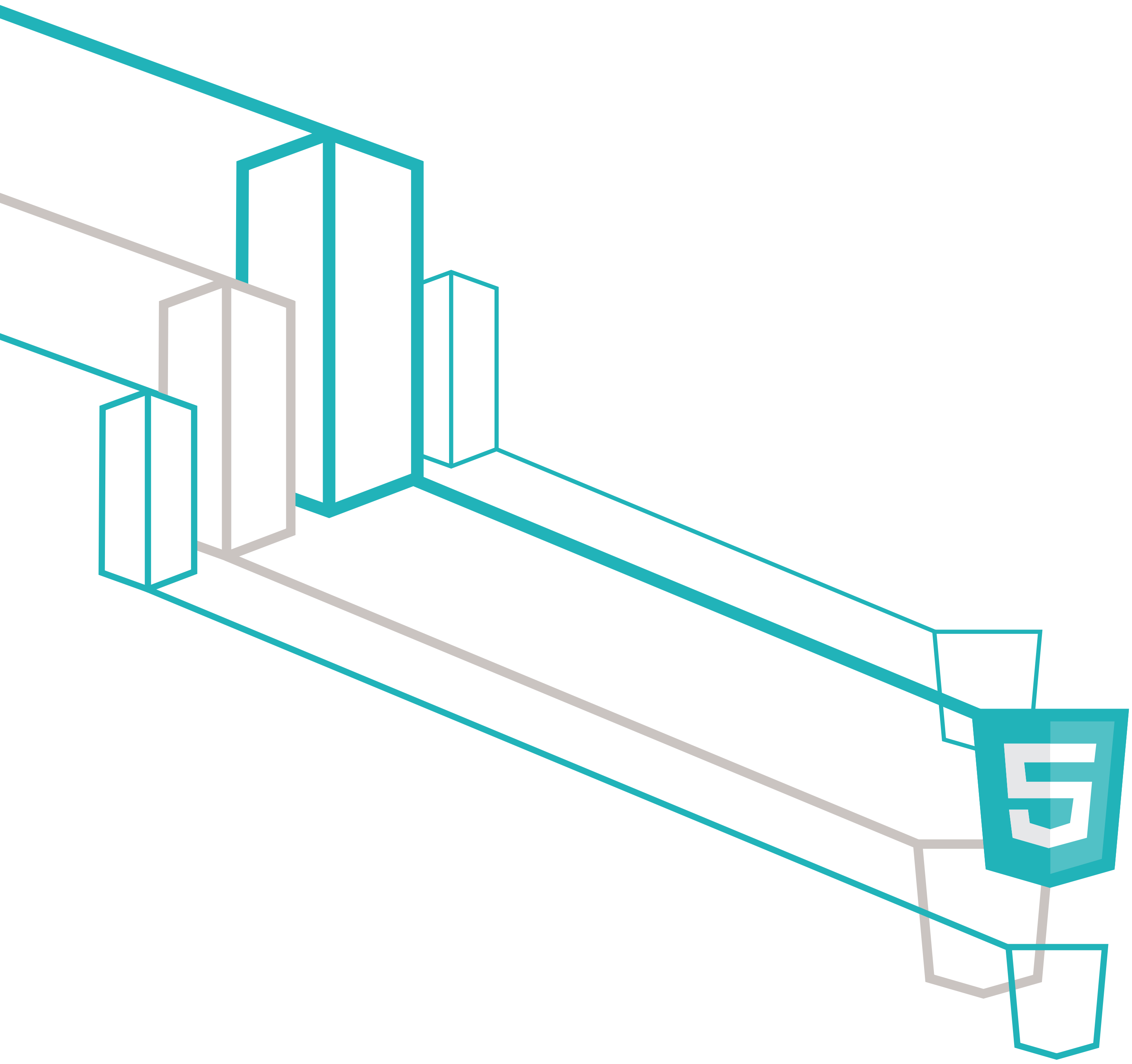
HTML5 AT ENTERPRISE SCALE

/ A WHITE PAPER BY COLIN EBERHARDT



SCOTT LOGIC

ALTOGETHER SMARTER

EXECUTIVE SUMMARY

Moving from desktop or plugin technologies (Flex, Silverlight, Java Applets) to HTML5 is a challenge for developers of large-scale enterprise applications. The technology can be deceptive, with high-fidelity HTML5 mockups providing great promise early in a project, only to be followed by spiralling development costs, bugs and unhappy users.

We have extensive experience of enterprise-scale HTML5 development, having worked on data-intensive websites, trading applications, analysis tools and content management systems. This white paper highlights the myriad challenges that face enterprise-scale HTML5 adopters, and provides high-level guidance and explicit recommendations that will help you overcome these challenges.

INTRODUCTION

OVER THE PAST FEW YEARS THE BROWSER HAS CEMENTED ITS POSITION AS AN IMPORTANT VEHICLE FOR DELIVERING APPLICATIONS TO END-USERS. WITH THE RECENT AND RAPID DEMISE OF PLUGIN TECHNOLOGIES (FLEX, SILVERLIGHT AND JAVA APPLETS), HTML5 HAS BECOME THE DE FACTO TECHNOLOGY FOR AUTHORING BROWSER-BASED APPLICATIONS.

Whilst HTML5 is widely supported by browser manufacturers, and allows your application to reach both desktop and mobile users, it is not the easiest technology to use for enterprise-scale application development. HTML5 is nebulous and constantly changing; the tools and frameworks that support it are also in a constant state of flux. This is a double-edged sword: the evolutionary nature of HTML5 means it is constantly improving and adapting, but this makes it hard for enterprise development, where the need to keep abreast of technology changes can be costly.

HTML5 can feel quite immature when compared directly to desktop application development technologies, which are relatively stable and unchanging.

This white paper looks at the wide-reaching impact the decision to adopt HTML5 will have on your software development process. This impact goes far beyond a change in programming language, touching on tools, development techniques and mindset. For each of these challenges we share Scott Logic's experiences and offer guidance on how to deal with the attendant risks.

HTML5 can feel quite immature

A BRIEF HISTORY OF HTML5

THE WEB IN THE EARLY 90S WAS LITTLE MORE THAN A COLLECTION OF INTERCONNECTED STATIC DOCUMENTS. EARLY ATTEMPTS AT CREATING APPLICATION-LIKE EXPERIENCES, WHERE THE USER INTERACTS WITH A STATEFUL / PERSONAL WEB-APP, RELIED ON SERVER-SIDE LOGIC AND AS A RESULT WERE RESTRICTIVE AND SLOW.

The late 90s and early 2000s was the era of the plugin, with first Java Applets, and later Flex and Silverlight allowing enterprises to deliver browser-based applications to both internal and external users. This ease of distribution, without the need for installers, resulted in these technologies becoming the ideal choice for portals, dashboards, CRM solutions and a whole host of other line-of-business applications. At roughly the same time, improvements in browser performance, and technologies such as AJAX that allowed the creation of dynamic web applications, were gaining momentum. Companies like Google and Facebook were pushing the boundaries of what could be achieved with 'native' web technologies.

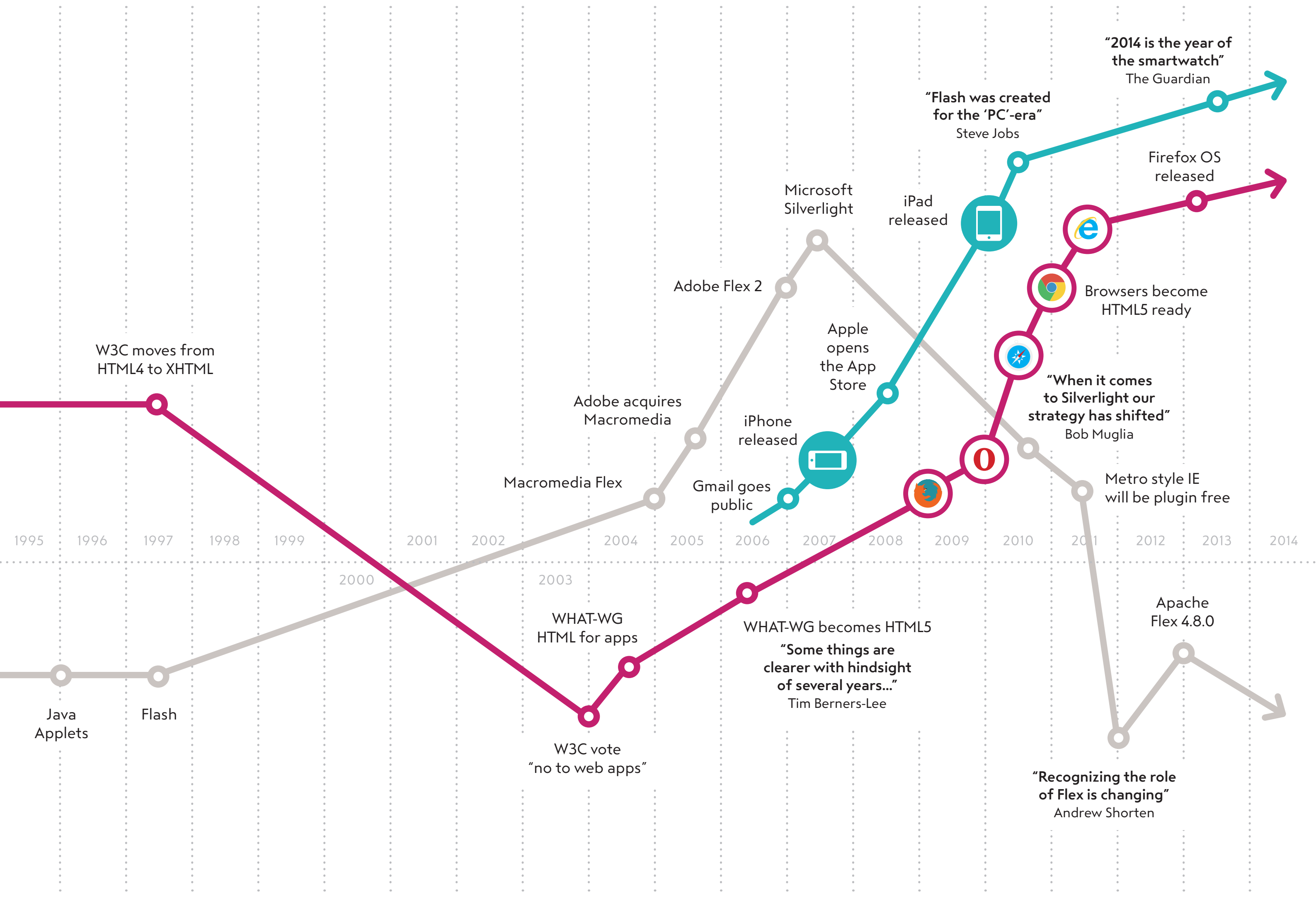
Around 2010 the way that people used the internet and interacted with computers drastically changed. The launch of the iPhone, Android, iPad and other mobile devices, coupled with the ever-increasing speed of internet connectivity over mobile networks, resulted in a seismic shift. By early 2012 smartphone sales were exceeding those of desktop PC[1], with tablets soon to follow suit.

Smartphones and tablets had little or no support for plugins, which resulted in both Silverlight and Flex being 'dropped' quite suddenly[2]. Furthermore, Google is making moves towards removing plugins altogether from their Chrome browser in the near future[3], and Mozilla are following suit[4]. Plugins simply cannot provide the same level of reach as HTML5, which is just as capable on smartphone and tablet devices as it is in the desktop browser. Furthermore, all major browser vendors are quite firmly behind HTML5, with new features being added at an unprecedented rate. The web is a more exciting place now than it has ever been before.

With HTML5, the phrase coined (and often mocked) by Sun Microsystems in the mid-90s "Write Once, Run Everywhere", is starting to look like a reality.

the way people used the internet drastically changed

[1] http://www.digitaltrends.com/mobile/smartphone-sales-exceed-those-of-pcs-for-first-time-apple-smashes-record/
[2] http://www.scottlogic.com/blog/2011/11/14/the-untimely-demise-of-the-plugin-and-how-lob-developments-will-suffer.html
[3] http://www.zdnet.com/chrome-puts-npapi-plugins-on-death-watch-7000021066/
[4] https://blog.mozilla.org/futurereleases/2013/09/24/plugin-activation-in-firefox/



The History of HTML

The Rise and Fall of Plugins

Changing User Expectations

TACKLING THE CHALLENGES

WHILST HTML5 IS THE MOST VIABLE SOLUTION FOR WEB-BASED APPLICATIONS, OWING TO ITS WIDE SUPPORT AND REACH, IT IS NOT AN EASY TECHNOLOGY TO USE. THE CHOICE TO USE 'NATIVE' WEB AS OPPOSED TO DESKTOP OR PLUGIN BASED TECHNOLOGIES WILL HAVE A SIGNIFICANT AND WIDE-REACHING IMPACT ON YOUR BUSINESS. HERE WE LOOK AT SOME OF THE CHALLENGES YOU WILL FACE AND HOW YOU MIGHT OVERCOME THEM.

/ TAKING THE PLUNGE

When businesses consider whether to use HTML5 for developing a new application, or are considering the migration of a desktop or plugin-based application, the question they will almost always ask themselves is: "Is HTML5 ready?". Unfortunately people often confuse this question with "When will the HTML5 specification be complete?", to which the answer is "never!". The WHAT Working Group, which oversees the HTML5 specification, have acknowledged the fact that the evolution of HTML has always been fluid and dynamic and that there is little point in versioning it. HTML5 is now simply HTML[5].

adopting HTML5 presents some fantastic opportunities

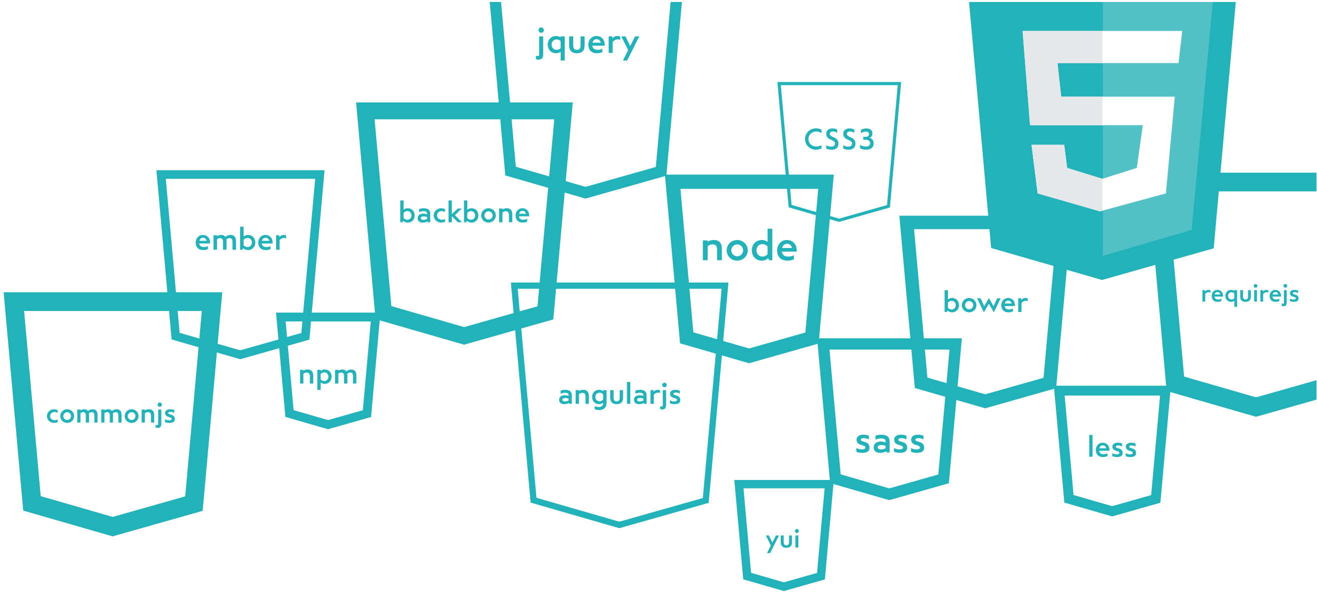
This doesn't answer the question of whether HTML5 is ready for enterprise application development. Probably the best way to answer this question is to look at the evidence around us. HTML5 is being used for numerous well-known consumer websites such as Facebook, Google Mail, Office 365 and in the enterprise for office applications, trading applications, on-line development tools, project management software and more.

Furthermore, adopting HTML5 presents some fantastic opportunities; the truly universal nature of HTML5 means that the same technology stack can be used for desktop, mobile, tablet and newer devices such as Smart TVs, significantly increasing the potential reach of your applications and services.

This still doesn't answer the question of whether HTML5 is a viable technology for your specific project. In practice, we have found that the best way to assess the viability is to plan out and prototype the development of your application. This allows you to explore the development cost, identify technical limitations and elicit all-important feedback from users and stakeholders.

USE A COMBINATION OF EXPERT ADVICE, REALISTIC PROTOTYPING AND USER FEEDBACK TO DETERMINE WHETHER HTML5 IS THE RIGHT CHOICE FOR YOU.

[5] http://blog.whatwg.org/html-is-the-new-html5



/ IT'S NOT SHRINK-WRAPPED

If you decide to use HTML5, the next step is to assemble the various tools, frameworks and libraries that are needed for enterprise-scale application development. This is where HTML5 differs the most from the plugin technologies of Flex, Silverlight and Java Applets that it has replaced.

HTML5 is still HTML. In other words, it is at heart still the same markup language that was designed for rendering static documents back at the beginning of the World Wide Web. Whilst HTML5 has added various features like local storage, web workers and canvas, which all make application development easier, it lacks much of the inherent structures, such as modules and packages, that large-scale applications require.

When developing with Flex or Silverlight you are buying into a single-vendor complete solution. For example with Flex, Adobe supply the FlexBuilder IDE which includes unit test tools, a build system, a suite of UI components, server communication libraries, source control integration and much more. On installing their tools you are presented with an environment that has everything you need for enterprise-scale application development.

HTML5 is used for a wide spectrum of applications, from static websites, to dynamic blogging or content managements systems, right up to enterprise applications with tens of thousands of lines of code. It was never intended to be a shrink-wrapped solution for enterprise application development.

HTML5 IDEs vary in their target audience and as a result vary in the functionality that they provide. An IDE that is intended for the development of graphically rich, yet mostly static websites is not the ideal tool for enterprise application development. Equally, the IDEs that are traditionally found in the enterprise (Eclipse and Visual Studio) are not the ideal tool. Products such as WebStorm, which have been designed primarily for HTML5 development provide a more feature-rich, and ultimately productive development environment.

Where HTML5 itself falls short, there are a multitude of third party libraries and frameworks, both open source and commercial, that fill these gaps. These frameworks vary in their aims, their functionality, their level of documentation and ultimately their quality. Selecting a suitable set of frameworks can significantly enhance productivity.

A typical approach to HTML5 development is to add frameworks and tools as and when they are required, with very little strategic thinking. For small-scale developments this is rarely an issue, but for enterprise applications this can result in a complex mess of overlapping frameworks (e.g. multiple charting or DOM utility libraries), which result in a bloated application with hard-to-manage dependencies

SELECT A SUITABLE IDE AND A SET OF FRAMEWORKS, THAT SPAN THE TECHNICAL REQUIREMENTS OF ENTERPRISE APPLICATIONS. A TECHNICAL PROOF OF CONCEPT THAT DEMONSTRATES THE KEY FEATURES OF THE APPLICATION IS AN EXCELLENT VEHICLE FOR ASSEMBLING THE REQUIRED TOOLS.

/ JAVASCRIPT IS NOT JAVA!

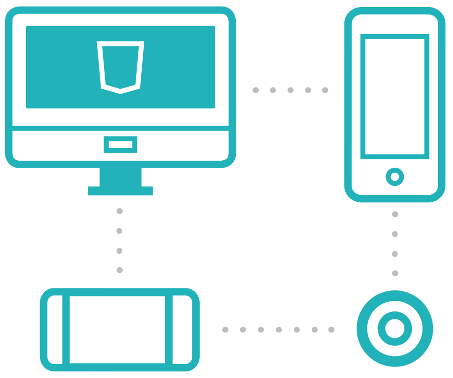
Skilled developers are able to transition between related object-oriented languages such as C# and Java with relative ease. However, learning JavaScript can be quite a challenge, even for experienced developers.

In contrast to most languages used for enterprise application development, JavaScript is dynamic and lacks a strongly-defined type system. As a result, it lacks the kind of code completion and refactoring tools that Java and C# developers are used to. Furthermore the dynamic nature of JavaScript means issues that developers would typically catch at compile time are instead discovered at run time.

HTML5 is one of the top job trends at the moment[6]. However, in our experience it can be hard to find JavaScript developers who have experience in developing enterprise-scale applications. Because HTML5 is used for a broad spectrum of uses, from simple websites to complex applications, many developers have had some experience of this technology, but very few can be considered masters.

INVEST IN TRAINING YOUR DEVELOPERS IF YOU SEEK TO TRANSITION THEM FROM TRADITIONAL OBJECT ORIENTED LANGUAGES TO JAVASCRIPT. AN EXCELLENT WAY TO MAKE THIS A SMOOTH TRANSITION IS TO BRING IN EXTERNAL EXPERTISE TO TEACH AND MENTOR THE DEVELOPMENT TEAMS.

HTML5 is one of the top job trends at the moment



/ TESTING 1 ... 2 ... 3

Testing is of paramount importance for enterprise applications, where bugs that emerge when a system is live can have serious business impact and a high resultant cost.

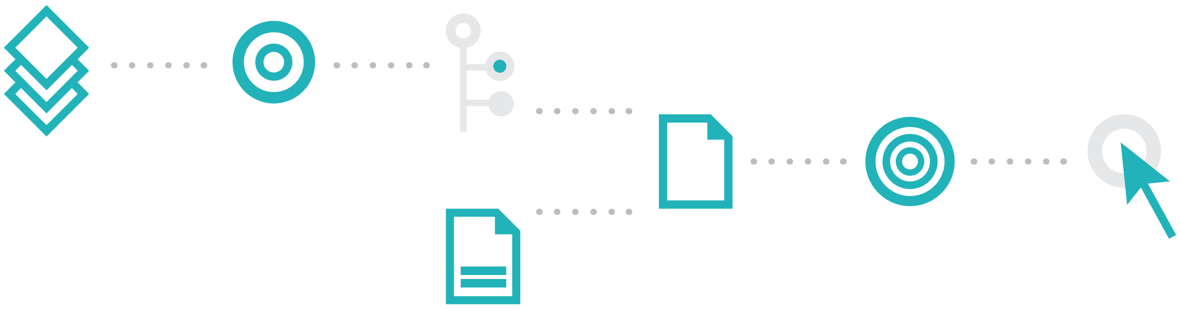
The dynamic nature of JavaScript, where certain classes of error are only caught at run time increases the need for a well thought out test plan. Such runtime errors can be minimized via unit tests that are run automatically as part of the development process.

With plugin and desktop development the execution environment for your application is almost exactly the same for every client. However, with HTML5, differences between browsers (e.g. Chrome, Safari, IE), and between versions of the same browser, can cause additional issues to arise. As a result, a greater emphasis on system testing is required for HTML5 applications.

There are some advanced browser automation tools, such as Selenium[7] that can minimize the cost of testing across a range of browsers. However, there will always be bugs relating to visual rendering issues that automated test tools will be unable to catch, hence there should always be an element of manual testing.

ADOPT A TEST STRATEGY INVOLVING UNIT TEST AND A COMBINATION OF MANUAL AND AUTOMATED SYSTEM TESTING. CONSIDER ALLOCATING A LARGER TEST BUDGET FOR HTML5 APPLICATIONS.

[6] http://www.indeed.com/jobtrends
[8] http://docs.seleniumhq.org/



/ MODULES AND CODE SHARING

Enterprise applications are often developed by large teams working in parallel on a number of different functional areas. The JavaScript language currently lacks any built-in structures for organizing related classes (such as packages or namespace), and lacks a way of creating modules (such as JARs or assemblies).

There are a number of solutions to these problems that are starting to become mature; these include npm and bower which are package managers for JavaScript code, and CommonJS or RequireJS which both provide runtime resolution of dependencies. These tools are certainly mature enough for enterprise use.

USE TOOLS TO MANAGE DEPENDENCIES AND ALLOW TEAMS TO WORK INDEPENDENTLY ON DISCRETE MODULES.

/ BUILD TOOLS

JavaScript does not require compilation and neither do CSS or HTML. For simple web pages and web sites there is no real need for build tools, since the development and production assets are one and the same. Whilst this is true for simple web pages, for enterprise-scale applications, the code that is delivered to production is not the same as the code which runs in development.

For JavaScript, compilation is required to resolve module dependencies, and optimization is required to reduce the download size. HTML will typically be pre-processed to replace environment variables and load the production or development JavaScript. We recommend that technologies such as SaSS and LESS are used to allow a more structured approach to CSS, with support for variables and improved semantics.

If your team is currently creating desktop applications or simple web apps they will no doubt be using build tools such as Ant, Maven or MSBuild. JavaScript has its own build tools, such as grunt, which are far more suited to the task of building enterprise HTML5 applications.

INVEST TIME IN SELECTING AND CONFIGURING A SUITABLE BUILD, USING MODERN TOOLS SUCH AS GRUNT.

/ LEGACY BROWSERS

Probably one of the biggest obstacles that businesses face when adopting HTML5 is the need to support legacy browsers. Whilst most home users have relatively modern browsers such as Chrome, there are still a great many corporate users using archaic browsers, typified by Internet Explorer 6. Legacy browsers present many obstacles to HTML5 development: they do not support HTML5 features such as canvas and video, their JavaScript engines are slow, and they have a great number of quirks and bugs.

It should not be considered a show-stopper that some of your users rely on legacy browsers. Many of the HTML5 features such as canvas, video and CSS3 have widely used and tested alternatives for legacy browsers. Furthermore, HTML5 support should not be considered a pre-requisite for web application development; any browser that supports JavaScript can be used to host a dynamic web application.

legacy browsers should not be considered a show-stopper

Broadly speaking there are two different approaches that can be used to accommodate legacy browsers:

1. Invest development time to tackle and solve the technical issues that arise from legacy browsers. This involves a combination of adding code that specifically targets these browsers, and the use of third-party tools, shims and abstraction layers. The aim here is to ensure that the application is pixel-perfect on each and every supported browser.

2. Allow the application to gracefully 'degrade' in order to support users with legacy browsers. The aim here is to deliver the same business functionality to all users, but with a lower-fidelity experience for legacy browsers.

In our experience, projects that adopt the first approach become very costly, with significant development time being invested in resolving issues for a small group of legacy browser users.

Our view is that the best approach is to gain an understanding of the browsers that end-users have available to them, and employ a combination of both approaches outlined above. The browsers can be divided into 'tiers' where the top tier have the best user experience, the middle tier are functional but have a reduced experience, and the lower tier are not supported. With this tiered approach, the cost impact of supporting specific browsers, and the end-user benefits (based on browser usage statistics), can be quantified.

COLLECT STATISTICS THAT DETERMINE THE BROWSERS THAT YOUR END-USERS HAVE AVAILABLE TO THEM, AND FROM THIS DERIVE A STRATEGY THAT MAPS THESE BROWSERS INTO 'TIERS' OF SUPPORT.

CONCLUSIONS

THIS PAPER HAS OUTLINED A NUMBER OF CHALLENGES, SOME OF WHICH YOU MAY HAVE BEEN AWARE OF, AND HOPEFULLY SOME THAT WILL HAVE BEEN NEW TO YOU. DESPITE THESE CHALLENGES, IT IS POSSIBLE TO RUN HIGHLY SUCCESSFUL HTML5 PROJECTS THAT REAP THE BENEFITS OF THIS UNIVERSAL TECHNOLOGY. A SUMMARY OF THESE RECOMMENDATIONS IS GIVEN BELOW:

- ✓ **Assemble the required tools and frameworks up-front**
- ✓ **Invest in HTML5 training**
- ✓ **Bring in external expertise to ‘steer’ your team**
- ✓ **Allow for a bigger budget for testing**
- ✓ **Use modules to manage dependencies**
- ✓ **Adopt JavaScript-centric build tools**
- ✓ **Collect browser version information from your users**
- ✓ **Adopt a strategic and tiered approach to supporting different browsers**

»

HTML5 is ever-changing, with the coming year promising to bring new technologies, tools and approaches. If you want to know more about what this year holds for HTML5, our follow-up white paper “HTML5: Future Directions” provides insights from our technology experts who have their fingers on the HTML5 pulse!

Scott Logic is a leading software consultancy with enterprise clients in finance, energy, healthcare and the public sector. Our services include software development & delivery, user experience design and an independent consultancy advising on strategic software selection and deployment. Our consultants, recruited on their exceptional qualifications, skill sets and knowledge, are central to continued project success and complete client satisfaction.

If you would like to discuss your HTML5 strategy, please get in touch:

enquiries@scottlogic.com

SCOTT LOGIC / ALTOGETHER SMARTER

3rd Floor, 1 St James' Gate
Newcastle upon Tyne
NE1 4AD

+44 845 224 1930

www.scottlogic.com