Prototype vs. jQuery

Nathaniel T. Schutta

Who am I?

- Nathaniel T. Schutta
  http://www.ntschutta.com/jat/
- @ntschutta
- Foundations of Ajax & Pro Ajax and Java Frameworks
- UI guy
- Author, speaker, teacher
- More than a couple of web apps

The Plan

- Why a library?
- Pros and cons.
- Ajax made easy.
- CSS selectors.
- Events.
- Animations and effects.

Use a library.

Not using one?

Probably doing it wrong.

Ajax isn’t rocket surgery.

But...doesn’t me we should go it alone.

Several high quality libraries to pick from.

Paradox of choice.

Most server side agnostic.

Things have improved since the 90s!

We still have browser issues to deal with.

<cough>IE 6.</cough>

Libraries abstract some of that away.

VM as it were.

Libraries = leverage.

Most have:

XHR wrapper.

CSS selectors.

Event handling.

Widgets, effects, animations.

JavaScript utilities.

How do you choose?

Many work well together.

Ask yourself...

What do you want?

Widgets? Complete environment? Utilities?

Can you read the code?

May need to from time to time.

How’s the docu?

What does Google return?

What is the community like?

When was the last update?

Can you get help?

Have to play with them.

Libraries have “flavors.”

Do you like the library's style?

Does it solve
YOUR problems?

Prototype vs. jQuery.

Both are small.

Readable code.

Similar features: XHR, CSS selectors, etc.

Excellent documentation.

Active mailing lists.

Widely used.

Fundamentally different philosophies.

Prototype is heavily influenced by Ruby.

jQuery - bit of a reaction to Prototype.

Each uses $ differently!

Each has influenced the other though...

Better, each has improved the other.

Pros and Cons.

Prototype's pros:

Adds useful functions to core elements.

Widgets and effects via script.aculo.us.

Ruby flavored JS.

Widely used in a variety of projects.

jQuery's pros:

Focussed on
HTML elements.

Doesn’t pollute global namespace.

DOM manipulation a snap.

Extensive plugins.

Widely used in a variety of projects.

Prototype's cons:

Where's the minified version?

Performance not always a priority.

Pollutes the global namespace.

jQuery's cons:

Parameter ordering in APIs not always intuitive.

Plugins required for a variety of functionality.

Some functions reassign ‘this.’

This is a bit nit-picky!

Both quite good...

Ajax made easier.

XHR isn’t complicated.

But abstractions help.

ActiveX vs. JS native...

All the wrappers
are pretty similar.

URI to call, parameters, HTTP method, callback.

Prototype: Ajax.Request.

Ajax.Request(url, [options])

Second arg is a hash of configuration options.

HTTP method, parameters, content type, callbacks.

```
new Ajax.Request('/fooApp/validate', {
    method: 'get',
    parameters: {zip: $F('zip'), city: $F('city'),
                 state: $F('state')},
    onComplete: function(request) {
        $('messages').update(request.responseText);
    }
});
```

$F - convenience method, retrieves value.

jQuery's approach.

Variety of methods.

$.ajax(options)

Also more specific calls.

$.getJSON(url, [data], [callback])

```
$.get("/DesigningForAjax/validate",
    {
        zip: $("#zip").val(),
        city: $("#city").val(),
        state: $("#state").val()
    },
    function(message) {
        $("#messages").html(message);
 });
```

Very similar!

JavaScript lacks namespaces - thus $.

Which is better?

CSS selectors.

CSS selectors are very powerful.

Browser support
can be lacking...

Libraries come in and smooth things out!

Prototype gives us $$.

Want everything
with a given style?

```
$$('.header').each(function(el) {el.observe("click", toggleSection)});
```

That's some terse code!

http://steve-yegge.blogspot.com/2008/02/portrait-of-n00b.html

$$ returns an array.

Prototype adds an each method to arrays.

Observe part of events...more in a minute!

When someone clicks a header element...

jQuery excels at CSS selectors.

Top notch support.

Outstanding performance.

Browsers do the heavy lifting when possible.

```
$(function(){
   $('.header').click(function() {
      $(this).next().toggle("blind", { direction: "vertical" }, 500);
   });
});
```

Put toggleSelect in the call in this example.

And we add a nice effect!

Little bit of code
= big return.

Which is better?

Event handling.

Coding to events gives us cleaner markup.

But browsers get in the way...again.

Unobtrusive JavaScript.

http://www.alistapart.com/articles/behavioralseparation

Library designers help us!

Prototype gives
us observe.

```
$$('.header').each(function(el) {el.observe("click", toggleSection)});
```

jQuery includes a variety of APIs.

Including some great helper methods.

```
$(function(){
   $('.header').click(function() {
      $(this).next().toggle("blind", { direction: "vertical" }, 500);
   });
});
```

Either way, we've attached a handler to the click event.

Which is better?

Animations and effects.

Neither stands alone.

Extensive add ons!

Want animations
and widgets?

script.aculo.us builds on top of Prototype.

Appear, fade, puff, pulse, fold, grow...

Drag and drop, controls.

Very useful stuff.

Let’s add an effect!

```
$('messages').update(request.responseText).highlight({duration: 1.0});
```

Notice the chaining?

jQuery relies on plugins.

jQuery UI.

Gives us widgets,
effects, draggable etc.

```
$(function(){
   $('.header').click(function() {
      $(this).next().toggle("blind", { direction: "vertical" }, 500);
   });
});
```

Very easy to
add a little flash.

Easy to go too far.

Seasoning, not the entree.

Blink tag anyone?

Used right, adds that extra bit of polish.

Help!

Both libraries have excellent online docu.

http://api.prototypejs.org/

http://docs.jquery.com/Main_Page

Each has books.

Google friendly ;)

Code is well written.

Easy to read.

Mailing lists are vibrant.

http://www.prototypejs.org/discuss

http://docs.jquery.com/Discussion

Which one is better?

It depends!

Honestly, both are great.

Can’t go wrong either way.

Passion on both sides...

Pros and cons.

http://blog.thinkrelevance.com/2009/1/12/why-i-still-prefer-prototype-to-jquery

Play with them.

Which one solves your problems?

Don’t like either?

Luckily, you've got plenty of choices!

Questions?!?

Thanks!

Please complete your surveys.