# Cloud and mobile Web-based graphics and visualization SIBGRAPI 2012 Full-Day Tutorial 22 August 2012, Ouro Preto – MG, Brazil

Haim Levkowitz
Department of Computer Science
University of Massachusetts Lowell
Lowell, Massachusetts 01854, USA
and
ICMC
University of São Paulo, São Carlso – SP, Brazil
haim@cs.uml.edu, http://www.cs.uml.edu/∼haim/

## Abstract

Cloud computing is becoming the most prevailing computing platform. The combination of mobile devices and cloud-based computing is changing how users consume and use computing resources. With the introduction and penetration of HTML5, and, in particular, its visual Canvas element, high-quality Web-based graphics has become a reality. WebGL offers capabilities comparable to OpenGL utilizing Web-based computing resources. It is now feasible to have high-performance graphics and visualization "in your palm," utilizing a mobile device as the front end interface and the display, performing the graphics "heavy lifting" on a cloud platform as needed. We argue that this will become the most common platform for computer graphics and visualization in the not-too-distant future.

The goals of this course are to make students familiar with the underlying technologies that make this possible, including cloud computing, mobile computing, their combination, HTML5 and the Canvas element, WebGL, other libraries, and general Web-based graphics and visualization.

Who should attend: researcher, practitioners, and students focused on the fields of cloud computing, mobile computing, graphics, visualization, Web-based environments and their applications. Students will gain a deep understanding of these techniques and technologies, and will become capable of applying their knowledge to develop interactive mobile- and cloud-based graphics and visualization applications. Previous knowledge of and experience with interactive computer graphics and visualization will is recommended.

**keywords:** cloud computing; computer graphics; visual analytics; visual data mining; visualization; Web-based graphics; Web-based visualization;

**About the instructor:** Haim Levkowitz is a visiting professor and a Fulbright Scholar to Brazil at ICMC, The University of São Paulo, São Carlos – SP, Brazil; and an associate professor of computer science and director of the Human-Information Interaction Research Group at the University of Massachusetts Lowell, Lowell, MA, USA. He is a world-renowned authority on visualization, perception, color, and their application in human-information interaction, data mining, and information retrieval. He is the author of "Color Theory and Modeling for Computer Graphics, Visualization, and Multimedia Applications" (Springer 1997) and co-editor of "Perceptual Issues in Visualization" (Springer 1995), as well as many papers in these subjects. He has more than 40 years experience teaching and lecturing, and has taught many tutorials and short courses, in addition to regular academic courses.

**Acknowledgments:** The author wishes to acknowledge the profound and essential contributions of Curran Kelleher to this tutorial.

## Contents

- Introduction, motivation, overview, and focus
- Visual HTML5
- Developing and deploying mobile apps
- Interactive visualization on mobile+cloud
- Data on the Web
- Concluding remarks, final thoughts, future outlook
- (Due to a page limitation, this notes-set is not complete. A relevant bibliography,additional materials, and copy of this document are available at **www.cs.uml.edu/∼haim/conferences/sibgrapi2012**.)

# SIBGRAPHI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization

Haim Levkowitz
University of Masschusetts Lowell
University of Sao Paulo, Sao Carlos





# Overview

- Why?
- Goals
- Who?
- Benefits
- Topics





# Why?

- Compute model changing
  - "visual computing" part of that
- How new wave of computing actually implemented?





- Cloud computing becoming most prevailing computing platform
- Mobile + cloud changing how users
  - consume & use compute resources
- HTML5, visual capabilities
  - canvas element
- ==> …





- ==> high-quality Web-based graphics --> reality
- WebGL ==> comparable to OpenGL
  - client-side + cloud-based
- ==> high-performance graphics and visualization "in your palm"





- mobile device = front end interface + display
- some / all graphics "heavy lifting" on cloud

# Claim:

- Most common platform for computer graphics and visualization
- in not-too-distant future
- >= 80% of visual computing migrate
  - only small percentage
    - desktop computing





# Goals

- Major components enabling shift?
- Bring previous/current practices in?
- Familiarize with underlying technologies that make this possible
  - including (but not limited to) …

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz


# Prevailing technologies

- cloud-based computing
- mobile computing
- cloud + mobile combination
- …

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz


- HTML5
  - canvas element
- WebGL graphics library
- Web-based graphics and visualization
- Web-based interactive development environments

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz


# Compare tools

- Criteria in choosing technologies
- Recommendations which tools to use

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz


# Benefits (to the world)

- Democratization
  - Tools: specialist --> everyone
  - Vis in apps for all
  - E.g., no “paper” in “newspaper”
    - E.g., Wired magazine …

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz

# Wired magazine (show)

- Paper
  - Static
- Tablet
  - Interactive content





# Wall Street Journal (show)

- Paper
  - Static
  - Some pictures
- Tablet
  - More pictures
  - Videos





# Resource shift

- Paper --> electronic
- Static --> interactive
- ==> Next few years, more publications exclusively on mobile devices
  - desktop: afterthought
  - paper: abandoned





# Future: interactive

- E.g., what's better for economy?
  - Austerity vs. stimulus
  - Interactive vis on mobile
  - ==> user: "what if?"





# Topics: Brief overview of

- the Internet;
  - history, goals, fundamentals
- the WWW;
  - history, goals, fundamentals
  - "e-bboard" --> interactive prog env





# Topics: computer graphics

- Visual computing:
  - CG vs. Image Processing vs. vision
  - objects-to-pictures vs pictures-to-pictures vs pictures-to-objects
- background material/presentations: Visual computing

# Topics:

- Introduction to
  - cloud computing
  - …





- mobile computing
  - iOS vs. Android vs. other
    - technical
    - deployment/logistic
  - native vs. Web-app
  - cross platform





- mobile and cloud computing
  - this is it!
  - “ping-pong" between local and remote

# Visual computing





# Overview

- Computer graphics
- Image processing
- Vision
- Visualization





# Computer graphics

- models/objects --> pictures
- "synthesize"
- "render"
- 2D, 3D --> 2D
- viewer, light source(s), object(s)
- realistic





# Image processing

- pictures --> pictures
- 2D --> 2D





# Vision

- pictures --> models/objects
- 2D x 2D --> 3D





# Visualization

- data --> visual
  - images
  - models
  - animation
- large amounts
- utilize perception power

# Cloud computing: introduction and overview





# Overview

- What is cloud computing
- Definitions
- Technology
- Impact
- Products and market examples





# What is cloud computing?

- Many definitions; here are some:
- Access applications hosted on web through web browser (Software as a Service -- *SaaS*)
- Pay-as-you-go model for IT resources accessed over Internet (Platform as a Service - *PaaS*)





- Use commodity computers
- distributed throughout internet
- to perform
  - parallel processing, distributed storage, indexing and mining of data





- Gartner: "… style of computing where massively scalable IT-related capabilities … provided 'as a service' across the Internet to multiple external customers"





# Common cloud themes

- They're big -- massively scalable
- Always there when need them
  - on-demand, dynamic
- Only use what need
  - elastic, no upfront commitments, use on short term basis

Accessibility = Any time, any place, any device

- Leverage low cost compute cycles
  - assured data storage in cloud
- Communications is pacing factor
- Challenge: balance platform agnostic vs. end point device innovations





# What is cloud computing?

- "location-independent compuing" (Wikipedia)
- Many definitions, always "location"
- The "cloud"
- But first, a little history ….





# A little history?

- Batch ("there"):
  - paper tapes, punch cards
  - dumm terminals
- More interactive (still "there")
  - still big "IBM/NCR …" in basement
  - big, heavy, expensive





# More history

- Mini computers (still "there")
  - PDP 11 / Eclispe / Nova
- Super minis (ditto)
  - VAX
- Worstations ("here" and "there")
  - Apollo, Sun, HP





- Client-server ("here" and "there")
  - Even graphics & interaction
    - X-Windows
- PCs ("here")
  - Apple, IBM, MS
- Laptop (very "here")





- Mobile ("here here")
- In Web browser ("here" … and "there")
  - like client-server
- How much "here"? How much "there"?

# The cloud

- resource delivery and usage model
  - hardware + software
- via network
  - “on-demand”
  - “at scale”
- in multi-tenant environment

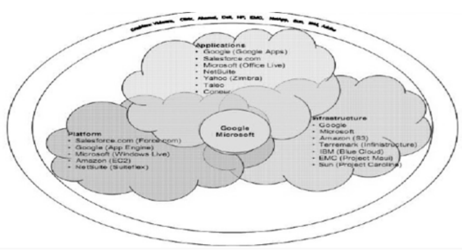


# The “cloud”

- network of providing resources
- hardware resources
  - infinitely scalable
  - can be used whenever



# The cloud






- Out there on Internet somewhere
  - accessible, location independent
- Transparent -- complexity concealed from users, virtualized, abstracted
- Service oriented
  - easy to use, SLAs, accessible



Simple Metaphor:
Like Power Company

Better Metaphor:
Cooperatively Owned
Semiconductor Fab

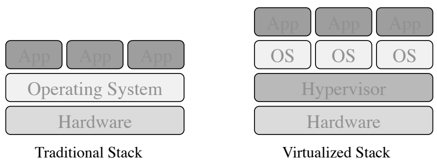


# Trends enabling (& driven by) cloud computing

- parallelism
- virtualization
- commodity components
- core elements outsourcing

# Virtualization

- key technology






# Commercial cloud providers




# Cloud deployment models

- *Internal (private) cloud*
  - cloud infrastructure operated within organization



- *Community cloud*
  - cloud infrastructure jointly owned by several organizations
  - supports specific community
    - shared concerns / needs, e.g.,
      - mission, security requirements, policy, compliance considerations



- *Public cloud*
  - cloud infrastructure owned by organization selling cloud services to
    - general public
    - large industry group



- *Hybrid cloud*
  - cloud infrastructure = composition of two or more clouds (internal, community, public)
  - remain unique entities but bound together by standardized / proprietary technology ==> enables data and application portability

## Business case for cloud computing

- Automation/On-Demand ==>
  - Better, Faster & Cheaper
- Move 'hand crafted' SW -->
  - repeatable assembly
- Reuse interchangeable components
- Repeatable processes w/ increased automation & collaboration





- Division of labor
  - developers focus on new software
- Ease of use
  - abstract complexity out of developers' lives
- Avoid over & under provisioning
  - CAPEX outlays





## Data Intensive Computing

- Index & make sense of large data sets
  - parallelization
- Pre-format data in large repositories for low BW transmissions
- Better access to data w/ large multi-tenant distributed cloud databases
- Default backup + cost effective archival of large data sets





## Tenets

- "* as a Service" (*aaS), * =
  - Software (SaaS)
  - Platform (PaaS)
  - Infrastructure (IaaS)
- Commodity HW
  - No special purpose (usually)





## Technologies (1/2)

- new advances in processors
- virtualization
- distributed storage
- broadband Internet access
- automated management
- fast, inexpensive servers





## Technologies (2/2)

- parallel computing
- distributed, large-scale server clusters
- server virtualization software

# Cloud computing market

SaaS: Software as a Service | Everything as a Service

PaaS: Platform as a Service

Iaas: Infrastructure as a Service

cloud technology enabler

hardware provider





# SaaS
# Software as a Service

- Software delivery model
- No hardware or software to manage
- Service delivered through a browser





# SaaS

## Advantages

- Pay per use
- Instant Scalability
- Security?
- Reliability?
- APIs





# PaaS
# Platform as a Service

## Platform delivery model

- Platforms built upon Infrastructure
  - expensive
- Estimating demand difficult
- Platform management not fun!





# PaaS

## Popular services

- Storage
- Database
- Scalability





# PaaS

## Advantages

- Pay per use
- Instant Scalability
- Security?
- Reliability?
- APIs

## PaaS

### Examples

- Google App Engine
- Mosso
- AWS: S3 (Amazon Web Services: Simple Storage Service)



## IaaS
## Infrastructure as a Service

### Computer infrastructure delivery model

Access to infrastructure stack:

- Full OS access
- Firewalls
- Routers
- Load balancing



## IaaS

### Advantages

- Pay per use
- Instant Scalability
- Security?
- Reliability?
- APIs



## IaaS

### Examples

- Flexiscale
- AWS: EC2 (Amazon Web Services: Elastic Compute Cloud)



## {S/P/I}aaS

### Common Factors

- Pay per use
- Instant Scalability
- Security?
- Reliability?
- APIs



### Advantages

- Lower cost of ownership
- Reduce infrastructure management responsibility
- Allow for unexpected resource loads
- Faster application rollout

# Visual HTML5





# Overview

- HTML5
- Building apps with HTML5
- Visual HTML5
  - Canvas
  - SVG - Scalable Vector Graphics
  - WebGL
  - 2D + 3D libraries





# HTML5

- HTML5 to Cloud & Mobile = Java to desktop computing:
  - cross-platform application building technology





# HTML5 (2)

- "revolutionizing the way the Web evolves, works, and is used"
  - -- "HTML5 leads a Web revolution" CACM, July 2012
- single specification
- whole set of technologies





# HTML5 (3)

- Associated standards
  - Document Object Model (DOM)
    - access & manipulate HTML docs
  - Cascading Style Sheets (CSS)
    - define presentation & appearance of HTML doc
  - JavaScript





# HTML5 (4)

- More broadly, include specific (APIs)
  - enable new browser-based graphics
  - geolocation
  - local storage
  - video capabilities

## HTML5 (5)

- Heart of W3C's Open Web Platform
- umbrella term
  - changes over time
- markup language & various technologies that pertain to it





## HTML5 (6)

- "The Web over 20 years has developed from a Web of more-or-less static documents to, now, a platform for applications."
- HTML = centerpiece
- HTML5 = most recent + most capable





## Roles

- W3C oversees development
  - HTML5 + graphics capabilitie
- W3C governs HTML5 standards





## Roles (2)

- Vendors
  - implement standards independently
  - freedom to innovate
    - e.g., Apple introduced Canvas
  - innovations --> W3C standards
  - ==> …





## Roles (3)

- standards implementations continually increase in quality
- technology evolution not responsibility of any single entity





## Building apps with HTML5

- Avoids vendor lock-in
- affords compatibility across most desktop browsers and mobile devices

# Visual HTML5

- "Visual" =
  - graphics
    - 2D + 3D
  - image
  - audio-visual





# Primary W3C graphics technologies today

- Canvas
- WebGL
- Scalable Vector Graphics (SVG)
- Graphics and visualization libraries built on these
- ==> higher levels of abstraction for working with interactive graphics





# 2D





# HTML5 Canvas

- *immediate-mode* graphics API for Web
- no new concept
- well known 2D graphics techniques into HTML and JavaScript
- Many similar APIs before Canvas …





# Similar APIs before Canvas

- Cairo
- Java2D
- .NET graphics libraries
- QT
- GTK





# Frameworks

- Give developers full access to graphical display
- ==> developers to build entire apps ground
- custom designed looks and feels & graphical behavior

## Now, on WWW + Mobile

- Capability to
  - Web
  - mobile devices for viewing
- ==> "Canvas is the single most powerful HTML5 element"
  - Geary (2012) "Core HTML5 Canvas: Graphics, Animation, and Game Development"





## SVG (Scalable Vector Graphics)

- DOM-based W3C standard for retained-mode vector graphics
- developers specify 2D scene graph by manipulating DOM
- SVG implementation responsible for rendering scene to bitmap for display whenever updates occur





- Data Driven Documents (D3.js)
- notable library
  - based on use of SVG
  - for developing interactive visualizations





## HTML5 Canvas API





## Canvas API

- Developers can insert rectangular bitmap
  - as element in HTML page
- access it through JavaScript API
- Canvas bitmap = array of colored pixels





## Canvas API (2)

- API provides functions for manipulating bitmap
- using well known graphics concepts and techniques

# Canvas API (3)

- Originally defined and implemented by Apple in 2004
- subsequently proposed as standard





# Canvas API (4)

- Today implemented in all major browsers
- polyfills
  - e.g., ExplorerCanvas and Google Chrome Frame
- support backward compatibility with older browsers





# Canvas API (5)

- Capabilities of Canvas in detail





# Canvas API (6)

- Uses stateful context
  - for determining properties of graphical elements drawn
- Canvas 2D context includes following variables
- …





# Canvas API (7)

- Variables:
  - Include table/list of var's or refer to survey paper





# Canvas API (8)

- 1. set context variables
- 2. draw paths the canvas
  - use variable values
- push + pop functions
  - context variables to/from stack,
    - context.save, context.restore

# Canvas API (9)

- Canvas element: functions
  - draw
    - lines, polygons, arcs, text
  - fill paths
    - solid colors, gradients (linear &radial), repeated image patterns





# Canvas API (10)

- Bezier curves support
- styles
  - line caps, joins
- comprehensive text drawing API

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz


# Canvas API (11)

- Exposes bitmap pixels directly
  - ==> allow construction of filters
    - e.g., blur, invert, emboss
  - also enables implementation of computer vision algorithms

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz


# Canvas API (12)

- Image data can be
  - exported
  - uploaded to a server

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz


# Canvas API (13)

- Compositing API
  - composition of multiple bitmaps
  - using various techniques
- Shadows
  - severe performance penalty

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz


# Canvas API (14)

- Can load and draw on Canvas
  - external image data
  - frames from HTML5 Video element

SIBGRAPI 2012 Tutorial: Cloud and mobile Web-based graphics and visualization
Human Information Interaction for Knowledge Extraction, Interaction, Utilization, Decision making
HI-I-KEIUD
© Copyright 2012 Haim Levkowitz

## Canvas API (15)

- Alone not enough to build interactive graphics applications





## Canvas API (16)

- Events provided by HTML DOM APIs
  - mouse and keyboard
  - multi-touch from mobile devices
- must
  - be accessed through JavaScript
  - drive graphic manipulations





## Canvas API (17)

- *requestAnimationFrame*: special function for implementing smooth animations
- …





## Canvas API (18)

- Better alternative to *setInterval* for animation loop
- because provides timing of animation frames
  - synchronized with refresh rate of graphics HW
  - …





## Canvas API (19)

- Implementations can provide additional optimizations
  - e.g., do not execute animations when page not visible





## Canvas API (20)

- JavaScript single threaded
  - ==> intensive computations that would slow animation loop
    - e.g., physics /image processing
  - can offload to other threads
    - using Web Workers

# 2D Graphics Libraries



# Processing.js

- Library for immediate-mode graphics
- uses HTML5 Canvas
- based on original Java-based Processing project



# Processing.js (2)

- Two majore components:
  - Processing language parser …
  - Processing graphics API implementation



# Processing language parser

- Transforms source code written in Java-like Processing language into JavaScript
- ==> enables developers to execute Processing programs in HTML5 environment
- with little /no code modification



# Processing graphics API implementation

- Uses
  - HTML5 Canvas -- for 2D features
  - WebGL -- for 3D features



# In practice

- Many developers prefer to use Processing API from JavaScript
- rather than write software in Processing language
- because …

# Processing.js (6)

- Because introduction of another language into project introduces complexity, and
- because Processing programs are difficult to debug





# Paper.js

- Scene graph library for 2D graphics
- uses HTML5 Canvas
- provides vector graphics scene graph
  - similar to Adobe Illustrator





# Paper.js (2)

- API inspired by and mostly compatible with Adobe Scriptographer
  - JavaScript scripting plugin for Adobe Illustrator
    - created by developers of Paper.js





# Paper.js (3)

- Provides support for common graphics primitives
  - groups, layers, paths (Bezier curves) with outline drawing, mouse and keyboard interaction, working with raster images, vector geometry operations





# Paper.js (4)

- Main contribution:
  - brings vector graphics model of Adobe Illustrator to Web
  - ==> giving developers straightforward cross-browser retained-mode 2D graphics API





# 3D

# WebGL

- JavaScript API for OpenGL ES 2.0
  - simplified OpenGL
- immediate-mode 3D graphics capability
- not fundamentally new
- hardware accelerated 3D graphics to HTML and JavaScript





# WebGL brings capabilities to WWW

- Render 3D scenes
- lighting
- textures
- shaders definition





# WebGL vs. OpenGL ES

- ES (= Embedded Systems)





# Scene graph libraries

- e.g., Three.js ==>
- retained-mode 3D graphics implementations





# Physics libraries

- e.g., Box2D.js ==>
  - physics simulation





# 3D Graphics Libraries

# Three.js

- = JavaScript library for 3D graphics
- abstracts rendering layer
- ==> allowing developers to write same code for rendering 3D graphics using either WebGL, HTML5 Canvas, or SVG



# Three.js (2)

- Introduces standard 3D graphics abstractions
  - scene graph, cameras, meshes, materials, textures, lighting models, common objects
    - e.g. cubes and spheres



# Three.js (3)

- Support for loading 3D models in Collada file format, and more
- Plenty examples using Three.js published on Web



# Three.js (4)

- Physics engine for Three.js
  - Physijs
  - built on Ammo.js
    - direct port of Bullet physics engine to JavaScript



# Data Driven Documents (D3)

- JavaScript library
  - written by Mike Bostock
- for 2D Web-based interactive data visualization



# D3 (2)

- Solves problem of performing Document Object Model (DOM) manipulation based on data
- declarative approach leveraging functional programming techniques
- ==> …

## D3 (3)

- Developers write concise statements to manipulate DOM
  - based on data
- rather than write verbose and convoluted data transformation code using raw DOM AP



## D3 (4)

- Does not introduce
  - own graphics API
  - scene graph model
- provides developers powerful tool for leveraging Web standards
  - e.g., SVG and CSS



## D3 (5)

- Successor to ProtoVis
  - visualization library
    - introduced own graphics vocabulary



## HTML5 wrap up

- Being deployed now
- not finished standard
- adoption varies



## HTML5 wrap up (2)

- E.g., no single standard for
  - video compression (codec)
  - streaming protocol
  - digital rights management (DRM)



## HTML5 wrap up (3)

- Video standards
  - Flash = NOT!
  - Apple iOS ==>
    - no Flash, yes HTML5
  - MS, Google – diff approaches
  - Browsers' coverage varies

# HTML5 wrap up (4)

- “But the individual specifications are at different maturity levels and will become standards at different times”
  - -- W3C

# Developing and deploying mobile apps





# Overview

- HTML5: "write once, run anywhere"
  - for developing mobile applications





# Native app alternative

- Android -- Java
- iOS -- Objective-C
- Windows Mobile -- MS tools
- ==> duplicate
  - effort
  - cost
  - skill sets





# HTML5 alternative

- Write apps once w/HTML5
  - + mobile-specific compatibility tricks
- deploy as native apps to all platforms
- available tools …





# HTML5 --> native tools

- PhoneGap
- Appcelerator Titanium
- ==>
  - powerful force HTML5 --> most widely used app dev approach





# Native apps

- Capabilities not available to Web pages
  - due to browser sandboxing

## Native apps (2)

- Access to local file system
  - native OS features
    - e.g., launch sub-applications with Android intents
  - hardware accelerated graphics
  - accelerometer





## Native apps (3)

- Camera
- compass
- geolocation
- notifications with alerts, sound, or vibration





## Native apps (4)

- Access to
  - user's contacts
  - revenue-generating app marketplaces
    - Apple Store
    - Android Market





## Native apps (5)

- High cost for multi platform dev
- repeat for each platform
- learn proprietary app dev tools
- port code to different programming languages
- alternative? …





## HTML5 apps

- HTML5 : viable cross-platform solution
- Dev app in HTML5
- Deploy …





## Deploy as

- Web page for
  - desktop browsers
  - Web page for mobile browsers
- native application for
  - each mobile platforms
- ==> …

## ==>

- Apps originally developed w/ HTML5
- ==> reach much larger audience
  - ==> much larger profit
- lower cost of application development



## HTML5 vs. native app

- HTML5 apps can "feel" like native
- but not automatic
- tricks to achieve native app "look-and-feel" …



## "Native" tricks

- Detect display resolution
- render appropriately scaled version of app
- different layouts for different display resolutions
- …



## "Native" tricks (2)

- If possible, scale user interface elements to match the display
  - simpler



## "Native" tricks (3)

- Detect device orientation changes
- render accordingly
- user interface elements might need to have different layout for different orientations
- ensure all work well



## "Native" tricks (4)

- Hide browser "telltales"
  - mobile browser's navigation bar
  - other distracting visual elements
  - ==> user only sees app full screen

## “Native” tricks (5)

- Use icon at correct resolution to look like app icon
  - shortcuts to saved pages ==> icon on mobile desktop
  - can be specified by page
  - …





## “Native” tricks (6)

- Resolution properly configured
- ==> icon on mobile desktop looks exactly like native app icon





## “Native” tricks (7)

- Use HTML5 local storage API storage
- ==> app can function without Internet connection
  - read & write state





## “Native” tricks (8)

- Even with cloud-driven apps
  - local storage temporary holds changes
  - synch’d with cloud service
    - when connection re-established
- ==> …





## “Native” tricks (9)

- ==> difference between usable app and unusable app





## Cross platform dev tools and environments

# Cross platform dev

- Most mobile dev platforms
- have UI component
- can display Web page within native app





# Cross platform dev (2)

- This means it is possible to build native apps that
- simply load a full screen view of a Web page, which can itself
- be a full featured HTML5 app. In this way, HTML5 apps can
- be deployed as native apps





# Cross platform dev (3)

- Native mobile app can displays Web page
- can modify page's runtime
  - ==> arbitrary functions in native app available to JavaScript code on loaded Web page





# Cross platform dev (4)

- ==> arbitrary functionality available to native apps
  - can be exposed to Web pages
  - via JavaScript API
- ==> HTML5 apps can gain all functionality available to native apps





# Cross platform dev (5)

- ==> pattern for deploying HTML5 apps as native apps:





# Cross platform dev (6)

- 1. Build native app that displays Web page
- 2. Expose native functionality to Web page via JavaScript API
- …

## Cross platform dev (7)

- 3. Build HTML5 app as Web page embedded within native app
  - take advantage of native functionality
    - via JavaScript API





## PhoneGap

- Pattern HTML5 apps --> native mobile apps straightforward
- ==> can be automated
- mobile device capabilities similar
- across platforms
- ==> JavaScript API to native functionality can be standardized





## PhoneGap (2)

- PhoneGap project automates HTML5 apps --> native mobile apps
- specifies single JavaScript API for accessing native capabilities across many mobile platforms





## PhoneGap (3)

- Using PhoneGap, developers can
  - author HTML5 apps once
  - derive native apps for multiple platforms in single automated step





## PhoneGap (3)

- Once native apps created
- author can go through typical steps
  - deploy app to revenue generating app marketplace
    - Apple App Store
    - Android Market





## Appcellerator Titanium

- Comprehensive mobile app development platform
- based on
  - JavaScript APIs
  - HTML5
- includes …

## Appcellerator Titanium (2)

- Software SDK for developing mobile apps
- Eclipse-based IDE equipped with platform-specific tooling
- suite of cloud-based services to speed app development
- app analytics service





## Appcellerator Titanium (3)

- Titanium Mobile SDK ==>
  - write app once in JavaScript
  - automatically deploy to severalmobile platforms
    - native apps
    - mobile Web





## Appcellerator Titanium (4)

- Process similar to PhoneGap
- Appcellerator dev env
  - Titanium Studio
    - Eclipse-based IDE
    - platform-specific functionality





## Appcellerator Titanium (5)

- ==> develop, test, deploy mobile apps





## Appcellerator Cloud Services

- Scalable back end features
- common to many apps
- normally needed to be implemented by app developers
- cloud features: user management, logins, photo uploads, push notifications, status updates





## Appcellerator analytics

- Service aids app developers
  - collect
  - analyze
- data about
  - users, sessions, actions taken within apps

# Deploying apps

- 1) Android Market …
- 2) iOS App Store …
- 3) Chrome Web Store …
- 4) Amazon App Store …
- 5) Zeewe …
- 6) TapJS …



# Deploying apps

- How to
  - "get your app out there"
  - generate revenue



# June 2012 stats

- 51.8% of smart phones -- Android
- 34.3% -- iOS
- ==> corresponding app marketplaces
  - Google Play (= "Android Market")
  - iOS App Store
- ==> reach most users



# Google Play (= "Android Market")

- Publish apps in Android Market
- payment and deployment to client devices managed by platform
- now, new interface, new app discoverability features
  - e.g., recommendation system

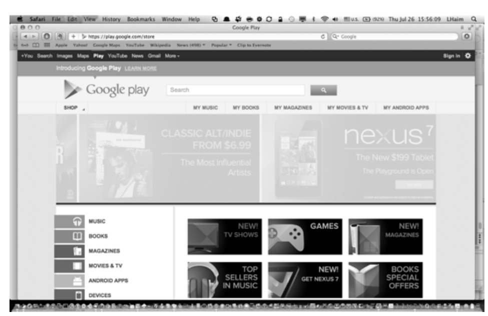


# Google Play (= "Android Market")

- Sells also music, books, movies
- Publishing = automated process
- free for developers
- straightforward app preparation and Web-based publishing process
- ==> app appea in marketplace within minutes



# https://play.google.com/store



# iOS App Store

- For iOS devices
  - iPhone, iPod Touch, iPad
- very strictly moderated,
- costs developers $99 per year
  - submit, & keep apps available at Store





# iOS App Store (2)

- Apps rejected for nebulous reasons






# Chrome Web Store

- App marketplace from Google
- aimed at Chrome users
- **"Chrome App" = Web app**
- **additional metadata stored by Chrome**
- **icon to launch app from Chrome home page**



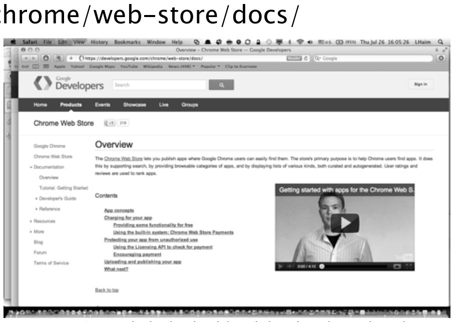


- Many apps free
- collecting payment possible
- when Chrome Web Store Payments system used
- $5 to become app author on Chrome Web Store
- no recurring fees





https://developers.google.com/chrome/web-store/docs/








# Other markets

- Amazon's App Store
- Zeewe, a marketplace for Mobile Web Apps
- TapJS, a game hosting service
- Playtomic

# Conclusion

- Based on market share statistics
- reach largest majority of potential audience by focusing on
- Apple's App Store (the only marketplace for iOS apps)
- Google Play for Android apps

# Interactive visualization on mobile+cloud



# Overview

- Google Body Browser …
- Google Maps …
- TileMill …
- Tableau Public …
- D3 examples built for mobile devices …
- more examples?



# Google Body Browser

- Google Body Browser
- now known as Zygote Body
  - http://www.zygotebody.com/
- virtual human body
- “visible human” project
- exemplary example application of WebGL



# Google Body Browser (2)

- peel away layers
  - skin, muscles, bones, vascular system, nervous system
- zoom & rotate 3D model
- click on body part
- ==> highlight & get name

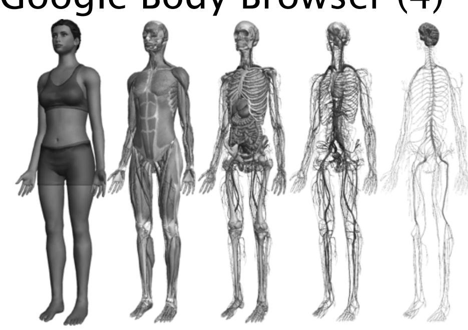


# Google Body Browser (3)

- search box ==> find body parts by name



# Google Body Browser (4)



# Google Body Browser (5)

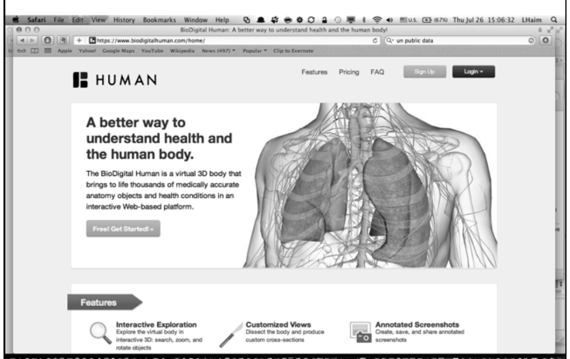
- has been ported to native Android app
- http://code.google.com/p/gdc2011-android-opengl/wiki/TalkTranscript
- ==> take advantage of native app features, e.g., data bundling, accelerated graphics



# Google Body Browser (6)

- Other similar projects
  - such BioDigital Human
    - https://www.biodigitalhuman.com/home/








# Google Maps

- Web-based mapping service perfect example of visual computing app
  - graphics “heavy lifting” of rendering map images
  - on server side



# Google Maps (2)

- navigate maps
- several views
  - satellite imagery, street maps, bicycle maps, terrain, weather, photos, traffic, more
- find directions from place to place
- share map configurations



# Google Maps (3)

- exists as
  - Web app for
    - desktop browsers
    - mobile devices
  - optimized native app for mobile platforms

# Google Maps (4)

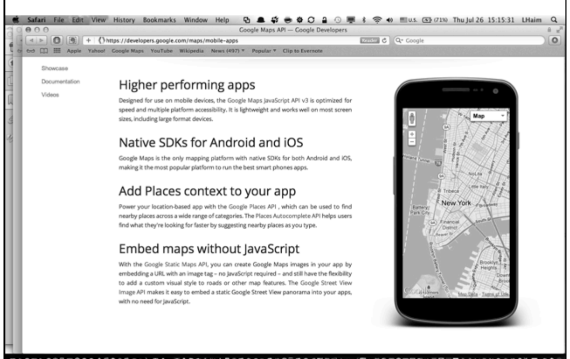
- native apps take advantage of device's geolocation
- provide additional features
  - "show me where I am now"
  - live driving and walking directions



# Google Maps (5)

- Google also provides API
- ==> developers effortlessly embed maps within their apps








# TileMill

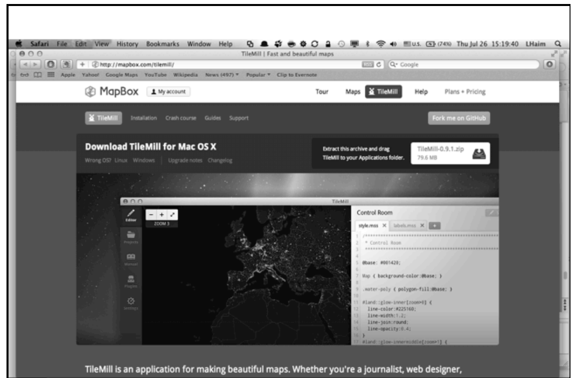
- http://mapbox.com/tilemill/
- open source project
- managed by MapBox
- render map tiles on server side
- styled using own CSS-like map styling
- language called CartoCSS



# TileMill (2)

- based on open source projects
  - Mapnik and Node.js
- powerful tool for developing interactive map applications
- can, e.g., overlay data vis on map, or
- re-style existing map data
  - e.g., OpenStreetMap





# Tableau Public

- http://www.tableausoftware.com/public/
- adaptation of Tableau visualization SW to social Web





# Tableau Public (2)

- Web-based version of Tableau
- enable interactive visual analysis
- relay user interactions to server-side instances of Tableau
- render visualization on server
- send rendered image to client





# Tableau Public (3)

- gallery feature
- ==> users can post visualizations
- comment on them
- share them across Web
- embed them into their own pages










# Web-based development showcases

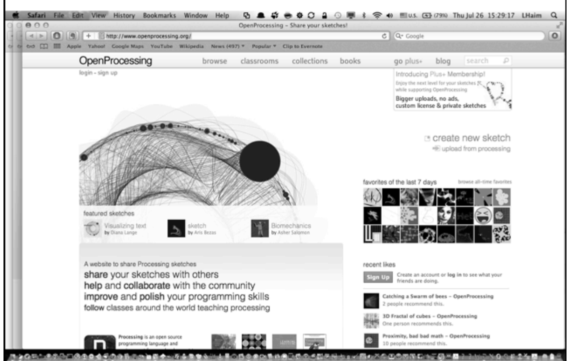
- OpenProcessing …
- Google Chrome Experiments …





# OpenProcessing

- **http://www.openprocessing.org/**
- online community platform
- devoted to sharing and discussing *Processing* sketches
- in collaborative, open-source environment

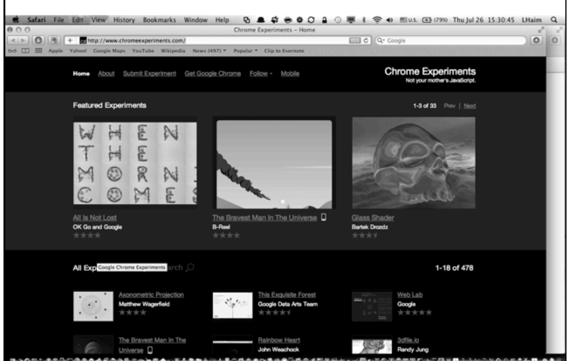





# Google Chrome Experiments

- http://www.chromeexperiments.com/
- initiative to collect example applications of HTML5 technology








# Google Chrome Experiments (2)

- Examples:
- The Wilderness Downtown …
- 3D Water Waves …
- Progressive Julia Fractal …
- WebGL Experiments …



# The Wilderness Downtown

- interactive music video
- enter your address when it starts
- then builds dynamic scenes
- based on images from that address
- taken from Google Maps and Google Street view



# 3D Water Waves

- demonstration of
  - fluid simulation
  - advanced 3D graphics techniques
- using WebGL

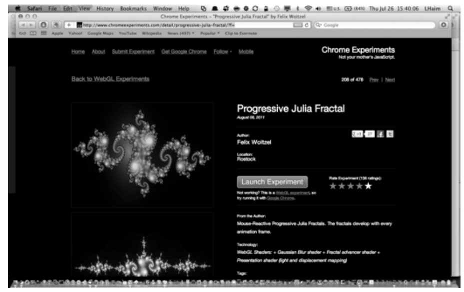
# Progressive Julia Fractal

- lightning fast implementation of Julia set rendering
- uses shader-based image distortion
  - by Felix Woitze
- demonstrates potential for general purpose GPU computing using WebGL





# http://www.chromeexperiments.com/detail/progressive-julia-fractal/?f=







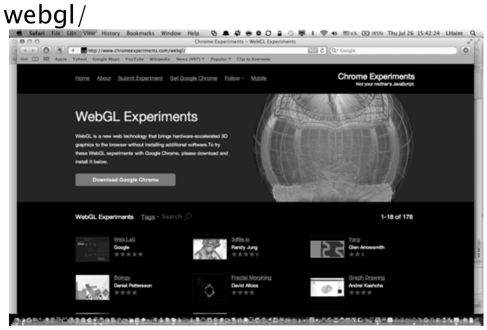

# WebGL Experiments

- http://www.chromeexperiments.com/webgl/
- demonstrations of several other WebGL-based implementation





# http://www.chromeexperiments.com/webgl/








# Conclusion

- Some examples of interactive mobile +cloud visualizations

# Data on the Web



# Overview

- Semantic Web and the democratization of data
- public data initiatives



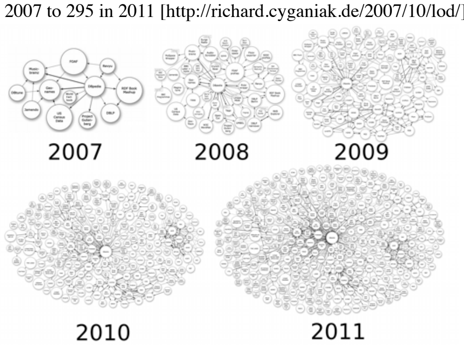
- Mobile+Cloud will democratize data analysis!
- technologies & initiatives
  - already launched
  - demonstrate this evolution



# Semantic Web and the democratization of data



Growth of Linked Open Data Cloud, from 12 datasets in 2007 to 295 in 2011 [http://richard.cyganiak.de/2007/10/lod/]






# Semantic Web

- "Web of Data"
- "Giant Global Graph"
- "Data Web"
- "Linked Data Web"
- "Enterprise Information Web"

## Semantic Web (2)

- key vision:
  - link explicit data published on Web
  - machine-readable
  - ==> enable applications
    - targeted search, data browsing, intelligent agents





## Resource Description Framework (RDF)

- foundational data rep'n framework for Semantic Web
- data as triples
  - (*subject*; *predicate*; *object*)
- ==> can represent any data stored in relational databases





## Web Ontology Language (OWL)

- Higher level of Semantic Web
- represent domain ontologies and inference rules





## Vocabularies

- provide data publishers common means to express domain concepts
- RDF Data Cube Vocabulary
  - enable integration of statistical data from many data providers





- More public data sets published within Semantic Web





## Semantic Web + Web-based Visual Computing

- Visual data analysis tools built to consume SW data
- ==> able to access data as it becomes available

- ==> rich Web-based visualization and analysis tools
- applied to SW information



# Public data initiatives

- UN, World Bank, US Federal Data, more …
- Example visualizations of public data:
  - GapMinder
  - Italy budget Viz using D3
  - BLS employment vis



# UN, World Bank, US Federal Data, more

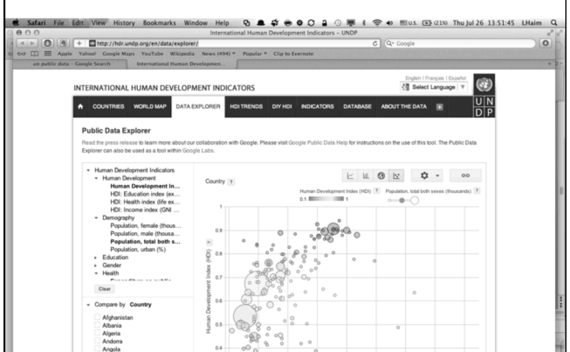
- UN published
  - 34 databases
  - 60 million records
- indicators over years / countries
- economics, energy, human dev, crime, health, tourism, telecomm, +



# Int'l Human Dev Indicators

- http://hdr.undp.org/en/data/explorer/
- Public Data Explorer

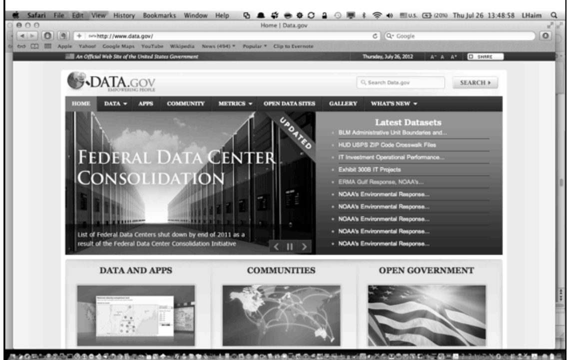




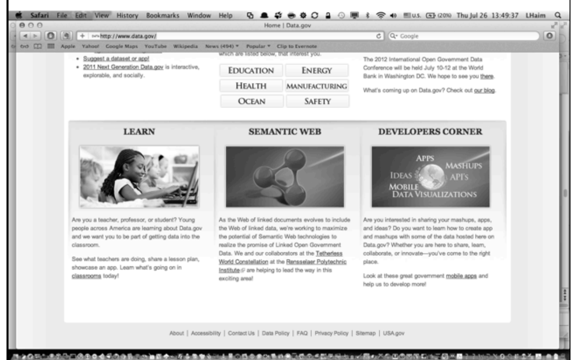



# Data.gov

- access to many data sets





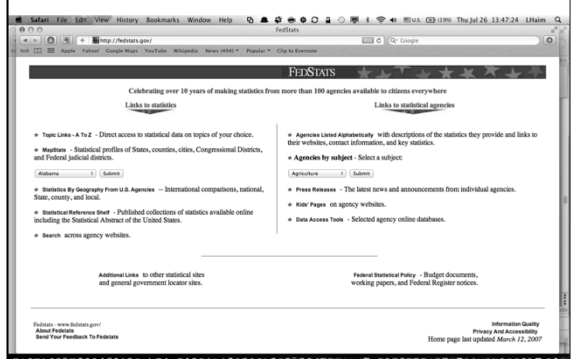






# Fedstats (fedstats.gov)

- links to numerous public data sets
- hosted by US government organizations








# Example visualizations of public data:

- GapMinder World …
- Italy budget Viz using D3 …
- BLS employment vis …



# GapMinder World

- http://www.gapminder.org/world/
- Web-based visualization
  - important trends around the World




# GapMinder World (2)

- interactive visualization
- ==> move through time
  - 1800 – today
- interact with various parameters
- ==> see how trends changed over time, geo location + other categories



# GapMinder World (3)

- Parameters:
- income / person, children / woman, child mortality, life expectancy, economy, society, education,
- energy, environment, health, infrastructure, population and work



# GapMinder World (4)

- Each parameter --> additional choice granularity
- Advanced users ==> advanced features
- ==>further enrich interaction, exploration



# GapMinder World (5)

- Some examples:
- **"Wealth & Health of Nations"**
  - **how long people live**
  - **how much money they make**
- "$CO_2$ emissions since 1820"
- …



# GapMinder World (6)

- "Africa is not a country!"
  - **huge diff's among African countries**
- "Is child mortality falling"
- "Where is HIV decreasing?"
  - changes in the number of people living with HIV

# Italy budget Viz using D3

- http://www.visup.it/misc/workshop/index.htm
- Web-based visualization
- access to public data
- Italy's administrative expenses during
  - 2002-2008






# Italy budget Viz (2)

- hover ==>
- effortless and effective navigation
- across 3 dims of data cube
- **D3 implementation**
- access via Firefox, Safari, Opera
  - not Internet Explorer



# BLS employment vis

- CNN Economy Tracker — US Bureau of Labor Statistics (BLS) employment visualization
- http://www.cnn.com/SPECIALS/map.economy/index.html?mapIndex=3&hpt=C2






- interactive Choropleth map
- colored in proportion to economic variable explored
- probe states + get detailed values
  - colored relative to variables
- data spans across US
  - states, industry, time

# Conclusion

- Public data
- + Semantic Web
- + Web-based interactive visualization
- ==> access to information
- “what if”?
- info-based learning + decision making

# Conclusions, final thoughts





## Conclusions

- Mobile and cloud ==> prevailing computing paradigm
- Graphics, visualization to take advantage
- HTML5 + Canvas + WebGL + SVG + other libraries





## Final thoughts

- Visual (and other) computing:
  - Local interaction – on mobile
  - “Horsepower” – on cloud
- Need
  - balancing
  - synchronization