

Cool AJAX Stuff

Jake Gordon, Jamey Greenwood

twitter.com/jakeg82 twitter.com/jameyhg twitter.com/allyearbooks

What were the problems we were trying to solve?



- Too many tabs
- JavaScript requirement

What AJAX had we already done?

- Quick rehash of the chat system
- 30-second 'long poll' rather than short poll or HTTP stream
- Modals

If Facebook, Gmail etc can do it... so can we!

- There's nothing magical/secret
- Firebug's our friend!
- Facebook has crazily complex JS:

Think in terms of URLs

- Every page must be a URL which can be requested in the regular way...
- ...or via XHR
- /path/to = full HTML
- /path/to?json = HTML segments and JS to run wrapped in JSON

Catching hyperlinks

- `link 1`
- `link 2`
- `link 3`
- ...all three work without JS and just go to /path/to

Don't forget forms!

```
<form method="post" action="/path/  
to">
```

```
<form method="post" action="/path/  
to" class="no-ajax">
```

```
<form method="post" action="/path/  
to" class="target-modal">
```

jQuery 1.3's new "live"

```
$('#a').live('click', function() { /*  
*stuff*/ });
```

- What you would have to do before 'live' → rebind 'click' events after every XHR request
- 'live' saves you having to worry about this, seems to work well

Catching all our links

```
$('.a:not(.no-ajax)').live('click',  
function() { /*stuff*/ });
```

also apparently needed to add:

```
// only left clicks  
if (event.button !== 0) return true;
```

Catching all our form submissions

- The event for a form submission is 'submit' rather than 'click' BUT...
- though it may work in FF, jQuery 1.3.2 doesn't support:

```
$('#form').live('submit', function(){/*stuff*/});
```

- instead use:

```
$('#input[type=submit]').live('click', function() {  
/*stuff*/ });
```

Catching all our form submissions (cont...)

- ...then get a handle on the form with `$(this).parents('form');`
- Read the jQuery `.live` docs if you want to use it, only works for certain events
- So we use:

```
$('.form:not(.no-ajax) input[type=submit]').  
live('click',function(/*stuff*/) {});
```

So... we've hijacked all links and form submissions

- Now we need to actually do something with them :)

Sending XHR requests for links (as opposed to forms)

- Use `$.ajax()` rather than `$.get()` as gives extras like `request.abort()` and error handler:

```
request = $.ajax({  
  type: 'get',  
  url: address_to_fetch,  
  error: handle_json_error, // called on error  
  success: handle_json // called on success  
});
```

URL specifics

- The URL comes into the function as /path/to but we need to add ?json or &json to it
- PHP needs to know this as it will affect if HTML or JSON output is returned

```
if (address.match("\\?") {  
    var address_to_fetch = address + '&json';  
}  
else {  
    var address_to_fetch = address + '?json';  
}
```

Sending XHR requests for forms

- Make use of the jQuery.forms plugin
- Serialize all form data for us
- Deals with file uploads... which can't actually use XHR (<iframe> instead)
- This caused an awful lot of agro! (e.g. document.domain in <iframe>)

Sending XHR requests for forms (cont...)

- `this_form.appendJSON();`
 - appends `?json` to `/path/to` (or `?json_upload` if a file upload)
- `this_form.ajaxSubmit(form_options);`
- `form_options` includes custom JS callbacks (ask Jamey)

document.domain

- Main document and all <iframe>'s need it set even if already on that domain
- Because our chat system uses it in an <iframe>, our main document needs it
 - Which means we also need it for the file upload <iframe>
 - ...and the IE history keeper <iframe> (coming up)

But what about the back button?

- ...and bookmarkable URLs?
- The #hash fragment to the rescue!
- /path/to# /path/to/2
- Set and read with `window.location.hash`
- Writing to it adds a history event for the back button
- Works great in Firefox!

IE... wooh.

- <http://yuiblog.com/blog/2007/02/21/browser-history-manager/>
- Basically for IE, need to use an iframe to store back button history
- Can't just create the iframe on the fly re. `document.domain`
- Need an existing iframe which needs this in it:
`<script type="text/javascript">document.domain = 'example.com';</script> <div id="state"></div>`

Read our current history state

```
if ($browser.msie) {  
    currentHash =  
    $('iframe#history_iframe').contents().text();  
}  
else {  
    currentHash = window.location.hash.substr(1);  
}
```

Write to the IE iframe

`document.write()` to iframe adds browser history/state:

```
new_iframe = '<html><head><script type="text\
javascript">document.domain = \'example.com\';<\
script><\/head><body><div id="state">' +
top.location.hash.substr(1) + '<\div><\body><\
html>';
```

```
// doing 'document.write' to iframe adds new entry to
browser history/state
```

```
document.getElementById('history_iframe').contentWindow
.document.write(new_iframe);
document.getElementById('history_iframe').contentWindow
.document.close();
```

Bookmarkable URLs

- `/path/to#/path/to/2`
- Normally would just load `/path/to`
- In fact, no way (?) around this
- But then we redirect to `/path/to/2`
- So unfortunately two requests
- Prevent epilepsy with `$('#html').hide();` and `$('#html').show();`

Detecting #hash fragment changes

- 50ms poll
- In fact, we use this poll for all links as well
- `$('a').live('click', function() { /* change #hash */ });`
- The poll picks up the #hash within 50ms

Thank you

- Questions?
- Pub?