Project Presentation: The Actuate JavaScript API

Forest Kingfisher – Content Developer, Actuate (2007 – 2014)

March 2021

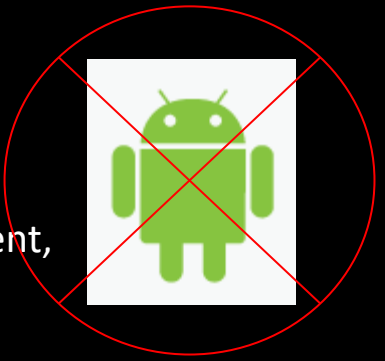
Contents

- Introduction: Why Java EE was not enough
- Project
- Results
- Learning

Welcome back to 2008

In Spring 2008 …

- AWS exits beta, and is not widely adopted
- The current iPhone is the iPhone 1
- Cloud services are a speculated technology
- There is no iPad
- There are no mobile apps
- There is no Android phone
- REST/Stateless Technologies are peripheral (like BitTorrent, gnu)



Introduction

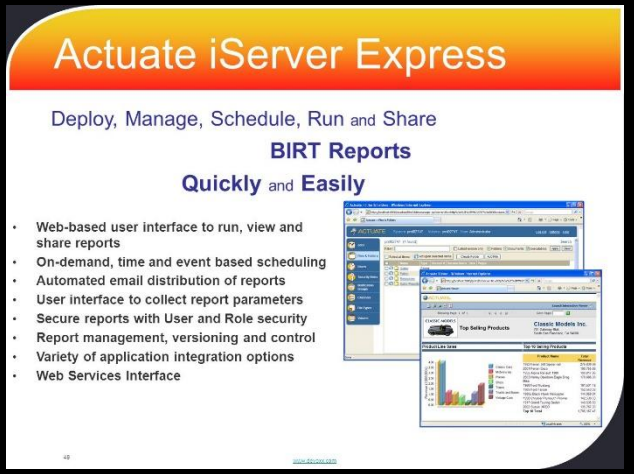
In 2008, I was a content developer at Actuate Corporation

Background: Network administration and web development

Tasks: Build custom implementations on the Actuate Platform and build training courses on how to do this with fully realized practical exercises

ACTUATE
The BIRT Company™

- Business Intelligence and real-time analytics software company
- Flagship product: iServer 9, a scalable BI services server running over any database (ODBC, JDBC, Any DB) on a Java EE implementation.
- SOAP/stateful communications with web client, iPortal, which was fully customizable and BI application development was native to Eclipse using the open source BIRT standard.



The benefits and problems with Java EE

Our customers liked our product but our training courses were considered necessary to implement anything for commercial purposes

Positives:

- Secure
- Internationalizable
- Optimized for wired broadband
- Robust
- Portable (cross-platform)
- Reasonably priced

Negatives:

- Slow over wireless internet
- Required frequent communication with the server
- Required skilled developers
- Required a large time commitment to commercialize

Contents

- Introduction
- Project: JavaScript API
- Results
- Learning

Solution: Asynchronous processes with a simpler and even more portable presentation programming language

Actuate was unwilling to abandon Java EE because it already had such a well performing backend API and Service structure. But a new client type was required.

Network Traffic = performance bottleneck

Learning Curve = productivity bottleneck

1. Perform as much processing at the client, such as interactions and calculations, without contacting the server.
2. Use a simpler programming language set so that programmers could easily drop our components into their company web content.

AJAX: Asynchronous JavaScript and XML

An AJAX solution would allow us to put any Actuate content into a or <iframe> on any page and JavaScript is generally much easier to learn than Java.

What I did … on top of everything else

The 2007 great recession placed increased urgency on the project, so we were told to deliver it in less than a year

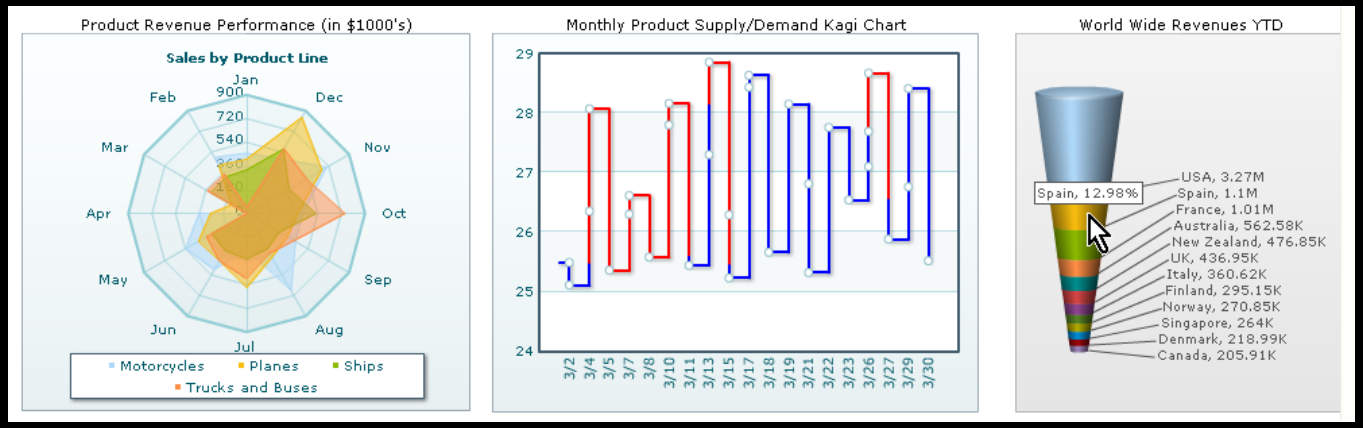
The orders came down to a team of 3 of us, who were to maintain all our current projects: 1 Java Services Engineer (Jun Zhai), 1 Network Integration Engineer (Jayvee Masongsong), and 1 Front-End Engineer (Forest Kingfisher). My tasks were as follows:

1. Create web pages and widgets that could be embedded in web pages that use the API to show off its capabilities – our customers must like them.
2. Document the capabilities in print and online forms, similar to JavaDoc, with tested examples for every function.
3. Build, test and publish a training course complete with programming exercises on how to build web components with the API.
4. QA has no resources to spare, so test the daily builds using my ready-to-ship components and work with the developers to resolve bugs.

Building widgets and web pages

For the first couple of weeks, the builds did not work at all.

Each night, the entire Actuate suite of products were built using the latest code. I had a fresh VM to load up the product in progress each morning, and I would start by installing iServer and iPortal. Then, I would run my pages and components on each browser to test out new functionality and verify previous functionality. In the end, a set of 11 pages, a dashboard, and 5 widgets were approved to be packaged with the Actuate 10 release.

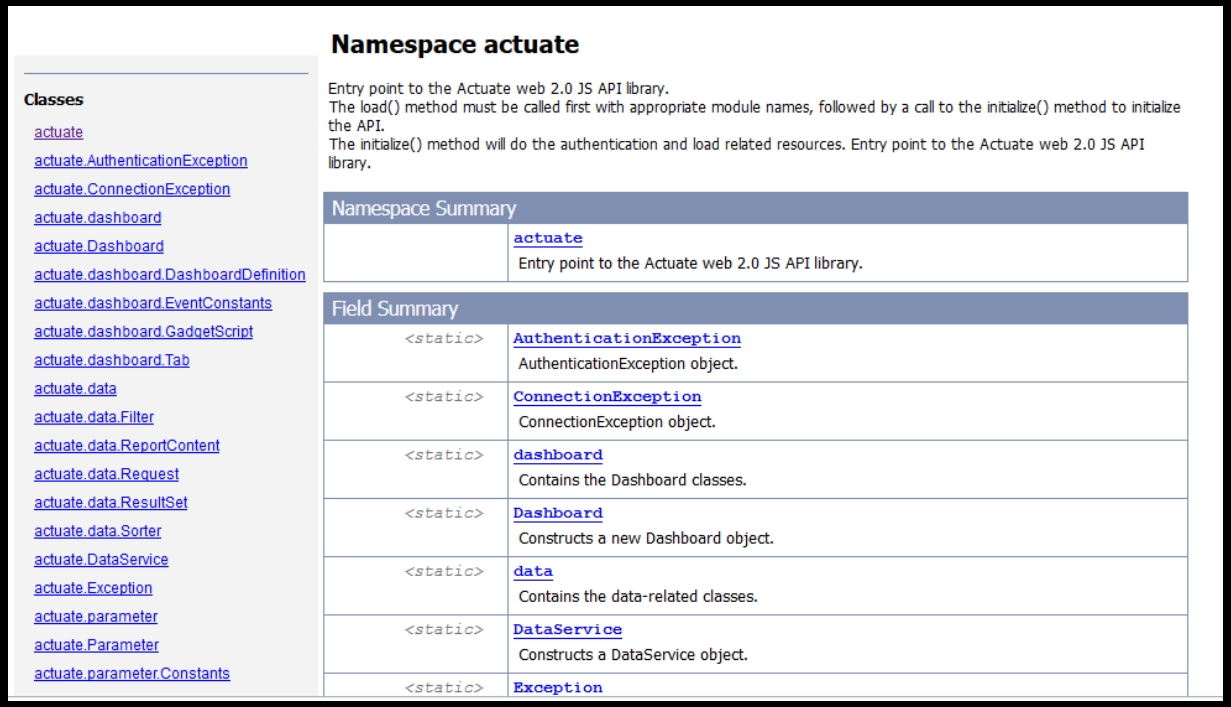


Many of my originally designed components never made it to the final build because our feature set evolved and syntax changed

Document capabilities

I have worked as a technical writer in many previous positions and was my primary responsibility at Google

While the first builds did not work, I had access to the source code, so I spent some time tweaking a JSDoc generator and commenting the code so that some JavaDoc style output would build with the product each morning.



Classes

- actuate
- actuate.AuthenticationException
- actuate.ConnectionException
- actuate.dashboard
- actuate.Dashboard
- actuate.dashboard.DashboardDefinition
- actuate.dashboard.EventConstants
- actuate.dashboard.GadgetScript
- actuate.dashboard.Tab
- actuate.data
- actuate.data.Filter
- actuate.data.ReportContent
- actuate.data.Request
- actuate.data.ResultSet
- actuate.data.Sorter
- actuate.DataService
- actuate.Exception
- actuate.parameter
- actuate.Parameter
- actuate.parameter.Constants

Namespace actuate

Entry point to the Actuate web 2.0 JS API library.
The load() method must be called first with appropriate module names, followed by a call to the initialize() method to initialize the API.
The initialize() method will do the authentication and load related resources. Entry point to the Actuate web 2.0 JS API library.

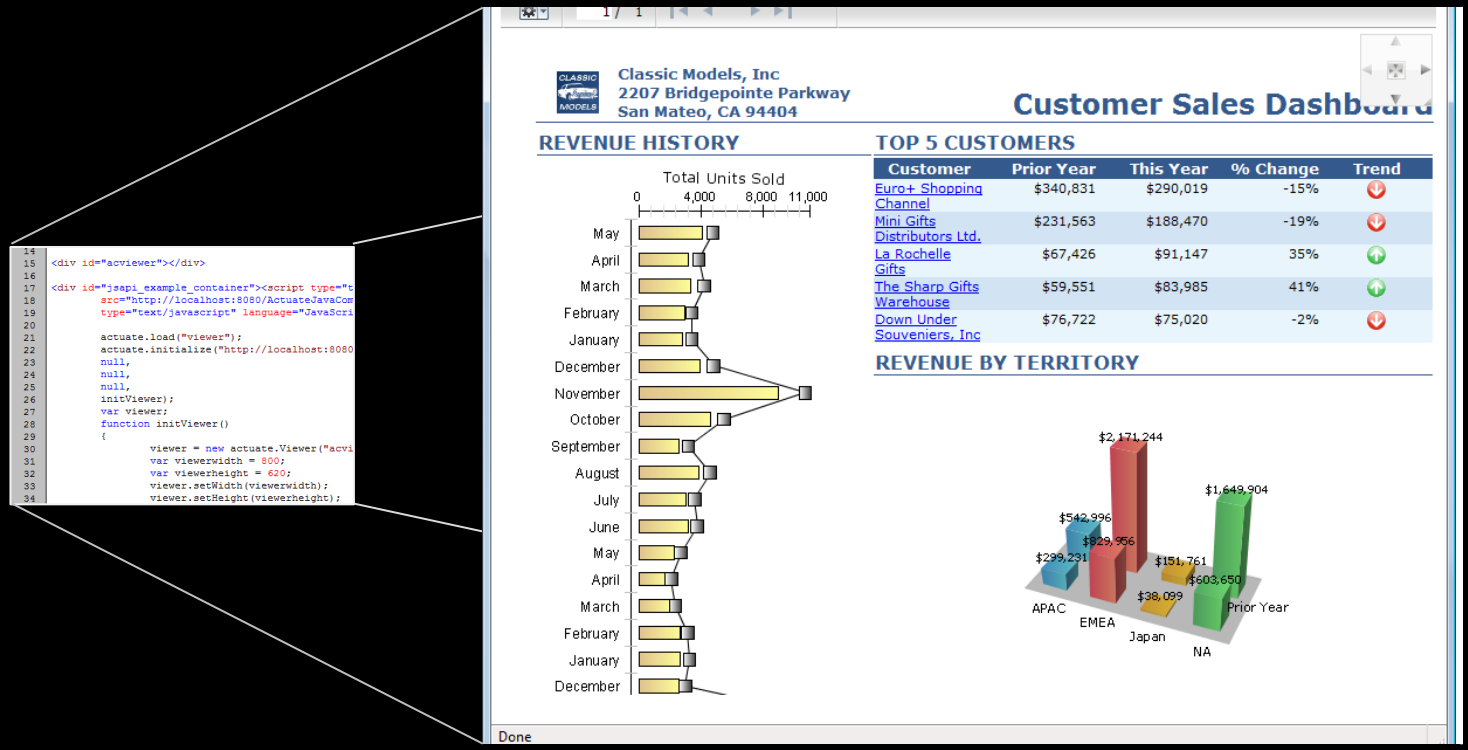
Namespace Summary	
	actuate Entry point to the Actuate web 2.0 JS API library.

Field Summary	
<static>	**AuthenticationException** AuthenticationException object.
<static>	**ConnectionException** ConnectionException object.
<static>	**dashboard** Contains the Dashboard classes.
<static>	**Dashboard** Constructs a new Dashboard object.
<static>	**data** Contains the data-related classes.
<static>	**DataService** Constructs a DataService object.
<static>	**Exception**

Developer Training Course, for sale

Actuate training courses provided additional support to our customers

Original materials for a training course walked a student developer through the process of writing code that output interactive web content, practically versed in every major capability of the JSAPI in one day. The training release followed the product release by three months.



Daily Builds meant Daily Testing

Actuate was a medium-sized tech company, so did no always provision maximum resources to every project.

I became very good at filing clear, reproducible bug reports for functional and non-functional defects.

1. We met daily, employing a novel project management style called Agile.
2. The build added a new features and broke that feature or an existing feature at least every week for the first 6 months. This cycle would repeat with further releases as we added new features like cross-tabs, pivot tables, interactive charts, jQuery integration, and more.
3. QA never got around to testing this feature, so I became the QA representative even though I was in content development.
4. I also became very good a troubleshooting installation and configuration of the product, which made me a resource for the iServer content developer.

Contents

- Introduction
- Project
- Results: Overwhelming success, for one tech moment
- Learning

Results

The recession also increased demand for business intelligence accuracy in oversight activities, which elevated our products in the financial sector

The Actuate 10, 10sp1, 10sp2, and 11 releases were some of our most successful. Our wireless performance problems vanished.

The JSAPI training course quickly became our highest demand training course, and the JSAPI was widely adopted by our partners and customers.

2009-2010 saw the largest post-recession growth in customers.

Actuate Customers added in 2009-10

Financial Services at the Core

ACTUATE
The people behind BIRT

But mobile and cloud would overtake the JSAPI

I was involved in many of the following projects but more resources were involved in each as part of the JSAPI windfall

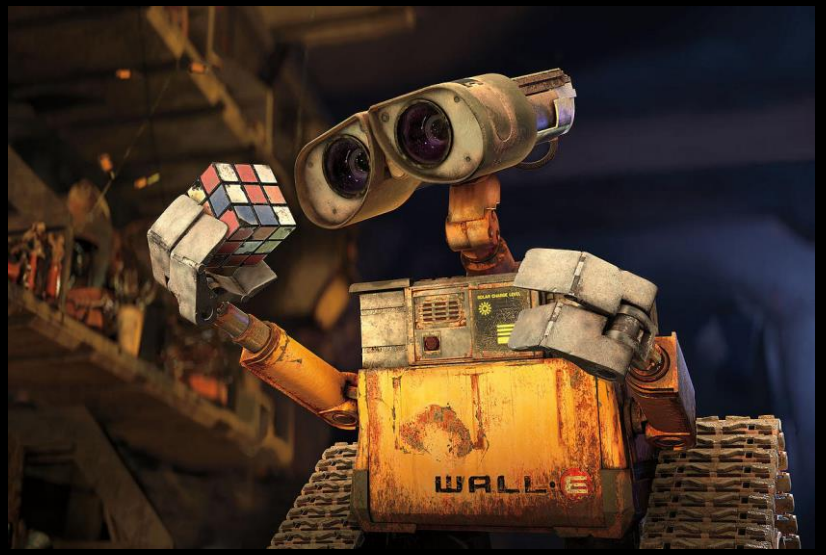
Our focus on wireless internet performance in 2008 provided benefits for future development:

- Actuate fielded a cloud service, BIRT on Demand, in 2011 which was reasonably successful
- Actuate entered the mobile app market with a BIRT Mobile
  - for *iPhone* in 2010
  - For *Android* in 2011

However, the seeds of Actuate's decline were sewn in 2010

- In 2009, Actuate reached a legal settlement with Oracle, which was using Actuate tech without a license through its acquisition of Seibel. As a result, Oracle created its own BI solution "Business Intelligence Enterprise Edition Plus" to compete with Actuate products in 2010. While not popular, it was often offered as a benefit to purchasing Oracle's DB license, so many customers followed.

Contents



- Introduction
- Project
- Results
- Learning: When tech serves a purpose …

The purpose of technology in our lives, our businesses

I am a huge fan of science and technology

1. The right technology at the right time can do amazing things, and yield great rewards.
2. While technology innovates in many ways continuously, circumstances provide the means to leverage them and profit from them. Know your markets, know your world.
3. Old tech or simpler tech can often be just as valid as new or complex tech. I was not in favor of JavaScript when the project began, but learned how it could serve our purposes and our customers.
4. Technologies are seldom meant to last. While data has a long shelf life, the tech solutions that are popular today are sure to be replaced with things more appropriate to new circumstances.
5. A company with greater resources turns slower on trends but has more resources to realize the best solution.

The next step:

Thank you + Q & A