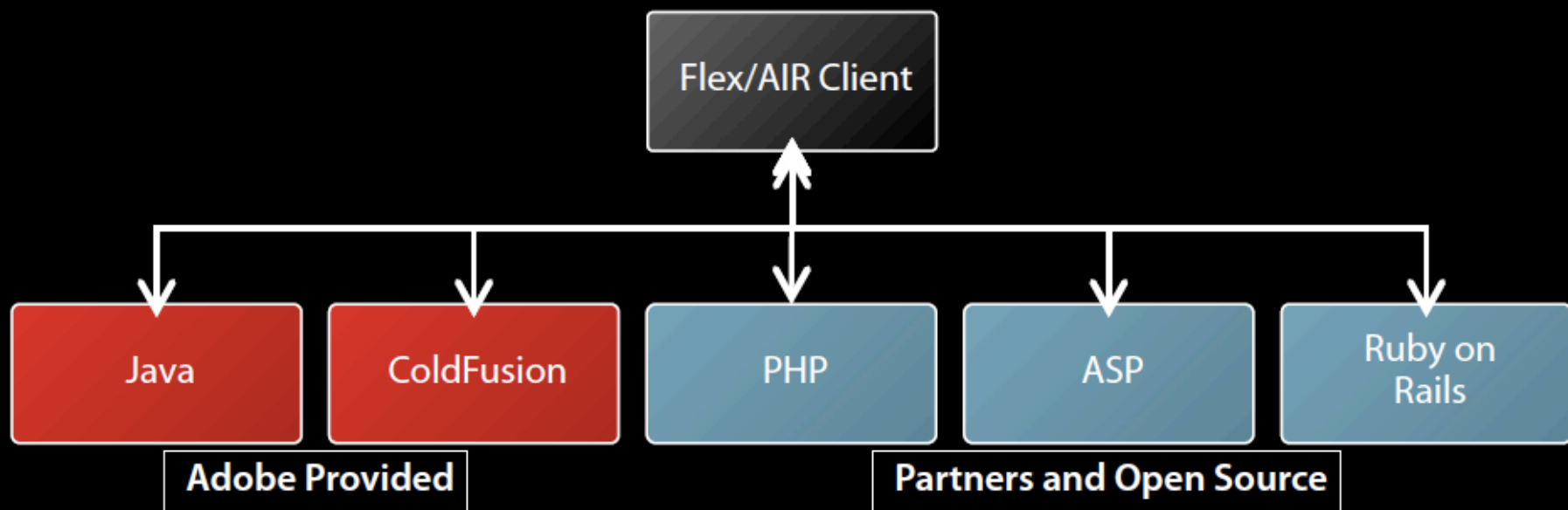


# How Flex Works on the Desktop (Adobe Integrated Runtime)



# Flex & Flash Community – a REAL Developer Community

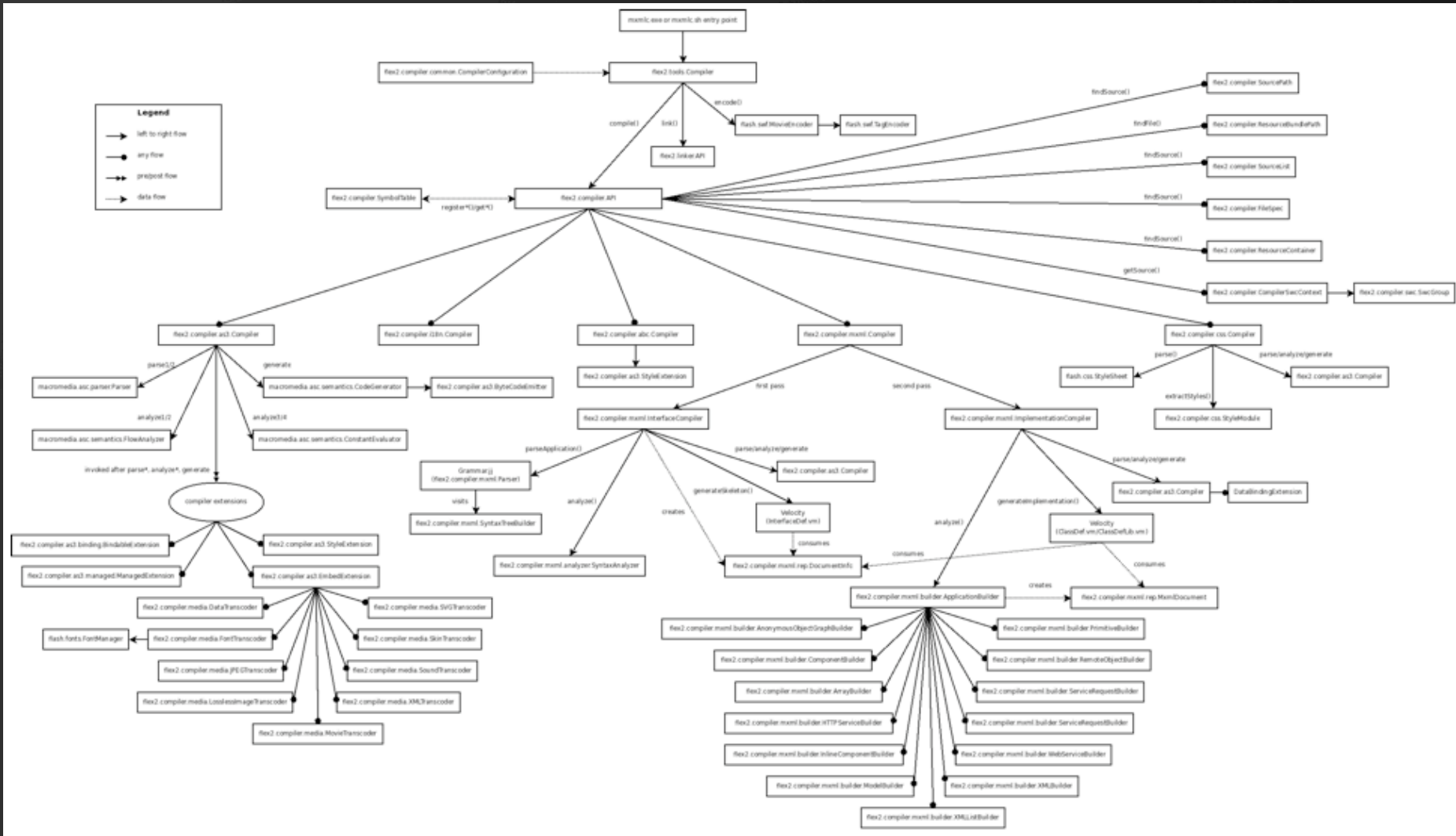
- Foster multiple projects/products supporting Flex/Flash remoting and messaging
- Consistent set of core features across different server implementations
- Common developer experience and programming model across different server technologies



# FLEX 4 SDK<sup>®</sup> COMPILERS

The logo consists of the letters 'Fx' in a bold, white, sans-serif font, centered within a dark gray square. The square has a subtle radial gradient, being slightly lighter in the center and darker towards the edges.

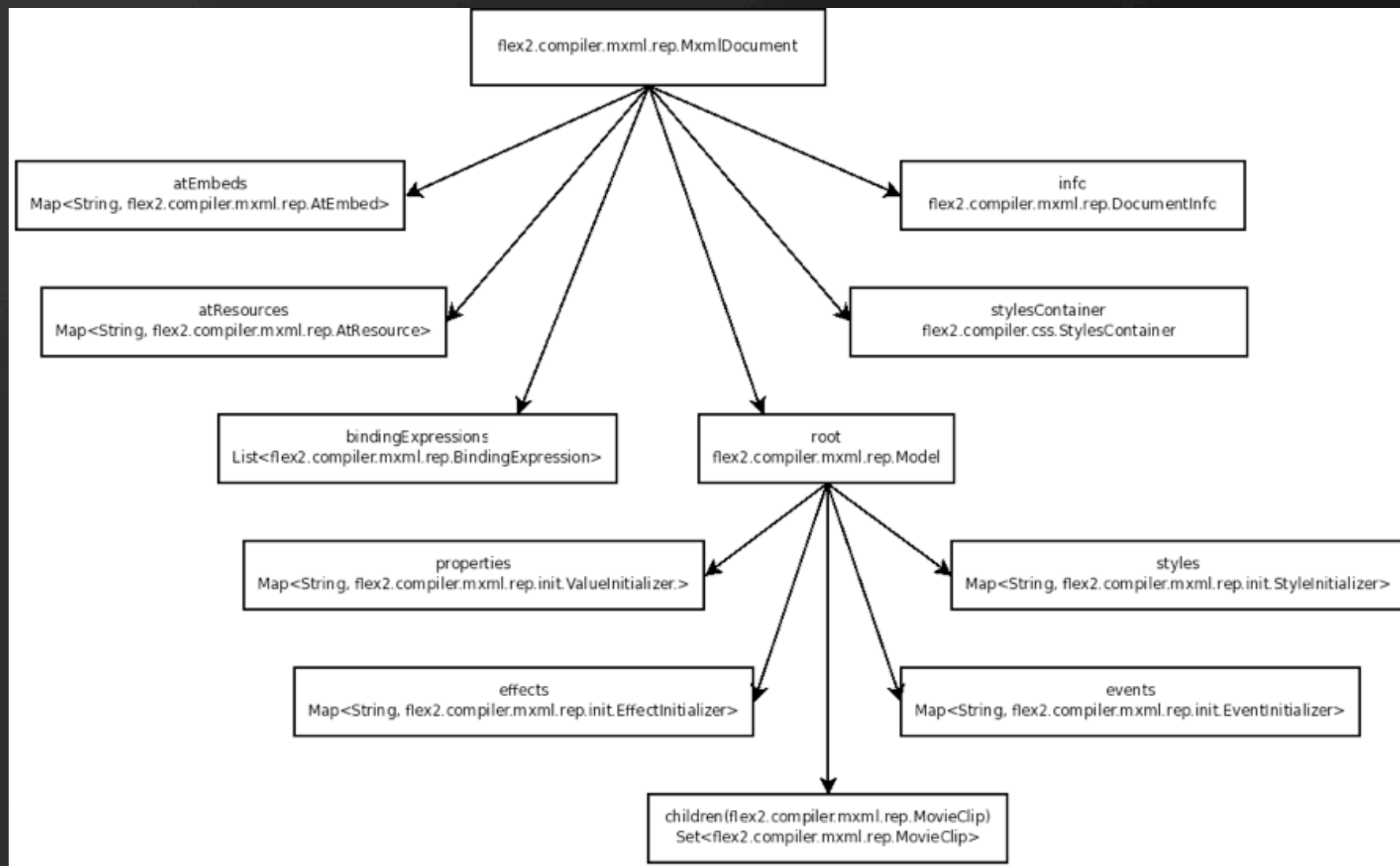
# Compilers



<http://opensource.adobe.com/wiki/download/attachments/12845394/compiler.png?version=1>



# MXML Document Container



# THE MEAT:



Why we are here!





Revolutionizing  
how the world  
engages with ideas  
and information



Adobe