relevance

jQuery

stuart halloway
http://thinkrelevance.com

what is jQuery?

- JavaScript "standard library"
- open source
- unobtrusive
- Ajax

one big idea

1. wrapped sets

the $ function

```
$(selector)
```

context defaults
to entire DOM

```
$(context, selector)
```

basic selectors

syntax	finds
`#foo`	ids
`.bar`	classes
`h2`	elements

selectors demo html

```html
<div id="section-1">
  <p id="p1" class="vivid leader">Paragraph 1</p>
  <p id="p2">Paragraph 2</p>
  <p id="p3" class="honorificabilitudinitatibus">
     Paragraph 3 has a <span>nested span</span></p>
  <p id="p4" class="vivid">Paragraph 4</p>
  <p id="p5">Paragraph 5 has a
     <span>doubly <span> nested span</span> </span></p>
  <div>This is not a paragraph</div>
</div>
```

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)
```

selectors demo impl

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

2. events

don’t worry about:

obtrusive event handlers

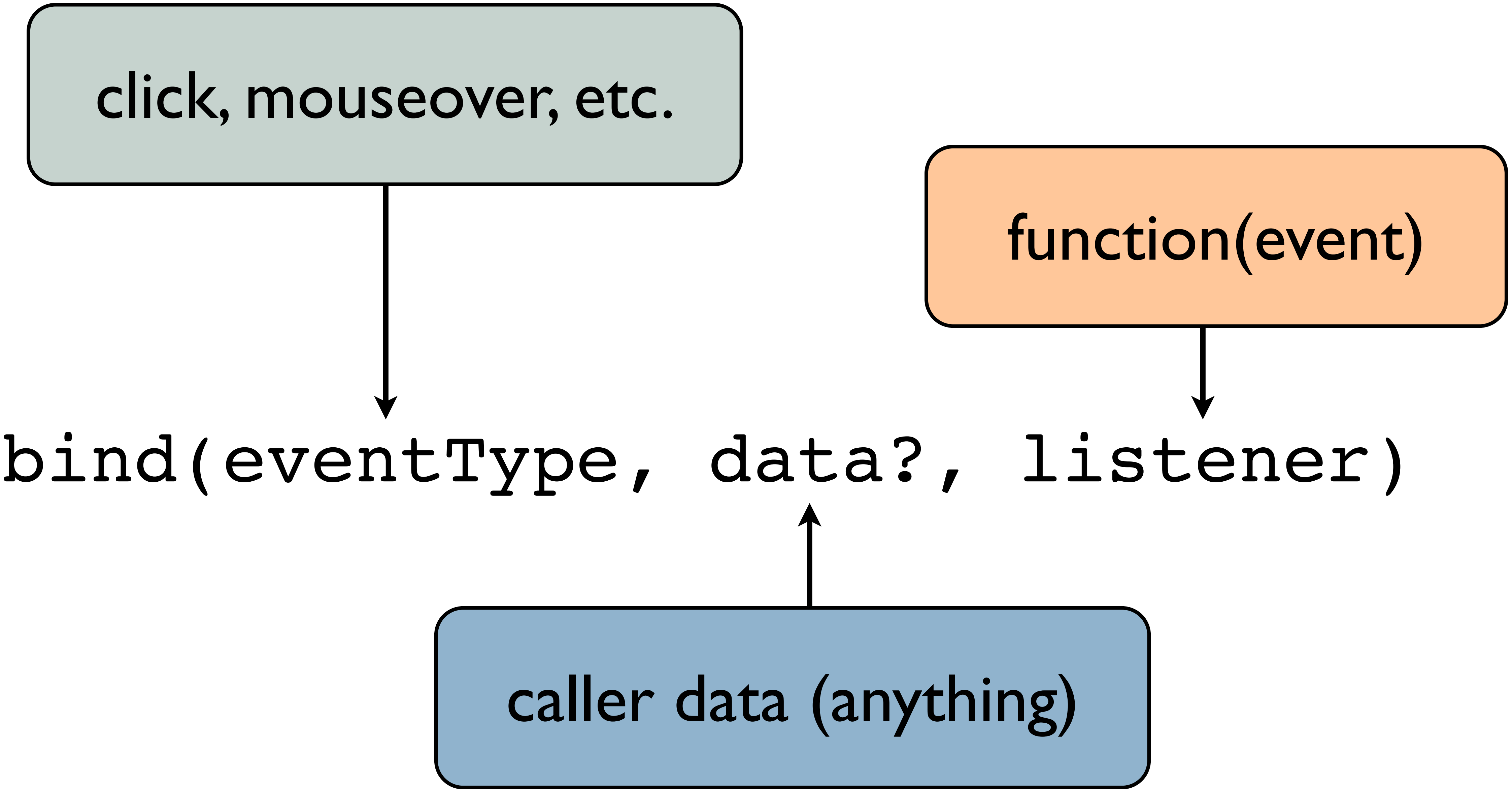
accessing the `Event` instance

event bubbling, capture vs. bubble phase

event properties

event listeners

binding event handler



click, mouseover, etc.

function(event)

```
bind(eventType, data?, listener)
```

caller data (anything)

interaction events

```
click
dlbclick
mousedown
mousemove
mouseout
mouseover
mouseup

keydown
keypress
keyup
```

```
blur
change
focus
select
submit
hover
toggle
unload
unblur
```

3. 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
```

4. conflict avoidance

preventing conflict

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

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

$
=> null
```

have it both ways

```
// $ not defined

(function($) {

  // $ == jQuery

})(jQuery);

// $ not defined
```

5. ajax

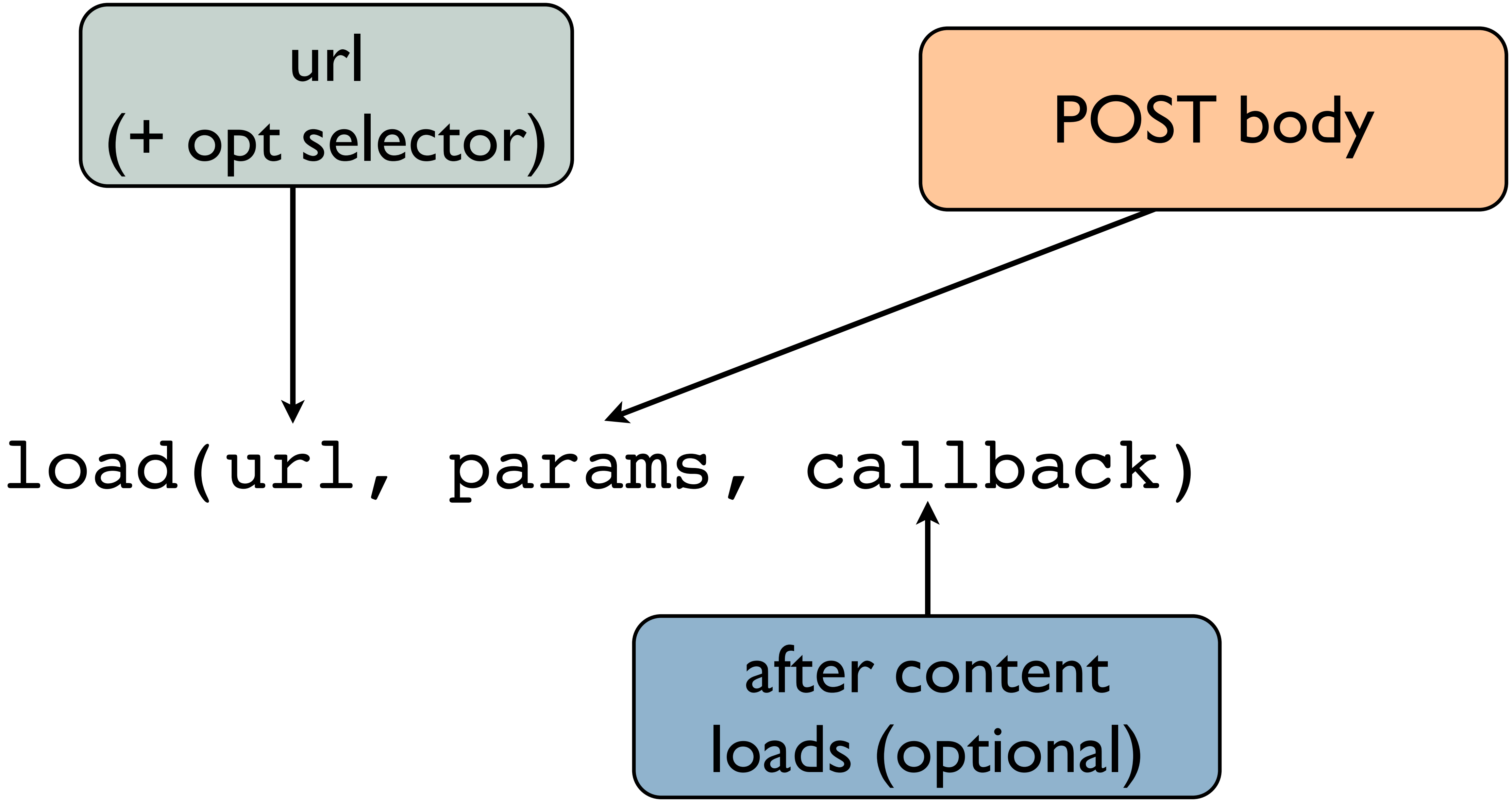
don’t worry about

creating the xhr instance

ready states

tracking xhr across request/response

load



url
(+ opt selector)

POST body

```
load(url, params, callback)
```

after content
loads (optional)

utility functions

```
serialize
serializeArray()

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

under the hood

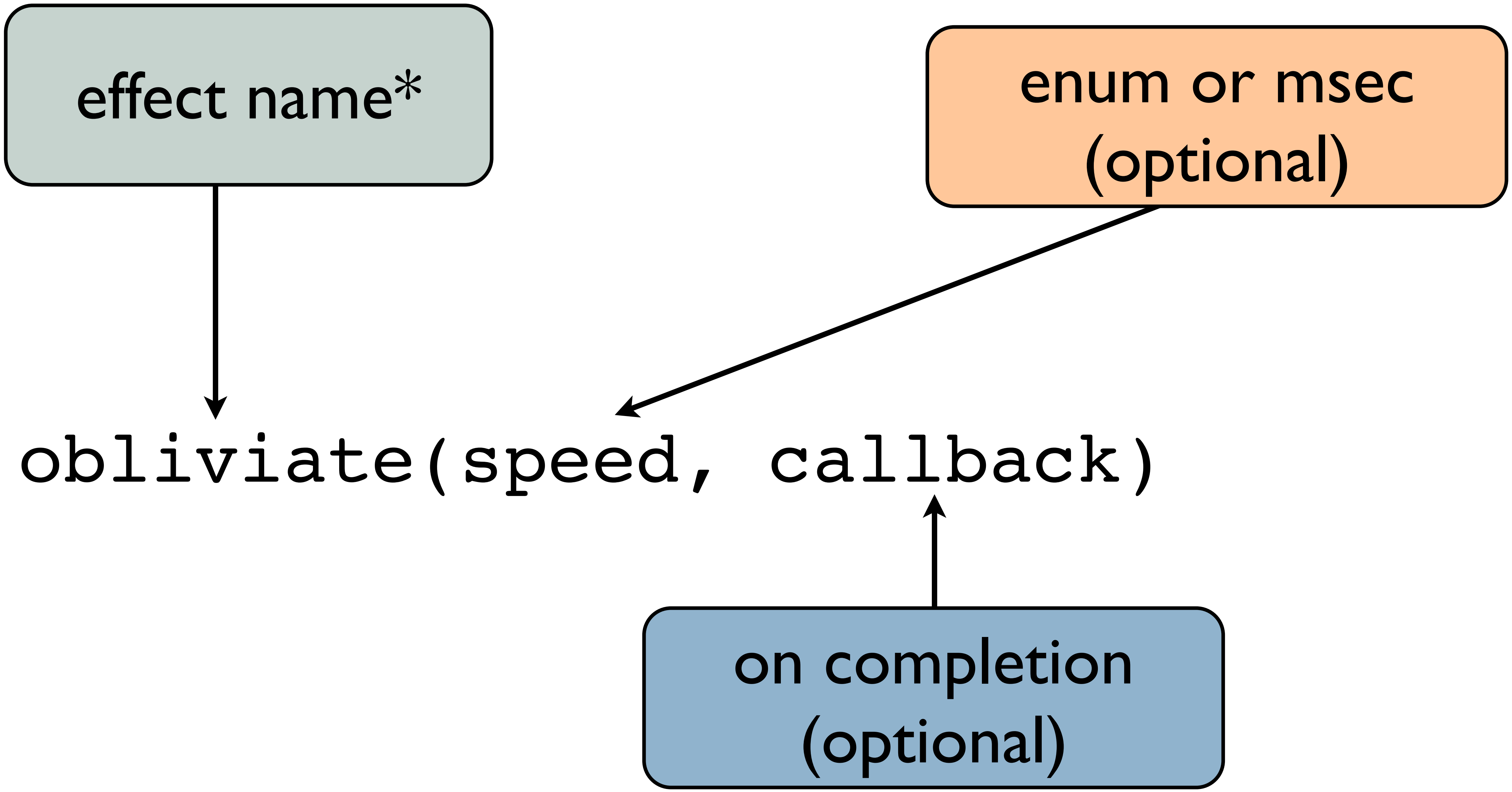
```
$.ajax(options)
$.ajaxDefaults(options)

options =
  url
  type data dataType contentType
  timeout async
  beforeSend success error
  ifModified processData
  global
```

global functions

```
ajaxStart(callback)
ajaxSend(callback)
ajaxSuccess(callback)
ajaxError(callback)
ajaxComplete(callback)
ajaxStart(callback)
```

6. effects and animations

anatomy of an effect



effect name*

enum or msec
(optional)

```
obliviate(speed, callback)
```

on completion
(optional)

*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());
    });
});
```

7. testing

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. events
3. utility functions
4. conflict avoidance
5. ajax
6. effects and animations
7. unit 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
Github:	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