

Intro To jQuery

Stefan Penner

Stefan Penner

- Dev + Usability guy.
- Developer @ Innovatis Inc.
- Skullspace Supporter

Contact Stuff

iamstef.net

github.com/iamstef

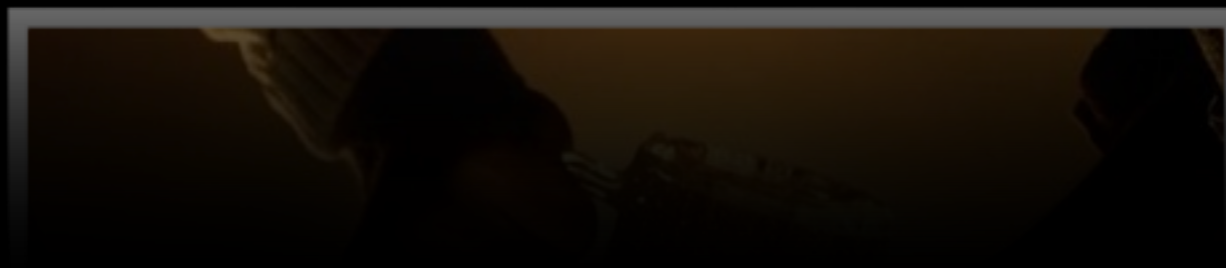
IRC: iamstef

twitter: @stefanpenner

PLEASE ASK QUESTIONS



Or be Left Behind



JavaScript development is growing more and more important, and with industry giants such as Microsoft? jumping on the jQuery bandwagon it's time to make the jump yourself. So what is jQuery, what can it do for you, and why should you use it? This talk will provide an introduction to JavaScript and a foundation for building dynamic, jQuery-powered websites. Topics include: DOM traversing, event handling, ninja hunting, animating, ajax interactions, and cross browser caveats. By the end the participants will have a well rounded foundation, enabling them to start building dynamic web sites.

JavaScript development is growing more

and more important, and with industry giants such as Microsoft jumping **on** the

jQuery bandwagon it's time to make the jump yourself. So what is jQuery, what can it do for you, and why should you use it? This talk will provide an introduction to

JavaScript and a foundation for building dynamic, jQuery-**powered**

we **b** **i** **s** **i** **t** **e**s. Topics include: DOM traversing, event handling, **ninja** hunting,

animating, ajax interaction **S**, and cross browser caveats. By the end the participants will have a well rounded foundation, enabling them to start building dynamic web sites.

JavaScript development on jQuery powered by ninja'S



NINJAS

They're everywhere.



NINJAS

There are four in this picture.

JavaScript Basics

```
var i = 1,  
    x = {},  
    y = [],  
    z = function(){  
        return 5;  
    };
```

```
function a(){  
    return 4;  
}
```

Control

Control

```
if(x === 1){  
  // ...  
}else if(y === 1){  
  // ...  
}else{  
  // ...  
}
```

Control

```
if(x === 1){  
  // ...  
}else if(y === 1){  
  // ...  
}else{  
  // ...  
}
```

```
(x === i) ? true : false;
```

Control

```
if(x === 1){  
  // ...  
}else if(y === 1){  
  // ...  
}else{  
  // ...  
}
```

```
(x === i) ? true : false;
```

```
switch(n)  
{  
  case 1:  
    execute code block 1  
    break;  
  case 2:  
    execute code block 2  
    break;  
  default:  
    // code to be executed if n is  
    different from case 1 and 2  
}
```

Hash/Object Literal

Hash/Object Literal

```
var theHash = {  
  key: 2,  
  something: 1,  
  somethingElse: true,  
  another: function(){  
    return "something";  
  }  
}
```

Hash/Object Literal

```
var theHash = {  
  key: 2,  
  something: 1,  
  somethingElse: true,  
  another: function(){  
    return "something";  
  }  
}
```

```
theHash.key  
> 2
```

Hash/Object Literal

```
var theHash = {  
  key: 2,  
  something: 1,  
  somethingElse: true,  
  another: function(){  
    return "something";  
  }  
}
```

```
theHash.key
```

```
> 2
```

```
theHash["key"]
```

```
> 2
```

Hash/Object Literal

```
var theHash = {  
  key: 2,  
  something: 1,  
  somethingElse: true,  
  another: function(){  
    return "something";  
  }  
}
```

```
theHash.key
```

```
> 2
```

```
theHash['key']
```

```
> 2
```

```
theHash['key'] = 5
```

```
> 5
```

Hash/Object Literal

```
var theHash = {  
  key: 2,  
  something: 1,  
  somethingElse: true,  
  another: function(){  
    return "something";  
  }  
}
```

```
theHash.key
```

```
> 2
```

```
theHash['key']
```

```
> 2
```

```
theHash['key'] = 5
```

```
> 5
```

```
theHash['key']
```

```
> 5
```

Hash/Object Literal

```
var theHash = {  
  key: 2,  
  something: 1,  
  somethingElse: true,  
  another: function(){  
    return "something";  
  }  
}
```

```
theHash.key
```

```
> 2
```

```
theHash['key']
```

```
> 2
```

```
theHash['key'] = 5
```

```
> 5
```

```
theHash['key']
```

```
> 5
```

```
for( var key in theHash){  
  console.log(theHash[key]);  
}
```

Array

// slower in some browsers

// faster in some browsers

Array

```
var theArray = [],  
    anotherArray = new Array(), //apparently slower  
    noneEmptyArray = ["1",2,"5"];
```

// slower in some browsers

// faster in some browsers

Array

```
var theArray = [],  
    anotherArray = new Array(), //apparently slower  
    noneEmptyArray = ["1",2,"5"];
```

// slower in some browsers

```
for(var i = 0; i < theArray.length; i++){  
    // code  
}
```

// faster in some browsers

Array

```
var theArray = [],  
    anotherArray = new Array(), //apparently slower  
    noneEmptyArray = ["1",2,"5"];
```

// slower in some browsers

```
for(var i = 0; i < theArray.length; i++){  
    // code  
}
```

// faster in some browsers

```
for(var i = 0, max = theArray.length; i < max; i++){  
    // code  
}
```

WTF?

WTF?

— ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ —

~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ —

~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ —

~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — []

WTF?

— ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ —

~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ —

~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ —

~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — ~ — []

> 42



jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. **jQuery is designed to change the way that you write JavaScript.**



jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. **jQuery is designed to change the way that you write JavaScript.**

Inventor: John Resig

Invented: 2005

Released: 2006

Reasoning: Consistent Easy API to the DOM.

Lots of Resources to Learn

But...

Best way to learn,
is to build.

lets make our Own
jQuery

lets make our Own
jQuery

sorta..

Simple jQuery Example

```
$( "#neat" ).show( "slow" );
```

Deeper

```
$("#neat").show("slow");
```

Deeper

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

Deeper

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

```
typeof (  
> SyntaxError: Unexpected token }
```

Deeper

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

```
typeof (  
> SyntaxError: Unexpected token }
```

```
typeof "#neat"  
> "string"
```

Deeper

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

```
typeof (  
> SyntaxError: Unexpected token }
```

```
typeof "#neat"  
> "string"
```

```
typeof $("#neat").show  
> "function"
```

Deeper

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

```
typeof (  
> SyntaxError: Unexpected token }
```

```
typeof "#neat"  
> "string"
```

```
typeof $("#neat").show  
> "function"
```

```
typeof $("#neat").show("slow")  
> "object"
```

even Deeper

```
$("#neat").show("slow");
```

even Deeper

```
$("#neat").show("slow");
```

```
<function>(<string>).<function>(<string>)
```

Step One

we clearly need a cool funky Web 2.0 name

Step One

we clearly need a cool funky Web 2.0 name

\$ is jQuery

_ is Underscore

Step One

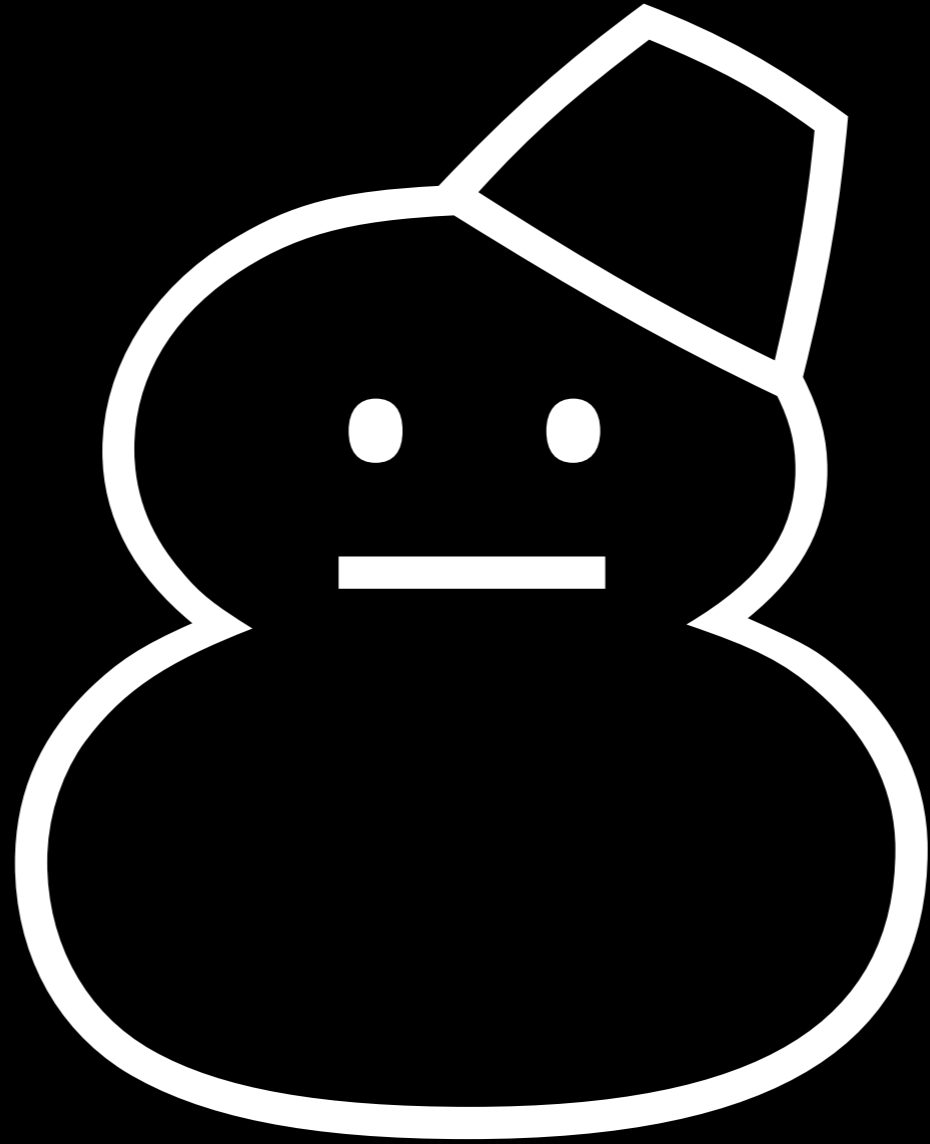
we clearly need a cool funky Web 2.0 name

\$ is jQuery

_ is Underscore

 is free.

☃



☃

```
$ ( "#neat" ) . show ( "slow" );  
Ⓜ ( "#neat" ) . show ( "slow" );
```

Holy crap, not only would it be a pain to type,
but it would be awesome!

```
> Ⓜ  
SyntaxError: Unexpected token ILLEGAL  
> var Ⓜ = 1;  
SyntaxError: Unexpected token ILLEGAL  
> var Ⓜ = {};  
SyntaxError: Unexpected token ILLEGAL  
> eval("var Ⓜ = {};  
SyntaxError: Unexpected token ILLEGAL
```

In memory of

R.I.P.

☃

Lets call our library **Snowman, S** for short.

In memory of



☃

Lets call our library **Snowman**, **S** for short.

Lets start

```
$("#neat").show("slow");
```

Lets start

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

Lets start

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

```
typeof (  
> SyntaxError: Unexpected token }
```

Lets start

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

```
typeof (  
> SyntaxError: Unexpected token }
```

```
typeof "#neat"  
> "string"
```

Lets start

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

```
typeof (  
> SyntaxError: Unexpected token }
```

```
typeof "#neat"  
> "string"
```

```
typeof $("#neat").show  
> "function"
```

Lets start

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

```
typeof (  
> SyntaxError: Unexpected token }
```

```
typeof "#neat"  
> "string"
```

```
typeof $("#neat").show  
> "function"
```

```
typeof $("#neat").show("slow")  
> "object"
```

Lets start

```
$("#neat").show("slow");
```

```
typeof $  
> "function"
```

```
typeof (  
> SyntaxError: Unexpected token }
```

```
typeof "#neat"  
> "string"
```

```
typeof $("#neat").show  
> "function"
```

```
typeof $("#neat").show("slow")  
> "object"
```

```
<function>(<string>).<function>(<string>)
```

so we need to make a function, that consumes a string, and returns something which is an object, which has methods on it!

What is function?

// declaration, since it's part of a Program (some funky issues)

```
function foo(){}
```

// expression, since it's part of an AssignmentExpression

```
var bar = function foo(){};
```

// expression, since it's part of an AssignmentExpression

```
var bar = function(){};
```

// expression, since it's part of a NewExpression

```
new function bar(){};
```

```
(function(){
```

```
  // declaration, since it's part of a FunctionBody
```

```
  function bar(){};
```

```
})();
```

<cite> <http://kangax.github.com/nfe/></cite>

What is a Closure?

Wrap our own library (jquery does it like this)

```
var myLib = (function(){  
  // private  
  var apple = 2,  
      herp = function(){  
    return 'derp';  
  }  
  
  // public  
  return {  
    apples: 'oranges',  
    five: function(){  
      return herp();  
    }  
  }  
})();
```

```
myLib.apples  
> 'oranges'  
myLib.apple  
> undefined  
myLib.five()  
> 'derp'
```


Ok Lets Start.

Simple Version

```
var Snowman = function(selector){  
  // our lib  
}
```

Proper Version

```
var Snowman = (function(){  
  // our private code  
  return function(selector){  
    // our public api.  
  }  
})();
```

Our Method Should return an “object”

```
var Snowman = (function(){  
  // our private code  
  return function(selector){  
  
  }  
})();
```

Our Method Should return an “object”

```
typeof 1  
> "number"
```

```
typeof Object  
> "function"
```

```
WTF?  
> typeof {}  
> "object"
```

```
typeof []  
> "object"
```

```
var Snowman = (function(){  
  // our private code  
  return function(selector){  
  
  }  
})();
```

Our Method Should return an “object”

```
typeof I  
> "number"
```

```
typeof Object  
> "function"
```

```
WTF?  
> typeof {}  
> "object"
```

```
typeof []  
> "object"
```

```
var Snowman = (function(){  
  // our private code  
  return function(selector){  
    return {};  
  }  
})();
```

Our Method Should return an “object”

```
typeof 1  
> "number"
```

```
typeof Object  
> "function"
```

```
WTF?  
> typeof {}  
> "object"
```

```
typeof []  
> "object"
```

```
var Snowman = (function(){  
  // our private code  
  return function(selector){  
    return {};  
  }  
})();
```

```
Snowman("hi")  
> {}
```

```
typeof Snowman("hi")  
> "object"
```

Next step. that object needs the method hide on it.

```
var Snowman = (function(){  
  // our private code  
  return function(selector){  
    return {  
      hide: function(){  
        console.log("hide hide");  
      }  
    };  
  }  
})();
```

Next step. that object needs the method hide on it.

```
var Snowman = (function(){  
  // our private code  
  return function(selector){  
    return {  
      hide: function(){  
        console.log("hide hide");  
      }  
    };  
  }  
})();
```

```
Snowman("hi").hide()  
> "hide hide"
```

Next step, our library needs to select the DOM element.

```
var Snowman = (function(){  
  
    return function(selector){  
        var element = document.getElementById(selector);  
        // snip..  
    }  
})();
```

Note this is a simple example, not truly cross browser, and only returns single elements, for real selector engines, look into sizzle, or NWMatcher.

Next step, lets hide and show are selected element.

```
var Snowman = (function(){  
    return function(selector){  
        var element = document.getElementById(selector);  
        return {  
            hide: function(){  
                element.style.display = "none";  
            },  
            show: function(){  
                element.style.display = "block";  
            }  
        };  
    };  
})();
```

Note this is a simple example, not truly cross browser, and only returns single elements, for real selector engines, look into sizzle, or NVMatcher.

Well, jQuery does some more stuff..

```
var Snowman = (function(){  
  
  return function(selector){  
    var element = document.getElementById(selector);  
    return {  
      selector: selector,  
      length: element ? 1 : 0,  
      addClass: function(newClass){  
        element.className += newClass;  
      },  
      removeClass: function(oldClass){  
        element.className = element.className.replace(new RegExp(oldClass,'gi'),'');  
      },  
      hide: function(){  
        element.style.display = "none";  
      },  
      show: function(){  
        element.style.display = "block";  
      }  
    };  
  };  
})();
```



LIMITATIONS

That's why you don't tell a retarded child they can be anything
they want when they grow up

jQuery has plugins,
So **WE** need plugins.

jQuery has plugins,
So **WE** need plugins.

```
$('.selector-of-some-kind').myAwesomePlugin();  
// so something off the jQuery name space, does something, to the  
collection selected by the css selector.. cool.
```

```
// Implementation  
jQuery.fn.myAwesomePlugin = function(options){  
  this.hide();  
};
```

but how?

but how?

prototypal inheritance

but how?
prototypal inheritance
eh?

Prototypal inheritance, simple example.

Prototypal inheritance, simple example.

```
function Person(){}
```

Prototypal inheritance, simple example.

```
function Person(){}  
  
Person.prototype.getName = function(){  
  return this.name;  
};
```

Prototypal inheritance, simple example.

```
function Person(){  
  
  Person.prototype.getName = function(){  
    return this.name;  
  };  
  
  function Me(){  
    this.name = "John Resig";  
  }  
}
```

Prototypal inheritance, simple example.

```
function Person(){  
  
  Person.prototype.getName = function(){  
    return this.name;  
  };  
  
  function Me(){  
    this.name = "John Resig";  
  }  
  
  Me.prototype = new Person();  
}
```

Prototypal inheritance, simple example.

```
function Person(){  
  
  Person.prototype.getName = function(){  
    return this.name;  
  };  
  
  function Me(){  
    this.name = "John Resig";  
  }  
  
  Me.prototype = new Person();  
  
  var me = new Me();
```

Prototypal inheritance, simple example.

```
function Person(){  
  
  Person.prototype.getName = function(){  
    return this.name;  
  };  
  
  function Me(){  
    this.name = "John Resig";  
  }  
  
  Me.prototype = new Person();  
  
  var me = new Me();  
  
  me.getName();  
  > "John Resig";
```

Prototypal inheritance, simple example.

```
function Person(){}

Person.prototype.getName = function(){
  return this.name;
};

function Me(){
  this.name = "John Resig";
}

Me.prototype = new Person();

var me = new Me();

me.getName();
> "John Resig";

My.prototype = {
  sayHi: function(){
    return this.getName() + " says hi";
  }
}
```

Prototypal inheritance, simple example.

```
function Person(){}

Person.prototype.getName = function(){
  return this.name;
};

function Me(){
  this.name = "John Resig";
}

Me.prototype = new Person();

var me = new Me();

me.getName();
> "John Resig";

My.prototype = {
  sayHi: function(){
    return this.getName() + " says hi";
  }
}

Me.prototype.getName = function(){
  return "Stefan Penner";
};
```

Prototypal inheritance, simple example.

```
function Person(){}

Person.prototype.getName = function(){
  return this.name;
};

function Me(){
  this.name = "John Resig";
}

Me.prototype = new Person();

var me = new Me();

me.getName();
> "John Resig";

My.prototype = {
  sayHi: function(){
    return this.getName() + " says hi";
  }
}

Me.prototype.getName = function(){
  return "Stefan Penner";
};

me.sayHi();
> "Stefan Penner says hi"
```

Prototype in jQuery?

Prototype in jQuery?

```
jQuery.fn = jQuery.fn.init = {  
  // all da methods  
};
```

Prototype in jQuery?

```
jQuery.fn = jQuery.fn.init = {  
  // all da methods  
};
```

So turns out jQuery.fn is almost just simply an alias to jQuery.prototype

and, the jQuery function actually instantiates, itself?

and, the jQuery function actually instantiates, itself?

```
var jQuery = function(selector,context){  
    return new jQuery.fn.init( selector, context, rootjQuery );  
}
```

and, the jQuery function actually instantiates, itself?

```
var jQuery = function(selector,context){  
    return new jQuery.fn.init( selector, context, rootjQuery );  
}
```

```
jQuery.fn = jQuery.prototype = {  
    constructor: jQuery,  
    init: function( selector, context, rootjQuery ) {  
        // snip... alot  
    } else {  
        return this.constructor( context ).find( selector );  
    }  
    // snip...  
},  
// snip..  
}
```

```
jQuery.fn.init.prototype = jQuery.fn;
```

and, the jQuery function actually instantiates, itself?

```
var jQuery = function(selector,context){  
    return new jQuery.fn.init( selector, context, rootjQuery );  
}
```

```
jQuery.fn = jQuery.prototype = {  
    constructor: jQuery,  
    init: function( selector, context, rootjQuery ) {  
        // snip... alot  
    } else {  
        return this.constructor( context ).find( selector );  
    }  
    // snip...  
},  
// snip..  
}
```

```
jQuery.fn.init.prototype = jQuery.fn;
```



Refactor

```
var Snowman = (function(){  
  
  return function(selector){  
    var element = document.getElementById(selector);  
    return {  
      selector: selector,  
      length: element ? 1 : 0,  
      addClass: function(newClass){  
        element.className += newClass;  
      },  
      removeClass: function(oldClass){  
        element.className = element.className.replace(new RegExp(oldClass,'gi'), "");  
      },  
      hide: function(){  
        element.style.display = "none";  
      },  
      show: function(){  
        element.style.display = "block";  
      }  
    };  
  };  
})();
```

Refactor

Refactor

```
var Snowman = (function(){
  var S = window.S = function(selector){
    return new S.fn.init(selector);
  }

  S.fn = S.prototype = {
    init: function(selector){
      this.selector = selector;
      this.element = document.getElementById(selector);
    },

    addClass: function(newClass){
      this.element.className += newClass;
    },

    removeClass: function(oldClass){
      this.element.className = this.element.className.replace(new RegExp(oldClass,'gi'), "");
    },

    hide: function(){
      this.element.style.display = "none";
    },

    show: function(){
      this.element.style.display = "block";
    }
  };

  S.fn.init.prototype = S.fn;

  return S;
})();
```

We now Have

- Basic DOM retrieval
- Clean API
- Show/Hide
- Plugin API

We still Need

- Chaining
- Events
- Animation
- ...

jQuery's Chaining

(use of monads/is a monad...)

In functional programming, a **monad** is a kind of abstract data type constructor used to represent computations (instead of data in the domain model). Monads allow the programmer to chain actions together to build a pipeline, in which each action is decorated with additional processing rules provided by the monad. Programs written in functional style can make use of monads to structure procedures that include sequenced operations,^{[1][2]} or to define some arbitrary control flows (like handling concurrency, continuations, side effects such as input/output, or exceptions).

- wikipedia

jQuery's Chaining

(use of monads/is a monad...)

In functional programming, a **monad** is a kind of abstract data type constructor used to represent computations (instead of data in the domain model). Monads allow the programmer to chain actions together to build a pipeline, in which each action is decorated with additional processing rules provided by the monad. Programs written in functional style can make use of monads to structure procedures that include sequenced operations,^{[1][2]} or to define some arbitrary control flows (like handling concurrency, continuations, side effects such as input/output, or exceptions).

- wikipedia

```
$( "#neat" ).show().addClass( "orange" );
```

jQuery's Chaining

(use of monads/is a monad...)

In functional programming, a **monad** is a kind of abstract data type constructor used to represent computations (instead of data in the domain model). Monads allow the programmer to chain actions together to build a pipeline, in which each action is decorated with additional processing rules provided by the monad. Programs written in functional style can make use of monads to structure procedures that include sequenced operations,^{[1][2]} or to define some arbitrary control flows (like handling concurrency, continuations, side effects such as input/output, or exceptions).

- wikipedia

```
$( "#neat" ).show().addClass( "orange" );  
<function>( <string> ).<monad>.<monad( <string> )>
```

Implement Chaining

Since the whole Snowman (jQuery) API, is on the Snowman prototype, simply passing the current object back after each manipulation/computation related provides this functionality.

```
addClass:function(newClass){
  this.element.className += newClass;
  return this;
},

removeClass: function(oldClass){
  this.element.className = this.element.className.replace(new RegExp(oldClass,'gi'), "");
  return this;
},

hide: function(){
  this.element.style.display = "none";
  return this;
},

show: function(){
  this.element.style.display = "block";
  return this;
}
```

Final Example

(real jQuery)

Simple Crossfade Gallery

The Foundation

HTML

- semantic markup

CSS

- style
- position
- color

JavaScript

- dynamic behaviour

HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Demo</title>
    <style> <!-- snip --!> </style>
    <script type="text/javascript" src="jquery.min.js"></script>
    <script type="text/javascript"> <!-- snip --!> </script>
  </head>
  <body>
    <div id="gallery">
      <div class="pane"></div>
      <div class="pane"></div>
      <div class="pane"></div>
      <div class="pane"></div>
    </div>
  </body>
</html>
```

CSS

```
#gallery{  
  width: 900px;  
  height: 300px;  
}
```

```
#gallery .pane{  
  display: none;  
  position: absolute;  
}
```

The JavaScript

The JavaScript

The API we want

The JavaScript

The API we want

```
$('#gallery').gallery(<options>);
```

The JavaScript

The API we want

```
$('#gallery').gallery(<options>);
```

The jQuery Plugin

The JavaScript

The API we want

```
$('#gallery').gallery(<options>);
```

The jQuery Plugin

```
$.fn.gallery = function(options){  
    var panes;  
  
    if(options && options.panes){  
        panes = $(options.panes);  
    }else{  
        panes = this.children();  
    }  
  
    this.data('gallery', new Gallery(panes));  
    return this;  
}
```

Logic

```
var Gallery = function(panes,duration){
```

Logic

```
};
```



```
var Gallery = function(panes,duration){  
  // setting initial state  
  var current = 0;  
  duration    = duration || 2000;  
  
  // looping anonymous function  
  (function(){
```

Logic

```
  })();
```

```
};
```


Logic

```
var Gallery = function(panes,duration){  
  // setting initial state  
  var current = 0;  
  duration    = duration || 2000;  
  
  // looping anonymous function  
  (function(){  
    var focus = panes.eq(current%(panes.length-1)),  
        parent = arguments.callee;  
  
    })();  
};
```

Logic

```
var Gallery = function(panes,duration){  
  // setting initial state  
  var current = 0;  
  duration    = duration || 2000;  
  
  // looping anonymous function  
  (function(){  
    var focus = panes.eq(current%(panes.length-1)),  
        parent = arguments.callee;  
  
    current++;  
  
  })();  
};
```

Logic

```
var Gallery = function(panes,duration){  
  // setting initial state  
  var current = 0;  
  duration    = duration || 2000;  
  
  // looping anonymous function  
  (function(){  
    var focus = panes.eq(current%(panes.length-1)),  
        parent = arguments.callee;  
  
    current++;  
  
    fadeIn(duration,function(){  
  
      }).  
  
    });  
  
  })();  
};
```

Logic

```
var Gallery = function(panes,duration){  
  // setting initial state  
  var current = 0;  
  duration    = duration || 2000;  
  
  // looping anonymous function  
  (function(){  
    var focus = panes.eq(current%(panes.length-1)),  
        parent = arguments.callee;  
  
    current++;  
    focus.css('z-index',9).  
      fadeIn(duration,function(){  
  
        }).  
  
    })();  
};
```

Logic

```
var Gallery = function(panes,duration){
  // setting initial state
  var current = 0;
  duration    = duration || 2000;

  // looping anonymous function
  (function(){
    var focus = panes.eq(current%(panes.length-1)),
        parent = arguments.callee;

    current++;
    focus.css('z-index',9).
      fadeIn(duration,function(){
        siblings().
          css('z-index',8);
      });
  })();
};
```

Logic

```
var Gallery = function(panes,duration){  
  // setting initial state  
  var current = 0;  
  duration    = duration || 2000;  
  
  // looping anonymous function  
  (function(){  
    var focus = panes.eq(current%(panes.length-1)),  
        parent = arguments.callee;  
  
    current++;  
    focus.css('z-index',9).  
      fadeIn(duration,function(){  
  
        setTimeout(parent,5000);  
      }).  
    siblings().  
      css('z-index',8);  
  
  })();  
};
```

Logic

```
var Gallery = function(panes,duration){
  // setting initial state
  var current = 0;
  duration    = duration || 2000;

  // looping anonymous function
  (function(){
    var focus = panes.eq(current%(panes.length-1)),
        parent = arguments.callee;

    current++;
    focus.css('z-index',9).
      fadeIn(duration,function(){
        $(this).
          siblings().
            hide();
            setTimeout(parent,5000);
          });
    siblings().
      css('z-index',8);

  })();
};
```

Live Demo!

So Much more.

Thank You

ASK QUESTIONS!