

# Why Use Flex for Web Development

*With the recent emergence of the HTML5 open source platform, there has been much debate over the future of Flex and whether or not it will maintain its stronghold as the web standard. While the choice of RIA ultimately depends on your intended users and expectations, with Flex you are able to develop a robust application in much less time than you would with HTML5.*

*This is due in large part to Flex's optional statically typed language, robust development environment and effective debugger. Combine that with a powerful, yet easy to use programming model for manipulating graphical elements and what you have is an environment in which software can be prototyped and developed quickly, problems can be debugged easily, development is almost trivial and portability across browsers is practically universal.*

## Overview

This white paper is intended to help the enterprise architect select a technology for quickly and efficiently implementing client UI's for enterprise applications that reach multiple applications with one set of code. The two alternatives being compared in this document are Adobe Flex and HTML. More specifically, we will look at HTML pages generated on the server on demand, and explore the features of HTML5 that are currently being compared to those of Flash.

## Flex

<b>Flex [1] is:</b>	<b>Specifically</b>	<b>Implications</b>
a framework ...	An open source library of <a href="#">ActionScript [2]</a> code for UI elements and behaviors.	Provides developers with ready-made components, a basis for extensions and customizations, and powerful high-level functions.
an optional declarative language ...	<a href="#">MXML [3]</a> , an XML-based means to describe user interfaces, their states and other properties.	<ul style="list-style-type: none"> <li>• A simpler way to write certain types of user interfaces.</li> <li>• Works well with visual UI builders and other tools.</li> </ul>
for creating applications to run on the <a href="#">Flash Player [4]</a> or <a href="#">AIR [5]</a> ,	Compiles to a <a href="#">SWF [6]</a> file (or AIR application) that is loaded and executed on the client machine.	<ul style="list-style-type: none"> <li>• Development concept is much closer to creating desktop applications than web (page-oriented) applications (i.e. simpler, more familiar).</li> </ul>
preferably using an IDE and debugger.	<a href="#">Flash Builder [7]</a> , an Eclipse plugin, with a debugger that can step through ActionScript client code and Java server-side code.	<ul style="list-style-type: none"> <li>• A powerful environment familiar to a great many developers.</li> <li>• A level of runtime debugging insight that is difficult to achieve with other technologies.</li> <li>• Compiler type checking is much better in AS3 than in JavaScript.</li> </ul>

## HTML

HTML is:	Specifically	Implications
a content (text, image) mark-up language ...	Tags for presentation structure and style.	<ul style="list-style-type: none"> <li>Mixes content (data) and appearance.</li> <li>Browsers are inconsistent in interpretation</li> </ul>
with optional embedded JavaScript ...	Essential for interaction and dynamic behavior.	<ul style="list-style-type: none"> <li>Pure HTML pages load faster.</li> <li>JavaScript is very flexible, but harder to resolve problems and debug due to its highly dynamic abilities and lack of type information.</li> </ul>
for creating web pages ...		<ul style="list-style-type: none"> <li>Page fetching concept in place by default.</li> <li>With Ajax approaches, partial page changes are achieved using JavaScript.</li> </ul>
preferably using a web development environment ...	For example: <a href="#">Dreamweaver [8]</a> or MS Visual Studio.	
and web application framework ...	For example: Struts on the server side or an <a href="#">Ajax framework [9]</a> such as <a href="#">jQuery [12]</a> on the client side.	
often with pages generated dynamically on the server.	Using PHP, JSP or ASP.NET.	<ul style="list-style-type: none"> <li>Complexity of J2EE EJB mechanism and JSP or other page generation mechanism.</li> <li>Multiple round trips to server (less responsive, traffic bandwidth, server load).</li> </ul>

## HTML5 - the next revision of HTML

<a href="#">HTML5 [10]</a> is:	Specifically	Implications
currently in draft state ...	<a href="#">Some features are likely to change [11]</a> .	It will take some time for support to be widespread [11], whereas Flex applications can run basically everywhere except on iPhones and iPads (can run there if compiled to AIR).
to incorporate application-like features.	For example: drag and drop, video playback, canvas.	<ul style="list-style-type: none"> <li>Features added to markup concept are reasonable and fit well into existing HTML framework.</li> <li>Canvas feature is flexible and works well; higher level components (libraries) will take time to evolve.</li> <li>Language (JavaScript) doesn't scale as well for large or complex applications.</li> <li>JavaScript is difficult to debug dynamically; possible browser incompatibilities as HTML5 is slowly accepted and deployed.</li> </ul>

## Development Effort Comparison

Flex advantages	Implications	Bottom line	HTML contrast
A given number of lines of code can be developed faster in Flex	Requires less development effort (coding, reviewing, testing)	Reduces cost of development	Multiple files have to be created or modified per feature, and must be kept in synch
Using Flex requires less code		Reduces time to market	
	Higher quality (fewer defects)	Increased acceptance and better reputation	
Flex code is easier to understand	Requires less development effort to fix bugs or add enhancements	Reduces cost of maintenance	Not easy to read and “get” what the various parts of an HTML-based application are doing
	New programmers can easily pick up the work done by others	Reduces cost of staffing, ramp-up time	

## Reasons Behind Advantages

Why ...	Reasons
... using Flex requires less code:	<ul style="list-style-type: none"> <li>The framework effectively creates a higher level language, implying fewer lines of code required to accomplish sophisticated tasks.</li> <li>The Flash Player does all the heavy lifting dealing with the display; the application can achieve any desired visual effect with very simple commands.</li> <li>Server code is reduced because it does not need to generate pages or anything UI-specific.</li> </ul>
... Flex code can be written faster:	<ul style="list-style-type: none"> <li>ActionScript, MXML and Flex all allow expressing intent very directly, the way a developer would instinctively think of the solution.</li> <li>Sophisticated and mature deployment tools can be used:               <ol style="list-style-type: none"> <li>Flash Builder - providing combined client and server interactive debugging, and</li> <li>Flash Catalyst - providing tight integration between code and creative work on UI skins while supporting parallel code and creative work flows.</li> </ol> </li> </ul>
... Flex code is easier to understand:	<ul style="list-style-type: none"> <li>The ActionScript language looks familiar - it is similar conceptually to Java or C#. The typical Java or C# developer can get fully productive on Flex in a matter of days.</li> <li>There are fewer separate pieces that need to be understood and integrated/synchronized mentally.</li> </ul>

## Example of Reduction in Development Effort (JSP case)

The following numbers show what we achieved when we ported Ensemble's [TaskJournal \[13\]](#) to Flex from its previous incarnation as a JSP application (Task Workbench). Raw time sheet hours were 25k for the JSP version and 6k for the Flex version. We can discount 25% of the JSP hours as being due to figuring out requirements that would have been applicable to Flex if done in the opposite order.

	<b>Flex (TaskJournal)</b>	<b>HTML generated by JSP (Task Workbench)</b>
Developer hours	6,000 (30%)	19,000
Number of files	385 (60%)	665
Size of files (bytes)	1,423,036 (40%)	3,596,493
Size of files (lines)	44,185 (40%)	112,651
Semicolons	16,361 (50%)	33,359

Note that the developer effort reduction is greater than the reduction of any of the size metrics, underscoring the claim that Flex is easier to write. The "number of files" metric is valid because the work done by the same architect, meaning the same level of granularity and structuring style, was applied. The reduction in number of files underscores the claim that Flex is simpler, since files correspond to "things to think about."

It is also important to consider why we migrated this application from JSP-based HTML to Flex: it was becoming impossible to add features without breaking things. We became totally averse to improving the application, even when faced with urgent business needs. Now we add new features continuously, and have fun doing so.

Caveat 1: The effort savings would be affected significantly if the HTML version had been implemented using a framework such as jQuery. There would still be a difference though, mainly because of the structural simplification and superior debugging of Flex.

Caveat 2: Development effort depends much more on the skill of team members than it does on tools or technology. Our case study is valid because the teams were of equal skill level. If you have a team that is very experienced in developing HTML-based applications using a modern JavaScript framework, the effort differentiator will not be as significant.

## Conclusion

When it comes to selecting a technology for implementing client UI's for enterprise applications, Flex's reduced development effort and time to market makes it a clear frontrunner over any "traditional" approach using HTML (such as JSP's). Even if the cost and time factors are discounted by an experienced development team and/or modern frameworks and tools, the slick user experience and ubiquitous distribution of the Flash platform adds enough weight to the argument for using Flex for web development.

## References

- [1] Flex <http://www.adobe.com/products/flex/>
- [2] ActionScript <http://en.wikipedia.org/wiki/ActionScript>
- [3] MXML <http://en.wikipedia.org/wiki/MXML>
- [4] Flash Player [http://en.wikipedia.org/wiki/Flash\\_player](http://en.wikipedia.org/wiki/Flash_player)
- [5] AIR [http://en.wikipedia.org/wiki/Adobe\\_Integrated\\_Runtime](http://en.wikipedia.org/wiki/Adobe_Integrated_Runtime)
- [6] SWF <http://en.wikipedia.org/wiki/Swf>
- [7] Flash Builder [http://en.wikipedia.org/wiki/Flash\\_Builder](http://en.wikipedia.org/wiki/Flash_Builder)
- [8] Dreamweaver <http://en.wikipedia.org/wiki/Dreamweaver>
- [9] Ajax Frameworks [http://en.wikipedia.org/wiki/List\\_of\\_Ajax\\_frameworks](http://en.wikipedia.org/wiki/List_of_Ajax_frameworks)
- [10] HTML5 <http://en.wikipedia.org/wiki/HTML5>
- [11] HTML5 compatibility table <http://caniuse.com/#cats=HTML5&statuses=rec,pr,cr,wd,ietf>
- [12] jQuery <http://en.wikipedia.org/wiki/JQuery>
- [13] Task Journal <http://www.ensemble.com/products/taskjournal.shtml>

### About Ensemble

Ensemble is an Adobe enterprise solution partner with proven expertise in LiveCycle, Flex, AIR and Acrobat. Our industry-specific solutions include Government, Financial Services, Media and Entertainment, Manufacturing, and several others. Based in Vancouver, Canada and London, UK we field our experts anywhere in the world. To learn more about Ensemble, its products and services, visit [ensemble.com](http://ensemble.com).

**Web:** [ensemble.com](http://ensemble.com)

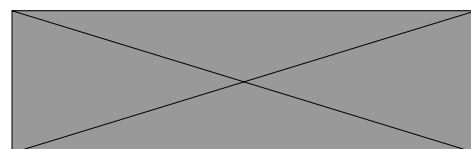
**E-mail:** [services@ensemble.com](mailto:services@ensemble.com)

#### North American Office

Unit 2268 - 13353 Commerce Parkway  
Richmond, B.C. Canada V6V 3A1  
**Tel:** + 1 604 231 9510

#### European Office

131 - 151 Great Titchfield Street  
London, UK W1W 5BB  
**Tel:** + 44 (0) 20 3178 6478



# ensemble