



WHO AM I?



WHAT IS JQUERY?

JOUERNA'S  
CORE PHILOSOPHY

GET SOME ELEMENTS.  
DO SOME STUFF.

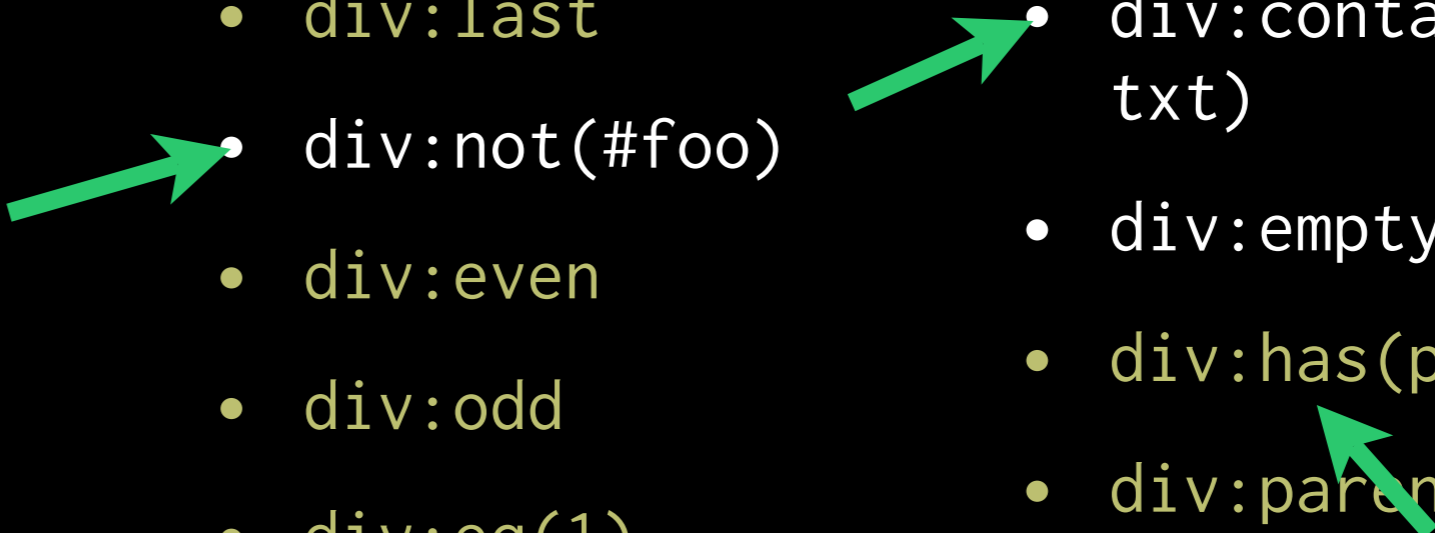
GET SOME ELEMENTS.  
BUT HOW?

WHY REINVENT THE  
WHEEL?

# CSS 3 PLUS

- `div`
- `div#foo`
- `div.class`
- `div, p, a`
- `div p`
- `div > p`
- `div + p`
- `div ~ p`
- `div:first`
- `div:last`
- `div:not(#foo)`
- `div:even`
- `div:odd`
- `div:eq(1)`
- `div:gt(1)`
- `div:lt(1)`
- `div:header`
- `div:animated`
- `div:contains(txt)`
- `div:empty`
- `div:has(p)`
- `div:parent`
- `div:hidden`
- `div:visible`

# CSS 3 PLUS

- div
  - div#foo
  - div.class
  - div, p, a
  - div p
  - div > p
  - div + p
  - div ~ p
  - div:first
  - div:last
  - div:not(#foo)
  - div:even
  - div:odd
  - div:eq(1)
  - div:gt(1)
  - div:lt(1)
  - div:header
  - div:animated
  - div:contains(txt)
  - div:empty
  - div:has(p)
  - div:parent
  - div:hidden
  - div:visible
- 

# CSS 3 PLUS

- div[foo]
- div[foo=bar]
- div[foo!=bar]
- div[foo^=bar]
- div[foo\$=bar]
- div[foo\*=bar]
- :nth-child(2)
- :nth-child(even)
- :first-child
- :last-child
- :only-child
- :input
- :text
- :password
- :radio
- :checkbox
- :submit
- :image
- :reset
- :button
- :file
- :hidden
- :enabled
- :disabled
- :checked
- :selected

GET SOME ELEMENTS.

```
$("table tr:nth-child(even) > td:visible")
```

DO STUFF.

`$("div")`

RETURNS JQUERY OBJECT

```
$("div").fadeIn()
```

RETURNS JQUERY OBJECT

```
$("div").fadeIn()  
.css("color", "green")
```

RETURNS JQUERY OBJECT

WE CALL THIS  
CHAINING

# CRAZY CHAINS

```
$(“ul.open”) // [ ul, ul, ul ]  
  .children(“li”) // [ li, li, li ]  
    .addClass(“open”) // [ li, li, li ]  
  .end() // [ ul, ul, ul ]  
  .find(“a”) // [ a, a, a ]  
    .click(function(){  
      $(this).next().toggle();  
      return false;  
    }) // [ a, a, a ]  
  .end(); // [ ul, ul, ul ]
```

# 5 PARTS OF JQUERY

DOM

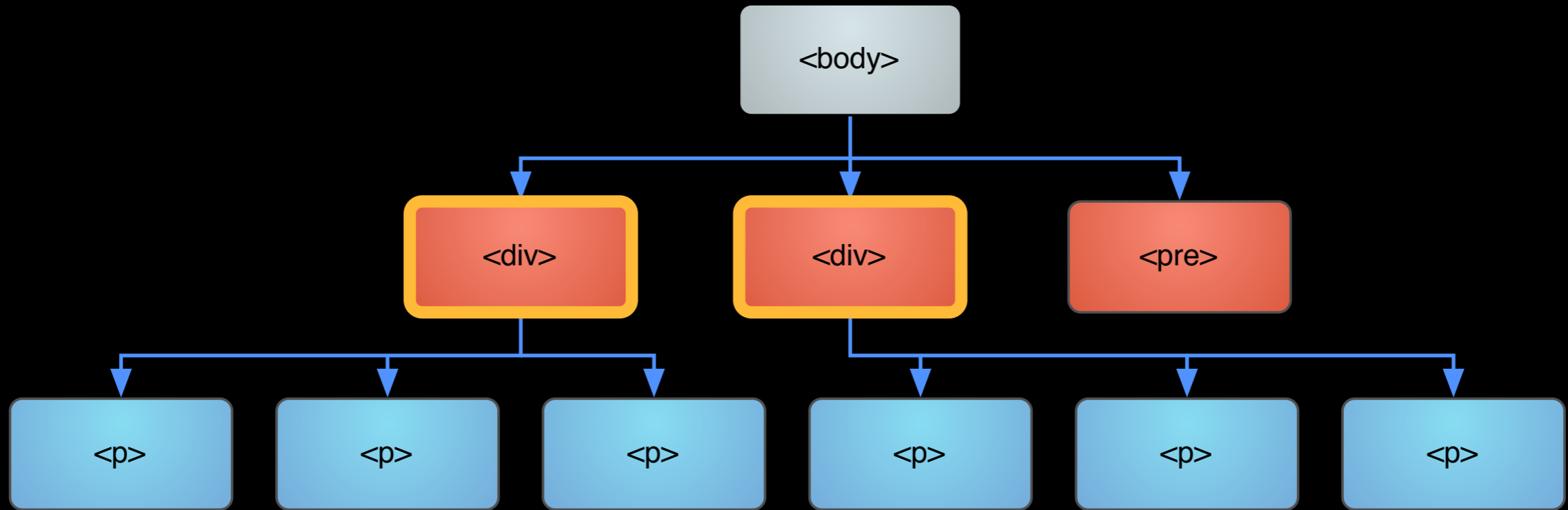
EVENTS

EFFECTS

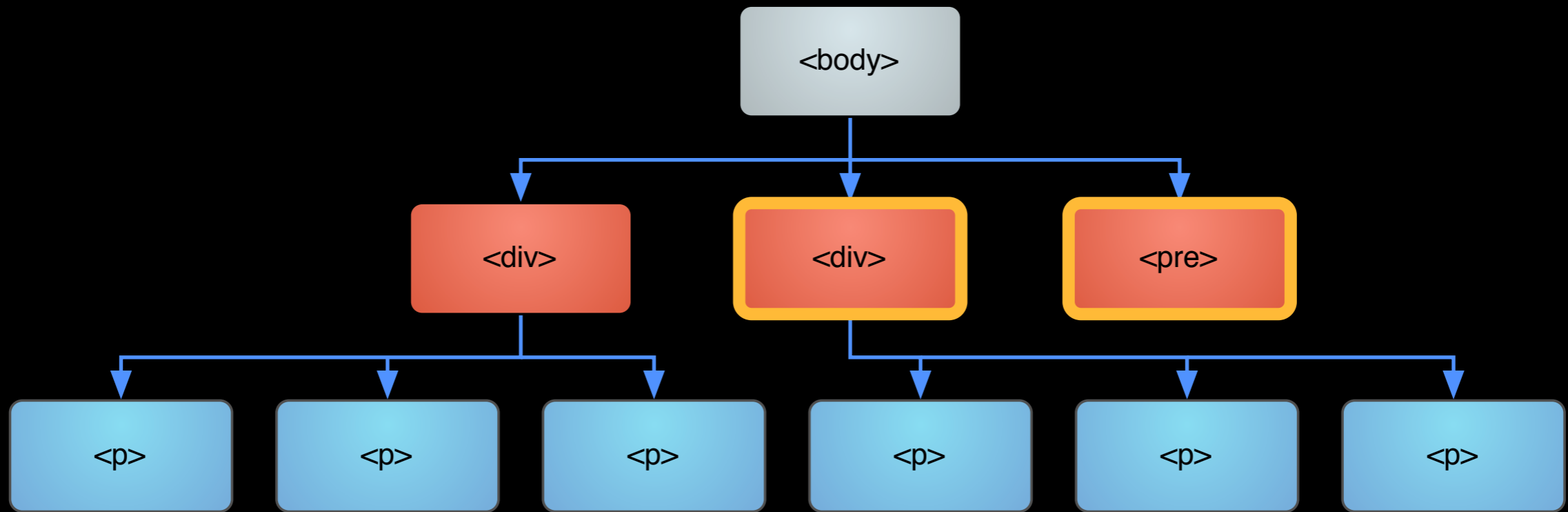
AJAX

PLUGINS

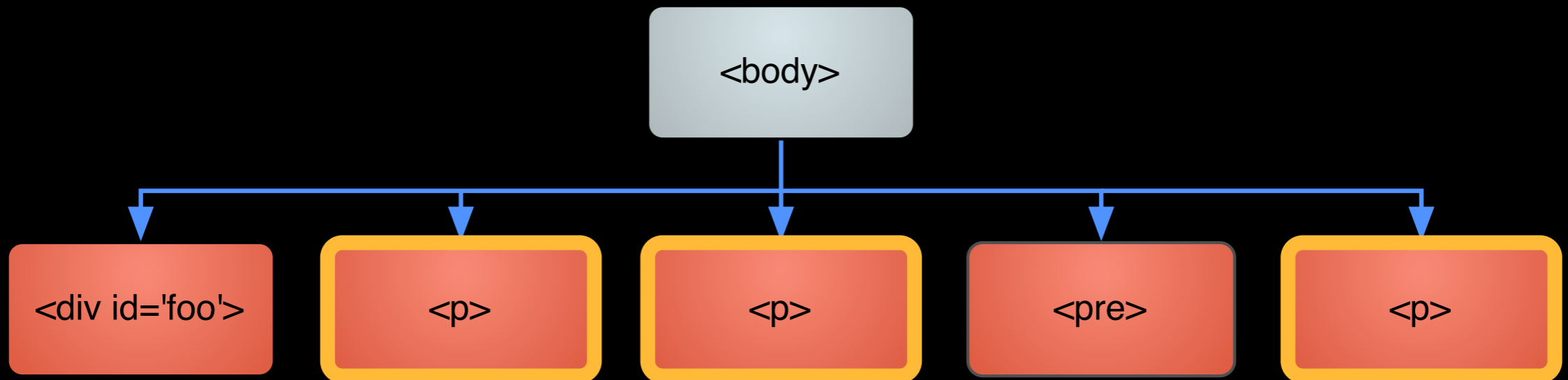
DOM



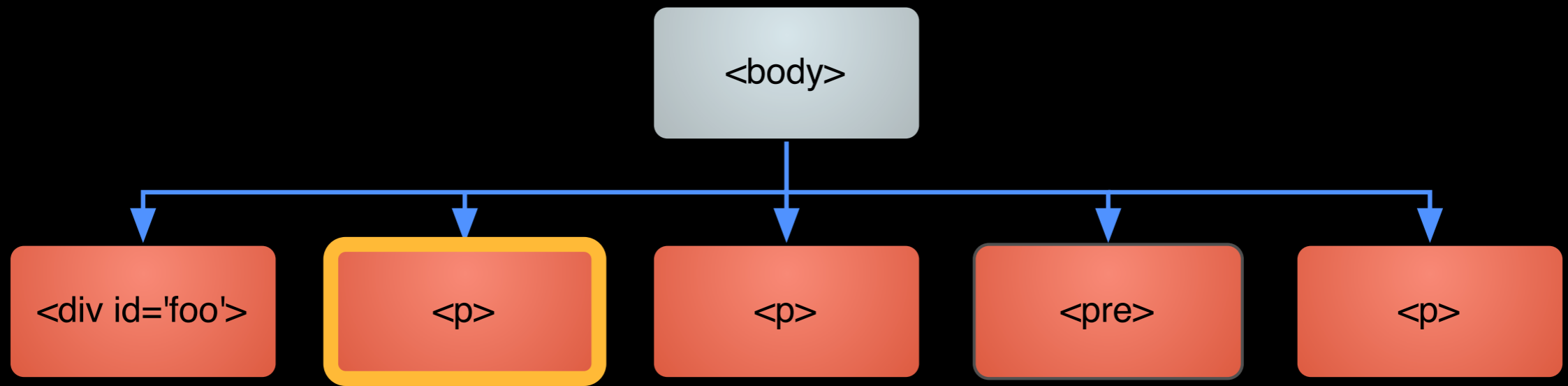
# \$("div").NEXT()



`$("div").nextAll("p")`

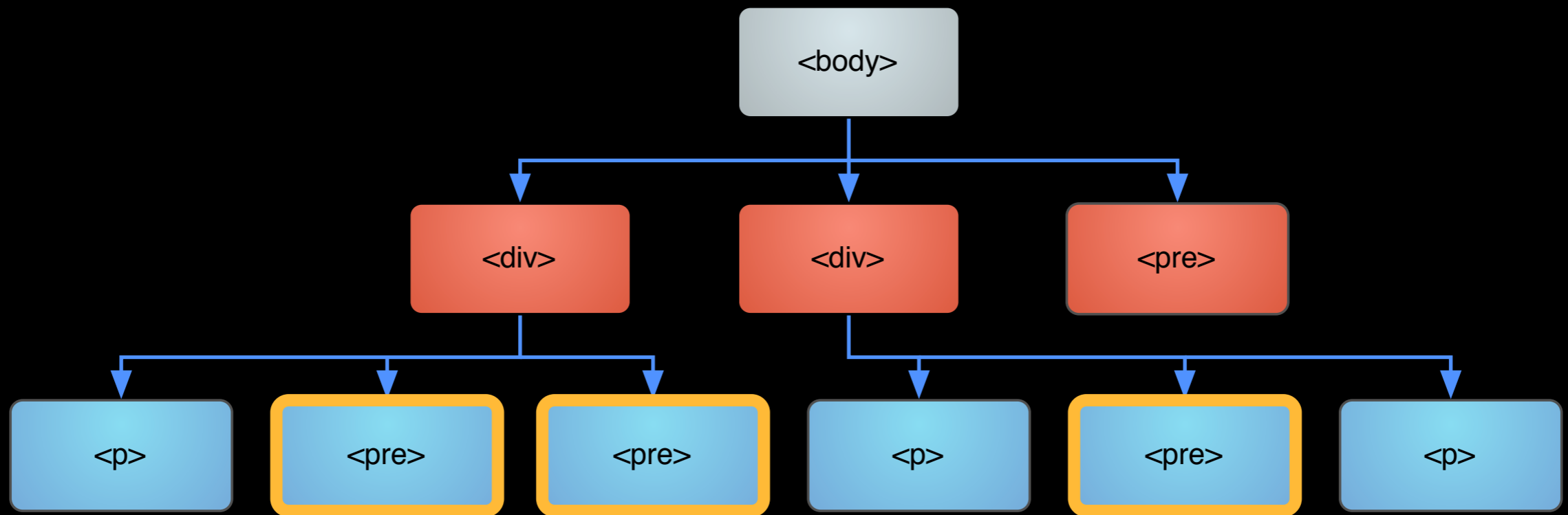


\$("DIV").NEXTALL(  
"P:FIRST")

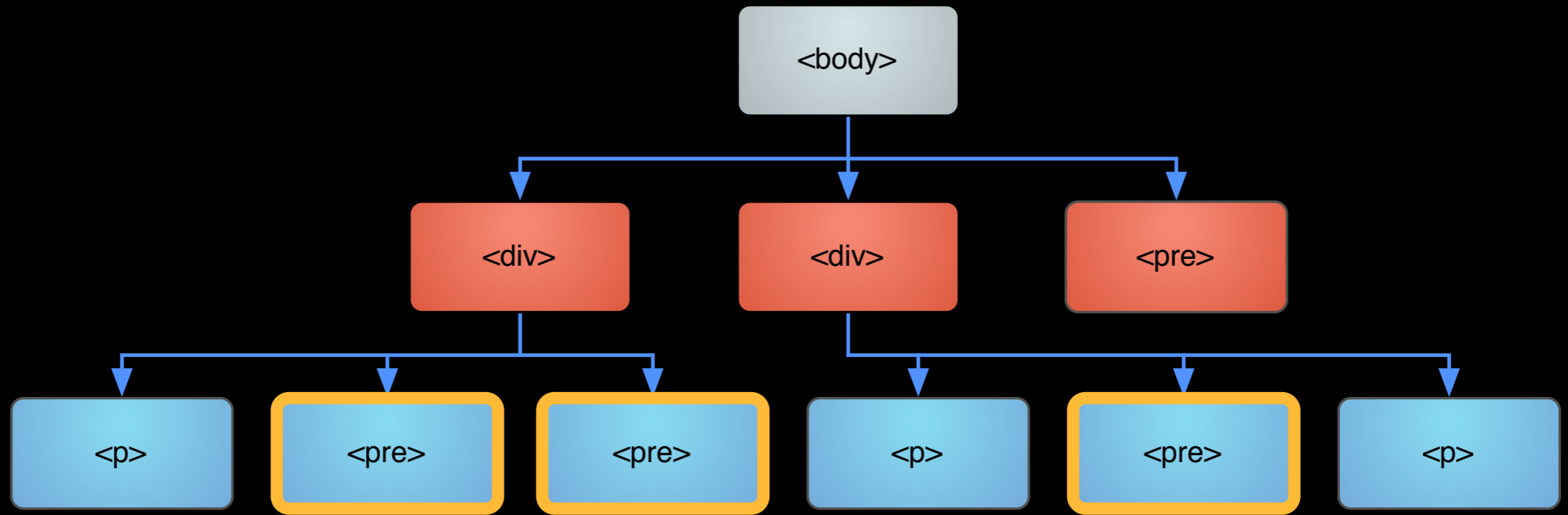


FINO

\$("DIV PRE")



\$("DIV").find("PRE")



# FILTER

```
$(“div:contains(some text)”)
```

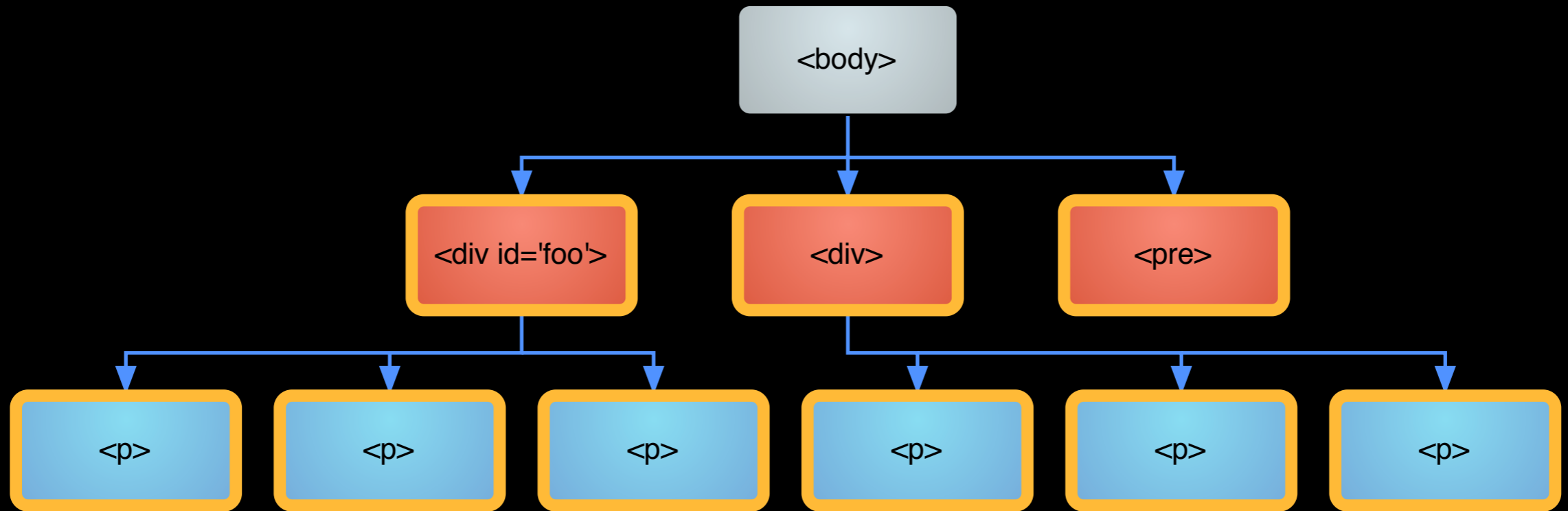
```
$(“div”).filter(  
    “:contains(some text)”)
```

# FILTER

```
$(“div”).filter(function() {  
    return $(this).text()  
        .match(“some text”)  
})
```

ANDSELFO

# DOM PARENTO AND SELF



# Getter ATTRIBUTES

*Getter* →  
\$("div").attr("id")

\$("div").attr("id", "hello")

Setter ↗

# ATTRIBUTES

```
$(“div”).attr(“id”,  
  function() { return this.name })
```

```
$(“div”).attr(  
  {id: “foo”, name: “bar”})
```

OTHER

Getter



```
$("div").html()
```

```
$("div").html("<p>Hello</p>")
```

Setter



*Getter*

OTHER

`$(“div”).text()`

`$(“div”).text(“Hello”)`

*Setter*

OTHER

Getter



```
$("div").val()
```

```
$("div").val("Hello")
```

Setter



NOTICING A  
PATTERN?

# 5 PARTS OF JQUERY

DOM

EVENTS

EFFECTS

AJAX

PLUGINS

# BIND

```
$(“div”).bind(“click”, function()  
    { ... })
```

Alias: 

```
$(“div”).click(function()  
    { ... })
```

“THIS”

REFERS TO THE ELEMENT BOUND



```
$("div").click(function(e) { ... })
```

# CORRECTED EVENT OBJECT

Property	Correction
target	The element that triggered the event (event delegation)
relatedTarget	The element that the mouse is moving in (or out) of
pageX/Y	The mouse cursor relative to the document
which	mouse: 1 (left) 2 (middle) 3 (right) keypress: The ASCII value of the text input
metaKey	Control on Windows and Apple on OSX

# TRIGGER

```
$(“div”).trigger(“click”)
```

Alias: `$(“div”).click()`

# TRIGGERHANDLER

DOESN'T TRIGGER THE BROWSER'S DEFAULT ACTIONS

# CUSTOM EVENTS

```
$(“div”).bind(“myEvent”, function()  
    { ... })
```

```
$(“div”).trigger(“myEvent”)
```

# NAMESPACES

```
    $("div")  
.bind("activate.autocomplete",  
    function() { ... })
```

```
    $("div").trigger("activate")
```

# 5 PARTS OF JQUERY

DOM

EVENTS

EFFECTS

AJAX

PLUGINS

# MAKE EASY THINGS EASY

```
$(“div”).load(“some_url”);
```

```
$(“div”).load(“some_url”, {data:  
“foo”},
```

IT'S EASY TO DO IT  
RIGHT

```
$.getJSON("some_url",  
function(json) { ... })
```

```
$.getJSON("some_url", {data: "foo"},  
function(json) { ... })
```

# IT'S CONSISTENT

```
$.get("some_url",  
      function(text) { ... })
```

```
$.post("some_url", {data: "foo"},  
       function(text) { ... })
```

# AND POWERFUL

## \$.AJAX OPTIONS

- async
- beforeSend
- cache
- complete
- contentType
- data
- dataType
- error
- global
- ifModified
- jsonp
- processData
- success
- timeout
- type

# AND POWERFUL

## GLOBAL AJAX SETTINGS

```
/* No Ajax requests exist, and one starts */
$("div.progress").ajaxStart(function() { $
(this).show() });

/* The last Ajax request stops */
$("div.progress").ajaxStop(function() { $
(this).hide() });

/* Any Ajax request is sent */
$("p").ajaxSend(function() { ... });

/* Any Ajax request completes (success or failure) */
$("div").ajaxComplete(function() { ... });

/* Any Ajax request errors out */
$("div").ajaxError(function() { ... });

/* Any Ajax request succeeds */
$("div").ajaxSuccess(function() { ... });
```

# 5 PARTS OF JQUERY

DOM

EVENTS

EFFECTS

AJAX

PLUGINS

METADATA

# METADATA

```
<input class="autocomplete"  
  metadata="{  
    cluster_id: 1,  
    get: '/customers/from_salesforce'  
    fields: {id:  
      'details[salesforce_id]'}  
  }" />
```


# IN RAILS

```
<%= autocomplete(  
  "Search for Salesforce Customer", {  
    :cluster_id => @cluster.id,  
    :get => "/customers/from_salesforce",  
    :fields => { :id =>  
      'details[salesforce_id]' }  
  }  
) %>
```

LIVEQUERY

# REPLACES EACH

```
$("input.autocomplete").each(function() { metadata
  var match = $(this).metadata().match || "name";
  $(this).autocomplete({
    ajax: $(this).metadata().get,
    insertText: function(obj) { return obj[match]; },
    match: function(typed) {
      var regex = new RegExp(typed, "i");
      return this[match || "name"]
        .toString().match(regex); },
    template: function(obj) {
      return "<li>" + obj[match || "name"] + "</li>";
    }
  })
});
```



# REPLACES EACH

```
<%= autocomplete(  
  "Search for Salesforce Customer", {  
    :cluster_id => @cluster.id,  
    :get => "/customers/from_salesforce",  
    :fields => { :id =>  
      'details[salesforce_id]'  
    }  
  }  
) %>
```

*reminder*

*ta*

});

# REPLACES EACH

*← catch that?*

```
$("#input.autocomplete").livequery(function() {  
  var match = $(this).metadata().match || "name";  
  $(this).autocomplete({  
    ajax: $(this).metadata().get,  
    insertText: function(obj) { return obj[match]; },  
    match: function(typed) {  
      var regex = new RegExp(typed, "i");  
      return this[match || "name"]  
        .toString().match(regex); },  
    template: function(obj) {  
      return "<li>" + obj[match || "name"] + "</li>";  
    }  
  })  
});
```

# REPLACES BIND

```
$("#a.edit_autocomplete").bind("click",  
  function() {  
    $(this).parent().hide()  
    .prev().show()  
    .find("input").focus();  
    return false;  
  });
```

# REPLACES BIND

*← catch that?*

```
$("#a.edit_autocomplete").livequery("click",  
function() {  
    $(this).parent().hide()  
    .prev().show()  
    .find("input").focus();  
    return false;  
});
```

# SEAMLESS INTERACTION

# AJAX

```
$("#input.autocomplete").livequery(function() {
    $(this).bind("activate.autocomplete", function(e, d) {
        var d;
        var se;
        var fi;
        if(meta)
            for
                $.post(meta.put, data, function(json) {
                    $(self).val("");
                    $(self).parent().hide().after("<h5>" + json[field] +
                        " <a href='#' class='edit_autocomplete'>(Edit)</a></h5>");
                }, "json");
            } else {
                for(prop in fields)
                    $(":input[name=\"" + fields[prop] + "\"").val(d[prop]);
            }
        });
    });
});
```

`var self = this;`

# AJAX

```
$("#input.autocomplete").livequery(function() {
    $(this).bind("activate.autocomplete", function(e, d) {
        var data = {}, self = this, meta = $(this).metadata()
        var self = this;
        var fields = meta.fields || meta.field;
        if(meta.type == "json") {
            for(prop in d) data[prop] = d[prop];
            $.ajax({
                url: meta.url,
                data: data,
                success: function(json) {
                    $(self).val("");
                    $(self).parent().hide().after("<h5>" + json[field] +
                        " <a href='#' class='edit_autocomplete'>(Edit)</a></h5>");
                }, "json");
            } else {
                for(prop in fields)
                    $("input[name=\"" + fields[prop] + "\"").val(d[prop]);
            }
        });
    });
});
```

# AJAX

```
$("#input.autocomplete").livequery(function() {
    $(this).bind("activate.autocomplete", function(e, d) {
        var data = {}, self = this, meta = $(this).metadata()
        var self = this;
        var fields = meta.fields, field = meta.field;
        if(meta.put) {
            for(prop in fields) data[fields[prop]] = d[prop];

            $.post(meta.put,
                $(self).val(),
                $(self).parent(),
                " <a href='#'
            }, "json");
        } else {
            for(prop in fields)
                $("input[name=\"" + fields[prop] + "\"").val(d[prop]);
        }
    });
});
```

function(json)


# IN RAILS

```
class CustomersController
  def update
    @customer = Customer.find(params[:id])
    @customer.update(params[:details])
    respond_to do |format|
      format.html
      format.json :json => @customer
    end
  end
end
```

*← send JSON*

# IN MERB

```
class Customers
  def update(id, details)
    @customer = Customer.get!(id)
    @customer.update(details)
    display @customer
  end
end
```

 *send JSON*

# AJAX

*receive JSON*



```
$("#input.autocomplete").livequery(function() {  
    $(this).bind("activate.autocomplete", function(e, d) {  
        var data = {}, self = this, meta = $(this).metadata()
```

```
        $.post(meta.put, data, function(json) {  
            $(self).val("");  
            $(self).parent().hide().after("<h5>" +  
                json[field] +  
                " <a href='#' class='edit_autocomplete'> " +  
                "(Edit)</a></h5>");  
        }, "json");
```

```
    }  
});  
});
```

WHO'S USING  
JQUERY?

# WHO, YOU ASK?

- Google
- Dell
- NBC
- CBS
- MSNBC
- Bank of America
- BBC
- Reuters
- Digg
- Business Week
- Newsweek
- Amazon
- Intel
- Oracle
- Cisco
- Slashdot
- Technorati
- Sourceforge

# NOT ENOUGH?

- Intuit
- American Eagle
- Salesforce
- Newsgator
- Boston Globe
- New York Post
- Miami Herald
- The Food Network
- The Onion
- Feedburner
- WB Records
- Def Jam Records
- AOL
- Classmates.com
- Fandango
- Pandora
- Vodafone
- Ars Technica

# YOU SAY YOU WANT MORE?

- Linux.com
- Super Smash Bros.
- Barack Obama
- Joost
- iFilm.com
- Mozilla
- Wordpress
- PEAR
- Trac
- Zend
- Hackety Hack
- Joomla
- Age of Empires 3
- myYearbook.com
- REI
- Poker Room
- isoHunt
- Ask a Ninja

GET A COMPLETE LIST  
AT

[HTTP://DOCS.JQUERY.COM/SITES\\_USING\\_JQUERY](http://docs.jquery.com/sites_using_jquery)

THIS IS ONLY THE  
BEGINNING

THANK YOU.