

Aalto University
School of Science
!FIXME **Set degree program** FIXME!

Kimmo Puputti

!Fixme **Add English title** Fixme!

!Fixme **Add English subtitle** Fixme!

Master's Thesis
Espoo, !FIXME **Add English date** FIXME!

DRAFT! — Monday 9th January, 2012 — DRAFT!

Supervisor: Professor Petri Vuorimaa, Aalto University
Instructor: Risto Sarvas D.Sc.(Tech.)

Aalto University
School of Science

!FIXME Set degree program FIXME!

ABSTRACT OF
MASTER'S THESIS

Author:	Kimmo Puputti		
Title:	!FIXME Add English title FIXME! !FIXME Add English subtitle FIXME!		
Date:	!FIXME Add English date FIXME!	Pages:	19
Professorship:	Media Technology	Code:	T-110
Supervisor:	Professor Petri Vuorimaa		
Instructor:	Risto Sarvas D.Sc.(Tech.)		
!FIXME Add English abstract FIXME!			
Keywords:	!FIXME Add English keywords FIXME!		
Language:	English		

Aalto-yliopisto
 Perustieteiden korkeakoulu
 Tietotekniikan tutkinto-ohjelma

DIPLOMITYÖN
 TIIVISTELMÄ

Tekijä:	Kimmo Puputti		
Työn nimi:	!FIXME Add Finnish title FIXME! !FIXME Add Finnish subtitle FIXME!		
Päiväys:	!FIXME Add Finnish date FIXME!	Sivumäärä:	19
Professuuri:	Mediatekniikka	Koodi:	T-110
Valvoja:	Professori Petri Vuorimaa		
Ohjaaja:	Tohtori Risto Sarvas		
!FIXME Add Finnish abstract FIXME!			
Asiasanat:	!FIXME Add Finnish keywords FIXME!		
Kieli:	Englanti		

Acknowledgements

!FIXME Add acknowledgements FIXME!

Thank you.

!FIXME Decide city... FIXME!, !FIXME Add English date FIXME!

Kimmo Puputti

Contents

0.1	Thesis Git repository info	7
1	Introduction: Smartphone Market and the Need for Cross-Platform Support	8
1.1	Smartphone Landscape	9
1.2	HTML5	9
1.2.1	History	9
1.2.2	Markup	9
1.2.3	CSS3	9
1.2.4	JavaScript APIs	9
1.2.5	Related APIs	9
1.3	Modern Mobile Web Application Architecture	9
1.3.1	Single-Page applications	9
1.3.1.1	JavaScript MVC Libraries	9
1.3.2	Responsive Design	9
1.3.3	Progressive Enhancement	9
1.3.4	UI Libraries	9
1.3.4.1	jQuery Mobile	9
1.3.4.2	jQTouch	9
1.3.4.3	Sencha Touch	9
1.3.5	Hybrid Applications	9
1.3.6	Wrapping Web Applications Application Stores	9
1.4	Performance Guidelines	9
2	Research Question: HTML5 - Hype versus Realities?	11
3	Methods: Example Application and Library	12
3.1	Qt Developer Days 2011 Conference Schedule Application . . .	12
3.1.1	Application Architecture	12
3.2	JSONCache JavaScript Library	13

4	Results: What Was Good and Where Were the Compromises	15
4.1	Targeting Different Platforms	16
4.1.1	Device Detection	16
4.1.2	Feature Detection	16
4.2	Targeting Different Screens	16
4.3	Handling Mobile Networks	16
4.3.1	Minimizing Data Transfer	16
4.3.2	Caching	16
4.3.3	Preloading	16
4.3.4	Offline Support	16
4.3.5	Handling Interruptions	16
4.4	Graphics and Animations	16
4.5	Performance Analysis	16
4.5.1	YSlow	16
4.5.2	PageSpeed	16
5	Discussion: Bright Future Ahead for HTML5	17
6	L^AT_EXtest	18
6.1	Citing	18

0.1 Thesis Git repository info

Build time: Monday 9th January, 2012 14:59

Git HEAD:

```
commit 8e1a41ea9b113a430b945b950a58cde843c5a0c3
Author: Kimmo Puputti <kpuputti@gmail.com>
Date:   Mon Jan 9 14:13:08 2012 +0200
```

Change perf rules into a list.

Repository status:

```
# On branch master
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
# modified:   methods.tex
# modified:   thesis.pdf
# modified:   thesis.tex
#
no changes added to commit (use "git add" and/or "git commit -a")
```


Chapter 1

Introduction: Smartphone Market and the Need for Cross-Platform Support

1.1 Smartphone Landscape

1.2 HTML5

1.2.1 History

1.2.2 Markup

1.2.3 CSS3

1.2.4 JavaScript APIs

1.2.5 Related APIs

1.3 Modern Mobile Web Application Architecture

1.3.1 Single-Page applications

1.3.1.1 JavaScript MVC Libraries

1.3.2 Responsive Design

1.3.3 Progressive Enhancement

1.3.4 UI Libraries

1.3.4.1 jQuery Mobile

1.3.4.2 jQTouch

1.3.4.3 Sencha Touch

1.3.5 Hybrid Applications

CHAPTER 1. INTRODUCTION: SMARTPHONE MARKET AND THE NEED FOR CROSS-P

- Use a Content Delivery Network
- Add an Expires Header
- Gzip Components
- Put Stylesheets at the Top
- Put Scripts at the Bottom
- Avoid CSS Expressions
- Make Javascript and CSS External
- Reduce DNS Lookups
- Minify JavaScript
- Avoid Redirects
- Remove Duplicate Scripts
- Configure ETags
- Make Ajax Cacheable
- Splitting the Initial Payload
- Loading Scripts Without Blocking
- Coupling Asynchronous Scripts
- Positioning Inline Scripts
- Writing Efficient JavaScript
- Scaling with Comet
- Going Beyond Gzipping
- Optimizing Images
- Sharding Dominant Domains
- Flushing the Document Early
- Using Iframes Sparingly
- Simplifying CSS Selectors

Chapter 2

Research Question: HTML5 - Hype versus Realities?

Chapter 3

Methods: Example Application and Library

3.1 Qt Developer Days 2011 Conference Schedule Application

The Qt Developer Days¹ is a conference for developers using the Qt cross-platform application and UI (?) framework². We created a mobile web application for conference with contextual and personalized session information and per-day schedule.

3.1.1 Application Architecture

The conference schedule³ is a single-page application (citation needed) with a lightweight backend written in Python using the Django Web Framework⁴.

The backend provides the static assets (JavaScript, CSS (?), images, etc.) and an API (?) for persisting session feedback to a MySQL⁵ relational database. It also generates the HTML5 AppCache (citation needed) offline cache manifest file based on the device type.

The frontend is a JavaScript application written using the Backbone⁶ MVC (?) framework. Other used JavaScript libraries include Underscore⁷,

¹<http://qt.nokia.com/qtdevdays2011/>

²<http://qt.nokia.com/>

³<http://m.qtdevdays2011.qt.nokia.com/>

⁴<https://www.djangoproject.com/>

⁵<http://www.mysql.com/>

⁶<http://backbonejs.org/>

⁷<http://underscorejs.org/>

jQuery⁸, Handlebars⁹, and Modernizr¹⁰. The HTML5 Mobile Boilerplate¹¹ was used as an initial markup structure for the application.

Wireless networks can be unreliable in conference settings, so offline support was also added using several different JavaScript techniques and HTML5 APIs.

The application was designed for touch screens on various platforms and screen sizes. The layout adjusts to the available space and provides rich interactive components. Integration to social networking services was also added to the application.

3.2 JSONCache JavaScript Library

JSONCache is a lightweight JavaScript library for fetching JSON (?) data in flaky networks. The library was designed especially to handle flaky mobile networks with connection problems and short interruptions. The goal is to avoid networking as long as possible and failing gracefully if network connections are not stable.

JSONCache provides two main functionalities: data caching and attempting to fetch the data multiple times.

The caching layer uses the client side localStorage (citation needed) cache of HTML5 (?). Data requests can be done using the JSONCache API (?) which always checks the local cache first before opening any network connections. If the data is already in the cache, the cached data is checked for validity and if the data has not been expired, it is returned immediately. If the data is not in the cache or it has been expired, a new network request is made and the received data is cached and returned to the requestor. The expiration time of a data item can be configured in the library settings.

JSONCache also tries to fetch the data multiple times to handle small interruptions in network connection. !FIXME **add example and explain that it is very common** FIXME!. If a data fetch fails, a new fetch is issued after a timeout (defined in the configuration). On subsequent attempts the timeout is increased, and after a defined number of attempts the fetch error is issued to the requestor.

Figure 3.1 shows an interactive demo of the JSONCache library. The demo¹² simulates the caching and fetching functionality of the library by

⁸<http://jquery.com/>

⁹<http://handlebarsjs.com/>

¹⁰<http://www.modernizr.com/>

¹¹<http://html5boilerplate.com/mobile>

¹²<http://kpuputti.github.com/JSONCache/demo/index.html>

simulating a flaky network according to the configuration.

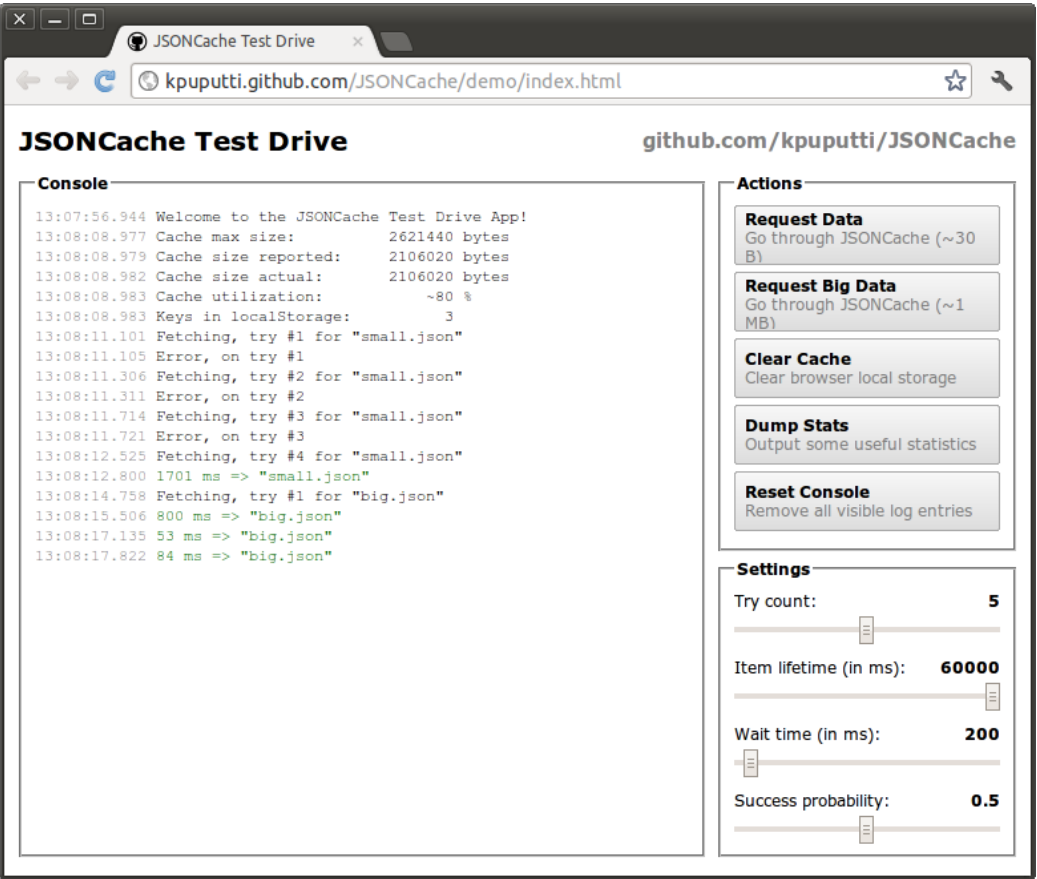


Figure 3.1: Interactive JSONCache demo.

Chapter 4

Results: What Was Good and Where Were the Compromises

4.1 Targeting Different Platforms

4.1.1 Device Detection

4.1.2 Feature Detection

4.2 Targeting Different Screens

4.3 Handling Mobile Networks

4.3.1 Minimizing Data Transfer

4.3.2 Caching

4.3.3 Preloading

4.3.4 Offline Support

4.3.5 Handling Interruptions

4.4 Graphics and Animations

4.5 Performance Analysis

4.5.1 YSlow

4.5.2 PageSpeed

Chapter 5

Discussion: Bright Future Ahead for HTML5

Chapter 6

L^AT_EXtest

6.1 Citing

- Berners-Lee [1]
- Mikkonen & Taivalsaari [4]
- Taivalsaari & Mikkonen [7]
- Pilgrim [5]
- Crockford [2]
- Souders [6]
- Garrett [3]
- Zakas [8]

Bibliography

- [1] BERNERS-LEE, T. Long live the web. *Scientific American* 303, 6 (2010), 80–85.
- [2] CROCKFORD, D. *JavaScript: The Good Parts*. O'Reilly Media / Yahoo Press, 2008.
- [3] GARRETT, J. J. Ajax: A new approach to web applications. *Adaptive path* 18 (2005). Available at: <http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications>. Accessed 5-January-2012.
- [4] MIKKONEN, T., AND TAIVALSAARI, A. Apps vs. Open Web: The Battle of the Decade. In *2nd Annual Workshop on Software Engineering for Mobile Application Development* (2011).
- [5] PILGRIM, M. *HTML5: Up And Running*. O'Reilly Media, 2010.
- [6] SOUDERS, S. *High Performance Web Sites*. O'Reilly Media, 2007.
- [7] TAIVALSAARI, A., AND MIKKONEN, T. The Web as an Application Platform: The Saga Continues. In *Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on* (2011), IEEE, pp. 170–174.
- [8] ZAKAS, N. C. *High Performance JavaScript*. O'Reilly Media / Yahoo Press, 2010.