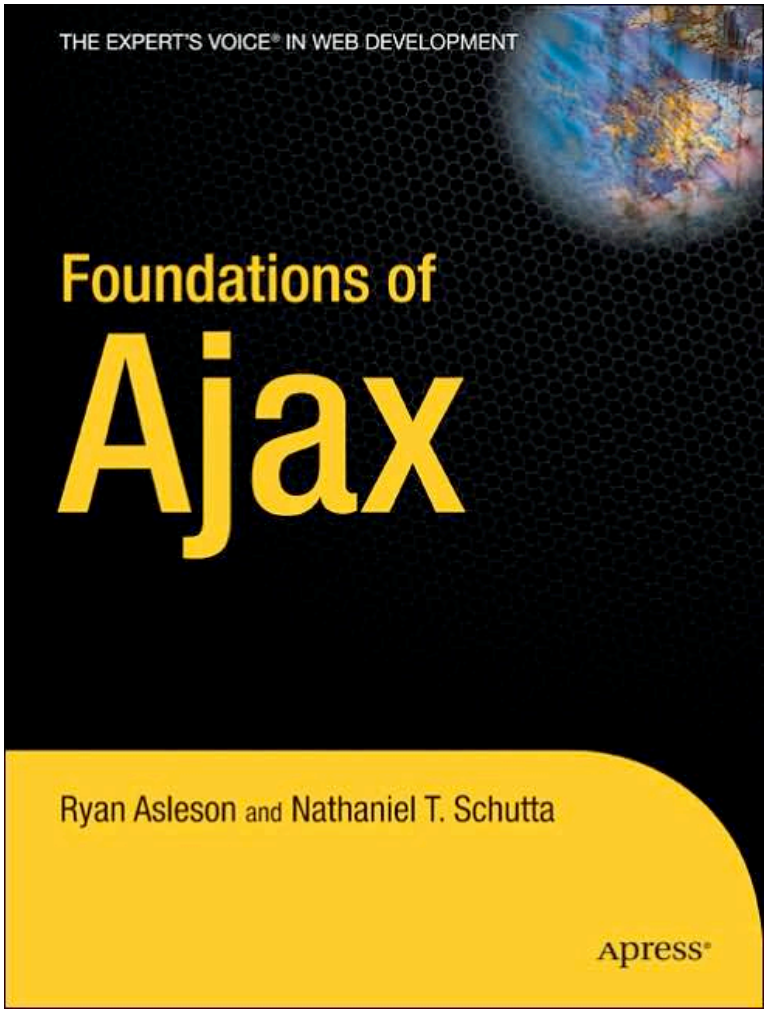
Foundations of Ajax

Ryan Asleson and Nathaniel T. Schutta

Who Are We?



- Ryan Asleson
- Nathaniel T. Schutta
  www.ntschutta.com/jat/
- First Ajax book!

The Plan

- Where have we been?
- Where are we now?
- Where are we going?

How'd we get here?

- It's all about the desktop
  - Very rich applications
  - Upgrades a pain (new hardware anyone?)
- The Web takes center stage
  - Simplified maintenance, low barrier of entry
  - Less functional apps, browser issues
- All about trade offs

Sorry, that's not how it works

- We conditioned users with thick apps
- Then we took that all away
- Convinced our users to accept thin apps
- The browser pushed us towards plain vanilla
- Applets, Flash, XUL/XAML/XAMJ
- Fundamental Issue - Web is based on a synchronous request/response paradigm

What is Ajax?

http://movies.yahoo.com/shop?d=hv&id=1808444810&cf=pg&photoid=521827&intl=us

A cleaner? A Greek hero?
A soccer club?

http://www.v-bal.nl/logos/ajax.jpg

http://www.cleansweepsupply.com/pages/skugroup1068.html

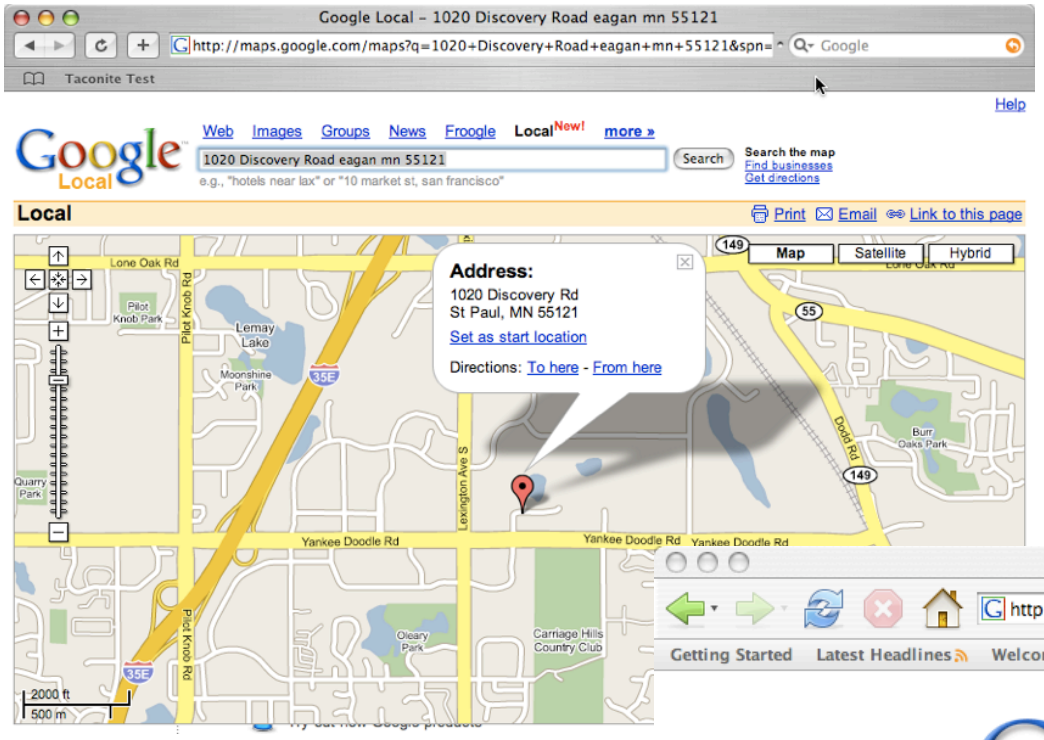
Give me an ‘A’

- Ajax is a catch-phrase - several technologies
- Asynchronous, JavaScript, XML, XHTML, CSS, DOM, **XMLHttpRequest** object
- More of a technique than a specific “thing”
- Communicate with XHR, manipulate the Document Object Model on the browser
- Don’t repaint the entire page!
- We gain flexibility

http://www.adaptivepath.com/publications/essays/archives/000385.php

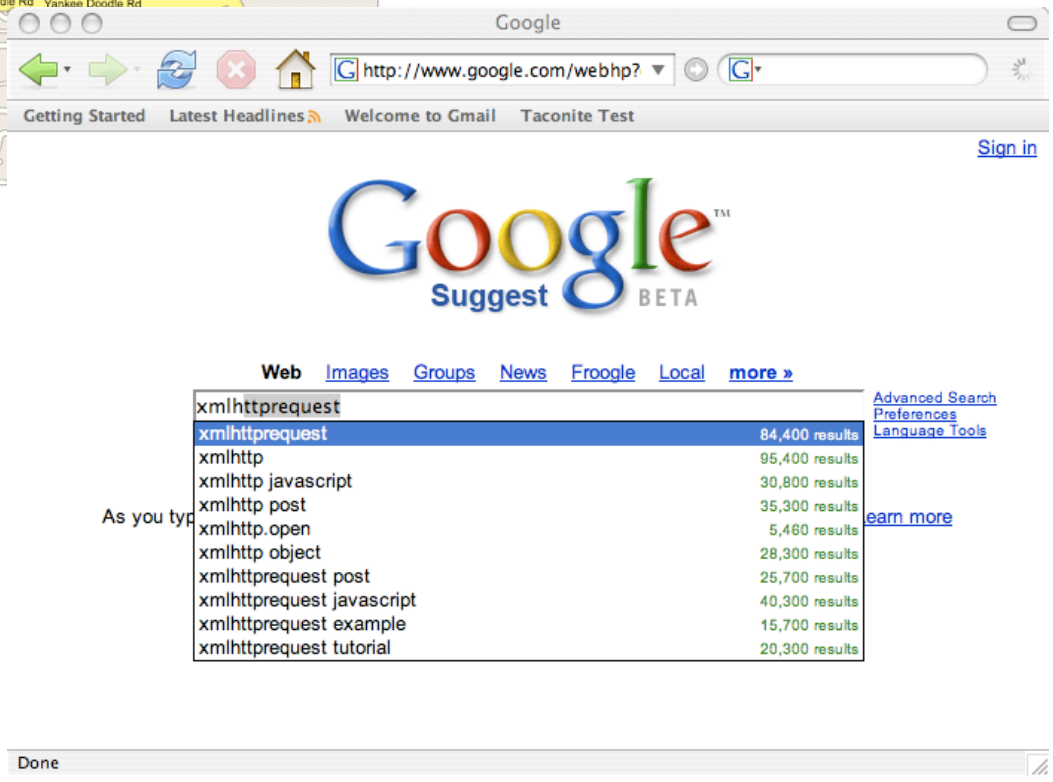
What's old is new again

- XHR was created by Microsoft in IE5
- Of course it only worked in IE
- Early use of JavaScript resulted in pain
- Many developers shunned the language
- XHR was recently adopted by Mozilla 1.0 and Safari 1.2
- And a new generation of web apps was born



Google Maps

Google Suggest



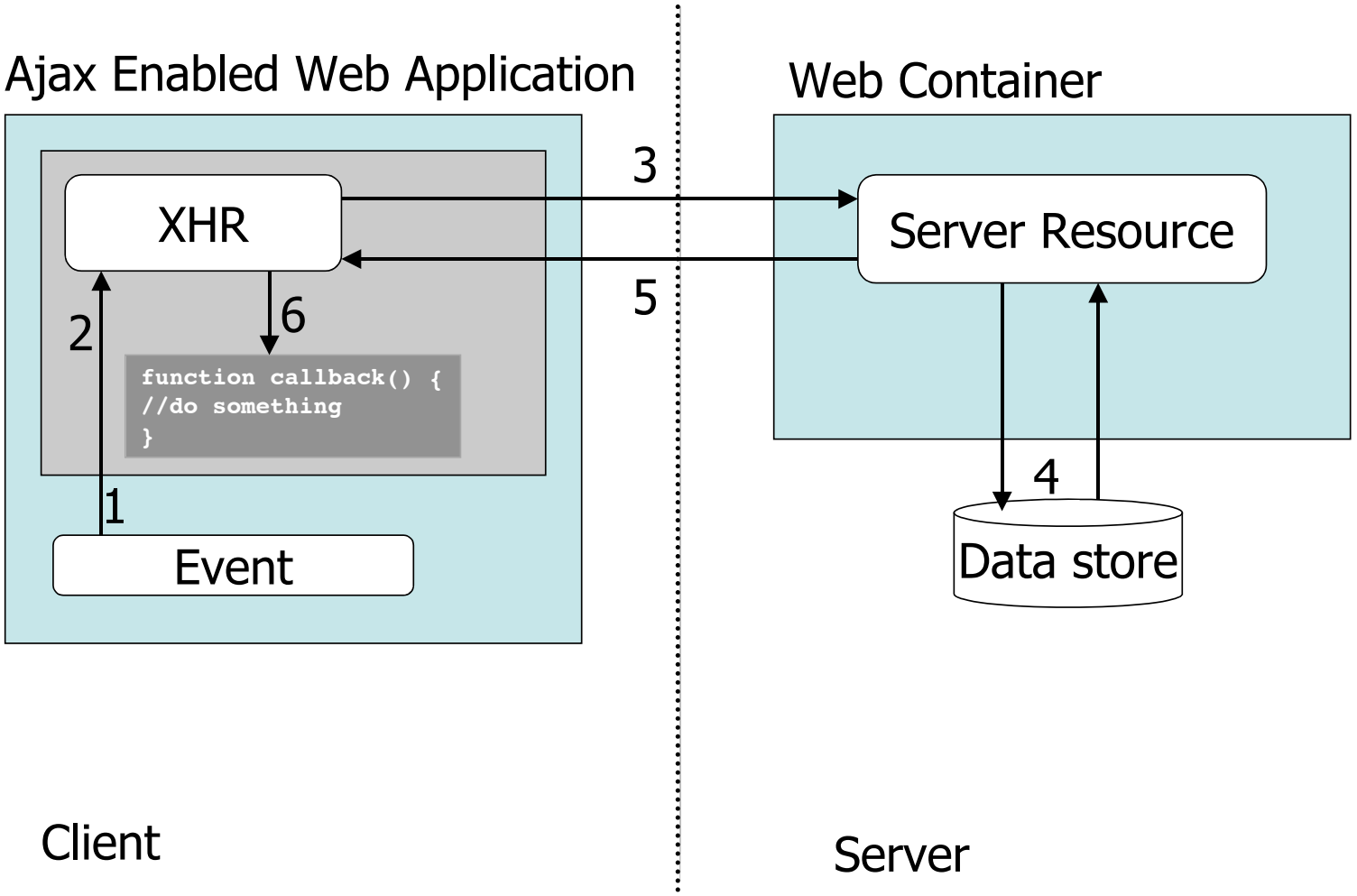
XHR Methods

Method	Description
`open("method", "url" [, asynch [, "username" [, "password"]]])`	Sets the stage for the call - note asynch flag.
`send(content)`	Sends the request to the server.
`abort()`	Stops the request.
`getAllResponseHeaders()`	Returns all the response headers for the HTTP request as key/value pairs.
`getResponseHeader("header")`	Returns the string value of the specified header.
`setRequestHeader("header", "value")`	Sets the specified header to the supplied value.

XHR Properties

Property	Description
`onreadystatechange`	The event handler that fires at every state change.
`readyState`	The state of the request: 0 = uninitialized, 1 = loading, 2 = loaded, 3 = interactive, 4 = complete
`responseText`	The response from the server as a string.
`responseXML`	The response from the server as XML.
`status`	The HTTP status code from the server.
`statusText`	The text version of the HTTP status code.

Typical Interaction



How's this work?

- Start a request in the background
- Callback invokes a JavaScript function - yes prepare yourself for JavaScript
- Can return: a) XML - responseXML, b) HTML - innerHTML c) JavaScript - eval
- Typically results in modifying the DOM
- We are no longer captive to the request/response paradigm!
- But I've done this before...
- IFRAME can accomplish the same concept

Sample Code

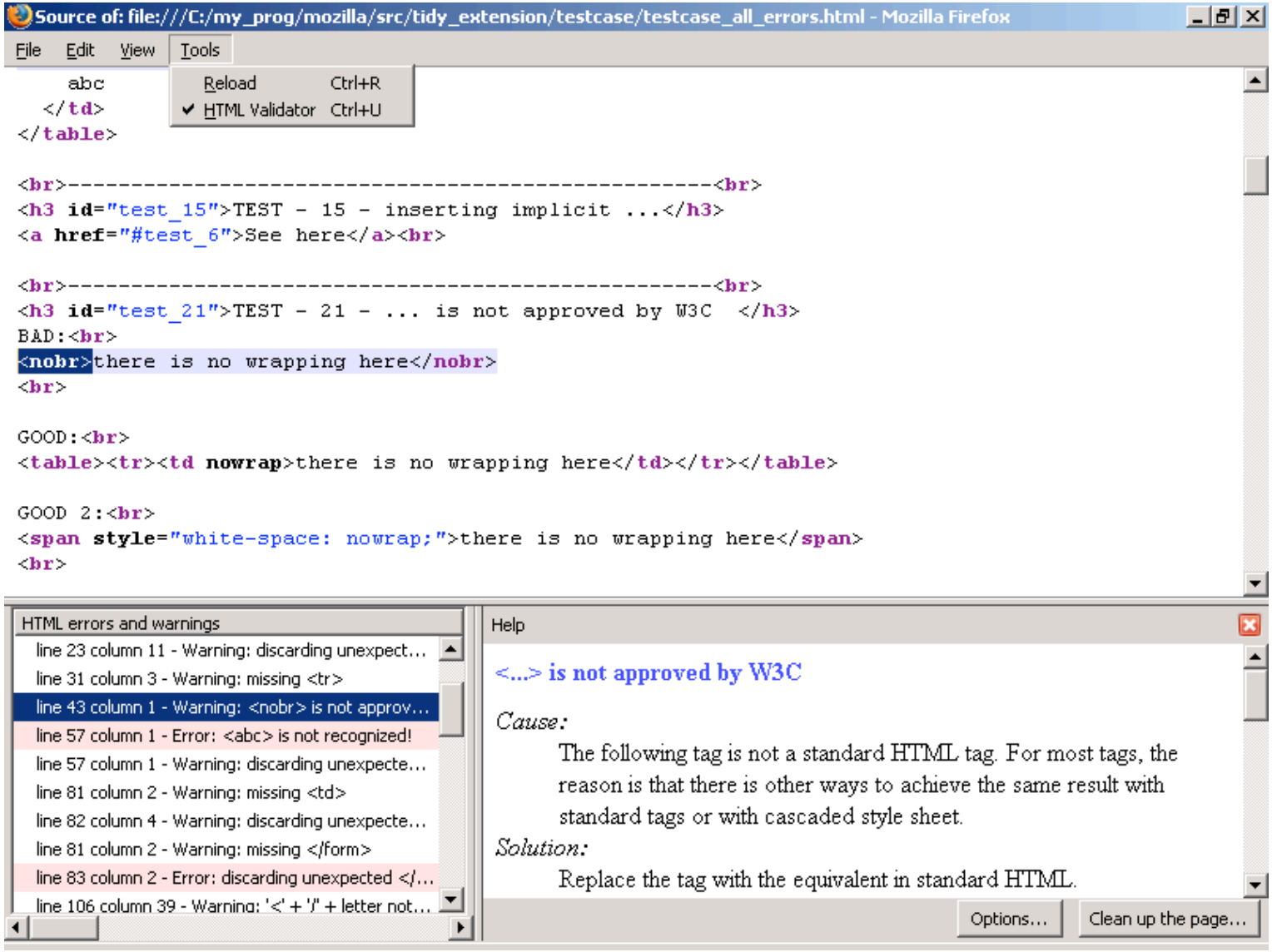
Unfortunately - some browser checking

```
function createXMLHttpRequest() {
    if (window.ActiveXObject) {
        xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
    }
    elseif(window.XMLHttpRequest) {
        xmlHttp = new XMLHttpRequest();
    }
}
function startRequest() {
    createXMLHttpRequest();
    xmlHttp.onreadystatechange = handleStateChange;
    xmlHttp.open("GET", "simpleResponse.xml");
    xmlHttp.send(null);
}
function handleStateChange() {
    if(xmlHttp.readyState == 4) {
        if(xmlHttp.status == 200) {
            alert("The server replied with: " + xmlHttp.responseText);
        }
    }
}
```

Spare me the pain

- Yes, JavaScript can hurt
- Tools are coming, for now check out these
- JSDoc (http://jsdoc.sourceforge.net/)
- Greasemonkey (http://greasemonkey.mozdev.org/)
- Firefox JavaScript Debugger
- Microsoft Script Debugger
- Venkman JavaScript Debugger (http://www.mozilla.org/projects/venkman/)
- Firefox Extensions
  - Web Developer Extension (http://chrispederick.com/work/webdeveloper/)

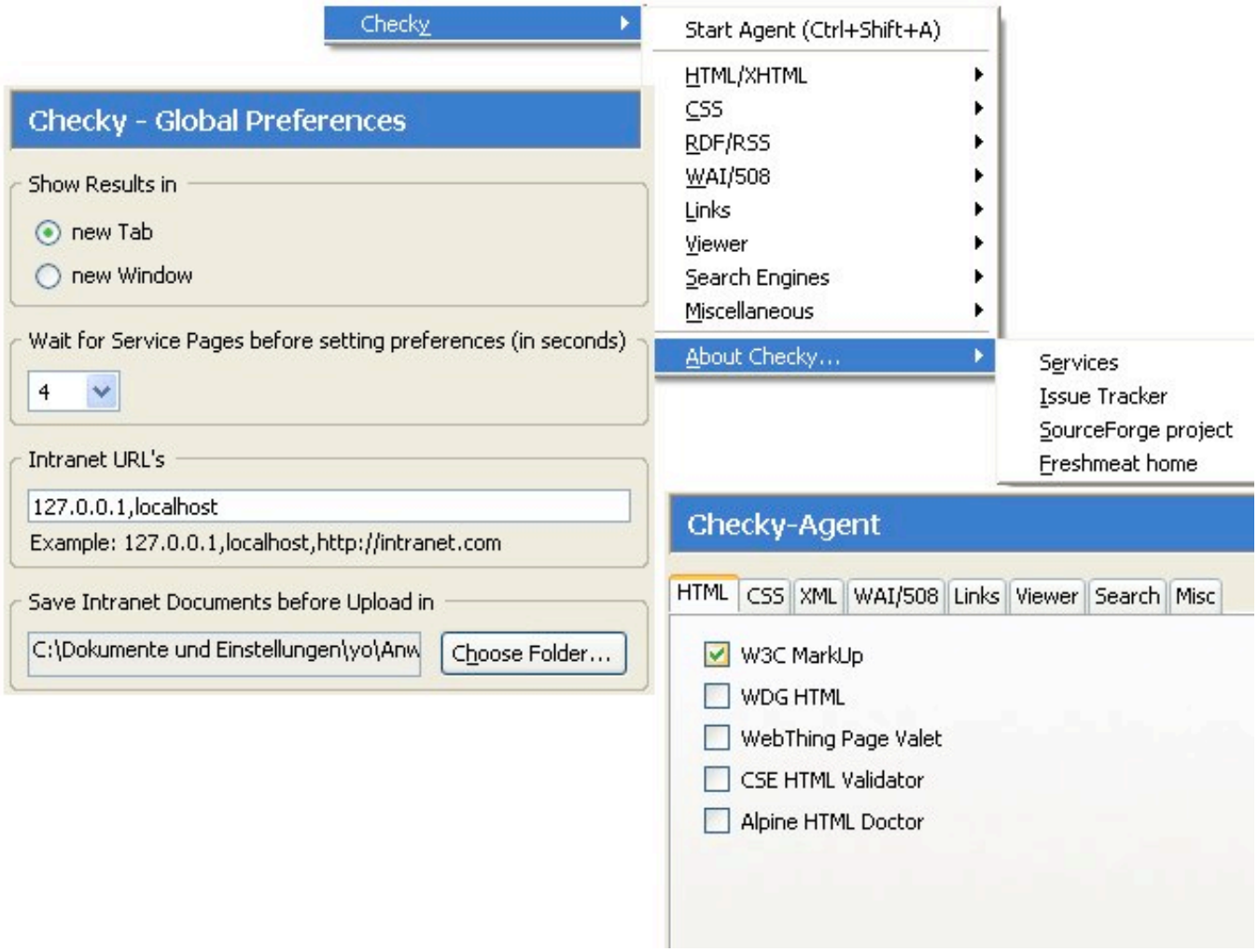
HTML Validator



http://users.skynet.be/mgueury/mozilla/screenshot.html

http://users.skynet.be/mgueury/mozilla/

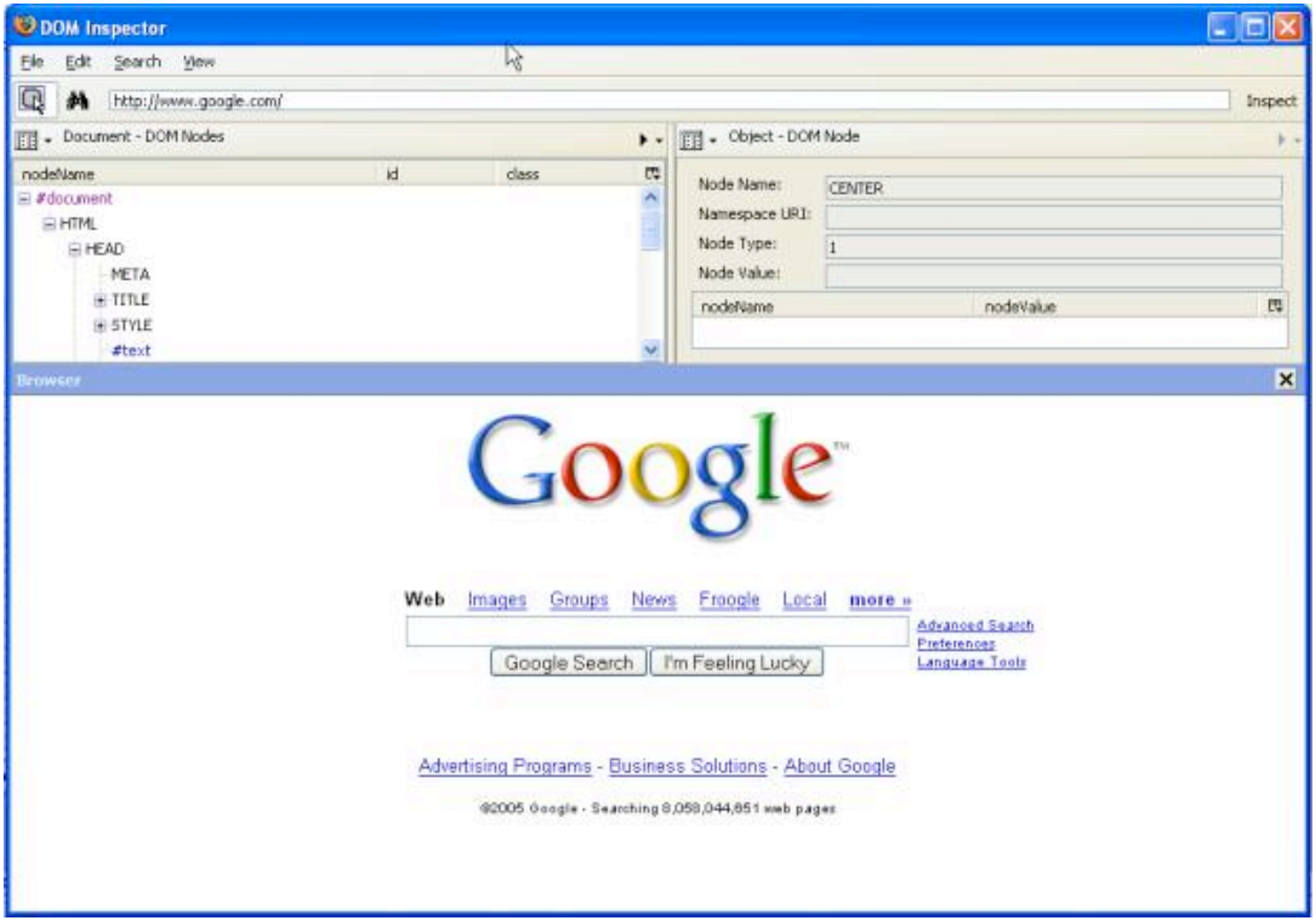
Checky



http://sourceforge.net/project/screenshots.php?group_id=69729

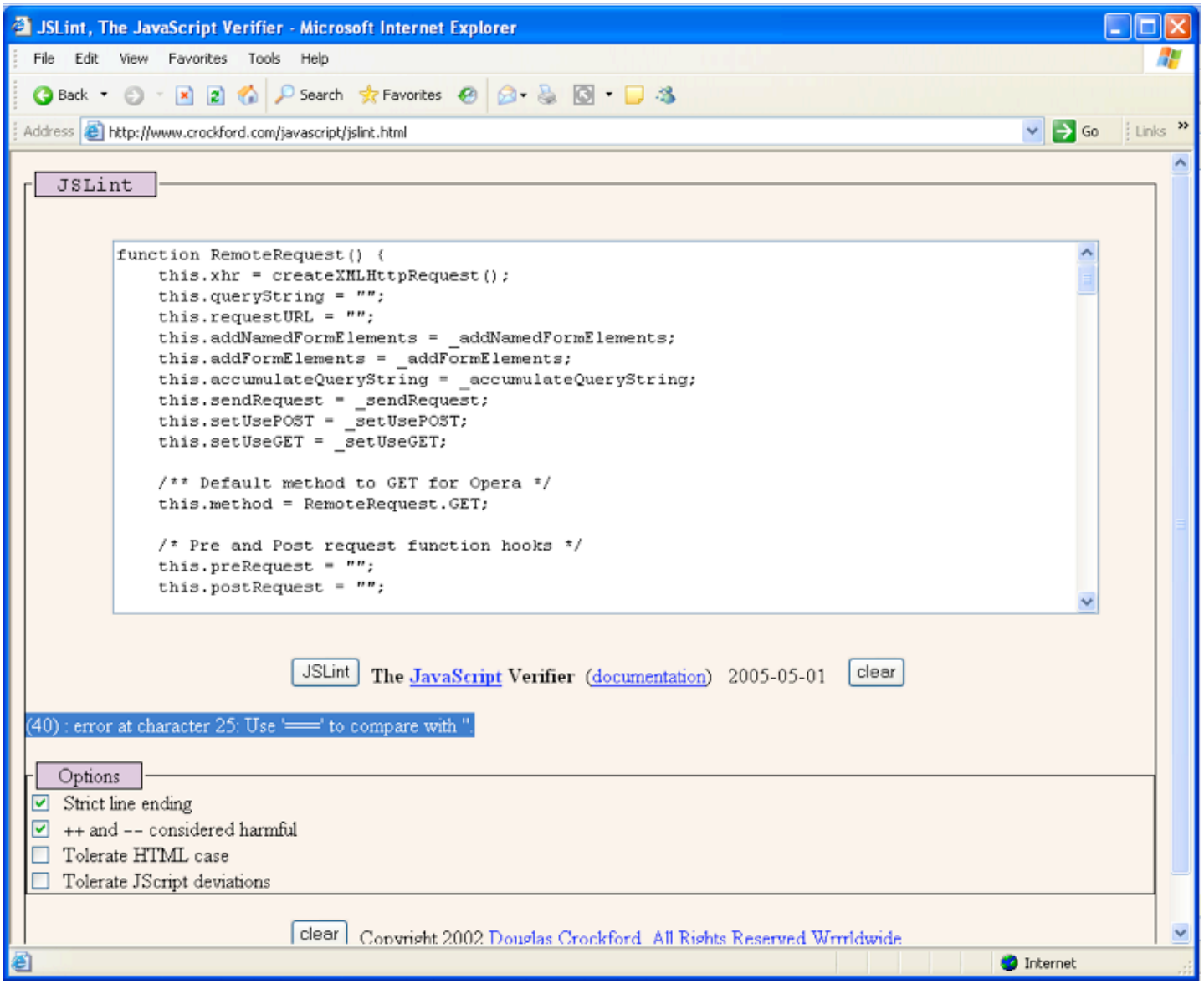
http://checky.sourceforge.net/extension.html

DOM Inspector



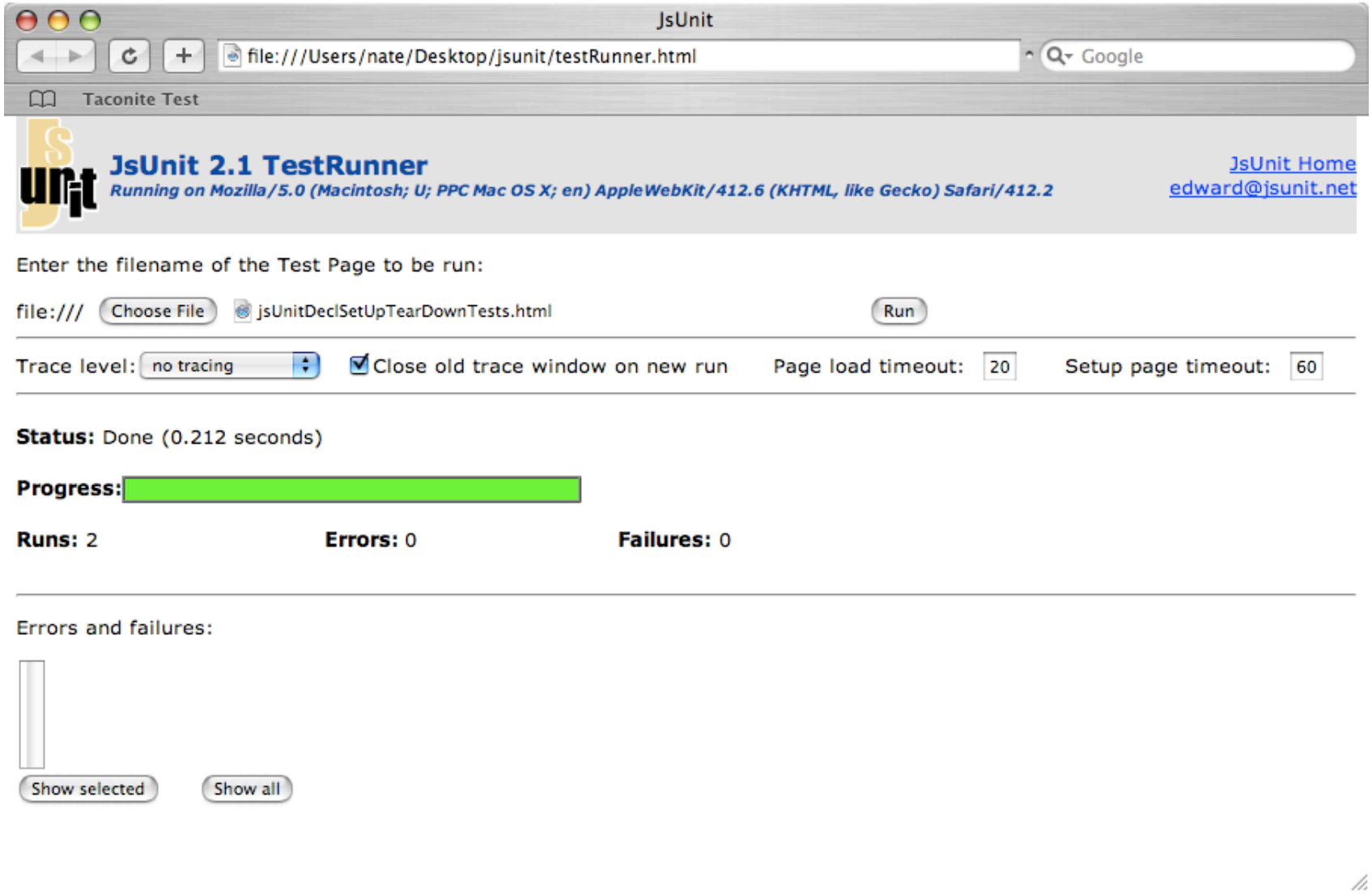
http://www.mozilla.org/projects/inspector/

JSLint



http://www.crockford.com/jslint/lint.html

JsUnit



http://www.edwardh.com/jsunit/

What about libraries?

- There are dozens
- Maturing space
- Most are very new - proceed with caution
- DWR, Dojo, Rico, Microsoft's Atlas, Ruby on Rails, Prototype, etc.
- Taconite

What's next?

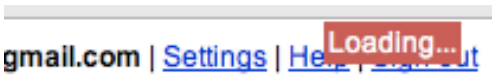
- Better tool support - just a matter of time
- Sun's Creator 2
- Library/toolkit space will consolidate
- User expectation will increase
- More sites will implement

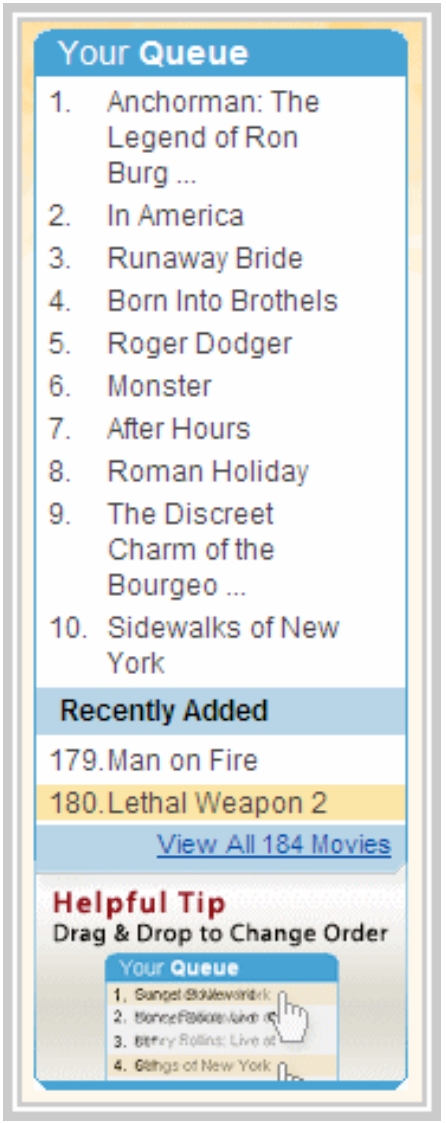
Now what?

- Start small
- Validation is a good first step
- Auto complete
- More dynamic tool tips
- Partial page updates
- Recalculate
- **It's all about the user!**

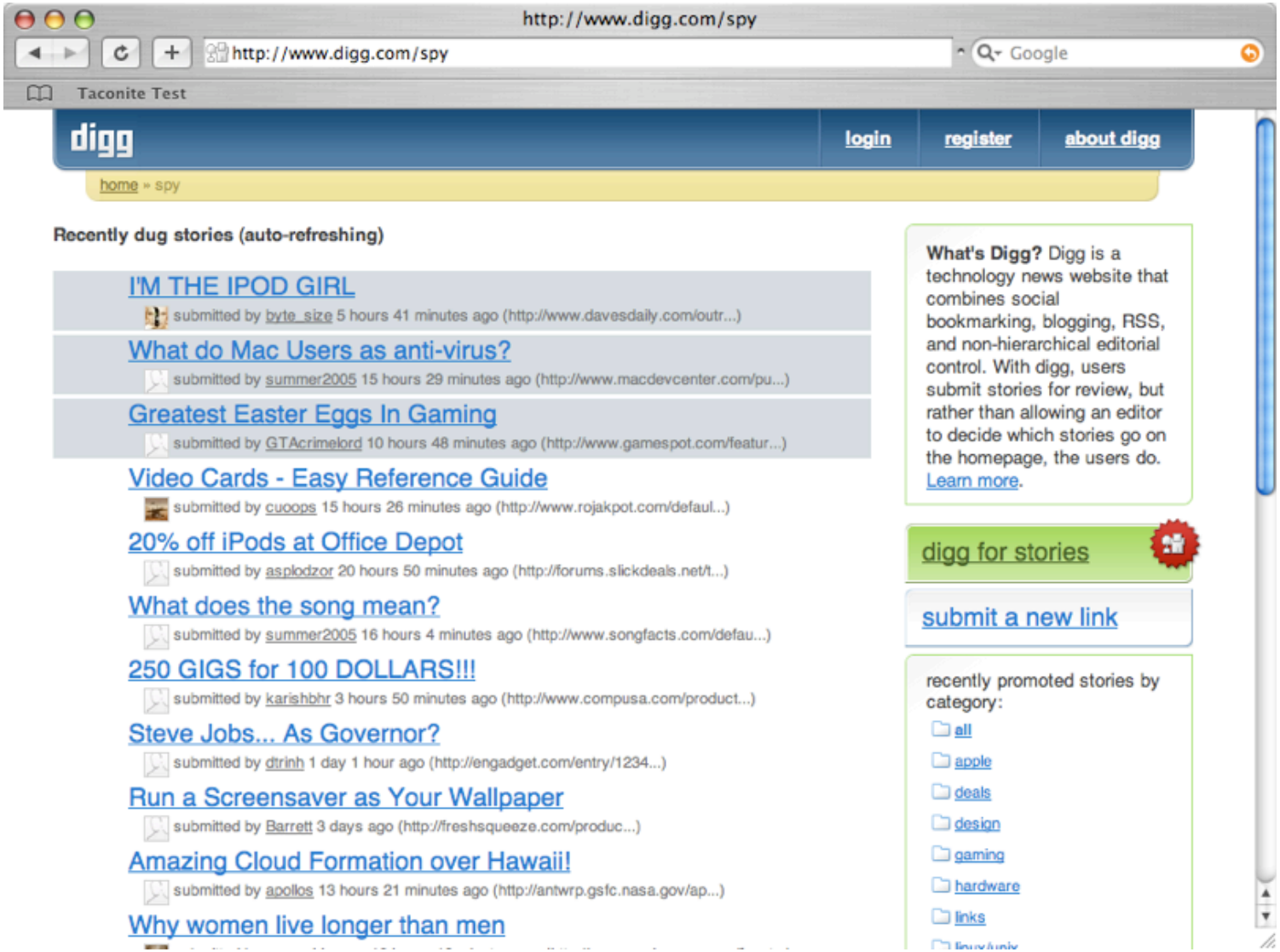
Proceed with caution

- Unlinkable pages - “Link to this page” option
- Broken back button
- Code bloat
- Graceful fallback - older browsers, screen readers
- Breaking established UI conventions
- Lack of visual clues - “Loading” cues



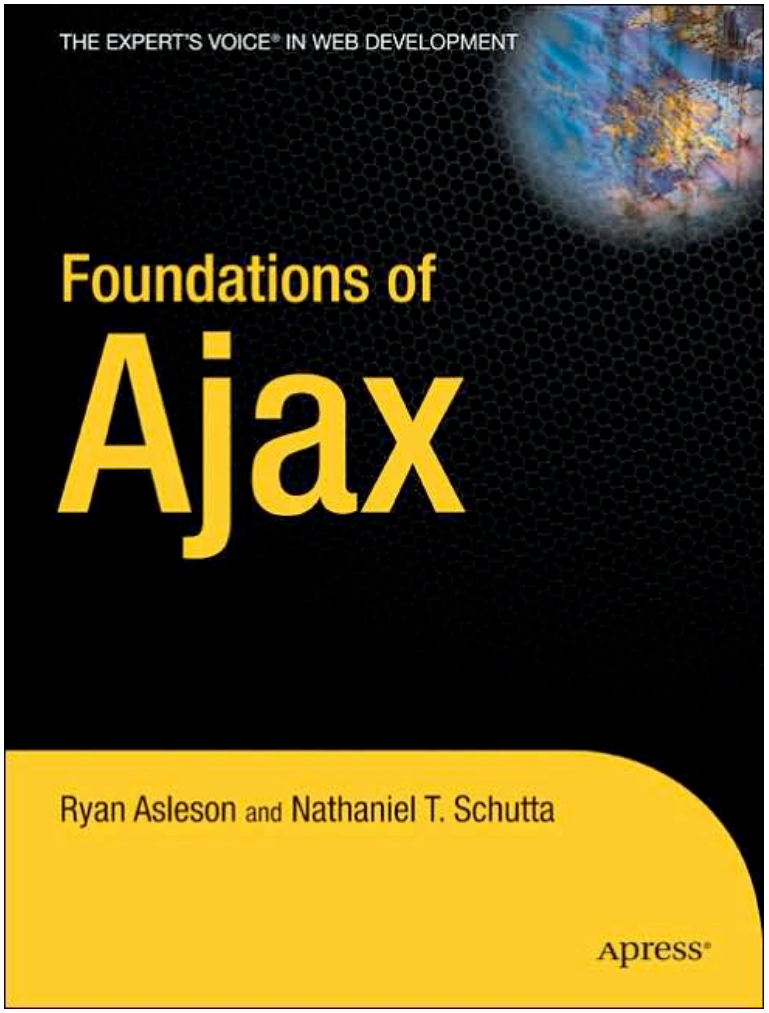


Fade Anything



Asynchronous changes - Fade Anything Technique

Give me more!



- www.ajaxian.com
- www.ajaxpatterns.org
- www.ajaxmatters.com/r/welcome
- www.ajaxblog.com/
- http://labs.google.com/
- www.adaptivepath.com/

To sum up

- Ajax changes the request/response paradigm
- It's not rocket science, it's not a cure all
- Test it with your users
- Start slow
- Embrace change!

Questions?!?

Thanks!