

Web Browser as an Application Platform

Antero Taivalsaari

Sun Microsystems Laboratories
November 27, 2007

<http://research.sun.com/projects/lively>
lively@sun.com



Sun Labs



Background

- The widespread adoption of the World Wide Web has dramatically altered the landscape of software development.
- Documents, photos, music, videos, news and various other services have already started migrating to the web.
- Software will be next!

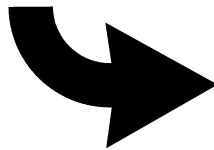
Paradigm Shift!

- The software industry is currently in the middle of a paradigm shift.
- Applications are moving to the Web.
 - > Applications are no longer written for a specific type of computer, operating system or device.
 - > Rather, they will be written for the Web, to be used via a web browser from anywhere, anytime.
- Web browser is increasingly taking the role that the operating system used to have.

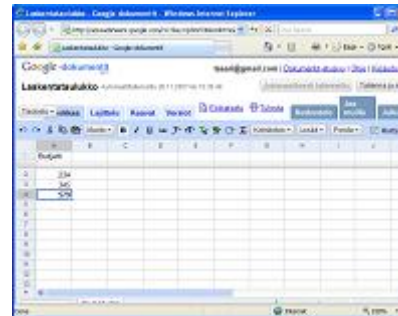
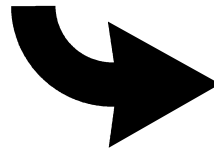
Evolution of the Web



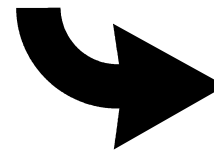
1) Simple pages with text and static images only
(e.g., www.google.com)



2) Animated pages with plug-ins
(e.g., www.cadillac.com)



3) Rich Internet Applications
(e.g., docs.google.com)



What's Next?

Web 2.0 – What Is It, Really?

- Support for “social” behavior.
 - > Support for worldwide sharing and collaboration.
 - > The same systems shared by users worldwide.



- Real applications on the Internet.
 - > Not just “pages” or “documents”.
- Compelling user interaction capabilities.
 - > Support for direct manipulation.
 - > No more full page refresh (à la IBM 3270 of the 1970s) when something changes.
 - > No more back button, reload button, stop button, ...

Unfortunately...

- The web browser was not designed for running real, desktop-style applications.
 - > It was designed in the early 1990s for viewing documents, forms and other page-structured artifacts – *not* applications.
- Programming capabilities on the web were an afterthought, not something inherent in the design of the browser.
- But this is the direction people are heading anyway.

Sun Labs Lively Kernel Project

The New Era (like it or not...)

- The Web is the new Application Platform.
- The Web Browser is the new Operating System.
- JavaScript is the *de facto* Programming Language on the Web.

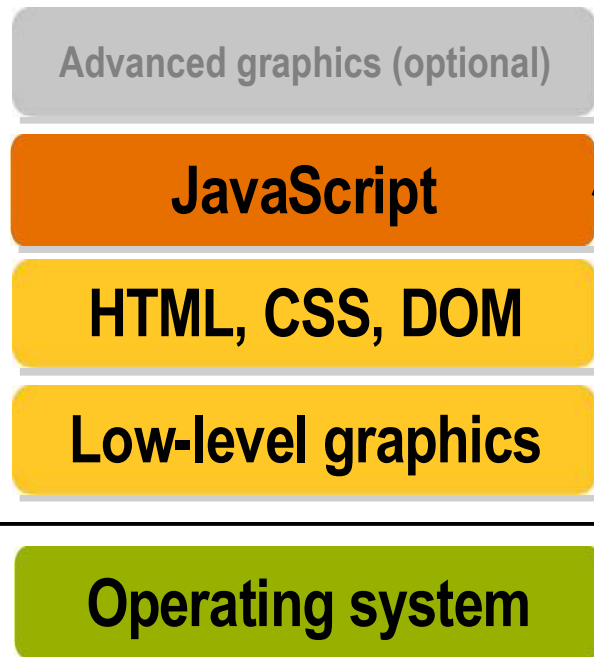
Sun Labs Lively Kernel

- Lively Kernel is a web application development environment written entirely in JavaScript.
- Runs in a regular web browser with no installation or plug-ins whatsoever.
- Supports real applications on the Web, with rich user interface features and direct manipulation capabilities.
- Enables application development and deployment without installation or upgrade hassles.
- Built around existing web technologies.

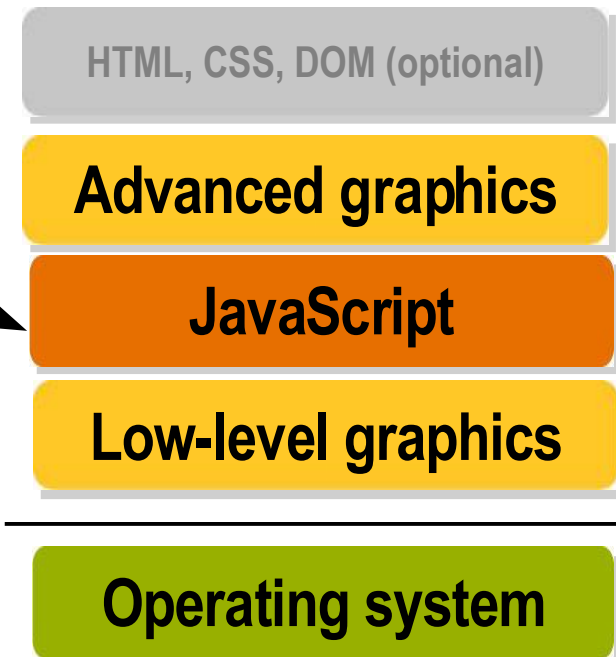
Turning Things “Upside Down”

- Start with JavaScript and a desktop-style graphics architecture.
- De-emphasize HTML and the current document-oriented approach to web development.

Conventional Web Client Stack



Lively Kernel Stack



Lively Kernel In a Nutshell

Key components:

- JavaScript programming language
 - Asynchronous networking support (async HTTP)
 - Desktop-style graphics model with zooming support
 - Morphic application framework and widgets
-
- Use existing technologies wherever possible!

Demo!

Web as a Platform – Experiences

- During our project, we've discovered problems in various areas related to the use of the web browser as an application platform.
 - 1) Usability and user interface issues
 - 2) Networking and security issues
 - 3) Browser interoperability and compatibility issues
 - 4) Development style and testing issues
 - 5) Deployment issues
 - 6) Performance issues

Usability and User Interface Issues

Highlights:

- Limited direct manipulation capabilities.
- Poorly suited I/O model between JavaScript and the browser (via DOM)
- Poorly suited networking model between the client and the server.
- “Legacy buttons” in the browser.

Networking and Security Issues

Highlights:

- “Same origin” networking policy restrictions.
- Only a limited number of simultaneous network requests allowed.
- No local storage support / no access to the local file system.
- In general: Limited access to host platform capabilities.

Browser Compatibility Issues

Highlights:

- Incompatible DOM implementations.
- Incompatible JavaScript implementations.
- Incompatible graphics library implementations.
- Disregard for official standards.
- Lack of official standards (e.g., lack of advanced JavaScript libraries, no agreement on the future of the JavaScript language itself).
- Plug-in availability.

Development Style and Deployment Issues

Highlights:

- JavaScript is an extremely permissive, dynamic language -> different development style required.
- Incompatible programs allowed (no static checking) -> code coverage testing is very important.
- JavaScript APIs are still limited.
====
- It is not clear what constitutes a “release.”
- Applications are online 24x7 -- when is it safe to update them?

Performance Issues

- JavaScript virtual machines are still very slow.
- Browser graphics libraries (e.g., SVG engines) are also slow.
- When people start writing more serious web applications, performance issues will become more evident.
- On the positive side: There are a lot of opportunities to improve performance.
- Current JavaScript VMs are surprisingly reliable and almost impossible to crash.

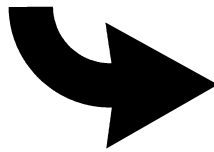
Conclusions

- Like it or not, the Web is increasingly the platform of choice for advanced software applications.
- Web-based applications have major benefits: no installation or upgrades needed, instant worldwide deployment, the ability to “own” the users' data.
- Web-based applications will dramatically change the way people develop, deploy and use software -> paradigm shift!
- We will provide a detailed summary of our experiences in a forthcoming research paper.

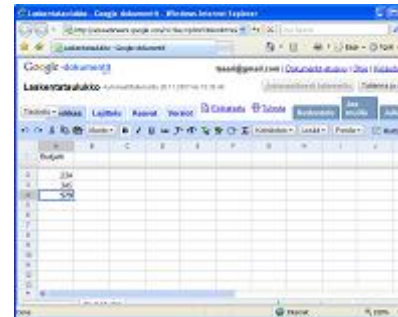
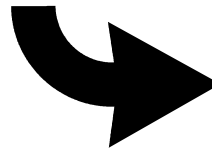
Evolution of the Web – What's Next?



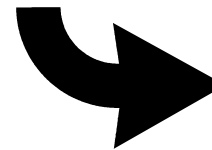
1) Simple pages with text and static images only
(e.g., www.google.com)



2) Animated pages with plug-ins
(e.g., www.cadillac.com)



3) Rich Internet Applications
(e.g., docs.google.com)



**Shared 2D/3D
Worlds
on the Web**

Logical next step: combine interaction & collaboration

4) Shared 2D/3D Worlds running on the Web

Discussion

Antero Taivalaari

<http://research.sun.com/projects/lively>

lively@sun.com



Sun Labs



Why JavaScript?

- JavaScript is ubiquitous.
 - > Every web browser has a JavaScript engine in it.
- JavaScript has developer appeal.
 - > Familiar to people with C, C++ or Java background.
 - > Not a “geek” language.
- JavaScript is dynamic.
 - > No more edit-compile-link-run-crash-debug cycles.
 - > Applications can be created, deployed and updated on the fly in a seamless environment.
- JavaScript has potential.
 - > Momentum still growing; performance will improve.