

A New Hero for the Web?

AJAX

Outline

- What is AJAX?
- What's the result?
- Is it new?
- What's involved in the technology?
- Pros, Cons, Issues
- Developer tools and examples
- Future directions, new methods



What is AJAX?

- **Asynchronous Javascript and XML**
- Programming technique for websites
- Collection of existing technologies
 - HTML (Hypertext Markup Language) + CSS (Cascading Style Sheets)
 - DOM (Document Object Model)
 - ECMA-compliant language, such as JavaScript
 - XMLHttpRequest Object in JavaScript
 - XML-formatted Data

What is AJAX?

- Markup and Visualization
 - HTML elements, CSS style
 - `<div>` `` tags to wrap content, style with CSS
- Access to marked-up elements
 - DOM, Object-Oriented view of the elements on a web page
- Manipulate elements in the DOM
 - Using JavaScript

What is AJAX?

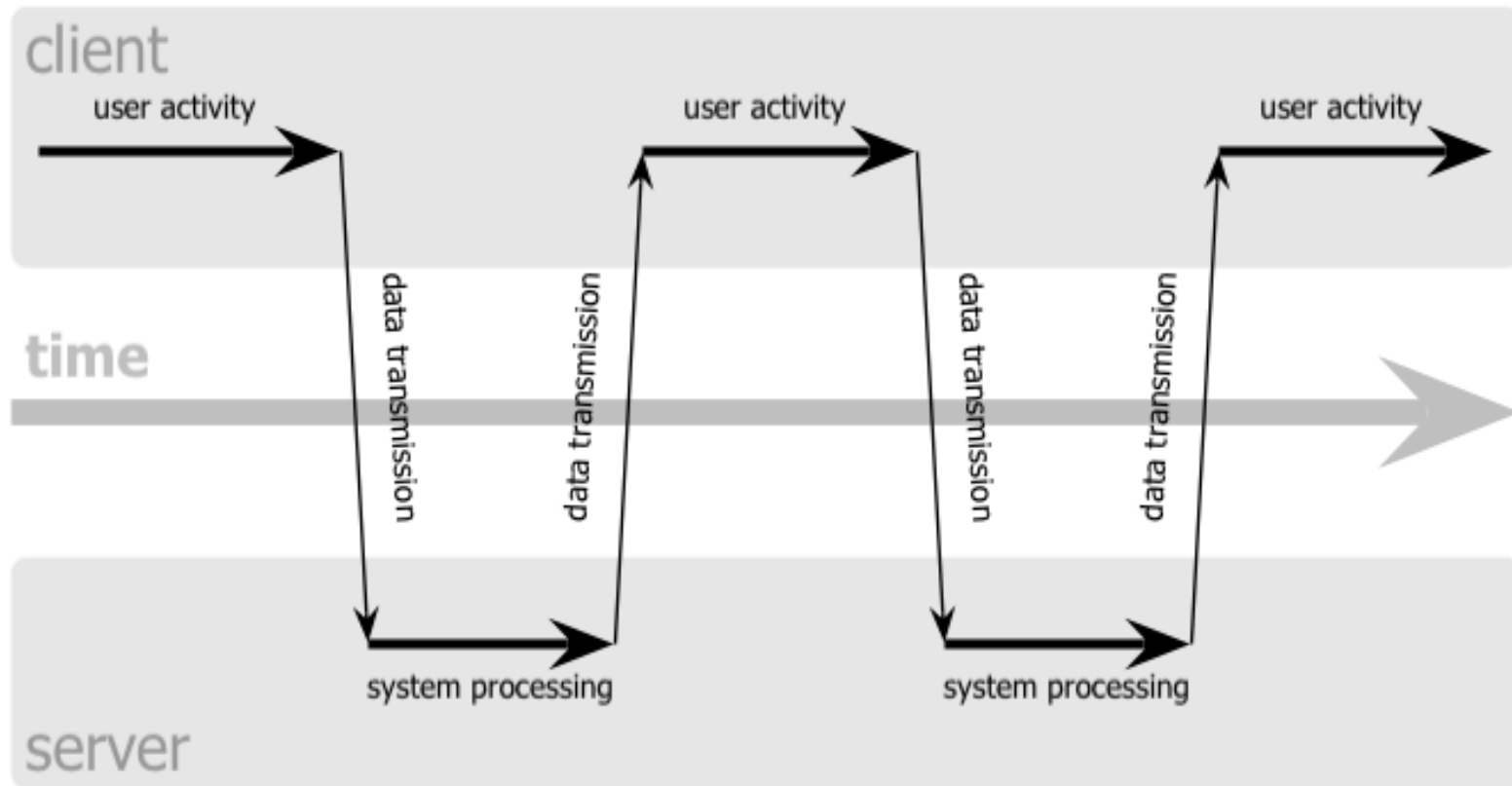
- Request more data from the server
 - XMLHttpRequest, class in JavaScript
- Format the requests
 - Often XML, but doesn't have to be ...
 - Plain text, JSON (JavaScript Object Notation), HTML, others often used
 - But AJ doesn't sound as ***extreme***, so it's AJAX

What's the Result?

- Generated web pages that do function calls back to the server for more content
- The web page is dynamically updated, without requiring the dreaded Refresh
 - Typically on user event (button press, item selection, etc.)
- Web pages can behave more like desktop applications

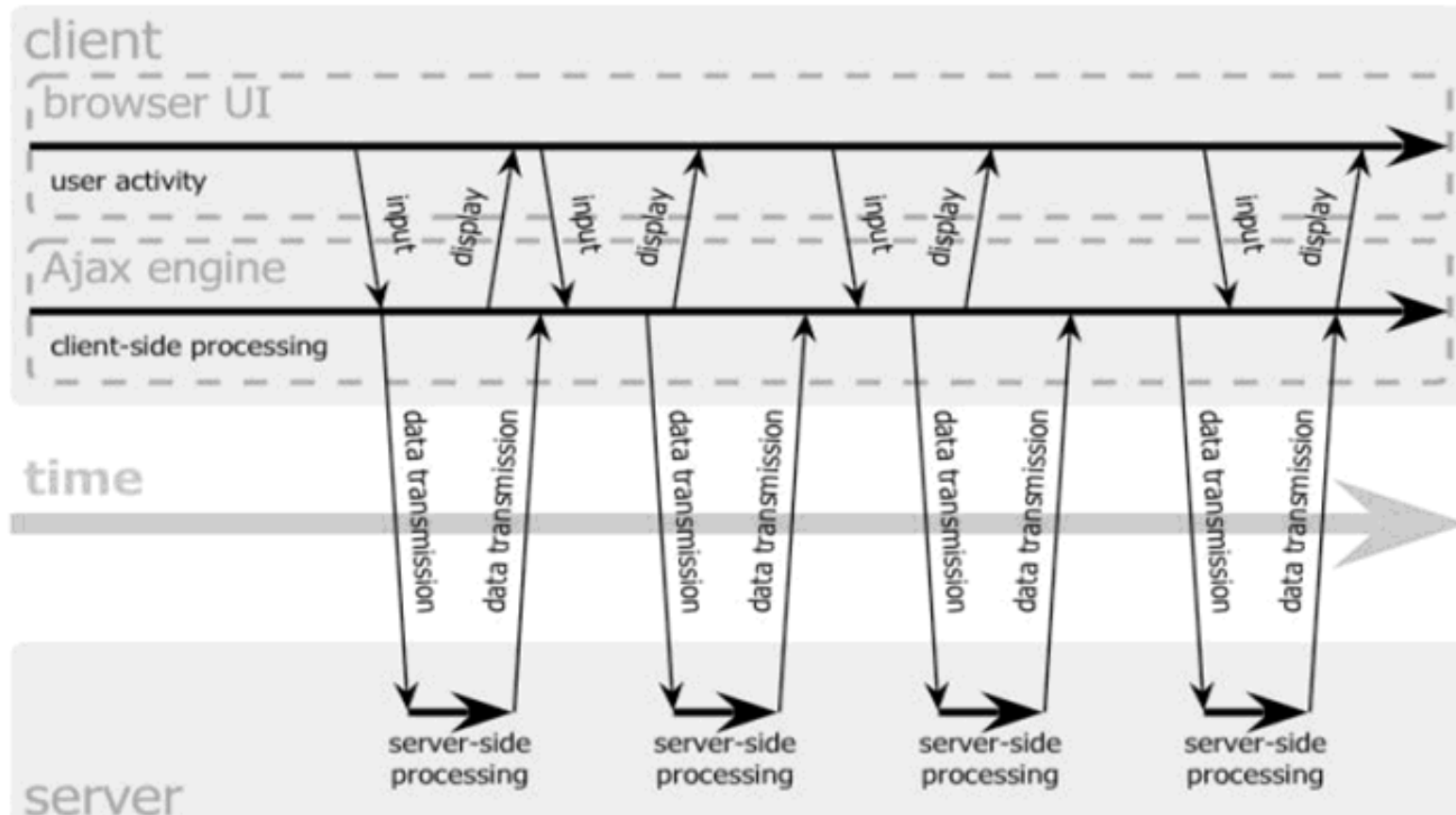
Web Flow: Synchronous

classic web application model (synchronous)



Web Flow: Asynchronous

Ajax web application model (asynchronous)



A New (2006) Idea?

- Term coined by JJ Garrett – February 2005
- “AJAXian” applications preceded use of term
- Microsoft publishes method for calling back to the server using ASP – August 2004
- Google Maps, Google Suggest: two early AJAXian applications

History of Dynamic Webpages

- 1996
 - Dynamic-HTML, dynamically setting the src variable within <Iframe>, <Layer> tags
- 1997
 - Macromedia Flash and Java load new source files without a reload
- 1998
 - Microsoft's Remote Scripting (MSRS) - Java object that could be called by JavaScript (Netscape Navigator 4 and Internet Explorer 4)

History of Dynamic Webpages

- 2000
 - MSRS commercially used in Outlook Web Access (part of Exchange Server 2000 package)
- 2002
 - MSRS morphs into XMLHttpRequest class
- 2003/2004
 - Remote scripting frameworks being developed by web technology firms
 - Google releases highly-visible AJAX applications

Web Browser Support

- Most critical component is XMLHttpRequest
- Supported since ...
 - Internet Explorer 5.0
 - Mozilla 1.0
 - Opera 8.0
 - Safari 1.2
 - iCab 3.0b



Pros vs. Cons

- Reduced bandwidth utilization
- Increased User Interactivity
- Reduced screen refresh
- New Acronyms and terminology!! :-)

- Usability issues related to Back button, bookmarking
- Hidden from non-JavaScript agents
- Difficulty meeting accessibility standards
- Heightened response time expectations

Alright, Let's See Some Code!

```
var req;

function getNewXMLData(url) {
    if (window.XMLHttpRequest) {
        req = new XMLHttpRequest();
        req.onreadystatechange = processReqChange;
        req.open("GET", url, true);
        req.send("");
    }
}

function processReqChange() {
    if (req.readyState == 4) { // only if req shows "loaded"
        if (req.status == 200) { // only if "OK"
            // ...processing statements go here...
        } else {
            // ... oops, there was a problem retrieving the XML data
        }
    }
}
```

Some Known Issues

- Internet Explorer caching model
 - Results of GET requests are cached if the URL doesn't change
 - Solution involves sending HTTP headers that precede the data
- `responseText` in `XMLHttpRequest` is ASCII
 - Breaks with non-ASCII characters
 - XML supports Unicode, but then you *have* to use XML for all communication with server

Some Known Issues

- Cross-browser Compatibility
 - XMLHttpRequest is in a different namespace in Internet Explorer than in Firefox
 - Have to add logic to all AJAX applications to detect browser-type
 - Uh oh, this is getting lame ...
- ... and many more common issues!

We need a Framework!

- Issues a reason to develop standard framework
- Dojo (Open Source)
 - Financial and manpower support from IBM, Sun
- Microsoft AJAX Toolkit (aka Atlas)
 - Server side extensions + Client toolkit
- Google Web Toolkit
 - Develop in Java, compile as Javascript
 - Apache License, but reports back with usage data!

Hey, What about Security?

- Client-side security concerns heightened compared to traditional HTML+CSS website model
- Potential to raise severity of existing JavaScript security concerns on client-side
 - ex. JavaScript doesn't have strong access control
- More security concerns on *server*-side than before
 - Due to increased complexity of applications

Hey, What about Security?

- Issues identified by Twynham
 - Technology is relatively new; good, tested examples are still few
 - Disjointed understanding of client and server side responsibility .. where to add what security?
 - AJAX developers end up writing way more scripts; many more points for failure
 - Insecure, unencrypted comms with server
 - Application-specific JavaScript viruses

Other Methods, Models

- New methods being developed
- Single-page Application
 - Entire application run as a single page, no refreshes at all
- Reverse-AJAX
 - Push data from the server, rather than pulling
- Comet
 - Extension of Reverse-AJAX, complete server event-driven model for client interaction

In Summary

- AJAX is DOM + JavaScript + Asynchronous requests to the server for more data
- AJAX is not *that* new, but still fresh, especially in regards to security best practices
- Microsoft ... sigh ... invented it
- There are issues, but they are being solved with libraries and frameworks



THE END.

References: <http://grabka.org/masters/ajax.html>