HTML5 MIGRATION

GET STARTED WITH A 'LITTLE BANG'

/ A WHITE PAPER BY COLIN EBERHARDT

SCOTT LOGIC

ALTOGETHER SMARTER

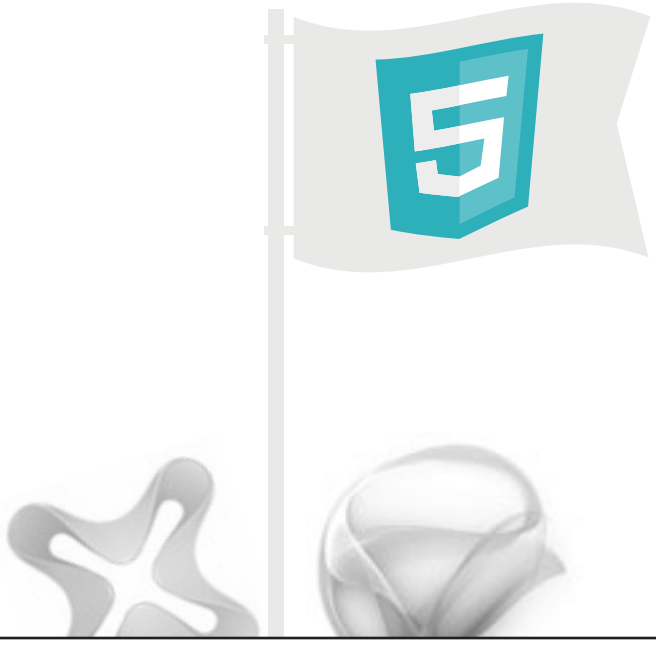
INTRODUCTION

IT'S 2015; THE FIGHT BETWEEN SILVERLIGHT, FLEX AND HTML5 FOR WEB DOMINATION WAS WON YEARS AGO, WITH HTML5 EMERGING VICTORIOUS. SURELY NOBODY IS STILL USING FLEX AND SILVERLIGHT ANYMORE; WELL, PERHAPS SURPRISINGLY, BUT YES?

At Scott Logic we have talked with a number of prospective clients with complex Flex or Silverlight applications they are yet to migrate, because among other reasons, they feel the cost would be prohibitive. By continuing to postpone the inevitable migration from these systems to HTML5, these business face losing customers who favour mobile and tablet devices.

The real risks involved in choosing to delay include a perceived and actual loss of in-house technical innovation, difficulty finding developers who want to work with legacy plug-in technologies and the genuine possibility that an application could suddenly stop working when browsers finally stop supporting plug-ins.

We've taken a closer look at why you should start the migration process, and just as importantly, how. We recommend a 'Little Bang' approach which tackles the cost issues to deliver an early return on investment. There really is no reason why you should put off migrating to HTML5 any longer!



WHY HTML5 WON THE WEB

HOW DID SO MANY COMPANIES COME TO INVEST SO MUCH IN LEGACY PLUG-IN TECHNOLOGIES? TO ANSWER THIS, LET'S TAKE A QUICK LOOK AT THE EVOLUTION OF THE WEB.

More than a decade ago, companies started moving applications from desktop to the web, taking advantage of its accessibility and the ease by which web-hosted software could be introduced and updated.

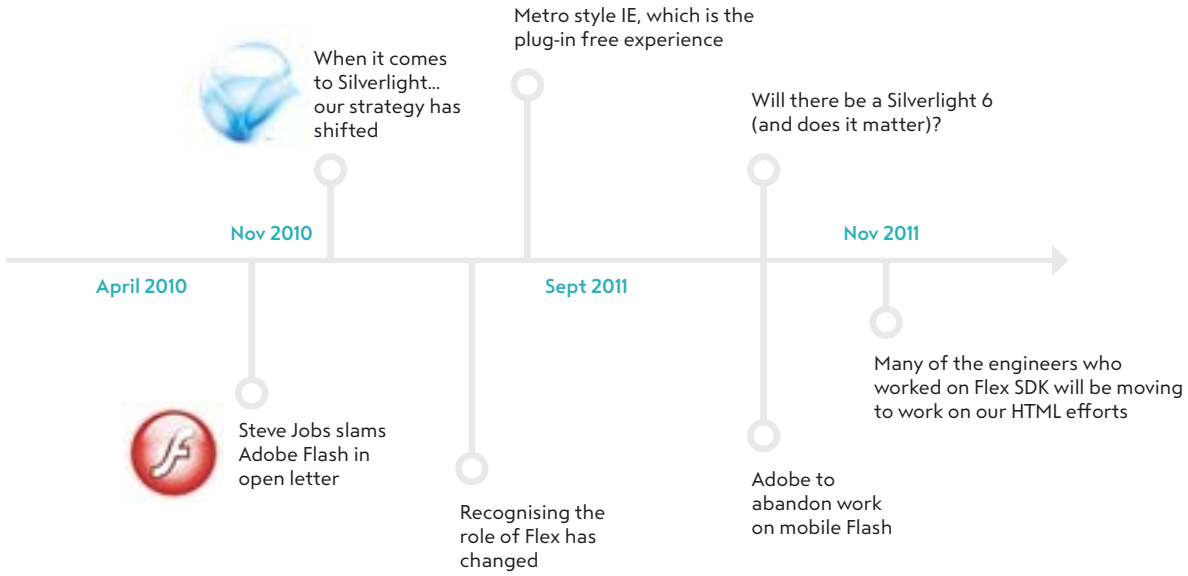
As HTML was immature at this point, organisations turned to plug-ins. First, Java Applets and Flash, followed by Silverlight and Flex. At around the same time, HTML5, a more feature-rich HTML specification was beginning to gain momentum.

In 2011 we published our own white paper at Scott Logic, *Flex, Silverlight or HTML5? Time To Decide* [1], in which we predicted that HTML5 would eventually supplant its plug-in rivals. What took us - and many others - by surprise was just how quickly this happened.

There are a number of factors that contributed to the meteoric rise of HTML5, including heavy investment by Google, Microsoft and Adobe. The HTML5 open source community was also lively, and of course the ubiquity of mobile devices was a major factor.

Unfortunately this put those who had heavily invested in plug-in technologies in a difficult position - HTML5 was still relatively immature, (see *The untimely demise of plug-ins* [2]), so many took the decision to wait before migrating...until migration became easier, for HTML5 to mature, or perhaps for a future uprising of plug-ins?! Many are still waiting.

It's easy to understand why; if you've invested millions in an amazing Flex or Silverlight application, you are likely to be reluctant to spend what could be millions more on an HTML5 equivalent.



[1] http://blog.scottlogic.com/2011/05/05/flex-silverlight-html5-time-to-decide.html
[2] http://blog.scottlogic.com/2011/11/14/the-untimely-demise-of-the-plugin-and-how-lob-developments-will-suffer.html

WHY MIGRATE, AND WHY NOW?

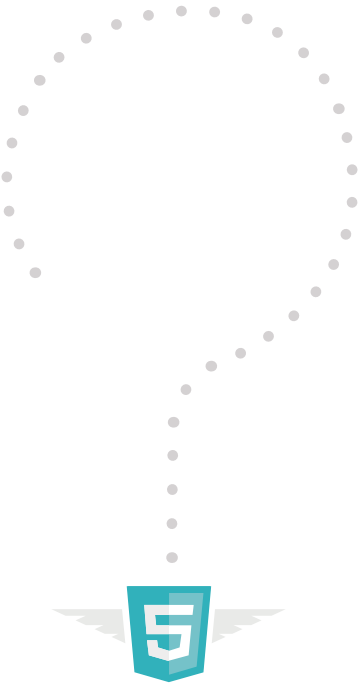
THERE ARE MANY TECHNICAL AND ECONOMIC REASONS FOR EMBARKING ON AN IMMINENT MIGRATION. ON ITS OWN, EACH OF THESE MIGHT NOT BE REASON ENOUGH, BUT WHEN COMBINED, THE STORY IS QUITE COMPELLING.

In terms of support, maintenance and ongoing development, Flex and Silverlight have a way to go. Microsoft released the last major update to Silverlight in 2011 and has moved the framework into long-tail maintenance mode. The Flex situation is a little brighter. Adobe open sourced the framework in 2011, donating it to the Apache Software Foundation, with a number of releases since.

Flex and Silverlight applications run within the browser using the plug-in model and associated APIs. Five years ago, a significant concern for Silverlight developers was the overall installation volume of the plug-in. There is now a far more significant threat to plug-ins; browser manufacturers are actively removing plug-in APIs, and there's a hard deadline by which these technologies will no longer be useable. Silverlight no longer works in Chrome since version 42, and since July 2015, Firefox has actively blocked the Adobe Flash Player that Flex relies upon.

The rise in popularity of mobile and tablet devices helped HTML5 win the plug-ins battle and highlights the fact that as we become increasingly mobile, businesses run the risk of losing custom if applications don't run on popular consumer devices.

The loss of innovation is another growing factor affecting both Silverlight and Flex. Active current technologies have lively ecosystems and communities, resulting in open source projects, conferences and ultimately innovation. The loss of this ecosystem, which Flex and Silverlight were both part of in the past, has resulted in these technologies stagnating, and developers are moving elsewhere to share knowledge and experience. It is becoming a significant challenge to find developers who want to work with either technology.



Many teams have postponed migration in the hope it would get easier, that tooling would be developed that would ease the process. There have been some innovations in this direction. For example, Google automatically converts Flash advertisements to HTML5 as part of its ad network. There are also some open source projects that attempt to run Silverlight applications within JavaScript emulation environments (eg. CSHTML5 and Fayde). However, these projects are a long way off being able to convert or run complex enterprise applications, and likely never will be. Migration isn't going to get any easier, and the longer a team postpones the inevitable, the more code there'll be to migrate.

The final obstacle that has historically stopped teams from migrating is the maturity of HTML5. I think it is now universally accepted that HTML5 is a mature and appropriate choice for a great number of enterprise applications. It has been widely embraced by Google, Facebook, Microsoft and Twitter, all of whom are releasing mature, production-ready frameworks.

It's also worth pointing out that HTML5 is a continually evolving specification; it will never be complete, and there will never be an HTML6, but this is a good thing!

WHY MIGRATE?

- **Silverlight and Flex are reaching end-of-life, as will your platform if you fail to migrate**
- **Migration isn't getting any easier - the longer you wait, the greater the cost as your platform grows**
- **Your competitors that have already migrated are now benefiting from being part of an innovative technology community**
- **Demand for mobile services is real, both in the workplace and at home**
- **HTML5 is mature and ready**
- **Building an internal capability including modern web and mobile development is essential given the increase of digital disruption.**

MIGRATION STRATEGIES

BROADLY SPEAKING THERE ARE TWO MIGRATION STRATEGIES:

/ BIG BANG

The 'Big Bang' approach is pretty self explanatory - create and match feature-for-feature an HTML5 equivalent of your Flex or Silverlight application. Once complete, turn off the plugin-based application and launch the HTML5 app. There are a number of factors to consider with this approach:

Pros:

- It's technically straightforward, assuming you have a team of capable HTML5 developers, as it doesn't require any integration between HTML5 and the plug-in code it's seeking to replace
- You only ever have to support one platform. When the migration is complete, simply turn off the plug-in application and turn on the HTML5
- HTML5 is still a marketable technology with companies heavily promoting the release of HTML5 applications. For example, as stated by Saxo Bank regarding its SaxoTraderGo application, "Enjoy the convenience of HTML5 technology for a richer, faster application experience, directly in your browser on any device you choose."

Cons:

- It can be very costly and time consuming. For a complex Flex or Silverlight application that took many years to develop, expect the migration project to also take years
- This means you'll be migrating a moving target as the plug-in-based application will no doubt undergo changes and enhancements throughout the process.

/ HYBRID MIGRATION

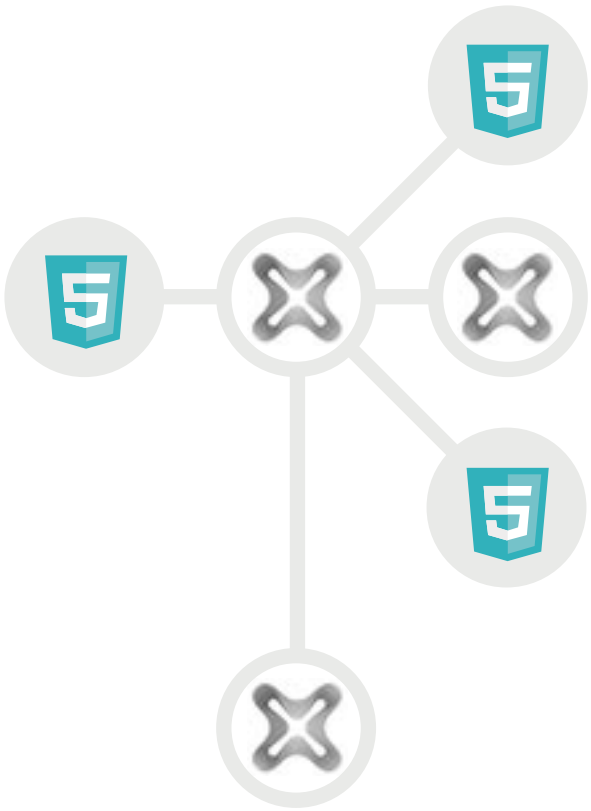
An alternative to the 'Big Bang' approach is to gradually migrate the application piece-by-piece. This results in a hybrid application where some of the logic and UI is written using HTML5, and some still uses Flex or Silverlight.

Pros:

- This minimises up-front investment, allowing for incremental delivery
- You only have a single platform to support

Cons:

- It's technically challenging. While plug-ins sit quite happily within HTML5 applications, it isn't so easy to integrate the other way round, eg. HTML5 embedded in a Flex or Silverlight application
- This results in a lot of throw-away code, with the complex 'glue' logic required to create the hybrid eventually deleted
- Hybrid applications result in User Experience (UX) compromises. It isn't always possible to locate the seam between the two technologies
- If your concern is reaching a mobile/tablet audience, this is a very slow route to delivery.



Neither approach to migration is ideal; the 'Big Bang' is technically simpler but requires considerable time and financial investment, while the hybrid approach is technically more complex and lengthy, and runs the risk of a compromised user experience.

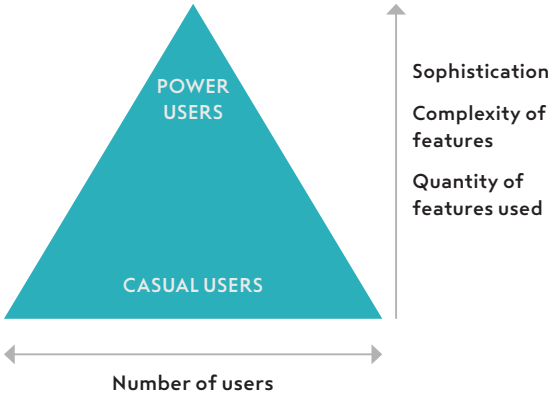
Somewhere between the two is a different and altogether more efficient process which you'll find by taking a closer look at your user base and their needs.

LEVELS OF SOPHISTICATION

YOUR USERS ARE NOT ALL ALIKE; THEY HAVE DIFFERENT NEEDS, AND DIFFERENT LEVELS OF EXPERIENCE AND SOPHISTICATION.

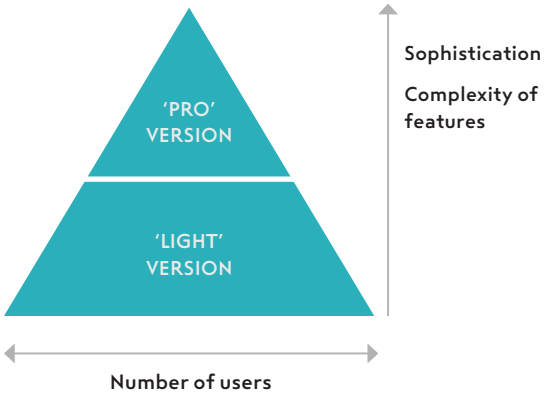
For most applications if you analyse the users you'll likely find there are only a small number who make use of the more complex features.

These are your 'power users'. At the other end of the spectrum, the majority will tend to use the most basic features.



Businesses often acknowledge this by releasing different versions of their application.

For example they might provide a 'pro' version with an expansive range of features, and a light version with a reduced feature set and simplified User Interface (UI).



BUT WHY STOP THERE? IT SHOULD BE POSSIBLE TO MAKE FURTHER DIVISIONS UNTIL YOU REACH A SMALL SET OF FEATURES TO SATISFY THE NEEDS OF A SUBSTANTIAL NUMBER OF USERS.

LITTLE BANG

ONCE AN INITIAL SET OF USERS AND FEATURES IS REVEALED, A NEW HTML5 PRODUCT CAN BE CREATED THAT TARGETS THEM SPECIFICALLY. THIS IS A 'BIG BANG' MIGRATION, BUT RATHER THAN TREATING IT AS A MONOLITH, IT CAN BE MIGRATED IN STAGES, OR 'LITTLE BANGS'

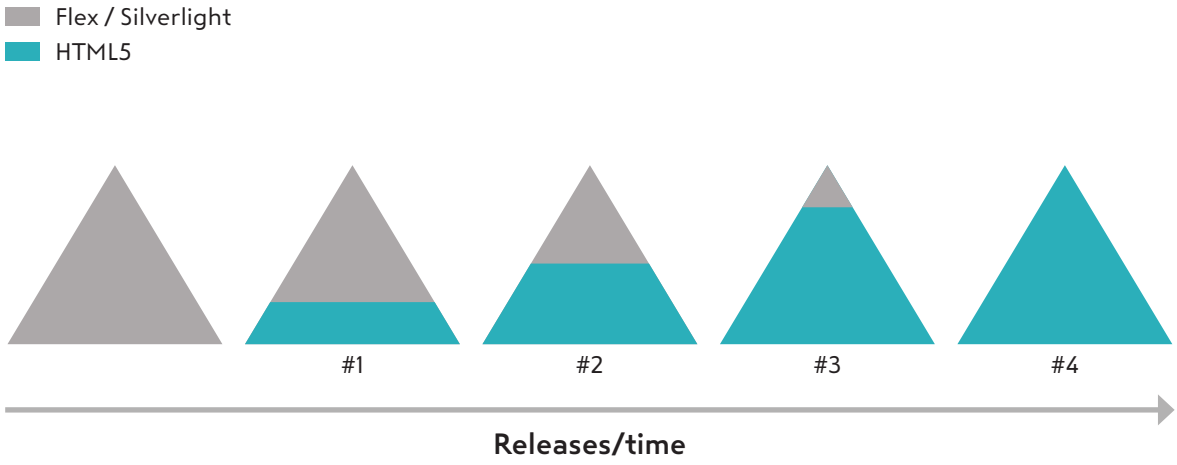
Migration provides an opportunity to refocus on users and remove inefficiencies, both technically and from a UX point of view. It also has a short time to market, delivering an early return on investment.

Another significant advantage of using 'Little Bangs' is that it allows the delivery of a mobile or tablet application early in a migration process, potentially with the first release. As a consequence it can reach a bigger target audience and provide a core message for the launch of the new platform.

After the first 'Little Bang' release, more features can be brought into the HTML5 application, at each stage bringing more users on board and increasing the platform's overall sophistication. This iterative approach is not only beneficial from a business perspective but also aligns you with the evolutionary nature of HTML5, allowing successive releases to capitalise on new HTML features as they mature.

Possibly the biggest disadvantage of this technique is that the legacy plug-in application will have to be maintained for a certain subset of users. However, in our experience, withdrawing a product and replacing it with an alternative can be quite a challenge.

Power users tend to be opinionated and opposed to change. What you can guarantee with the 'Little Bang' approach is that the number of users on your legacy platform will decrease over time and they can be given the option to move between platforms at their own speed.

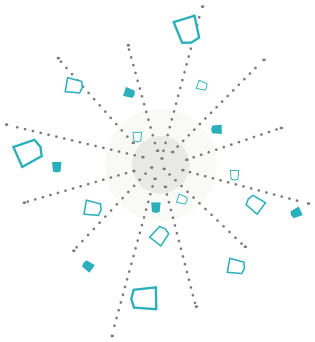


CONCLUSIONS

Migration of complex business applications is a costly exercise and there is an understandable reluctance to embark on such a journey. However, we'd argue that putting off the inevitable only increases the future cost and runs the risk of losing customers to more pro-active competitors.

To make the migration process and costs manageable, a 'Little Bang' approach should be adopted, where the most widely used features are gradually migrated to HTML5. This lowers the technical risk, delivers an early return on investment and allows you to take advantage of technical innovations to provide mobile and tablet solutions, and start building your future platform.

- Colin Eberhardt



Scott Logic is a leading software consultancy with enterprise clients in finance, energy, healthcare and the public sector. Our services include software development and delivery, User Experience (UX) design and independent consultation 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 at

enquiries@scottlogic.com

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

+44 333 101 0020

48 Warwick Street
London
W1B 5AW

+44 333 101 0020

25 King Street
Bristol
BS1 4PB

+44 333 101 0020

1st Floor, 80 George Street
Edinburgh
EH2 3BU

+44 333 101 0020

www.scottlogic.com