CHALMERS

# Rich Internet Applications (RIAs)

A Comparison Between Adobe Flex, JavaFX and Microsoft Silverlight

*Master of Science Thesis in the Programme Software Engineering and Technology*

CARL-DAVID GRANBÄCK

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Göteborg, Sweden, October 2009

Rich Internet Applications (RIAs)
A Comparison Between Adobe Flex, JavaFX and Microsoft Silverlight

CARL-DAVID GRANBÄCK

Examiner: BJÖRN VON SYDOW

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden, October 2009

# Abstract

This Master's thesis report describes and compares the three Rich Internet Application (RIA) frameworks Adobe Flex, JavaFX and Microsoft Silverlight. Through a qualitative study, their technical and non-technical characteristics have been identified and compared, together with an analysis of the current market. A prototype application has been implemented in JavaFX to evaluate its applicability in more detail. The purpose of this report is to provide guidance in choosing what plugin-based RIA framework to commit to.

Flex has been on the market twice as long as Silverlight, and five times longer than JavaFX. It is the most mature and established framework, but Silverlight has experienced a rapid development with frequent releases, and established itself as the main competitor to Flex. The characteristics and features of these two frameworks do not differ much. The biggest difference is their market penetration, where Flex (Flash) has 95% of the desktop market, and Silverlight 26%. Both frameworks offer great tools and a designer-developer workflow that can enhance productivity.

JavaFX reaches 71% of the market by running on top of the Java Runtime, but it has several shortcomings such as accessibility and usability issues, as well as buggy components and no visual designer. The study concludes that JavaFX is not currently able to compete with neither Silverlight nor Flex. In addition to these plugin-based RIA frameworks, it is also important to follow the progress of HTML 5, the next generation markup language for the Web, which has been predicted to become a game-changer in web application development.

## Keywords

Rich Internet Applications, RIA, Adobe Flex, JavaFX, Microsoft Silverlight, comparison, comparative study

## Sammanfattning

Detta examensarbete beskriver och jämför de tre RIA-ramverken (Rich Internet Application) Adobe Flex, JavaFX och Microsoft Silverlight. Deras tekniska och icke-tekniska egenskaper och funktionalitet har identifierats och jämförts genom en kvalitativ studie, tillsammans med en analys av det aktuella marknadsläget. En prototypapplikation har utvecklats i JavaFX för att bättre kunna bedöma dess lämplighet. Syftet med denna rapport är att underlätta vid valet av ett plugin-baserat RIA-ramverk.

Flex har funnits på marknaden dubbelt så länge som Silverlight, och fem gånger längre än JavaFX. Det är det mest stabila och etablerade ramverket, men Silverlight har upplevt en snabb utveckling med frekventa nylanseringar och har blivit en huvudkonkurrent till Flex. Egenskaperna och funktionaliteten hos dessa två ramverk skiljer sig inte mycket åt. Den största skillnaden är marknadsandelen, där Flex (Flash) har 95% av datormarknaden och Silverlight 26%. Båda ramverken erbjuder utmärkta verktyg och ett så kallat *designer-developer -workflow* som kan öka produktiviteten.

JavaFX når 71% av marknaden genom att det körs ovanpå Javas runtime, men har flera brister som t.ex. tillgänglighet och användbarhet, samt buggiga komponenter och inget visuellt verktyg för designers. Den här studien kom fram till att JavaFX för tillfället inte kan mäta sig med varken Silverlight eller Flex. Utöver de här plugin-baserade RIA-ramverken är det även viktigt att följa utvecklingen av HTML 5, nästa generations märkspråk för webben, vilket tros kunna påverka stort inom webbutveckling.

### Nyckelord

Rich Internet Applications, RIA, Adobe Flex, JavaFX, Microsoft Silverlight, jämförelse, jämförande studie

## Acknowledgements

This Master's thesis report has been written as the final part of the Master's programme Software Engineering and Technology at Chalmers University of Technology, Sweden. The subject was chosen in collaboration with Capgemini in Gothenburg, where the thesis also has been performed.

I would like to thank the following people for their help and the support that they have given throughout my work:

**Capgemini**

Samuel Ericson
David Glans
Lars Johansson
Hannes Vedin

**Chalmers University of Technology**

Björn von Sydow

*Carl-David Granbäck*

Gothenburg, Sweden

October 2009

# Table of Contents

# Glossary

| | |
|---|---|
| AIR | Adobe Integrated Runtime |
| AJAX | Asynchronous JavaScript and XML |
| AMF | ActionScript Messaging Format |
| API | Application Programming Interface |
| CPU | Central Processing Unit |
| CSS | Cascading Style Sheets |
| DOM | Document Object Model |
| EJB | Enterprise JavaBean |
| FPS | Frames Per Second |
| FXG | Flex Graphics |
| GIF | Graphics Interchange Format |
| GPL | GNU General Public License |
| GPU | Graphics Processing Unit |
| GUI | Graphical User Interface |
| GWT | Google Web Toolkit |
| JSF | JavaServer Faces |
| HTML | Hyper Text Markup Language |
| HTTP | Hypertext Transfer Protocol |
| IDE | Integrated Development Environment |
| IPTV | Internet Protocol Television |
| JNLP | Java Network Launching Protocol |
| JRE | Java Runtime Environment |
| JSON | JavaScript Object Notation |
| PDF | Portable Document Format |
| REST | Representational State Transfer |
| RIA | Rich Internet Application |
| RSS | Really Simple Syndication |
| RTMP | Real Time Messaging Protocol |
| RTSP | Real Time Streaming Protocol |
| SEO | Search Engine Optimization |
| SDK | Software Development Kit |
| SOAP | Simple Object Access Protocol |
| SVG | Scalable Vector Graphics |
| SWF | Shockwave Format (or Small Web Format) |
| UDP | User Datagram Protocol |
| UI | User Interface |
| W3C | The World Wide Web Consortium |
| WPF | Windows Presentation Foundation |
| WYSIWYG | What You See Is What You Get |
| XAML | Extensible Application Markup Language |
| XML | Extensible Markup Language |

# 1. Introduction

This section aims to reveal the story behind this Master's thesis, and provide a background of the subject. The introduction will briefly describe the history of Rich Internet Applications and why this topic is interesting. It will also explain the purpose and objectives, including to whom this report is aimed at.

## 1.1. Background

The software industry is a constantly evolving ecosystem, where new techniques and new applications are frequently born. Thanks to increasingly faster devices powering this massive environment, and the fact that most of them are connected to the Internet, the possibility to create richer user experiences has indeed become tangible. New possibilities give rise to desires, and desires give rise to needs, which eventually turns into expectations. Therefore, to satisfy the users' needs and prepare for their expectations, it is of great importance to analyze new techniques and new trends.

Today, a clear trend could be seen where more and more frameworks, for creating richer user experiences, appear. The dominant companies within this field of the software industry are competing with each other to provide the best solutions, and it could be hard to decide what technology to commit to in this rapidly changing environment. Hence, software architects and developers are facing multiple options when choosing frameworks for this type of applications.

### 1.1.1. Rich Internet Applications

HTML was initially created to statically display text and images on web pages, and not designed for high interactivity. Today, this specification would not be enough to cater for the users' needs, since they expect interactive and highly responsive websites. This is where the term Rich Internet Applications (RIAs) comes into play. RIAs is an umbrella term used for describing applications that are more desktop-like, with "richer" content and more interactivity. This includes both lightweight browser-based frameworks with JavaScript and AJAX that do not require a plugin to be installed, but also more heavy-weight plugin-based RIA frameworks, sometimes referred to as RIA platforms, which this report will focus on.

There is a wide variety of frameworks available today, and the trend towards richer interfaces is prevalent. However, RIAs are not something completely new. Long before the term was even coined, in 1995, Sun introduced something called Java Applets - small applications that could run inside the browser, using Java's runtime plugin. After that, Macromedia released Flash, which original intent was to enable animations on the Web. JavaScript gained popularity in the end of the 90s, and added even more interactivity to static web pages. These techniques were slowly being mixed, and in 2004 another well-known term was coined, namely Web 2.0.

Web 2.0 can be thought of as being a way to think about the evolution of the Web, the way developers create applications for it, and the way the users use it. The term is often associated with rich and user-friendly interfaces, less page-reloading (using AJAX) and the use of standards. Web 2.0 is also referred to as the "participatory Web", where more user participation, collaboration and open APIs together open up possibilities for much more powerful applications. (O'Reilly, 2005)

Nowadays, web applications can run not only in the browser, but also in many mobile devices and offline as desktop applications by using browser plugins like for example Google Gears or Mozilla Prism. But still, JavaScript- and AJAX-based web applications cannot leverage the same power as plugin-based frameworks. Plugin-based frameworks can directly manipulate graphics, perform multi-threaded computations and fully support audio and video. Major companies like Adobe, Microsoft and Sun Microsystems all provide their framework in this field, and there is a clear trend towards richer web applications in general. Last but not least, the specification of HTML 5, the next generation markup language for the Web, is currently under development and the progress of this upcoming standard is important to follow.

Throughout this report, Adobe Flex will often be referred to as Flex, and Microsoft Silverlight to simply Silverlight. Another important statement to make is the versions that this report is based upon, namely Adobe Flex 4 Beta, JavaFX 1.2 and Microsoft Silverlight 3. In this case, the meaning of a beta version is that both functionality, specifications and documentation is preliminary and might change over time, even though it is not likely that the majority of it would. Some information about the beta version, such as system requirements for the final release, has not been available at the time of writing, and in such cases information from the latest stable version has been used.

### 1.1.2. Capgemini

Capgemini is one of the world's foremost providers of consulting, technology and outsourcing services with headquarters in Paris, France. Having offices in more than 30 countries, Capgemini reported 2008 global revenues of EUR 8.7 billion and employs 90,000 people worldwide. At the Gothenburg office in Sweden, all Capgemini divisions are represented, and within custom software development there is a team focusing on Java technologies.

By being a consultant company with a wide customer base, Capgemini continuously needs to stay on top of trends and technologies. A strategic framework called TechnoVision has been initiated to conform to these needs, and Rich Internet Applications has been recognized as a key IT trend. A thorough study of RIAs can help Capgemini to keep up with this trend, and simplify their architectural decision makings from a technical perspective.

## 1.2. Purpose

The purpose of this thesis report is to study Rich Internet Applications in general, and compare the plugin-based frameworks Adobe Flex, JavaFX and Microsoft Silverlight in particular. The chosen RIA frameworks will be compared from a technical and non-technical perspective, looking at both features, market and trends. This thesis aims to identify the differences between these three well established frameworks, and evaluate each framework's applicability and maturity. The ambition is that the result of the study will provide guidance in finding what framework to use, and what aspects that could affect technical decisions. Due to a focus on Java technologies at Capgemini in Gothenburg, extra attention will be paid to JavaFX by getting hands-on experience through a prototype implementation.

## 1.3. Objectives

The objectives of this Master's thesis report are essentially to answer the following questions:

- What are the technical and non-technical differences between the three RIA frameworks Adobe Flex, JavaFX and Microsoft Silverlight?
- How does the current market look, and what speaks for each framework today and in the future?
- How can JavaFX be used as a front-end system, and what advantages/disadvantages would this imply?

## 1.4. Target Groups

This report is aimed at people who want to learn about Rich Internet Applications, and more specifically the plugin-based frameworks Adobe Flex, JavaFX and Microsoft Silverlight. The study would be of interest for anyone ranging from IT students, developers and software architects, to business analysts or industry leaders that want to know the possibilities and limitations with each framework, as well as future trends and what aspects that can guide them in their technical decisions.

## 1.5. Delimitations

This study focuses on the three plugin-based RIA frameworks Adobe Flex, JavaFX and Microsoft Silverlight. It also gives a general background of related frameworks and similar techniques, but only to explain the differences and not go into detail. Each framework has been studied to such an extent that all relevant differences have been discovered, and all important aspects have been identified.

A JavaFX prototype has been developed to complement the analysis from a practical perspective. Its purpose is to clarify aspects of the theoretical part of the study. The prototype is not meant to replace an existing application, or be put into operation and publicly deployed, but rather constitute a proof of concept implementation to demonstrate the applicability of JavaFX. It has been based upon an already existing Flex application, but due to a limited amount of time, the prototype was only implemented in JavaFX and hence works as a means of comparison between these two frameworks, but not Silverlight.

# 2. Method

This section describes and justifies the method used in this Master's thesis. On a high level, the work was divided into two different parts. Firstly, a theoretical part referred to as the framework comparison, which included information gathering, analysis and comparison. Secondly, a practical part consisting of a prototype implementation.

## 2.1. Framework Comparison

The frameworks were chosen in collaboration with Capgemini, who were interested in the three established RIA frameworks Adobe Flex, JavaFX and Microsoft Silverlight, and especially the possibilities of JavaFX as a front-end. A qualitative study based on a comparison of the chosen frameworks was performed. First, sources and characteristics about RIAs in general, was identified. Aspects like technical differences and information about the current market were later collected and analyzed.

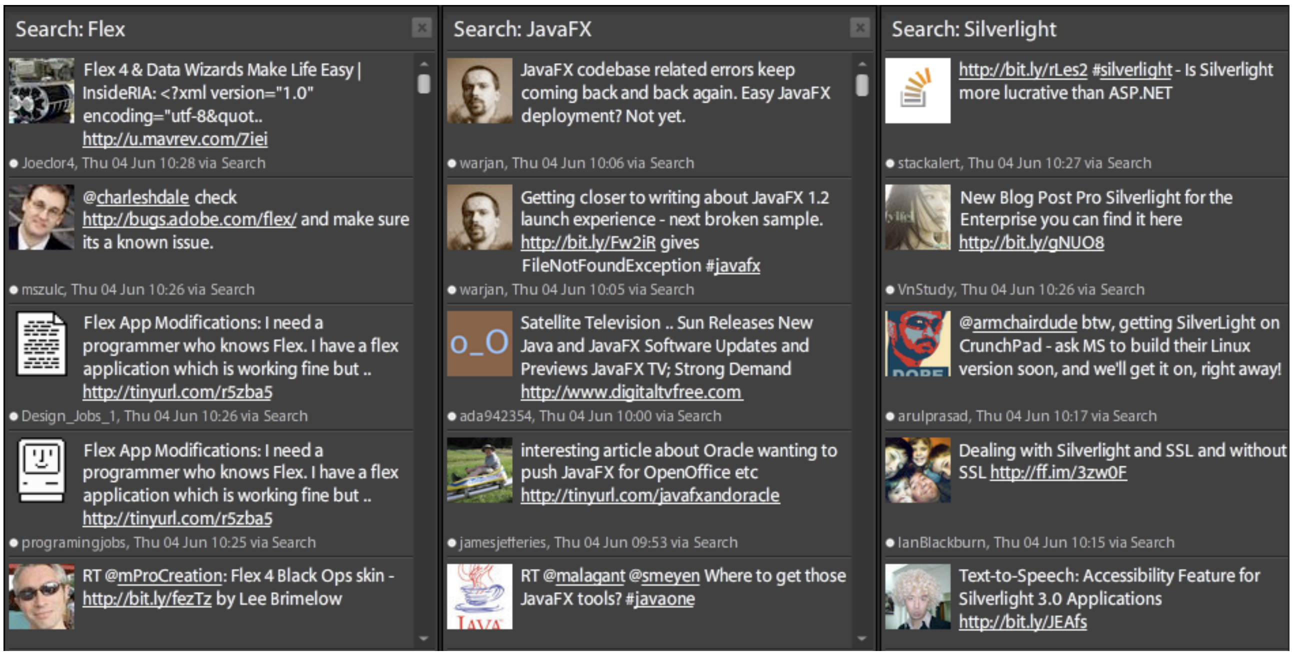
The methodology of the framework comparison was divided into the following two sub-tasks:

### 2.1.1. Identifying Sources

Since the topic of RIAs at the time of writing experiences such a rapid change concerning both its definition and existing RIA solutions, most of this work has been conducted through information gathering from the Internet. Resources have mostly been identified by web search (e.g. Google), but the information for the comparison was mainly gathered from each frameworks' website. Conferences such as Sun's JavaOne, Adobe's MAX and Microsoft's MIX and their later released webcasts constitute a major source of information. White papers written by analysts at market research companies like Forrester and Gartner have given hints on trends and where the business is heading. Capgemini also has its own internal collaboration tools and forums, tools that were scarcely used, but to some extent. An unstructured and informal interview with a Capgemini employee, with special expertise of Silverlight, was also conducted.

When it comes to the latest findings and speculations etc., weblogs (blogs) have played a great part. Even social networks like Twitter have been monitored by tags using an Adobe AIR application called TweetDeck (see the example below in Figure 1), and webcasts and podcasts have also been downloaded. Subscriptions to keyword alerts at Forrester's and Gartner's websites were also set up, which sends notifications as soon as something on what is being monitored gets published. Each of the companies Adobe, Sun and Microsoft have employed their own technical evangelists, whose jobs are to promote their products by giving talks, writing blogs and showing demos. Although this source of information is not particularly objective, it has been used to some extent to get questions answered directly through e-mail conversations.

To get an idea about the current market, different kinds of information sources were used. The online book store Amazon.com was used for looking into the amount of existing literature, job search websites were used for determining the extent to which the different frameworks were asked for in job postings. Even forums were used, to map out the number of discussions for each framework, and hence get an overall idea about their relative popularity within developer communities. Forums have also been used as a way to gain knowledge in specific details by asking questions, not found elsewhere in documentation, regarding the development of the prototype.



Search: Flex

Flex 4 & Data Wizards Make Life Easy | InsideRIA: <?xml version="1.0" encoding="utf-8&quot.. http://u.mavrev.com/7iei
● Joeclor4, Thu 04 Jun 10:28 via Search

@charleshdale check http://bugs.adobe.com/flex/ and make sure its a known issue.
● mszulc, Thu 04 Jun 10:26 via Search

Flex App Modifications: I need a programmer who knows Flex. I have a flex application which is working fine but .. http://tinyurl.com/r5zba5
● Design_Jobs_1, Thu 04 Jun 10:26 via Search

Flex App Modifications: I need a programmer who knows Flex. I have a flex application which is working fine but .. http://tinyurl.com/r5zba5
● programingjobs, Thu 04 Jun 10:25 via Search

RT @mProCreation: Flex 4 Black Ops skin - http://bit.ly/fezTz by Lee Brimelow

Search: JavaFX

JavaFX codebase related errors keep coming back and back again. Easy JavaFX deployment? Not yet.
● warjan, Thu 04 Jun 10:06 via Search

Getting closer to writing about JavaFX 1.2 launch experience - next broken sample. http://bit.ly/Fw2iR gives FileNotFoundException #javafx
● warjan, Thu 04 Jun 10:05 via Search

Satellite Television .. Sun Releases New Java and JavaFX Software Updates and Previews JavaFX TV; Strong Demand http://www.digitaltvfree.com
● ada942354, Thu 04 Jun 10:00 via Search

interesting article about Oracle wanting to push JavaFX for OpenOffice etc http://tinyurl.com/javafxandoracle
● jamesjefferies, Thu 04 Jun 09:53 via Search

RT @malagant @smeyen Where to get those JavaFX tools? #javaone

Search: Silverlight

http://bit.ly/rLes2 #silverlight - Is Silverlight more lucrative than ASP.NET
● stackalert, Thu 04 Jun 10:27 via Search

New Blog Post Pro Silverlight for the Enterprise you can find it here http://bit.ly/gNUO8
● VnStudy, Thu 04 Jun 10:26 via Search

@armchairdude btw, getting SilverLight on CrunchPad - ask MS to build their Linux version soon, and we'll get it on, right away!
● arulprasad, Thu 04 Jun 10:17 via Search

Dealing with Silverlight and SSL and without SSL http://ff.im/3zw0F
● IanBlackburn, Thu 04 Jun 10:15 via Search

Text-to-Speech: Accessibility Feature for Silverlight 3.0 Applications http://bit.ly/JEAfs

**Figure 1.** *Twitter keyword monitoring using an Adobe AIR application called TweetDeck.*

### 2.1.2. Analysis

The first part of the analysis was to identify what attributes to compare with the chosen RIA frameworks, i.e. the distinguished characteristics and features of such applications. This was achieved by an analysis of each frameworks' technical specification, taken both from the websites and sometimes also confirmed by e-mail conversations with technical staff from each company.

After having identified all interesting characteristics, each frameworks' corresponding attributes were mapped down into a table in order to provide a good overview for comparison. Different tables were used for different types of characteristics: technical, non-technical and current market/statistics. Existing solutions developed with each framework have also been evaluated to some extent.

Basically, the analysis meant going through all the identified sources for useful information. Features, white papers, blogs and statistics were then analyzed, and finally a comparison followed up by a discussion was performed.

## 2.2. Prototype Implementation

The applicability of JavaFX as a front-end system was evaluated in more detail than Flex and Silverlight. The reason for this was because of a focus on Java technologies at Capgemini in Gothenburg, and the fact that JavaFX at the time of writing was a rather new and unknown framework. The possible use of JavaFX as a front-end, and the interoperability between these two techniques in order to reuse existing Java knowledge, was of high interest to study.

The purpose of the prototype implementation was to create a proof of concept application in JavaFX and to clarify the strengths and weaknesses that were identified in the theoretical part of the study, and also to demonstrate its actual applicability and maturity. This prototype was, in agreement with Capgemini, based upon one of their existing projects. This particular project consisted of a Flex application, and this application naturally became a way of comparing Flex and JavaFX.

By first receiving a briefing on the existing Flex application, an overview of the architecture and technical details was established. From this introduction, a requirement specification was created and certain functionality, irrelevant for the proof of concept prototype, was delimited. The development process was performed using an agile approach, due to the progressive learning of JavaFX.

The choice of this particular application was made due to its applicability to uncover interesting areas of a RIA framework. The prototype makes use of graphics such as transformations and effects, as well as networking and XML parsing. Furthermore, it shows how an integration with Java can be achieved.

# 3. RIA Frameworks

This section will first explain what Rich Internet Applications mean in more detail, and look at specific techniques other than the chosen frameworks. Secondly, each of the frameworks Flex, JavaFX and Silverlight, and their related technologies, will be described. The actual comparison between the frameworks can be found in the analysis in section 4.

## 3.1. Rich Internet Applications

Rich Internet Applications (RIAs), is an umbrella term used for describing applications that are more desktop-like, with "richer" content and more interactivity. According to a white paper from Gartner Inc., RIA frameworks can be divided into two categories: JavaScript/AJAX-based frameworks and plugin-based frameworks. Frameworks based on JavaScript are browser-based and tend to be more lightweight, while on the contrary plugin-based are more heavy-weight with a bigger download footprint. (Valdes, 2009)

Interactivity, responsiveness and richness are three general characteristics of RIAs. They often offer support for validation and error-handling, drag-and-drop functionality and richer controls like calendars and sortable lists etc. With the help of AJAX, web applications do not have to reload the whole web page but can instead give faster feedback to the user by reloading parts of it, e.g. with a real-time search and filtering feature. These aspects make an application behave more like a desktop application, and moreover, RIAs often contain animations and media content. All of these aspects usually make applications provide a better user experience and can, if implemented wisely, reduce the complexity for the end user. (Zetie, 2005)

AJAX, which stands for Asynchronous JavaScript and XML, is a term coined by Jesse J. Garrett in 2005 when he tried to describe different techniques that enabled more powerful websites. AJAX is not a technology, but rather a term to describe a set of functionality, the biggest being asynchronous data retrieval using a XMLHttpRequest-object in JavaScript. The definition of AJAX also consists of a data format like XML or JSON for communication between the browser and the server, HTML and CSS for describing the interface, and finally JavaScript to manipulate it and add interactivity (Garrett, 2005).

### 3.1.1. JavaScript/AJAX-based Frameworks

To develop rich websites with JavaScript and AJAX from scratch can be tedious and complex, since different browsers interpret JavaScript differently and there are no native UI controls. A solution to this is to use frameworks that eliminate the need for cross-browser coding, and also provide ready-to-use controls. Another strength of using these frameworks is that there is no need for a plugin to be installed in the browser, and it is independent of server-side platforms. Frameworks in this category allow you to code in either JavaScript or a JavaScript subset, or in another language that gets compiled into JavaScript. Knipp & Valdes (2009), found out that 80% of the top 50 public websites use AJAX, and it could therefore be considered very established.

Below is a list of some of the most popular JavaScript and AJAX frameworks/libraries, both commercial, community-based and open-source:

| JavaScript/AJAX Framework | Comment |
|---|---|
| Cappucino Web Framework | Open-source framework where applications are created using a new language, Objective-J. |
| Dojo Toolkit | This open-source framework is sponsored by IBM, Oracle and other companies. |
| Ext JS | Contains high-quality widgets and has got a good reputation within the enterprise sector. |
| Google Web Toolkit (GWT) | Applications are written in Java, which are later cross-compiled into optimized and browser-specific JavaScript. |
| GWT-Ext & Ext GWT | UI widget libraries to GWT and Ext JS, respectively. |
| jQuery | A very popular community-based, open-source and lightweight JavaScript/AJAX framework. Industry support from Microsoft and Nokia. |
| OpenLaszlo | Commercial RIA framework, where applications are created using its own XML-like programming language. |
| Prototype and script.aculo.us | One of the first open-source JavaScript/AJAX frameworks, still popular. Often used together with script.aculo.us that adds more interface functionality. Used by Apple, CNN and Nasa. |
| SproutCore | A rather new JavaScript framework, supported by Apple. |
| Yahoo! User Interface (YUI) | Open-source framework with utilities and many UI components. |

**Table 1.** *Popular JavaScript and AJAX frameworks/libraries. (Valdes, 2008)*

Examples of famous JavaScript/AJAX implementations are Google Maps (uses AJAX to asynchronously load images while panning), Google Suggest (uses AJAX to fetch matching search queries based upon what is partially written) and Facebook (makes heavy use of AJAX using their own framework).

JavaScript/AJAX-based frameworks might be a better choice if bandwidth is the most important aspect of a project, since they in general are much smaller than RIAs made with a heavier plugin-based framework. JavaScript/AJAX-based frameworks will also become even more powerful with the standardization of HTML 5, the next generation of web standards, described in the next section.

### 3.1.2. HTML 5

HTML 5, which is currently under development, will be the next standard markup language for web pages. According to the specification, HTML 5 will be covering most of the differences between JavaScript/AJAX-based- and plugin-based RIA frameworks. (W3C, 2009)

A first draft was released in January 2008, and some web browsers like e.g. Firefox, have already implemented parts of the specified functionality like the `<canvas>`, `<audio>` and `<video>` tag, local storage and background processes. At the time of writing, Microsoft has not yet announced to what extent their web browser Internet Explorer is going to support HTML 5. Except for the features mentioned above, HTML 5 will also contain more controls than the usual `<input type="button">` and `<textarea>` etc., in order to make the language richer. A data grid and a progress bar are two examples of these new controls. The `<canvas>` tag allows the user to make drawings directly in the browser and to display interactive graphics like charts. It will also become possible to discover online and offline connectivity, and therefore, together with local storage, the user will be able to surf a website even when not being connected to the Internet (W3C, 2009)

Behind the development of the specification are representatives from both Google and Apple, and the goal of HTML 5 is to move the Web away from proprietary technologies such as plugin-based RIA frameworks. HTML 5 will definitely be able to compete with Flex, Silverlight and JavaFX, and could make these plugins appear unnecessary. It will take time for HTML 5 to get established and penetrate the market though, and at the time of writing, experts at both Microsoft and Adobe think that a major breakthrough lies 5-10 years into the future. (Krill, 2009)

### 3.1.3. Plugin-based RIA frameworks

JavaScript- and AJAX-based web applications cannot leverage the same power as plugin-based frameworks, for example like direct manipulation of graphics, dealing with vector graphics and performing multi-threaded computations. The most distinguished characteristics of plugin-based RIAs are applications with extensive support for media (audio/video/etc.), highly interactive, cross-platform, cross-device, cross-browser and desktop-like with offline and out-of-browser support. A weakness of applications developed with this type of frameworks is their bigger footprint due to the larger size of compiled applications, which leads to a longer waiting period on the initial download of the application (Knipp & Valdes, 2009).

Except for the biggest plugin-based frameworks Adobe Flex, JavaFX and Microsoft Silverlight, there is also a framework called Curl that falls into this category. Curl was initially founded in the US in 1998, but later acquired by a Japanese company. It is a framework with more enterprise focus and has been on the market since 2002. Among other features, it has out-of-browser- and offline support, and it is being used in around 400 enterprises (mostly in Japan). (Valdes, 2008)

The three other frameworks that this study will focus on, Adobe Flex, JavaFX and Microsoft Silverlight, will be described in detail in the following sections.

## 3.2. Adobe Flex

Adobe Flex was the first of the three chosen RIA frameworks to be released. It has been on the market since 2004, when it was released by Macromedia. Adobe Systems acquired Macromedia after less than one year, and has been continuing the development of Flex with great success, making it the most popular RIA framework today. At the time of writing, the current stable version of Adobe Flex is Flex 3, but Flex 4 (beta) has been previewed and that is the version that this report will focus on. Flex 4 Beta, with the code name Gumbo, was released in June 2009, and according to Adobe, a stable version will be released in the beginning of 2010 (Adobe: Gumbo, 2009). The framework itself and the SDK are released open-source, licensed under the Mozilla Public License, but the associated tools are proprietary.

In Flex 4, Adobe has put most focus on a theme referred to as *Design in Mind*. The highest priority of Design in Mind has been to make the look and feel of applications easy to customize and skin. Through Adobe's research, they found out that applications tend to look similar using the built-in theme, and the customers want to avoid this by instead designing their own unique look that conforms to their own graphical profile. To cater for this requirement, Flex 4 provides a component and skinning architecture that enables support for powerful design tools. This is also reflected in MXML, which is Adobe's user interface markup language. The MXML format has been adjusted to enable tools to better describe features such as states, transitions and effects. (Adobe: Gumbo, 2009)

With the fourth version of Flex, a new designer tool called Flash Catalyst has been released, where designers can create user interfaces without having to write any code. The tool could be seen as the key to the designer-developer workflow, which closes the gap between the design tool Creative Suite and the developer tool Flash Builder, and therefore also between designers and developers. Along with these updates, a new format has been introduced called FXG (Flex Graphics), that is an XML-based graphics interchange format for the Flash platform. Except for meta information, the contents of an FXG-file is pure MXML, and could therefore be included directly in the source code. The FXG format links skinning features and designer tools together, and could also describe transformations and filters, in addition to shapes, text and images. (Adobe: Gumbo Themes, 2009)

Flex applications run using the Flash Player runtime, and since ActionScript is the programming language for Flash, it is also used for Flex. ActionScript 3.0 is the latest version, which is an object-oriented language, requiring Flash Player 9 or higher. Apart from ActionScript, Flex applications can also be written in MXML that is quite similar to HTML. MXML is a declarative XML-based markup language for describing user interfaces and laying out components, but can also be used for implementing non-visual features. It was introduced to simplify development of user interfaces and cater for better tooling support. When deploying a Flex application, the MXML is parsed and compiled into ActionScript, and the ActionScript is compiled into a SWF-file that runs in the Adobe Flash Player. (Adobe: Gumbo, 2009)

Below is an example of a Flex application with the minimum amount of code:

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
    <mx:Button label="I'm a button!" />
</mx:Application>
```

Search engine indexing of Flash content has previously been a major issue, but today, it is possible for search engines like Google to index all kinds of SWF-files through sophisticated algorithms. Google has solved the problem by using a crawler that acts like a user visiting the website, i.e. clicking buttons, entering input etc. In June 2009, Google announced enhanced indexing, that enables external resources like XML and HTML files that are loaded into the SWF, to be indexed as well. (Google: Flash Indexing, 2009 & Google: Improved Flash Indexing, 2008)

The product family of Adobe is extensive, and several technologies and applications are used in conjunction with Flex. The most relevant in this context are presented below:

### 3.2.1. Adobe Flash

Adobe Flash is one of the most popular techniques for presenting rich data and media on the Web today. Its history stretches back as far as 1996, when it was introduced by Macromedia. It is important to realize the importance of Flash, since Flex applications actually compiles into ActionScript, which in turn compiles into a SWF file (pronounced *swiff*). A SWF file contains the content and functionality of a Flash/Flex application, and it is this file that runs inside the Flash Player runtime.

In comparison to Flash, Flex should be seen more like a framework and an environment including tools, than enables the creation of larger and more complex Flash applications. Except for the animations etc., that Flash initially was built for, Flex offers integration with server-side applications, databases, advanced UI components and MXML. In order to make use of the majority of the features in Flex, including ActionScript 3.0, the applications require Adobe Flash Player 9 to be installed. Flash 10 is required for the latest features such as GPU acceleration and 3D effects (Adobe: Flash, 2009).

There is also a lightweight and optimized version of the Flash Player that is called Adobe Flash Lite, which purpose is to bring Flash applications to mobile devices, Internet-connected TVs and other consumer electronic devices. However, Flash Lite is not able to run neither Flex applications nor ActionScript 3.0, so Flex is currently not available for mobile devices. (Adobe: Flash, 2009)

### 3.2.2. Adobe AIR

Adobe AIR stands for Adobe Integrated Runtime and is a proprietary runtime that makes it possible to run Flex applications on the desktop and offline. Not only does the runtime support Flex and Flash, but also HTML and JavaScript/AJAX. The current stable release (1.5) does not provide support for mobile devices, but according to Adobe, there are plans for this in future releases. AIR is available on Microsoft Windows and Mac OS X, and at the time of writing, a public beta is available for Linux. Since AIR is a native application runtime and not a browser plugin, it allows access to file system resources, and it also contains a built-in database based on SQLite. (Adobe: AIR FAQ, 2009)

### 3.2.3. Tools

#### Adobe Flash Catalyst

Adobe Flash Catalyst is a new design tool for creating user interfaces and prototypes with interactive content, without having to write any code. Flash Catalyst was released as a beta version in early 2009. The purpose is to serve as an environment for designers and RIA architects, who later on can hand over their projects to Flex developers that will add functionality with Flash Builder. This closes the gap between designers and developers, giving designers the ability to quickly create fully functional prototypes. Flash Catalyst is easy for designers to learn, since the environment is similar to the one in Creative Suite, like Photoshop or Illustrator. Artwork and layers from e.g. Photoshop projects are preserved and can be turned into interactive components like buttons, scrollbars etc. when imported into Flash Catalyst. Furthermore, the application contains a timeline, on which the states and transitions of the application are mapped onto. Under the hood, a Catalyst project is described with MXML-code, making interchange between applications easy. A Catalyst project can hence seamlessly be edited and further developed in Flash Builder. (Adobe: Flash Catalyst, 2009)

#### Adobe Flash Builder

At the time of writing, the stable version of this tool is Adobe Flex Builder 3, but it will change name to Adobe Flash Builder 4 in the next version, that at the time of writing has been previewed. Flash Builder is a commercial, non open-source, IDE for developing Flex applications, built upon the Eclipse platform. It is, in contrast to other Flex tools, aimed at developers and not designers. With Flash Builder and the designer-developer workflow, developers can easily continue where designers ended, and add functionality to the graphical elements exported from either Creative Suite or Flash Catalyst. (Adobe: Flash Builder, 2009)

In the development of Flash Builder 4, data-centric development has been of high importance. Whether a service operation is to be reached through e.g. REST, SOAP or BlazeDS (described below), it is easily bound to Flex components in the Flash Builder by drag-and-drop. The data grid component, which is often used to display collections of data fetched from other services, comes with features like paging and sorting that works out of the box. (Adobe: Flash Builder - What's New, 2009)

For the purpose of testing, Flash Builder comes with two features called network monitor and FlexUnit. The network monitor allows the developer to see the requests/replies and data sent between the client and server, in order to debug applications and measure performance. FlexUnit is Adobe's unit testing framework for Flex and ActionScript 3.0 applications, which offers full testing capabilities similar to JUnit used for Java. (Adobe: Flash Builder - What's New, 2009)

### 3.2.4. Other Flex-related Technologies and Applications

#### BlazeDS

BlazeDS is a Java remoting and web messaging technology for exchanging data between client and server. Flex applications are easily integrated with a Java back-end by utilizing BlazeDS, because they can directly invoke methods of Java objects deployed at the server side. This makes it easy to mix existing Java code bases with front-ends developed using Flex. BlazeDS features real-time push functionality, which means that the server can "push" data to the client over HTTP, without the client having to repeatedly poll for it which makes applications much more responsive. The technology was previously a part of Adobe LiveCycle, but became separate and licensed as open-source (LGPL v3) in 2007, as a contribution to the developer community. (Adobe: BlazeDS, 2009)

**Adobe LiveCycle ES (Enterprise Suite)**

Adobe LiveCycle is a commercial server solution with the purpose of automating business processes like filling in online forms, capturing data, document output etc. and more or less replace the manual workflow needed for this kind of activities. This is achieved in an interactive manner by a blend of PDF and Flex/Flash technologies, creating visually appealing and process-oriented user interfaces that can run either online or offline (Adobe: LiveCycle, 2009). An example of a LiveCycle application, demonstrated by Creanga (2009) at the Javaforum Conference in Gothenburg, was an interactive web form, running in Flash inside an ordinary PDF document. This form was connected to a back-end for fetching data and validating user input with the server, including conflict management in case that the data was being edited by another client at the same time.

### 3.2.5. History Timeline

The following figure illustrates important milestones for Adobe Flex:

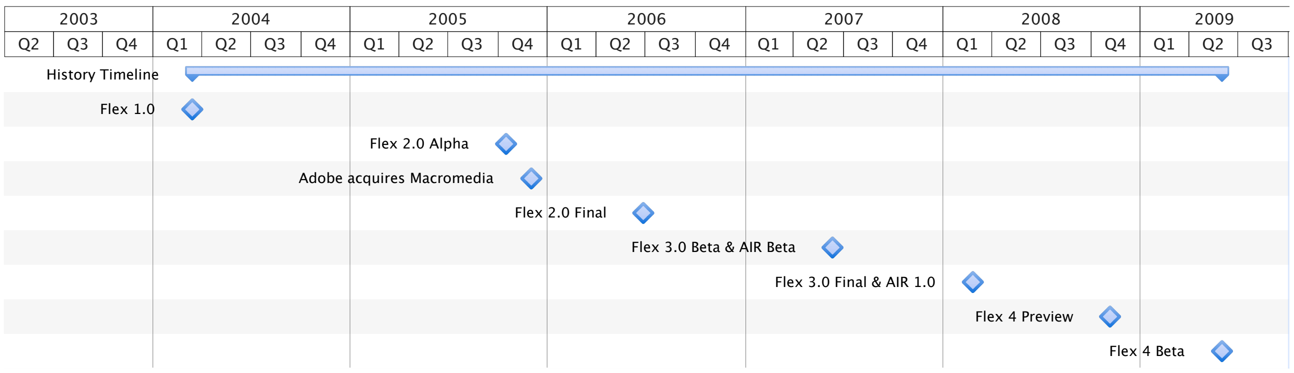


**Figure 2.** *Important milestones for Adobe Flex.*

**March 2004**

Macromedia releases Flex 1.0, the initial version.

**October 2005**

Flex 2.0 (Alpha) released.

**3 December 2005**

Adobe Systems acquires its former competitor Macromedia and hence both Flash and Flex.

**28 June 2006**

Flex 2.0 (Final) released.

### 11 June 2007

Flex 3.0 Beta 1 (codename Moxie) released, including integration with Adobe's Creative Suite and AIR Beta released.

### 25 February 2008

Flex 3.0 (Final) released and Flex 3 SDK is announced to become open-source, while Flash and Flex Builder remains proprietary. Adobe AIR 1.0 is also released.

### 5 November 2008

Preview of Flex 4 during Adobe MAX 2008, codename Gumbo.

### 1 June 2009

Flex 4 Beta released.

### Spring 2010

A stable version of Flex 4 is supposed to be released.

## 3.3. JavaFX

JavaFX is a framework for creating Rich Internet Applications, developed by Sun Microsystems and first released in December 2008. Sun's description of the framework is cited below:

*"JavaFX is an expressive rich client platform for creating and delivering rich Internet experiences across all the screens of your life."* (Sun Microsystems: JavaFX FAQ, 2009)

At the time of writing, the latest version of JavaFX is 1.2, released as stable the 2nd of June 2009. JavaFX is divided into the three different editions: JavaFX Desktop, JavaFX Mobile and JavaFX TV. JavaFX Desktop is often referred to as simply JavaFX. The concept of profiles is at the heart of the JavaFX multi-platform usage. The developer can choose to use libraries for either the common, the desktop or the mobile profile. The common profile contains libraries that are shared between desktop and mobile, and can hence be deployed to both platforms without any further modification. By making use of either the desktop or the mobile profile, more device-specific features can be accessed. JavaFX TV, on the other hand, does not have a profile at the time of writing, but is supposed to be released later in 2009, although it was previewed during the JavaOne Conference in June 2009.

An architectural overview of the components of the JavaFX platform is presented below:

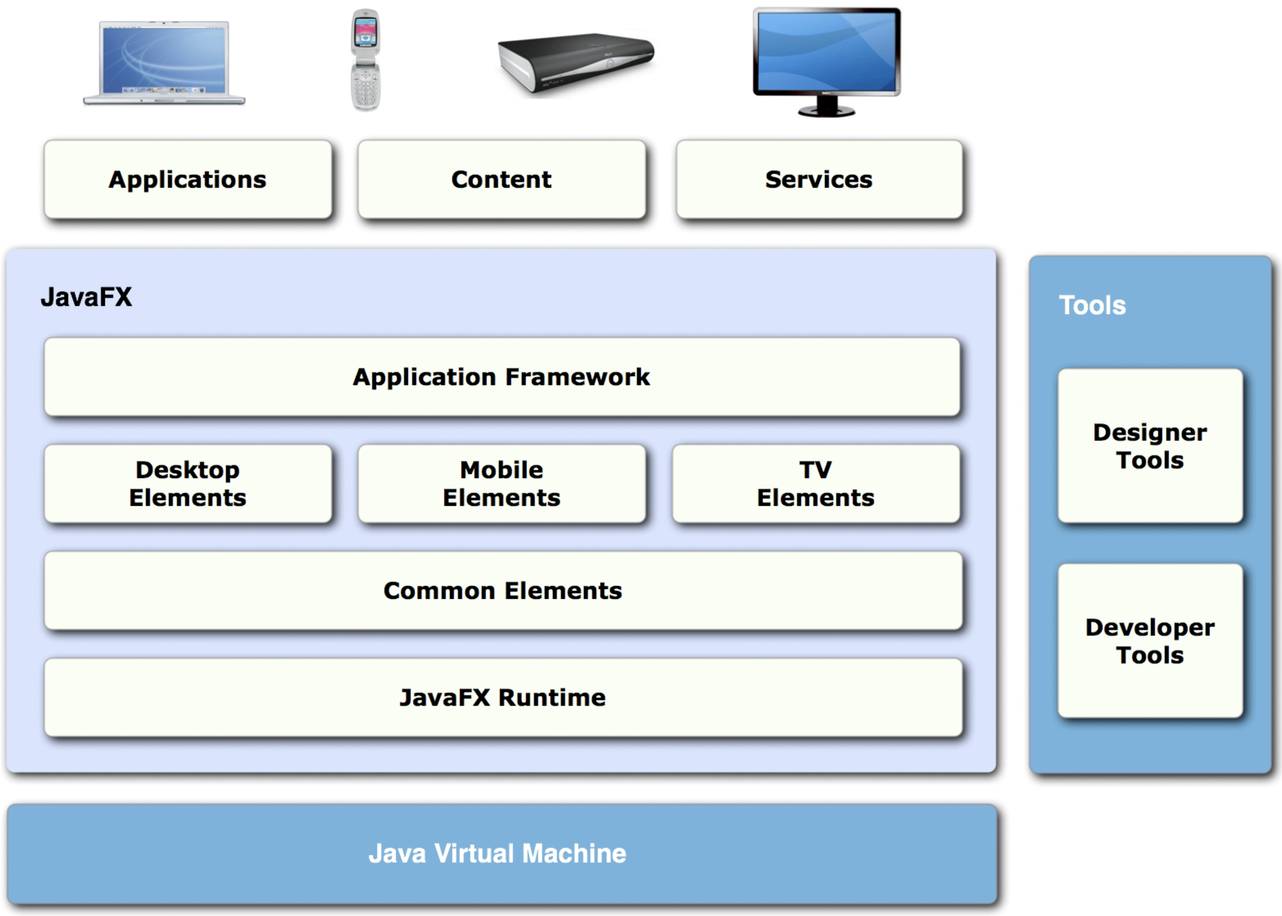


**Figure 3.** *JavaFX platform architecture.*

The purpose of JavaFX is to create graphical applications, or "front-ends", with rich interactive content. The framework is shipped with UI components like sliders, listviews and charts, even though it does not have as many as Flex and Silverlight. These components are skinnable, so that the visual appearance can be changed to better fit together with the other graphics of the application. Skinning can also be made by using the well established style sheet language CSS. Furthermore, animations, transformations and effects are easily applied to any graphical component. To handle the layout of the components, six different layout classes exist. (Sun Microsystems: JavaFX FAQ, 2009)

For media support, Sun has partnered with On2, a company providing a codec called On2 Video VP6 which is included in JavaFX. JavaFX has its own cross-platform media format called FXM. By using On2's Flix software, most of the common media formats can be converted into FXM and be played wherever JavaFX is deployed. The Real Time Streaming Protocol (RTSP) is used for streaming video. (Heiss, 2008) and (Sun Microsystems: JavaFX FAQ, 2009)

JavaFX has also been designed with the Internet and general network connectivity in mind. Several ways of talking with other servers and services exist. One of the popular methods of achieving this today is by the means of web services. A web service is defined by W3C as "a software system designed to support interoperable machine-to-machine interaction over a network" (Haas, 2004). Another common term used in conjunction with web services is Representational State Transfer (REST). RESTful services often refers to RESTful web services, and is a technique to exchange information by conforming to a specific network architecture, which JavaFX supports. The data format that RESTful services speaks is generally XML or JSON, which JavaFX Script provides parsers to, and the communication is done via HTTP by POST, GET, PUT and DELETE methods. JavaFX also includes support for reading RSS and ATOM web feeds.

JavaFX can seamlessly make use of existing Java libraries by simply importing them and then use them as they would be used in a plain Java application. Furthermore, JavaFX applications run on top of the Java runtime (JRE), so JavaFX is fully integrated with the ordinary runtime. Version 1.5 of the JRE, or higher, is required to run JavaFX applications. The first time a user runs a JavaFX Desktop application, a small add-on to the existing runtime will be downloaded but apart from that, it runs upon the standard JRE.

On desktop computers, applications can either run inside or outside the browser. To install a JavaFX application locally, the user can simply use the drag-to-install feature by dragging it from the browser and dropping it onto the desktop, if Java 1.6 update 10 is installed. If not, the user can choose to save it to the desktop. In both cases, it becomes deployed to the local computer, and since it can run outside the browser, it can also run offline when there is no Internet connection.

The JavaFX compiler, the scene graph and the JavaFX tools (NetBeans plugin and Eclipse plugin) are open-source, licensed under the GPL 2.0 open-source license. Sun is planning to release the runtime and the JavaFX associated file formats as open-source too, but at the time of writing, the runtime core is proprietary. (Kaul, 2008)

### 3.3.1. JavaFX Script

Together with the release of the first version of JavaFX (1.0), the new language JavaFX Script was born. JavaFX Script is statically typed and both declarative, procedural and object-oriented. The code is compiled into regular Java bytecode, which means seamless integration with Java and also that JavaFX can take full advantage of other properties of Java, e.g. the virtual machine's proven security model etc.

JavaFX Script makes it easy to code in a visual context, due to its declarative nature. This means new possibilities for designers, who sometimes might need to edit the code, and also opens up more possibilities for better design tools. Declarative programming describes *what* to do, rather than *how* to compute it (Coenen, 1999), and this is why the code becomes more intuitive for non-developers to understand. Content authors as well as designers are able to feel familiarity with the way the code is organized due to its format, that closely matches the layout of GUIs, and it enables any user to easier visualize the GUI that the code will generate. According to Sun, the language is primarily designed for web developers and technical designers. (Sun Microsystems: JavaFX FAQ, 2009)

Below is an example of a JavaFX application with the minimum amount of code:

```
Stage {
    scene: Scene {
        content: Button {
            text: "I'm a button!"
        }
    }
}
```

This exemplifies the foundations of the graphical layouting in JavaFX, which is quite different from Java's way of defining user interfaces. The `Stage` is the top container, which is equivalent to Java's `Frame`, but can also be made transparent or without the standard toolbar etc. Then there is the `Scene`, which is a canvas in which all the `Nodes` are rendered. The `Scene` can either contain a sequence of `Nodes`, or it can contain a `Group`, which itself is a `Node`. Components and controls are all `Nodes`, and effects applied to a `Group` of nodes affects all the children, which is why this whole concept, known as the scene graph, is so useful.

One of the most powerful features of the language is binding. Binding enables a variable to be bound to other variables, making it synchronized. This eliminates the need of using the listener design pattern to keep application data consistent. This relationship is achieved using the `bind` keyword. Since everything in JavaFX is expressions, like functions, conditions, loops and blocks, binding can be applied on anything, and can be either unidirectional or two-way (bidirectional). As an example of how binding works, have a look at the following code:

```
def x = bind if (condExpr) expr1 else expr2
```

This single line of code results in `x` automatically getting updated to become `expr2`, as soon as some part of the expression `condExpr` changes so that the expression becomes false. Since binding requires the system to recalculate binding expressions and keep track of all the changes automatically, complex dependencies should not be overused due to performance reasons.

Another useful feature is the handling of events. As opposed to Java, the listener pattern does not have to be used in order to make certain actions when events occur. Instead, JavaFX Script has an event handling syntax similar to the one used in JavaScript. This is best described with an example:

```
var myButton = Button {
    onMousePressed: function(event) {
        // action to perform when mouse is pressed
    }
}
```

In this simple example, `onMousePressed()` will be called as soon as `myButton` has been pressed.

List comprehensions, i.e. to create lists based on other lists, is another useful feature of JavaFX Script. It can also be found in many other programming languages and has especially been influenced by functional languages like Haskell. This is an example of how to retrieve a list of all the numbers between 1 and 10 who's squares are less than 20 with this technique:

```
var nums = [1..10]
var result = nums[x | x*x < 20]
```

This expression will result in `[1,2,3,4]`.

Furthermore, there are other interesting features of the language such as triggers, which are functions that get invoked at certain events or changes to variables or objects' state. There is also the concept of time, whereby `3s` really means 3 seconds and can be treated accordingly at compile time, for example to describe transitions and the animation of key frames.

### 3.3.2. JavaFX Mobile

JavaFX Mobile was released during the Mobile World Congress in Barcelona, February 2009. It runs upon Java ME (Micro Edition), and can hence leverage the power of Java ME like accessing the file system, GPS and camera etc. This edition is designed to run on any mobile device ranging from low-end mobiles to high-end smartphones, including the Android, Blackberry and Windows Mobile platforms. (Sun Microsystems: JavaFX FAQ, 2009)

Developers can run and test mobile applications through the mobile emulator, provided by the SDK, which currently only supports Microsoft Windows. They can also easily deploy them to mobile devices through the SDK, and designers are able to see how their design will look in mobiles by using JavaFX Production Suite. One thing to notice is when using the mobile profile (common profile) in NetBeans, `javafx.ext.swing` package disappears, which is because JavaFX Mobile does not support Swing. It is the same for other packages like `javafx.scene.effect` and `javafx.reflect` etc.

### 3.3.3. JavaFX TV

JavaFX TV was previewed during the JavaOne conference the 2nd of June 2009, and was claimed to be released later on in 2009. With JavaFX TV, it will be possible to develop applications that interact with a television remote control, and interfaces to be placed on top of video, making the experience more customizable and interactive. According to Sun, the platform will be available on televisions, set-top boxes and gaming consoles. Since Blu-ray players has native Java support, JavaFX TV will be able to run there as well. (Sun Microsystems: JavaFX FAQ, 2009)

### 3.3.4. Tools

#### NetBeans IDE

NetBeans is the development environment that Sun recommends for developing JavaFX applications. The IDE has support for building, debugging and previewing applications, and the user can choose to deploy to either desktop, browser or mobile. A palette is used to view the available UI controls, which are added by drag-and-drop. However, it does not contain a visual designer, but the definitions of these controls are rather added straight into the code. Functionality for handling transformations, effects and animations is also built into the IDE in a similar fashion. (Sun Microsystems: JavaFX FAQ, 2009)

#### Eclipse IDE

There is an official open-source plugin for JavaFX on the Eclipse platform, supporting version 3.4 of the IDE. At the time of writing, this plugin is still under development and several bugs are yet to be corrected. Just like in NetBeans, the environment provides support for adding UI controls to the code by drag-and-drop and apply transformations and effects etc. (Sun Microsystems: JavaFX FAQ, 2009)

#### JavaFX Production Suite

The JavaFX framework offers a designer-developer workflow, where the designer and developer can work together more efficiently. Through JavaFX Production Suite, designers are able to create their graphics in Adobe Creative Suite, and then export them into a JavaFX format which the developers can import into the application. This allows designers and developers to collaborate more effectively.

However, the JavaFX Production Suite is not a visual designer, but basically only a plugin for Adobe CS (Creative Suite) version 3 and 4. It is an export tool that can export graphics, both images and animations, from either Adobe Photoshop or Adobe Illustrator, to the JavaFX format. This format is a compressed format referred to as JavaFX Content File (FXZ extension), which in turn contains a JavaFX Data File (FXD extension). The FXD file is a textual description of the graphics with JavaFX Script code. In addition to the FXD file, the FXZ file may also contain images and fonts.

Objects and layers in e.g. Adobe Photoshop, that have a "jfx" name prefix are preserved in the FXD format, and become object names that the developer can refer to in order to use them in the code. This enables the designer to create design mockups early in the design process that can be used by the developer. Later graphical changes can easily be incorporated by the developer by simply replacing the FXZ file exported by the designer. Furthermore, designers are able to see how their design will look on different devices by viewing it in the JavaFX Media Factory.

The figure below illustrates a gameboard in Adobe Illustrator. Each tile in this image has a `jfx:` prefix, which means that they can be used as objects in JavaFX Script.

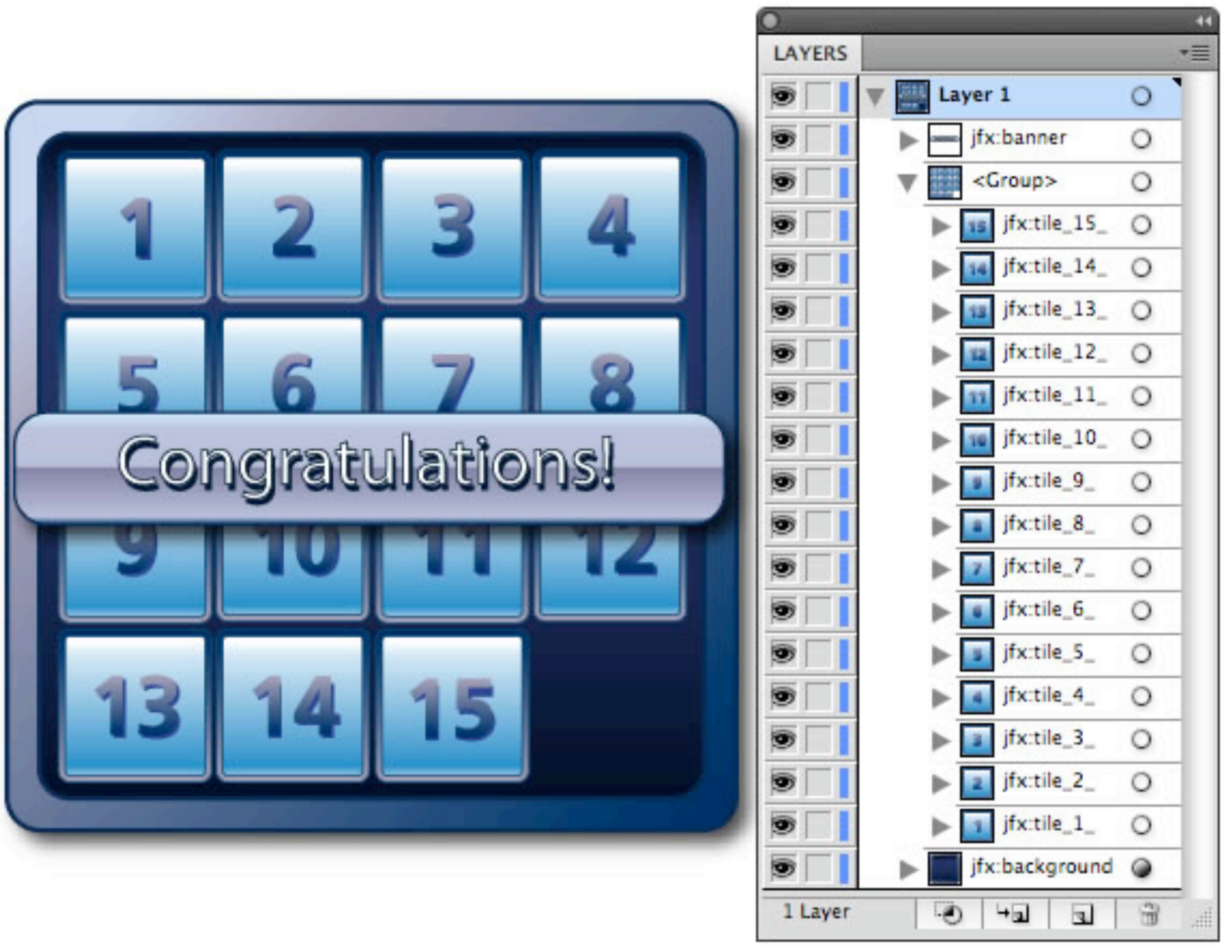


**Figure 4.** *Example of an image that can be exported from Adobe Illustrator. Nodes prefixed with* `jfx:` *can be used directly in the JavaFX Script code. (Sun Microsystems: JavaFX Production Suite, 2009)*

## JavaFX Authoring Tool

JavaFX Authoring Tool was previewed during the JavaOne conference in June, 2009. It is a collaborative tool that will be used by visual content developers to create cloud-based interactive applications. It contains a timeline editor for dealing with keyframes and animation sequences. It is a WYSIWYG application, with no compiling needed, and it deploys applications to both desktop, browser, mobile and TV set-top boxes. Another feature is visual wiring, which for example means that a button could be wired to the play/pause-property of a video, or a slider to be wired to the current time of a video. Much more details than these were not revealed at the preview session, but the tool is planned to be released in the end of 2009. (Sun Microsystems: JavaOne, 2009)

## Inkscape

Inkscape, an open-source vector graphics editor, has also got support for exporting JavaFX designs, similar to the plugins available for Adobe Creative Suite. (Neto, 2008)

### 3.3.5. History Timeline

The following figure illustrates important milestones for JavaFX:

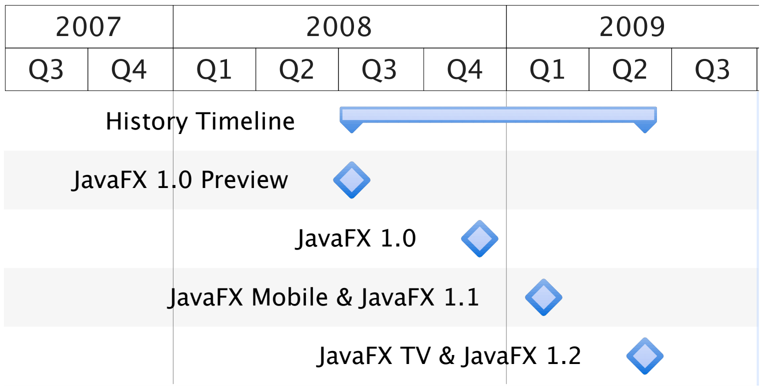


**Figure 5.** *Important milestones for JavaFX.*

**September 2005**

Sun Microsystems acquires a company called SeeBeyond Technology Corporation, where Chris Oliver had been working on a project called F3 which stands for "Form Follows Function". F3 later changed name to JavaFX Script.

**May 2007**

Sun Microsystems announces JavaFX during the JavaOne Conference.

**July 2008**

Sun Microsystems releases JavaFX 1.0 Preview.

**4 December 2008**

JavaFX Desktop (JavaFX 1.0 SDK), JavaFX Script 1.0 and JavaFX Production Suite released.

**12 February 2009**

JavaFX Mobile and JavaFX 1.1 SDK released.

**2 June 2009**

JavaFX 1.2 SDK, JavaFX Mobile 1.2 released. JavaFX Authoring Tool and JavaFX TV were also demoed, but not released to the public yet.

## 3.4. Microsoft Silverlight

Microsoft's framework for developing Rich Internet Applications is called Silverlight. At the time of writing, the latest stable release is Silverlight 3, released 9th of July 2009. The framework is considered to be the main competitor to Flash/Flex and has been on the market since December 2006, when the 1.0 beta was released.

The Silverlight runtime is basically a subset of Windows Presentation Foundation (WPF), which is Microsoft's standard model for developing user interfaces and a central part of the .NET Framework, built upon DirectX. Both Windows Vista and Windows 7 are built using WPF as the graphical foundation, which proves the strengths of this rendering model. The core functionality, and some newly added features, have been reduced into a plugin that can be installed on the client, and this is what is known as Silverlight. This means that it is a lighter and more portable version of WPF, designed to be distributable across the Web yet still bringing the power of its parent model. Silverlight applications can run either inside the browser or, since version 3, offline and outside of the browser, accessible from the start-menu or the desktop.

XAML is Microsoft's XML-based markup language for defining user interfaces, more or less similar to Adobe's MXML format. It is used throughout Microsoft's .NET Framework 3.X, including WPF and its subset Silverlight. XAML comprises more than only interface definitions and is also used for data binding and events, with the purpose of separating the presentation layer from the underlying logic. (Microsoft: XAML, 2009)

Below is an example of a Silverlight application with the minimum amount of code:

```
<Canvas xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">
    <Button Content="I'm a Button!" />
</Canvas>
```

One of the strengths of Silverlight is the ability to access the extensive library contained in the .NET Framework, as well as already existing code. To develop Silverlight applications, any .NET language can be used. Among plenty of other languages, this includes the most popular ones C# and VB.NET. As long as the functionality of existing code bases are confined within the scope of Silverlight (since it is a subset of WPF), code reuse is as simple as copy-paste. Coming from a .NET developer background, the only prerequisite to start with Silverlight is to know the WPF model and XAML, making it very easy to start developing straight away.

The most noticeable media feature in Silverlight 3 is support for smooth streaming for live and on-demand true HD video, up to 1080p resolution. Smooth streaming is a hybrid media delivery technique described as HTTP-based adaptive streaming, first demonstrated during the 2008 Beijing Summer Olympic Games. This is achieved through a combination of traditional streaming like RTSP which is stateful, and HTTP progressive download, where it is possible to seek in the video and request different parts of it, as well as it can be played while it is being downloaded. The combination of the two results in a technique that acts like streaming but is based on HTTP progressive download using ordinary HTTP GET requests. The data is downloaded in a series of small chunks, which are linearly re-assembled at the client side. Depending on available network resources and CPU usage, the client can determine the bit rate (chunk size), which leads to varied quality on the output while still delivering a smooth experience. The HTTP requests are on the form of a RESTful URL, where both the bit rate and requested fragment are supplied. Support for Smooth Streaming is included in Microsoft's web server IIS using the extension IIS Media Services. (Microsoft: IIS Smooth Streaming, 2009).

Silverlight features a media pipeline where several different types of formats and codecs can be decoded outside the runtime and then rendered in Silverlight. In addition, media playback is backed up by hardware GPU acceleration to support true HD video in fullscreen mode.

More graphics features include the ability to add transitions, animations and 3D effects, just like with Flex and JavaFX. Moreover, it contains functionality for both styling and theming. Styling in Silverlight is not done with CSS, but instead using XAML. This concept is called templates, and templates can be based upon each other, hence creating cascaded styles just like with CSS. It is possible to use different graphical styles and Silverlight is shipped with nine different themes. It is also possible to find third-party themes, either free or commercial.

To save data locally, Silverlight uses something called *Isolated Storage*, which could be described as a virtual file system, that allocates 1 MB per application. It is an alternative to using cookies, with the benefit of better performance and caching for larger data sets. However, the drawbacks are that the system administrator might block the particular folder from being writable, and the data is not 100% secured unless it is encrypted. (Snow, 2008)

### 3.4.1. .NET RIA Services

ASP.NET is Microsoft's widely used web application framework for creating websites and web services in any .NET language. In order to provide a good architecture for Silverlight applications, .NET RIA Services was introduced. This framework brings ASP.NET and Silverlight together, and address the complex aspects of building multi-tier applications. In multi-tier applications, data flows between different layers, or tiers, e.g. between a client and a server. This flow passes through trust boundaries, as illustrated in Figure 6 below, and what .NET RIA Services does is to provide support for applications to be able to communicate across these boundaries. (Microsoft: .NET RIA Services, 2009)

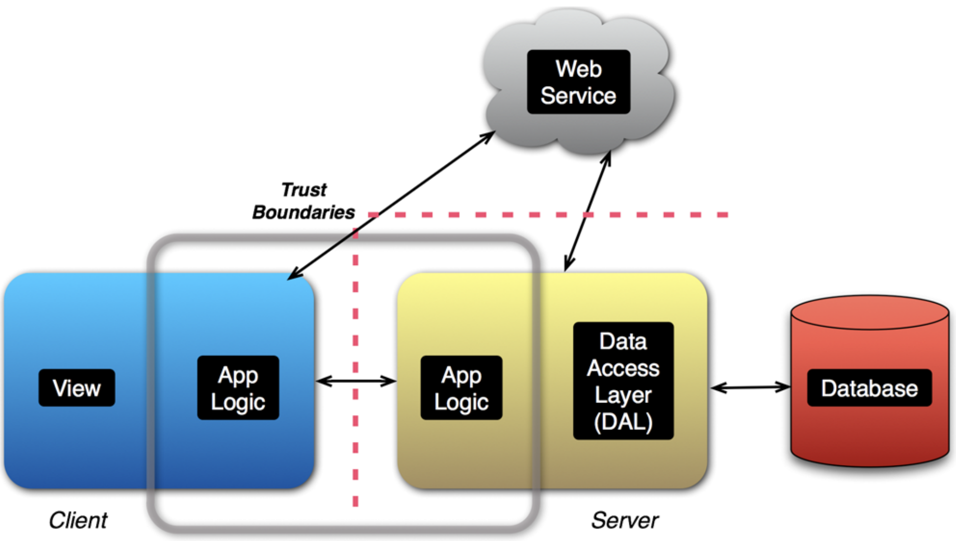


**Figure 6.** *Typical architecture of a multi-tier application, with the gray box defining where .NET RIA Services operates.*

The gray box between the client application logic and the server application logic, in Figure 6 above, is where .NET RIA Services comes into play. It ties together the gap of the tiers by providing tools for different operations on resources. More specifically, it provides patterns and services for data fetching and binding, data validation, authentication and roles. This functionality, which is present in ASP.NET, would be rather hard and tedious to code when communicating across such tier boundaries. But with this framework it is generated more or less automatically, which facilitates the work substantially. The central part for this to work is to implement a `DomainService` class. This class defines the application's business logic, and hence specifies what operations should be allowed on what resources and so on. (Microsoft: .NET RIA Services, 2009)

Another benefit of this framework is that it becomes easier to perform unit testing on an application's mid-tier. With the architecture of .NET RIA Services, the web service layer, or database layer for that matter, can be mocked out and replaced by any data source for testing purposes. (Microsoft: .NET RIA Services, 2009)

### 3.4.2. Silverlight Mobile

Silverlight Mobile has been announced but not released yet. A working example has been demoed once, and according to Microsoft, Silverlight will be supported by Windows Mobile and Nokia N60. The release is said to be done in the end of 2009, but more information than this is not available at the time of writing (Microsoft: Silverlight For Mobile, 2009)

### 3.4.3. Tools

#### Microsoft Visual Studio

One of the benefits with Silverlight is the great tooling support. Just as for other .NET applications, The Microsoft Visual Studio IDE is used for developing Silverlight applications using the Silverlight Tools add-on. The powerful IDE Visual Studio 2008 got voted the "Best IDE" in InfoWorld's 2009 Technology of the Year Awards (InfoWorld, 2009), and it has been on the market since 1997. Visual Studio offers plenty of opportunities for most parts of the software development life cycle, ranging from database schema designer and GUI designer to version control and other collaborative features for team support. (Microsoft: Visual Studio, 2009)

Silverlight applications can be visualized in a preview mode in Visual Studio 2008, but the next version (2010) will feature a fully editable and interactive designer, where controls can be dragged directly into the application layout. (Microsoft: Silverlight, 2009)

#### Eclipse4SL

Eclipse4SL is an open-source project with the purpose of providing tools for the Eclipse IDE to support development of Silverlight applications. The project's focus is to facilitate for Java developers to develop with Silverlight and improve the ability to communicate with Java Web Services. Projects saved in Eclipse can be opened in both Visual Studio and Expression Blend, and moreover, the plugin contains a XAML editor to preview designs. (Eclipse4SL, 2009)

### Microsoft Expression Blend

Microsoft Expression Blend is a user interface design tool for creating user interfaces to WPF and Silverlight applications, included in the tool suite Microsoft Expression Studio. Similar to Flex and Flash Catalyst, Silverlight also features a designer-developer workflow whereby it is possible to exchange designs between Visual Studio and Expression Blend with the shared file format XAML. One of the more powerful features of this tool is called SketchFlow. In this module, designers can create their low-fidelity mockups and instantly make them interactive using screens, states, transitions and sample data, making prototyping very easy to work with. SketchFlow has the ability to automatically create a documentation of the design, and what it does is to create a Microsoft Word document containing table of contents, showing application flow and describing each screen etc., resulting in a complete and nice-looking design document.

Moreover, Expression Blend can import graphical assets from Adobe Photoshop and Adobe Illustrator. The meta-information about these assets, i.e. layers etc., is preserved, and creating custom controls from such designs becomes trivial due to Expression Blends ability to turn such assets into rich data bound controls like for example a custom scrollbar. (Microsoft: Expression Blend, 2009)

## 3.4.4. History Timeline

The following figure illustrates important milestones for Silverlight:

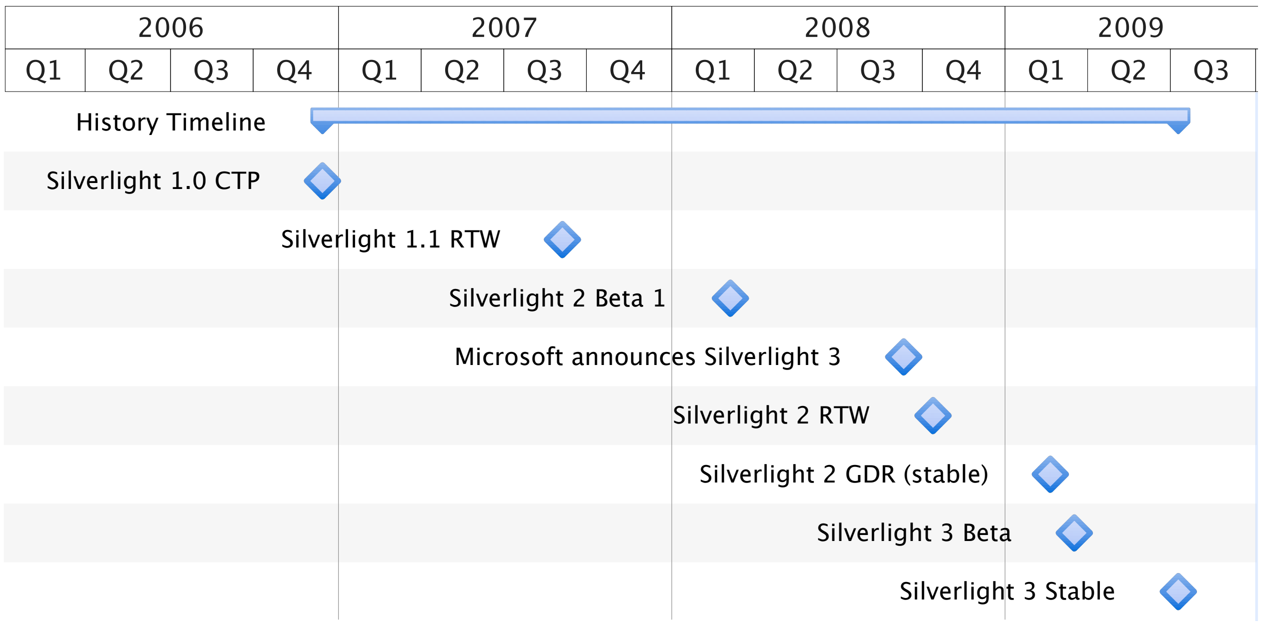


**Figure 7.** *Important milestones for Silverlight.*

### December 2006

Microsoft releases a CTP (pre-beta release) of Silverlight 1.0, initially referred to as Windows Presentation Foundation/Everywhere (WPF/E).

**5 September 2007**

Silverlight 1.1 released as a RTW (Release To Web).

**5 March 2008**

Silverlight 2 Beta 1 released.

**12 September 2008**

Silverlight 3 is announced.

**14 October 2008**

Silverlight 2 released as a RTW.

**19 February 2009**

Silverlight 2 becomes stable (General Distribution Release).

**18 March 2009**

Silverlight 3 Beta unveiled and released at the MIX09 conference.

**10 July 2009**

Silverlight 3 is released (stable).

# 4. Analysis

In order for a comparison between Adobe Flex, JavaFX and Microsoft Silverlight to be possible, their characteristics were first identified. This section contains comparison tables with both technical- and non-technical characteristics. Secondly, it covers the current market, statistics and existing solutions developed with each framework. The application and the requirements that constitute the basis of the prototype will also be presented.

## 4.1. Technical Characteristics

| Technical Characteristic | Description |
|---|---|
| 3D Effects | The ability to display graphics in what visually looks like 3D. This often means that coordinates are mapped onto a 2D-surface and adding shadows etc. in real-time, making it look like 3D effects. |
| Animations, Transformations & Effects | Abilities to rotate, zoom, fade etc. and also do this over time (animations). |
| Browser Back/Forward | Can the back and forward buttons of the browser be used for navigating through the application, to improve usability? |
| Charts | Are there any chart components available? This is important for enterprise applications. |
| Controls & Components | How many graphical controls and components exist? |
| CSS Styling | Ability to style the components using the CSS (Cascading Style Sheets) language. |
| Data Binding | This is a feature of the underlying language that enables variables to be bound to (depend on) other variables, and automatically update their value when the dependent variable(s) changes. |
| Data Grid | Is there a data grid component available? Is there support for pagination out of the box? This is important for many types of applications, in particular for enterprise applications. |
| Deep Linking | Deep linking is a term describing the ability to address a specific page with a hyperlink, e.g. to enable bookmarking and improve Search Engine Optimization (SEO). |
| Deep Zoom | Deep zoom is a functionality that allows zooming and panning on huge images and/or image sets, kind of like Google Maps. |
| Drag-to-Install | Is it possible to install the application locally, by dragging it from the browser onto the desktop? |
| File Upload | Is it possible to upload files from the client to the application? |

| Technical Characteristic | Description |
|---|---|
| GPU Acceleration | Is the framework able to make use of the Graphics Processing Unit (GPU) to handle graphics and 3D-rendering? This would result in more efficient use of the CPU and enhance the performance. |
| Hardware Platforms | What hardware platforms can applications run upon? |
| HD Video | The ability to play high-definition video (minimum 720p resolution, 1280x720 pixels). Whether or not the framework supports H.264 compression, which has become a standard for online HD video. |
| JavaScript Interaction | Is it possible to call JavaScript code from within applications? Is the reverse possible too? |
| Local Data Storage | Ability for the application to store data locally. |
| Localization | Functionality to localize (translate) the application to different languages. |
| Multi-page Support | Web applications often consist of several pages, or states, that the user can navigate between. This feature covers out of the box support for this purpose. |
| Multi-threading | Support for multi-threaded executions. |
| Network Protocols | What kind of network protocols like SOAP, RTSP etc., and web service protocols like REST, XML-RPC etc. does the framework support? |
| Offline Access | Offline access means that the application can be executed outside the browser, and hence also when there is no Internet connection. |
| Printing | Is it possible to print from within the applications? |
| Profiling | Is it possible to measure the performance in terms of CPU and memory usage of the application? |
| Programming Languages | What languages can be used to develop applications? |
| Push Technology | Ability for the server to push data to the client without using polling techniques. |
| RSS/Atom Feeds | Whether or not there is built-in support for RSS- or Atom web feeds. |
| Runtime File Size | What is the size of the runtime that is needed? |
| Search Engine Support | Whether or not the content is accessible by search engines. |
| Skinnable Controls | Whether or not controls are skinnable or not means that their look and feel, i.e. their graphical appearance, can be changed. |

| Technical Characteristic | Description |
|---|---|
| Software Requirements | What software is required to run the applications? This aspect has been categorized into Operating Systems, Browsers and Runtime. |
| Startup Procedure | How seamless is the startup procedure from a user perspective? |
| Themes | Are there any themes that can be used to change the graphical appearance of controls? |
| Unit Testing | Is there any testing framework for performing unit testing? |
| Validation | Built-in support for validating data entry and informing the user, a task that is very common to all applications, but often tedious for the developer to code. |
| Webcam & Microphone Support | Can video from the webcam and audio from the microphone be captured and used in the application? |

**Table 2.** *Description of technical characteristics.*

The comparison tables in this report are color-coded to visualize and easier get a general view of the relative advantages and disadvantages with each characteristic (each row) between each framework. The colors could be interpreted as a relative scale were green being better, yellow fair and red not as good as the others. In the table below, a cell is usually marked yellow if a feature is not native but it is still possible to achieve the same functionality. If all three frameworks share the same properties for one characteristic, all its cells are displayed in green.

These are the colors used in the tables below: 🟩 **Green** 🟨 **Yellow** 🟥 **Red**

| Technical Characteristic | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| 3D Effects | Yes, in Flash Player 10 | Yes, with PerspectiveTransform | Yes, with Perspective 3D |
| Animations, Transformations & Effects | 🟩 Yes | 🟩 Yes | 🟩 Yes |
| Browser Back/ Forward | 🟩 Yes | 🟥 No | 🟩 Yes |
| Charts | 🟩 Yes | 🟩 Yes | 🟩 Yes |

| Technical Characteristic | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| Controls & Components | 50+<br><br>+ many other components, both open-source and commercial | 20+<br><br>+ components from Java Swing and the open-source project JFXtras | 60+<br><br>+ components from the open-source project Silverlight Toolkit |
| CSS Styling | Yes | Yes | No, only XAML |
| Data Binding | Yes | Yes | Yes |
| Data Grid | Yes, but no pagination out of the box | No, but could be implemented with a JTable from Java Swing (no pagination) | Yes, with pagination |
| Deep Linking | Yes | No | Yes |
| Deep Zoom | Yes, but third-party | No | Yes |
| Drag-to-Install | No | Yes (requires Java 1.6 update 10) | No |
| File Upload | Yes, integrated in Flash 10 | Not native, but works with Java | Yes, through third-party controls |
| GPU Acceleration | Yes, with Flash 10 | No | Yes |
| Hardware Platforms | Desktop/laptop computers | Desktop/laptop computers<br>Mobile phones<br>TV<br>Set-top boxes (including IPTV)<br>Blu-ray<br>Gaming consoles<br>(Java is also included in some cars) | Desktop/laptop computers<br><br>(Microsoft has announced Silverlight to be released for Windows Mobile and Nokia N60 in the end of 2009) |
| HD Video | Yes, with H.264. Also support for true HD video (1080p) | Yes, but not H.264. Also support for true HD video (1080p) | Yes, with H.264. Also support for true HD video (1080p) |
| JavaScript Interaction | Yes, both ways | Yes, both ways | Yes, both ways |

| Technical Characteristic | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| Local Data Storage | Yes | Yes | Yes |
| Localization | Yes | Yes | Yes |
| Multi-page Support | Yes, using ViewStacks | No | Yes, using the Navigation Application template in Visual Studio |
| Multi-threading | No | Yes, since JavaFX can utilize Java's threading capabilities. | Yes, since Silverlight implements a subset of .NET |
| Network Protocols | Streaming media: RTMP (Real Time Messaging Protocol)<br><br>Web services:<br>• XML-RPC<br>• SOAP<br>• REST<br><br>Binary protocol: AMF (ActionScript Messaging Format) | Streaming media: RTSP (Real Time Streaming Protocol)<br><br>Web services:<br>• SOAP<br>• REST<br><br>UDP, XML-RPC etc. due to Java integration | Streaming media: Adaptive Streaming (combination of RTSP and HTTP progressive downloading)<br><br>Web services:<br>• WCF<br>• ASMX<br>• XML-RPC (third-party)<br>• REST<br><br>Binary protocol (binary XML) |
| Offline Access | Yes, with Adobe AIR | Yes | Yes |
| Printing | Yes | Yes | No, needs to export to HTML/CSS/PDF |
| Profiling | Yes | Yes | Yes, but not native. Either convert it to a WPF application or use XPerf or Silverlight Spy (third-party) |
| Programming Languages | MXML<br>ActionScript | JavaFX Script<br>Java | All .NET languages, i.e.:<br>• C#, VB.NET etc.<br>XAML |
| Push Technology | Yes, through BlazeDS | No | Yes |

| Technical Characteristic | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| RSS/Atom Feeds | Yes, but not native. Available as a free ActionScript 3.0 library named *tas3syndicationlib* at Google Code | Yes | Yes |
| Runtime File Size | Windows XP & Vista: 1.9 MB<br>Mac OS X: 5.7 MB<br>Linux: 3.9 MB | Windows XP & Vista: 15.9 MB<br>Mac OS X: None, shipped by default<br>Linux: 19.4 MB | Windows XP & Vista: 4.7 MB<br>Mac OS X: 8.7 MB<br>Linux (Moonlight project): 0.9 MB |
| Search Engine Support | Yes | No | Yes |
| Skinnable Controls | Yes | Yes | Yes |
| Software Requirements (OS) | Microsoft Windows<br>Mac OS X<br>Linux<br>Solaris | Microsoft Windows<br>Mac OS X<br>Linux (Ubuntu)<br>OpenSolaris<br>Android<br>BlackBerry OS<br>Windows Mobile | Microsoft Windows<br>Mac OS X<br>+ Linux/FreeBSD/Solaris by using the open-source implementation Moonlight |
| Software Requirements (browser) | Cross-browser with runtime. Available for the most popular browsers. | Cross-browser with runtime. Available for the most popular browsers. | Cross-browser with runtime. Available for the most popular browsers. |
| Software Requirements (runtime) | Flash version 9, or 10 to make use of the latest Flex applications<br>Adobe AIR for offline use | Java Runtime 1.5 | Silverlight Runtime 1.0 |

| Technical Characteristic | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| Startup Procedure | Invisible | Depends on how the application runs, and how it has been signed, but most often many dialogs pops up. See Appendix III for more information.<br><br>Invisible after signing a security certificate to be trusted, and running inside the browser. | Invisible, but sometimes the runtime detection is buggy and the user is told to download the runtime even if it is present. |
| Themes | Yes | No | Yes |
| Unit Testing | Yes, with FlexUnit | Yes, but not native. The open-source project JFXtras provides a unit testing solution. JUnit can also be tweaked to work. | Yes |
| Validation | Yes | No, need to be coded manually | Yes |
| Webcam & Microphone Support | Yes | No | No |

**Table 3.** *Comparison between technical characteristics of Flex, JavaFX and Silverlight.*

## 4.2. Non-Technical Characteristics

| Non-Technical Characteristic | Description |
|---|---|
| Costs | Are there any costs associated with the framework and its tools? |
| Designer Tools | What tools are offered for designers? |
| Developer Tools | What tools and IDEs are offered for developers? |
| License | What license does the framework and its associated tools have? Is it open-source? |

**Table 4.** *Description of non-technical characteristics.*

| Non-Technical Characteristic | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| Costs | Adobe Flash Builder<br>• Standard: 2 375 SEK<br>• Professional: 6 625 SEK<br>Adobe Flash Catalyst<br>• Currently only in beta | None | Microsoft Visual Studio<br>• Standard: 3 995 SEK<br>• Professional: 10 620 SEK<br>Microsoft Expression Studio<br>• 4 340 SEK |
| Designer Tools | Adobe Creative Suite (for exporting graphics to Flash Catalyst or Flash Builder)<br>Adobe Flash Catalyst | JavaFX Production Suite (Adobe Creative Suite export plugin)<br>JavaFX Authoring Tool (not released yet) | Microsoft Expression Blend (part of Microsoft Expression Studio) including SketchFlow |
| Developer Tools | Adobe Flash Builder 4 (built upon Eclipse)<br>FlexBean (NetBeans plugin)<br>Ensemble Tofino (Visual Studio plugin)<br>+ support for AIR and ActionScript in several more IDEs | NetBeans<br>Eclipse plugin | Microsoft Visual Studio<br>Eclipse4SL (Eclipse plugin, open-source) |
| License | Adobe Flex SDK: open-source (Mozilla Public License)<br>BlazeDS: open-source (LGPL v3)<br>Adobe Flash Player: Proprietary freeware EULA<br>Adobe AIR: Proprietary EULA<br>Adobe Flash Builder: Proprietary EULA<br>Adobe Catalyst: Proprietary EULA | Compiler: open-source (GPL 2.0)<br>Tools: open-source (GPL 2.0)<br>Runtime: EULA | Proprietary MS-EULA<br>Components from Silverlight Toolkit are licensed under Microsoft Public Licence (MS-PL) |

**Table 5**. *Comparison between non-technical characteristics of Flex, JavaFX and Silverlight.*

## 4.3. Current Market and Statistics

### Adobe Flash Player

On Adobe's website, a survey of the Flash Player version penetration made by Millward Brown is publicly available. It is important to note that this proprietary survey was commissioned by Adobe Systems, and hence not very objective. This survey covers *mature markets* (US, Canada, UK, Germany, France and Japan) and *emerging markets* (China, S. Korea, Russia, India and Taiwan). The survey was performed by letting the people visit a website which showed different images in different Flash version formats, and from this, the Flash Player version could be determined. The sample size in the survey was 1,000 in the US, and 400 in the other countries. The results were weighted against a weighting scheme that took both Internet penetration and population density into account, and the error margin was 3-6% calculated at a confidence interval of 95%. (Adobe: Flash Penetration, 2009)

In order to make use of the majority of the features in Flex, including ActionScript 3.0, Flex applications require Adobe Flash Player 9 to be installed. As of June 2009, mature markets were estimated to 98.8% for Flex-enabled versions. The emerging markets had a corresponding 97.1%. More detailed information on how the study was performed can be found on the Adobe's website. (Adobe: Flash Penetration, 2009)

### Java Runtime

JavaFX requires Java Runtime 1.5 or higher to run. Java has a developer base of over six million people, and 9 out of 10 new PCs are shipped with the Java Runtimes as well as 8 out of 10 handsets shipped in 2008 (Sun Microsystems: JavaFX FAQ, 2009) & (Ritter, 2009). Java is currently available on over 1 billion desktops, 2.6 billion mobile phones and 1 billion set-top boxes, according to statistics presented at the JavaOne Conference in 2009 (Sun Microsystems: JavaOne, 2009).

According to Arindam Bhattacharya, a JavaFX Platform Product Manager at Sun Microsystems, the JavaFX install base was more than 250 million on the desktop in June, 2009. By the same time, the SDK had been downloaded by 450,000+ developers, and the IDE plugins by 250,000+. (Bhattacharya, 2009)

### Silverlight Runtime

Statistics of Silverlight downloads and install bases are not publicly available, and Microsoft has not commissioned any survey similar to what Adobe has. Another option is to look at independent statistics websites like StatOwl.com or RIAStats.com, that collects information from users who visit certain websites.

### Statistics Websites

StatOwl.com is a website with statistical analysis and market research of Internet usage trends. It gathers data by tracking what end-users are running while visiting other websites by including a small JavaScript that identifies installed runtimes. These websites later report this data back to StatOwl. According to one of their employees, Michael McCallister, they have enough sources to provide statistically significant results, i.e. more than 100. The US and

Canada are well covered, but data for other parts of the world is lacking. The following figures are taken from StatOwl.com:

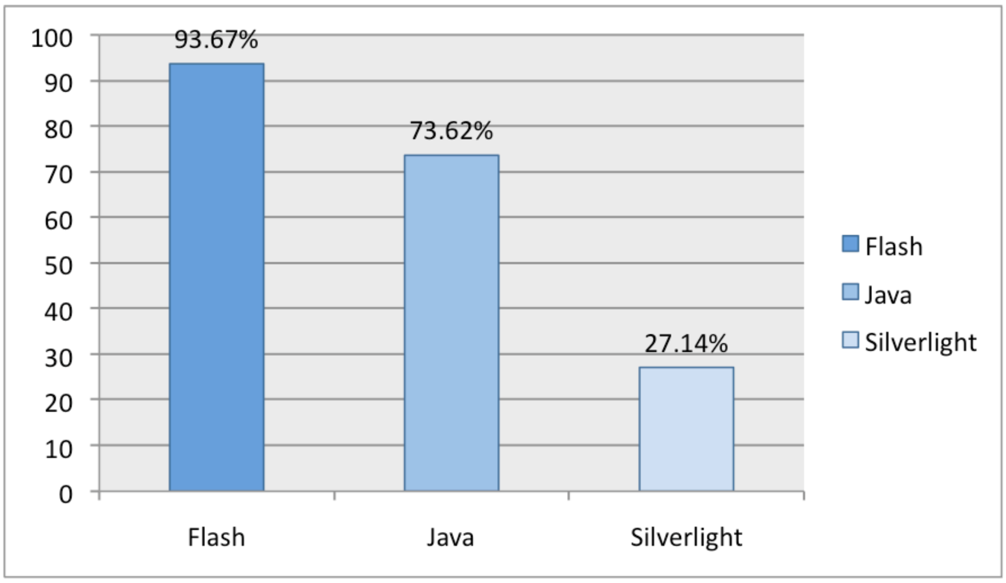


**Figure 8.** *Statistics of runtime penetration in the browser in May, 2009. (StatOwl.com, July 2009)*

The following pie-chart statistics of runtime penetrations in the browser are taken from another website, RIAStats.com. They have been collected from 1,098,189 daily unique browsers across 62 public websites (not websites with RIA content) over a period of 30 days prior to 1st of July, 2009. RIAStats.com is not backed up by any company or interest group, but rather independent and owned and operated by a single individual. However, since the websites are not known, it is hard to comment on the authenticity of the statistics.

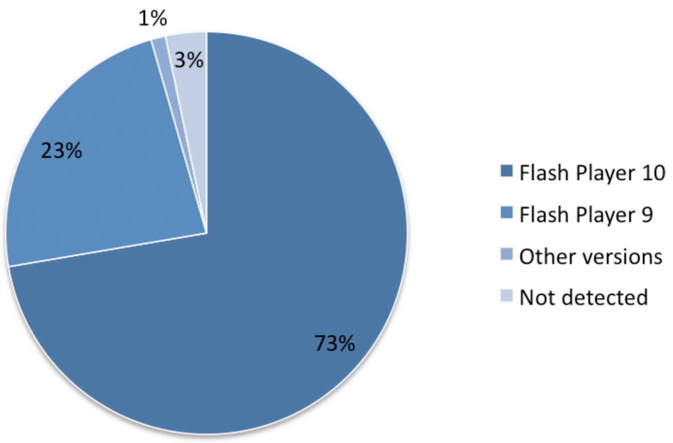


**Figure 9.** *Adobe Flash Player Runtime Penetration. (RIAStats.com, July 2009)*

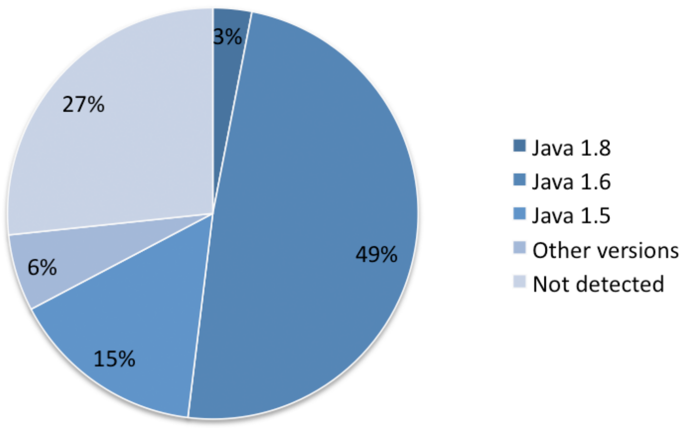


***Figure 10.*** *Java Runtime Penetration. (RIAStats.com, July 2009)*

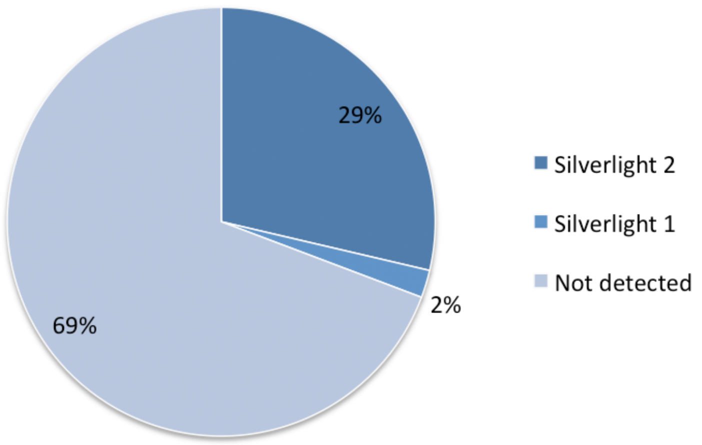


***Figure 11.*** *Silverlight Runtime Penetration. (RIAStats.com, July 2009)*

The table below contains a summary of statistics for the runtime penetrations for RIA-enabled install bases. In this context, RIA-enabled means Flash Player 9+ for Flex, Java Runtime 1.5+ for JavaFX, and Silverlight is simply covering all versions.

| | **Statistics Source** | | |
|---|---|---|---|
| | **StatOwl.com** | **RIAStats.com** | **Adobe** |
| **Adobe Flash Player 9+** | 93.67% | 95.54% | Mature markets: 98.8%<br><br>Emerging markets: 97.1% |
| **Java Runtime 1.5+** | 73.62% | 64.29% | N/A |
| **Silverlight Runtime 1.0+** | 21.14% | 30.85% | N/A |

**Table 6.** *Statistics of runtime penetrations for the browser in July, 2009.*
*(StatOwl.com, RIAStats.com and Adobe: Flash Penetration, July 2009)*

## Google Insights For Search

Google Insights for Search is a tool for measuring the interest level over time of any topic by analyzing historical web searches made on Google. The results can be filtered by the three parameters time, location and category; and they are computed relative to the total number of searches done over the chosen period of time. The y-axis in the graph below indicates the interest level, which is normalized to cancel out anomalies and presented on a scale from 0-100, where 100 represents the highest peak of the search terms. Several approximations are used to compute the results. (Google: Insights for Search, 2009)

The search was filtered to the time period September 2006-September 2009 and worldwide location, and was furthermore drilled down to only include the following category: *Computers & Electronics > Programming.*

These are the keywords that were used in the search, and note that the "+"-sign denotes the OR-operator:

**Flex:** *flex + adobe flex + "adobe flex"*

**Silverlight:** *silverlight + microsoft silverlight + "microsoft silverlight"*

**JavaFX:** *javafx + java fx + "java fx"*

The following figure illustrates the results retrieved from Google Insights for Search, for the parameters specified above:

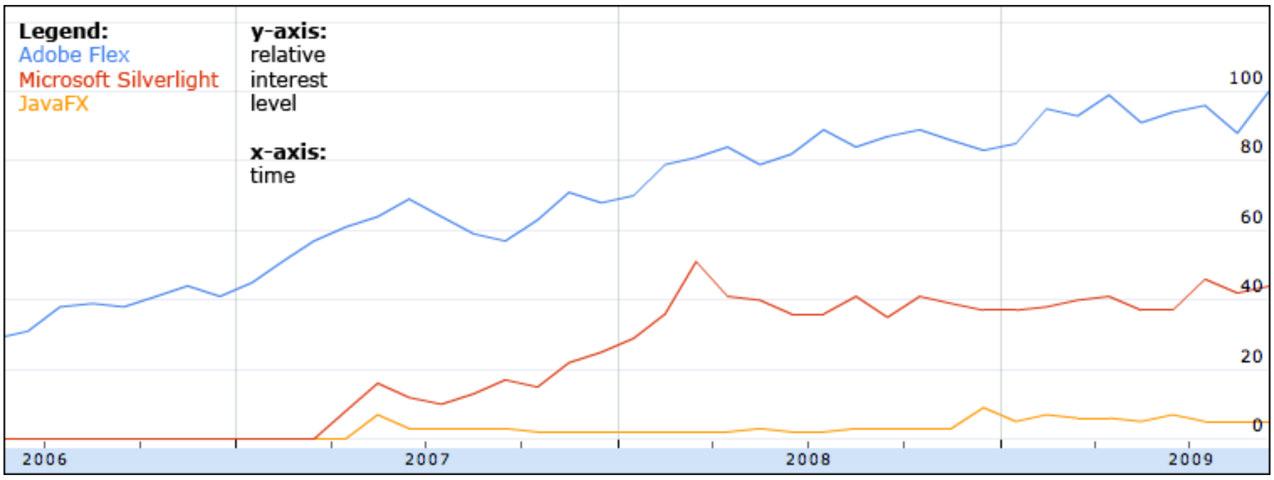


**Figure 12.** *Relative interest levels of Flex, Silverlight and JavaFX. This graph was computed with Google Insights for Search.*

From this graph, the following interest levels can be seen in September 2009:

| RIA Framework | Relative Interest Level |
|---|---|
| Adobe Flex | 100 (67.1%) |
| Silverlight | 44 (29.5%) |
| JavaFX | 5 (3.4%) |

**Table 7.** *Relative interest levels of Flex, Silverlight and JavaFX in September 2009.*

Google Insights for Search can also display regional differences of the interest levels. It is important to point out that it is only possible get relative data for regions, and not absolute, since the data has been normalized by the total traffic from each respective region. As an example, Sri Lanka only has about 40% less interest level as compared to India on Adobe Flex, but this does probably not mean that Sri Lanka only makes 40% less web searches in total. The number of Internet users in India was 42,000,000 in 2007, as compared to 771,700 in Sri Lanka in 2008 (Internet World Stats, 2009), from which it can be concluded that this would be very unlikely since the amount of Internet users in Sri Lanka is only 1.8% compared to India. Instead, Google Insights for Search presents an interest level relative to other searches from within the same region, i.e. a level of popularity and not an absolute level.

The following figures illustrate the relative regional interest levels for each framework:

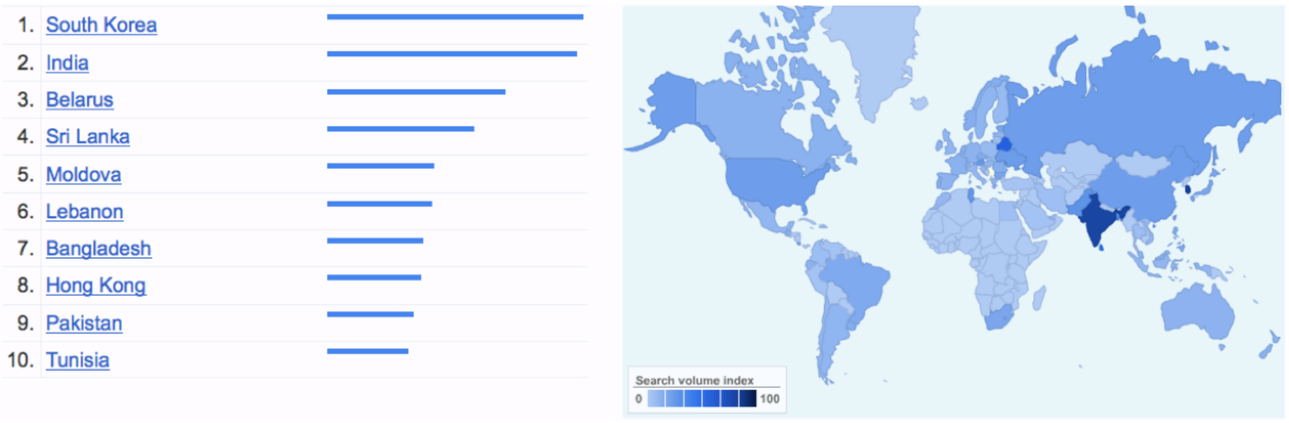


**Figure 13.** *Relative regional interest level for Adobe Flex.*

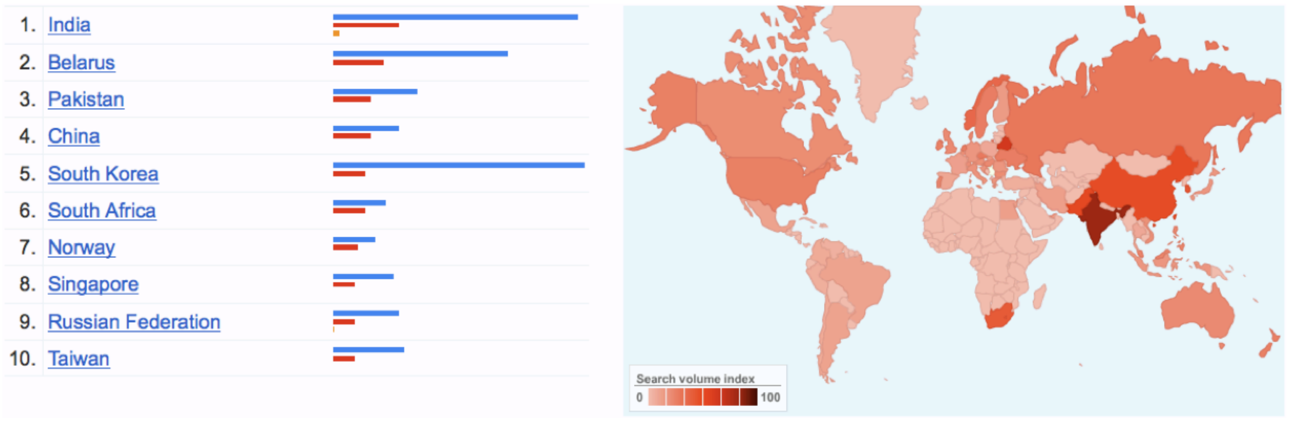


**Figure 14.** *Relative regional interest level for Microsoft Silverlight.*

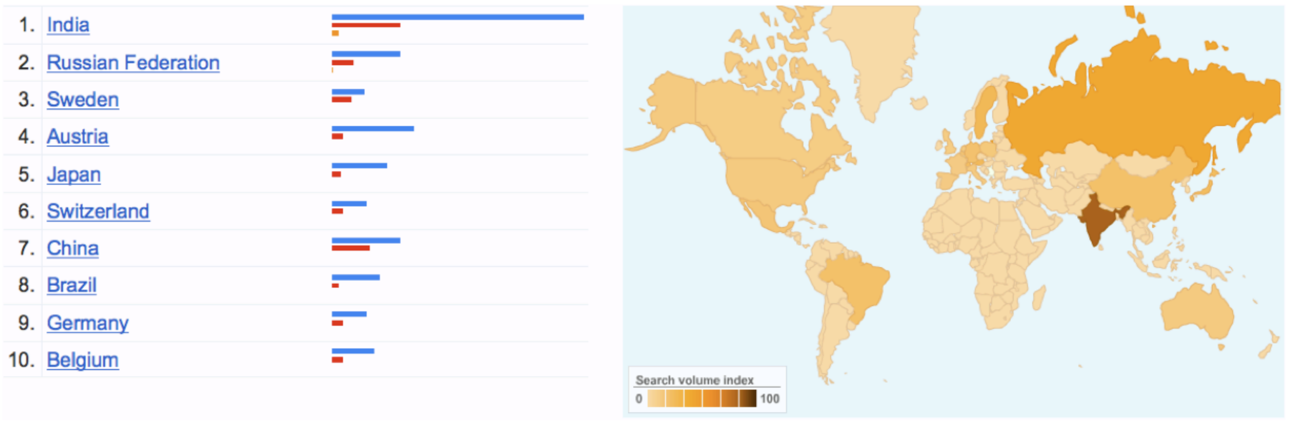


**Figure 15.** *Relative regional interest level for JavaFX.*

## Online Book Store - Amazon.com

Another way to measure how well-established and popular a certain technology is, or in this case a framework, is to look at how much literature that can be found about it. However, topic maturity must be taken into account, since it for natural reasons not has been written as much about a new subject as an older one. This is most likely a way of looking at history rather than future trends, but nevertheless important to mention as it is yet another measurable input, and somehow probably correlates with the size of the current developer community. In this thesis, relevant books at Amazon.com, which is America's largest online retailer, were counted by searching for book titles in the category *Computers & Internet* that included each framework name.

| RIA Framework | Number of relevant books on Amazon.com |
|---|---|
| Adobe Flex | 35 (48.0%) |
| Silverlight | 29 (39.7%) |
| JavaFX | 9 (12.3%) |

**Table 8.** *Distribution of relevant books on Amazon.com in August 2009.*

## Developer Forums

Online forums, where developers can ask questions about either general thoughts or specific problems, were also analyzed. Each framework had their own official forum; forums.adobe.com, forums.sun.com and www.silverlight.net/forums. Another popular forum called Stack Overflow (www.stackoverflow.com) where it is possible to tag entries, was also used.

## Job Search Websites

Three types of websites have been identified and used in this survey. These are job boards, a job search engine and a professional networking website. At the time of writing, it is considered to be recession so companies are not hiring as much people as they do when the economy is more stable, but this survey is still made relative each framework and should therefore provide an accurate ratio. However, one have to account for the likelihood that companies may allocate less resources for new technologies and instead put more focus on well-established ones, but it could be the contrary too. As an example from the software industry, Microsoft chose to keep their $9 billion R&D budget the fiscal year (2009), instead of cutting it (Rigby, 2009).

The following four websites and settings were chosen for the job search survey:

- Monster.com, which is the world's largest employment website
    - Country: U.S.
    - Categories: Computer Software, Computer/IT Services, Internet Services
    - Search by: Job Title
- CareerBuilder.com, which is the largest job board website in the U.S.
    - Country: U.S.
    - Category: Information Technology
    - Search by: Keywords
- Indeed.com, which is a job search engine that fetches results from thousands of other websites.
    - Country: U.S.
    - Search by: Keyword
- Naukri.com, India's largest job board website.
    - Country: India
    - Categories: IT-Software/Software Services
    - Search by: Keyword

Demographical job demands have not been accounted for. Of these four websites, the first three are U.S.-based, and the fourth covers India, but it should still give a reasonable picture of the market. Since India is known for its big community of software developers, the job site Naukri.com, which is India's largest, was also chosen apart from the U.S.-based websites. The searches took place on the 13th of August, 2009.

The search phrases that have been used at the job sites were "flex" for Adobe Flex, "silverlight" for Microsoft Silverlight, and "javafx" for JavaFX. A position where they seek developers for Adobe Flex is almost always entitled "Flex Developer" and not "Adobe Flex Developer". Since it has been possible to choose what categories to search in, the fact that Adobe was missing out did not affect the search results significantly. For Monster.com, it was possible to search for job title instead of keywords, and this was done in order to get even more significant results. Since the keyword selection was a bit ambiguous, the result of this market analysis should only be considered as a rough measurement.

A fifth website that was used was LinkedIn (www.linkedin.com), which is very popular in its niche market, focusing on business-oriented social and professional networking. It has got over 44 million members from over 200 countries, where about half of the members are from the U.S. The search was made for both current job positions and expertise of the members, and only title was chosen due to many irrelevant results if keywords were included. Also, for Flex, "Adobe" had to be added in order to minimize the risk of getting irrelevant hits. The search was drilled down to the following categories: Computer Software, Information Technology and Services, Internet.

The following table summarizes the current market and statistics, including the results for the job search websites:

| Current Market and Statistics | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| Runtime Penetration | StatOwl.com: 93.67%<br>RIAStats.com: 95.54%<br>Adobe (mature markets): 98.8%<br>Adobe (emerging markets): 97.1% | StatOwl.com: 73.62%<br>RIAStats.com: 67.29% | StatOwl.com: 21.14%<br>RIAStats.com: 30.85% |
| Demographics | South Korea, India, Belarus, Sri Lanka, Moldova. (Google: Insights for Search, 2009) | India, Russia, Sweden, Austria, Japan. (Google: Insights for Search, 2009)<br><br>US, India, Russia, South Korea, Germany, Brazil. (Bhattacharya, 2009) | India, Belarus, Pakistan, China, South Korea. (Google: Insights for Search, 2009) |
| Downloads | N/A | 450,000+ downloads of the tools and SDK<br><br>250,000+ downloads of the IDE plugins (Bhattacharya, 2009) | N/A |
| Relative interest level on Google Insights for Search | 67.1% | 3.4% | 29.5% |
| Number of relevant books on Amazon.com | 35 (48.0%) | 9 (12.3%) | 29 (39.7%) |
| Number of discussions on each framework's official forum | 33,740 (44.8%) | 2,162 (2.9%) | 39,377 (52.3%) |
| Number of entries on the forum Stack Overflow | 2,214 (54.6%) | 75 (1.9%) | 1,765 (43.5%) |

| Current Market and Statistics | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| Number of jobs on Monster.com | 69 (84.1%) | 0 (0.0%) | 11 (15.9%) |
| Number of jobs on CareerBuilder.com | 354 (66.4%) | 0 (0.0%) | 119 (33.6%) |
| Number of jobs on Indeed.com | 1,929 (70.3%) | 8 (0.3%) | 808 (29.4%) |
| Average for the four job websites above | 72.7% | 0.2% | 27.1% |
| Number of hits on title on LinkedIn.com | 108 (43.2%) | 22 (8.8%) | 120 (48.0%) |

**Table 9.** *Comparison between current market and statistics for Flex, JavaFX and Silverlight.*

## 4.4. History Timeline

Adobe Flex has been on the market since the beginning of 2004, whereas Silverlight entered nearly 3 years later, in the end of 2006. JavaFX is the newest framework which was announced in mid 2007, but not previewed until one year later, in mid 2008. At the time of writing, this makes Flex more than 5 years old, Silverlight about 2.5 years and JavaFX only 1 year old.

The following figure illustrates the complete history timeline with important milestones for each framework:

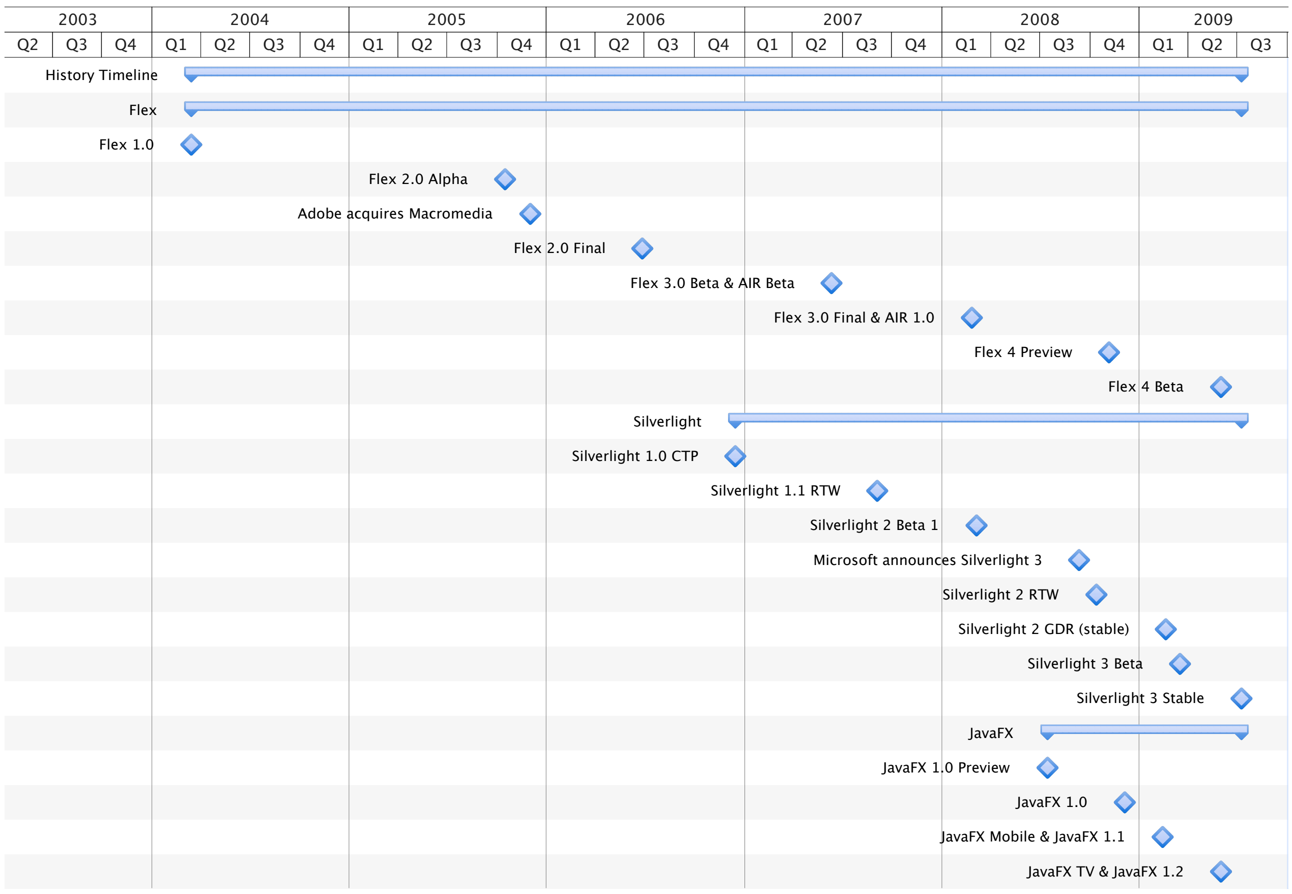


**Figure 16.** *Important milestones for Adobe Flex, JavaFX and Microsoft Silverlight.*

Adobe Flex, initially developed and maintained by Macromedia has been on the market more than twice as long as both Silverlight and JavaFX. Adobe has so far released updates to Flex once a year on average, with a trend towards being more frequent. Silverlight, unlike Flex, has experienced a rapid development with more frequent releases, and shows an even clearer trend than Flex of being released more frequently.

## 4.5. Existing Solutions

The most relevant and innovative applications, for the purpose of demonstrating the possibilities with these technologies, are described below. However, this is not a complete list but rather a selection among other solutions.

### Flex

In addition to the applications described below, Flex is used by many well-known companies like The New York Times, Volkswagen UK, NVIDIA, Fox Racing Inc and BlueTide Management. (Adobe: Customer Showcase, 2009) and (Adobe: Marketplace, 2009).

eBay is the world's largest online marketplace website, and eBay Desktop is its desktop counterpart. Instead of having to continuously watch bids on the website, or monitor incoming e-mails and refresh web pages, eBay Desktop allows for instant notifications on bid activity. The user can also subscribe to alerts on specific search criteria, and receive feeds when something is published. eBay Desktop was developed by the award-winning UI design and development agency EffectiveUI, whose clients include Adobe, Apple, Discovery Channel, Ford and GE. (Adobe: Customer Showcase, 2009)

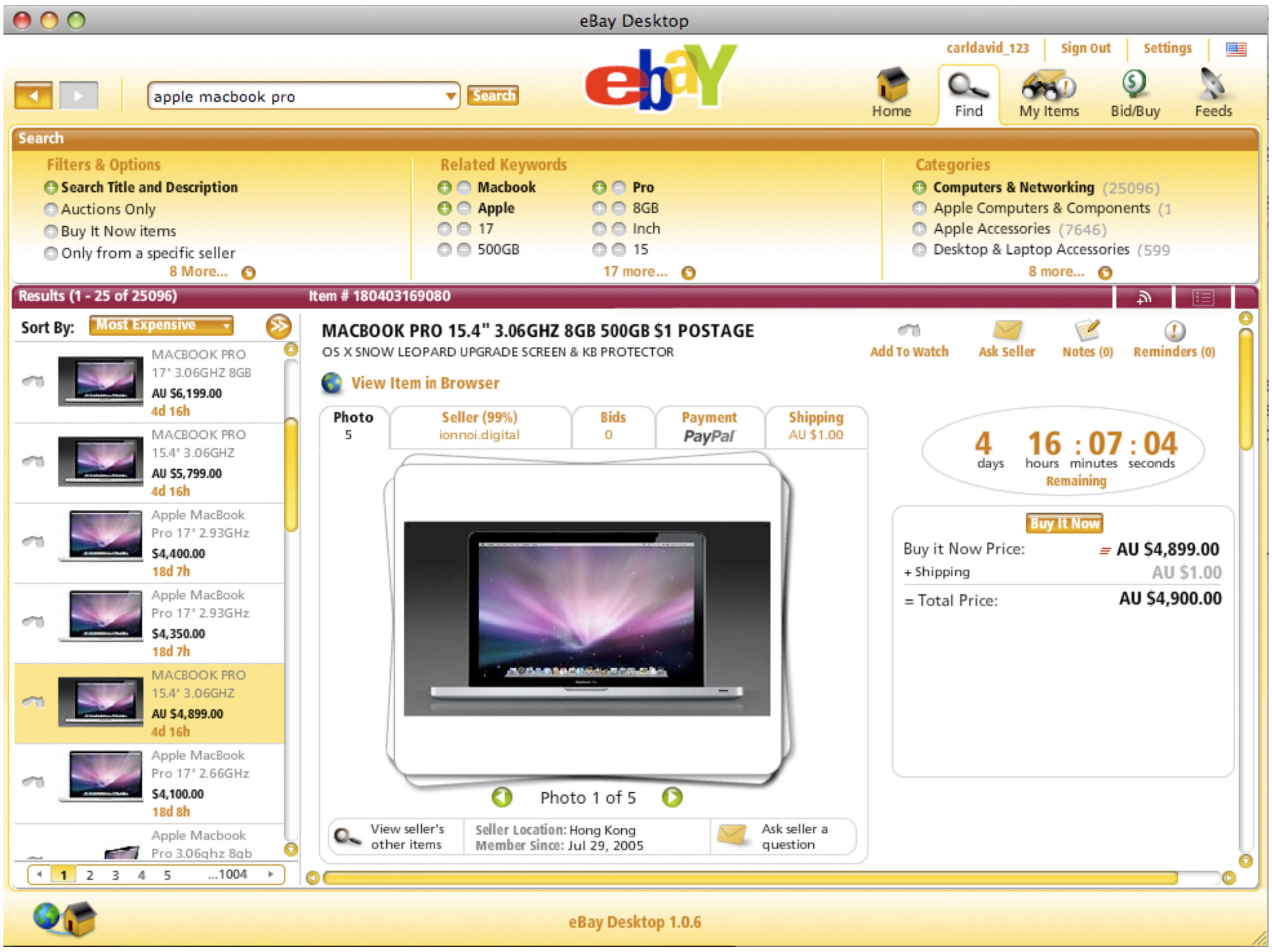

eBay Desktop
carldavid_123
Sign out
Settings
apple macbook pro
Search
Home
Find
My Items
Bid/Buy
Feeds
Search
Filters & Options
Search Title and Description
Auctions Only
Buy It Now items
Only from a specific seller
8 More...
Related Keywords
Macbook
Apple
17
500GB
Pro
8GB
Inch
15
17 more...
Categories
Computers & Networking (25096)
Apple Computers & Components (1
Apple Accessories (7646)
Desktop & Laptop Accessories (599
8 more...
Results (1 - 25 of 25096)
Item # 180403169080
Sort By: Most Expensive
MACBOOK PRO 17" 3.06GHZ 8GB AU $6,199.00 4d 16h
MACBOOK PRO 15.4" 3.06GHZ AU $5,799.00 4d 16h
Apple MacBook Pro 17" 2.93GHz $4,400.00 18d 7h
Apple MacBook Pro 17" 2.93GHz $4,350.00 18d 7h
MACBOOK PRO 15.4" 3.06GHZ AU $4,899.00 4d 16h
Apple MacBook Pro 17" 2.66GHz $4,100.00 18d 8h
Apple Macbook Pro 3.06ghz 8gb
1 2 3 4 5 ...1004
MACBOOK PRO 15.4" 3.06GHZ 8GB 500GB $1 POSTAGE
OS X SNOW LEOPARD UPGRADE SCREEN & KB PROTECTOR
View Item in Browser
Add To Watch
Ask Seller
Notes (0)
Reminders (0)
Photo 5
Seller (99%) ionnoi.digital
Bids 0
Payment PayPal
Shipping AU $1.00
4 days 16 hours : 07 minutes : 04 seconds Remaining
Buy It Now
Buy it Now Price: AU $4,899.00
+ Shipping AU $1.00
= Total Price: AU $4,900.00
Photo 1 of 5
View seller's other items
Seller Location: Hong Kong
Member Since: Jul 29, 2005
Ask seller a question
eBay Desktop 1.0.6

**Figure 17.** *Screenshot of eBay Desktop, a Flex application running with Adobe AIR.*

Parleys, another Flex application, is an e-learning platform used for viewing talks, both presentation slides and the video itself. It is possible to view different channels, tag, vote and comment webcasts and also to participate in timeline based discussion threads. Parleys also features a desktop version that runs with Adobe AIR, which means that talks can first be downloaded and then watched offline. (Adobe: Marketplace, 2009)

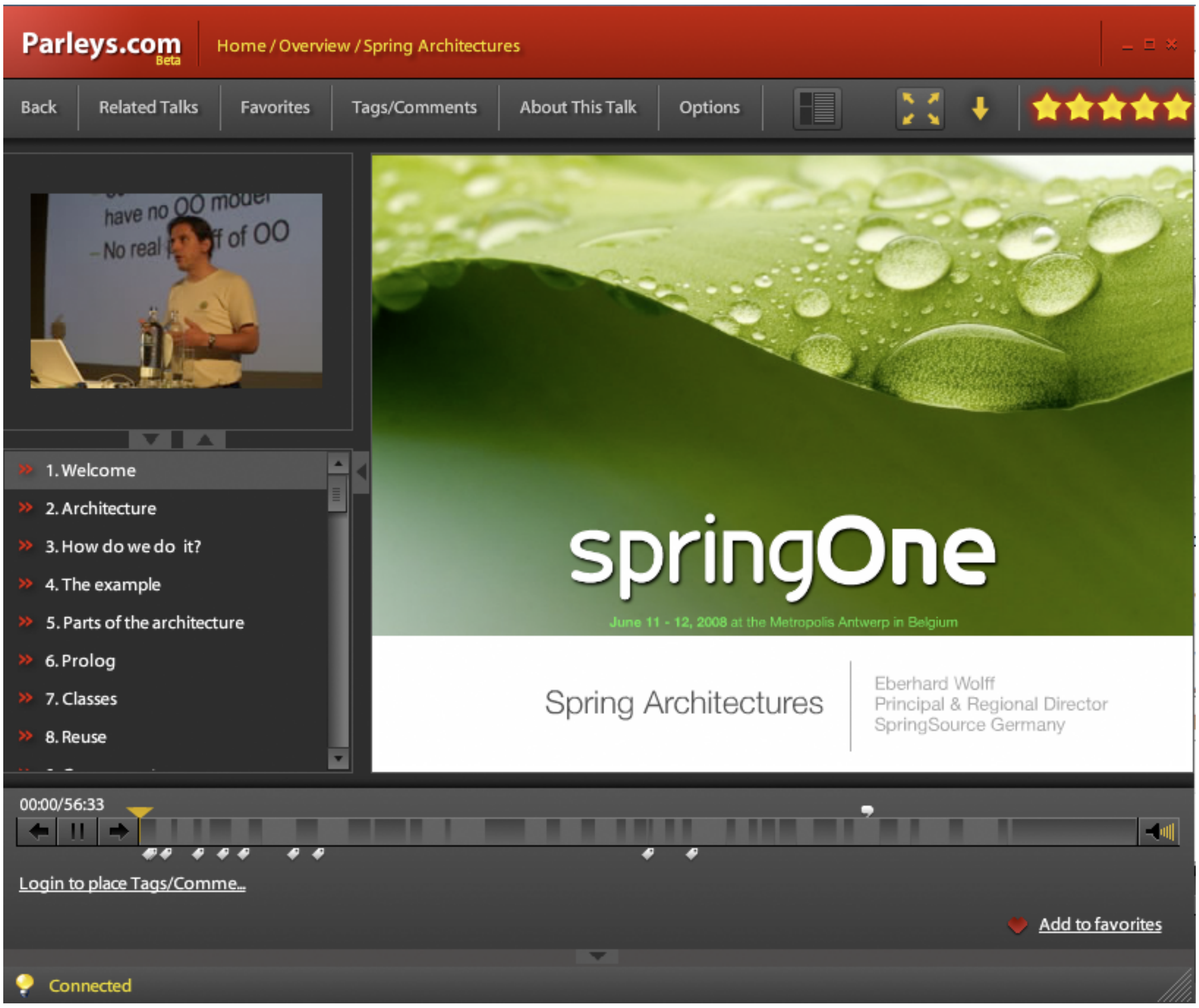


**Figure 18.** *Screenshot of Parleys Desktop, an e-learning platform made with Flex and Adobe AIR.*

Master's students at Carnegie Mellon University, studying Human-Computer Interaction, created a complex prototype of a touch-screen dental system in only six weeks, including testing. Among other features, the prototype supports patient diagnosis and rotating 3D models of a patients mouth. (Adobe: Customer Showcase, 2009) and (Hashemi, et al., 2008).

The following screenshots illustrate parts of the prototype:

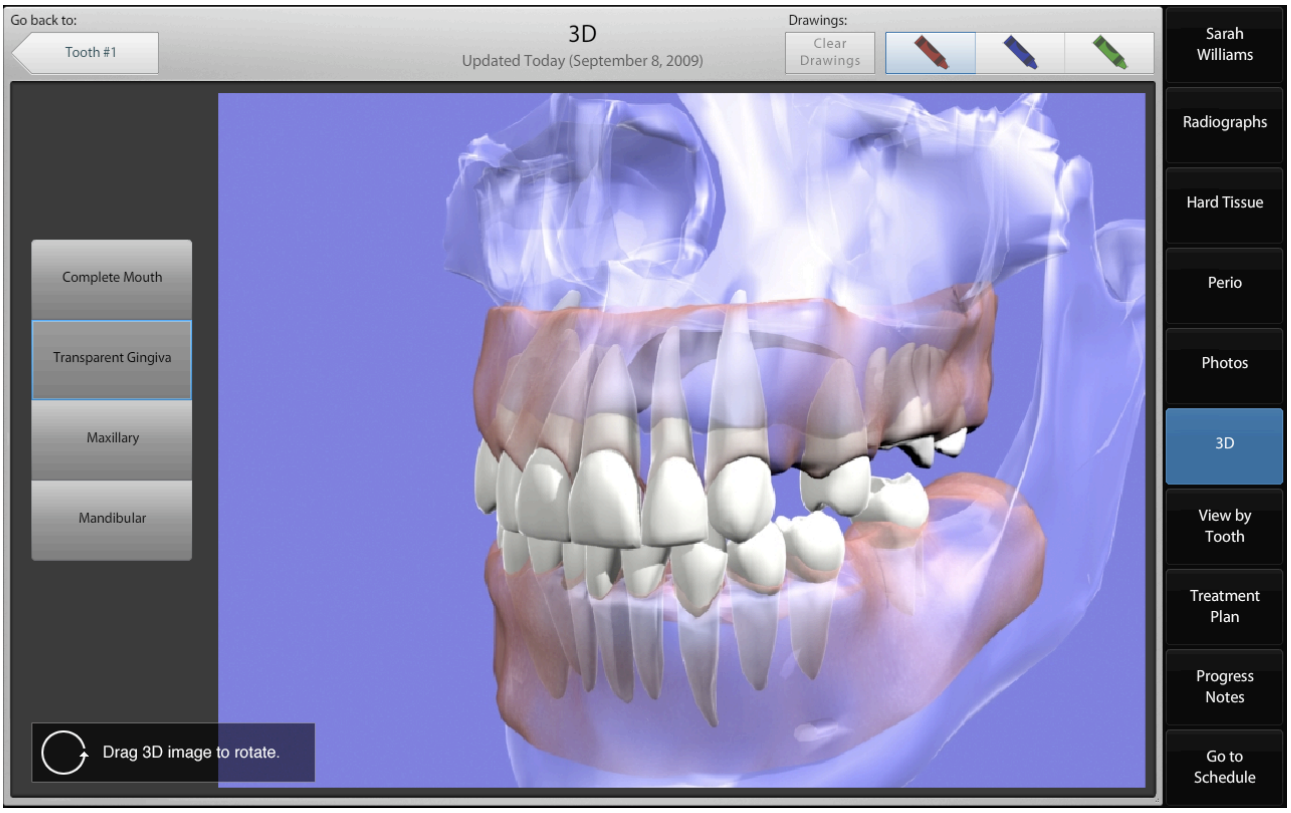


**Figure 19.** *Screenshot of a dental system prototype made in Flex by Master's students at Carnegie Mellon University.*

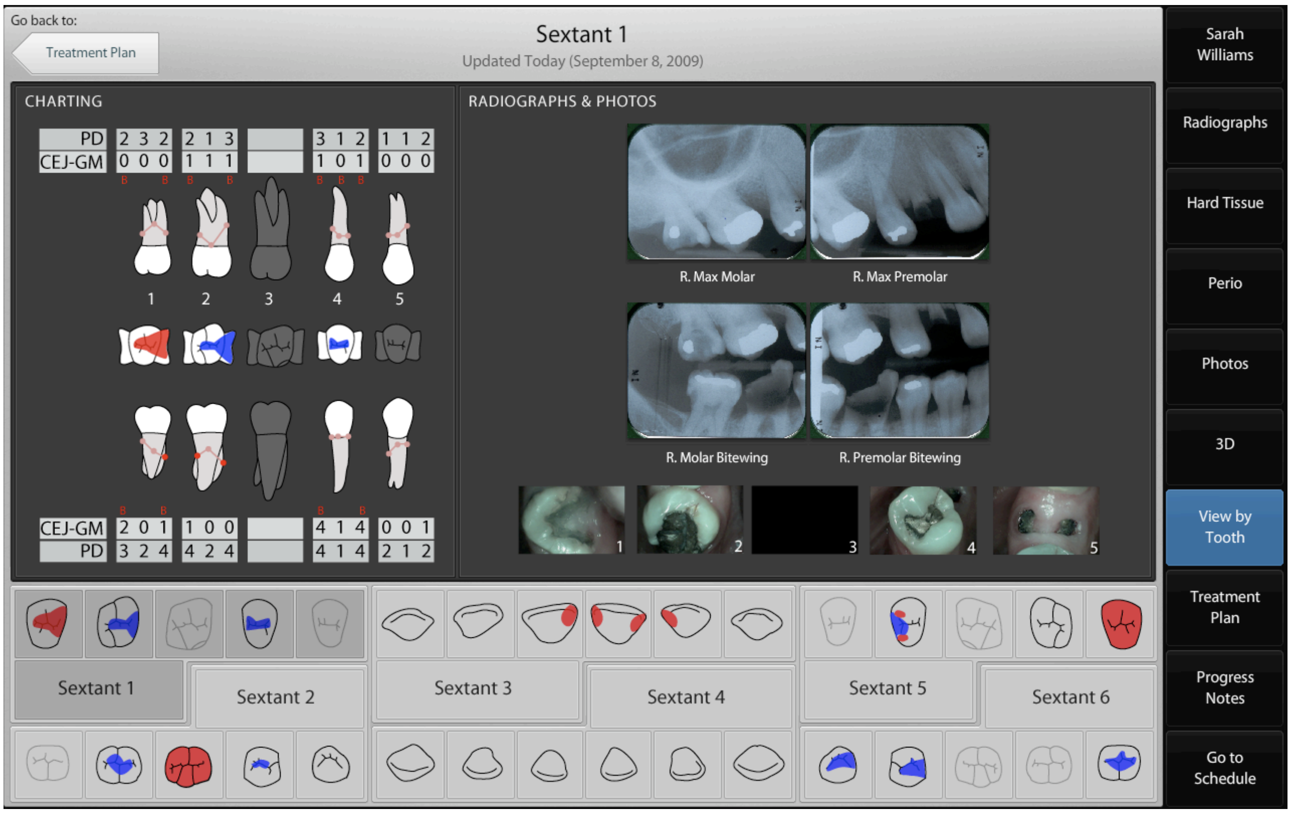


**Figure 20.** *Screenshot of a dental system prototype made in Flex by Master's students at Carnegie Mellon University.*

## JavaFX

At the time of writing, there is only one good example of a commercial application that can be found about JavaFX. The application, Indaba Music's Session Console 2.0, makes it possible for musicians to record, edit and remix audio. The application is not deployed inside the browser, but is downloaded and executed separately using Java Web Start. (Indaba Music, 2009)

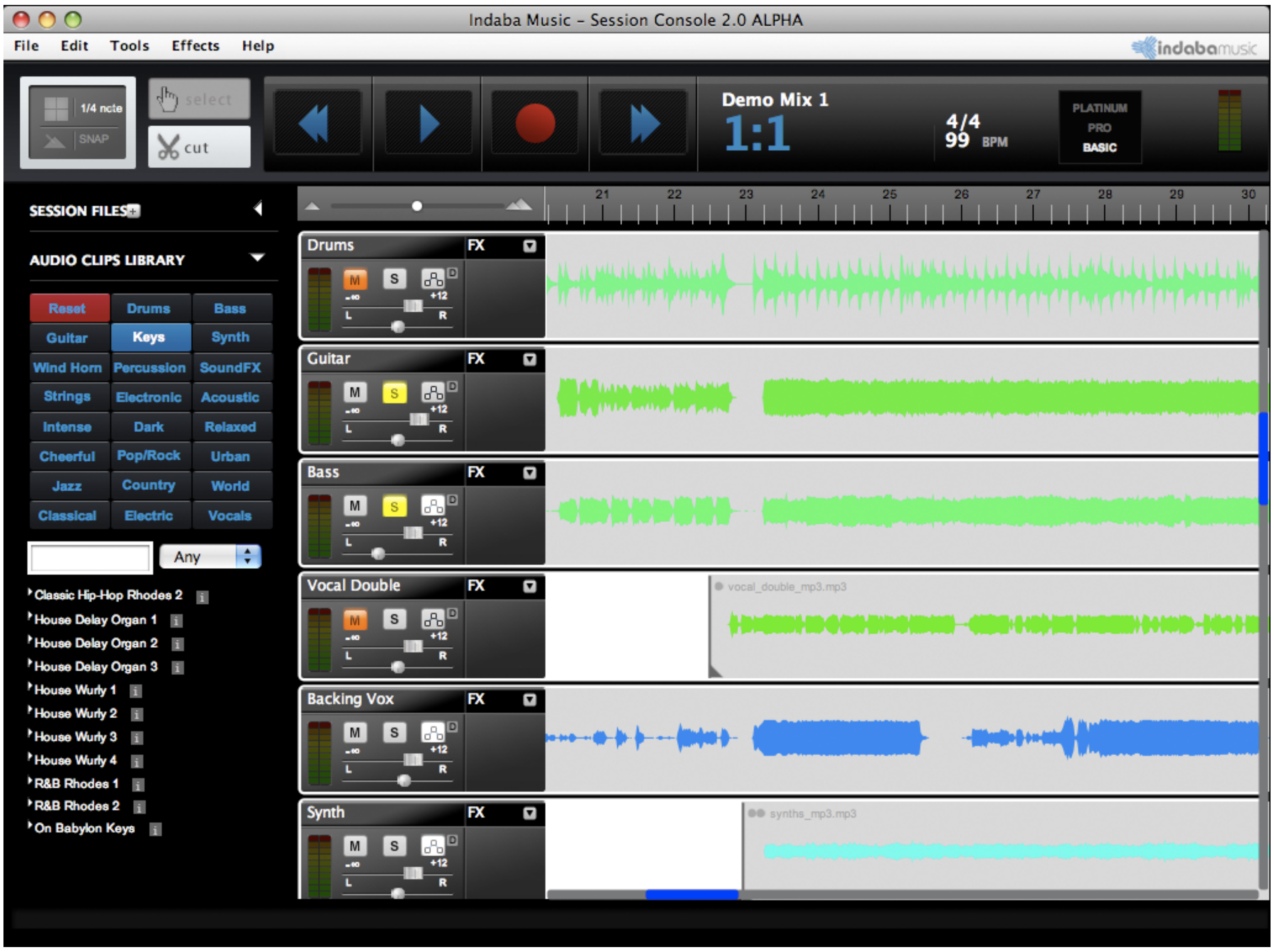


**Figure 21.** *Screenshot of Indaba Music's Session Console 2.0, an example of a commercial JavaFX application.*

At the Sun Java Store (www.java.com/en/store), which is a JavaFX application itself, several more applications can be found but these are mainly developed out of interest and enthusiasm for JavaFX. According to Nicolas Lorain at the JavaFX Marketing Team, a number of high caliber commercial web applications are currently under development at the time of writing. A press release from 10th of September 2009 reveals a JavaFX implementation from CITYTECH, which extends the web application framework JBoss Seam by providing a rich interface that visually displays operational data with geographical and financial information (PRWeb, 2009). However, neither of these applications have been seen, so little can be said about them.

In July 2009, Sun published the results from a coding competition called JavaFX Coding Challenge. The winner application is called Music Explorer FX, a visual tool for discovering new music. The user can search for an artist and listen to the songs, view images and see links to music videos, blogs and reviews etc. These applications make use of web services such as Last.fm, Flickr, YouTube and Twitter to create their content. (Sun Microsystems: JavaFX Coding Challenge, 2009)

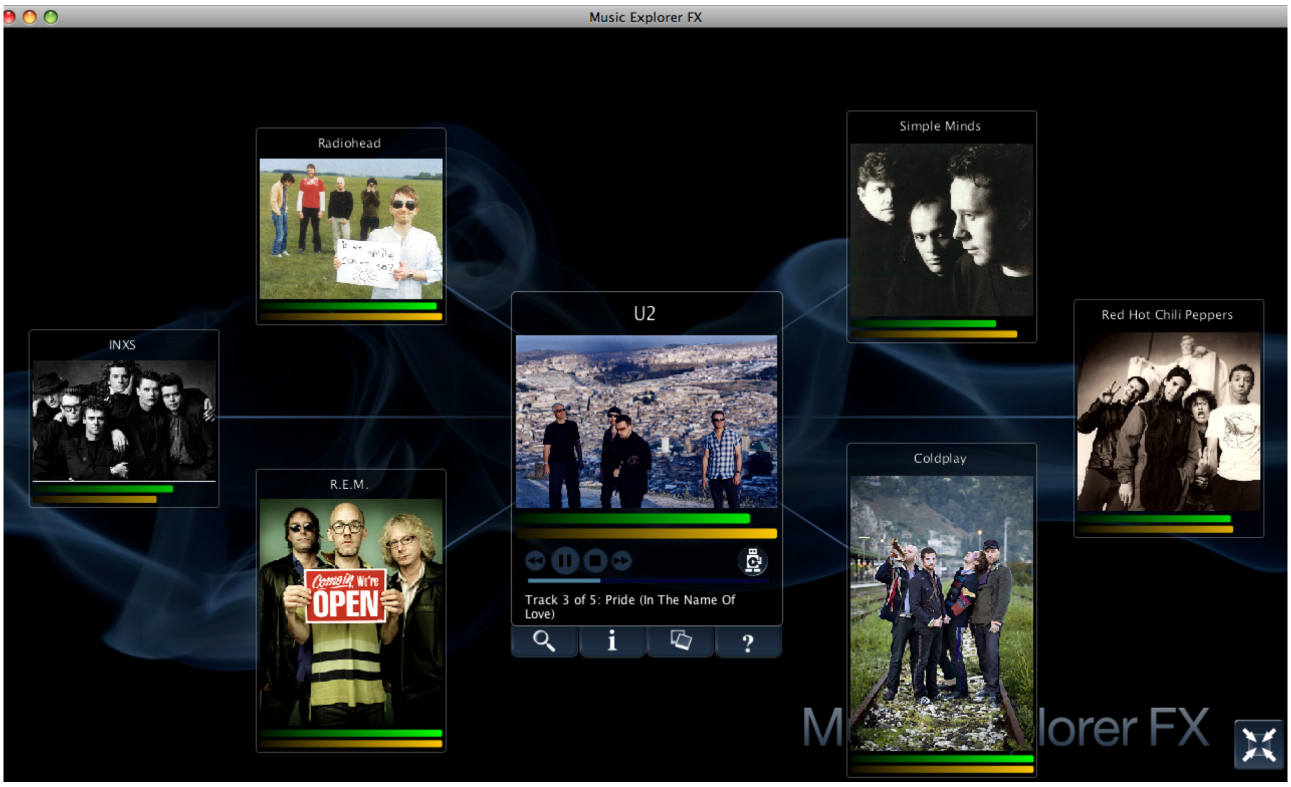


**Figure 22.** *Screenshot of Music Explorer FX, the winner application in the JavaFX Coding Challenge.*

Lifescope, an application that collects online information about individuals and their friends and visualizes them in different ways, was voted 2nd in the competition. It does not crawl the Web entirely to create this content, but instead saves information about the user after having created an account, whereby contacts, photos and blog addresses are uploaded.

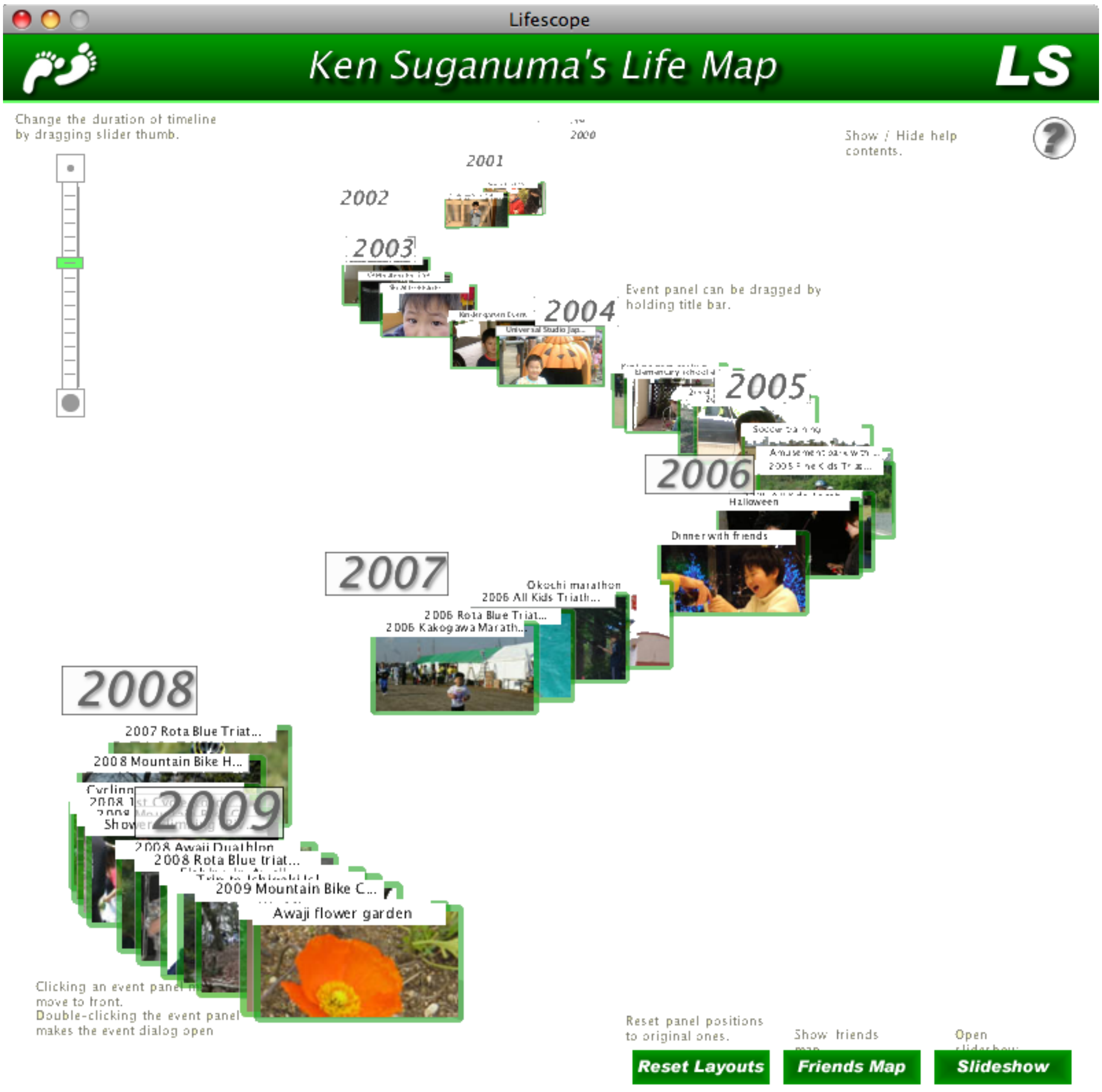


**Figure 23.** *Screenshot of Lifescope, another JavaFX implementation.*

## Silverlight

Except for the solutions and companies mentioned below, other Silverlight users include BMW, National Instruments, Norwegian TV2, Renault, Ryanair and Toyota. Microsoft also uses Silverlight on several internal projects and solutions. (Folkesson, 2009)

When Continental Airlines needed to update their call center reservations system, the agents in this system, who were power users, required the project to go beyond the limitations of traditional web applications. By choosing Silverlight, Continental Airlines claimed to reach good results concerning time-to-market, return on investments, and functionality for their agents (Microsoft: Continental Airlines, 2009).

AOL, an american global Internet service provider and media company, has built a webmail client with Silverlight, to cater for a greater demand of personalization and overall experience (Muchmore, 2008).

Each famous bar-restaurant Hard Rock Cafe has a collection of icons and souvenirs from rock 'n' roll artists, and this collection is nowadays being digitalized by photos that are published on the Web. On their website Hard Rock Memorabilia, they have made a creative Silverlight application by making use of the Deep Zoom feature. All 1000+ high-resolution photos are arranged into a big collage, which the user can zoom in on to reach great detail on each photo. Accompanied by this is information about the object and for some also video content. (Hard Rock, 2009)

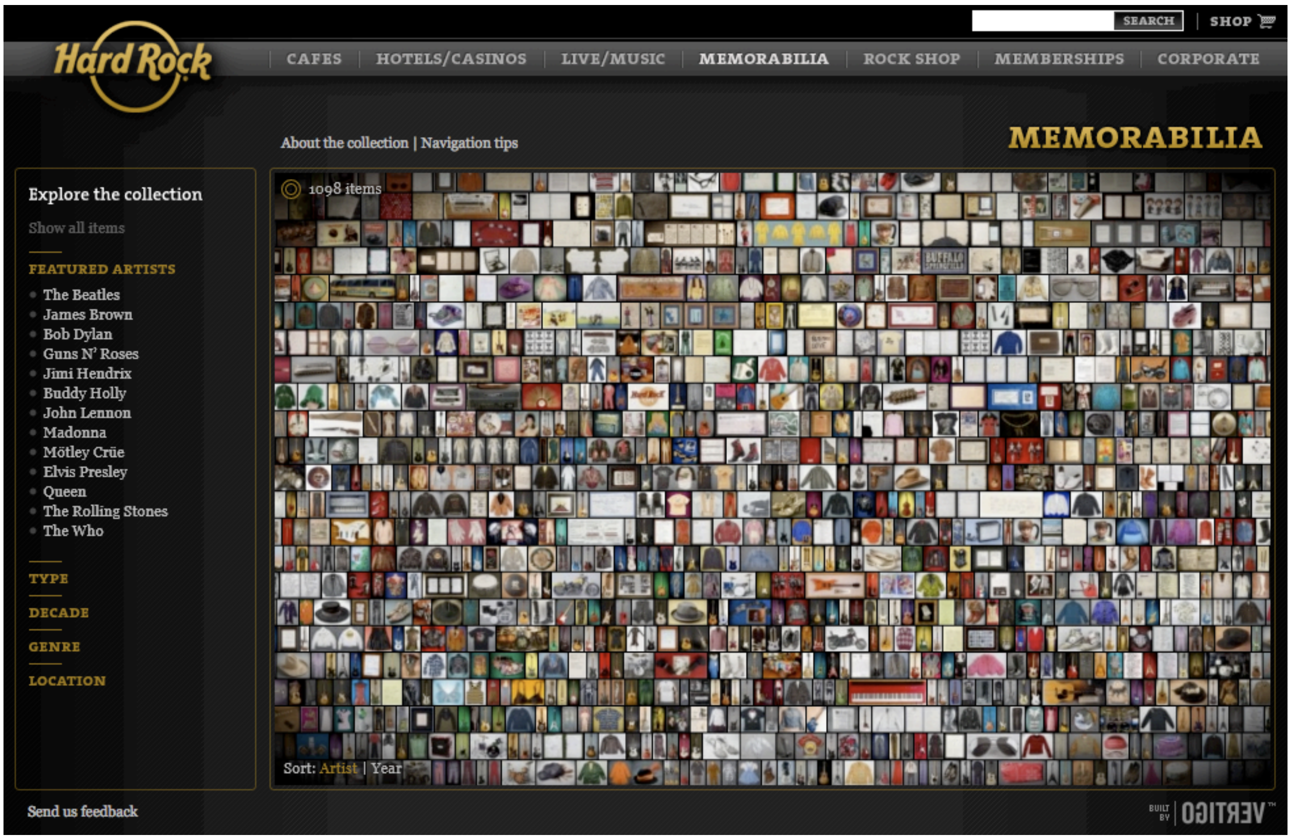
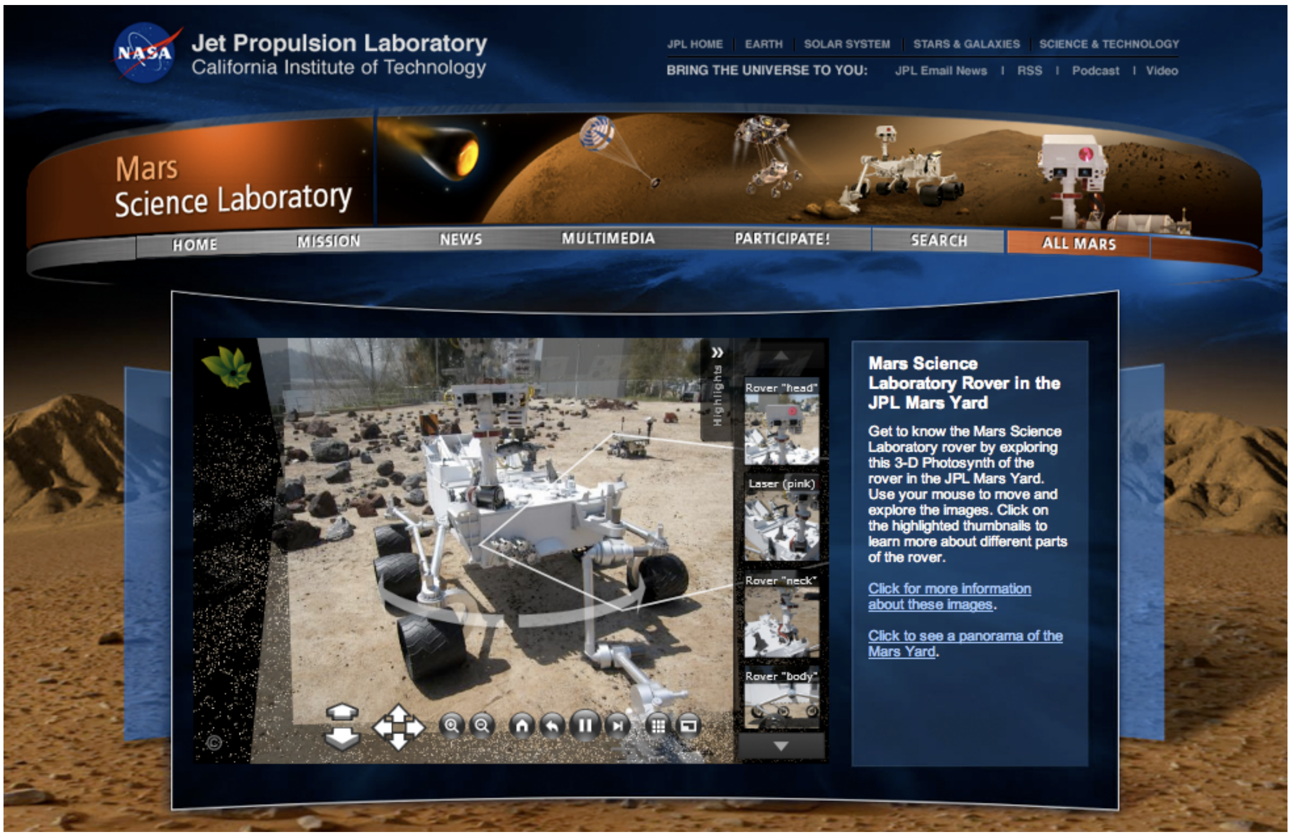


**Figure 24.** *Hard Rock Memorabilia, a Silverlight application which makes use of the Deep Zoom feature.*

Photosynth is an application from Microsoft Live Labs that generates a 3D model from several images of the same object (Microsoft: Photosynth, 2009). There is a Silverlight implementation of this application, that both Nasa and CNN use to show models of the Mars rover and the Obama Inauguration, respectively (NASA: Mars Science Laboratory, 2009) and (CNN: Obama Inauguration, 2009)


Jet Propulsion Laboratory
California Institute of Technology
JPL HOME | EARTH | SOLAR SYSTEM | STARS & GALAXIES | SCIENCE & TECHNOLOGY
BRING THE UNIVERSE TO YOU: JPL Email News | RSS | Podcast | Video
Mars Science Laboratory
HOME
MISSION
NEWS
MULTIMEDIA
PARTICIPATE!
SEARCH
ALL MARS
Highlights
Rover "head"
Laser (pink)
Rover "neck"
Rover "body"

Mars Science Laboratory Rover in the JPL Mars Yard

Get to know the Mars Science Laboratory rover by exploring this 3-D Photosynth of the rover in the JPL Mars Yard. Use your mouse to move and explore the images. Click on the highlighted thumbnails to learn more about different parts of the rover.

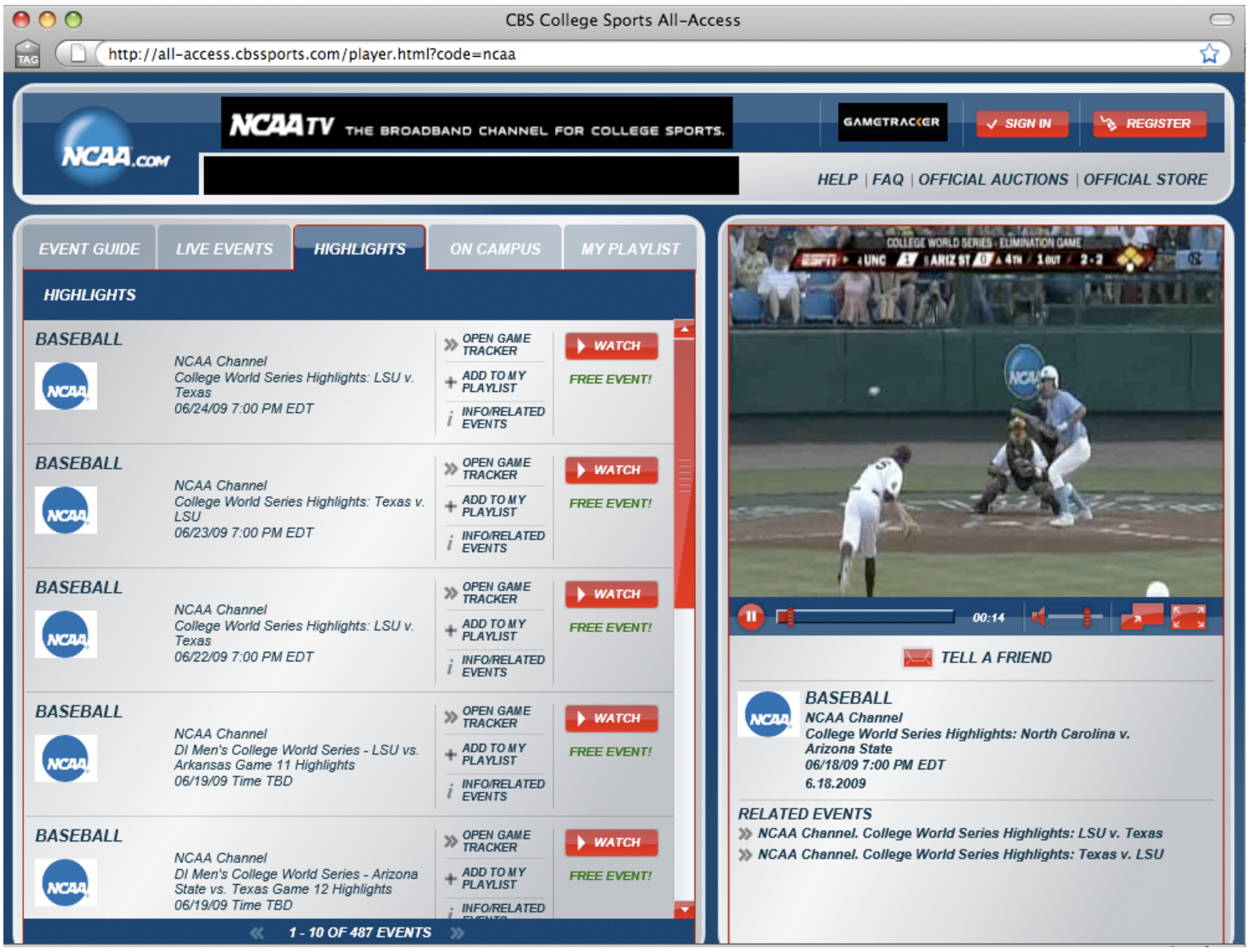
Click for more information about these images.

Click to see a panorama of the Mars Yard.

**Figure 25.** *NASA uses Silverlight on their website to show a Photosynth 3D model of their Mars rover.*

In 2008, the renown television network NBC streamed the Beijing 2008 Olympic Games live from their website www.nbcolympics.com using Silverlight, and had 52 million unique visitors. At the MIX09 conference, Microsoft and NBC announced a partnership for the 2010 Vancouver Olympics Games. NBC senior vice president Perkins Miller said that the event will be delivered using adaptive smooth streaming, supporting full 720p resolution and enabling the users to pause, rewind and view video in slow motion. (Oiaga, 2009)

NCAA, or the National Collegiate Athletic Association, is an association that organizes the athletics programs at colleges and universities in the US and Canada. On their website, they also stream live events using Silverlight (Folkesson, 2009). The following screenshot shows their web application:


CBS College Sports All-Access
http://all-access.cbssports.com/player.html?code=ncaa
NCAA.com
NCAA TV THE BROADBAND CHANNEL FOR COLLEGE SPORTS.
GAMETRACKER
SIGN IN
REGISTER
HELP | FAQ | OFFICIAL AUCTIONS | OFFICIAL STORE
EVENT GUIDE
LIVE EVENTS
HIGHLIGHTS
ON CAMPUS
MY PLAYLIST
HIGHLIGHTS
1 - 10 OF 487 EVENTS
00:14
TELL A FRIEND

| | | | |
|---|---|---|---|
| BASEBALL | NCAA Channel<br>College World Series Highlights: LSU v. Texas<br>06/24/09 7:00 PM EDT | OPEN GAME TRACKER<br>ADD TO MY PLAYLIST<br>INFO/RELATED EVENTS | WATCH<br>FREE EVENT! |
| BASEBALL | NCAA Channel<br>College World Series Highlights: Texas v. LSU<br>06/23/09 7:00 PM EDT | OPEN GAME TRACKER<br>ADD TO MY PLAYLIST<br>INFO/RELATED EVENTS | WATCH<br>FREE EVENT! |
| BASEBALL | NCAA Channel<br>College World Series Highlights: LSU v. Texas<br>06/22/09 7:00 PM EDT | OPEN GAME TRACKER<br>ADD TO MY PLAYLIST<br>INFO/RELATED EVENTS | WATCH<br>FREE EVENT! |
| BASEBALL | NCAA Channel<br>DI Men's College World Series - LSU vs. Arkansas Game 11 Highlights<br>06/19/09 Time TBD | OPEN GAME TRACKER<br>ADD TO MY PLAYLIST<br>INFO/RELATED EVENTS | WATCH<br>FREE EVENT! |
| BASEBALL | NCAA Channel<br>DI Men's College World Series - Arizona State vs. Texas Game 12 Highlights<br>06/19/09 Time TBD | OPEN GAME TRACKER<br>ADD TO MY PLAYLIST<br>INFO/RELATED | WATCH<br>FREE EVENT! |

BASEBALL
NCAA Channel
College World Series Highlights: North Carolina v. Arizona State
06/18/09 7:00 PM EDT
6.18.2009

RELATED EVENTS
NCAA Channel. College World Series Highlights: LSU v. Texas
NCAA Channel. College World Series Highlights: Texas v. LSU

**Figure 26.** *Video streaming of events and live events at NCAA.com using Silverlight.*

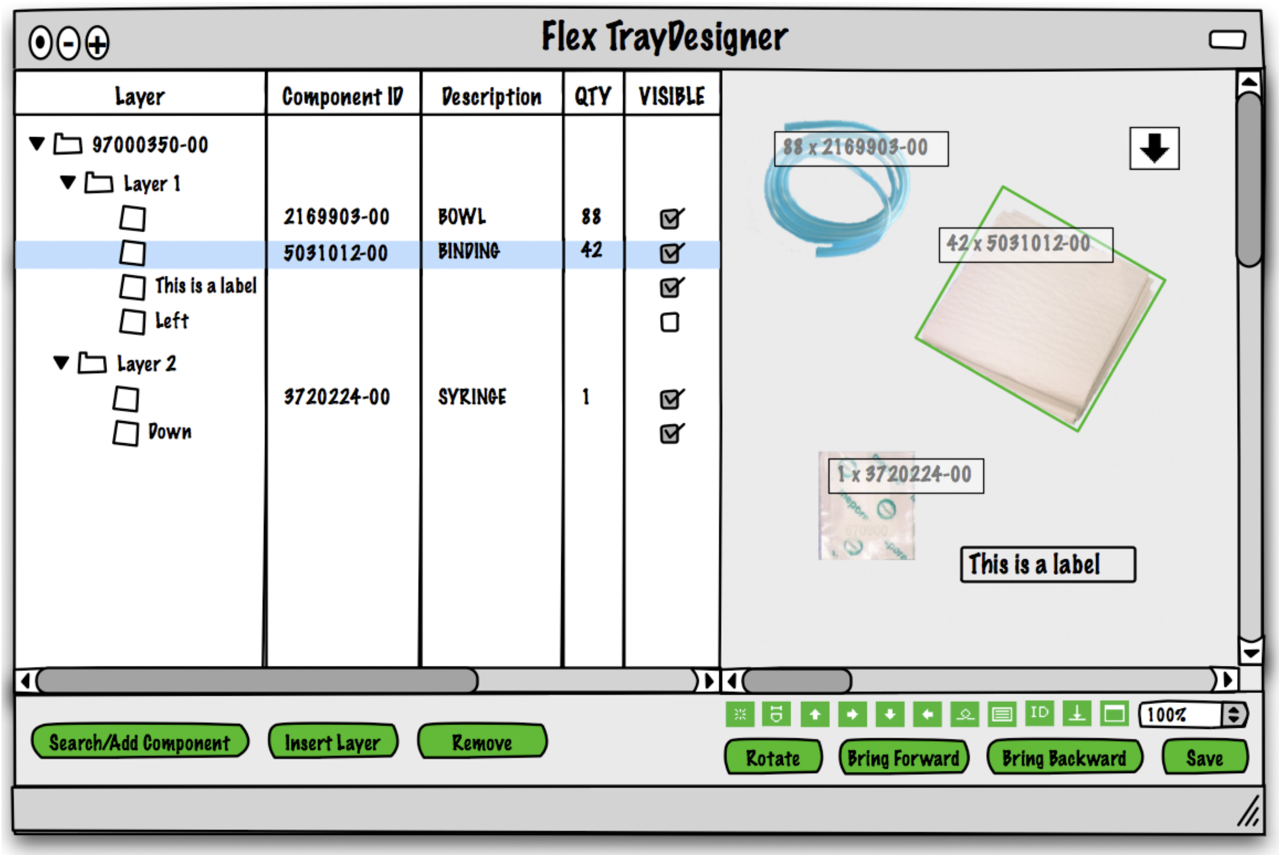
## 4.6. Prototype

To repeat the purpose of the prototype, a proof of concept application in JavaFX was created, in order to clarify the strengths and weaknesses that was identified in the theoretical part of the study, and to demonstrate its actual applicability and maturity. The prototype was based on an existing project that had been made at Capgemini in Gothenburg. This project consisted of a SAP portal, which in turn contained a Flex application. This particular application was a good subject for the matter of demonstrating JavaFX, and comparing it with Flex. The implemented prototype unveils the most interesting aspects of JavaFX, like for example graphics such as transformations and effects, as well as networking and XML parsing. Furthermore, it shows how an integration with regular Java classes can be achieved. The original Flex application, and the interpretation that laid the foundation for the prototype, is described below.

The original Flex application is a visual designer for creating *medical operation trays*, upon which components and packaging instructions can be added. A tray design consists of different layers, and each layer can contain three types of objects: components, labels and instructions about how to assemble it. Information about each component, and a tray design as a whole, is fetched and saved through the use of web services in an XML format following a certain structure. A finalized and approved design is later on converted into PDF documents that work as a blueprint for the actual production of a tray.

The design of the interface consists of three different panels. First, there is a data grid that displays the layers with their objects, and the object's properties like ID, description, quantity, position, rotation etc. Then there is a canvas panel, where the graphical representation of the objects and their properties are visualized and can be moved, rotated, scaled etc. Components and instructions consist of images, whereby the former are fetched from a content server, and the latter stored locally. Finally, below these two panels, are two panels with buttons for adding and manipulating objects, as well as adding new layers and changing the way the objects on the canvas are rendered.

The following figure shows a simplified interpretation of the original Flex application, illustrated by a low-fidelity GUI sketch:



| Layer | Component ID | Description | QTY | VISIBLE |
|---|---|---|---|---|
| ▼ 97000350-00 | | | | |
| ▼ Layer 1 | | | | |
| ☐ | 2169903-00 | BOWL | 88 | ☑ |
| ☐ | 5031012-00 | BINDING | 42 | ☑ |
| ☐ This is a label | | | | ☑ |
| ☐ Left | | | | ☐ |
| ▼ Layer 2 | | | | |
| ☐ | 3720224-00 | SYRINGE | 1 | ☑ |
| ☐ Down | | | | ☑ |

**Figure 27.** *Low-fidelity GUI sketch of the interpretation of the original Flex application.*

The right part of the bottom panel contains a series of small icons, with the functionality described in the table below:

| Icon | Description |
|---|---|
| ⁜ | Toggle highlight |
| ⌶ | Toggle horizontal/vertical alignment of component ID |
| ⬆ | Add up instruction |
| ➡ | Add right instruction |
| ⬇ | Add down instruction |
| ⬅ | Add left instruction |
| ⚲ | Add a knot |
| ▤ | Add a label |
| ID | Toggle show/hide component ID |
| ⤓ | Toggle see-through layers |
| ▭ | Toggle fullscreen |

**Table 10.** *Description of some of the GUI icons in the original Flex application.*

Except for these design aspects, the application also has some non-visual features. The data grid contains validation rules for the object's properties, for example, a component's ID cannot be edited, and the X/Y-value that describes its position cannot be negative. When a new component is to be added, a dialog pops up with a list where it is possible to search for, and add, components fetched with web services from the SAP portal in XML. Each design can contain an unlimited number of layers, and each layer can contain an unlimited number of components. On the contrary, each layer can only have up to one label and up to six instructions.

There is also certain rules about the order (Z-order) and movement of objects. First of all, layer 1 is the bottom layer and the higher the layer id, the more shallow in the physical design will an object be placed. For the objects in each layer in the data grid, the order *components*, *label*, *instructions* should always be consistent. On the bottom layer, if a label is tried to be brought backward (up), a message will say that it cannot be moved any further. If it would have been on the second layer, it would have jumped to the first valid position of the underlying layer, assuming that layer did not already have a label assigned to it.

The original application contained more features than the description above covers. The chosen and implemented features were selected according to their relevance in a proof of concept prototype to demonstrate features of JavaFX, and the rest were therefore excluded.

# 5. Result

This section contains an overview of the findings from the analysis, as well as other's work in this area.

## 5.1. Framework Comparison

This is a summary of the most important technical and non-technical differences found:

| | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| **Positive** | • Native file upload<br>• Small runtime file size<br>• Webcam & Microphone support<br>• Great design tools | • Supports many hardware platforms<br>• Runs on mobile phones<br>• Supports many software platforms<br>• Drag-to-Install<br>• Free<br>• Open-source<br>• Multi-threading | • Native deep zoom functionality<br>• Many programming languages<br>• Great tools<br>• Multi-threading |
| **Negative** | • No multi-threading<br>• Only the ActionScript language | • Accessibility<br>‣ No search engine support<br>‣ No deep linking<br>‣ No browser back/forward<br>• Controls & Components<br>‣ Lacks many, even though some can be implemented with Swing<br>‣ No DataGrid<br>‣ No native multi-page support<br>‣ No themes<br>• Big runtime file size<br>• No GPU Acceleration<br>• No built-in validation<br>• No push technology<br>• Poor tools and no visual designer | • No printing<br>• Proprietary |

**Table 11.** *Summary of important technical and non-technical differences of Flex, JavaFX and Silverlight.*

Apparently, JavaFX is lacking on many aspects in comparison to the other frameworks. This is probably due to its quite recent introduction on the RIA market. The three most essential characteristics that JavaFX is lacking are controls & components, tools and accessibility. At the time of writing, JavaFX is missing many important GUI components needed to create good applications. Even though it is possible to use components from Java Swing, the interaction between these two, as showed later with the resulting prototype, is not very stable. Many other features do not have native support in JavaFX, but can still be achieved using Java (hence many yellow cells in the comparison table). Flex and Silverlight, on the other hand, are shipped with many components and third-party themes, both open-source and commercial.

While both Flex and Silverlight offer great tools for both designers and developers, JavaFX has poor tooling support. According to Valdes (2009), this is a crucial aspect when it comes to choosing what framework to commit to. Flex has the best designer tools, while Silverlight has to be considered having the best developer tools with Visual Studio. Finally, JavaFX has some accessibility issues like no support for search engines and no ability for the user to use the back/forward buttons in the browser.

The current market and related statistics are summarized in the following table:

| Current Market and Statistics | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| Runtime Penetration (average) | 94.6% | 70.5% | 26.0% |
| Relative interest level on Google Insights for Search | 67.1% | 3.4% | 29.5% |
| Distribution of books on Amazon.com | 48.0% | 12.3% | 39.7% |
| Distribution of discussions on each framework's official forum | 44.8% | 2.9% | 52.3% |
| Distribution of entries on the forum Stack Overflow | 54.6% | 1.9% | 43.5% |
| Average distribution of positions on four different job search websites | 72.7% | 0.2% | 27.1% |
| Distribution of hits on LinkedIn.com | 43.2% | 8.8% | 48.0% |

**Table 12.** *Summary of current market and statistics for Flex, JavaFX and Silverlight.*

Flex obviously has the biggest runtime penetration, followed by JavaFX (Java) and finally Silverlight. These figures will be discussed further in the discussion, along with the result from Google Insights for Search. Half of the books on Amazon.com were about Flex, whereas about 40% covered Silverlight and 10% JavaFX. On each framework's official forum, Silverlight and Flex had almost the same amount of discussions, while JavaFX only had a few percentage. When comparing this ratio of forum activity statistics with the independent forum Stack Overflow, the percentages are about the same as with the official forums.

On job search websites, two thirds of the jobs were about Flex, one third Silverlight and barely no jobs involving JavaFX. However, the keyword selection was a bit ambiguous, so the result of this market analysis should only be considered as a rough measurement.

Finally, other advantages and disadvantages with each framework are summarized in the table below:

| | Adobe Flex | JavaFX | Silverlight |
|---|---|---|---|
| **Positive** | • Market<br>‣ Oldest and most mature framework<br>‣ Big penetration<br>‣ Many existing solutions<br>• Developed by the vendor of state-of-the-art software for designers | • Makes use of the power and security of Java<br>• Reuse of existing Java code<br>• Same code deploys on different devices | • Makes use of the power and security of .NET<br>• No learning curve for .NET developers<br>• Reuse of existing C# or VB.NET code etc.<br>• Advanced and popular on-demand and live adaptive HD smooth streaming |
| **Negative** | N/A | • Non-seamless startup procedure<br>• Slow startup time<br>• Slow execution performance<br>• Flickers and disappears when scrolling on a web page<br>• Not much documentation<br>• Small developer community<br>• Only one commercial solution<br>• Buggy components, IDE, documentation, API, browser-support and website | • Cross-browser does not work to 100%, some problems with Firefox<br>• Not a big runtime install base |

**Table 13.** *Other advantages and disadvantages with Flex, JavaFX and Silverlight.*

JavaFX also has problems with the user experience and performance. The user experience is a very important aspect of a framework's success (Valdes, 2009), and JavaFX is far behind both Flex and Silverlight in this aspect, with drawbacks as simple as annoying flickering when scrolling a web page, many dialogs when starting an application (see Appendix III) and slow startup time and execution performance.

It is highly interesting to notice that Silverlight compares very well with Flex according to Table 11 and Table 13 above. Both frameworks offer great tools, and one of the biggest benefits with Silverlight is the possibility to use any .NET language and hence the easy migration for someone from the extensive .NET developer community, as well as more or less copy-paste code reuse. However, as seen in the statistics above, its perceivably biggest obstacle is the runtime install base, which is merely around 26% as compared to Flex's 95%. This difference, which in fact is not as big as it seems due to Flash's wide use within simple animations and advertising, will be discussed further in the discussion section.

## 5.2. Other's Work

Another thesis, made by Moritz (2008), confirms many of these findings, even though the market has evolved in 1.5 years and the work was done before JavaFX had been released to the public. Moritz conducted a study of RIAs, including JavaFX, Flash and Silverlight. He concluded that Flash was the winner of fancy user interfaces, startup speed and design tools. Silverlight was considered to lack cross-platform and cross-device support, especially for mobile devices.

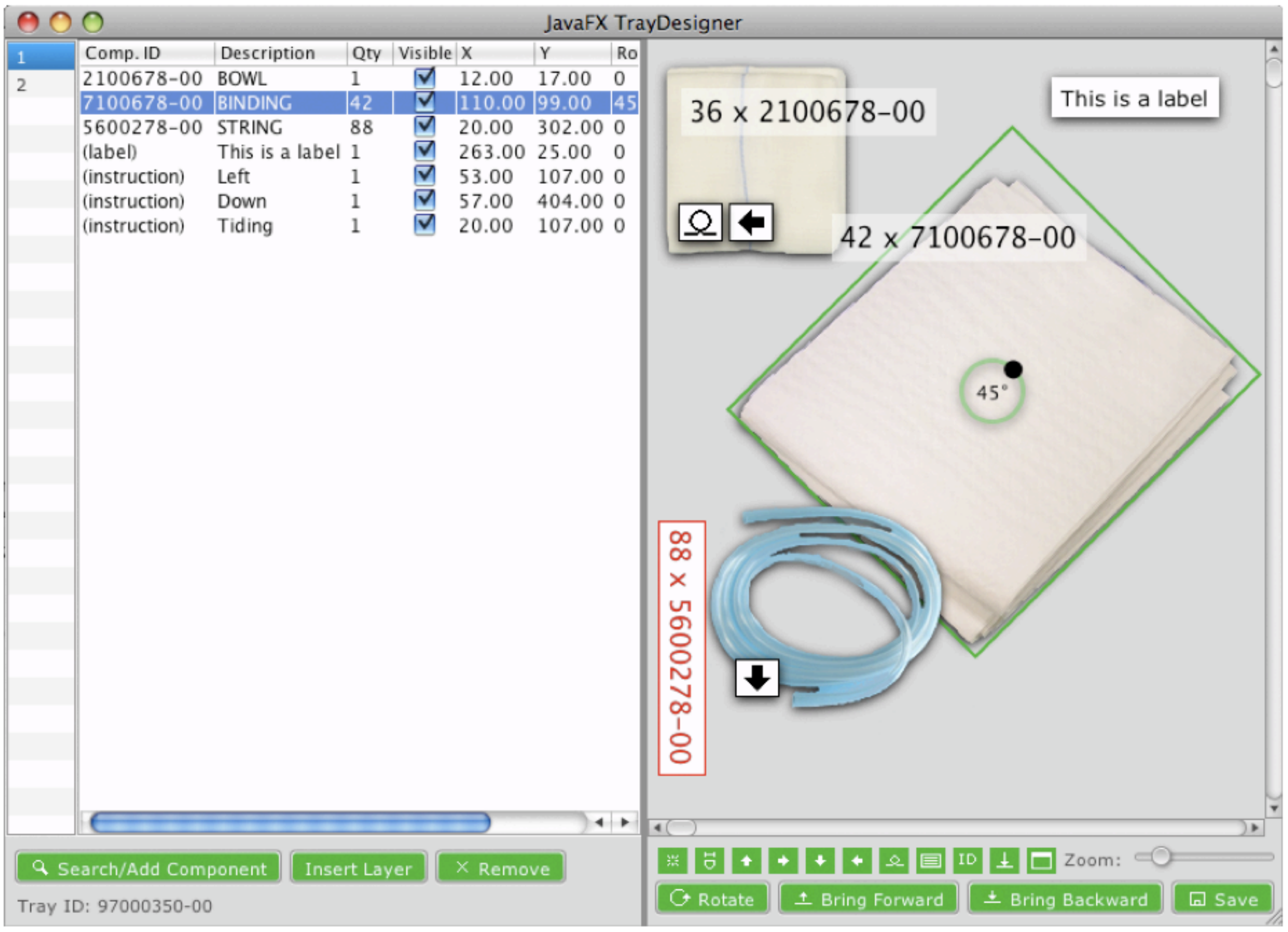
In his thesis, which put most effort in JavaFX, similar to this, many of the aspects that were considered to be crucial for JavaFX's success, have still not been corrected. This includes better tooling support and improved communication with Java. JavaFX was thought of as having the power to become a leading player on the client-server market, due to its business purpose coverage, but needed more work to be put into it in order to compete on the front-end market, which this thesis report confirms. Whether or not the belief that JavaFX still can become a leading player in RIA will be further commented on in the discussion section, along with findings in white papers from analysts at market research companies.

## 5.3. Prototype

A simplified version of the Flex Application described in the analysis was implemented in JavaFX as a proof of concept prototype. The resulting application is visually similar to the original application, with the only differences being the way to display the layers, the Visible-column and the zoom functionality. The way components are rotated was also solved in a different manner, but except for these aspects, the result did not differ much visually.

The perception of the performance and speed, including the general user experience and "flow" of this implementation, compared to the original Flex application, was more or less the same. The Flex application was not tested outside the browser using AIR, but when the prototype was deployed to the browser, they performed equally. However, JavaFX was much slower at the startup than the Flex application.

This is a screenshot of the final prototype developed in JavaFX:



| Comp. ID | Description | Qty | Visible | X | Y | Ro |
|---|---|---|---|---|---|---|
| 2100678-00 | BOWL | 1 | ✓ | 12.00 | 17.00 | 0 |
| 7100678-00 | BINDING | 42 | ✓ | 110.00 | 99.00 | 45 |
| 5600278-00 | STRING | 88 | ✓ | 20.00 | 302.00 | 0 |
| (label) | This is a label | 1 | ✓ | 263.00 | 25.00 | 0 |
| (instruction) | Left | 1 | ✓ | 53.00 | 107.00 | 0 |
| (instruction) | Down | 1 | ✓ | 57.00 | 404.00 | 0 |
| (instruction) | Tiding | 1 | ✓ | 20.00 | 107.00 | 0 |

Tray ID: 97000350-00

**Figure 28.** *Screenshot of the final JavaFX Prototype.*

The class diagram of the final prototype can be found in Appendix I, and a description of all the classes and files in Appendix II.

It is of high interest to analyze this result from a developer's perspective, in order to determine the maturity and actual applicability of JavaFX, that Capgemini was interested in. The following section will therefore elaborate on details concerning the development, and determine pitfalls encountered during the implementation process that could be avoided in future projects.

As opposed to Adobe Flex and Silverlight, JavaFX does not have a native data grid component. Instead, Java's equivalent `JTable` component had to be used for displaying the objects on each layer, since a JavaFX data grid made up from scratch would be out of scope for this thesis. By using a `JTable`, the prototype also demonstrated how an integration between JavaFX and Java could be achieved. To use a Java Swing component in JavaFX is very simple. All that needs to be done is to extend the abstract class `SwingComponent`, and return the Java component in the overridden method `createJComponent()`. In JavaFX Script, the creation of Java objects is completely seamless, i.e. the same Java code can be used in JavaFX as in Java and this is one of the strengths of the language, making it very easy to make use of existing code bases.

A drawback with the `JTable` component is that it does not allow a tree to be rendered in a column. Another alternative was to use SwingLabs `JXTreeTable`, a modified `JTable`, but such a third-party component was chosen not to be used in this thesis. Besides, it did not seem to be justified to use a tree when only two levels of text had to be visualized. Instead of a tree, a separate `ListView` was used for displaying the different layers. Nevertheless, one benefit of the `JTable` was the ability to render other Swing components inside its cells. This proved to be ideal for visualizing the boolean Visibility property in the data grid with a `CheckBox`, instead of having the user to write an "x" to specify that a certain object was visible, which was necessary in Flex. However, the latest version of Flex supports other components to be rendered inside data grid cells.

The gray divider in the middle of the window can be used for resizing the surrounding panels by dragging it. In Java, this component is known as a `JSplitPane`, and in Flex as a `HDividerBox`. JavaFX did not have an equivalent component, so at first the idea was to incorporate a `JSplitPane` from Java Swing, but this turned out to be a bad idea, so this functionality had to be implemented from scratch.

Sun claims great possibilities for interaction between Java and JavaFX, but, there is only support for Java components to be used inside of JavaFX components, and not the inverse (Fowler, 2009). Since the `JSplitPane` would be used as a top-level component, only Java components could have been used as children nodes throughout the prototype, which would contradict its purpose to demonstrate JavaFX.

Another not so straight-forward issue was to be able to call JavaFX methods from Java, since the canvas renderer had to be notified when the `JTable` had been edited. There was no built-in way of doing this, but it was solved with a clever but simple solution by using a design pattern called Template Method. A Java interface called `IEditCompleted` was specified, which in turn contained a hook method called `updateModel()`, that worked like a callback function. The method that needed to be called was implemented in the overridden hook method of a JavaFX class that extended this interface. The reason for that it all worked was that JavaFX can extend a Java class, or in this case an interface.

The mixing of Java Swing components and JavaFX components is not flawless, even the natively supported Swing components does not work seamlessly with JavaFX. At first, a `SwingComboBox`, which is supported by JavaFX by the Swing wrapper class, was going to be used as the zoom control. But when implemented, it did not behave as expected from a layout perspective and did not pop up over the canvas panel, but was only showed inside the short footer panel. A `Slider` component was used to change the zoom level instead.

Another issue was that flickering occurred when changing the size of the `JTable`, but except for this, JavaFX delivered a rather smooth experience when deployed out of the browser. On the other hand, the original Flex application had to refresh the canvas each time an attribute was changed or a component was moved back or forth, but this was not necessary in JavaFX.

In general, many of the components in JavaFX suffered from bugs. A new bug was found in the `ClipView` component, which was submitted to Sun's issue tracking system at Project Kenai. Several compiler errors that have been encountered have been submitted there as well. For example, the JavaFX `Slider` control has certain variables (specified in the API) to control the display of "tick marks" along the slider, like `majorTickUnit` and `minorTickCount`, but these do not seem to affect the control at all. Moreover, the variable `labelFormatter` takes a function as a value, which in turn should return a string according to the API, but this function is never even called.

Another observation is that JavaFX's arrays (called sequences) do not support two or more dimensions, which is rather weird because JavaFX is aimed at coding graphics and front-ends, and these concepts are often used in this context. Fortunately, a data structure implemented in Java can be used instead.

The node-based scene-graph and the graphical thinking in general is great in JavaFX. It feels intuitive and it is possible to create more advanced graphics than with Swing, in a much shorter amount of time and with less confusing code. Transformations like rotating and scaling etc. are easily added, and even if the prototype development did not include animation (time-based transformations), it should be easy due to the incorporated scene-graph and concept of time in the language.

Another nice feature of JavaFX is list comprehensions, and the following excerpt for implementing one of the design rules shows a nice example of how to make use of this:

```
public function addNode(newNode:TrayNode):Integer {
    ...
    // Validate rule: max 6 instruction nodes per layer
    if (sizeof model.nodes[node | node instanceof InstructionNode and
                                   node.layer == newNode.layer] == 6) {
        // Rule is broken
    }
    ...
}
```

This piece of code is very intuitive, and is interpreted like this: if the number of `InstructionNodes` on the same layer as the new node is 6, then do not add it.

The prototype was deployed both as a stand-alone application and to the browser, which was done easily. It was also ported to JavaFX Mobile and deployed on the NetBeans JavaFX emulator, in order to look at how how this could be achieved. This did not require much effort, and the code only had to be adapted to the mobile profile (common profile). The common profile differs in the way that certain packages cannot be used, for example `javafx.ext.swing` (since JavaFX Mobile does not support Swing), `javafx.scene.effect` and `javafx.reflect`.

The only adjustments that were made to the prototype were a basic re-layout and removing the Swing data grid (`JTable`). The use of such a component in a mobile device, or this kind of application for that matter, would be hard to justify, and in this case it was not even supported. The prototype was ported to the mobile platform for the purpose of testing, and apparently an application of this kind would need a lot of re-design in order to be usable with a mobile device. Two issues were encountered though: the layout did not behave as expected, so the transition from the desktop profile to the common was not flawless. Secondly, GIF images could not be displayed in the mobile prototype either, hence the absence of the green toolbar in the screenshot below:

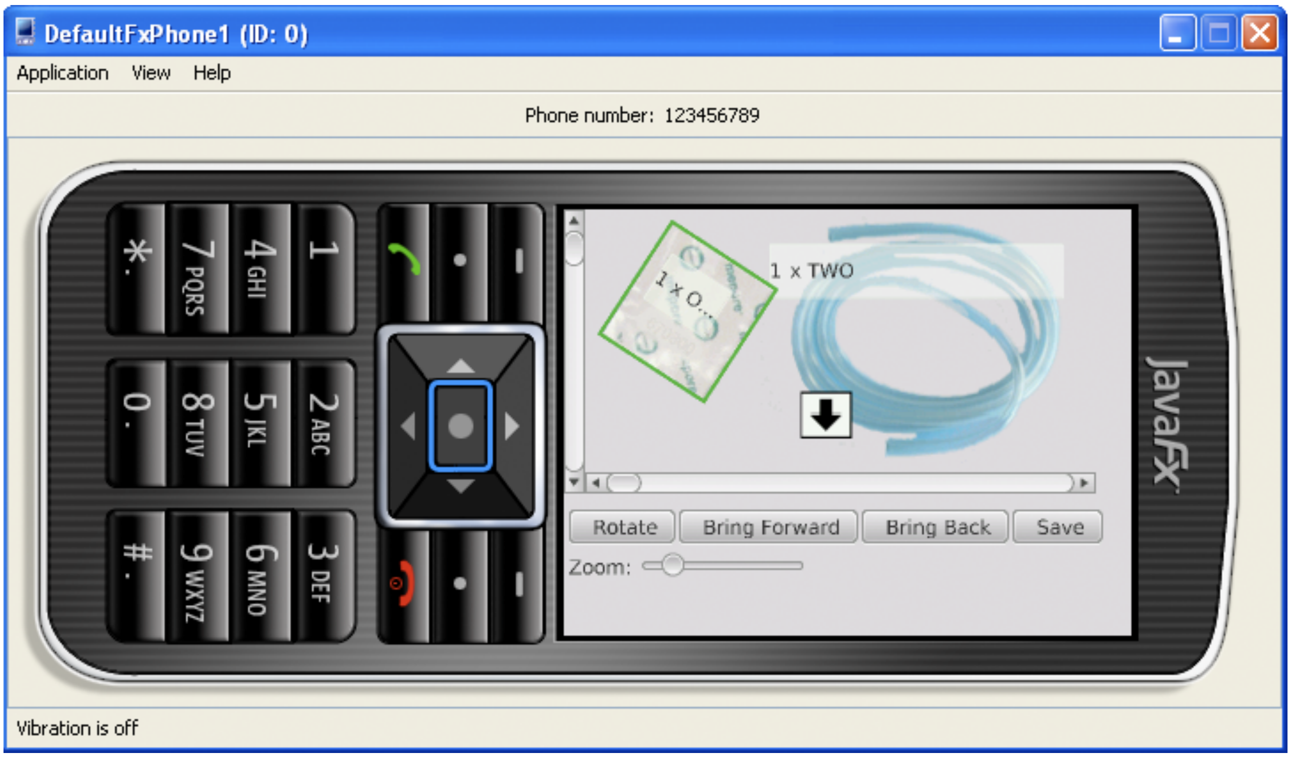


**Figure 29.** *The prototype ported to JavaFX Mobile and running in the NetBeans JavaFX mobile emulator.*

Error messages in NetBeans, which is the recommended IDE for developing JavaFX applications, were not very informative and sometimes had misspellings and repeated sentences. The JavaFX website was not even properly updated but contained tutorials and samples that only were compatible with a previous version of the runtime. Obviously, it is not only the components but also the IDE, documentation and API that needs to be improved, and JavaFX's immaturity will be further discussed in the next section.

# 6. Discussion

At the time of writing, Flex is more than 5 years old, Silverlight about 2.5 years and JavaFX only 1 year old. Since Flex has been on the market more than twice as long as Silverlight, and five times longer than JavaFX, it has gained a majority of the market share and is also the most mature framework. Silverlight, unlike Flex, has experienced a rapid development with more frequent releases though, and there is a trend showing that its release cycles are getting even shorter. The distribution of activity in forums and blogs is rather equal between Flex and Silverlight.

Google Insights for Search suggests that Flex is much more popular. The average trend on Google Insights for Search since Silverlight was first announced perfectly aligns with the trend of Flex. But, since the release of Silverlight 2 Beta in March 2008, the relative interest level for Silverlight on Google has decreased and the graph has almost flattened at the time of writing, while Flex is keeping the same trend.

However, there is an ambiguity when including the search term ”flex”, since irrelevant results can show up even if a certain category has been chosen. ”silverlight” and ”javafx” are simply much more unique keywords than ”flex”. When the search was expanded to begin at February 2004, one month before Flex was first announced, instead of September 2006, the results still show a relative level of 20 units. Perhaps this level could be taken as the margin for error, which is why the results from this search need to be considered with some reservation.

It is important to note that a big part of the 95% runtime penetration of the Flash Runtime is used for running non-Flex applications, i.e. ordinary Flash files for simple animations and advertising etc, and cannot be used to measure the level of popularity of Flex, but rather only the current market potential. The same observation can also be applied on Java, which is not only used for running JavaFX. Still, Flex and JavaFX have a lot bigger current potential than Silverlight, which has an install base of around 26%. It could also be noted that Microsoft, with around 90% of the market share for operating systems and 60% of the market share for web browsers (StatCounter, 2009), has the power to rapidly make this figure increase by shipping Silverlight with Internet Explorer and having it as a compulsory download in Windows Update.

Each of these frameworks should be seen as a work in progress, since updates are frequent and this particular field of software technology is constantly moving. However, it is still possible to comment on the future, and the Gartner analyst Valdes (2009) identifies five different aspects that will become important when tracking the evolution of RIA frameworks:

- Integration with server side
- HTML 5
- Declarative XML
- Tools and IDEs
- Support for mobile devices

In order to comment on the future of the chosen frameworks, their current state can be compared with these aspects. JavaFX's strength lies within integration with server side and support for mobile devices. It does not support declarative XML for describing UIs, but has instead its own declarative language JavaFX Script. The only option for designers is to use plugins to Adobe Photoshop and Adobe Illustrator, and JavaFX has no visual designer for the user interface in contrast to the other frameworks, which poses a hindrance to productivity.

Flex has both great tools and declarative XML, offering a great designer-developer workflow. Flex's integration with a Java back-end like the Spring Framework is realized using BlazeDS. Even JBoss Seam, which is a web application framework for combining Enterprise JavaBeans (EJBs), and JavaServer Faces (JSF), can be utilized by Flex.

Silverlight has both a strong integration with .NET back-ends, declarative XML and great IDE and design tools. Microsoft has even announced Silverlight Mobile, but an obstacle for them to overcome when venturing on the mobile market is their non-dominance, in contrast to the desktop market. Nevertheless, Microsoft makes a strong impression, covering four of these aspects. Adobe provides Flash Lite for mobile devices, but this runtime cannot currently run Flex applications and there has been no sign of Adobe having an intention to announce such support in the near future.

Finally, there is HTML 5, which progress is important to closely follow in this context, since it has been predicted to become a game-changer in web application development (Krill, 2009). In combination with improved JavaScript performance, SVG (vector graphics) and CSS3 (the next version of the markup language CSS), HTML 5 will definitely challenge these RIA frameworks. This will not happen in the next few years though, but experts at both Microsoft and Adobe think that a major breakthrough lies 5-10 years into the future (Krill, 2009). However, tracking browser support for HTML 5 is equally important, and some web browsers already cater for parts of the specification and applications are emerging which proves the power of HTML 5. An example of this is Google Wave, an online tool for real-time communication and collaboration built using GWT (Google: Wave, 2009). By demonstrating such a proof of concept application, in combination with their offline-plugin Gears, Google shows a great ability for these technologies to compete with plugin-based RIAs.

In another white paper by Valdes (2008), success in RIAs is claimed to depend more on a user-centered design/development process rather than the choice of technology. When choosing framework, users are not necessarily looking for new technology, but rather for what is more visually appealing and pleasant functionality wise, i.e. the user experience should be better.

JavaFX's startup procedure is far from seamless if it is deployed with Java Web Start and signed with unverified certificate authorities, and the usability aspect, which is part of the user experience, is very important when it comes to reason about a framework's applicability and its future. Valdes (2009), phrases this statement like this: "*usability drives user experience; user experience delivers business value*". The only commercial JavaFX application available at the time of writing, Indaba Music's Session Console 2.0, requires the user to click 5 times after the initial startup until the application can be used. These steps are visualized in Appendix III.

Other domains where RIAs developed with these frameworks can be used are within multimedia (e.g. media players), animations, games, and GUI/data visualization for business applications. When JavaFX TV is released, JavaFX can also be used for home entertainment like televisions, set-top boxes, gaming consoles and Blu-ray players.

There are some online benchmarking tools, such as Bubblemark and GUIMark, that claim to be able to measure the performance of RIAs. This is done by a simple 2D animation test with balls bouncing around inside a confined area, while continuously measuring the FPS-rate (Frames Per Second). However, there is a general opposition against these tests, mainly because web browsers might have limits on how many FPS they allow certain plugins to have (Ward, 2008). Many external factors affect such tests, like the fact that they might have been developed by different people and implemented in very different ways. They could make use of caching or other optimizations, and browsers might also execute such plugin-based applications differently. All these factors make it hard to design reasonable tests for RIAs, and there are not any real benchmarks available yet (Ward, 2008).

The security aspect of the frameworks is also hard to measure. JavaFX applications run inside a sandbox under the Java security model, which is very old and proven. Unless applications are signed, they will never get access to potentially sensitive resources, and if they are, the user needs to choose to trust the certificate authority that signed the application. According to the renown Danish computer security company Secunia, Flash/Flex has had many security vulnerabilities uncovered throughout the years (Secunia, 2009), and still has, even though it is the most mature framework. Silverlight, on the other hand, has not yet suffered from any security exploits, and since the framework is a subset of the .NET Framework, it can also make use of .NET's security model (Knipp & Valdes, 2009).

Worth mentioning about JavaFX in this study is the fact that it was the only framework that was studied in practice by implementing the prototype. It has to be considered, that by taking such a close-up view of only one framework, the description of the others might have been a bit positively biased. Even though JavaFX is the newest framework, Silverlight and Flex could also have weaknesses and bugs, even if the result did not present many. It was outside the scope of this work to make an implementation in all frameworks, but naturally, this constitutes an opportunity for future work.

This thesis' objectives were to determine the differences between Flex, JavaFX and Silverlight, both feature- and market wise. It was also to elaborate, in more detail, on JavaFX's applicability to be used as a front-end for a real application. By performing an in-depth analysis of each framework, followed up by discussions and comparisons with white papers and other's work, as well as getting hands-on experience with JavaFX by implementing a prototype and evaluating this process, these objectives can be considered fulfilled.

The methodology used for this thesis, described in detail in the beginning of this report, has been a systematic collection of relevant information. Validity is hard to reference when using a qualitative approach. However, the reliability ought to be good since the method has been carefully described and is therefore repeatable, altough specifications and the market do change rapidly. The fact that other's work, including white papers and blogs, seem to have reached similar conclusions, could probably be seen as a good indicator of reliability.

It was hard to verify that a framework *did not have* a particular feature, and such information had to been confirmed with e-mail conversations with evangelists. As previously stated, the market analysis also suffered from some ambiguities. During the prototype implementation, some problems were encountered due to lacking documentation, non-informative compilation errors, buggy components and little information on forums. Apart from these difficulties, it was fun to learn about new technologies, and especially the declarative scripting language JavaFX Script and it's many new ways of thinking of, and approaching, programming problems.

# 7. Conclusion

Having been on the market twice as long as Microsoft Silverlight, and five times longer than JavaFX, Adobe Flex is not surprisingly the most mature framework of this comparison. However, the technical and non-technical characteristics of Flex and Silverlight do not differ much. In fact, Microsoft has rather quickly established itself as the main competitor to Flash and Flex in the context of plugin-based RIA frameworks. The biggest difference is in the runtime market penetration, whereby Flex (Flash) has 95% and Silverlight 26% of the market share. JavaFX has been on the market for only one year and is far behind the other frameworks and still has many shortcomings and weaknesses.

If JavaFX had more stable components and a greater selection, it would be a good alternative for Java developers instead of using Swing when developing front-ends to stand-alone applications. But for public and commercial web applications, when the clients use a variety of different environments and settings, it would currently not be a good choice due to the non-seamless startup procedure with security alerts and slow startup time. It appears that JavaFX's user experience is not enough to compete with neither Silverlight nor Flex.

To summarize the prototype implementation, custom components had to be implemented and existing Java components tweaked to work with JavaFX, due to lack of existing ones like e.g. data grids and split panes. Many components, including the API, were not well-built and contained several errors and flaws. The integration between JavaFX and Java is easily done, but unfortunately not very stable and needs to be improved. At its current state, JavaFX applications moreover have problems with bad performance and crashing the browser.

While Flex and Silverlight are compared in the latest RIA white paper published in June 2009 by the market research company Gartner, JavaFX is not even mentioned (Knipp & Valdes, 2009). Moritz (2008), claimed that JavaFX had the power to become a leading player on the RIA market, but 1.5 years later and after the results unveiled in this thesis report, this seems unlikely. JavaFX might find a possible use within applications aimed to a limited group of users, in intranets or environments that are identical and configured centrally, in order to work more seamlessly. But in its current state, it is not recommended for commercial applications where user experience and accessibility is of high importance.

Flex descended from Flash, an engine built for animation and graphics, and such features works very smooth without flickering and are one of it's strengths. Flex and Silverlight compares well even for existing solutions, but the Flex applications studied in this report have turned out to be a bit more robust. Both frameworks have proved to be able to deliver truly rich experiences. JavaFX only has one commercial application available, which is an "alpha" release, but Sun claims that more high-caliber applications are under development.

Flex, being the dominant framework, is better suited than the other frameworks for public commercial applications where many users needs to be reached, due to the market penetration of the Flash Player and its well established technology. Knipp & Valdes (2009), arrive at the same conclusion, but furthermore claim that Silverlight would be a better choice if the developers already have competence in .NET, and not too much concern about accessibility. Silverlight is definitely able to compete with Flex. It has experienced a rapid development with frequent releases, and the fact that it is backed up by great tools, almost copy-paste integration with .NET and has got a wide developer base speaks for its future evolution to compete with being one of the most powerful plugin-based RIA frameworks on the market.

The relatively sparse usage of JavaFX, as compared to the other frameworks, is confirmed by looking at the current market. There are barely any results on job search websites, and not much information to be found about it online nor on forums. In this context, the figures of Flex and Silverlight are rather similar, but the result of the market analysis should only be taken as a rough guideline, since there were some ambiguities with the keyword selection.

All three frameworks use a declarative way of defining interfaces, which is good for many reasons. It makes code exchange between tools much easier, which leads to more advanced and better tooling support. Its declarative format also improves communication between designers and developers, since the code is easy to understand and directly reflects the visual result. Flex's MXML and Silverlight's XAML are both XML-based markup languages that are quite similar syntactically. JavaFX Script, which is not XML-based but still declarative in its nature, is not as intuitive but nevertheless easy to understand.

By describing user interfaces with declarative XML and XML-like formats, the frameworks can cater for better tooling support and an enhanced designer-developer workflow, which is important for the productivity. With such a workflow, designers can create functionality without writing any code, and later on hand it over to developers that can add additional features. JavaFX does not have a visual designer, although a similar tool has been previewed but not yet released. Flex has Flash Catalyst, and Silverlight has Microsoft Expression Blend, both which are great designer tools. Silverlight developers also have the benefit of using the award-winning IDE Microsoft Visual Studio. However, these tooling benefits do not come without a price. Both Flex and Silverlight have stricter licenses and more associated costs than JavaFX.

To again highlight the findings, JavaFX does currently not seem to be able to compete with neither Silverlight nor Flex. The framework needs more components, better tools, better accessibility and most of all being able to deliver a better user experience with a more stable performance, as well as a seamless startup procedure. Silverlight would need to gain a bigger market penetration to compete with Flex, and needs proof of concept examples of it's ability to run on mobile devices to succeed in all areas of RIAs. It's main competitor Flex is good in many aspects, but would also need to focus on the mobile market. This market, where Sun is already promoting JavaFX, will become more important in the future, together with home entertainment systems.

When analyzing what technology to choose, the importance of project requirements needs to be stressed. In order to make the best decisions, the following type of questions need to be considered:

- Who are the users?
- What do they expect?
- What would they appreciate?
- What development expertise do we already possess?
- Can we reuse code bases?

These type of questions are crucial in many kinds of development projects, including the field of Rich Internet Applications. It is also important to reason about whether to use JavaScript/ AJAX-based vs. plugin-based frameworks, since they have different advantages and disadvantages. The former is not within the scope of this thesis to discuss in detail, but such frameworks are likely to be better suited depending on the requirements. A combination can also be successful, as proven with for example YouTube and Facebook, two web sites that mix AJAX and Flash technology.

Finally, the progress of the next generation web markup language HTML 5, which future was discussed in the previous section, will be important to follow. This upcoming standard has been predicted to become a game-changer in web application development (Krill, 2009). According to its specification, it will be covering most of the differences between JavaScript/ AJAX-based- and plugin-based RIA frameworks (W3C, 2009), and in combination with JavaScript, CSS3 and SVG, it could become very powerful. As an upcoming standard and a work in progress, its time to market is not very short, and experts at both Adobe and Microsoft estimate that a major breakthrough lies 5-10 years into the future (Krill, 2009).

New techniques and technologies are frequently born, continuously creating new opportunities. The Internet is a giant sandbox where creativity prosper and ideas from ingenious individuals are heard every day. The result uncovered in this Master's thesis report will help software developers and architects to decide what plugin-based RIA framework to choose, at least for the moment. It is up to the users and the community to decide what framework that will continue to stay on top, but one thing is for sure - there will always be more than one option.

# References

Adobe: AIR FAQ. (2009). *Developer FAQ - Adobe AIR - Adobe Learning Resources.*
Available: <http://learn.adobe.com/wiki/display/air/Developer+FAQ>. Accessed: 25 June 2009.

Adobe: BlazeDS. (2009). *BlazeDS - BlazeDS - Confluence.*
Available: <http://opensource.adobe.com/wiki/display/blazeds/BlazeDS>. Accessed: 13 June 2009.

Adobe: Customer Showcase. (2009). *Adobe - Customer Showcase.*
Available: <http://www.adobe.com/cfusion/showcase/index.cfm>. Accessed: 7 September 2009.

Adobe: Flash. (2009). *Adobe Labs - Adobe Flash Platform Technologies.*
Available: <http://labs.adobe.com/technologies/flash>. Accessed: 25 June 2009.

Adobe: Flash Builder. (2009). *Adobe Labs - Adobe Flash Builder 4.*
Available: <http://labs.adobe.com/technologies/flashbuilder4>. Accessed: 22 June 2009.

Adobe: Flash Builder - What's New. (2009). *What's new in Flash Builder 4 beta | Adobe Developer Connection.*
Available: <http://www.adobe.com/devnet/flex/articles/flashbuilder4_whatsnew.html>. Accessed: 29 June 2009.

Adobe: Flash Catalyst. (2009). *Adobe Labs - Adobe Flash Catalyst.*
Available: <http://labs.adobe.com/technologies/flashcatalyst>. Accessed: 13 June 2009.

Adobe: Flash Penetration. (2009). *Adobe - Flash Player Version Penetration.*
Available: <http://www.adobe.com/products/player_census/flashplayer/version_penetration.html>. Accessed: 2 July 2009.

Adobe: Gumbo. (2009). *Gumbo - Flex SDK - Confluence.*
Available: <http://opensource.adobe.com/wiki/display/flexsdk/Gumbo>. Accessed: 15 June 2009.

Adobe: Gumbo Themes. (2009). *Gumbo Themes - Flex SDK - Confluence.*
Available: <http://opensource.adobe.com/wiki/display/flexsdk/Gumbo+Themes>. Accessed: 18 June 2009.

Adobe: LiveCycle. (2009). *Adobe - LiveCycle Enterprise Suite.*
Available: <http://www.adobe.com/products/livecycle>. Accessed: 18 June 2009.

Adobe: Marketplace. (2009). *Adobe Marketplace - Welcome to the Adobe Marketplace.*
Available: <http://www.adobe.com/cfusion/marketplace>. Accessed: 7 September 2009.

Bhattacharya, Arindam. arindam.bhattacharya@sun.com. (2009). *JavaFX Questions (technical & market).*
[E-mail] Message to Carl-David Granbäck (carldavid.granback@gmail.com). Sent 19 June 2009 15:14.

CNN: Obama Inauguration. (2009). *CNN.com - Special Reports - The 44th President - The Moment.* CNN.
Available: <http://www.cnn.com/SPECIALS/2009/44.president/inauguration/themoment>. Accessed: 14 September 2009.

Coenen, Frans. (1999). *Characteristics of declarative programming languages.*
Available: <http://www.csc.liv.ac.uk/~frans/OldLectures/2CS24/declarative.html#detail>. Accessed: 7 June 2009.

Creanga, Cornel. (2009). *Adobe Flex and Java.* Javaforum Conference. Gothenburg. 27 May 2009.

Eclipse4SL. (2009). *Eclipse Tools for Microsoft Silverlight.*
Available: <http://www.eclipse4sl.org>. Accessed: 19 August 2009.

Field, Robert. (2009). *JavaFX Language Reference*. Sun Microsystems.
Available: <openjfx.java.sun.com/current-build/doc/reference/JavaFXReference.html>. Accessed: 10 June 2009.

Folkesson, Robert. (2009). *Robert Folkesson - Några intressanta Silverlight-lösningar*.
Available: <http://blogs.msdn.com/robf/archive/2009/03/30/n-gra-intressanta-silverlight-l-sningar.aspx>. Accessed: 14 September 2009.

Fowler, Amy. (2009). *Insider's Guide to Mixing Swing and JavaFX | Java.net.*
Available: <weblogs.java.net/blog/aim/archive/2009/06/insiders_guide.html>. Accessed: 15 September 2009.

Garrett, J. Jesse. (2005). *Ajax: A New Approach to Web Applications.*
Available: <http://www.adaptivepath.com/ideas/essays/archives/000385.php>. Accessed: 18 September 2009.

Google: Flash Indexing. (2009). *Official Google Webmaster Central Blog: Flash indexing with external resource loading.*
Available: <http://googlewebmastercentral.blogspot.com/2009/06/flash-indexing-with-external-resource.html>. Accessed: 22 June 2009.

Google: GWT. (2009). *Product Overview - Google Web Toolkit - Google Code*.
Available: <http://code.google.com/webtoolkit/overview.html>. Accessed: 21 September 2009.

Google: Improved Flash Indexing. (2008). *Official Google Webmaster Central Blog: Improved Flash indexing.*
Available: <http://googlewebmastercentral.blogspot.com/2008/06/improved-flash-indexing.html>. Accessed: 22 June 2009.

Google: Insights for Search. (2009). *Google Insights* for Search.
Available: <http://www.google.com/insights>. Accessed: 3 September 2009.

Google: Wave. (2009). *About Google Wave*.
Available: <http://wave.google.com/help/wave/about.html>. Accessed: 3 October 2009.

Haas, Hugo. (2004). *Web Services Glossary.* The World Wide Web Consortium (W3C).
Available: <http://www.w3.org/TR/ws-gloss>. Accessed: 9 June 2009.

Hard Rock. (2009). *Hard Rock Memorabilia*.
Available: <http://memorabilia.hardrock.com>. Accessed: 8 September 2009.

Hashemi, Sam. Kase, Jaanus. Weber, Jackie & Williams, Elliott. (2008). *DMD Project - A prototype touch-based dental system*. Available: <http://www.dmdproject.com>. Accessed: 14 September 2009.

Heiss, J. Janice. (2008). *Encode Once, Play Anywhere: An Interview With JavaFX Media Lead Engineer Tony Wyant*.
Available: <http://java.sun.com/developer/technicalArticles/Interviews/wyant_jfx.html>. Accessed: 14 June 2009.

Indaba Music. (2009). *INDABA MUSIC ANNOUNCES DIGITAL AUDIO WORKSTATION BUILT ON SUN MICROSYSTEMS' JAVAFX PLATFORM - Indaba Music*. Available: <http://www.indabamusic.com/corporate/pr/20090603-indaba-music-announces-digital-audio-workstation-built-on-javafx>. Accessed: 8 September 2009.

InfoWorld. (2009). *2009 Technology of the Year Awards: App Dev | Developer World - InfoWorld.*
Available: <http://www.infoworld.com/d/developer-world/2009-technology-year-awards-app-dev-911>. Accessed: 19 August 2009.

Internet World Stats. (2009). Available: <http://www.internetworldstats.com>. Accessed: 7 September 2009.

Kaul, Jeet. (2008). *JavaFX - the road ahead.*
Available: <http://blogs.sun.com/meetjeet/entry/javafx_the_road_ahead>. Accessed: 10 June 2009.

Knipp, Eric. Valdes, Ray. (2009). Navigating the Ajax vs. 'Heavy RIA' Dilemma. Gartner, Inc.

Krill, Paul. (2009). *HTML 5: Could it kill Flash or Silverlight? | Developer World - InfoWorld*.
Available: <http://www.infoworld.com/d/developer-world/html-5-could-it-kill-flash-and-silverlight-291>.
Accessed: 21 September 2009.

Microsoft: Continental Airlines. (2009). *Microsoft Case Studies: Continental Airlines*.
Available: <http://www.microsoft.com/casestudies/Case_Study_Detail.aspx?casestudyid=4000004918>.
Accessed: 14 September 2009.

Microsoft: Expression Blend. (2009). *Blend Overview < Product Information | Microsoft Expression*.
Available: <http://www.microsoft.com/expression/products/Blend_Overview.aspx>. Accessed: 19 August 2009.

Microsoft: IIS Smooth Streaming. (2009). *IIS Smooth Streaming Technical Overview - Microsoft White Paper*.
Available: <http://www.microsoft.com/downloads/details.aspx?FamilyID=03D22583-3ED6-44DA-8464-B1B4B5CA7520&displaylang=en>. Accessed: 21 August 2009.

Microsoft: .NET RIA Services. (2009). *Microsoft .NET RIA Services July 2009 Preview*.
Available: <http://www.microsoft.com/downloads/details.aspx?FamilyID=76bb3a07-3846-4564-b0c3-27972bcaabce>. Accessed: 20 August 2009.

Microsoft: Photosynth. (2009). *Microsoft Photosynth*.
Available: <http://www.photosynth.net>. Accessed: 14 September 2009.

Microsoft: Silverlight (2009). *Get Started : The Official Microsoft Silverlight Site*.
Available: <http://www.silverlight.net/getstarted/silverlight3/default.aspx>. Accessed: 19 August 2009.

Microsoft: Silverlight For Mobile (2009). *Silverlight for mobile : The Official Microsoft Silverlight Site*.
Available: <http://www.silverlight.net/learn/mobile>. Accessed: 3 October 2009.

Microsoft: Visual Studio. (2009). *Visual Studio 2008 Professional Edition with Intellisense - Product Information*.
Available: <http://www.microsoft.com/visualstudio/en-us/products/professional/default.mspx>. Accessed: 19 August 2009.

Microsoft: XAML. (2009). *XAML Overview*.
Available: <http://msdn.microsoft.com/en-us/library/ms752059.aspx>. Accessed: 18 August 2009.

Moritz, Florian. (2008). *Rich Internet Applications (RIA): A Convergence of User Interface Paradigms of Web and Desktop Exemplified by JavaFX*. Diploma Thesis, University of Applied Science Kaiserslautern, Germany. Available: <http://www.flomedia.de/diploma/documents/DiplomaThesisFlorianMoritz.pdf>. Accessed: 2 October 2009.

Muchmore, Michael. (2008). *AOL Introduces Silverlight-based Web Mail Client*. PC Magazine.
Available: <http://www.pcmag.com/article2/0,2817,2273654,00.asp>. Accessed: 8 September 2009.

NASA: Mars Science Laboratory. (2009). *Mars Science Laboratory: Photosynth*. NASA.
Available: <http://marsprogram.jpl.nasa.gov/msl/multimedia/interactives/photosynth/index.html>. Accessed: 14 September 2009.

Neto, Silveira. (2008). *Inkscape and JavaFX working together*.
Available: <http://silveiraneto.net/2008/11/21/inkscape-and-javafx-working-together>. Accessed: 14 June 2009.

Norbye, Tor. (2009). *Planet Cast Episode Three - Danny Coward interviews Tor Norbye on JavaFX*.
Available: <http://mediacast.sun.com/users/dannycoward/media/pc003/details>. Accessed: 15 June 2009.

Oiaga, Marius. (2009). *Silverlight to Stream the Next Olympics Fully in HD - The 2010 Winter Olympic Games - Softpedia*.
Available: <http://news.softpedia.com/news/Silverlight-to-Stream-the-Next-Olympics-Fully-in-HD-107265.shtml>. Accessed: 14 September 2009.

O'Reilly, Tim. (2005). *What Is Web 2.0*. O'Reilly Media.
Available: <http://oreilly.com/web2/archive/what-is-web-20.html>. Accessed: 17 September 2009.

Parleys. (2009). Home - Parleys - Parleys.com - The Next Generation RIA eLearning Platform.
Available: <http://www.parleys.com>. Accessed: 7 September 2009.

PRWeb. (2009). *CITYTECH's Innovative JavaFX Dashboard Piques Interest at JBoss World.*
Available: <www.prweb.com/releases/citytechinc/javafx/prweb2854564.htm>. Accessed: 14 September 2009.

RIAStats.com. (2009). *Rich Internet Application Statistics*. Travis Collins, DreamingWell.com.
Available: <http://www.riastats.com>. Accessed: 1 July 2009.

Rigby, Bill. (2009). Reuters. *Gates sees tech helping U.S. out of recession.*
Available: <www.reuters.com/article/technologyNews/idUSTRE54K5XM20090521>. Accessed: 14 August 2009.

Ritter, Simon. (2009). *JavaFX: The Platform for Rich Internet Applications*. Sun Microsystems.
Available: <http://www.jfokus.se/jfokus09/preso/jf09-KeynoteJavaFXThePlatform4RIA.pdf>. Accessed: 25 September 2009.

Rogowski, Ron. (2007). The Business Case For Rich Internet Applications. Forrester Research, Inc.

Secunia. (2009). Adobe Flash Player Multiple Vulnerabilities - Secunia Advisories - Vulnerability Information - Secunia.com. Available: <http://secunia.com/advisories/35948>. Accessed: 10 October 2009.

Snow, Mike. (2008). *Silverlight Tip of the Day #19: Using Isolated Storage - Silverlight Tips of the Day - Blog by Mike Snow*.
Available: <http://silverlight.net/blogs/msnow/archive/2008/07/16/tip-of-the-day-19-using-isolated-storage.aspx>. Accessed: 20 August 2009.

StatCounter. (2009). *StatCounter Global Stats*.
Available: <http://gs.statcounter.com>. Accessed: 6 October 2009.

StatOwl.com. (2009). StatOwl.com - Statistical analysis and market research of Internet usage trends.
Available: <http://www.statowl.com>. Accessed: 1 July 2009.

Sun Microsystems: JavaFX Coding Challenge. (2009). *JavaFX Coding Challenge.*
Available: <http://www.javafx.com/challenge>. Accessed: 14 September 2009.

Sun Microsystems: JavaFX FAQ. (2009). *JavaFX FAQs | How to and General Questions About Java FX*.
Available: <http://www.javafx.com/faq>. Accessed: 7 June 2009.

Sun Microsystems: JavaFX Production Suite. (2009). *Getting Started With JavaFX Production Suite.*
Available: <http://javafx.com/docs/gettingstarted/production_suite>. Accessed: 1 September 2009.

Sun Microsystems: JavaOne. (2009). *2009 JavaOne Conference - General Session Details and Video Replays*.
Available: <http://java.sun.com/javaone/2009/general_sessions.jsp>. Accessed: 25 September 2009.

Valdes, Ray. (2008). MarketScope for Ajax Technology and RIA Platforms. Gartner, Inc.

Valdes, Ray. (2009). Key Issues in Rich Internet Application Platforms and User Experience. Gartner, Inc.

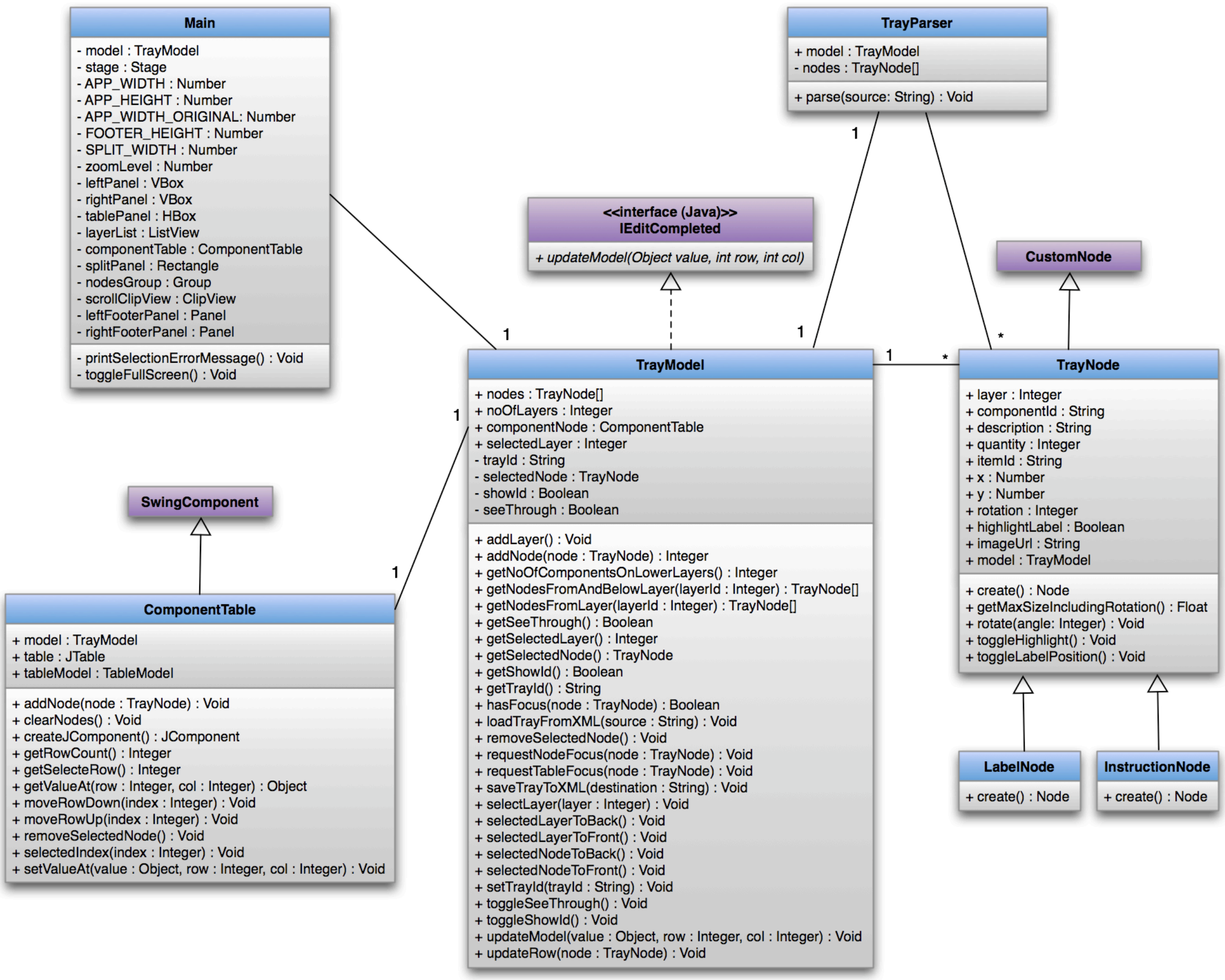
W3C. (2009). *HTML 5*. W3C. Available: <http://www.w3.org/TR/html5/>. Accessed: 21 September 2009.

Ward, James. (2008). *Bursting Bubbles*.
Available: <http://www.jamesward.com/blog/2008/04/10/bursting-bubbles>. Accessed: 7 October 2009.

Zetie, Carl. (2005). The Rise Of Rich Internet Applications. Forrester Research, Inc.

# Appendix I: Prototype Class Diagram

The figure below illustrates the class diagram of the JavaFX Prototype. The included classes, attributes and methods have been chosen according to their relevance for the purpose of giving an informative overview, hence some have been left out. `SwingComponent` and `CustomNode` are abstract JavaFX classes, while `IEditCompleted` is a Java interface.


Main
- model : TrayModel
- stage : Stage
- APP_WIDTH : Number
- APP_HEIGHT : Number
- APP_WIDTH_ORIGINAL: Number
- FOOTER_HEIGHT : Number
- SPLIT_WIDTH : Number
- zoomLevel : Number
- leftPanel : VBox
- rightPanel : VBox
- tablePanel : HBox
- layerList : ListView
- componentTable : ComponentTable
- splitPanel : Rectangle
- nodesGroup : Group
- scrollClipView : ClipView
- leftFooterPanel : Panel
- rightFooterPanel : Panel
- printSelectionErrorMessage() : Void
- toggleFullScreen() : Void
TrayParser
+ model : TrayModel
- nodes : TrayNode[]
+ parse(source: String) : Void
<<interface (Java)>>
IEditCompleted
+ updateModel(Object value, int row, int col)
CustomNode
SwingComponent
TrayModel
+ nodes : TrayNode[]
+ noOfLayers : Integer
+ componentNode : ComponentTable
+ selectedLayer : Integer
- trayId : String
- selectedNode : TrayNode
- showId : Boolean
- seeThrough : Boolean
+ addLayer() : Void
+ addNode(node : TrayNode) : Integer
+ getNoOfComponentsOnLowerLayers() : Integer
+ getNodesFromAndBelowLayer(layerId : Integer) : TrayNode[]
+ getNodesFromLayer(layerId : Integer) : TrayNode[]
+ getSeeThrough() : Boolean
+ getSelectedLayer() : Integer
+ getSelectedNode() : TrayNode
+ getShowId() : Boolean
+ getTrayId() : String
+ hasFocus(node : TrayNode) : Boolean
+ loadTrayFromXML(source : String) : Void
+ removeSelectedNode() : Void
+ requestNodeFocus(node : TrayNode) : Void
+ requestTableFocus(node : TrayNode) : Void
+ saveTrayToXML(destination : String) : Void
+ selectLayer(layer : Integer) : Void
+ selectedLayerToBack() : Void
+ selectedLayerToFront() : Void
+ selectedNodeToBack() : Void
+ selectedNodeToFront() : Void
+ setTrayId(trayId : String) : Void
+ toggleSeeThrough() : Void
+ toggleShowId() : Void
+ updateModel(value : Object, row : Integer, col : Integer) : Void
+ updateRow(node : TrayNode) : Void
TrayNode
+ layer : Integer
+ componentId : String
+ description : String
+ quantity : Integer
+ itemId : String
+ x : Number
+ y : Number
+ rotation : Integer
+ highlightLabel : Boolean
+ imageUrl : String
+ model : TrayModel
+ create() : Node
+ getMaxSizeIncludingRotation() : Float
+ rotate(angle: Integer) : Void
+ toggleHighlight() : Void
+ toggleLabelPosition() : Void
LabelNode
+ create() : Node
InstructionNode
+ create() : Node
ComponentTable
+ model : TrayModel
+ table : JTable
+ tableModel : TableModel
+ addNode(node : TrayNode) : Void
+ clearNodes() : Void
+ createJComponent() : JComponent
+ getRowCount() : Integer
+ getSelecteRow() : Integer
+ getValueAt(row : Integer, col : Integer) : Object
+ moveRowDown(index : Integer) : Void
+ moveRowUp(index : Integer) : Void
+ removeSelectedNode() : Void
+ selectedIndex(index : Integer) : Void
+ setValueAt(value : Object, row : Integer, col : Integer) : Void
1
1
1
1
1
1
1
*
*

# Appendix II: Prototype File Descriptions

**`CheckBoxCellRenderer.java`**

This class makes sure that `JCheckBoxes` are rendered for boolean values in the data grid, instead of showing the textual values true/false. This applies to the Visibile-column.

**`CheckBoxEditor.java`**

This class makes sure that a `JCheckBox` is used for editing boolean values.

**`ComponentTable.fx`**

This class creates and contains the Java Swing component `JTable`. This is achieved by extending the JavaFX class `SwingComponent` and overriding `createJComponent()`. This data grid displays the properties of all the components for each layer in the design.

**`IEditCompleted.java`**

This Java interface makes it possible for a Java method to call a JavaFX method by implementing the design pattern Template Method. The interface contains a hook method called `updateModel()`, which works like a callback function. This functionality is used by the cell editors to be able to notify the model that it should be updated after a cell has been edited. JavaFX classes that wish to be called only have to implement this interface and override `updateModel()`, which in this case is done in `TrayModel`.

**`InstructionNode.fx`**

This is a subclass of `TrayNode` that is either an arrow (that shows how the content should be folded) or a tiding (e.g. a bag that needs to be tied together). This type of node differs from the `TrayNode` in the way that it is not able to be highlighted and it does not have a label attached to it.

**`IntegerEditor.java`**

This class makes sure that a formatted text field is rendered when editing cells in a column that can only contain `Integer` values. Such a validation is performed after a cell has been edited, and if the validation is okay, it notifies the model by calling its hook method `updateModel()` through the `IEditCompleted` interface.

**`LabelEditor.java`**

The concept of this class is the same as for `IntegerEditor`, with the only difference being that no validation is performed, since any `String` is allowed rather than only `Integer` values.

**`LabelNode.fx`**

Similar to an `InstructionNode`, `LabelNode` subclasses `TrayNode` and is a node that only contains text, which also can be edited. It is used for describing the design, i.e. putting a label on the layer (since only one is allowed per layer).

**`Main.fx`**

This is the main file from which the application starts. It contains all the graphical elements that makes up the application, and also code for initializing the model and some settings. This file contains the `Stage`, which in JavaFX is equivalent to Java's `Frame`, i.e. the main window.

**`NumberEditor.java`**

This class is identical to `IntegerEditor`, except for allowing floating-point numbers instead of integers.

**`TableModel.java`**

This class specifies what table columns that should be editable in the data grid by overriding `isCellEditable()`.

**`TrayModel.fx`**

This is the central model that contains all the nodes and information about the design. Several methods for manipulating properties of the nodes exist, including functionality for switching the Z-order of them and get- and set methods for properties concerning the canvas rendering. It extends the Java interface `IEditCompleted` to enable callbacks from Java code, or more precisely from the Java cell editors, when a cell in the data grid has been edited. Moreover, it also contains the method for saving the tray design to XML.

**`TrayNode.fx`**

`TrayNode` is the base class for nodes in the design, and also the actual class for regular components, i.e. non-instruction and non-label nodes. It subclasses `CustomNode` in order to be able to be displayed as a `Node`, and its `create()` method returns the graphical representation on the form of an image. A node can be moved by dragging, and some other visual features may change when performing certain actions on it.

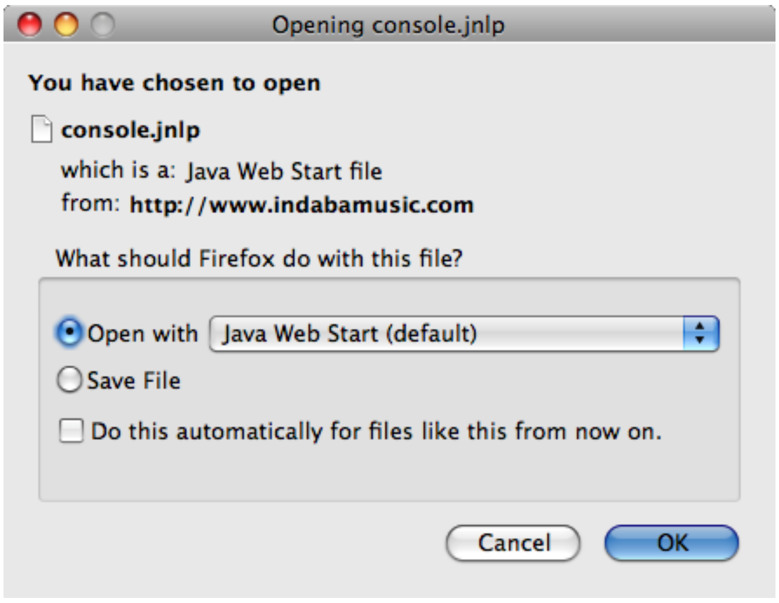
**`TrayParser.fx`**

This class is responsible for fetching and parsing tray designs by first making a `HttpRequest` to a specific URL. After the specified XML file has been downloaded, it then traverses and parses the XML tree structure, and creates the nodes that are finally added to the model.

# Appendix III: JavaFX Startup Procedure

This is an example of JavaFX's startup procedure for the application Indaba Music's Session Console 2.0, deployed with Java Web Start and signed with an unverified certificate authority.

**Step 1.** Choose whether to open the JNLP-file with Java Web Start, or save it.


Opening console.jnlp

You have chosen to open

console.jnlp

which is a: Java Web Start file

from: http://www.indabamusic.com

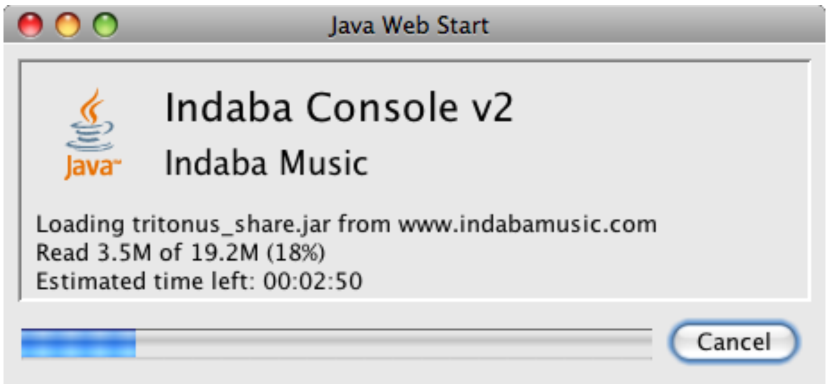
What should Firefox do with this file?

[✓ Open with] Java Web Start (default)

Save File

Do this automatically for files like this from now on.

**Step 2.** Application is being downloaded (does not require the user to click)

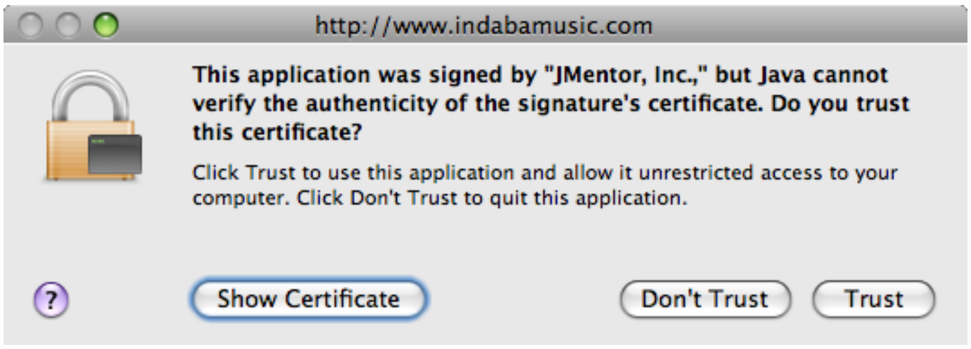

Java Web Start

Indaba Console v2

Indaba Music

Loading tritonus_share.jar from www.indabamusic.com
Read 3.5M of 19.2M (18%)
Estimated time left: 00:02:50

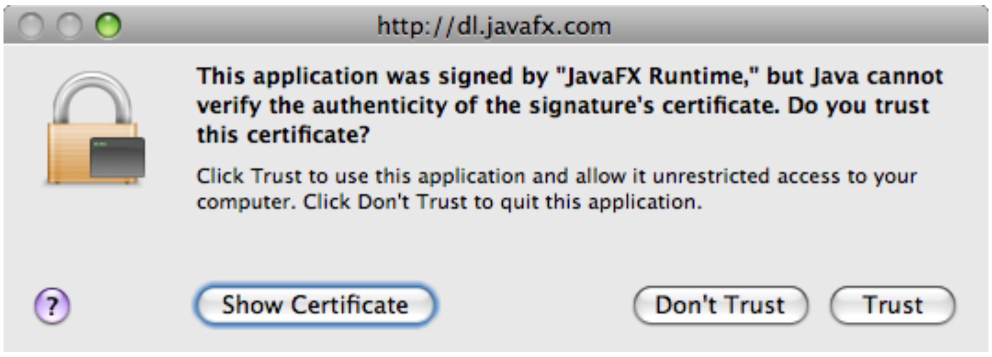
**Step 3.** Choose to trust the application-specific security certificate.


http://www.indabamusic.com

This application was signed by "JMentor, Inc.," but Java cannot verify the authenticity of the signature's certificate. Do you trust this certificate?

Click Trust to use this application and allow it unrestricted access to your computer. Click Don't Trust to quit this application.

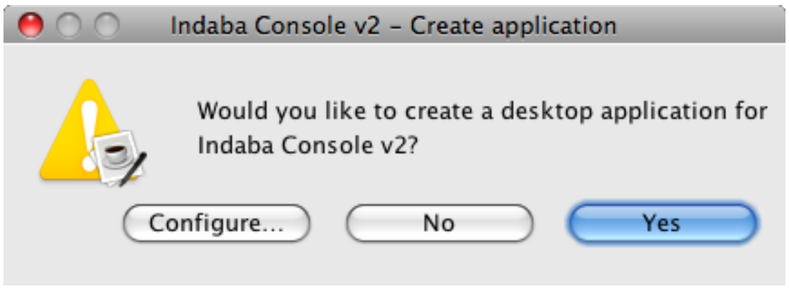
**Step 4.** Choose to trust the JavaFX Runtime security certificate.


http://dl.javafx.com

**This application was signed by "JavaFX Runtime," but Java cannot verify the authenticity of the signature's certificate. Do you trust this certificate?**

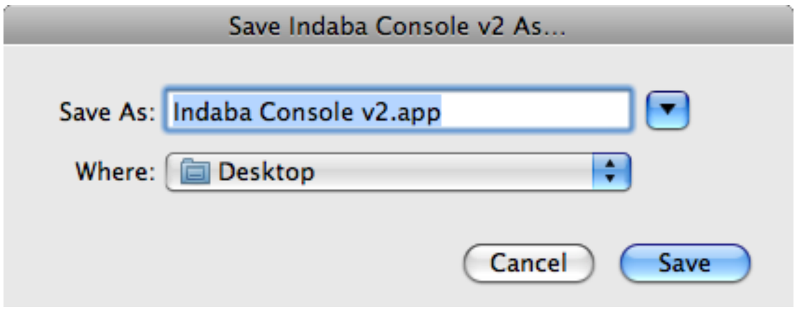
Click Trust to use this application and allow it unrestricted access to your computer. Click Don't Trust to quit this application.

**Step 5.** Choose whether or not to save it to a specific location.


Indaba Console v2 – Create application

Would you like to create a desktop application for Indaba Console v2?

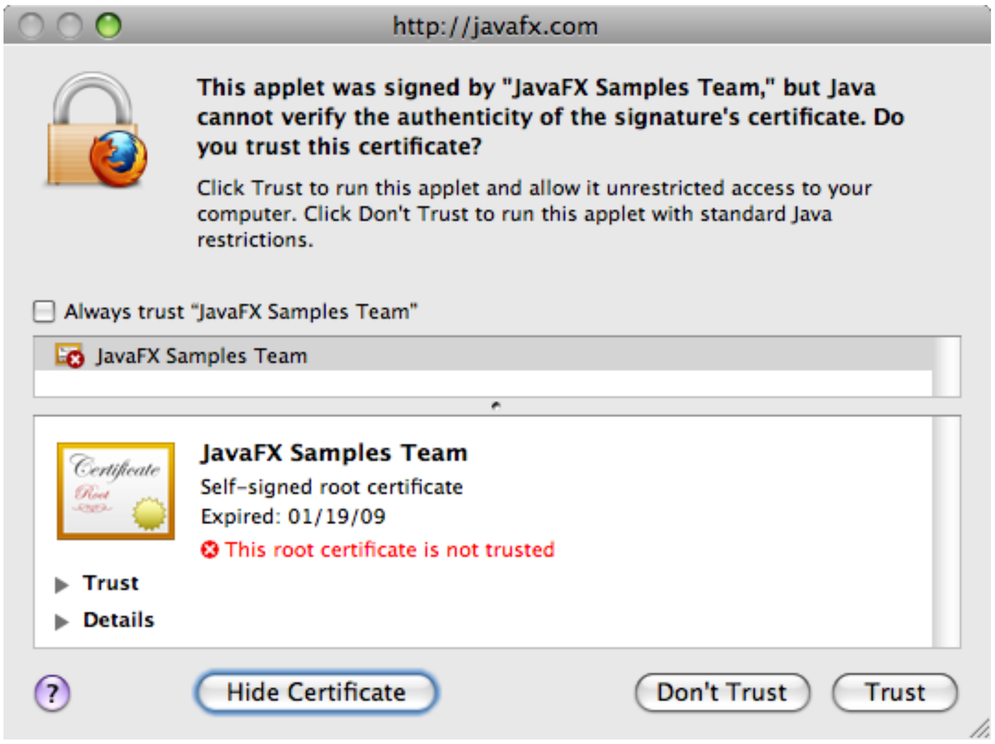
**Step 6.** Choose where to save the application (if yes was chosen in the previous step).


Save Indaba Console v2 As...

Save As: Indaba Console v2.app

Where: Desktop

However, if an application is deployed to run inside the browser, there is no need for the saving- and downloading dialogs to appear, but the certificate-dialog still does, as illustrated below:


http://javafx.com

**This applet was signed by "JavaFX Samples Team," but Java cannot verify the authenticity of the signature's certificate. Do you trust this certificate?**

Click Trust to run this applet and allow it unrestricted access to your computer. Click Don't Trust to run this applet with standard Java restrictions.

☐ Always trust "JavaFX Samples Team"

JavaFX Samples Team

**JavaFX Samples Team**
Self-signed root certificate
Expired: 01/19/09
This root certificate is not trusted

As seen in the screenshot above, the user has the ability to "always trust" certificates from a certain signature. By checking this option, the startup procedure becomes seamless, just like with Flex and Silverlight applications.