

jQuery

<http://github.com/relevance/jquery-demos>

stuart halloway

<http://thinkrelevance.com>

This work is licensed under the Creative Commons Attribution-Noncommercial-Share Alike 3.0 License.

<http://creativecommons.org/licenses/by-nc-sa/3.0/>

what is jquery?

unobtrusive

standard library

open source

ajax

one big idea

wrapped sets

1. selectors

the \$ function

`$(selector)`



context defaults
to entire DOM

`$(context, selector)`

basic selectors

syntax	finds
#foo	id
.bar	class
h2	element
li a	a descendants of li
li > a	a direct children of li
ol, ul	ols and uls
#main+p	p directly preceded by sibling #main
#main~p	p directly preceded by sibling #main

attribute selectors

syntax	tests
[href]	presence of class
[href = / foo]	equality
[href ^ = http :]	starts with
[href \$ = . jpg]	ends with
[href * = relevance]	contains
[href ! = / foo]	not equality

dom position selectors

selectors	
:first	:eq
:first-child	:gt
:last	:lt
:last-child	:even
:only-child	:odd
:nth-child(2), :nth-child(:even), :nth-child(odd), :nth-child(4n), :nth-child(3n+2)	

state selectors

state
:hidden
:visible
:enabled
:disabled
:selected
:checked
:animated
:contains(text)
:has(selector)
:parent

element types

shortcuts	
:button	:input
:checkbox	:password
:file	:radio
:header	:reset
:hidden	:submit
:image	:text

refining wrapped sets

method	notes
<code>add(expr)</code>	adds elements to set
<code>andSelf</code>	top two sets in stack
<code>contents</code>	element
<code>end</code>	previous set in stack
<code>filter(expr)</code>	keep matching elements
<code>map(f(idx,e))</code>	calls f for each element
<code>not(expr)</code>	remove matching elements
<code>slice(b,e)</code>	elements in range

family relations

selectors	
children	parents
next	prev
nextAll	prevAll
offsetParent	siblings
parent	
closest (ancestor)	
find (descendants)	

queries and enumeration

method
index
is
get
size
each

selectors demo impl

```
$(function() {  
    $("#highlight").submit(function() {  
        $("*").css("color", null);  
        $($("#selector").val()).css("color", "blue");  
        return false;  
    });  
});
```


finding paragraph 3

```
#p3  
p:eq(2)  
p:even:eq(1)  
p:contains(3)  
p[class^=honor]  
p[class$=bus]  
p:has(span):not(#p5)  
p:nth-child(3)
```

be careful with **not**

`:not (p : text (Paragraph))`

`p : not (text (Paragraph))`

document ready

```
$(document).ready(function () {  
    // do when document is ready  
})
```

```
$(function() {  
    // same as above  
})
```

2. dom manipulation

creating a node

```
$("#<a>").text("Clojure").attr({  
  href: "http://clojure.org"  
}).css({  
  border: "3px solid blue",  
  padding: "3px",  
  background: "#8888ff"  
}).prependTo("body");
```

changing nodes



`(dest).append(source)`

append
prepend
before
after
replaceWith
wrap
wrapAll
wrapInner



`(src).appendTo(dest)`

appendTo
prependTo
insertBefore
insertAfter
replaceAll

content

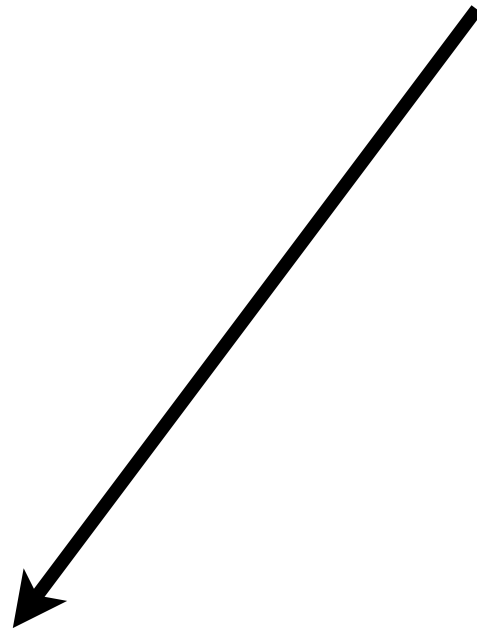
method
<code>html()</code>
<code>html(markup)</code>
<code>text()</code>
<code>text(str)</code>
<code>remove</code>
<code>empty</code>
<code>clone</code>

attributes

method
<code>attr()</code>
<code>attr(name)</code>
<code>attr(map)</code>
<code>css(map)</code>
<code>removeAttr(name)</code>

attribute normalization

```
$("#h2").attr("float", "right")
```



```
<h2 cssFloat="right">...</h2>
```

boolean normalization

```
$("#h2").attr("disabled", true)
```

```
<input value="Frak!" type="submit" disabled="***">
```

```
$("#h2").attr("disabled", false)
```

```
<input value="Frak!" type="submit">
```

sizing things up

method	notes
width	wrappers for css
height	
innerHeight	height - border + padding
innerWidth	
outerHeight (padding?)	height + border ?padding
outerWidth (padding?)	
offset	relative to doc
position	rel to offset parent

nonstandard attributes

```
$("#h2").attr("data-level", "sidebar")
```



legal in
html 5

```
<h2 data-level="sidebar">Re: ...</h2>
```

custom data

method
<code>data()</code>
<code>data(name, value)</code>
<code>removeData(name)</code>

custom data

method	
<code>addClass(sds)</code>	space-delimited string!
<code>removeClass(sds)</code>	
<code>toggleClass(name)</code>	
<code>hasClass(name)</code>	

don't worry about

normalizing attribute names

parsing css class string

global variables

calculating width and height

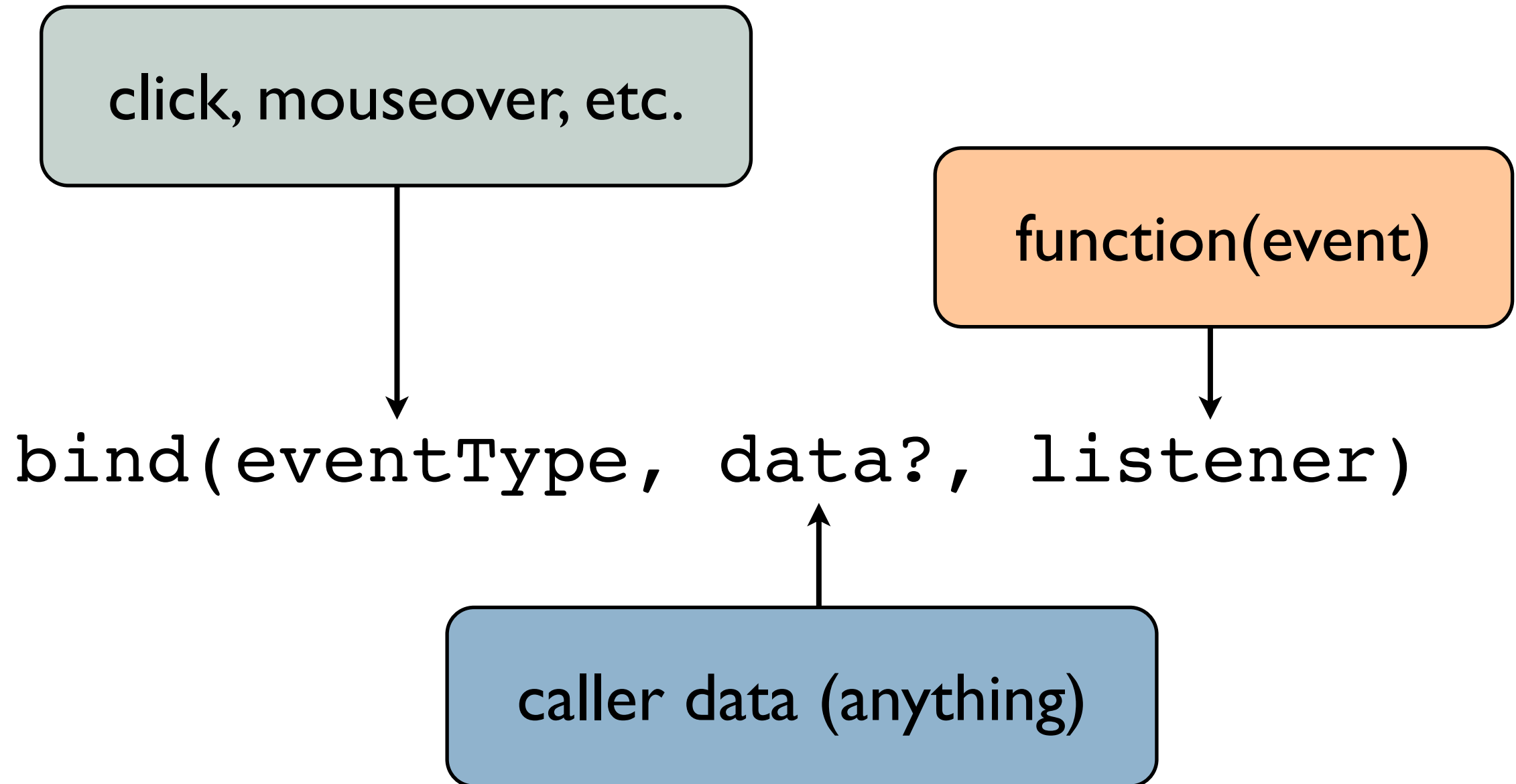
3. events

event example

```
<div id="field"></div>  
X: <span id="x"></span>  
Y: <span id="y"></span>
```

```
$("#field").mousemove(function(e) {  
    $("#x").text(e.pageX);  
    $("#y").text(e.pageY);  
}).hover(null, function(e) {  
    $("#x,#y").text("");  
});
```

binding event handler



interaction events

click

dblclick

mousedown

mousemove

mouseout

mouseover

mouseup

keydown

keypress

keyup

blur

change

focus

select

submit

hover

toggle

unload

unblur

`submit(...)`



`bind("submit", ...)`

normalized event props

altKey	screenX
ctrlKey	screenY
currentTarget	shiftKey
data	result
metaKey	target
pageX	timestamp
pageY	type
relatedTarget	which

normalized event fns

preventDefault	isDefaultPrevented
stopPropagation	isPropagation Stopped
stopImmediate Propagation	isImmediate PropagationStopped

hovering

~~mouseover/mouseout~~

`.hover(enter, leave)`



`.mouseenter(enter).mouseleave(leave)`

unbinding

```
foo.bind("submit.new", f1)
baz.bind("submit", f2)
bind.bind("submit", f3)
```

<code>#("*").unbind("submit.new")</code>	unbind namespace event
<code>#("*").unbind("submit")</code>	unbind “default” namespace
<code>#("*").unbind("submit", f2)</code>	unbind specific fn

live events

```
$(":submit").live("click", function() {  
    this.disabled = true;  
})
```

time

pow

```
$("<input>").attr({  
    type: "submit",  
    value: "pow"  
}).appendTo($("body"));
```

pow

pow

and there's more!

method	notes
one	like bind , but one-shot
trigger	trigger like user would
triggerHandler	fire code only
toggle	bind n listeners round-robin

don't worry about

dom level 0

bubbling

event instance

multiple handlers

dom level 2

capture

ie event model

4. utility functions

\$.grep

```
$.grep([1,2,3,4,5], isOdd)  
=> 1,3,5
```

```
$.grep([1,2,3,4,5], isOdd, true)  
=> 2,4
```

```
$.grep([1,2,3,4,5], isOdd, "blam")  
=> 2,4
```

\$.extend

```
x = {floorWax: "lemon"}  
y = {dessertTopping: "chocolate"}
```

```
$.extend(x,y)  
=> floorWax: lemon  
    dessertTopping: chocolate
```

\$.map

```
jQuery.map([1,2,3],  
  function(x) { return x*x; }  
)  
=> [1,4,9]
```

```
jQuery.map("attack", function(x) {  
  return x.charCodeAt(0)+1;  
})  
=> [98,117,117,98,100,108]
```

\$.inArray

```
$.inArray( "b", "foobar" )
```

```
=> 3
```

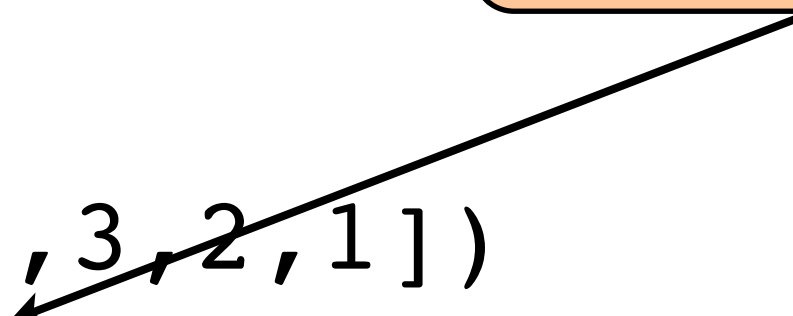
```
$.inArray( "c", "foobar" )
```

```
=> -1
```

\$.unique is evil

WTF?

```
$.unique([1,2,3,2,3,2,1])  
=> [1,2,3,2,3,2,1]
```



```
$.unique(["foo", "bar", "foo"])  
=> [foo,bar,foo]
```

unique works
only on node sets

\$.merge

```
$.merge([1,2], ["a", "b"])  
=> 1,2,a,b
```

```
$.merge([], null)  
=> Result of expression  
'second' [null] is not an object.
```

```
$.merge([1,2], {name: "Bill"})  
=> 1,2
```

\$.makeArray

```
$.makeArray(1)
```

```
=> [1]
```

```
$.makeArray([1])
```

```
=> [1]
```

```
$.makeArray(1, 2)
```

```
=> [1]
```

```
$.makeArray({name: "Bo"})
```

```
=> [[object Object]]
```

type testing

```
$.isArray([1,2,3])  
=> true
```

```
$.isFunction(alert)  
=> true
```

```
$.isFunction($)  
=> true
```

\$.trim

```
$.trim( "    foo    " )
```

```
=> foo
```

```
$.trim( "\nbar\n" )
```

```
=> bar
```

\$.param

```
$.param( {a: 1} )
```

```
=> a=1
```

```
$.param( {a: 1, a: 2} )
```

```
=> a=2
```

```
$.param( {a: {nested: 1}, b: 2} )
```

```
=> a=%5Bobject+Object%5D&b=2
```

\$.support

`$.support`

```
=> leadingWhitespace: true  
    tbody: true  
    objectAll: true  
    htmlSerialize: true  
    style: true  
    hrefNormalized: true  
    opacity: true  
    cssFloat: true  
    scriptEval: true  
    noCloneEvent: true  
    boxModel: true
```

5. conflict avoidance

preventing conflict

\$

=> function (selector, context) { ... }

jQuery.noConflict()

=> function (selector, context) { ... }

\$

=> null

have it both ways

```
// $ not defined
```

```
(function($) {
```

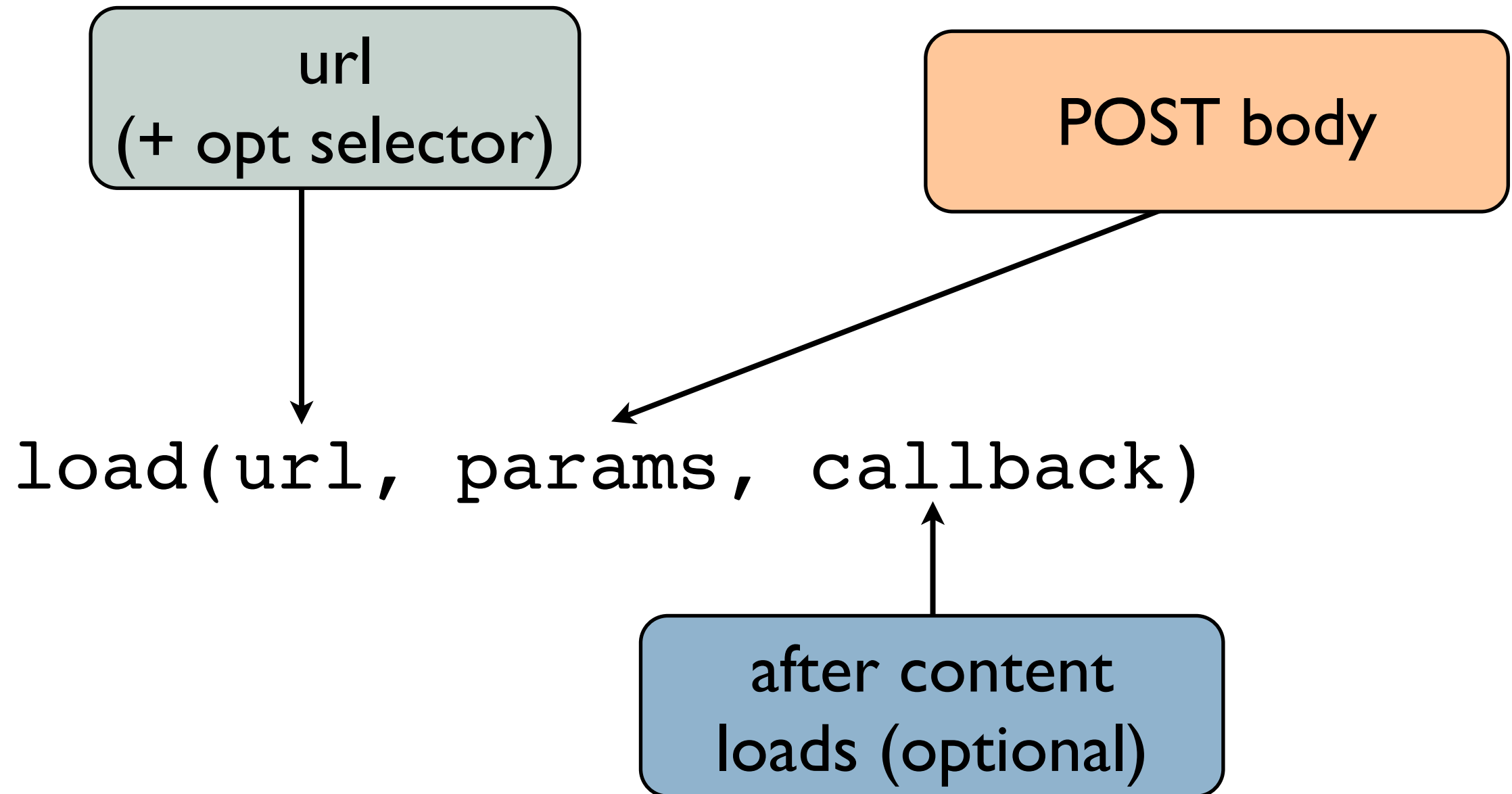
```
    // $ == jQuery
```

```
})(jQuery);
```

```
// $ not defined
```

6. ajax

load



utility functions

`serialize`



query string

`serializeArray`



array

```
$.get(url, params, callback)
```

```
$.getJSON(url, params, callback)
```

```
$.post(url, params, callback, type)
```

html | text | xml | json | script | jsonp

under the hood

```
$.ajax(options)
```

```
$.ajaxSetup(options)
```

```
options =
```

```
  url
```

```
  type data dataType contentType
```

```
  timeout async
```

```
  beforeSend success error
```

```
  ifModified processData
```

```
  global
```

events summary

local	global	notes
	ajaxStart	sent when starting & no requests active
beforeSend	ajaxSend	outbound hook
success	ajaxSuccess	HTTP success
error	ajaxError	HTTP error
complete	ajaxComplete	regardless of HTTP status
	ajaxStop	sent when last active request completes

don't worry about

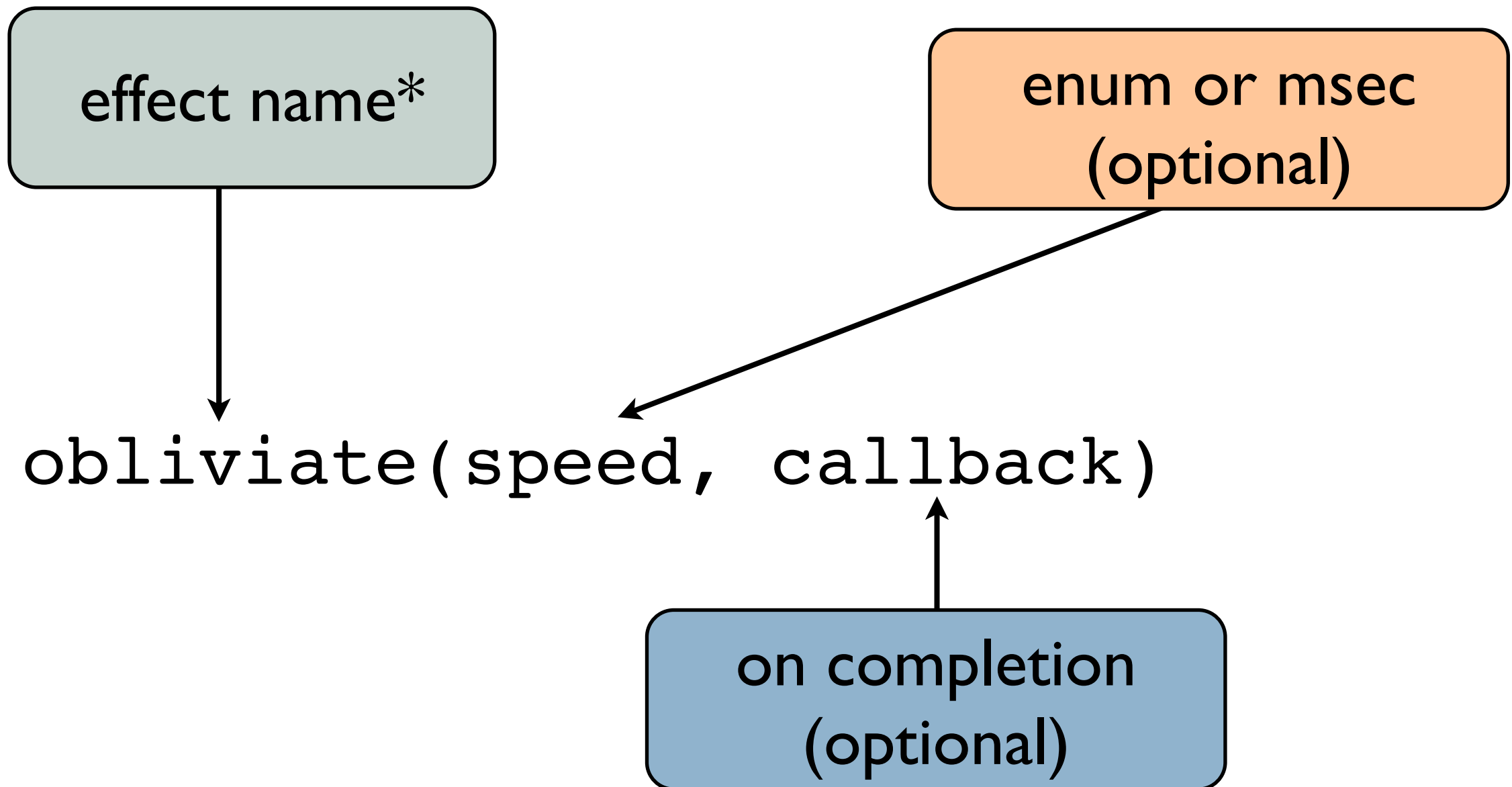
creating the xhr instance

ready states

tracking xhr across request/response

7. effects and animations

anatomy of an effect



*there is no function named obliviate

some basic effects

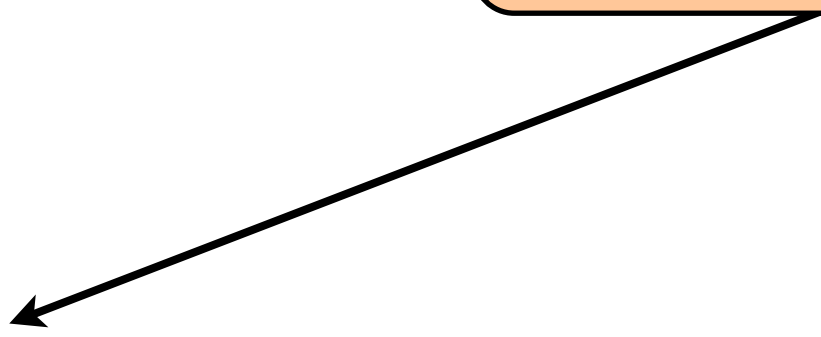
effect	modifies
show	size & opacity
hide	size & opacity
toggle	size & opacity
fadeIn	opacity
fadeOut	opacity
slideDown	reveal
slideUp	cover

effects demo code

```
$(function() {  
    $(".call-value").click(function() {  
        $("#target")[$(this).val()]();  
    });  
    $(".eval-value").click(function() {  
        eval($("#target")." +  
            $(this).val());  
    });  
});
```

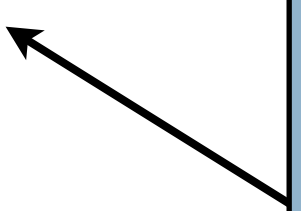
animate

css values to
animate

An arrow points from the orange box to the 'width: 25,' and 'height: 25,' lines in the code block.

```
animate({  
  width: 25,  
  height: 25,  
}, {  
  /* options */  
});
```

duration
easing
complete
queue
step

An arrow points from the blue box to the '/* options */' line in the code block.

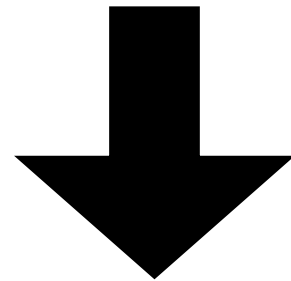
8. extending jquery

get text from anything

```
if (jQuery(this).is(":input")) {  
    return jQuery(this).val();  
} else {  
    return jQuery(this).text();  
}
```

this would be easier

```
if (jQuery(this).is(":input")) {  
    return jQuery(this).val();  
} else {  
    return jQuery(this).text();  
}
```



```
jQuery(this).valOrText();
```

add to jquery.fn

```
jQuery.fn.valOrText = function() {  
    if (this.is(":input"))  
        return this.val();  
    else  
        return this.text();  
};
```

not done yet...

support all aritys

```
jQuery.fn.valOrText = function() {  
  if (this.is(":input"))  
    return this.val.apply  
      (this, arguments);  
  else  
    return this.text.apply  
      (this, arguments);  
};
```

not done yet...

dry it up

```
jQuery.fn.getTextFn = function() {  
    return this.is(":input") ?  
        this.val : this.text;  
}
```

```
jQuery.fn.valOrText = function() {  
    this.getTextFn().apply(this, arguments);  
};
```

not done yet...

enumerate wrapped set!

```
// "set" case  
var args = arguments;  
this.each(function() {  
    $(this).getTextFn().apply($(this), args);  
});  
return this;
```

done (set)

get returns first item

```
var elem = this[0];  
if (elem) {  
    return $(elem).getTextFn()  
        .apply($(elem), arguments);  
} else {  
    return undefined;  
}
```

done (get)

valOrText

```
jQuery.fn.valOrText = function() {  
  if (arguments[0] === undefined) {  
    var elem = this[0];  
    if (elem) {  
      return $(elem).getTextFn()  
        .apply($(elem), arguments);  
    } else {  
      return undefined;  
    }  
  } else {  
    var args = arguments;  
    this.each(function() {  
      $(this).getTextFn().apply($(this), args)  
    });  
    return this;  
  }  
};
```

9. testing

unit testing:
screw.unit

screw.unit example

```
Screw.Unit(function(){  
  describe("Your application javascript", function(){  
    it("does something", function(){  
      expect("hello").toEqual("hello");  
    });  
  });  
});
```


mocking:
smoke

smoke example

```
it("can stub with Smoke!", function() {  
  stub(Foo, "bar").and_return(7);  
  expect(Foo.bar()).to(equal, 7);  
});
```

```
it("can mock with Smoke!", function() {  
  mock(Foo).should_receive("bar")  
    .with_arguments(10).exactly(1, "time").and_return(42);  
  expect(Foo.bar(10)).to(equal, 42);  
});
```

putting it all
together:
blue-ridge

<http://github.com/relevance/blue-ridge>

headless builds

```
rake test:javascrpts
(in /Users/stuart/presentations/refactoring-javascript)
Running application_spec.js with fixture 'fixtures/application.html'...
..

2 test(s), 0 failure(s)
0.456 seconds elapsed
Running numberformatter_spec.js with fixture 'fixtures/numberformatter.html'...
.....

40 test(s), 0 failure(s)
0.518 seconds elapsed
```

in-browser
builds

12345

99.00

40 test(s), 0 failure(s)
0.659 seconds elapsed

numberFormatter.normalizeOptions

- i. *detects required decimal zeros*
- ii. *ignores pure optional zeros*
- iii. *detects absence digit groups*
- iv. *detects presence of digit groups*

times100

- i. *works without decimal point*
- ii. *works with decimal point*

relevance's choices

screw.unit (bdd/unit testing)

smoke (mocking)

rhino (headless test runs)

env.js (stub browser environment)

blue ridge (framework around the above)

where we have been...

1. wrapped sets
2. dom manipulation
3. events
4. utility functions
5. conflict avoidance
6. ajax
7. effects and animations
8. extending jquery
9. testing

jQuery
in Action: <http://www.manning.com/bibeault/>

Slides &
Samples: <http://github.com/relevance/jquery-demos>

Email: stu@thinkrelevance.com

Office: 919-442-3030

Twitter: twitter.com/stuarthalloway

Facebook: [stuart.halloway](https://www.facebook.com/stuart.halloway)

Github: [stuarthalloway](https://github.com/stuarthalloway)

Talks: <http://blog.thinkrelevance.com/talks>

Blog: <http://blog.thinkrelevance.com>

Book: <http://tinyurl.com/clojure>

How we work: <http://howwework.thinkrelevance.com>

Biblio: <http://tinyurl.com/agile-biblio>