

# Ajax

---

for PHP Developers

Sr. Developer @ Magnani Caruso Dutton

PHP Developer since end of 3.x

JavaScript hacker since '99

*Javascript developer/evangelist* since October '07

Blogger: <http://lovemikeg.com/>

Speaker: <http://lovemikeg.com/talks>

Do you code PHP 5 or PHP 4?

Anyone in here *love* JavaScript?

Anyone in here *loathe* JavaScript?

“AJAX”

does not exist.

“AJAX”

The term was originally coined by Jesse James Garret in '05 as “Asynchronous JavaScript + XML”

does not exist.

“I needed something shorter than ‘Asynchronous JavaScript+CSS+DOM+XMLHttpRequest’ to use when discussing this approach with clients.”

Jesse James Garret — March 13, 2005

does not exist.

“AJAX”

Bottom line: Ajax is not a single technology

does not exist.

“AJAX”

Ajax is a group of technologies which make web applications *feel* like desktop applications by eliminating a page refresh.

does not exist.



“AJAX”

Like *all* things web, Ajax has evolved considerably since its incarnation.

... including its definition.

does not exist.

AJAX  
is now  
Ajax

Depends on

XHTML + CSS for Presentation

DOM for dynamic display and interaction

XML + XSLT for data interchange and manipulation

XMLHttpRequest for data retrieval

JavaScript to glue everything together

Offers the same result, without formal requirements.

AJAX

Ajax is now

Ajax

This makes Ajax behaviors and patterns available to Flash, desktop applications, etc.

“Ajax” is proper.  
AJAX is now  
Consider “AJAX” deprecated.

# The Sales Pitch

# The Sales Pitch

Ajax can make your web applications more rich and responsive.



# The Sales

Ajax can reduce server load.

# Pitch

Ajax applications introduce a new approach to web development.

... and is kinda fun to code toward

# The Sales Pitch

Ajax is a buzzword which is here to stay.

making it highly marketable

Ajax applications tend to be more inviting (and oftentimes more usable) than standard web applications.

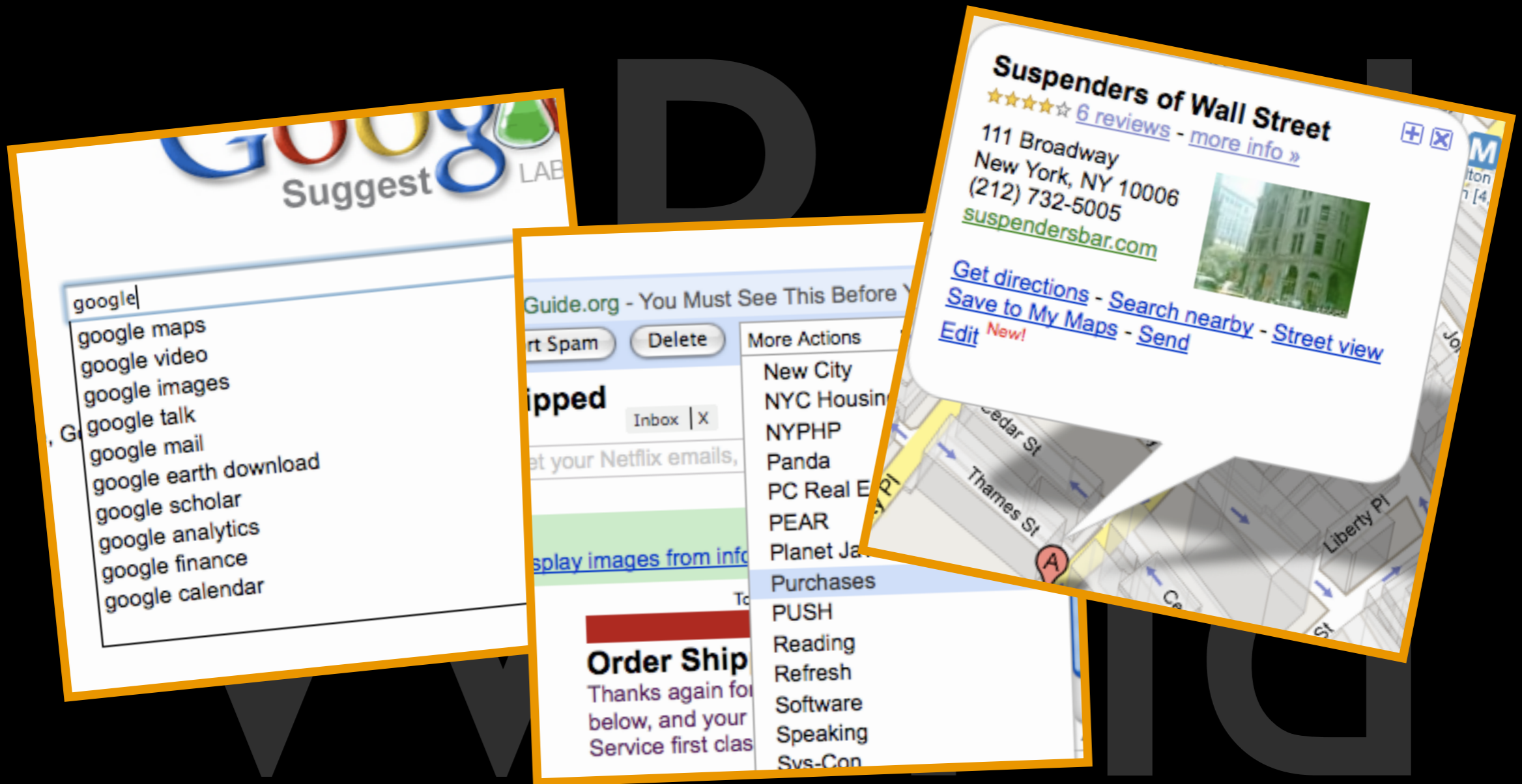
Although, not necessarily more accessible...

# The Sales Pitch

Even my parents knew what Ajax was.

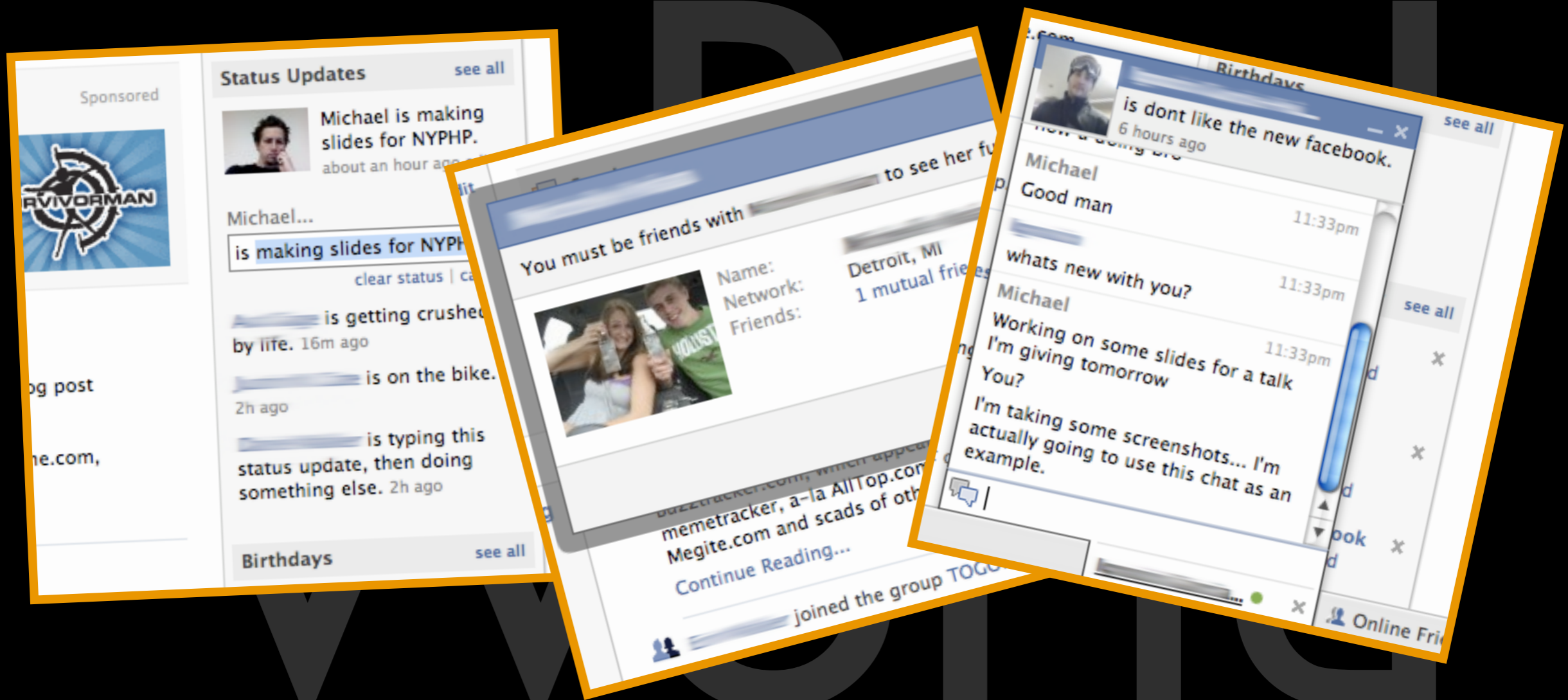
“That’s that *Google Satellite* thing. Right?”

in the **Real**  
**World**









# The Ajax Workflow

1. Instantiate the XMLHttpRequest Object
2. Configure the connection
3. Tell it what to do as its state changes
4. Send the request

```
// Instantiate
xhr = new XMLHttpRequest;

// Configure
xhr.open('GET', uri, true);

// What to do when as it changes state
xhr.onreadystatechange = stateChangeCallback;

// Send the request
xhr.send(null);
```

Looks simple enough

... but don't forget about IE 6 (more on that later)

```
// Instantiate  
xhr = new XMLHttpRequest;
```

```
interface XMLHttpRequest {
    readyState      : Number;
   .responseText   : String;
   .responseXML     : Document;
    status          : Number;
   .statusText      : String;

    onreadystatechange : EventListener;

    open (method, uri, async)      : void;
    send (data)                   : void;
    abort()                       : void;
    setRequestHeader(name, value) : void;
    getResponseHeader(name)       : String;
    getAllResponseHeaders()       : String;
};
```

### Tell the instance:

What request method to use

Where to go

Whether or not to go there asynchronously

```
// Configure  
xhr.open('GET', uri, true);
```

### Possible readyState values:

0: Unsent

1: Opened

2: Headers have been received

3: Loading

4: Done

```
// What to do when as it changes state  
xhr.onreadystatechange = stateChangeCallback;
```



With [semi] optional data:

Only required if sending POST data

GET requests need null (only in IE)

```
// Send the request  
xhr.send(null);
```

# Live Demo

(Code Download)

# Differences Between Browsers

# Differences

Believe it or not, XMLHTTP was a  
Microsoft technology

Via an ActiveXObject instance

# Between Browsers

## Differences

Later on the W3 standardized the object now known as XMLHttpRequest

Via a direct XMLHttpRequest instance

## Browsers

## Differences

Up until IE7, Microsoft used an ActiveX control to instantiate the XMLHttpRequest object.

## Between Browsers

# Differences

Other clients just did what the W3 told them.

# Between

# Browsers

Differences

Between

You *have to* code for this.

Browsers



Step 1: Code a function for W3 XHR

# Differences

```
var getW3XHR = function () {  
    return new XMLHttpRequest;  
};
```

# Browsers

## Step 2: Code a function for Explorer XHR

```
var getExplorerXHR = function () {  
    var xhr, axo, ex;  
    var objects = ['Microsoft', 'Msxml2', 'Msxml3'];  
    for (var i = 0; i < objects.length; i++) {  
        axo = objects[i] + '.XMLHTTP';  
        try {  
            xhr = new ActiveXObject(axo);  
            return xhr;  
        } catch (ex) {}  
    }  
    throw "Unable to create XHR object."  
};
```

### Step 3: Code a single getXHR function

```
var getXHR = function () {  
    if (window.XMLHttpRequest) {  
        return getW3XHR;  
    }  
    else if (window.ActiveXObject) {  
        return getExplorerXHR;  
    }  
}();
```

# Differences

It's not cross browser.

At least while IE6 is widely used.

# Browsers

# Differences

IE doesn't properly interpret this in  
`onreadystatechange`.

Reference an external variable instead.

# Browsers

## Differences

IE messes with the response headers.

use `headerValue.match(/something/)` to detect the value you want.

## Browsers

## Differences

Aborting XHR's is a pain

Make sure you unset `onreadystatechange` because it will fire on `abort()`

IE doesn't like it when you null the event, instead you have to set it to an empty function

## Browsers

# Helpful Hints



## “JavaScript Object Notation”

Considerably lighter than XML

Generally much quicker too

Security concerns can be mitigated by using JSONRequest.

<http://json.org>

See previous slide.

Plain text and HTML (set via `innerHTML`) are ridiculously fast.

Non-IE browsers register text nodes as siblings... very annoying.

Don't forget to send as `text/xml`... otherwise it gets stuffed into `responseText`

If you must use XML

You didn't try hard enough

`foo.getElementsByTagName('bar')` is your friend

Ask yourself, “Is Ajax really necessary?”

Ajax should *progressively enhance* an already established UX... not create it.

If you suck at disciplined development, always “Phase Two” the Ajax.

If you're eval'ing JSON, run it through a RegExp first.

Blindly setting innerHTML is silly too: inline event handlers are a bad thing.

Just because you don't see it, doesn't mean there's any extra security.

Mozilla Developer Center: Ajax

[developer.mozilla.org/en/docs/AJAX](http://developer.mozilla.org/en/docs/AJAX)

Quirksmode XMLHTTP Articles

[quirksmode.org/blog/archives/coding\\_techniques/xmlhttp/index.html](http://quirksmode.org/blog/archives/coding_techniques/xmlhttp/index.html)

Wikipedia:Ajax

[en.wikipedia.org/wiki/AJAX](http://en.wikipedia.org/wiki/AJAX)

Ajax: A New Approach to Web Applications

[adaptivepath.com/ideas/essays/archives/000385.php](http://adaptivepath.com/ideas/essays/archives/000385.php)

### W3 XMLHttpRequest Recommendation

[w3.org/TR/XMLHttpRequest/](http://w3.org/TR/XMLHttpRequest/)

### Explorer XMLHttpRequest Documentation

[msdn.microsoft.com/en-us/library/ms535874\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/ms535874(VS.85).aspx)



# Questions?

# Thanks!

<http://lovemikeg.com/>