JQUERY

Performance

Websites need to be **responsive**
for the best user experience

There are **2 categories** of
optimizations which we can perform

There are 2 categories of optimizations which we can perform

Server requests

DOM rendering

Server requests

Set up servers on **multiple datacenters** so the request is served from the one closest to the user

Ensure data requested is in **small** chunks

Server requests

This class is focused on client side technologies so all of this is out of scope here

Performance

There are **2 categories** of optimizations which we can perform

Server requests

DOM rendering

Performance

DOM rendering

A fact of life is, DOM rendering is very very slow

Any HTML updates that you make should ideally not require multiple DOM changes

Performance

DOM rendering

A good developer will coalesce DOM updates and make them in one go

Performance

DOM rendering

Pretty much all the tips here will talk about ways to **minimize** DOM updates

Performance

There are **2 categories** of optimizations which we can perform

Server requests

DOM rendering

create and append elements

You have a **loop** to build up the element to add to the DOM

Complete the manipulation **before** you append to the DOM

Performance

create and append elements

```
var names = ['Janani', 'Swetha', 'Vitthal', 'Jitu'];
var list = '';
for (var i = 0; i < names.length; i++) {
    $('ul').append('<li>' + names[i] + '</li>');
}
```

Performance

create and append elements

```
var names = ['Janani', 'Swetha', 'Vitthal', 'Jitu'];
var list = '';
for (var i = 0; i < names.length; i++) {
    list += '<li>' + names[i] + '</li>';
}

$('ul').append(list);
```

Once you've built up the entire list append it to the DOM

Performance

editing an existing element

Do **not** manipulate an element's HTML while it is **in the DOM**

remove() or **detach()** it before performing edits!

Performance

editing an existing element

```
var names = ['Janani', 'Swetha', 'Vitthal', 'Jitu'];

var list = $('ul');
var parent = list.parent();

list.detach();

for (var i = 0; i < names.length; i++) {
    list.append('<li>' + names[i] + '</li>');
}

parent.append(list);
```

Detach the list from the DOM

Performance

editing an existing element

```
var names = ['Janani', 'Swetha', 'Vitthal', 'Jitu'];

var list = $('ul');
var parent = list.parent();

list.detach();

for (var i = 0; i < names.length; i++) {
    list.append('<li>' + names[i] + '</li>');
}

parent.append(list);
```

Now you can directly append to the list

Performance

editing an existing element

```
var names = ['Janani', 'Swetha', 'Vitthal', 'Jitu'];

var list = $('ul');
var parent = list.parent();

list.detach();

for (var i = 0; i < names.length; i++) {
    list.append('<li>' + names[i] + '</li>');
}

parent.append(list);
```

Add the list back once the changes are done

actions on elements

Ensure that elements exist in your selection before performing actions on them

jQuery does not check see if a selection has elements before it runs actions

actions on elements

```
$('#nonexistent').addClass('highlight');
```

If the element does not exist this results in wasted actions

Performance

actions on elements

```
var jqEl = $('#nonexistent').addClass('highlight');
if (jqEl.length) {
    jqEl.addClass('highlight');
}
```

This is better but it places a **heavy burden** on the developer

Performance

actions on elements

```
jQuery.fn.checkAndApply = function(fn) {
    this.length && fn.apply(this);
    return this;
};
```

```
$('#nonexistent').checkAndApply(function() {
    this.addClass('highlight');
});
```

Create a **wrapper** functions which checks for the existence of selected elements

Performance

actions on elements

```
jQuery.fn.checkAndApply = function(fn) {
    this.length && fn.apply(this);
    return this;
};
```

```
$('#nonexistent').checkAndApply(function() {
    this.addClass('highlight');
});
```

Use the wrapper when performing actions!

Optimize Selectors

jQuery selectors now use the browser's native selection mechanism

Avoid jQuery specific stuff which cannot leverage the native selection mechanism

Performance

Optimize Selectors

Avoid jQuery specific stuff which cannot leverage the native selection mechanism

```
$('#teacherstable tr:even');
$('#teacherstable tr:nth-child(odd)');
```

Performance

Optimize Selectors

Be less specific with your selectors,
do not specify the entire hierarchy

```
$('.republican ul.presidents li.governor');
```

Optimize Selectors

Be less specific with your selectors,
do not specify the entire hierarchy

```
$('.republican li.governor’);
```

Performance

Optimize Selectors

```
$('.republican li.governor');
```

Be specific on the right side of your selector

Performance

Optimize Selectors

```
$('.republican li.governor');
```

Less specific on the left

Optimize Selectors

```
$('#donald vicepresident');
```

id based selectors are generally optimized, definitely use them

JQUERY

Plugins

Plugins

Every method that you call on the jQuery object can be considered a **plugin**

.fadeOut()

.click()

.slideUp()

.hide()

Plugins

.fadeOut()
.click()
.slideUp()
.hide()

A plugin is simply a new method you use to **extend** the jQuery **prototype** object

$('div').makeCool()

$('div').flame()

$('div').shine()

$('div').hop()

Plugins

```
$('div').makeCool()
$('div').flame()
$('div').shine()
$('div').hop()
```

The whole idea of a plugin is to **act** on a set of jQuery elements

Plugins

```
$('div').makeCool()
$('div').flame()
$('div').shine()
$('div').hop()
```

You can add some behavior, change its look and feel, build complex interactions, anything!

Plugins

```
$('div').makeCool()
$('div').flame()
$('div').shine()
$('div').hop()
```

The jQuery community has a **wide variety of plugins**, look through those before you code up a new feature yourself

Plugins

wide variety of plugins

The quality and documentation of plugins may vary

Plugins

wide variety of plugins

The quality and documentation of plugins may vary

jQuery UI is maintained by the jQuery team so is obviously awesome

Plugins

wide variety of plugins

Before you use a plugin check when it was last updated and what kind of documentation and community support it has

Plugins

https://learn.jquery.com/plugins/finding-evaluating-plugins/

http://plugins.jquery.com/

EXAMPLE 45

SlickCarousel.html

SlickCarousel.html

Here is a really cool carousel plugin called **Slick**

We'll download, install and use this plugin

SlickCarousel.html

Example45DownloadSlick

SlickCarousel.html

Example45Demo

SlickCarousel.html

```
<title>Slick</title>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick.css"/>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick-theme.css"/>
<script src="../../lib/jquery.js"></script>
<script type="text/javascript" src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
<script type="text/javascript" src="../../slick-1.6.0/slick/slick.min.js"></script>
```

After you download and unzip a plugin we need to **reference** it in our code

SlickCarousel.html

```
<title>Slick</title>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick.css"/>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick-theme.css"/>
<script src="../../lib/jquery.js"></script>
<script type="text/javascript" src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
<script type="text/javascript" src="../../slick-1.6.0/slick/slick.min.js"></script>
```

Include the CSS required by Slick

SlickCarousel.html

```
<title>Slick</title>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick.css"/>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick-theme.css"/>
<script src="../../lib/jquery.js"></script>
<script type="text/javascript" src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
<script type="text/javascript" src="../../slick-1.6.0/slick/slick.min.js"></script>
```

The jQuery core components which we've used so far

SlickCarousel.html

```
<title>Slick</title>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick.css"/>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick-theme.css"/>
<script src="../../lib/jquery.js"></script>
<script type="text/javascript" src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
<script type="text/javascript" src="../../slick-1.6.0/slick/slick.min.js"></script>
```

The Slick Javascript code and jQuery migrate

SlickCarousel.html

```
<title>Slick</title>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick.css"/>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick-theme.css"/>
<script src="../../lib/jquery.js"></script>
<script type="text/javascript" src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
<script type="text/javascript" src="../../slick-1.6.0/slick/slick.min.js"></script>
```

The documentation on the Slick site indicates that both these are needed

SlickCarousel.html

```
<title>Slick</title>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick.css"/>
<link rel="stylesheet" type="text/css" href="../../slick-1.6.0/slick/slick-theme.css"/>
<script src="../../lib/jquery.js"></script>
<script type="text/javascript" src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
<script type="text/javascript" src="../../slick-1.6.0/slick/slick.min.js"></script>
```

When you use plugins make sure you look at **examples and demos** and reference all the files needed

SlickCarousel.html

```
<body>
<h3>
    Slick is a pretty awesome plugin
</h3>
<div class="dogs">
    <div class="dog-image">
        <img src="Moje.jpg"/>
    </div>
    <div class="dog-image">
        <img src="Oba.jpg"/>
    </div>
    <div class="dog-image">
        <img src="Moje.jpg"/>
    </div>
    <div class="dog-image">
        <img src="Oba.jpg"/>
    </div>
</div>
</body>
```

Any plugin should tell you how the HTML it acts on should be set up

SlickCarousel.html

```
<body>
<h3>
    Slick is a pretty awesome plugin
</h3>
<div class="dogs">
    <div class="dog-image">
        <img src="Moje.jpg"/>
    </div>
    <div class="dog-image">
        <img src="Oba.jpg"/>
    </div>
    <div class="dog-image">
        <img src="Moje.jpg"/>
    </div>
    <div class="dog-image">
        <img src="Oba.jpg"/>
    </div>
</div>
</body>
```

Slick acts on the **child** elements of the container on which we apply the plugin i.e. the **dogs **

SlickCarousel.html

```
<body>
<h3>
    Slick is a pretty awesome plugin
</h3>
<div class="dogs">
    <div class="dog-image">
        <img src="Moje.jpg"/>
    </div>
    <div class="dog-image">
        <img src="Oba.jpg"/>
    </div>
    <div class="dog-image">
        <img src="Moje.jpg"/>
    </div>
    <div class="dog-image">
        <img src="Oba.jpg"/>
    </div>
</div>
</body>
```

The elements to display in the carousel

SlickCarousel.html

```
$(document).ready(function () {
    $('.dogs').slick({
        infinite: true,
        slidesToShow: 3,
        slidesToScroll: 3
    });
});
```

It just takes one line of code with a configuration object to set things up!

SlickCarousel.html

```
$(document).ready(function () {
    $('.dogs').slick({
        infinite: true,
        slidesToShow: 3,
        slidesToScroll: 3
    });
});
```

We want **infinite** scroll, **see** only 3 slides at a time and **scroll** just 3 slides at a time

SlickCarousel.html

```
$(document).ready(function () {
    $('.dogs').slick({
        infinite: true,
        slidesToShow: 3,
        slidesToScroll: 3
    });
});
```

This is just the tip of the iceberg, Slick is very customizable and has a whole bunch of other options

SlickCarousel.html

```
$(document).ready(function () {
    $('.dogs').slick({
        infinite: true,
        slidesToShow: 3,
        slidesToScroll: 3
    });
});
```

Worth exploring!

JQUERY

Custom Plugins

It's super easy to write your own plugin for jQuery

You might want some functionality at different parts of your app and want to make it easy to add this behavior

Let's first understand how we can extend the jQuery prototype object to get our plugin to work

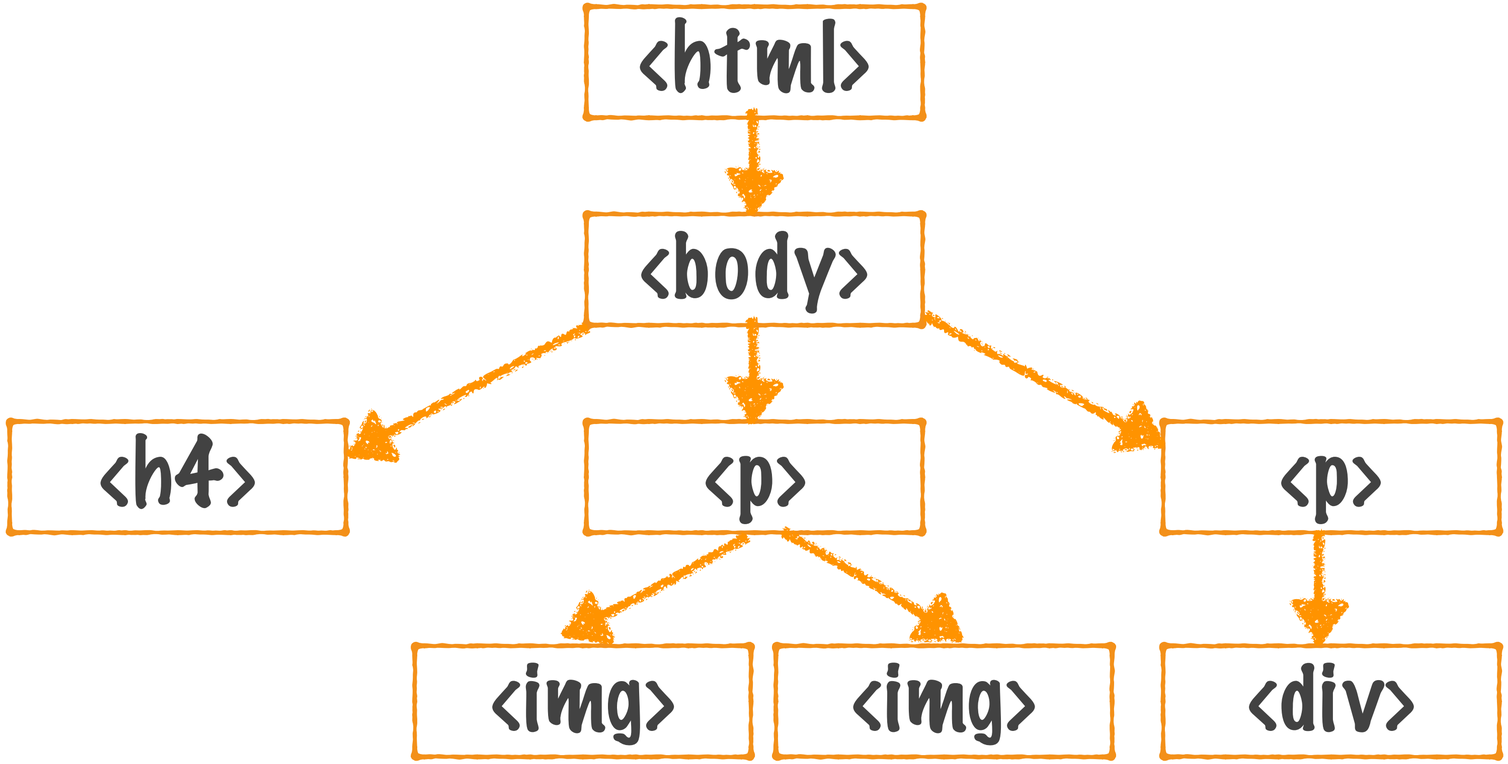
extend the jQuery prototype object

```
$('h3').addClass('red-color');
```

Using the $ function returns a jQuery object

Recap

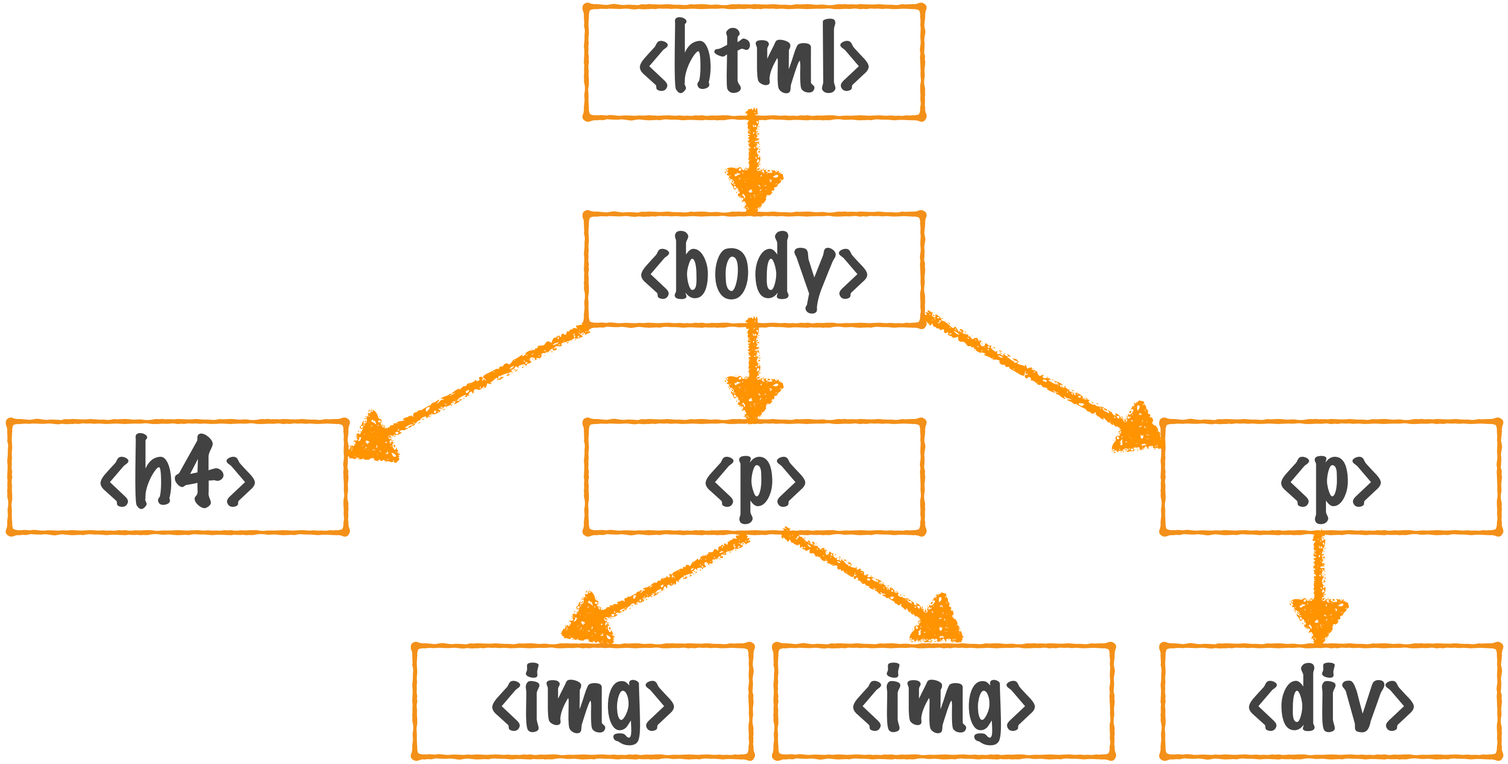
Selectors



What if you wanted to select some elements from the DOM?

Recap

Selectors

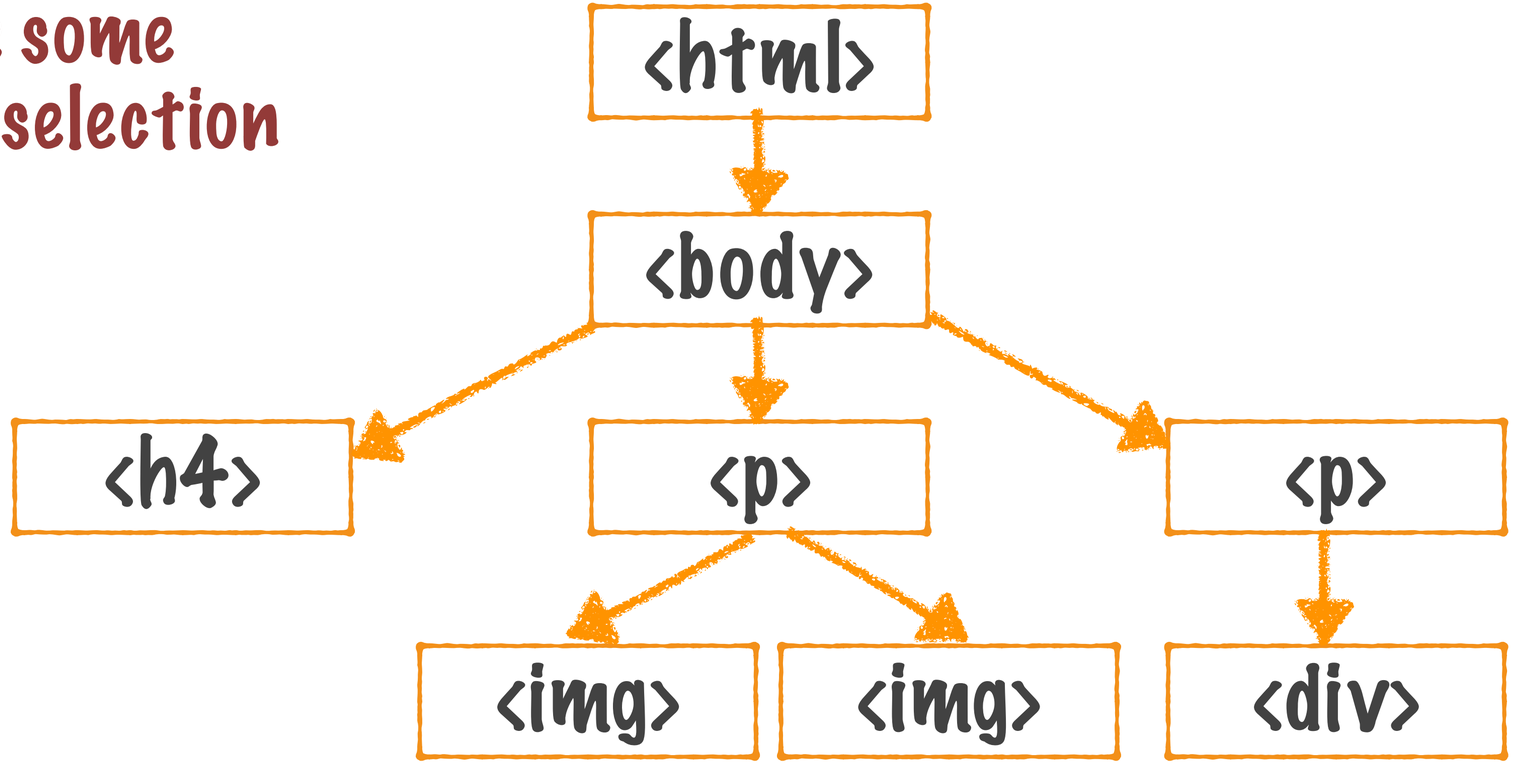


You have some criteria for selection

Recap

Selectors

You have some criteria for selection



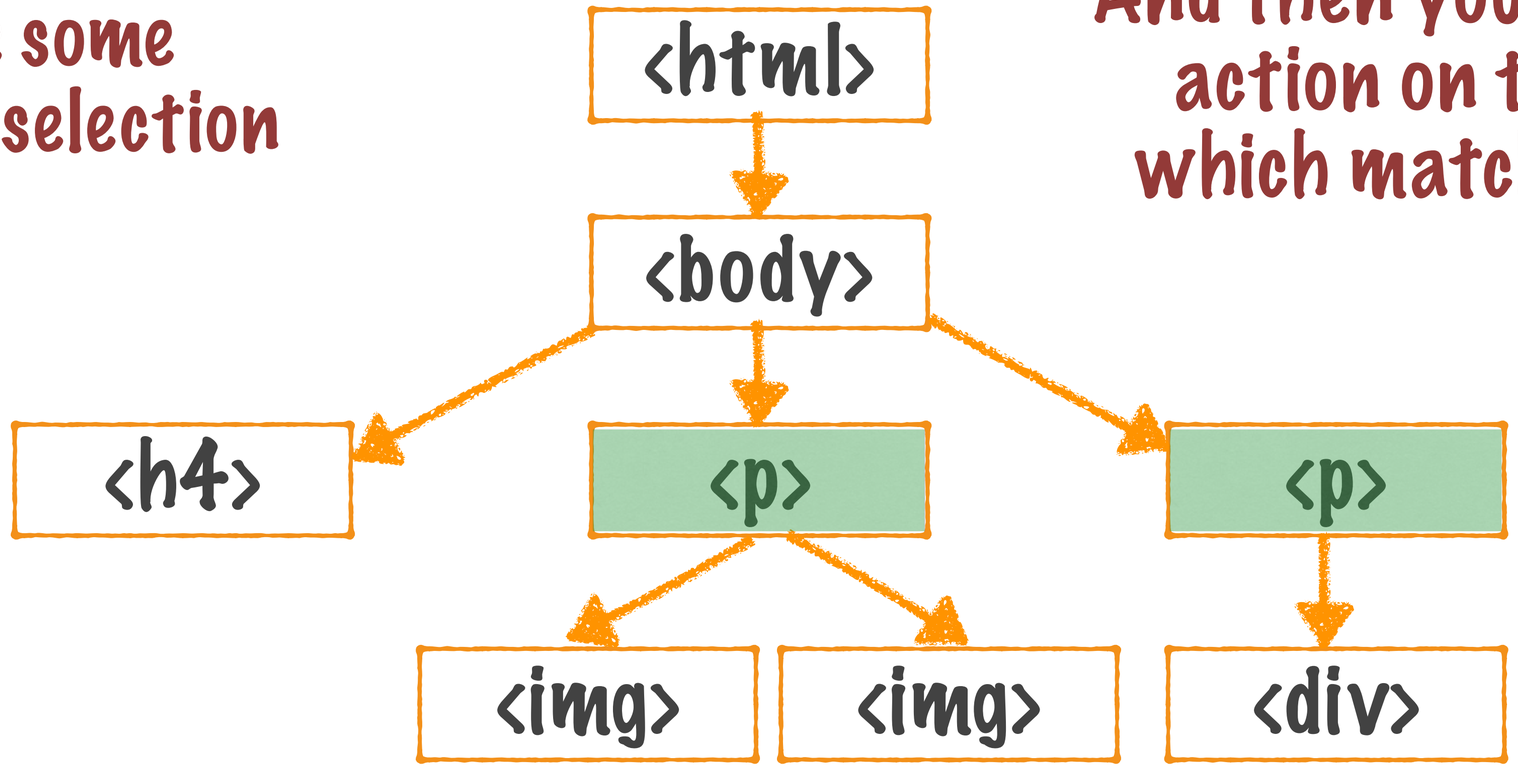
And then you perform some action on the elements which match the criteria

Recap

Selectors

You have some criteria for selection

And then you perform some action on the elements which match the criteria



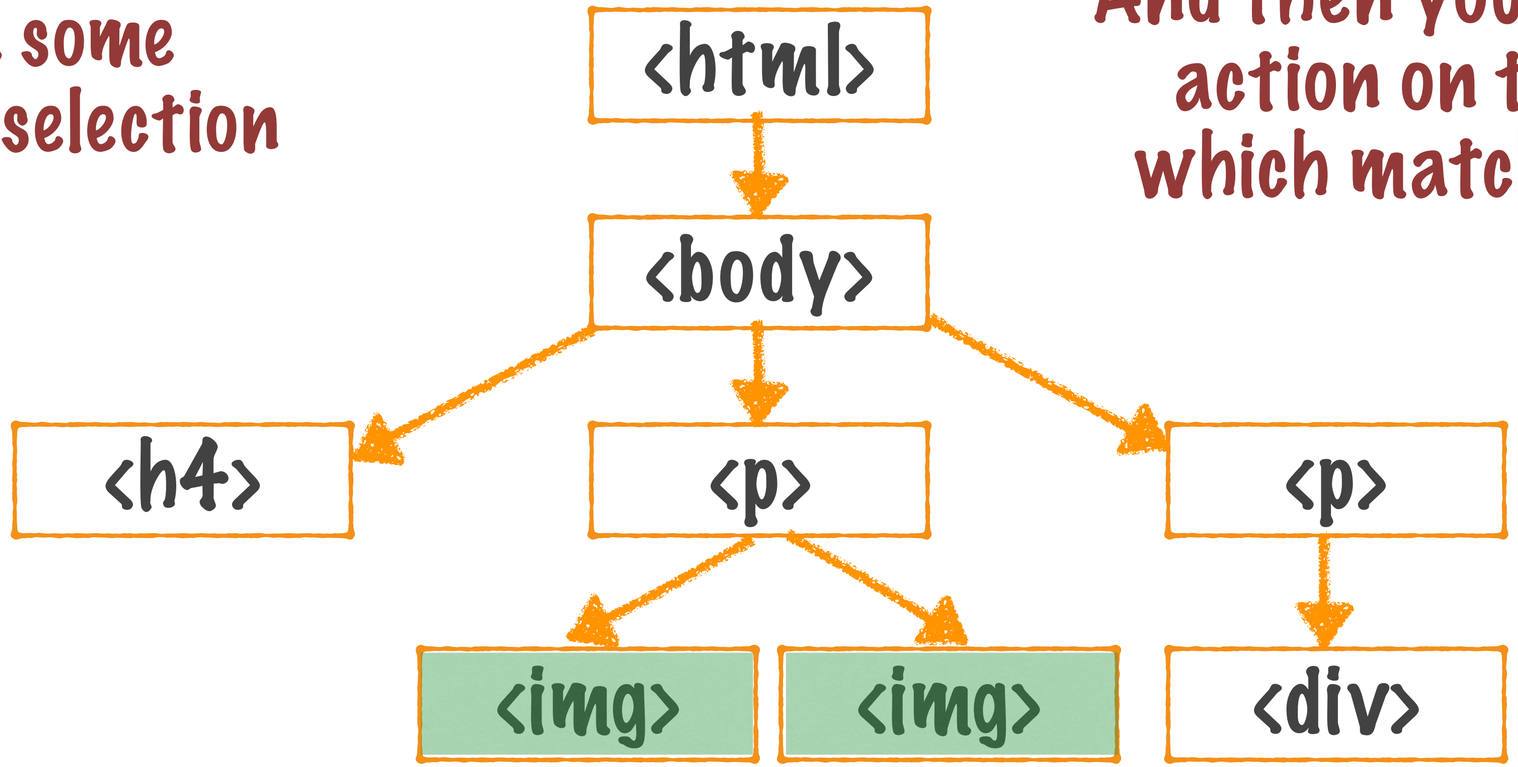
Select all the paragraphs in this document

Recap

Selectors

You have some criteria for selection

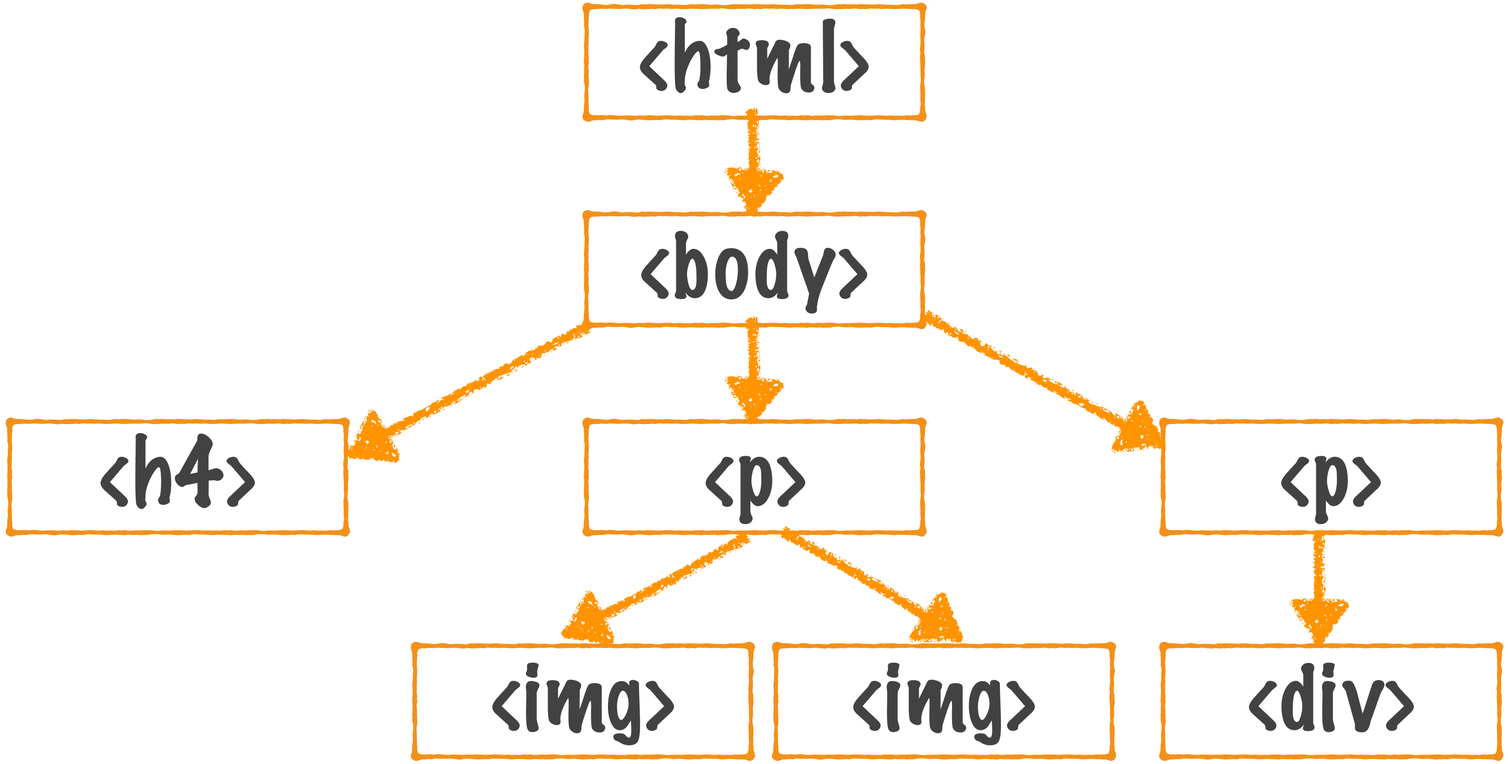
And then you perform some action on the elements which match the criteria



Select all images in this document

Recap

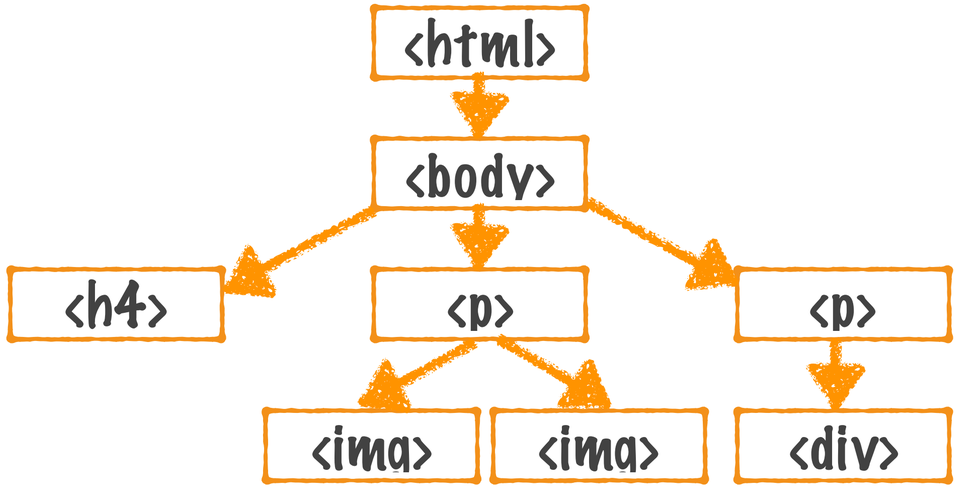
Selectors



jQuery makes selections, even very complicated ones super easy

Recap

Selectors



The $() function selects elements in the document

And wraps them in a jQuery object

Recap

Selectors

And wraps them in a
jQuery object

This jQuery object is not the native Javascript element

It is a jQuery wrapper!

Recap

It is a jQuery wrapper!

The wrapper holds **multiple elements** which have been selected

Recap

Selectors

It is a jQuery wrapper!

The wrapper holds multiple elements which have been selected

The wrapper has a variety of functions which can be applied to all the selected elements

Recap

Selectors

The wrapper holds **multiple elements** which have been selected

The wrapper has a **variety of functions** which can be applied to all the selected elements

You can manipulate selected elements in one line of code!

extend the jQuery prototype object

```
$('h3').addClass('red-color');
```

Using the $ function returns a jQuery object

extend the jQuery prototype object

```
$('h3').addClass('red-color');
```

The addClass() is applied to the jQuery object returned

extend the jQuery prototype object

```
$('h3').addClass('red-color');
```

The jQuery object gets these methods from the $.fn object

$.fn object

All the functions we've seen so far are available in this object

$.fn object

In order to add new functions
we simply add them to this

EXAMPLE 46

FancyButton.html

FancyButton.html

Example46Demo

FancyButton.html

```
<body>
<div class="button1">
    Click Me
</div>
<br><br>
<div class="button2">
    Hello!
</div>
```

FancyButton.html

```
<body>
<div class="button1">
    Click Me
</div>
<br><br>
<div class="button2">
    Hello!
</div>
```

Getting this look and feel for the buttons is not just CSS, it is a custom plugin

FancyButton.html

```
$.fn.fancyButton = function() {
    this.css({
        'display': 'inline-block',
        'height': '34px',
        'width': '150px',
        'vertical-align': 'middle',
        'text-align': 'center',
        'font-family': 'sans-serif',
        'background-color': 'chocolate',
        'border': 'solid 1px chocolate',
        'opacity': '0.7',
        'color': 'antiquewhite',
        'cursor': 'pointer',
        'margin': '2px 0px',
        'line-height': '34px'
    });

    this.hover(function(e) {
        $(this).css({
            'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
            'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
        });
    });

    return this;
};
```

Here is the plugin code

FancyButton.html

```
$.fn.fancyButton = function() {
    this.css({
        'display': 'inline-block',
        'height': '34px',
        'width': '150px',
        'vertical-align': 'middle',
        'text-align': 'center',
        'font-family': 'sans-serif',
        'background-color': 'chocolate',
        'border': 'solid 1px chocolate',
        'opacity': '0.7',
        'color': 'antiquewhite',
        'cursor': 'pointer',
        'margin': '2px 0px',
        'line-height': '34px'
    });

    this.hover(function(e) {
        $(this).css({
            'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
            'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
        });
    });

    return this;
};
```

Add the fancyButton method to $.fn

FancyButton.html

```
$.fn.fancyButton = function() {
    this.css({
        'display': 'inline-block',
        'height': '34px',
        'width': '150px',
        'vertical-align': 'middle',
        'text-align': 'center',
        'font-family': 'sans-serif',
        'background-color': 'chocolate',
        'border': 'solid 1px chocolate',
        'opacity': '0.7',
        'color': 'antiquewhite',
        'cursor': 'pointer',
        'margin': '2px 0px',
        'line-height': '34px'
    });

    this.hover(function(e) {
        $(this).css({
            'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
            'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
        });
    });

    return this;
};
```

This will allow us to call this function on the jQuery object

FancyButton.html

```
$.fn.fancyButton = function() {
    this.css({
        'display': 'inline-block',
        'height': '34px',
        'width': '150px',
        'vertical-align': 'middle',
        'text-align': 'center',
        'font-family': 'sans-serif',
        'background-color': 'chocolate',
        'border': 'solid 1px chocolate',
        'opacity': '0.7',
        'color': 'antiquewhite',
        'cursor': 'pointer',
        'margin': '2px 0px',
        'line-height': '34px'
    });

    this.hover(function(e) {
        $(this).css({
            'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
            'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
        });
    });

    return this;
};
```

Within the function apply CSS to style the button in the way you want to

FancyButton.html

```
$.fn.fancyButton = function() {
    this.css({
        'display': 'inline-block',
        'height': '34px',
        'width': '150px',
        'vertical-align': 'middle',
        'text-align': 'center',
        'font-family': 'sans-serif',
        'background-color': 'chocolate',
        'border': 'solid 1px chocolate',
        'opacity': '0.7',
        'color': 'antiquewhite',
        'cursor': 'pointer',
        'margin': '2px 0px',
        'line-height': '34px'
    });

    this.hover(function(e) {
        $(this).css({
            'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
            'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
        });
    });

    return this;
};
```

Notice the use of this rather than $(this)

FancyButton.html

```
$.fn.fancyButton = function() {
    this.css({
        'display': 'inline-block',
        'height': '34px',
        'width': '150px',
        'vertical-align': 'middle',
        'text-align': 'center',
        'font-family': 'sans-serif',
        'background-color': 'chocolate',
        'border': 'solid 1px chocolate',
        'opacity': '0.7',
        'color': 'antiquewhite',
        'cursor': 'pointer',
        'margin': '2px 0px',
        'line-height': '34px'
    });

    this.hover(function(e) {
        $(this).css({
            'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
            'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
        });
    });

    return this;
};
```

The css() function is applied to the **same element** as the fancyButton() function

FancyButton.html

```
$.fn.fancyButton = function() {
    this.css({
        'display': 'inline-block',
        'height': '34px',
        'width': '150px',
        'vertical-align': 'middle',
        'text-align': 'center',
        'font-family': 'sans-serif',
        'background-color': 'chocolate',
        'border': 'solid 1px chocolate',
        'opacity': '0.7',
        'color': 'antiquewhite',
        'cursor': 'pointer',
        'margin': '2px 0px',
        'line-height': '34px'
    });

    this.hover(function(e) {
        $(this).css({
            'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
            'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
        });
    });

    return this;
};
```

Emphasize the button on hover with some styling changes

FancyButton.html

```
$.fn.fancyButton = function() {
    this.css({
        'display': 'inline-block',
        'height': '34px',
        'width': '150px',
        'vertical-align': 'middle',
        'text-align': 'center',
        'font-family': 'sans-serif',
        'background-color': 'chocolate',
        'border': 'solid 1px chocolate',
        'opacity': '0.7',
        'color': 'antiquewhite',
        'cursor': 'pointer',
        'margin': '2px 0px',
        'line-height': '34px'
    });

    this.hover(function(e) {
        $(this).css({
            'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
            'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
        });
    });

    return this;
};
```

Change the opacity and color on mouseenter

FancyButton.html

```
$.fn.fancyButton = function() {
    this.css({
        'display': 'inline-block',
        'height': '34px',
        'width': '150px',
        'vertical-align': 'middle',
        'text-align': 'center',
        'font-family': 'sans-serif',
        'background-color': 'chocolate',
        'border': 'solid 1px chocolate',
        'opacity': '0.7',
        'color': 'antiquewhite',
        'cursor': 'pointer',
        'margin': '2px 0px',
        'line-height': '34px'
    });

    this.hover(function(e) {
        $(this).css({
            'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
            'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
        });
    });

    return this;
};
```

Return the element so that this function can be **chained** with others

FancyButton.html

```
$('.button1').fancyButton().addClass('fancified');
$('.button2').fancyButton();
```

```
<body>
<div class="button1">
    Click Me
</div>
<br><br>
<div class="button2">
    Hello!
</div>
```

This is all you need to do to create a fancy button!

FancyButton.html

```
$('.button1').fancyButton().addClass('fancified');
$('.button2').fancyButton();
```

```
<body>
<div class="button1">
    Click Me
</div>
<br><br>
<div class="button2">
    Hello!
</div>
```

Chain other jQuery calls with the fancyButton() call

FancyButton.html

```
$('.button1').fancyButton().addClass('fancified');
$('.button2').fancyButton();
```

```
<body>
<div class="button1">
    Click Me
</div>
<br><br>
<div class="button2">
    Hello!
</div>
```

There is **no difference** between our custom plugin and other jQuery functions we've seen so far!

EXAMPLE 47

MoreFancyButton.html

MoreFancyButton.html

The previous example wasn't complete

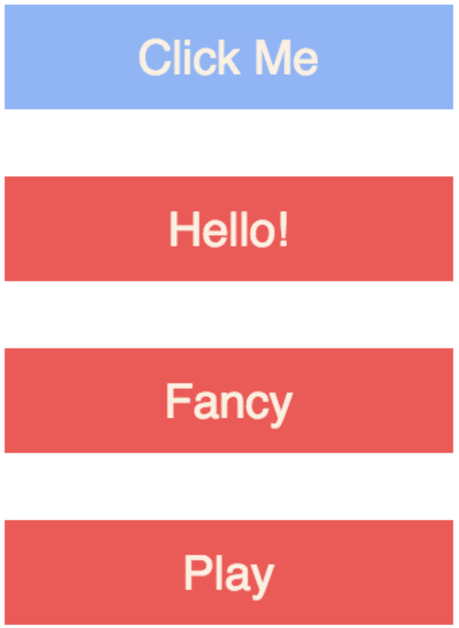
There are some other good practices to follow while developing a custom plugin

MoreFancyButton.html

Example47Demo

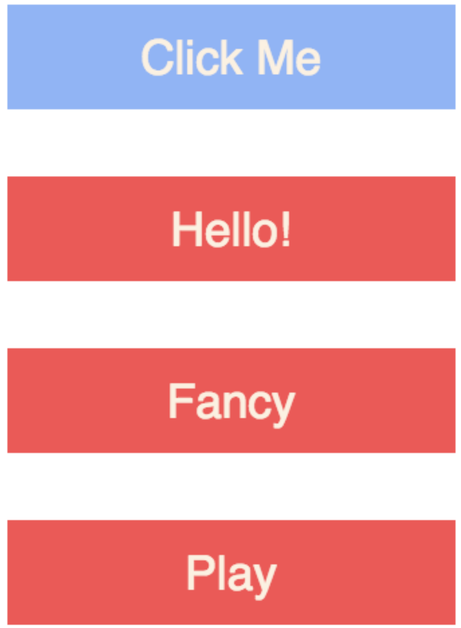
MoreFancyButton.html

```
<body>
<div class="button1">
    Click Me
</div>
<br><br>
<div class="button">
    Hello!
</div>
<br><br>
<div class="button">
    Fancy
</div>
<br><br>
<div class="button">
    Play
</div>
</body>
```



MoreFancyButton.html

```
<body>
<div class="button1">
    Click Me
</div>
<br><br>
<div class="button">
    Hello!
</div>
<br><br>
<div class="button">
    Fancy
</div>
<br><br>
<div class="button">
    Play
</div>
</body>
```



This fancyButton gives us the ability to specify **different colors** for the button

MoreFancyButton.html

```
(function ($) {
    $.fn.fancyButton = function(color, fontFamily) {
        this.each(function() {
            var properties = {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            };

            var options = {};
            if (color) {
                options.backgroundColor = color;
                options.border = 'solid 1px ' + color;
            }
            if (fontFamily) {
                options.fontFamily = fontFamily;
            }
            var extendedProperties = $.extend(properties, options);


            $(this).css(extendedProperties);

            $(this).hover(function(e) {
                $(this).css({
                    'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                    'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
                });
            });
        });

        return this;
    };
} (jQuery));
```

There are a bunch of improvements we've made to the plugin

MoreFancyButton.html

```
(function ($) {
    $.fn.fancyButton = function(color, fontFamily) {
        this.each(function() {
            var properties = {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            };

            var options = {};
            if (color) {
                options.backgroundColor = color;
                options.border = 'solid 1px ' + color;
            }
            if (fontFamily) {
                options.fontFamily = fontFamily;
            }
            var extendedProperties = $.extend(properties, options);


            $(this).css(extendedProperties);

            $(this).hover(function(e) {
                $(this).css({
                    'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                    'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
                });
            });
        });

        return this;
    };
} (jQuery));
```

The $ variable might be used by another library and jQuery might be running in the no conflict mode

MoreFancyButton.html

```
(function ($) {
    $.fn.fancyButton = function(color, fontFamily) {
        this.each(function() {
            var properties = {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            };

            var options = {};
            if (color) {
                options.backgroundColor = color;
                options.border = 'solid 1px ' + color;
            }
            if (fontFamily) {
                options.fontFamily = fontFamily;
            }
            var extendedProperties = $.extend(properties, options);


            $(this).css(extendedProperties);

            $(this).hover(function(e) {
                $(this).css({
                    'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                    'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
                });
            });
        });

        return this;
    };
} (jQuery));
```

However our plugin might rely on $ being available for jQuery

MoreFancyButton.html

```
(function ($) {
    $.fn.fancyButton = function(color, fontFamily) {
        this.each(function() {
            var properties = {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            };

            var options = {};
            if (color) {
                options.backgroundColor = color;
                options.border = 'solid 1px ' + color;
            }
            if (fontFamily) {
                options.fontFamily = fontFamily;
            }
            var extendedProperties = $.extend(properties, options);


            $(this).css(extendedProperties);

            $(this).hover(function(e) {
                $(this).css({
                    'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                    'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
                });
            });
        });

        return this;
    };

} (jQuery));
```

Place your entire plugin in an immediately invokable function expression

Have $ as the argument to this function expression

MoreFancyButton.html

```
(function ($) {
    $.fn.fancyButton = function(color, fontFamily) {
        this.each(function() {
            var properties = {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            };

            var options = {};
            if (color) {
                options.backgroundColor = color;
                options.border = 'solid 1px ' + color;
            }
            if (fontFamily) {
                options.fontFamily = fontFamily;
            }
            var extendedProperties = $.extend(properties, options);


            $(this).css(extendedProperties);

            $(this).hover(function(e) {
                $(this).css({
                    'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                    'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
                });
            });
        });

        return this;
    };
} (jQuery));
```

Place your entire plugin in an **immediately invokable function expression**

Have $ as the argument to this function expression

This preserves the $ variable for your plugin!

MoreFancyButton.html

immediately invokable function expression

```
(function () {
  // function code here
})
```

The parens treat this function as an expression rather than a declaration

MoreFancyButton.html

immediately invokable function expression

```
(function () {
  // function code here
})
```

An immediately invoked function expressions runs as soon as it is defined

MoreFancyButton.html

```
(function ($) {
    $.fn.fancyButton = function(color, fontFamily) {
        this.each(function() {
            var properties = {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            };

            var options = {};
            if (color) {
                options.backgroundColor = color;
                options.border = 'solid 1px ' + color;
            }
            if (fontFamily) {
                options.fontFamily = fontFamily;
            }
            var extendedProperties = $.extend(properties, options);


            $(this).css(extendedProperties);

            $(this).hover(function(e) {
                $(this).css({
                    'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                    'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
                });
            });
        });

        return this;
    };
} (jQuery));
```

Now for the fancyButton plugin method

MoreFancyButton.html

```
(function ($) {
    $.fn.fancyButton = function(color, fontFamily) {
        this.each(function() {
            var properties = {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px peachpuff',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            };

            var options = {};
            if (color) {
                options.backgroundColor = color;
                options.border = 'solid 1px ' + color;
            }
            if (fontFamily) {
                options.fontFamily = fontFamily;
            }
            var extendedProperties = $.extend(properties, options);

            $(this).css(extendedProperties);

            $(this).hover(function(e) {
                $(this).css({
                    'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                    'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
                });
            });
        });

        return this;
    };
} (jQuery));
```

It's customizable, you can specify a **color** for the button as well as a **font** for the button text

MoreFancyButton.html

```
(function ($) {
    $.fn.fancyButton = function(color, fontFamily) {
        this.each(function() {
            var properties = {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            };

            var options = {};
            if (color) {
                options.backgroundColor = color;
                options.border = 'solid 1px ' + color;
            }
            if (fontFamily) {
                options.fontFamily = fontFamily;
            }
            var extendedProperties = $.extend(properties, options);


            $(this).css(extendedProperties);

            $(this).hover(function(e) {
                $(this).css({
                    'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                    'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
                });
            });
        });

        return this;
    };
} (jQuery));
```

This plugin acts on **all elements** in a selection

MoreFancyButton.html

```
this.each(function() {
        var properties = {
            'display': 'inline-block',
            'height': '34px',
            'width': '150px',
            'vertical-align': 'middle',
            'text-align': 'center',
            'font-family': 'sans-serif',
            'background-color': 'chocolate',
            'border': 'solid 1px chocolate',
            'opacity': '0.7',
            'color': 'antiquewhite',
            'cursor': 'pointer',
            'margin': '2px 0px',
            'line-height': '34px'
        };

        var options = {};
        if (color) {
            options.backgroundColor = color;
            options.border = 'solid 1px ' + color;
        }
        if (fontFamily) {
            options.fontFamily = fontFamily;
        }
        var extendedProperties = $.extend(properties, options);


        $(this).css(extendedProperties);

        $(this).hover(function(e) {
            $(this).css({
                'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
            });
        });
    });

    return this;
};
```

Specify the CSS in a properties object

MoreFancyButton.html

```
this.each(function() {
        var properties = {
            'display': 'inline-block',
            'height': '34px',
            'width': '150px',
            'vertical-align': 'middle',
            'text-align': 'center',
            'font-family': 'sans-serif',
            'background-color': 'chocolate',
            'border': 'solid 1px chocolate',
            'opacity': '0.7',
            'color': 'antiquewhite',
            'cursor': 'pointer',
            'margin': '2px 0px',
            'line-height': '34px'
        };

          var options = {};
          if (color) {
              options.backgroundColor = color;
              options.border = 'solid 1px ' + color;
          }
          if (fontFamily) {
              options.fontFamily = fontFamily;
          }
        var extendedProperties = $.extend(properties, options);


        $(this).css(extendedProperties);

        $(this).hover(function(e) {
            $(this).css({
                'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
            });
        });
    });

    return this;
};
```

Set the color and font to the user specified values

MoreFancyButton.html

```
this.each(function() {
        var properties = {
            'display': 'inline-block',
            'height': '34px',
            'width': '150px',
            'vertical-align': 'middle',
            'text-align': 'center',
            'font-family': 'sans-serif',
            'background-color': 'chocolate',
            'border': 'solid 1px chocolate',
            'opacity': '0.7',
            'color': 'antiquewhite',
            'cursor': 'pointer',
            'margin': '2px 0px',
            'line-height': '34px'
        };

        var options = {};
        if (color) {
            options.backgroundColor = color;
            options.border = 'solid 1px ' + color;
        }
        if (fontFamily) {
            options.fontFamily = fontFamily;
        }
        var extendedProperties = $.extend(properties, options);

        $(this).css(extendedProperties);

        $(this).hover(function(e) {
            $(this).css({
                'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
            });
        });
    });

    return this;
};
```

The $.extend() **overwrites** the original properties with the user specified values

MoreFancyButton.html

```
this.each(function() {
        var properties = {
            'display': 'inline-block',
            'height': '34px',
            'width': '150px',
            'vertical-align': 'middle',
            'text-align': 'center',
            'font-family': 'sans-serif',
            'background-color': 'chocolate',
            'border': 'solid 1px chocolate',
            'opacity': '0.7',
            'color': 'antiquewhite',
            'cursor': 'pointer',
            'margin': '2px 0px',
            'line-height': '34px'
        };

        var options = {};
        if (color) {
            options.backgroundColor = color;
            options.border = 'solid 1px ' + color;
        }
        if (fontFamily) {
            options.fontFamily = fontFamily;
        }
        var extendedProperties = $.extend(properties, options);

        $(this).css(extendedProperties);

        $(this).hover(function(e) {
            $(this).css({
                'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
            });
        });
    });

    return this;
};
```

Apply the CSS to the buttons

MoreFancyButton.html

```
this.each(function() {
        var properties = {
            'display': 'inline-block',
            'height': '34px',
            'width': '150px',
            'vertical-align': 'middle',
            'text-align': 'center',
            'font-family': 'sans-serif',
            'background-color': 'chocolate',
            'border': 'solid 1px chocolate',
            'opacity': '0.7',
            'color': 'antiquewhite',
            'cursor': 'pointer',
            'margin': '2px 0px',
            'line-height': '34px'
        };

        var options = {};
        if (color) {
            options.backgroundColor = color;
            options.border = 'solid 1px ' + color;
        }
        if (fontFamily) {
            options.fontFamily = fontFamily;
        }
        var extendedProperties = $.extend(properties, options);

        $(this).css(extendedProperties);

        $(this).hover(function(e) {
            $(this).css({
                'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
            });
        });
    });

    return this;
};
```

Use $(this) because we are within the this.each() callback

MoreFancyButton.html

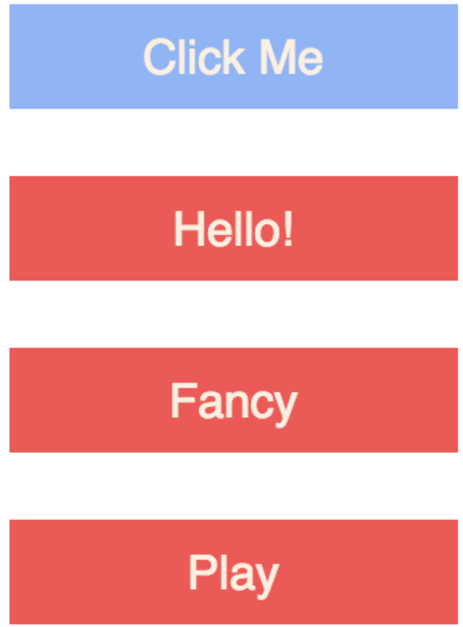
```
this.each(function() {
        var properties = {
            'display': 'inline-block',
            'height': '34px',
            'width': '150px',
            'vertical-align': 'middle',
            'text-align': 'center',
            'font-family': 'sans-serif',
            'background-color': 'chocolate',
            'border': 'solid 1px chocolate',
            'opacity': '0.7',
            'color': 'antiquewhite',
            'cursor': 'pointer',
            'margin': '2px 0px',
            'line-height': '34px'
        };

        var options = {};
        if (color) {
            options.backgroundColor = color;
            options.border = 'solid 1px ' + color;
        }
        if (fontFamily) {
            options.fontFamily = fontFamily;
        }
        var extendedProperties = $.extend(properties, options);


       $(this).css(extendedProperties);
          $(this).hover(function(e) {
              $(this).css({
                  'opacity': e.type === 'mouseenter' ? '1.0' : '0.7',
                  'color': e.type === 'mouseenter' ? 'white' : 'antiquewhite'
              });
          });
    });

    return this;
};
```

The hover() method is the same as before

MoreFancyButton.html

```
$('.button1').fancyButton('cornflowerblue');
$('.button').fancyButton('#de1515');
```

```
<body>
<div class="button1">
    Click Me
</div>
<br><br>
<div class="button">
    Hello!
</div>
<br><br>
<div class="button">
    Fancy
</div>
<br><br>
<div class="button">
    Play
</div>
</body>
```



JQUERY

Widget Factory

Widget Factory

The custom plugin we set up was a **stateless** plugin

This works for simple stuff where there is no **lifecycle** or **state** to be maintained

Widget Factory

For the more advanced plugins that we use, this is insufficient

jQuery provides a Widget Factory to allow us to create more **complex** plugins

Widget Factory

more complex plugins

They have code for initialization, state management

This results in a lot of **boilerplate** code!

Widget Factory

more complex plugins

The widget factory exists to **minimize** this boiler plate

And provide a **consistent API** for all these functions

Widget Factory

```
$.widget(name [, base], prototype)
```

This is the jQuery UI widget factory

Widget Factory

```
$.widget( name [, base], prototype)
```

A string which has the
`namespace.widgetName`

Widget Factory

```
$.widget(name [, base], prototype)
```

The optional base widget to inherit from

Widget Factory

```
$.widget(name [, base], prototype)
```

It defaults to jQuery.Widget

Widget Factory

```
$.widget(name [, base], prototype)
```

Many jQuery UI widgets inherit from other widgets

Widget Factory

```
$.widget(name [, base], prototype)
```

ui.draggable inherits from ui.mouse

Widget Factory

```
$.widget(name [, base], prototype)
```

The object to use as a prototype for the widget to inherit from

Widget Factory

```
$.widget(name [, base], prototype)
```

This object will have standard properties which your widget will **inherit**

EXAMPLE 48

Widgets.html

Widgets.html

```
<body>
</body>
```

Widgets.html

```
<body>
</body>
```

We'll set up the **same button** as in the previous example - with some **additional** functionality

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

Here is how we set up a widget for this button

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

Set up the widget within an immediately invokable function expression

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

The name of the widget is

loony.fancyButton

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

The ui namespace is used by jQuery UI, we can use any other namespace

Widgets.html

```
$(document).ready(function () {
      (function ($) {
          $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

We will invoke the widget using only the fancyButton name

Widgets.html

```
$(document).ready(function () {
      (function ($) {
          $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

We will invoke the widget using only the fancyButton name

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

This widget will inherit from the base widget jQuery.Widget

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

The **options** property allows us to specify **default values** for the widget

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

The options specify the CSS and the text for the button

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

The _create property defines a function which serves like a 'constructor' for the widget

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

This is a **known property** to the widget infrastructure

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

Create a button element within this object

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

this references an object and not an element

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

The _button is a private variable on this object

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

The element on which the widget is applied is available in this context as

this.element

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

The default options are available as well as this.options

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

Call a private function to update the button

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
               this._button = $('<div></div>');
               $(this.element).append(this._button);
               this._update();
            },

            _update: function() {
               this._button.css(this.options.css);
               this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

The _update is a private function which we have defined

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
               this._button = $('<div></div>');
               $(this.element).append(this._button);
               this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

Set the CSS and the text of the button element here

Widgets.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
               this._button = $('<div></div>');
               $(this.element).append(this._button);
               this._update();
            },

        _update: function() {
           this._button.css(this.options.css);
           this._button.text(this.options.text);
        }
        });
    } (jQuery));

    $('body').fancyButton();
});
```

Set up the fancyButton on the <body> element

EXAMPLE 49

WidgetsWithFunctions.html

WidgetsWithFunctions.html

Let's extend the previous example

Add a function to set the **text** and **styles** for the button

WidgetsWithFunctions.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            },

            _setOption: function(key, value) {
                if (key == 'css') {
                    var newCss = $.extend(this.options.css, value);
                    this.options.css = newCss;
                    this._update();
                } else if (key == 'text') {
                    this.text(value);
                }
            },

            text: function(text) {
                if (!text) {
                    return this._button.text();
                }
                this.options.text = text;
                this._update();
            }
        });
```

The default options are set up the same way as before

WidgetsWithFunctions.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            },

            _setOption: function(key, value) {
                if (key == 'css') {
                    var newCss = $.extend(this.options.css, value);
                    this.options.css = newCss;
                    this._update();
                } else if (key == 'text') {
                    this.text(value);
                }
            },

            text: function(text) {
                if (!text) {
                    return this._button.text();
                }
                this.options.text = text;
                this._update();
            }
        });
    } (jQuery));
```

The _create function which acts as the widget constructor is also the same

WidgetsWithFunctions.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            },

            _setOption: function(key, value) {
                if (key == 'css') {
                    var newCss = $.extend(this.options.css, value);
                    this.options.css = newCss;
                    this._update();
                } else if (key == 'text') {
                    this.text(value);
                }
            },

            text: function(text) {
                if (!text) {
                    return this._button.text();
                }
                this.options.text = text;
                this._update();
            }
        });
    } (jQuery));
```

The private function _update sets the CSS and text on the button

WidgetsWithFunctions.html

```
text: function(text) {
    if (!text) {
        return this._button.text();
    }
    this.options.text = text;
    this._update();
}
```

Here is a public method which allows you to **set the text** of the button after it has been created

WidgetsWithFunctions.html

```
text: function(text) {
    if (!text) {
        return this._button.text();
    }
    this.options.text = text;
    this._update();
}
```

These are callable by developers using this widget

WidgetsWithFunctions.html

```
text: function(text) {
    if (!text) {
        return this._button.text();
    }
    this.options.text = text;
    this._update();
}
```

It accepts an argument

WidgetsWithFunctions.html

```
text: function(text) {
    if (!text) {
        return this._button.text();
    }
    this.options.text = text;
    this._update();
}
```

If not argument is specified it acts as a **getter** and returns the current text on the button

WidgetsWithFunctions.html

```
text: function(text) {
    if (!text) {
        return this._button.text();
    }
    this.options.text = text;
    this._update();
}
```

Update the **options** with the new text and then just call the **_update** which will update the **button** display

WidgetsWithFunctions.html

```
_setOption: function(key, value) {
    if (key == 'css') {
        var newCss = $.extend(this.options.css, value);
        this.options.css = newCss;
        this._update();
    } else if (key == 'text') {
        this.text(value);
    }
},
```

Every widget has an option() method which allows you to get and set options on it

WidgetsWithFunctions.html

```
_setOption: function(key, value) {
    if (key == 'css') {
        var newCss = $.extend(this.options.css, value);
        this.options.css = newCss;
        this._update();
    } else if (key == 'text') {
        this.text(value);
    }
},
```

The option() function calls the _setOption under the hood for every option property set

WidgetsWithFunctions.html

```
_setOption: function(key, value) {
    if (key == 'css') {
        var newCss = $.extend(this.options.css, value);
        this.options.css = newCss;
        this._update();
    } else if (key == 'text') {
        this.text(value);
    }
},
```

This allows you to hook into and run your code each time an option is set

WidgetsWithFunctions.html

```
_setOption: function(key, value) {
    if (key == 'css') {
        var newCss = $.extend(this.options.css, value);
        this.options.css = newCss;
        this._update();
    } else if (key == 'text') {
        this.text(value);
    }
},
```

The input arguments are the key and value being set on the options

WidgetsWithFunctions.html

```
_setOption: function(key, value) {
    if (key == 'css') {
        var newCss = $.extend(this.options.css, value);
        this.options.css = newCss;
        this._update();
    } else if (key == 'text') {
        this.text(value);
    }
},
```

For **CSS** changes we update the default options object and the button display

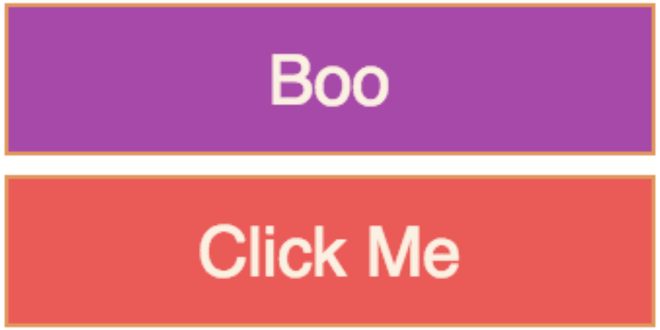
WidgetsWithFunctions.html

```
_setOption: function(key, value) {
    if (key == 'css') {
        var newCss = $.extend(this.options.css, value);
        this.options.css = newCss;
        this._update();
    } else if (key == 'text') {
        this.text(value);
    }
},
```

For **text** changes just set the new text value on the button

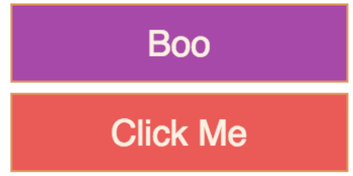
WidgetsWithFunctions.html

```
<body>
<div class="onebutton">
</div>
<div class="twobutton">
</div>
</body>
```



Two buttons, two styles and two values of text on the button

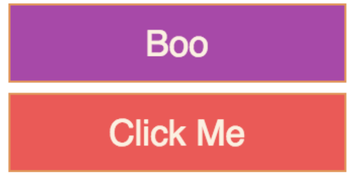
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

Create the button

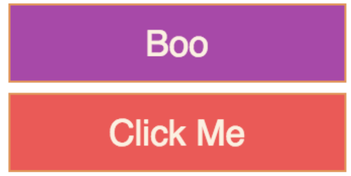
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

Call the option method to set its background color to purple

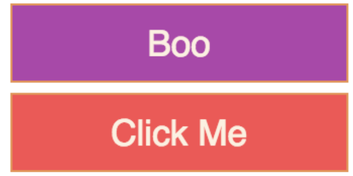
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

A method on the widget is called by using the widget name i.e. fancyButton

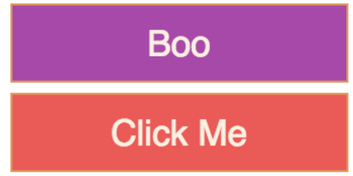
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

And the method name specified as an argument

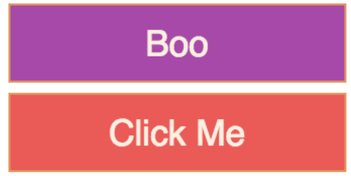
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

And the method name specified as an argument

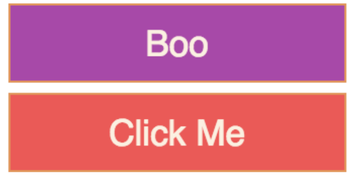
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

The argument to the method is the **key and value pair** for an option

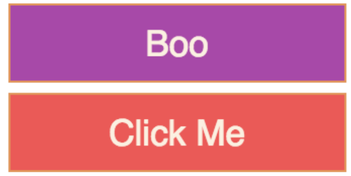
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

For the **css** key

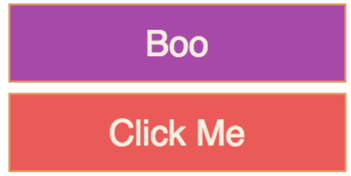
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

Set the background color

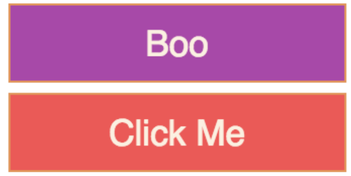
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

The structure of the value is something we define

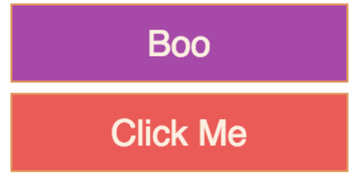
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

The **text** of the button can be set using the **option** method as well

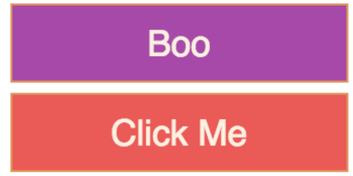
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

Or you can use the **text** method which we wrote explicitly to deal with text

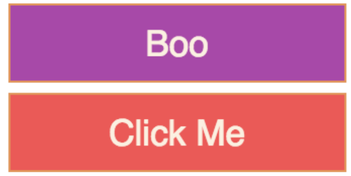
WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');

var text = $('.twobutton').fancyButton('text');
console.log(text);
```

This is definitely less verbose and more useful

WidgetsWithFunctions.html




```
$('.onebutton').fancyButton();
$('.onebutton').fancyButton('option', {'css': {'background-color': 'purple'}});
$('.onebutton').fancyButton('option', {'text': 'Boo'});

$('.twobutton').fancyButton();
$('.twobutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
$('.twobutton').fancyButton('text', 'Click Me');
```

```
var text = $('.twobutton').fancyButton('text');
console.log(text);
```

The text method acts as a **getter** when the input argument is not specified

WidgetsWithFunctions.html

```
$(document).ready(function () {
    (function ($) {
        $.widget('loony.fancyButton', {
            options: {
                'css': {
                    'display': 'inline-block',
                    'height': '34px',
                    'width': '150px',
                    'vertical-align': 'middle',
                    'text-align': 'center',
                    'font-family': 'sans-serif',
                    'background-color': 'chocolate',
                    'border': 'solid 1px chocolate',
                    'opacity': '0.7',
                    'color': 'antiquewhite',
                    'cursor': 'pointer',
                    'margin': '2px 0px',
                    'line-height': '34px'
                },
                'text': 'Submit'
            },

            _create: function() {
                this._button = $('<div></div>');
                $(this.element).append(this._button);
                this._update();
            },

            _update: function() {
                this._button.css(this.options.css);
                this._button.text(this.options.text);
            },

            _setOption: function(key, value) {
                if (key == 'css') {
                    var newCss = $.extend(this.options.css, value);
                    this.options.css = newCss;
                    this._update();
                } else if (key == 'text') {
                    this.text(value);
                }
            },

            text: function(text) {
                if (!text) {
                    return this._button.innerText;
                }
                this.options.text = text;
                this._update();
            }
        });
    } (jQuery));
```

Much of this code is the same as the previous example

EXAMPLE 50

WidgetsWithEvents.html

WidgetsWithEvents.html

A widget that you've created might want to **trigger** its own events

A **play/pause** button on a player **toggles state** each time it is clicked

WidgetsWithEvents.html

trigger its own events

A play/pause button on a player
toggles state each time it is clicked

It might want to trigger a
'playEvent' and 'pauseEvent' so
other components know whether
the music is playing or not

WidgetsWithEvents.html

```
<body>
<div class="onebutton">
</div>
</body>
```

A single button with two states - paused and playing

WidgetsWithEvents.html


```
$('.onebutton').fancyButton({
    playEvent: function() {console.log('Play the music!');},
    pauseEvent: function() {console.log('Pause the music!');}
}).addClass('fancybutton');
$('.onebutton').fancyButton('option', {'css': {'background-color': '#de1515'}});

$('.fancybutton').click(function() {
    $(this).fancyButton('toggleState');
});
```

Let's see how we use the widget events first

WidgetsWithEvents.html


```
$('.onebutton').fancyButton({
    playEvent: function() {console.log('Play the music!');},
    pauseEvent: function() {console.log('Pause the music!');}
}).addClass('fancybutton');
$('.onebutton').fancyButton('option', {'css': {'background-color': '#de1515'}});

$('.fancybutton').click(function() {
    $(this).fancyButton('toggleState');
});
```

While setting up the button specify the listeners as well

WidgetsWithEvents.html

Play

```
$('.onebutton').fancyButton({
    playEvent: function() {console.log('Play the music!');},
    pauseEvent: function() {console.log('Pause the music!');}
}).addClass('fancybutton');
$('.onebutton').fancyButton('option', {'css': {'background-color': '#de1515'}});

$('.fancybutton').click(function() {
    $(this).fancyButton('toggleState');
});
```

The button triggers the playEvent and the pauseEvent

WidgetsWithEvents.html


```
$('.onebutton').fancyButton({
    playEvent: function() {console.log('Play the music!');},
    pauseEvent: function() {console.log('Pause the music!');}
}).addClass('fancybutton');
$('.onebutton').fancyButton('option', {'css': {'background-color': '#de1515'}});

$('.fancybutton').click(function() {
    $(this).fancyButton('toggleState');
});
```

The listeners simply log the event to console

WidgetsWithEvents.html


```
$('.onebutton').fancyButton({
    playEvent: function() {console.log('Play the music!');},
    pauseEvent: function() {console.log('Pause the music!');}
}).addClass('fancybutton');
$('.onebutton').fancyButton('option', {'css': {'background-color': '#de1515'}});
```

```
$('.fancybutton').click(function() {
    $(this).fancyButton('toggleState');
});
```

Toggle the current state of the button when you click it

Example50Demo

WidgetsWithEvents.html

```
(function ($) {
    $.widget('loony.fancyButton', {
        options: {
            'css': {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            },
            'pause': true
        },

        _create: function() {
            this._button = $('<div></div>');
            $(this.element).append(this._button);
            this._update();
        },

        _update: function() {
            if (this.options.pause) {
                this._button.text('Pause');
            } else {
                this._button.text('Play');
            }
            this._button.css(this.options.css);
        },

        _setOption: function(key, value) {
            if (key == 'css') {
                this.options.css = $.extend(this.options.css, value);
            } else if (key == 'pause') {
                this.options.pause = value;
            }
            this._update();
        },

        toggleState: function() {
            this.options.pause = !this.options.pause;
            this._update();
            if (this.options.pause) {
                this._trigger('pauseEvent');
            } else {
                this._trigger('playEvent');
            }
        }
    });
} (jQuery));
```

A lot of the code should look very familiar

WidgetsWithEvents.html

```
        options: {
            'css': {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            },
            'pause': true
        },
```

The options has CSS related stuff but does not specify the text of the button

The text is only **Play** or **Pause** depending on the state

WidgetsWithEvents.html

```
        options: {
            'css': {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            },
            'pause': true
        },
```

The options hold the **default state** of the button i.e. it starts off in the pause state

WidgetsWithEvents.html

```
_create: function() {
    this._button = $('<div></div>');
    $(this.element).append(this._button);
    this._update();
},
```

```
_update: function() {
    if (this.options.pause) {
        this._button.text('Pause');
    } else {
        this._button.text('Play');
    }
    this._button.css(this.options.css);
},
```

The _create code remains the same

WidgetsWithEvents.html

```
_create: function() {
    this._button = $('<div></div>');
    $(this.element).append(this._button);
    this._update();
},

_update: function() {
    if (this.options.pause) {
        this._button.text('Pause');
    } else {
        this._button.text('Play');
    }
    this._button.css(this.options.css);
},
```

The _update code **updates the state** of the button in addition to its display properties

WidgetsWithEvents.html

```
    _create: function() {
        this._button = $('<div></div>');
        $(this.element).append(this._button);
        this._update();
    },

    _update: function() {
        if (this.options.pause) {
            this._button.text('Pause');
        } else {
            this._button.text('Play');
        }
        this._button.css(this.options.css);
    },
```

The button text says Play or Pause based on what is set in the options

WidgetsWithEvents.html

```
_setOption: function(key, value) {
    if (key == 'css') {
        this.options.css = $.extend(this.options.css, value);
    } else if (key == 'pause') {
        this.options.pause = value;
    }
    this._update();
},
```

The pause value can also be updated via the option method

WidgetsWithEvents.html

```
toggleState: function() {
     this.options.pause = !this.options.pause;
     this._update();
     if (this.options.pause) {
         this._trigger('pauseEvent');
     } else {
         this._trigger('playEvent');
     }
 }
```

Toggle state changes the state of the button from play to paused and vice versa

WidgetsWithEvents.html

```
toggleState: function() {
     this.options.pause = !this.options.pause;
     this._update();
     if (this.options.pause) {
         this._trigger('pauseEvent');
     } else {
         this._trigger('playEvent');
     }
 }
```

Trigger events based on what the new state is

WidgetsWithEvents.html

```
toggleState: function() {
     this.options.pause = !this.options.pause;
     this._update();
     if (this.options.pause) {
         this._trigger('pauseEvent');
     } else {
         this._trigger('playEvent');
     }
 }
```

The _trigger() function fires an event from the widget

WidgetsWithEvents.html

```
(function ($) {
    $.widget('loony.fancyButton', {
        options: {
            'css': {
                'display': 'inline-block',
                'height': '34px',
                'width': '150px',
                'vertical-align': 'middle',
                'text-align': 'center',
                'font-family': 'sans-serif',
                'background-color': 'chocolate',
                'border': 'solid 1px chocolate',
                'opacity': '0.7',
                'color': 'antiquewhite',
                'cursor': 'pointer',
                'margin': '2px 0px',
                'line-height': '34px'
            },
            'pause': true
        },

        _create: function() {
            this._button = $('<div></div>');
            $(this.element).append(this._button);
            this._update();
        },

        _update: function() {
            if (this.options.pause) {
                this._button.text('Pause');
            } else {
                this._button.text('Play');
            }
            this._button.css(this.options.css);
        },

        _setOption: function(key, value) {
            if (key == 'css') {
                this.options.css = $.extend(this.options.css, value);
            } else if (key == 'pause') {
                this.options.pause = value;
            }
            this._update();
        },

        toggleState: function() {
            this.options.pause = !this.options.pause;
            this._update();
            if (this.options.pause) {
                this._trigger('pauseEvent');
            } else {
                this._trigger('playEvent');
            }
        }
    });
} (jQuery));
```

JQUERY

The jQuery UI Library

The jQuery UI Library

This is a widget and interaction library which is built and maintained by the jQuery team

You can **use plugins** from this library with **confidence** that they will be supported and widely used

Get started with jQuery UI by downloading and setting it up on your machine

The jQuery UI Library

JQueryUIDemosAndDownload
JQueryUIUnzipAndIndexFile

EXAMPLE 51

JQueryUI.html

Example51Demo

JQueryUI.html

```
<link href="../../jquery-ui-1.12.0.custom/jquery-ui.css" rel="stylesheet">
<script src="../../lib/jquery.js"></script>
<script src="../../jquery-ui-1.12.0.custom/jquery-ui.js"></script>
```

Make sure you include the JS and CSS required for jQuery UI

JQueryUI.html

```
<body>
<h2>Datepicker</h2>
<div>
    Date: <input class="datepicker">
</div>

<h2>Autocomplete</h2>
<div>
    <input class="autocomplete">
</div>
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

The HTML set up for each of these jQuery UI components tend to be ridiculously simple

JQueryUI.html

```
<body>
<h2>Datepicker</h2>
<div>
    Date: <input class="datepicker">
</div>
```

Simply specify an input which will accept the date

JQueryUI.html

```
<body>
<h2>Datepicker</h2>
<div>
    Date: <input class="datepicker">
</div>
```

```
$('.datepicker').datepicker({
    inline: true
});
```

Just select the input and call the datepicker() function on it!

JQueryUI.html

```
<body>
<h2>Datepicker</h2>
<div>
    Date: <input class="datepicker">
</div>
```

```
$('.datepicker').datepicker({
    inline: true
});
```

jQuery UI components generally accept a configuration object, which allows you to control certain elements

JQueryUI.html

```
<body>
<h2>Datepicker</h2>
<div>
    Date: <input class="datepicker">
</div>
```

```
$('.datepicker').datepicker({
    inline: true
});
```


```
<h2>Autocomplete</h2>
<div>
    <input class="autocomplete">
</div>
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: Jean and Daniel Granger <br>
        Can be known by: Bushy hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```


And that's it! You have a fully functional date picker set up

JQueryUI.html

```
<h2>Autocomplete</h2>
<div>
    <input class="autocomplete">
</div>
```

The autocomplete widget is also usually used with an input box

JQueryUI.html

```
<h2>Autocomplete</h2>
<div>
    <input class="autocomplete">
</div>
```

```
var names = [
    'Harry',
    'Hermione',
    'Ron',
    'Draco',
    'Ginny',
    'Cedric',
    'Voldemort'
];
$('.autocomplete').autocomplete({
    source: names
});
```

Just specify the source of the autocomplete as an array

JQueryUI.html

```
<h2>Autocomplete</h2>
<div>
    <input class="autocomplete">
</div>
```

```
var names = [
    'Harry',
    'Hermione',
    'Ron',
    'Draco',
    'Ginny',
    'Cedric',
    'Voldemort'
];
$('.autocomplete').autocomplete({
    source: names
});
```

Aaaand you're done!

JQueryUI.html

```
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

This outer contains the tabs

JQueryUI.html

```
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

A list of text on the tabs

JQueryUI.html

```
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

3 tabs, 3 list items

JQueryUI.html

```
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

The tabs should be links which point to the contents

JQueryUI.html

```
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

The **href** parameter is the **id** of the which holds the tab contents

JQueryUI.html

```
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

And finally the contents displayed when each tab is clicked

JQueryUI.html

```
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

```
$( ".tabs" ).tabs();
```

The actual code to get this HTML into the tab structure is just one line

JQueryUI.html

These are just a few examples of jQuery components

The library is rich and each component has many options

Explore it!

JQueryUI.html

```
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

The HTML for tabs is a little more involved

JQueryUI.html

```
<body>
<h2>Datepicker</h2>
<div>
    Date: <input class="datepicker">
</div>

<h2>Autocomplete</h2>
<div>
    <input class="autocomplete">
</div>
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

The HTML set up for each of these jQuery UI components tend to be ridiculously simple

JQueryUI.html

```
<body>
<h2>Datepicker</h2>
<div>
    Date: <input class="datepicker">
</div>

<h2>Autocomplete</h2>
<div>
    <input class="autocomplete">
</div>
<h2>Tabs</h2>
<div class="tabs">
    <ul>
        <li><a href="#tabs-harry">Harry</a></li>
        <li><a href="#tabs-hermione">Hermione</a></li>
        <li><a href="#tabs-ron">Ron</a></li>
    </ul>
    <div id="tabs-harry">
        Harry Potter <br>
        Parents: James and Lily Potter <br>
        Can be known by: Scar on forehead
    </div>
    <div id="tabs-hermione">
        Hermione Granger <br>
        Parents: First names unknown. Dentists <br>
        Can be known by: Brown, curly hair
    </div>
    <div id="tabs-ron">
        Ronald Weasley <br>
        Parents: Molly and Arthur Weasley <br>
        Can be known by: Red hair
    </div>
</div>
</body>
```

The HTML set up for each of these jQuery UI components tend to be ridiculously simple

EXAMPLE 52

JQueryUIEffects.html

The jQuery UI library has an **effect()** method which allows you to apply animations to elements

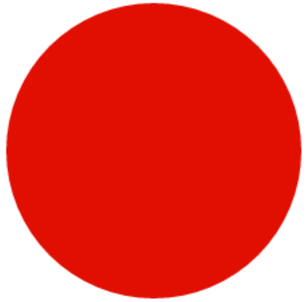
jQueryUIEffects.html

This has animations support beyond what is available in core jQuery

Example52Demo

JQueryUIEffects.html

```
<body>
<div class="outer-ball">
    <div class="ball">
    </div>
</div>
<div class="link bounce">
    Bounce
</div>
<br>
<div class="link puff">
    Puff
</div>
<br>
<div class="link scale">
    Toggle scale
</div>
<br>
<div class="link color">
    Color animation
</div>
</body>
```



Bounce

Puff

Toggle scale

Color animation

JQueryUIEffects.html

```
$('.bounce').click(function() {
    $('.ball').effect('bounce', 2000);
});
$('.puff').click(function() {
    $('.ball').effect('puff', 2000);
});
$('.scale').click(function() {
    $('.ball').toggle({effect: 'scale', direction: 'horizontal'});
});
$('.color').click(function() {
    var colors = ['green', 'red', 'blue', 'black'];
    $('.ball').animate({
        'background-color': colors[Math.floor(Math.random() * colors.length)]
    });
});
```

'bounce' is one of the effects offered by the effect() function

jQueryUIEffects.html

```
$('.bounce').click(function() {
    $('.ball').effect('bounce', 2000);
});
$('.puff').click(function() {
    $('.ball').effect('puff', 2000);
});
$('.scale').click(function() {
    $('.ball').toggle({effect: 'scale', direction: 'horizontal'});
});
$('.color').click(function() {
    var colors = ['green', 'red', 'blue', 'black'];
    $('.ball').animate({
        'background-color': colors[Math.floor(Math.random() * colors.length)]
    });
});
```

Just call effect() and specify bounce!

JQueryUIEffects.html

```
$('.bounce').click(function() {
    $('.ball').effect('bounce', 2000);
});
$('.puff').click(function() {
    $('.ball').effect('puff', 2000);
});
$('.scale').click(function() {
    $('.ball').toggle({effect: 'scale', direction: 'horizontal'});
});
$('.color').click(function() {
    var colors = ['green', 'red', 'blue', 'black'];
    $('.ball').animate({
        'background-color': colors[Math.floor(Math.random() * colors.length)]
    });
});
```

Or puff

JQueryUIEffects.html

```
$('.bounce').click(function() {
    $('.ball').effect('bounce', 2000);
});
$('.puff').click(function() {
    $('.ball').effect('puff', 2000);
});
$('.scale').click(function() {
    $('.ball').toggle({effect: 'scale', direction: 'horizontal'});
});
$('.color').click(function() {
    var colors = ['green', 'red', 'blue', 'black'];
    $('.ball').animate({
        'background-color': colors[Math.floor(Math.random() * colors.length)]
    });
});
```

You can also toggle effects

JQueryUIEffects.html

```
$('.bounce').click(function() {
    $('.ball').effect('bounce', 2000);
});
$('.puff').click(function() {
    $('.ball').effect('puff', 2000);
});
$('.scale').click(function() {
    $('.ball').toggle({effect: 'scale', direction: 'horizontal'});
});
$('.color').click(function() {
    var colors = ['green', 'red', 'blue', 'black'];
    $('.ball').animate({
        'background-color': colors[Math.floor(Math.random() * colors.length)]
    });
});
```

Toggle the scale effect

JQueryUIEffects.html

```
$('.bounce').click(function() {
    $('.ball').effect('bounce', 2000);
});
$('.puff').click(function() {
    $('.ball').effect('puff', 2000);
});
$('.scale').click(function() {
    $('.ball').toggle({effect: 'scale', direction: 'horizontal'});
});
$('.color').click(function() {
    var colors = ['green', 'red', 'blue', 'black'];
    $('.ball').animate({
        'background-color': colors[Math.floor(Math.random() * colors.length)]
    });
});
```

The scale effect takes in configuration parameters

jQueryUIEffects.html

```
$('.bounce').click(function() {
    $('.ball').effect('bounce', 2000);
});
$('.puff').click(function() {
    $('.ball').effect('puff', 2000);
});
$('.scale').click(function() {
    $('.ball').toggle({effect: 'scale', direction: 'horizontal'});
});
$('.color').click(function() {
    var colors = ['green', 'red', 'blue', 'black'];
    $('.ball').animate({
        'background-color': colors[Math.floor(Math.random() * colors.length)]
    });
});
```

Colors can also be animated with this library - core jQuery does not animate color changes

jQueryUIEffects.html

```
$('.bounce').click(function() {
    $('.ball').effect('bounce', 2000);
});
$('.puff').click(function() {
    $('.ball').effect('puff', 2000);
});
$('.scale').click(function() {
    $('.ball').toggle({effect: 'scale', direction: 'horizontal'});
});
$('.color').click(function() {
    var colors = ['green', 'red', 'blue', 'black'];
    $('.ball').animate({
        'background-color': colors[Math.floor(Math.random() * colors.length)]
    });
});
```

Just use the **animate** method as before - including the jQuery UI will **automatically add effects** to the color changes

JQUERY

jQuery UI Theme Roller

jQuery UI Theme Roller

You're not limited to the themes provided by the jQuery UI library

You can roll your **own** theme!

jQuery UI Theme Roller

You can roll your own theme!

jQuery provides a web page where you can set up colors, fonts and other details of how you want your pages to look

jQuery UI Theme Roller

You can roll your **own** theme!

Then just download and use the JS and CSS provided for that theme!

jQuery UI Theme Roller

ThemeRollerVideo

EXAMPLE 53

JQueryUICustom.html

JQueryUICustom.html

Example53Demo

JQueryUICustom.html

The code for this example is exactly the same as Example 51

JQueryUICustom.html

```
<link href="../../jquery-ui-1.12.0.loony/jquery-ui.css" rel="stylesheet">
<script src="../../lib/jquery.js"></script>
<script src="../../jquery-ui-1.12.0.loony/jquery-ui.js"></script>
```

Just include the right CSS and JS files that you've downloaded for this theme