

Snippets code from my daily experience



June 10, 2010

A survey on Safari 5, Chrome and Firefox extensions API

Filed under: [chrome](#), [firefox](#), [jetpack](#), [safari](#) — dafi @ 2:28 pm

A couple of hours after [Safari 5](#) publication I've received an email asking me to port Table2Clipboard to Safari, after two days I've received five emails asking to port on Safari both [Table2Clipboard](#) and [ViewSourceWith](#).

The same happens with Google [Chrome](#), people asks to migrate extensions to their new favorites browsers but this is in many cases very difficult or impossible.

Giorgio was very clear on his post "[Before you ask... \(No NoScript on Safari\)](#)" and NoScript is infinitely more important and popular than my bonsai extensions.

Now I want to summarize what it is possible to do on Chrome and Safari and what isn't possible to do *from an extension developer point of view*.

Keep in mind that from an extension developer perspective **Firefox is a platform**. Safari and Chrome are applications with minimal support for pluggable code.

Minimal support doesn't mean you can't create great extensions but it means you are very limited in **advanced** topics like web page listeners, clipboard, file and network system access and many other programming areas.

[Jetpack](#) from my point of view is closer to [Mozilla Platform](#) than WebKit/Chrome extensions framework so for simplicity when I speak about Firefox I include Jetpack (please apologize me for this simplistic and superficial classification)

Below you find a table with extensions capabilities supported by browsers at time I wrote this post.

It comes from my needs to implements extensions on Chrome and Safari, it is incomplete and anybody is welcomed to help me to complete or correct it.

| | | |
|---|---------|-------------------|
| | Firefox | Yes |
| Content DOM Modification (eg injection) | Chrome | Yes |
| | Safari | Yes |
| Accessing to web pages to modify or simply traverse the DOM is the main browser activity and it is fully accessible from extensions Chrome and Safari introduce the concept of JS (and CSS) injection to separate access to content DOM from extension DOM, Firefox is like the wild fast west you can do (almost) anything | | |
| | Firefox | Yes |
| Tabs and Windows Access | Chrome | Yes |
| | Safari | Yes |
| Enumerating, inserting and closing windows and tabs are common tasks, both Safari 5 and Chrome have very well designed APIs. Firefox API is too low level (XPCOM from JS) but luckily FUEL simplifies all these operations, Jetpack has a very handy API. On Safari I was unable to find APIs to listen tabs creation and destruction, Chrome has very well documented APIs to do that. | | |
| | Firefox | Yes |
| Tabs Customization | Chrome | No |
| | Safari | No |
| The UI widget tab is available only on Firefox thanks to XUL. If you want to colorize tabs or add operations to context menu you simply can't on Chrome and Safari, this is by design. This is a very advanced topic and extensions can live without this feature but new UI ideas must repose in peace for ever. | | |
| | Firefox | Yes |
| Browser UI Widgets Customization | Chrome | Partial by design |
| | Safari | Partial by design |
| Firefox has no constraints, Chrome allows to add icons on address bar (page actions), Chrome and Safari allow to add toolbar buttons, Safari allows to add context menu items, too. At this time the context menu is accessible from APIs in Chrome only from an experimental API | | |
| | Firefox | Yes |
| Clipboard Support | Chrome | Partial |

dafitumblr

Feed dafizilla

8 readers
BY FEEDBURNER

ohloh

OHLOH PROFILE

What I said in the past

Select Month

Blogroll

- Amazon wish list
- Babelzilla
- dafizilla website
- Sannio LUG
- stacktrace.it
- xulit.org

Category Cloud

apache_httpd babelzilla
clipboard css cygwin
extension firefox
flock gecko java jboss
komodo localization
macro morekomodo
mozilla nsIClipboard
nsIStyleSheetService
openkomodo
richscrollbar scintilla
seamoney
sunday_thought
table2clipboard
Uncategorized utf-8
viewsourcewith xbl
xpcom xul

Recent Comments

- on [A survey on Safari 5, Chrome a...](#)
- on [A survey on Safari 5, Chrome a...](#)
- on [A survey on Safari 5, Chrome a...](#)
- on [A survey on Safari 5, Chrome a...](#)
- on [A survey on Safari 5, Chrome a...](#)

Pages

- About

June 2010
M T W T F S S
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30

« Jan

test

- Register
- Log in
- Entries RSS
- Comments RSS
- WordPress.com

Search

| | | |
|---|---------|-----|
| | Safari | No |
| <p>Is clipboard support an advanced topic? I'm not sure.</p> <p>Chrome clipboard support is strange, it only allows to copy full tab content and text selection, the text selection can be copied using the trick <code>document.doCommand('copy')</code> so not a real API exists. Developers can't copy their own strings but only content already present on web pages.</p> <p>Firefox allows to copy strings from code and the advanced usage permits to choose the flavors, for example plain text and HTML code.</p> <p>Table2Clipboard is easy to port on other browsers (95% is DOM content code) but how can I copy to clipboard my own generated strings? Sadly I can't</p> | | |
| | Firefox | Yes |
| File System Access | Chrome | No |
| | Safari | No |
| <p>For security reasons Chrome doesn't allow to open local files (at least from simple APIs), this sounds reasonable but can be very limiting for advanced usage. Chrome (and maybe Safari) can communicate with the file system writing NSAPI Plugins this is more complicated than using a native Javascript API but it sounds consistent with security constraints.</p> <p>I consider ridiculous that Chrome doesn't allow to access to <code>file://</code> pages from extensions</p> | | |
| Run OS local applications | Firefox | Yes |
| | Chrome | No |
| | Safari | No |
| Running local applications from extensions follows the same rules of 'File System Access' and same solutions (ie NSAPI Plugins) | | |
| | Firefox | Yes |
| HTTP low level Access | Chrome | No |
| | Safari | No |
| I think this lack on Chrome/Safari makes almost impossible to implement efficient web pages inspections as done by NoScript | | |
| XMLHttpRequest and Cross-domain | Firefox | Yes |
| | Chrome | Yes |
| | Safari | Yes |
| Any modern web application uses XMLHttpRequest but with Cross-domain Restrictions, from extensions all browsers allow to make Cross-domain calls | | |

Safari is less extensible than Chrome

Safari extension API set at this time is very limited also when compared with Chrome, as mentioned above this doesn't mean all extensions are toys but the user experience can be limited and the developer creativity is seriously damaged.

Treat extensions developers as first class developers

I think extension developers must be able to do same things the application developers can do, obviously security constraints are welcomed but please don't kill developer freedom in name of an opaque concept of security.

Is the browser less or more secure without [AdBlockPlus](#) or [NoScript](#)?


Possibly related posts: (automatically generated)

- 10 Must-have Google Chrome Extensions <http://bit.ly/5Q1W5t>
- Favorite web browser? No idea, honestly
- Keep your bookmarks with you everywhere!
- LastPass extension now securely manages your passwords and forms on Safari 5

Ads by Google



11 Comments

1. The way I see it, Firefox is built on a LOT of JavaScript and  XML technologies (XUL etc) and extensions get access to EVERYTHING, making Firefox extremely extendable with little extra work by the devs, although extensions can now become a security risk once installed (hence the critical and excellent work of filtering such malicious extensions from being posted on AMO). Extensions also must be loaded in the browser process causing longer startup times and making it all too easy for the inexperienced to blame Firefox for an extension's high CPU or memory usage. Extensions can also use native code (FlashGot does, at least indirectly. Some Aero Glass themes seem to do so directly, though I never really tried my hand at Firefox extensions so my knowledge is limited.)


On the other hand Chrome (and I assume Safari) is built on top of native code for the browser window, plus some HTML/JS for some stuff such as the Bookmark Manager, Downloads/History pages, and the Developer Tools. Extensions have only a limited set of APIs they can work with and are sandboxed and strictly controlled. It is a lot easier to see which extension is eating lots of resources and the browser loads up and is usable quickly even if the extensions are taking a while to load. Extensions become available to the user as they load (this is also typically quickly). Extensions can make use of NPAPI plugins to do things in native code outside of the sandbox but such extensions are more thoroughly screened for the Chrome extension gallery.

Both have advantages and drawbacks. Firefox's main edge is its complete customization through extensions. Chrome's API set is small by comparison, but Chrome's sandboxing and process separation help stability and help identify problem extensions.


BTW Chrome has a proper API for Clipboard coming in the chrome.* namespace.

The file:// restriction on extensions is likely to prevent extensions from uploading data from a user's PC without permission, since extensions can pretty much do an XMLHttpRequest to any site they want. This restriction doesn't bother me much since extensions can still access any files in their own extension folder through the chrome-extension:// protocol. There are some restrictions on file:// urls themselves which bother me much more since they break things I need, but that isn't really on topic so I won't go into that.


Comment by Dan — June 10, 2010 @ 3:34 pm

2. Do most users even use that much addons? Or addons that need extended functionalities? There's a tradeoff to everything. 

Comment by John — June 10, 2010 @ 4:33 pm

3. I could be way off, here, but isn't it possible to hack around Chrome's clipboard support to copy arbitrary strings by adding hidden content to the DOM, and then copying that to the clipboard through the available API? Ugly, yes. But I would be surprised if it didn't work. Note that I've never used Chrome, I'm just throwing the idea out there. 

Comment by Jason Oster — June 10, 2010 @ 4:58 pm


AFAIK Chrome needs a text selection to copy but I can be wrong. 

BTW The problem is copy not only plain text but also other flavors

Comment by dafi — June 10, 2010 @ 5:02 pm

4. [...] Real Difference Is The Platform I just read this blog post on extensibility of Safari, Chrome, and Firefox – and it hits a very important point. Chrome is [...]

Pingback by Home of KaiRo: The Real Difference Is The Platform — June 10, 2010 @ 5:38 pm

5. > Is the browser less or more secure without AdblockPlus or NoScript? 

I'd take process sandboxing over those any day of the week. For me, the renderer process not being able to write anywhere on the disk is much better security than something that protects you only from script vulnerabilities and breaks 90% of legit pages (NoScript), or something reactive that isn't really meant to increase protection (AB+).

Comment by HH — June 10, 2010 @ 5:59 pm

6. @HH



Assuming the sandbox has no flaws, that still will not protect you against purely web-based attacks, like XSS theft of information.

NoScript protects against iframe attacks as well.

And of course, I use NoScript just to keep the level of junk online to an on-demand minimum. That includes flash, video and audio.

Comment by nemo — June 10, 2010 @ 7:56 pm

7. Oh, and webgl, where even if the process can't write to disc, it can certainly screw up the GPU, and who knows what privs that driver is running with.



until webgl has whitelists by default, noscript is a necessity with it.

Comment by nemo — June 10, 2010 @ 8:03 pm

8. Answering your question, Firefox is definitely more secure with Adblock Plus and NoScript.



What those morons (aka Google and Apple) are trying to do is minimize the amount of work they have to do in order to deliver extensions. Mozilla goes to great lengths to test and evaluate all extensions submitted to their gallery, and Firefox benefits a lot from that. Chrome's extension gallery is a joke, not only in comparison, but in all actuality too. It. Is. A. Joke.

Comment by Tiago Sá — June 10, 2010 @ 9:42 pm

9. "If you want to colorize tabs or add operations to context menu you simply can't on Chrome and Safari, this is by design."



Although this is not possible in Chrome, its very much possible in Safari 5. See here for more info

:http://developer.apple.com/safari/library/documentation/Tools/Conceptual/SafariExtensionGuide/AddingContextualMenuItems/AddingContextualMenuItems.html#//apple_ref/doc/uid/TP40009977-CH4-SW1

Comment by Pradeek — June 16, 2010 @ 9:50 am

10. still prefer firefox!



Comment by Freelance Website Design — June 17, 2010 @ 12:40 pm

[RSS feed for comments on this post.](#)

[Blog at Wordpress.com](#) .