

---

## Introduction

---

Get this presentation at: <http://www.ashleyit.com/ajax/transport.html>

### Brent Ashley

- consultant, scripting specialist
- long history promoting web remoting

## Overview

The universal acceptance of the Ajax meme only became possible after more than 7 years of evolution and convergence of tools and techniques. Much of the evolution surrounded the transport layer, and there's still some change to come. I'm going to help you understand the issues that have driven the evolution so far and prepare you for future directions.

## Agenda

- History
  - how browsers and web apps have evolved
  - how and why different background transports have been tried along the way
  - how Ajax came to be the solution that stuck
- Transport Layer
  - what challenges you need to understand when choosing a transport
  - What transports are available
    - How they work
    - Pros, Cons
    - Examples and code
  - Simple multi-transport app
- Where to from here?
  - Now
  - Future

---

## - Pre-Ajax Web Apps

---

[Early Browser Scriptability](#) : [Microsoft Remote Scripting](#)

[Other Embeddable Transports](#) : [Browser Native Transports](#) : [Convergence](#)

---

## Early Browser Scriptability

---

### Netscape and IE Versions 2.x to 4.x (~1998)

| "Dark Days of DHTML" - [Dan Webb](#)

### Dynamic content

- Initially very rudimentary scriptability
  - form elements - validation, manipulation
  - document.write()
  - images, mouseovers
- JavaScript, JScript
  - specifications in flux - 1.0, 1.1, 1.2, 1.3
- DOM evolving
  - e.g. no document.images[] in IE3
  - Events: NS - capture; IE - bubbling; (Moz/NS6 - listeners)
- DHTML - Layers and Divs
  - src attribute
  - Post-Render manipulation
    - animation

### Developer sanity

- A scarce commodity - [Ode to Netscape 4.x haunting, ranting](#)
- improved somewhat by Cross-Browser Compatibility Libraries
  - [DynAPI](#), [cross-browser.com](#), [Dithered 1k DHTML API](#)
  - equalization wrappers, often lowest common denominator

### Bottom Line

- Whizz bang presentation workable though often painful
- notta lotta page-level application or data interaction

---

## Microsoft Remote Scripting

---

### Remote Procedure Calls

- Part of Microsoft Scripting Library
  - Java Applet on client
  - Active Server Pages on server
- invoke server procedures from client
  - proxy object
  - data marshalling

1998 - [Dino Esposito's MIND Article](#)

1999 - [Andrew Clinick's MSDN Tutorial](#)

### Java Issues

- Browser Settings
- Firewalls
- IE Mac JVM not scriptable
- Browser Upgrades
  - Netscape 5
    - security model
  - XP
    - Bye-bye JVM

---

## Other Embeddable Transports

---

### XMLHttpRequest

- Created by Microsoft for Outlook Web Access
- Internet Explorer 5
- ActiveX Object
  - only IE until Mozilla 1.0 release in 2002

### Macromedia Flash

- generally built as self-contained browser-hosted apps
- seen as external dependency
  - wide deployment across main browsers
  - no fringe browser support
- later Flash remoting
  - ColdFusion backend

---

## Browser Native Transports

---

### Community Collaboration

- [microsoft.public.scripting.remote](#) and other newsgroups
- geeky sites then later blogs

### Frameset

- wrap app in frameset
  - scripted communication between frames

### Iframe

- [JSRS](#) (2000)
  - Cross-browser
  - multiple server environments
- [Eric Costello's Apple article](#)

### Img Cookie

- [DepressedPress - GIF as Pipe](#)
- [RSLite](#) (2001)

### On-Demand Javascript

- [DotVoid](#) (2002) Danne Lundqvist

### HTTP Streaming

- [KnowNow](#) (2000) - prototype later [open-sourced as mod-pubsub](#)
  - custom server side, javascript microserver on client
  - publish/subscribe event routing

---

## Convergence

---

Early 2000's - Browser features converge

### Standards (~1999)

- [DOM](#)
- [CSS](#)
- JavaScript 1.5 / [ECMAScript 3](#)

### XMLHttpRequest adoption (2001+)

- Mozilla 1.0
- Safari 1.2
- Opera 8.0

### Browser Adoption, Attrition

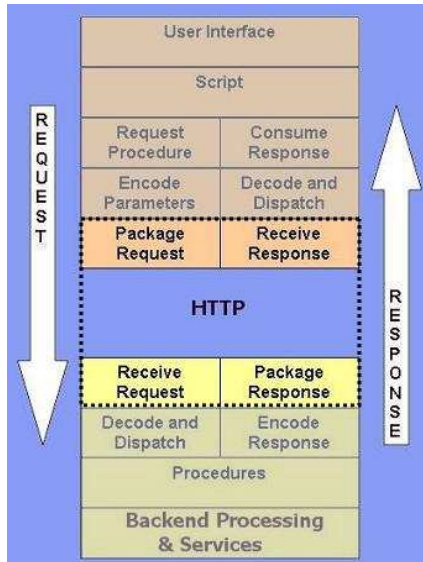
- Y2K, technology refresh
- Up with Mozilla / Firefox / Opera / Safari / IE 5+
- Down with Netscape 4.x, IE pre-5

### 2005 - Jesse [utters the A-word](#)

- [O frabjous day!](#) Callooh! Callay!

## Transport Layer

### :: Rich Application Processing Model ::



## Transport Layer

- Spans divide between client (above) and server (below)
  - includes request/response packaging interface
  - depending on transport may extend deeper to encoding layer

---

## - Transports in Detail

---

[XHR](#) : [IFrame](#) : [Img Cookie](#) : [On-Demand Javascript](#) : [Flash](#) : [HTTP Streaming \(Comet\)](#) : [DOM-Load-And-Save](#) : [JSONRequest](#) : [Fragment Identifier](#) : [Subspace](#) : [Purple Include](#)

---

# XHR

---

## References

[MSDN](#) : [Mozilla](#) : [W3C Draft Spec](#)

[TwinHelix](#) (HTMLHttpRequest work-alike object with fallback to Iframe)

## Features

- object available in IE/Mozilla/Opera/Safari...
- Sync and Async
- POST and GET

## Pros

- It's the X, don't you know
- [W3C draft spec](#)
  - [Ian Hickson's cross-domain proposal](#)
- not limited to XML

## Cons

- Same Origin
- no support by older or fringe browsers
- in IE, ActiveX control is sometimes blocked by security settings
  - not in IE7 with native browser object
- [Alex Bosworth - Rocky Shoals of Ajax Development](#)

## Caveats

- IE7 XHR ActiveX bug - [make sure you use native](#)
- IE7 XMLHttpRequest doesn't behave like a native object with expandos and prototype

## Code

```
function createXMLHttpRequest() {
  try{ return new XMLHttpRequest(); }
  catch(e) {}
  try{ return new ActiveXObject("Msxml2.XMLHTTP"); }
  catch (e) {}
  try{ return new ActiveXObject("Microsoft.XMLHTTP"); }
  catch (e) {}
  alert('Could not create XHR!');
}
```

```
function doXHRSync( url ){
  var xhr = createXMLHttpRequest();
  xhr.open("GET", url, false);
  xhr.send(null);
  processResponse( xhr.responseText );
}

function doXHRAsync( url ){
  var xhr = createXMLHttpRequest();
  xhr.onreadystatechange = function() {
    if( xhr.readyState == 4 ){
      processResponse( xhr.responseText );
    }
  }
  xhr.open("GET", url, true);
  xhr.send(null);
}
```

# Iframe

---

## References

[Eric Costello's Apple article](#) : [AjaxPatterns](#)

## Features

- Hidden Iframe(s)
- GET or POST
- Callbacks

## Pros

- wide browser compatibility

## Cons

- browser history
- that #@%^! IE click
- iframe [not supported](#) by XHTML strict
- cross-domain policy
  - [iframe communication hacks](#)

## Code

```
<iframe
  id='Iframe'
  style="height=1px;width=1px;visibility:hidden"
/>

function doIframe( url ){
  var iframe = document.getElementById('Iframe');
  iframe.onload=onIframeLoad;
  iframe.src = url;
}

function onIframeLoad(){
  var iframe = document.getElementById('Iframe');
  response = iframe.contentWindow.document.body.innerHTML;
  processResponse( response );
}
```

# Img Cookie

---

## References

[EggHeadCafe Article](#) : [RSLite](#)

## Features

- create an img object, set its url
- server sends cookie, empty image

## Pros

- lightweight
- really really extra wide compatibility

## Cons

- limited data due to GET and cookie limits

## Code

```
function doIMGCookie( url ){
    var img = new Image();
    document.cookie =
        'IMGCookie=x; expires=Fri, 31 Dec 1999 23:59:59 GMT;';
    img.src = url;
    window.setTimeout( "callbackCookie()", 1000 );
}

function callbackCookie(){
    var response = null;
    var aCookie = document.cookie.split("; ");
    for (var i=0; i < aCookie.length; i++){
        var aCrumb = aCookie[i].split("=");
        if (aCrumb[0] == 'IMGCookie'){
            response = unescape(aCrumb[1]).replace(/\+/g, ' ');
        }
    }
    if ( response != null ){
        processResponse( response );
    } else {
        alert( 'cookie not set' );
    }
}
```

# On-Demand Javascript

---

## References

[DotVoid](#) : [AjaxPatterns](#)

## Features

- create a script tag, set its src attribute
- server returns javascript to execute

## Pros

- no Same Origin restriction
- code executes without explicit callback

## Cons

- limited to returning Javascript

## Code

```
function doOnDemandJS( url ) {
  var head = document.getElementsByTagName('head').item(0);
  var old = document.getElementById('OnDemandJS');
  if (old) head.removeChild(old);

  script = document.createElement('script');
  script.src = url;
  script.type = 'text/javascript';
  script.defer = true;
  script.id = 'OnDemandJS';
  void(head.appendChild(script));
}
```

## Example

[del.icio.us bookmarks service](#)

- uses JSON to create bookmarks object

---

## Flash

---

### Actionscript Objects

#### [XMLSocket](#)

- Since Flash 5
- XML encoding

#### [flex.net.socket](#)

- Flash 8.5 / Flex 2.0
- Raw TCP/IP socket

### Javascript Helpers

Control Flash from JS: [AFLAX](#)

Control JS from flash: [Flex/Ajax Bridge](#)

#### [Flash4Ajax](#)

### Pros

- integrates with Flash ui components
- wide distribution
- persistent TCP connection

### Cons

- plugin dependency
- custom server
- port > 1024
- same-domain
  - mitigated by crossdomain.xml override
    - [security caveats](#) (use separate API domain)
- [AFLAX Demos](#)
- [FABridge Sample App](#)

### Examples

---

## HTTP Streaming (Comet)

---

### References

[Ajax Patterns](#) : [Alex Russell introduces Comet](#) : [Phil Windley on Comet](#)

### Features

- constantly open http connection
- server push

### Pros

- low latency
- event-based server push

### Cons

- client 2 connection limit
- server resources
  - mitigate by reconnecting per message
- pragmatically requires custom server component
  - [CometD](#)
  - [LightStreamer](#)
  - [PHP Cometd implementation](#)

### Target

- chat
- realtime event monitoring

### Code

- see [Ajax Patterns](#)

---

## DOM-Load-And-Save

---

### Reference

[W3C Spec](#)

### Features

- set of interfaces for loading and saving document objects and fragments
  - Serializer, Parser
  - Input, Output Streams
  - Stream Reader and Writer
- Parser takes specified action against supplied DOM context node once loaded
  - Append as Children
  - Replace Children
  - etc
- [Taconite](#) works similarly but with XHR as transport

### Pros

- W3C Recommendation

### Cons

- only barely implemented by Opera 7.6+
- partial Mozilla implementation

# JSONRequest

---

## References

[Douglas Crockford's Proposal](#)

partially implemented by [CrossSafe](#)

Collin Jackson's [Firefox Extension](#)

## Features

- Newly proposed browser builtin object
  - uses HTTP
  - reduced headers, no cookies
  - excess requests queued
  - must be JSON
  - Duplex connections accomodated
  - returns Request id

## Pros

- exempt from Same Origin restriction
- less overhead (headers, cookies)
- builtin attack throttles with random delay
- duplex allows server event push
- request ID solves sequencing

## Cons

- I didn't think of it
- JSON only
- duplex can use both connections depending on implementation
  - stretch RFC's "[should](#)" to allow for more connections?

## Code

```
requestNumber = JSONRequest(
  "https://json.weather.yahoo.com/request",
  {
    user: "doctoravatar@yahoo.com",
    t: "v1Ij",
    zip: 94089,
    forecast: 7
  },
  function (requestNumber, value, exception) {
    if (value) {
      processWeather(value);
    }
  }
);
```

```
        } else {  
            processError(exception);  
        }  
    }  
);
```

---

## Fragment Identifier

---

### References

[James Burke's original blog entry](#)

### Features

- modifies "anchor" portion of url of an iframe to communicate with its parent.

### Pros

- lightweight
- secure
- wide compatibility
- implemented by Dojo library

### Cons

- based on an anomaly that isn't defined by a specification

### Code

[See example from James](#)

---

## Subspace

---

## References

[WWW2007 Paper](#)

[CrossSafe](#)

[Google GData](#)

## Features

- uses javascript closures and hidden iframes with modified domain identifiers to create secure communication channel between iframes from different sources.

## Pros

- secure
- wide compatibility

## Cons

- unknown code ownership issues - Microsoft-sponsored research with no declared IP intentions.

## Code

see [CrossSafe](#) implementation

---

## Purple Include

---

### References

[Purple Include home](#)

### Features

- an abstraction on top of XHR
- allows inclusion of parts of other pages in your web page
- uses xpath syntax to select content from another page

### Pros

- simple
- flexible

### Cons

- need proxy for cross-site transclusions

### Code

[Purple Include Examples](#)

---

## - Challenges

---

### Implementation in the browser

- ScottAndrew on [ReadyState](#)
- [ActiveX restrictions in IE security patches](#)
- [Ajax on Mobile Devices](#)

### Same Origin restriction

- [cross-site scripting](#)
- cookies vulnerable
- inhibits direct mashups

### Client connection limit

- concurrent calls ([RFC 2616-8.1.4](#))
  - streaming

### Overhead

- HTTP headers
- [XML vs JSON](#) vs HTML vs Delimited

### Fault Tolerance

- Harry Fuecks on [Network Reliability](#)  
and [Latency and Sequencing \(AJAX@localhost\)](#)

### Performance

- [XMLHttp vs Iframes](#)

### Async programming model

- [Approximating Sync with continuations](#)

### Security vs Scalability

- [Shaping the Future of Secure Ajax Mashups](#)

---

## - All Together Now

---

### Naming

- Many ways to skin this cat, it ain't just an acronym any more.
  - encodings, for instance - JSON, formatted HTML, text
  - AJAX, AJAH, AHAH, AFLAX etc are simply implementations of the Ajax stack.

### Simple multi transport example application

- Yet another [Ajax Buzzword Bingo](#)

---

## Onward And Upward

---

### Status Quo

- XHR for most things
  - add fault tolerance improvements

### JSONRequest

- Could become preferred over XHR (assuming browsers implement it)
  - no cookie vulnerabilities
  - cross-site mashup support
  - request id
  - streaming support

### Exciting Developments

[JSONRequest Firefox Extension](#)

[CrossSafe](#)

[Google GData](#)

### Discussion points

[Douglas Crockford on Mashups](#)

[AdSafe.org](#)

---

## Credits

---

### TiddlyWiki

Jeremy Ruston created the original [TiddlyWiki](#) reusable non-linear personal web notebook.

The particular variant used in this presentation was derived from Nathan Bowers' [GTD TiddlyWiki](#) because it looks nice.

[Harry Fuecks](#) provided helpful critical review, advice and links.

---

[MainMenu](#) : [SiteTitle](#) : [AdvancedOptions](#)