Embedded scripting:

Graphs, maps, games, and other exciting goodies to stick in your MediaWiki site

Brion Vibber
Wikimedia Foundation

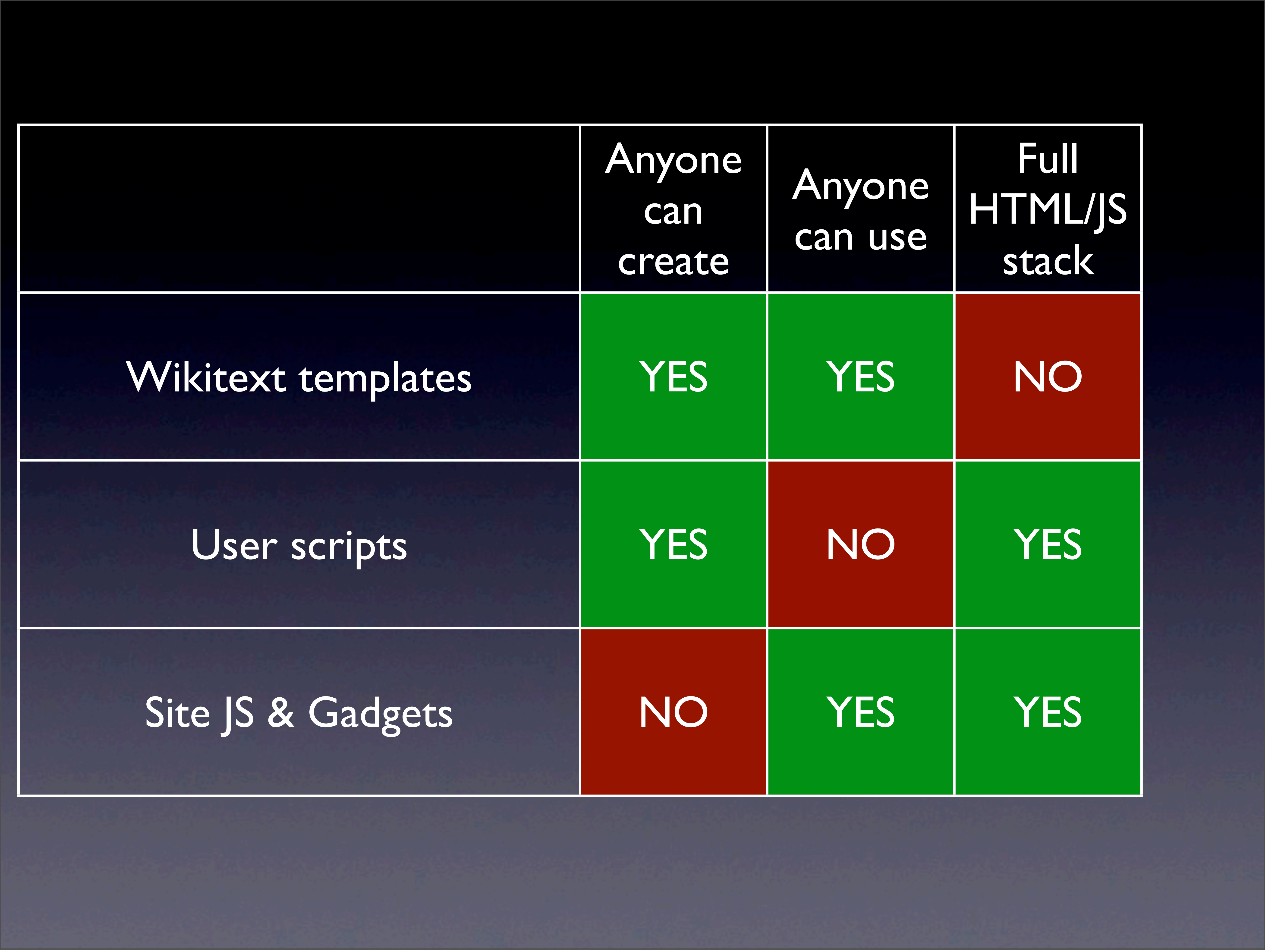
Wikimania 2012
July 13, Washington DC

Our users are creative

Give them tools!

Thursday, July 12, 12

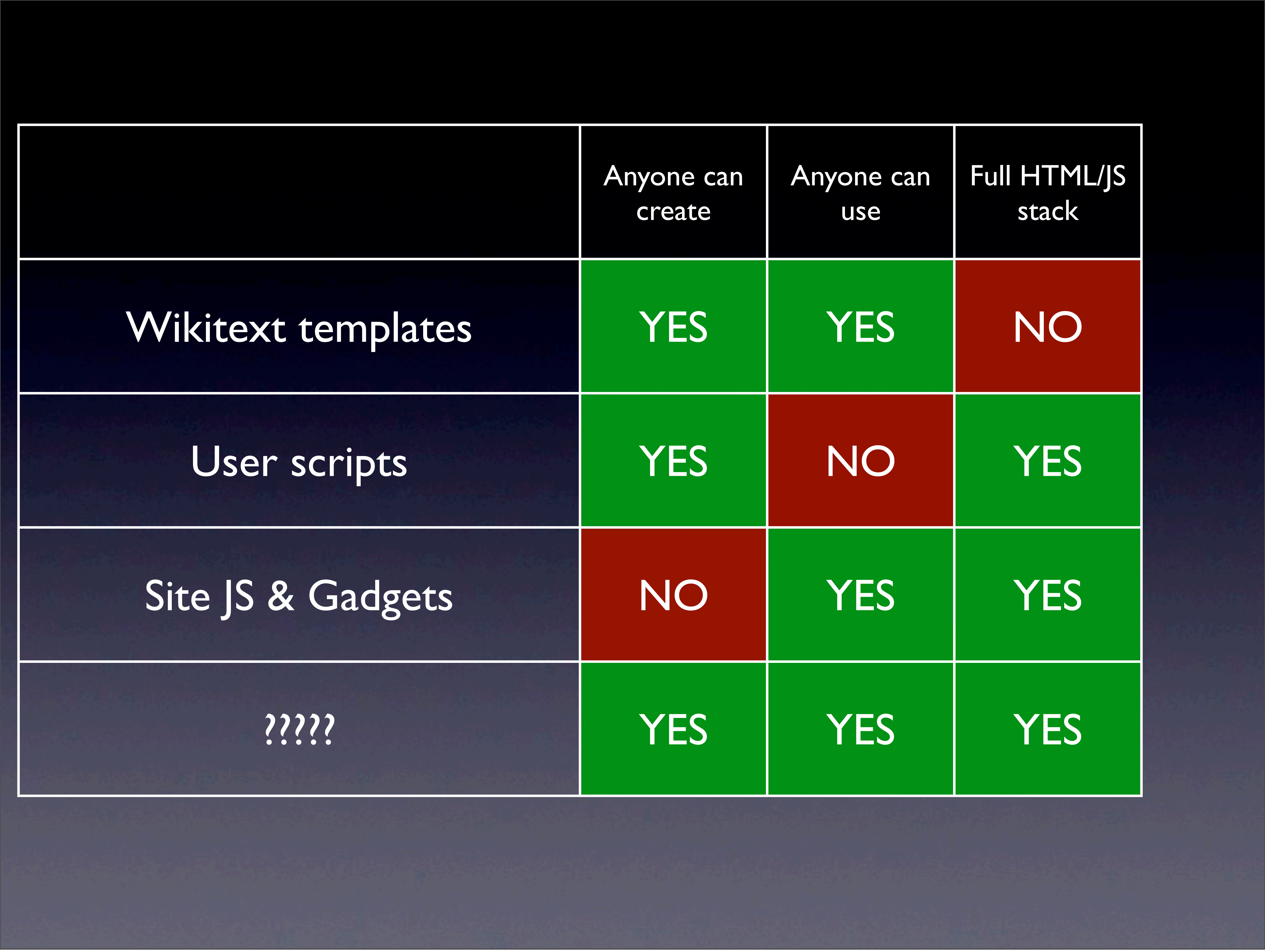
Our users are creative: give them tools and they will make awesome things! Many great features on Wikipedia, Commons, etc have started out as customized site JavaScript, template hacks, or other fun things. Some stay that way, others get transitioned to core or extension features.



	Anyone can create	Anyone can use	Full HTML/JS stack
Wikitext templates	YES	YES	NO
User scripts	YES	NO	YES
Site JS & Gadgets	NO	YES	YES

Thursday, July 12, 12

Wikitext templates are the “native wiki” extension method, but interactivity and HTML support is limited to what MediaWiki and its extensions can provide. Beyond that you have to move to user scripts or site JS -- giving you full access to the HTML/JavaScript stack, but limiting either who can create them or who can use them. Only you see your user scripts, and others have to opt in to share them. Site JS can be used by anyone, but can only be edited by administrators -- both of these limitations are for safety.

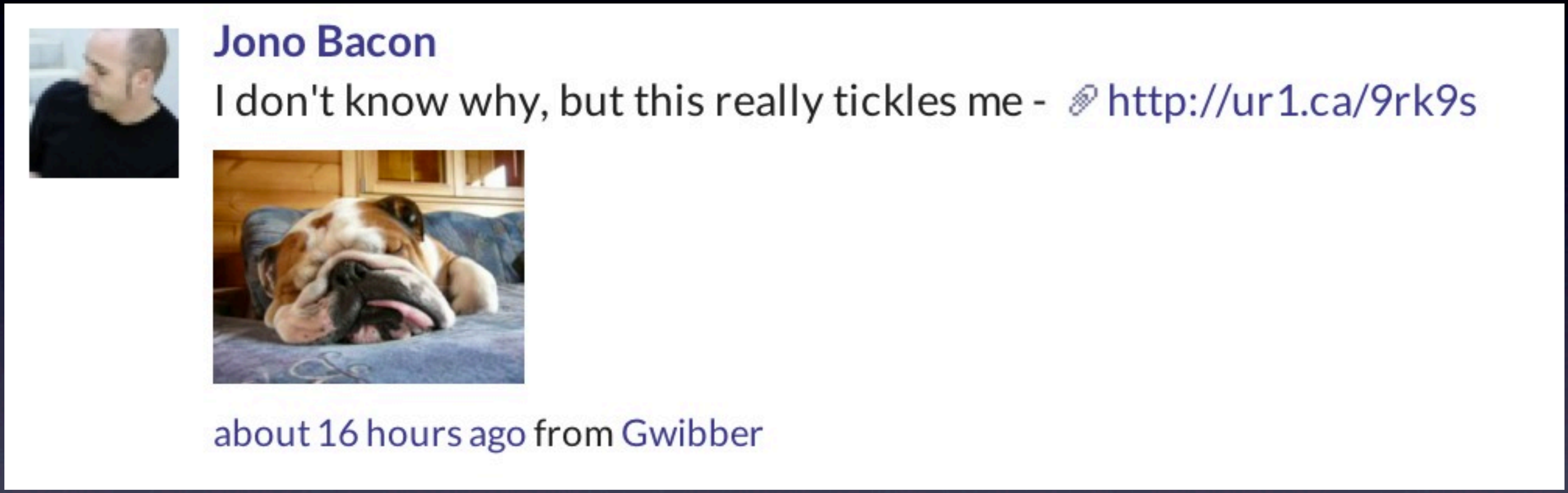


	Anyone can create	Anyone can use	Full HTML/JS stack
Wikitext templates	YES	YES	NO
User scripts	YES	NO	YES
Site JS & Gadgets	NO	YES	YES
?????	YES	YES	YES

Thursday, July 12, 12

So what can we do to let people use the full power of HTML and JavaScript, without putting artificial limitations on who can create or use them? How could we embed arbitrary HTML and JavaScrript into content safely?

oEmbed



Jono Bacon

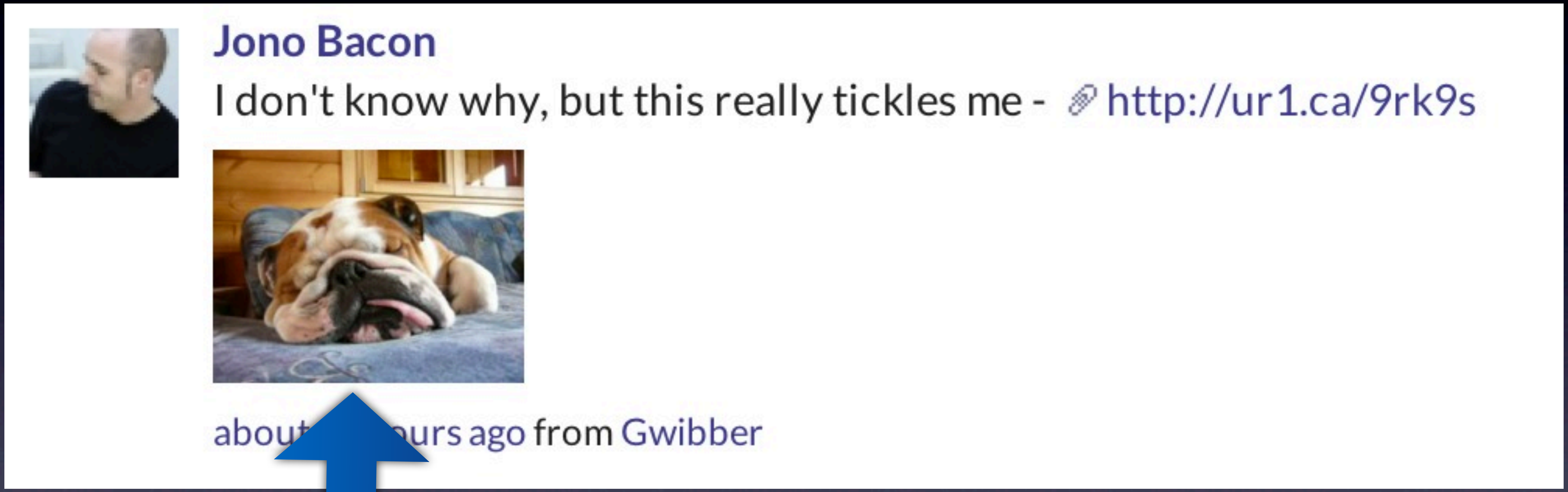
I don't know why, but this really tickles me - http://ur1.ca/9rk9s

about 16 hours ago from Gwibber

Thursday, July 12, 12

I first became interested in embedding when doing work on StatusNet. It uses the oEmbed discovery & query protocol to fetch thumbnail images for Flickr, YouTube, etc links embedded in posts. oEmbed also allows for sending arbitrary HTML to embed videos and such directly, but taking foreign HTML is a security risk.

oEmbed



Jono Bacon

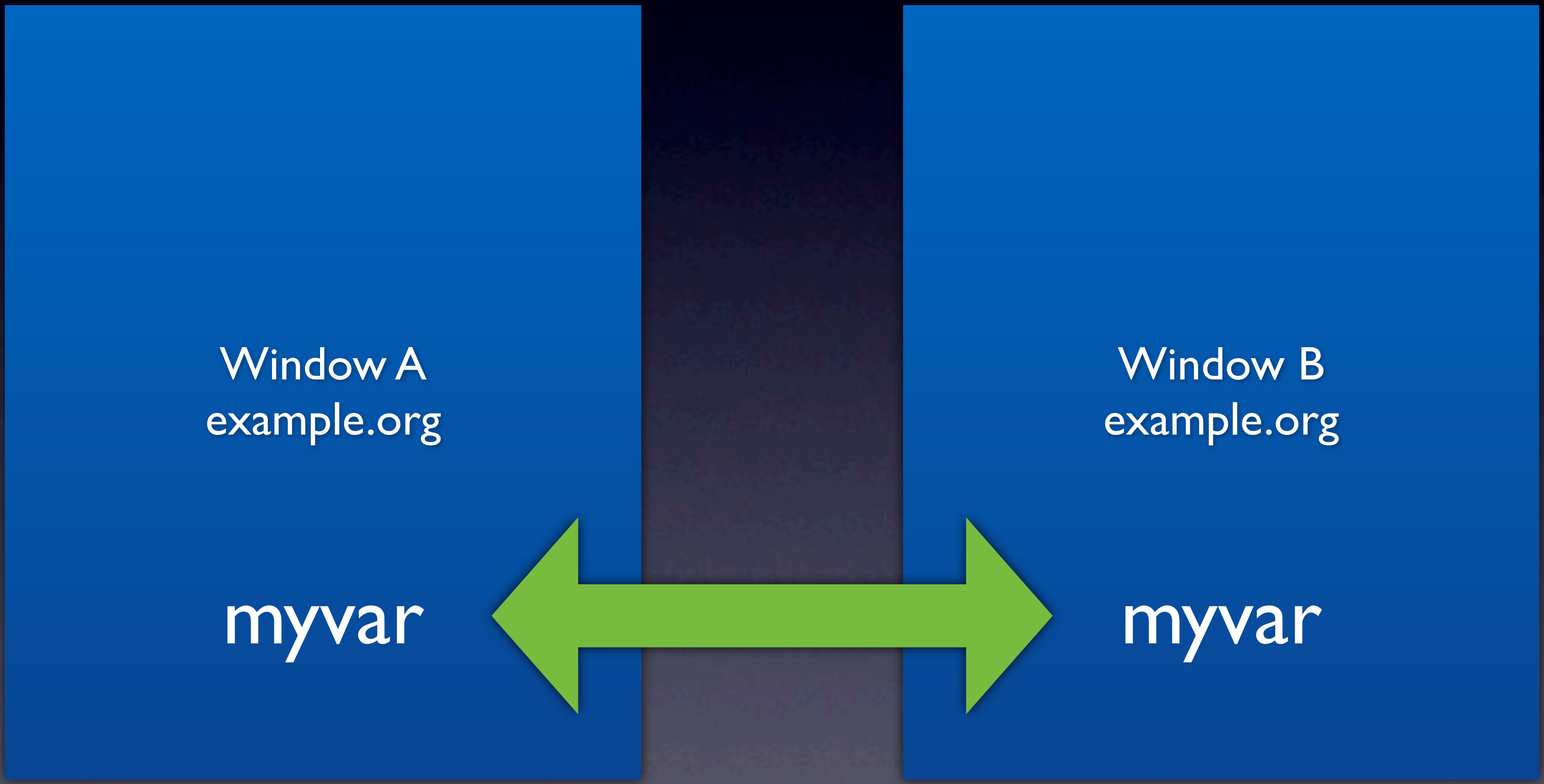
I don't know why, but this really tickles me - http://ur1.ca/9rk9s

abou urs ago from Gwibber

Y U NO VIDEO CLIP??

Thinking embedding would be useful for wikis too, I did some research on how to use things like oEmbed more safely. The spec recommends using an iframe and a separate domain, though this can be tricky to implement for smaller sites.

Same-origin JS



Window A
example.org

myvar

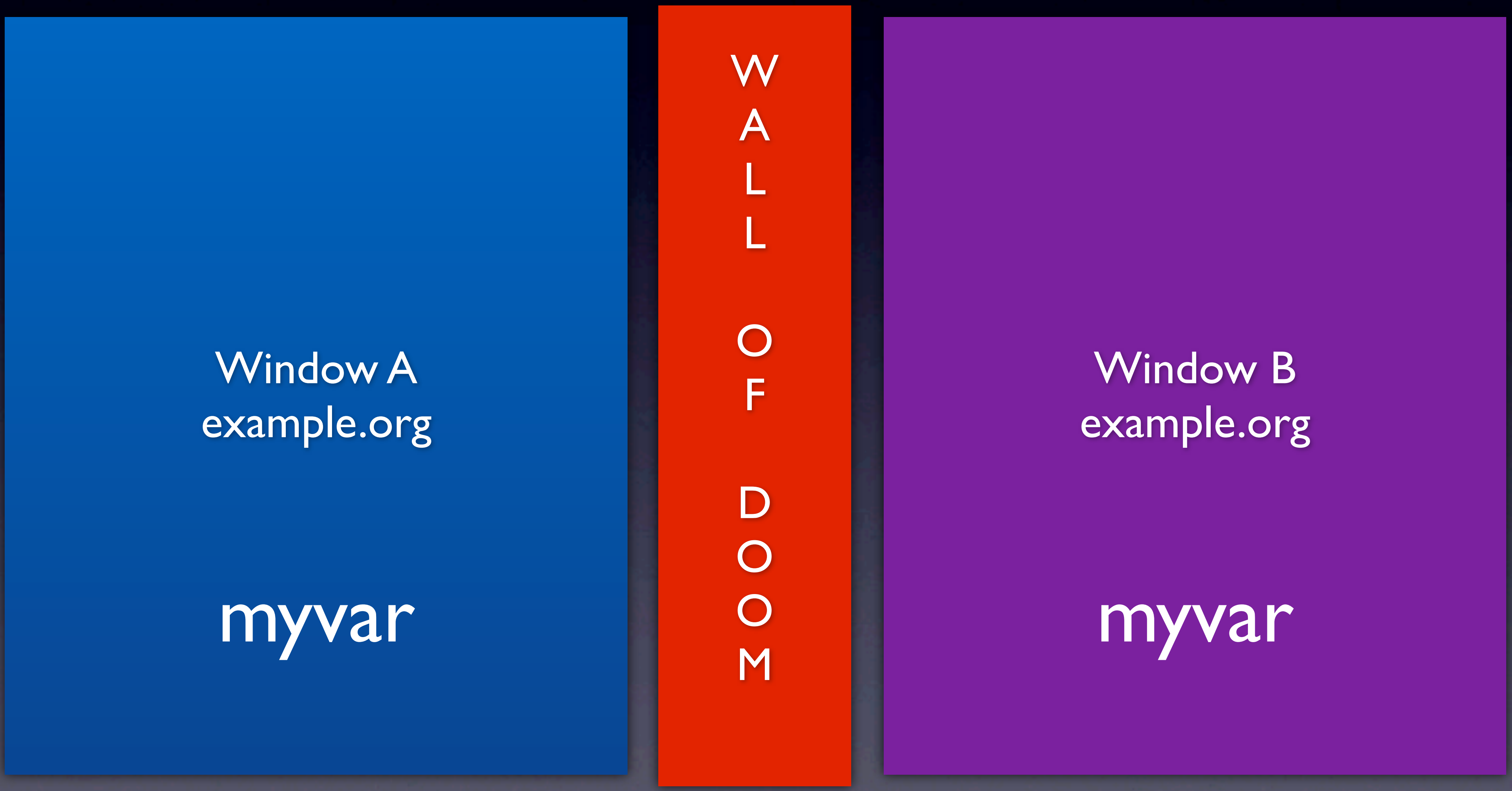
Window B
example.org

myvar

Thursday, July 12, 12

Browsers allow different windows, tabs, or frames that live in the same domain (same-origin rule) to access each others' variables and functions directly. This is convenient, but means that simply opening another page and putting arbitrary JavaScript into it is a big security hole.

Cross-origin JS



Window A
example.org

myvar

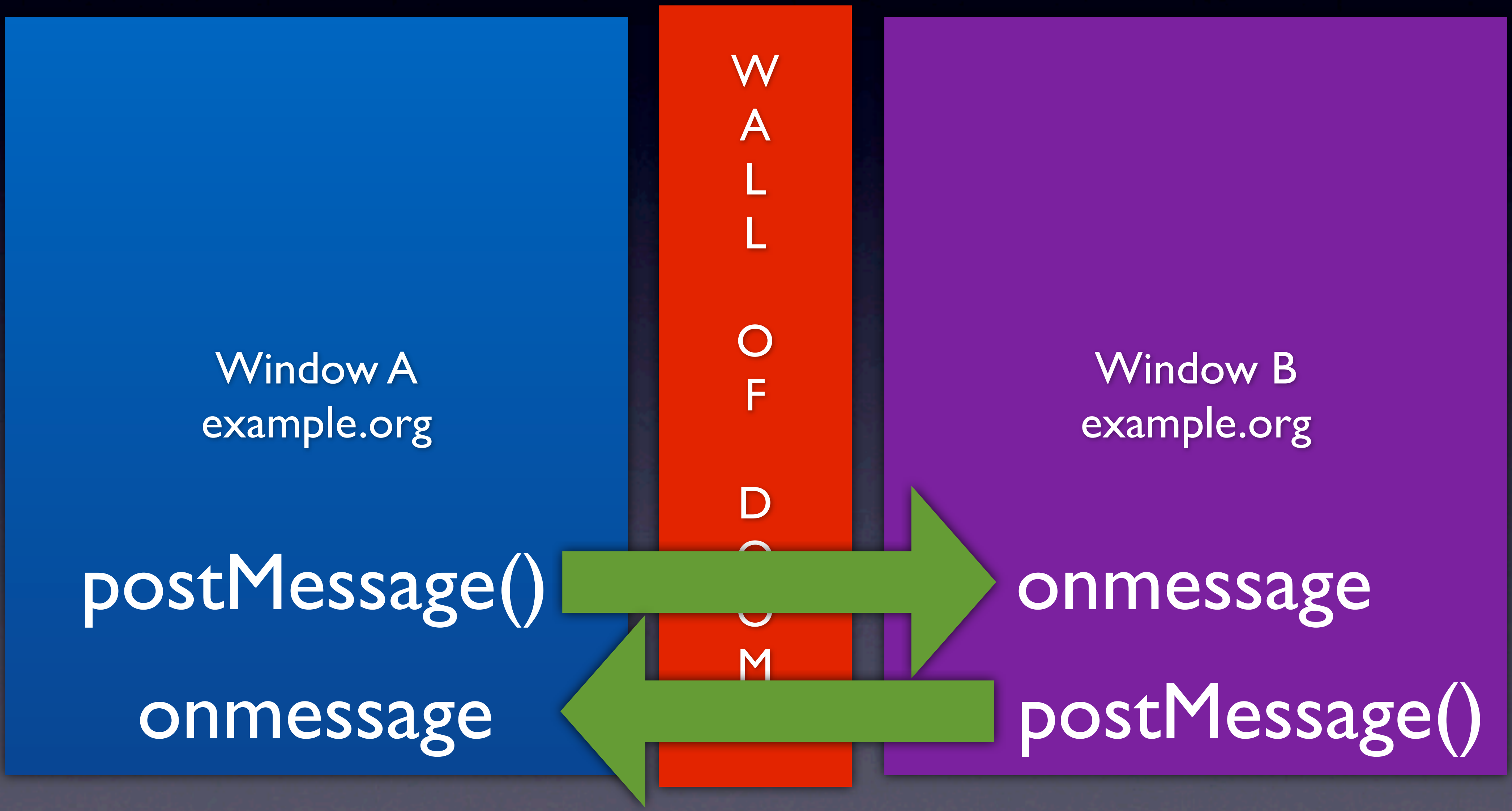
WALL OF DOOM

Window B
example.org

myvar

Browsers enforce JavaScript & DOM security by forbidding direct access between windows running on separate domains. This effectively prevents different sites' JavaScript from interfering with each other, unless you have XSS security holes of course!

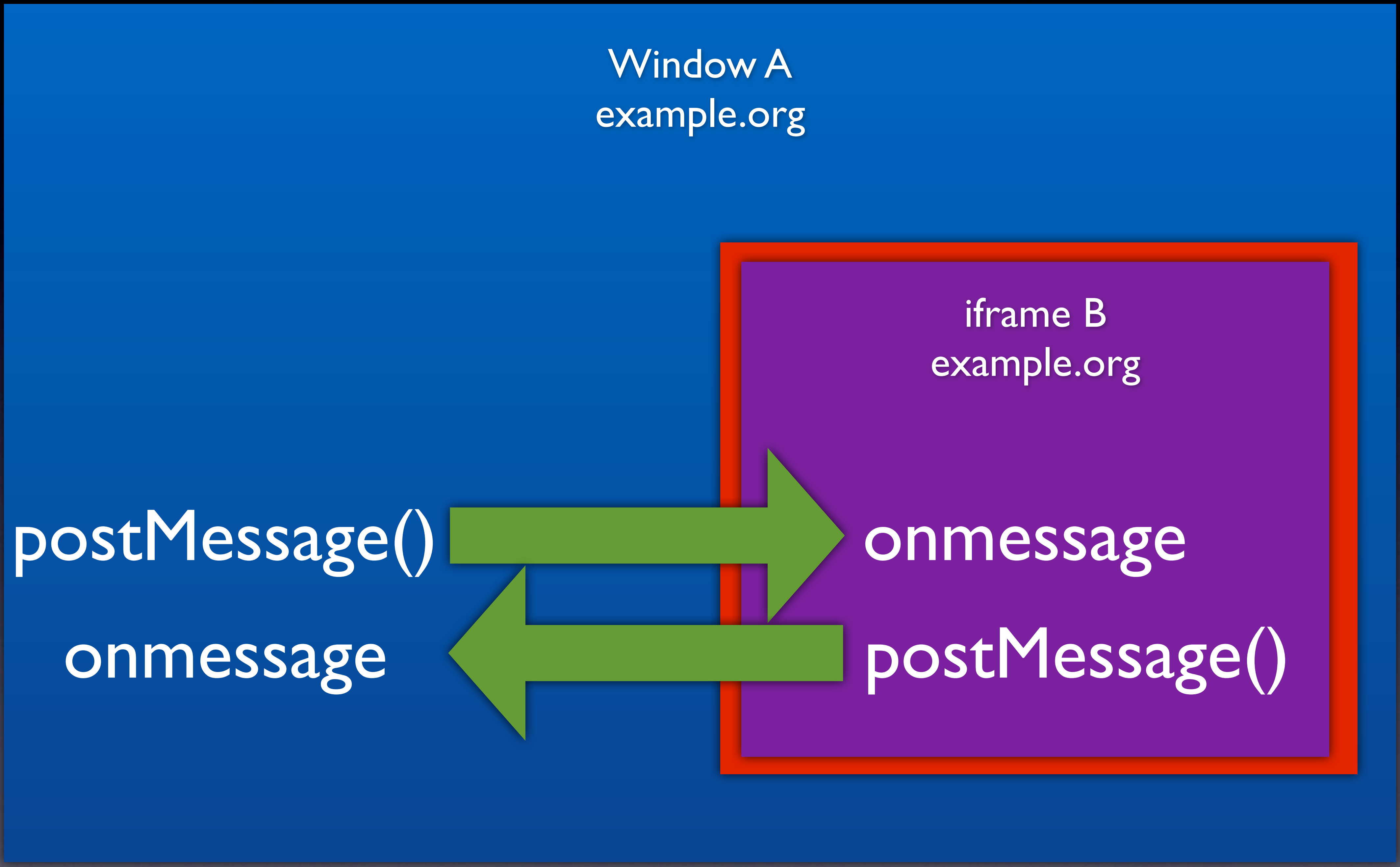
window.postMessage



Thursday, July 12, 12

The window.postMessage() interface is supported by all major browsers, and allows sending strings or JSON objects across domains. Because any cross-domain action is potentially dangerous, you have to opt in to receiving the messages with an event handler, and you have some responsibility for making sure your communications are safe.

iframe!



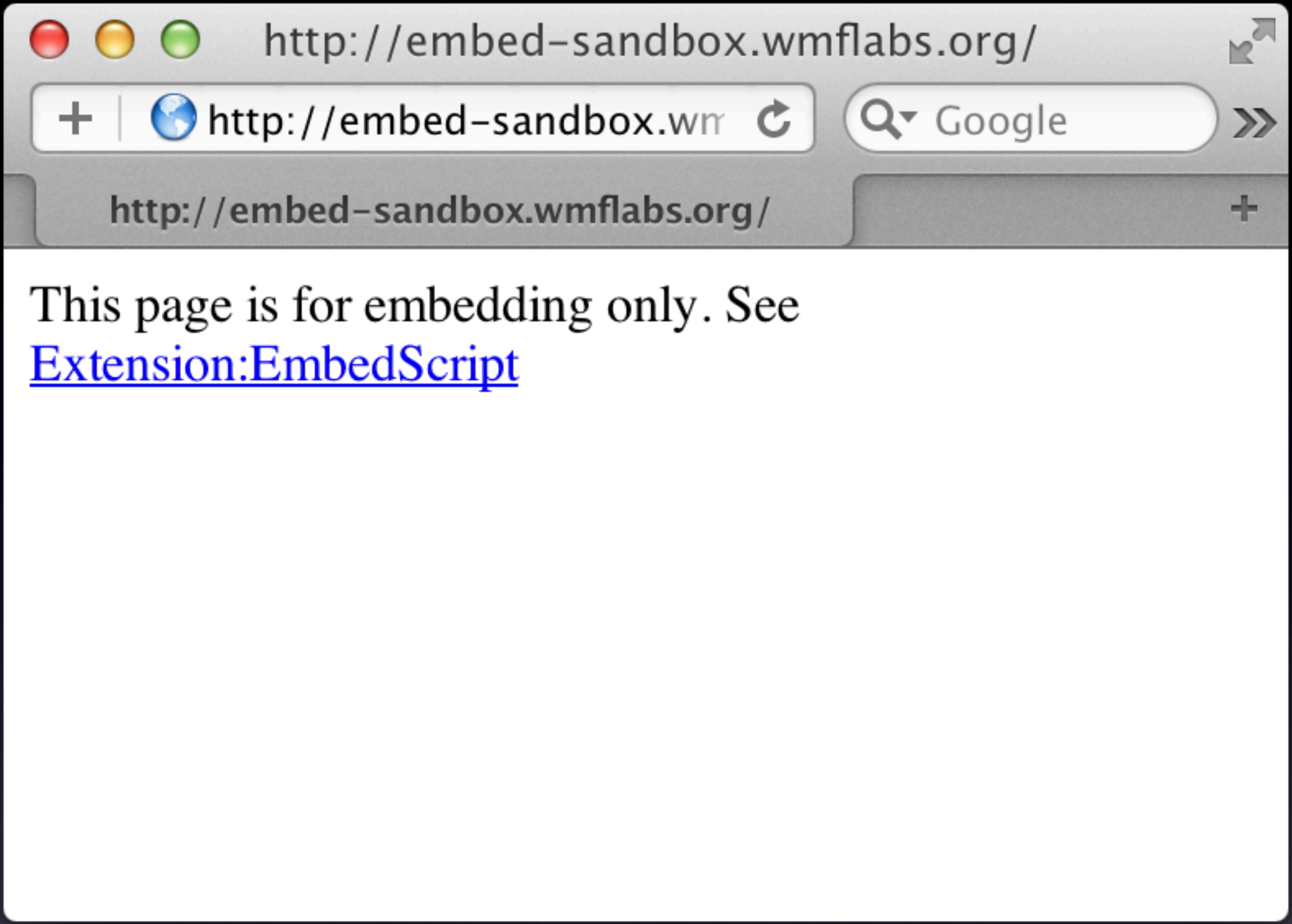
Thursday, July 12, 12

An iframe embedded in another window acts just like two separate windows or tabs. Plus, we get to verify that the parent window won’t change, so it makes our messaging more secure. Any message from the parent can be assumed to be trusted for our purposes.

iframes for sale
old iframes for new

Thursday, July 12, 12

So how do you go about making an iframe on a foreign domain without having to control your own extra servers and extra domains and worry about sticking content onto those extra domains?

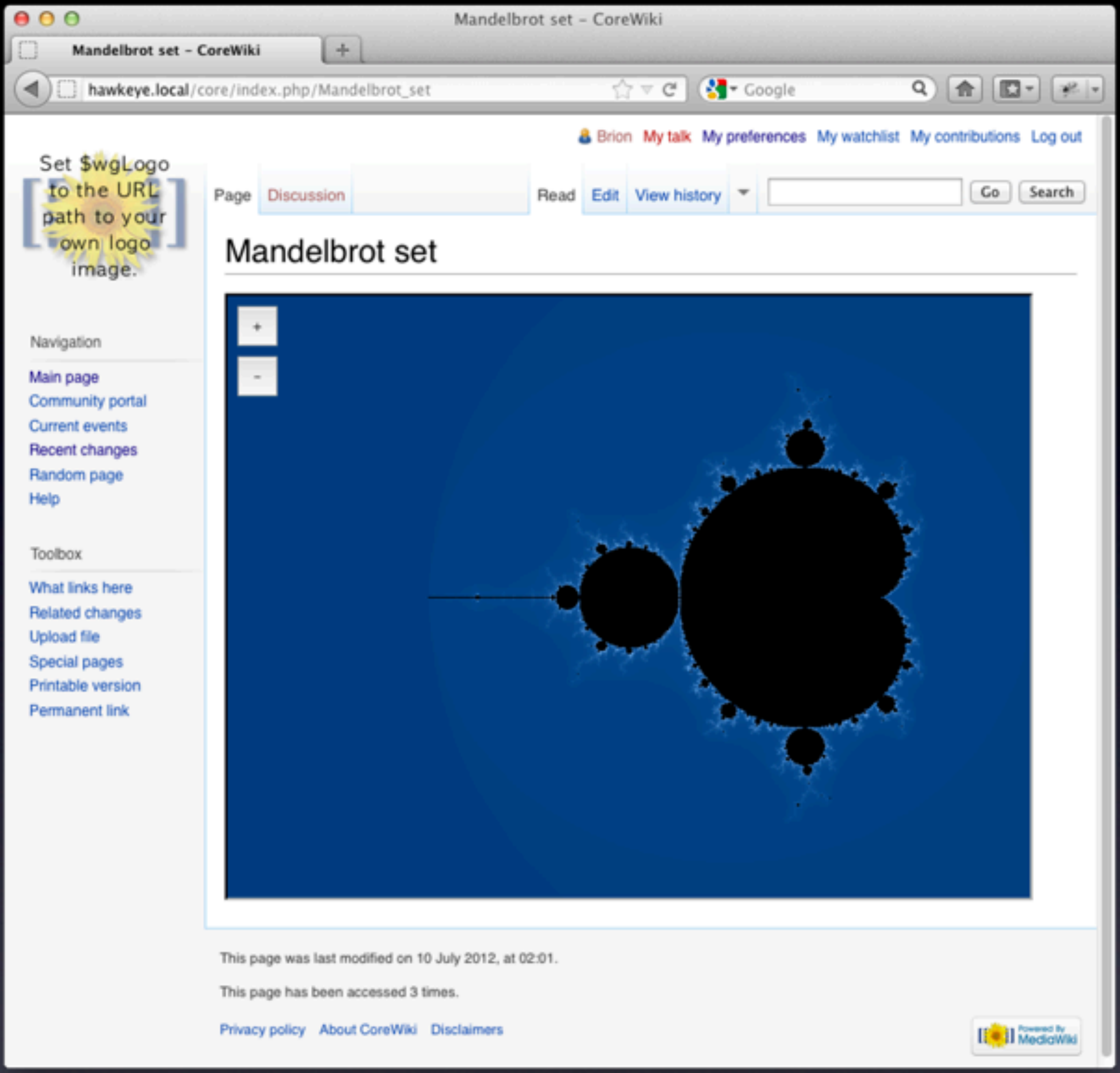


http://embed-sandbox.wmflabs.org/
http://embed-sandbox.wm
Google
http://embed-sandbox.wmflabs.org/

This page is for embedding only. See Extension:EmbedScript

Thursday, July 12, 12

Answer: make ONE domain that lets you inject arbitrary code, but only when used as an iframe. Currently I have an experimental version up at embed-sandbox.wmflabs.org



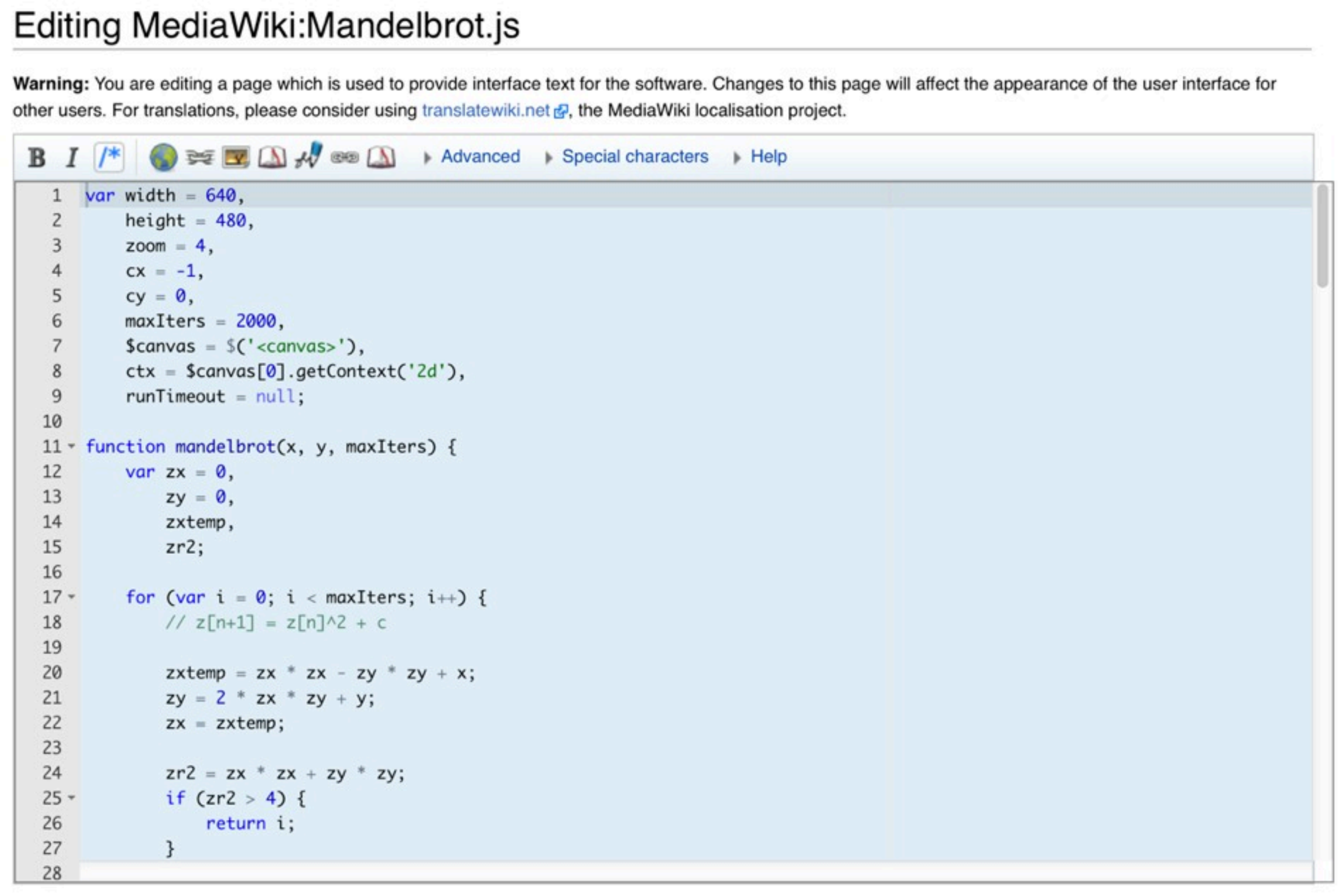
Mandelbrot set

This page was last modified on 10 July 2012, at 02:01.

This page has been accessed 3 times.

Thursday, July 12, 12

I’ve got a simple MediaWiki extension which embeds the iframe, then lets you inject JavaScript code into it to execute.



Editing MediaWiki:Mandelbrot.js

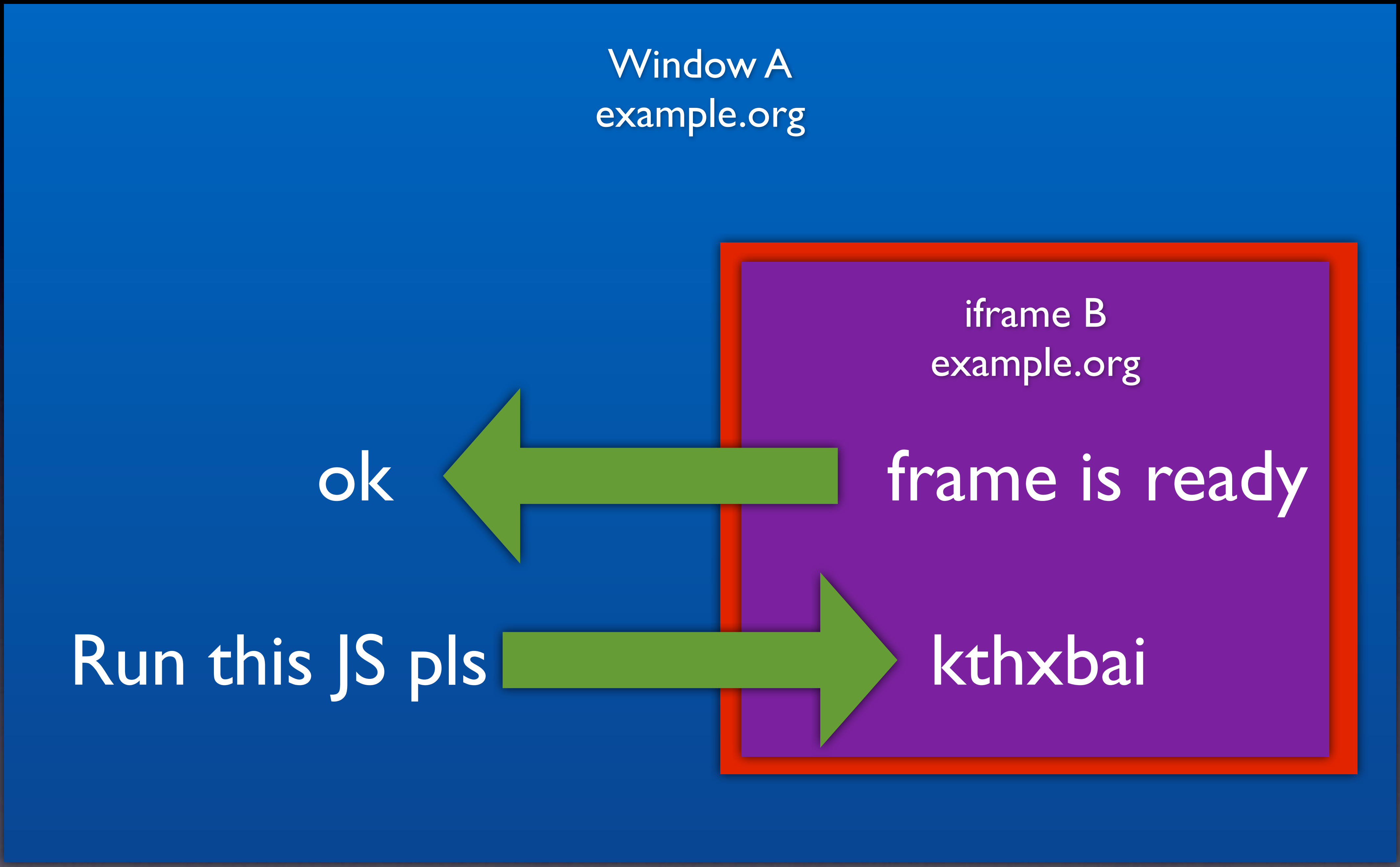
Warning: You are editing a page which is used to provide interface text for the software. Changes to this page will affect the appearance of the user interface for other users. For translations, please consider using translatewiki.net, the MediaWiki localisation project.

```
var width = 640,
    height = 480,
    zoom = 4,
    cx = -1,
    cy = 0,
    maxIters = 2000,
    $canvas = $('<canvas>'),
    ctx = $canvas[0].getContext('2d'),
    runTimeout = null;

function mandelbrot(x, y, maxIters) {
    var zx = 0,
        zy = 0,
        zxtemp,
        zr2;

    for (var i = 0; i < maxIters; i++) {
        // z[n+1] = z[n]^2 + c

        zxtemp = zx * zx - zy * zy + x;
        zy = 2 * zx * zy + y;
        zx = zxtemp;

        zr2 = zx * zx + zy * zy;
        if (zr2 > 4) {
            return i;
        }

```

Thursday, July 12, 12

This example graphs the Mandelbrot set on a <canvas> element.
Note that the scripts don't *have* to be in MediaWiki: namespace, I've just done that for the convenience of triggering the CodeEditor extension.

how it woooorks



Window A
example.org

iframe B
example.org

ok

frame is ready

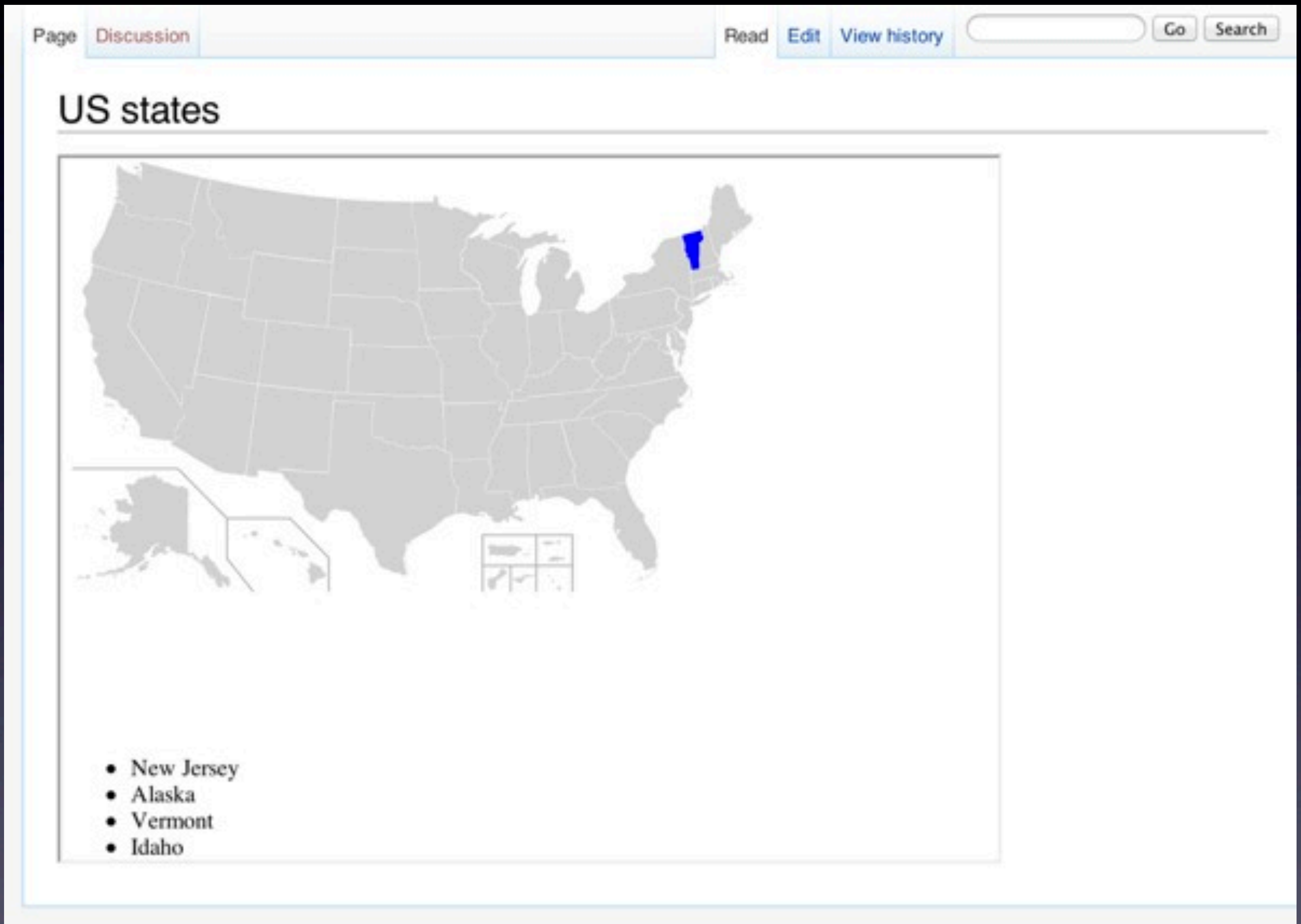
Run this JS pls

kthxbai

Thursday, July 12, 12

The embedding iframe sends a message to the parent window informing it when it’s ready: with jQuery loaded etc. The parent frame knows the message is secure because it owns the iframe, it’s not an arbitrary window. The parent then sends back the JavaScript to execute -- which the child frame knows is safe because it trusts its parent explicitly.

Let's make a game



US states

- New Jersey
- Alaska
- Vermont
- Idaho

Thursday, July 12, 12

-> go to demonstration
-> demonstrate mandelbrot & pythag examples
-> show building US state name guessing game from an SVG file and a little code.

Needs some polish

- Sizing other than 640x480!
- Auto-play vs click-to-play
- Wildcard subdomains for more security
- Debugging tools

Brion Vibber
bvibber@wikimedia.org
brion on freenode.net
@brionv
http://brionv.com