Flex vs. HTML 5 for RIAs

Pamela Fox
@pamelafox
http://imagine-it.org

Ignite Sydney 2009
The web is now strewn with rich web apps, and honestly, they’re kind of a bitch to make. In the next 5 minutes, I’ll describe 2 emerging technologies that will make web app development faster.
RIA stands for “Rich Internet Application.” It’s basically a web app that has all the functionality a user expects of a desktop app – like Google Spreadsheets compared to MS Excel. RIAs are pretty much a hack right now, but with new technologies, they’re becoming less so.
HTML5 is the next evolution of HTML, and will be backward compatible with existing HTML. As the spec is being worked on, parts of it being gradually adopted by the various browsers. The whole spec likely won’t be officially done for another decade.

```html
<!DOCTYPE HTML>
<html>
  <head><title>HTML 5</title></head>
  <body>Hello World!</body>
</html>
```
Flex is basically Flash for application designers - so it combines the power of Flash with things like UI widgets. AIR is a framework that works with Flex to let you make downloadable desktop applications.

<?xml version="1.0" encoding="utf-8"?>

<mx:Application
xmlns:mx="http://www.adobe.com/2006/mxml">

<mx:Panel title="My Application">

<mx:Label text="Hello World!"/>

</mx:Panel>

</mx:Application>
Adobe has open-sourced the Flex SDK/Spec/Compiler, but not the flash player runtime. The WHATWG group is working on the open HTML5 spec, but most browser runtimes are closed.
UI Widgets

Flex prides itself on providing many different UI widgets (containers, lists, menus, navigators, etc). HTML5 shies away from this and other presentational tags, just adding datagrid and menu. And as indicated by that question mark, there are no implementations of these tags yet, so we have no idea what they might look like.
Forms

HTML5 adds many new input types, Flex adds a couple nifty controls, and both of them offer new ways of validating user input without much effort. For example, you can check the validity of an email address (which would otherwise be a huge RegEx). You can also query a boolean to see if a form is valid, and in HTML5, you can use pseudo classes for styling invalid form fields.

```
mx.controls.ColorPicker,
mx.controls.DateChooser
```

```
input type="list", ="range",
= "date"
```

```
mx.validators.PhoneNumberValidator
formIsValid
```

```
input type="email", ="url"
checkValidity
```
Vector Graphics

Vector graphics means drawing lines and shapes of different styles. Flash has always had graphics as its focus and been amazing at this, HTML has petered around with various incompatible specs like SVG and VML. Finally, HTML5 introduces the canvas element which gives us fast vector graphics.

flash.display.Graphics
<canvas></canvas>

Flex

HTML5
3D Graphics

3D Graphics means creating scenes, setting up cameras/lighting, and doing 3d transformations. Though Flash only recently offered actual native 3d support in Player 10, there are many open-source flash 3d projects that simulate 3d just fine. The HTML5 spec suggests that there may be a 3d context for the canvas object, and a few browsers are experimenting with what that might look like.

```
flash.geom.Matrix3D

PaperVision3D, ...

canvas.getContext('3d')
```

Flex

HTML5
Once you have graphics, you want to manipulate them. Flex lets you change the individual pixels, or easily apply different filters, effects, and transforms. Canvas only gives you raw pixel access, but with that, anything is possible... just harder.
Video

Flash has always been the obvious choice for displaying videos on the web -- Youtube is the big example of that - and Flex makes it even easier with more controls. HTML finally introduces native browser support for videos via a video tag which will most likely support the OGG video format.
History

There are two types of history a user expects in an app. The first is navigational history - i.e. browser back button - which both Flex and HTML5 support. The second is undo history, such as in a drawing app, and only HTML5 proposes native support for that.
Persistent Connections

It’s always been possible to do this by connecting to sockets in Flash, which is why we see so many games and collaborative apps in Flash. There are various hacks for how to do this in HTML, such as COMET, but HTML5 makes it infinitely easier by providing native support for web sockets.
Drag + Drop

It’s not just dragging and dropping inside an application, but also dragging, dropping and copying, pasting from the desktop or across different applications. All of this is made possible in HTML5, and in Flex AIR applications.

`flash.desktop.Clipboard`

`flash.events.NativeDragEvent`,

`flash.desktop.TransferableData`

Flex (AIR)

Good

SPAM

DataTransfer,

DragEvent

HTML5
Users expect their apps to access stuff on the file system. Flex AIR apps can both read and write to the file system, while HTML5 just maintains the upload input type. HTML5 does however have a localStorage API which can accomplish similar things.
Offline Access

This means the app must cache its resources and data, and as a bonus, know whether its offline or online. Flex AIR apps and HTML5 websites have the ability to do all of this. There’s no implementation of it for HTML5 yet, but you can use Gears for a similar experience.

**Resources**

<table>
<thead>
<tr>
<th>Flex (AIR)</th>
<th>HTML5</th>
</tr>
</thead>
<tbody>
<tr>
<td>WindowedApplication</td>
<td>(&lt;html manifest=&quot;http...&quot;&gt;) ApplicationCache</td>
</tr>
</tbody>
</table>

**State**

<table>
<thead>
<tr>
<th>Flex (AIR)</th>
<th>HTML5</th>
</tr>
</thead>
<tbody>
<tr>
<td>air.Event.NETWORK_CHANGE URLMonitor</td>
<td>Navigator.onLine</td>
</tr>
</tbody>
</table>

**Data**

<table>
<thead>
<tr>
<th>Flex (AIR)</th>
<th>HTML5</th>
</tr>
</thead>
<tbody>
<tr>
<td>flash.data.SQLConnection</td>
<td>Database, SQLTransaction, SQLResultSet localStorage</td>
</tr>
</tbody>
</table>
Development

It’s always easier to build apps with an IDE. For Flex, there’s FlexBuilder, built off the Eclipse platform and just a few hundred dollars. For HTML, there are many options but the most popular is DreamWeaver (also from Adobe!).
Unit testing is easy in both Flex and HTML5 via ASUnit and JSUnit. Integration (or end-to-end) testing is a bit trickier. There are popular open-source options for HTML5, and nothing really established for Flex yet.
Compatibility

Flex developers need to worry about the version of Flash Player installed, while HTML developers need to worry about all the different browsers and different versions of browsers. In the HTML5 Spec, there’s a legend next to each section indicating browser support.

And, of course, everyone has to worry about mobile support. Or lack thereof.
When you’re deciding what language to write your web app in, the big advantage of Flex is that it actually exists. The HTML5 spec is exactly that - just a spec- so the best you can hope for is gradual adoption over the next decade. So it’s up to you whether you want a quick fix or a long-term solution.