

# Redefining Web Interfaces

JPSpan, PHP, Javascript

OSCON 2005

David Uhlman

Customer Happiness Guru & CEO

Uversa Inc.

Presentation available on:

[www.lintellect.org](http://www.lintellect.org)

[www.uversainc.com](http://www.uversainc.com)

[duhlman@uversainc.com](mailto:duhlman@uversainc.com)



# About Me

- Favorite Pastime: Making coffee while web apps load between clicks
- Open Source advocate, developer, writer, speaker
- Executive at several Open Source companies, now CEO of Uversa Inc.
- My Motto: You can make a living selling Free Software but it depends on how you define "a living"



# What are we talking about exactly?

- Defining the “Click and Wait” Problem
  - How bad is it for the real use case you are trying to improve
  - Has web based been ignored previously because of interface/performance concerns?
- Why “Web Based” Again?
  - The Perception: “Web based applications are slow”
  - New Reality: “Can be as slow or as fast as the desktop in many cases now”



# What are we talking about exactly? (continued)

- History of options
  - Flash, Java, Other Plugins
- AJAX – Asynchronous Javascript & XML
  - Library, approach
- RIA – Rich Internet Application
  - More encompassing term
  - Inclusive of other kinds of solutions and features
  - RAJI, Rich Application Javascript Interface



# Get Rich Quick

- **Get Rich Quick**
  - 1) Take common app feature
  - 2) Now re-do it using AJAX
  - 3) ???
  - 4) Profit!!!
- **AJAX is not the answer to everything**
  - Adds complexity to apps, especially debugging
  - On larger scales can cause new types performance problems
  - Overall makes browser crash happier



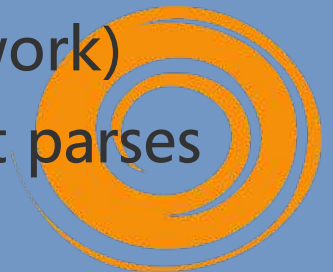
# Back to the Beginning Again

- Client Server -> Stateless -> Client Server ->
  - Stateless HTTP has been great but limits full fledged application feel due to “Click and Wait”
  - AJAX brings back what feels like a more client server approach
- What does AJAX do?
  - Enables server processing and communication out of band with page reloads
  - Taken further, complex rich interface widgets are now possible in pure DHTML



# Under the Hood

- Our Friend Javascript
- Why JPSSpan?
  - Object Oriented, auto registration “reflection”
  - Simple approach
- Everything is still HTTP like normal
  - Client calls mapped function on javascript object
  - Out of band, begins with Javascript post of XML to PHP “server” (cookies are passed so sessions work)
  - PHP Responds with serialized javascript, client parses and executes result, often an array



# Under the hood (continued)

- Asynchronous vs. Synchronous Requests
  - What does that mean?
  - Trade offs, complexity vs. user experience
- Weight and Performance
  - Round trip, gzip, error handling
- Gotchas
  - Browsers didn't necessarily test for all this. I.E. leaks objects like crazy
  - Unintended Consequences: Security & Performance



# Under the hood (continued 2)

- XMLHttpRequest is the workhorse
  - PHP always presents the “typeless” challenge
  - Can be used straight to accomplish very simple tasks
  - This is where JPSpan diverges by abstracting the low level with an simple API
- Handlers
  - Manual mapping of local javascript object methods or auto-map entire object
  - `$S = & new JPSpan_Server_PostOffice();`  
`$S->addHandler(new HelloWorld());`
  - JPSpan now passes through PHP errors



# How this changes everything

- Web based apps offer easy deployment, minimum maintenance, rapid customization, especially in PHP
- Historically limited to simple scenarios and or progressive users
- High performance data entry was always a problem. The payroll example...
- This is an iceberg, we haven't gotten near the water level yet. Just wait for SVG...



# How this changes everything (continued)

- This is all just a tool to extend the reach of server side logic
- New challenges are now building more powerful DHTML components to take advantage
- This makes SVG very compelling
  - FireFox will include SVG “inline” which is very exciting



# Nitty Gritty

- Dissecting the Hello World example
- How to debug?
  - on the wire
  - server logging
  - broken output
  - error pass through with alerts
- Performance choices
  - Server, Client, compression, weight



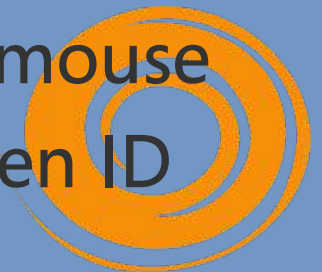
# Nitty Gritty (continued)

- Approaches to connect remote logic with the rest of your application
- Just because you can remote something doesn't mean you should
- Dealing with IE
  - Drip, the leak detector,  
<http://jgwebber.blogspot.com/2005/05/drip-ie-leak-detector.html>
- Remoting is most powerful when combined with other javascript at the client that enhances the UI
- Managing complexity



# The AutoComplete Box

- Improves web interface design when using simple keywords to search large datasets
- AutoComplete life cycle
  - Text is entered into standard input box
  - on each key entry an AJAX call is made
  - a DHTML displays several columns of information
  - User makes selection with keyboard or mouse
  - Selection is put into form space as hidden ID



# The AutoComplete Box (continued)

- Modular
  - Just a few lines of javascript in the page featuring the component, import the autocomplete.js file
  - back end just has to be the JPSSpan class and an object that supports returning a remoting friendly associative array
- Looking at the autocomplete.js in depth
  - Performance choices in key timings
  - Why do this synchronously?



# Notifications

- PUSH is the new PULL
  - Yes, it is still polling but to the user it feels like PUSH
  - Why not real asynchronous here?
- There is a scalability ceiling to this method but it offers something that can't be done other ways
- Apache and the web browsers can behave strangely depending on keep-alive timings and other network related parameters



# Notifications (continued)

- Our use case is to drive phone call links in web applications
  - call comes in, friendly banner appears offering links to that person in application
- Code is very simple, details are tricky
  - Lightweight JPSpan call is made to check for status change on javascript timer, timing is crucial
  - On status change event drives heavy weight call to get details



# Editable Grid

- BAM, Take it up a notch
  - Hugely complex widget to solve hugely complex use case
  - Still primitive compared to spreadsheet but improving all the time
- What does it do?
  - Offers rows and columns where cells can be edited without page reload
  - Validation of cell content against server side rules
  - Can be tied to locking and consistency protection



# Editable Grid (continued)

- Depends on a lot of client side javascript to handle tracking of what cell, what data, what changed
  - Behind the scenes every cell is just an ID, name and value
  - JPSpan calls trigger updates or update functions with those as arguments
- A number of hard problems
  - Paginating rows, pre-fetching
  - Browser mis-behavior



# Real World Applications

- ClearHealth
  - auto complete, editable grid
- SureInvoice
  - auto complete, retrofit
- ProContent
  - text management
- XRMS/Asterisk
  - notifications



# Resources

- JPSpan
  - <http://jpspan.sourceforge.net>
- Josh Eichorn's Blog
  - <http://blog.joshuaeichorn.com>
- Common Mistakes
  - [http://sourcelabs.com/ajb/archives/2005/05/ajax\\_mista](http://sourcelabs.com/ajb/archives/2005/05/ajax_mista)
- Debugging
  - <http://blog.monstuff.com/archives/000252.html>
- Uversa's AJAX components
  - <http://www.op-en.org>, See Celini
- Potential Pear HTML\_AJAX
  - <http://pear.php.net/pepr/pepr-proposal-show.php?id=276>



# Parting Gift

- Download the code and presentation from this:
  - <http://oscon.uversainc.com>
- Look for a more in-depth walk-through of the auto-complete coming up on OsNews.com
- Questions? Comments? Thanks!

