

# WebGL Seminar @ TUT

<http://lively.cs.tut.fi/seminars/WebGL2011>

Prof. Tommi Mikkonen (Tampere University of Technology, Finland)

Dr. Antero Taivalsaari (Nokia Research Center & TUT)



# Background

- History of computing and software development is full of disruptive periods and paradigm shifts.
- The computing industry goes through major changes every 10-15 years.
- Examples of disruptive eras:
  - Minicomputers in 1970s
  - Personal computers in the 1980s
  - Web 1.0 in the 1990s
  - Mobile software in the 2000s
  - Cloud computing in the 2010s

# Disruptive Trend Today: Web-Based Software

- The widespread adoption of the World Wide Web is reshaping our world in various ways.
- Documents, photos, music, videos, news and various other artifacts and services have already started migrating to the Web.
- Many industries (e.g., publishing and entertainment) are currently undergoing dramatic transformations.
- The software industry is currently experiencing a similar transformation, or a paradigm shift.

# Web Applications – Implications

- Web-based software will dramatically change the way people develop, deploy and use software.
- No more installations!
  - > Applications will simply run off the Web.
- No more upgrades!
  - > Always run the latest application version.
- Instant worldwide deployment!
  - > Whatever we release here in Tampere is instantly visible in Tammisaari, Tampa Bay, Tandragee, Tasmania or Tanzania.
  - > No middlemen or distributors needed.
- No CPU dependencies, OS dependencies, ...
  - > The Web is the Platform.

# Unfortunately...

- The web browser was not designed for running real 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.
- Until recently, the capabilities of the web browser to execute and display truly interactive applications and content has been limited.
  - > Various additional components or plugins (Flash, Shockwave, Quicktime, Silverlight, ...) have been necessary to add more interactive types of content to the browser.

# Evolution of the Web

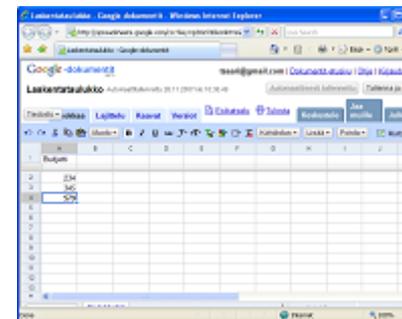


1) Simple pages with text and static images only  
(e.g., <http://www.google.com>)

2) Animated pages with plug-ins  
(e.g., <http://www.cadillac.com>)



3) Rich Internet Applications  
(e.g., [docs.google.com](http://docs.google.com))



What's Next?

# Web Development vs. Software Engineering

## Impedance Mismatch

<b>Web Development</b>	<b>Conventional SW Development</b>
<ul style="list-style-type: none"><li>- Documents</li><li>- Page / form oriented interaction</li><li>- Managed graphics, static layout</li><li>- Instant worldwide deployment</li><li>- Source code and text favored</li><li>- Development based mostly on conventions and “folklore”</li><li>- Informal development practices</li><li>- Target environment not designed for applications</li><li>- Tool-driven development approach</li></ul>	<ul style="list-style-type: none"><li>- Applications</li><li>- Direct manipulation</li><li>- Directly drawn, dynamic graphics</li><li>- Conventional installation</li><li>- Binary representations favored</li><li>- Development based on established engineering principles</li><li>- More formal development</li><li>- Target environment specifically intended for applications</li><li>- A wide variety of development approaches available</li></ul>

# The Evolving Web Browser

- There are numerous ongoing web standards activities.
  - > In fact, it is often difficult to identify the really important activities from all the noise – “alphabet soup” problem.
- There are two very important standards that will significantly enhance the capabilities of the Web:
  - > *HTML5* (<http://www.w3.org/TR/html5>)
  - > *WebGL* (<http://www.khronos.org/webgl>)
- HTML5 will enable desktop-style web applications that can be used in offline mode in addition to normal web-based operation.
- WebGL will add the ability to display 3D graphics directly in the web browser without any plug-in components.

# HTML5: Main New Features

- `<canvas>` element for immediate mode 2D drawing
- Timed media playback (`<video>` and `<audio>` tags)
- Offline storage database (offline web applications)
- Interactive document editing
- Drag-and-drop support
- Cross-document messaging
- Browser history management
- MIME type and protocol handler registration
- Microdata (HTML annotations)
- For details see, e.g.: <http://diveintohtml5.org/>

# WebGL

# Introduction to WebGL

- *WebGL* is a cross-platform web standard for hardware accelerated 3D graphics API.
  - > Developed by Mozilla, Khronos Group, and a consortium of other companies including Apple, Google and Opera.
- The main feature that WebGL brings to the Web is the ability to display 3D graphics natively in the web browser.
- WebGL support is already available in recent versions of the popular web browsers.
  - > Firefox (4B1 and later), Chrome (7 and later), Safari (nightly builds)
  - > Not yet available in Internet Explorer (not even in IE9)
  - > See [http://www.khronos.org/webgl/wiki/Getting\\_a\\_WebGL\\_Implementation](http://www.khronos.org/webgl/wiki/Getting_a_WebGL_Implementation)

# WebGL from Programmer's Viewpoint

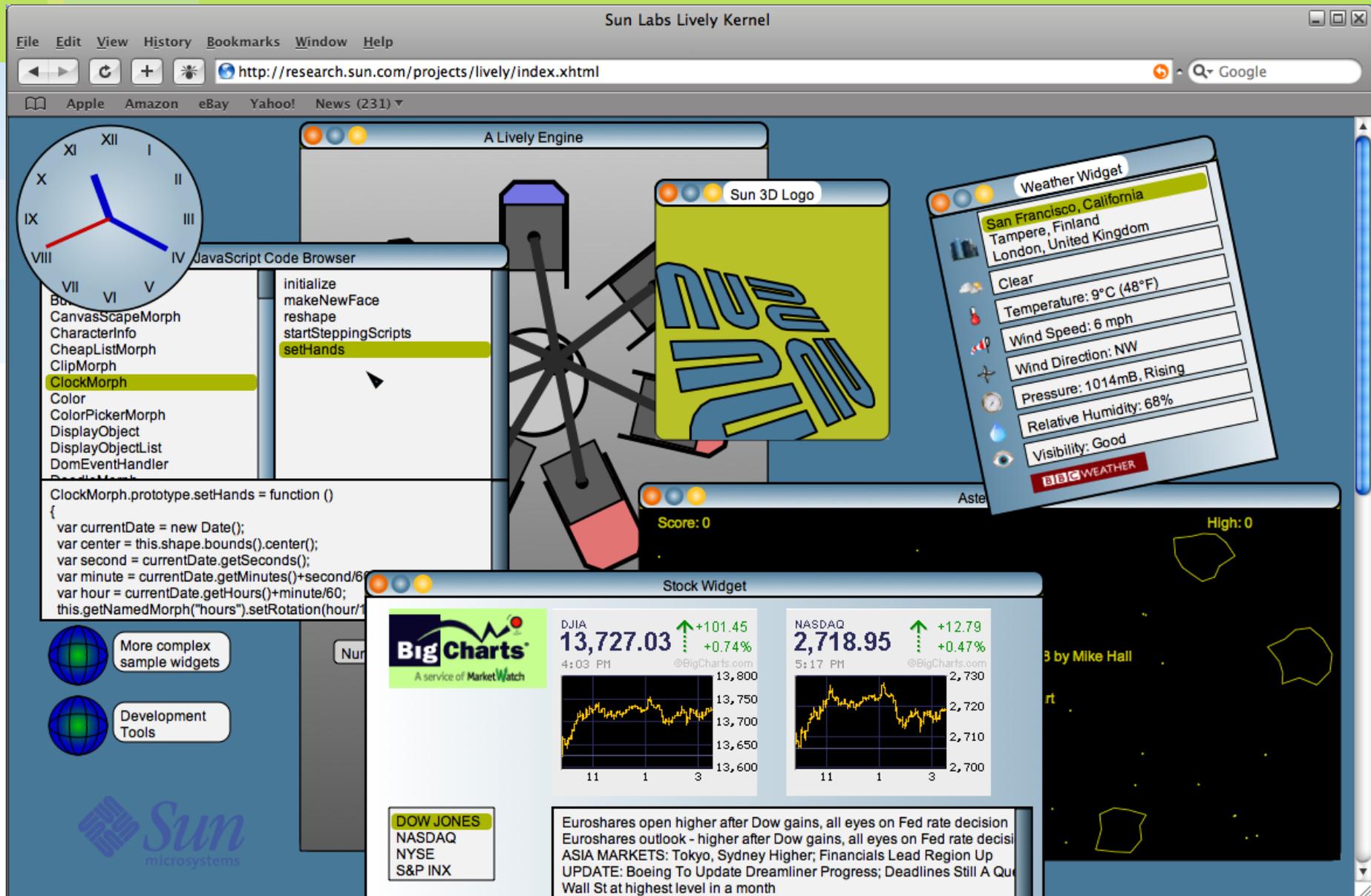
- WebGL is based on OpenGL ES 2.0, and it uses the OpenGL shading language GLSL.
  - > <http://www.khronos.org/opengles>
  - > <http://www.opengl.org/documentation/glsl>
- WebGL runs in the HTML5's `<canvas>` element.
- WebGL data is generally accessible through the web browser's Document Object Model (DOM) interface.
- A comprehensive JavaScript API is provided to open up OpenGL programming capabilities to JavaScript programmers.
  - > If you are familiar with OpenGL (and JavaScript), it should be easy to get started.

# Why is WebGL Relevant?

- Chances are, today you are already spending 60-90% of your time on a computer using the web browser.
  - > ... but not if you like to play high-end games!
- So far, it has been very difficult to convince game developers to take web-based software seriously.
  - > too slow, poor graphics support, poor developer experience, ...
- WebGL will effectively eliminate the “last safe bastion” of conventional binary applications.
- WebGL will make it possible to run not only interactive 3D applications but 2D applications as well.
  - > So far, it has been difficult to write procedural (as opposed to declarative) 2D software that runs in a standard web browser.

# Lively Kernel (<http://lively-kernel.org/>)

## Example of a Highly Interactive 2D Desktop Environment

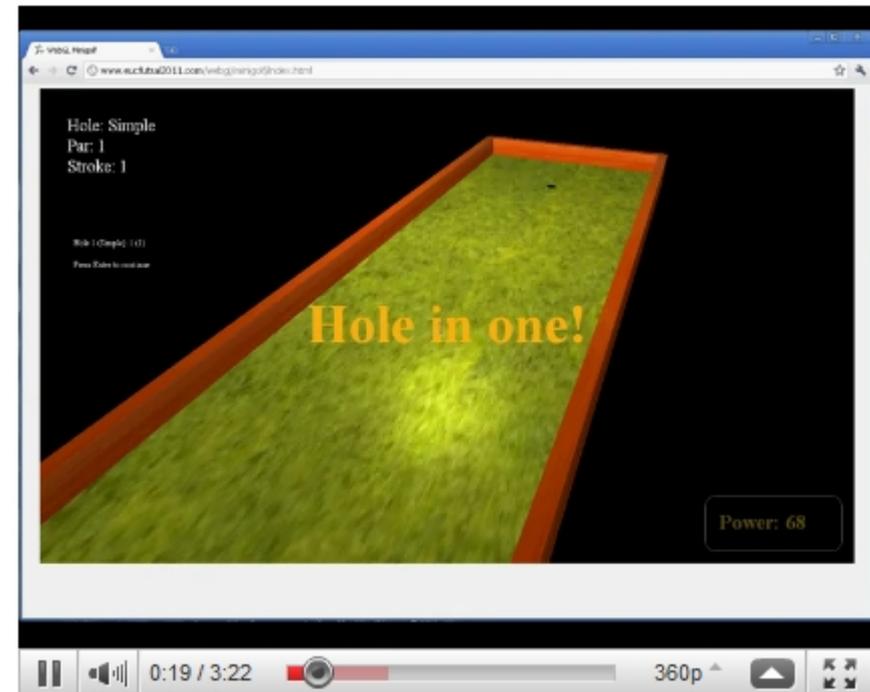
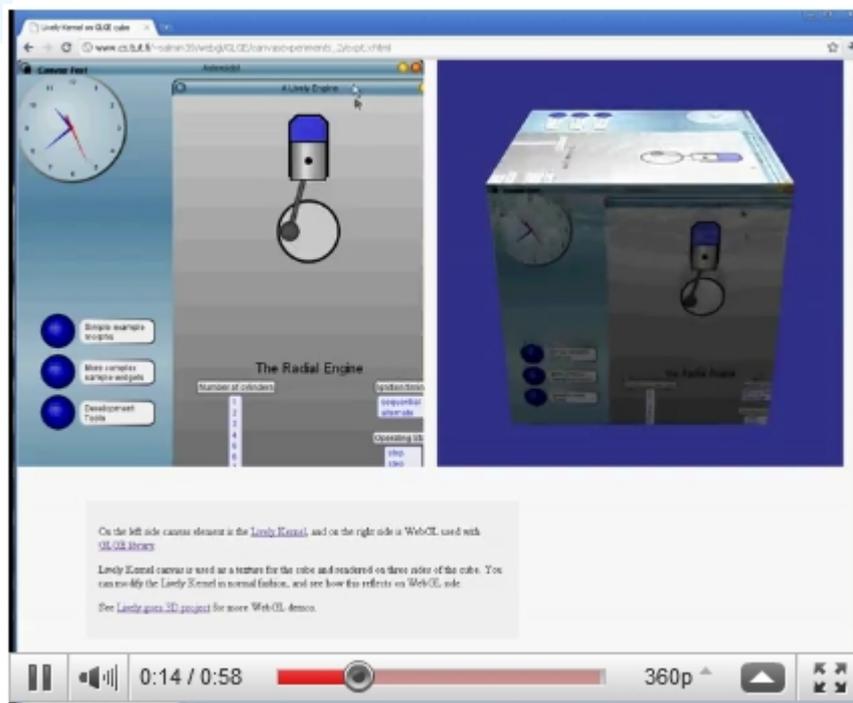


The screenshot displays the Sun Labs Lively Kernel desktop environment. The browser window shows the URL `http://research.sun.com/projects/lively/index.xhtml`. The desktop features several interactive widgets:

- A Lively Engine**: A central window displaying a 3D mechanical assembly.
- Sun 3D Logo**: A yellow window with a blue 3D logo.
- Weather Widget**: A tilted window showing weather data for San Francisco, California, including temperature (9°C / 48°F), wind speed (6 mph), and direction (NW).
- Stock Widget**: A window displaying stock market data from BigCharts, including DJIA (13,727.03) and NASDAQ (2,718.95) with line charts.
- JavaScript Code Browser**: A window showing code for `ClockMorph.prototype.setHands`.
- Clock**: A circular analog clock widget.
- Navigation**: Buttons for "More complex sample widgets" and "Development Tools".
- Footer**: Sun Microsystems logo and a news section with headlines like "Euroshares open higher after Dow gains".

# Lively 3D – WebGL Research @ TUT

- Lively 3D is a research project at TUT that investigates the use of WebGL in building a highly interactive web-based application development environment.
  - > <http://lively.cs.tut.fi/>
  - > <http://livelygoes3d.blogspot.com/>



# WebGL Libraries

- Various WebGL libraries are available to raise the level of abstraction and improve programmer productivity:
  - > C3DL (<http://www.c3dl.org/>)
  - > Copperlicht (<http://www.ambiera.com/copperlicht/>)
  - > CubicVR (<http://www.cubicvr.org/>)
  - > EnergizeGL (<http://energize.cc/>)
  - > GLGE (<http://www.glge.org/>)
  - > O3D (<http://code.google.com/p/o3d/>)
  - > SceneJS (<http://scenejs.org/>)
  - > SpiderGL (<http://spidergl.org/>)
  - > WebGLU (<http://github.com/OneGeek/WebGLU>)
  - > X3DOM (<http://www.x3dom.org/>)

# WebGL Examples

- Miscellaneous links from the Web:
  - > <http://planet-webgl.org/>
  - > <http://learningwebgl.com/blog/>
  - > <http://learnwebgl.appspot.com/>
  - > <http://learnwebgl.blogspot.com/>
  - > <http://www.ibiblio.org/e-notes/webgl/webgl.htm>
  - > <http://www.dankantor.com/html5/html5-webgl.php>
  - > <http://twitter.com/mrdoob/status/10408503797620736>
  - > <http://code.google.com/p/quake2-gwt-port/>
  - > <http://code.google.com/p/webhierarkia/wiki/WebGL>
  - > [https://developer.mozilla.org/en/WebGL/Animating\\_objects\\_with\\_WebGL](https://developer.mozilla.org/en/WebGL/Animating_objects_with_WebGL)

# About the Seminar: Practical Arrangements

# Why This Seminar?

- HTML5 and WebGL will dramatically change people's perception about the web browser as an application environment.
- In this seminar we will:
  - > Study WebGL and the various libraries and tools in this area.
  - > Build applications using those technologies.
  - > Drill deeper into those technologies that seem most likely to succeed.
  - > More broadly: Raise the awareness of the importance of WebGL and web browser as an application platform.

# Practical Arrangements

- The seminar will be held on **Fridays, 12:15 - 13:45** in **Tietotalo TC103**.
- Next seminar session on **Friday, December 17:**  
*WebGL Technical Overview*  
by Matti Anttonen and Arto Salminen.
- Weekly student presentations will begin on **January 7, 2011**.
  - > Detailed schedule to be announced.

# How to Get Credits?

- Maximum number of credits: 3-5 op
- Attendance: 1 op
- Seminar presentation (30-45 min) on selected WebGL library/technology: 2 op
- Successfully written, new demo application and/or written report on selected technology: additional 2 op

# Choosing Presentation Topics

- Please choose your presentation topic and the preferred presentation date as soon as possible.
- Send e-mail to: *tommi.mikkonen[at]tut.fi*
- Topics allocated on a “first-come-first-serve” basis.
- First available presentation slot(s): January 7, 2011.
- Seminar web site page will be updated regularly to list the chosen presentation topics:
  - > <http://lively.cs.tut.fi/seminars/WebGL2011/>
- Presentations can be held in Finnish or English.
  - > English preferred if there are non-Finnish-speaking participants.

# Proposed Outline for Presentations

- Introduction
  - > high-level overview, purpose of the technology, background/history
- Technical overview of the technology
- Small examples
- Walkthrough of a more comprehensive example illustrating the use of the technology
- Evaluation
  - > benefits, drawbacks, general usefulness, possible measurements
- Summary
- **Presentation length: 30-45 min (incl. 10-15 min for questions)**

# Available Presentation Topics

- > C3DL (<http://www.c3dl.org/>)
- > Copperlicht (<http://www.ambiera.com/copperlicht/>)
- > CubicVR (<http://www.cubicvr.org/>)
- > EnergizeGL (<http://energize.cc/>)
- > GLGE (<http://www.glge.org/>)
- > O3D (<http://code.google.com/p/o3d/>)
- > Processing.js (<http://processingjs.org/>)
- > SceneJS (<http://scenejs.org/>)
- > SpiderGL (<http://spidergl.org/>)
- > WebGLU (<http://github.com/OneGeek/WebGLU>)
- > X3DOM (<http://www.x3dom.org/>)
- > WebGL in QtWebKit (<http://qt.nokia.com/products/library/qtwebkit>)

## Action Items for Next Week (Dec 17)

1. Choose your preferred presentation topic.
2. Come up with a great idea for a possible demo application that you would like to write.

**Thank You!  
Questions?**

<http://lively.cs.tut.fi/seminars/WebGL2011>

