

A close-up photograph of a person's hands resting on a red running track. The hands are positioned on a white starting line, with fingers spread. The track surface is a reddish-brown color with a pebbled texture. The lighting is bright, casting shadows on the track.

Getting started with jQuery

Gill Cleeren
@gillcleeren

Hi, I'm Gill!



Gill Cleeren

MVP and Regional Director
.NET Architect @ Ordina
Trainer & speaker

Twitter

@gillcleeren



gill@snowball.be

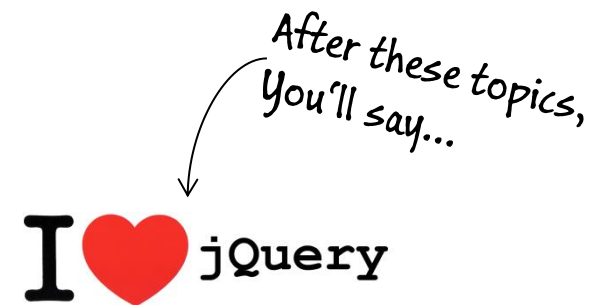
I'm a Pluralsight author!

- Courses on Windows 8, social and HTML5
- <http://gicl.me/mypscourses>




What we'll be looking at...

- Hello jQuery!!
- The 3 jQuery fundamentals
- Creating and manipulating elements
- Working with events
- Built-in animations and effects
- Talking to the server with Ajax
- Working with WebForms and MVC
- jQuery UI
- jQuery plugins
- Using the CDN



Throughout the session...

- You'll see some **I**  `jQuery`
- Goal: show a particular place where jQuery really stands out



HELLO JQUERY!

Hello jQuery!



- jQuery is
 - Most popular, cross-browser JavaScript library
 - Focusing on making client-side scripting of HTML simpler
 - Easy navigating the DOM
 - Handling events
 - Working with Ajax
 - Open-source, first released in 2006
 - Current release is 1.11 and 2.1
 - Same API
 - 2.X branch doesn't support IE 6, 7 and 8
 - Recommended to use 1.X for public sites

Why jQuery?

- Many JavaScript frameworks try bending the language out of its natural form
- jQuery aims at leveraging CSS, HTML and JavaScript
- Advantages
 - Lightweight
 - Easy to learn using familiar CSS syntax and intuitive

```
$('#something').hide().css('background', 'red').fadeIn();
```

- Many plugins available
- Easy to extend and compatible
- Support from Microsoft
- Rich community

*You can read
this, right?*

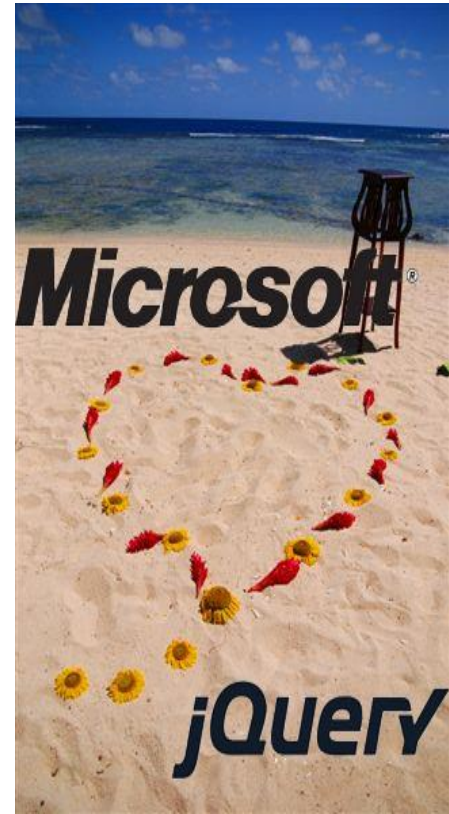
You are not alone!

Many LARGE companies use jQuery for their sites, including:



Microsoft and jQuery

- Included with Visual Studio
 - MVC
 - WebForms
- Microsoft is/was contributor to jQuery
 - Created templating, data linking and globalization (2010)
 - Not actively maintained now though
- CDN from Microsoft



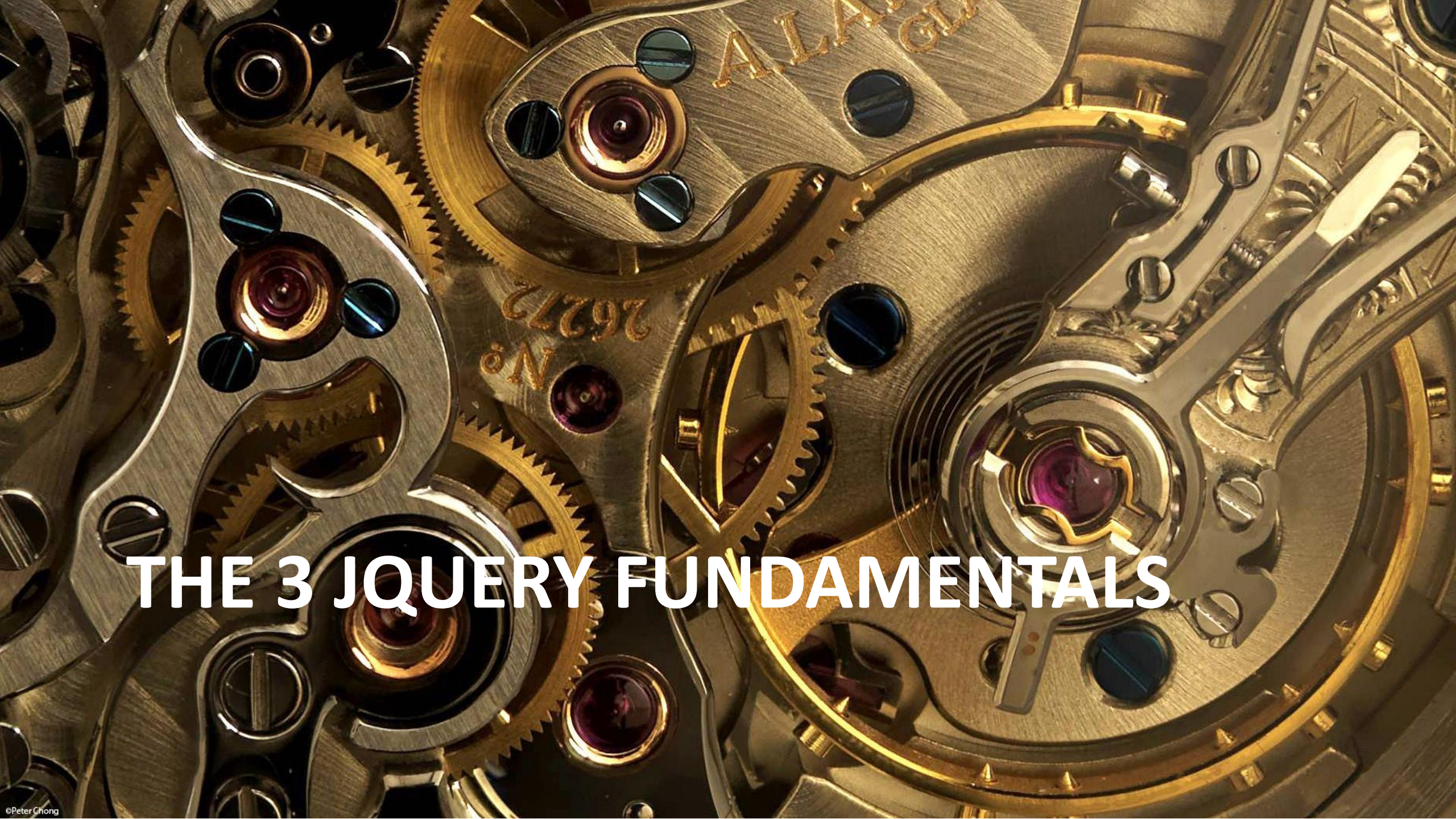
Script, don't get in my way!

- jQuery helps us writing *Unobtrusive JavaScript* code
- You don't want to mingle style with HTML
- Why would you want to mingle *behavior* with HTML?

```
<script type="text/javascript">
  window.onload = function() {
    document.getElementById('testButton').onclick = function() {
      document.getElementById('xyz').style.color = 'red';
    };
  };
</script>
```

- This will become a heavy job without jQuery!

*This code is
downloaded every time*



THE 3 JQUERY FUNDAMENTALS

Fundamentals #1: \$

- \$ function (aka jQuery() function) returns
 - A JavaScript object containing an array of DOM elements
 - In the order they were found in the document
 - Matching a specified selector (for example a CSS selector)
 - Known to mankind as a wrapper or wrapped set

```
$("#div.someClass").show();
```

*Finds all DIVs with
class someClass and
Sets them visible*

- It returns the same group of elements, can be chained

```
$("#div.someClass").show().addClass("SomeOtherClass");
```

*To the same set, this
adds another class*

Fundamental #2: the ready handler

- Script execution should wait until DOM elements are ready
 - You say: `window.onload`?
 - Sadly, this waits for everything to be loaded, including images etc
 - Script execution is *too late*
- Instead, we need to wait only until the DOM tree is created
 - Can be difficult in cross-browser situations
 - Easy-peasy with jQuery

```
$(document).ready(function() {  
    $("div.someClass").show();  
});
```

```
$(function() {  
    $("div.someClass").show();  
});
```



Fundamental #3: selectors

- At the core of jQuery lies its **selector** engine
- `$()` is heavily overloaded
 - Making a selection
 - Creating new HTML elements

Fundamental #3: selectors

- Most basic: CSS selectors

- Can be combined

```
$("#p a.someClass")
```

Selects all a's with someClass applied within a paragraph

```
$("#p a.someClass, div")
```

Also includes all DIVs on the page

- Child selector

```
$("#ul.someList > li > a")
```

Selects all links, directly in an LI, within an UL with someList class

- Attribute selector

```
$("#a[href*='http://www.snowball.be']")
```

Selects all links that contain a reference to my site

```
$("#span[class^='some']")
```

Select all SPANS whose class att starts with some

```
$("#span[class]")
```

Select all SPANS that have a class

Fundamental #3: selectors

- Position

`$("#a:first")`

Selects first A we can find on the page

`$("#div:even")`

Selects the "even" DIVs on a page

Selects the first cells within a table named someTable

`$("#table#someTable td:first-child")`

- Psuedo-classes (CSS filter selectors & custom selectors)

`$("#input:checked")`

Selects checked inputs (including the ones that weren't checked initially)

`$("#:password")`

Selects all INPUTS of type password

Selects all 'not-checked' inputs

`$("#input:not(:checked)")`

More selectors

- Full list at <http://www.w3.org/TR/css3-selectors/>

Pattern	Meaning
*	any element
E	an element of type E
E[foo]	an E element with a "foo" attribute
E[foo^="bar"]	an E element whose "foo" attribute value begins exactly with the string "bar"
E:nth-child(n)	an E element, the n-th child of its parent
E:first-child	an E element, first child of its parent
E:empty	an E element that has no children (including text nodes)
E:link E:visited	an E element being the source anchor of a hyperlink of which the target is not yet visited (:link) or already visited (:visited)
E > F	an F element child of an E element
E + F	an F element immediately preceded by an E element

Selecting elements using selectors

DEMO

Fundamental #3.1: creating elements

- `$('....')` selects an element \leftrightarrow `$('#')` creates an element

```
$(function(){  
    $('#<div>I'm off</div>')  
    .appendTo('body');
```

```
$(function(){  
    $('#<img>', {  
        src: 'someImage.jpg',  
        alt: 'Some nice image'  
    })  
    .appendTo('body');
```

Creating elements using \$

DEMO



CREATING AND MANIPULATING ELEMENTS

Working with the result of \$

- Once we have a wrapped set, we can go wild with it!
 - Handle the set as a whole
 - Work with individual elements

Working with the result of \$

- A wrapped set is like an array of elements, normal “array operations” can be done on it
 - Check the size

```
$('#a').size();
```

- Access an individual element

```
$('#a')[0];
```

```
$('#a').get(0);
```

- Loop over the elements

```
$('#img').each(function(n){  
    this.alt='image['+n+']';  
});
```


Working with the result of \$

- Set operations (continued)

- Add and remove elements

```
$("#a[class]").add("#a[href]");
```

- Filter elements

```
$("#a").filter("[href^='http://']");
```

- Remember that we are always returning the set

- Chaining is always possible!

```
$("#a[class]")  
    .add("#a[href]")  
    .filter("[href^='http://']")  
    ...;
```

Working with the set

DEMO

Attributes

- When we want to change how an element looks, we can change its attributes
- jQuery provides the attr() method
 - 2 variations based on number and types of parameters
 - Read a specified property from first element in wrapped set

```
$("#myImage").attr("alt");
```

- Set a property on all elements in the wrapped set (0 or more)

```
$('#myImage').attr('alt', 'Me in Paris');
```

Attributes (2)

- jQuery makes it easy to apply and remove CSS classes
 - `addClass()`, `removeClass()`, `toggleClass()` and `hasClass()`
- Changing individual CSS elements is supported
 - `css()` can be used to get or set CSS on an element

```
$('#mydiv').css("background-color", "yellow");
```

Working with elements

- `html()` can be used to get or set the content of an element

```
$('#mydiv').html();
```

Retrieves the HTML content of mydiv

- `text()` can retrieve combined textual content of all elements, including their children

- If the elements are form elements, we need to use `val()`

```
$('#input:checkbox:checked').val();
```

Retrieves the value from a checked checkbox


Working with attributes

DEMO



WORKING WITH EVENTS

Events: A bit of history

- Once upon a time, a browser called Netscape introduced an event model: DOM Level 0 Event Model
 - Creates event handlers as references to a function on a property
 - Not what we need if we want to create *Unobtrusive JavaScript*
 - Only one event handler per element for specific event
- Only got standardized until DOM Level 2 Event Model
 - Based on a system of event listeners (addEventListener)
 - IE decided to go its own way (attachEvent)
- Using event was a real mess because of browser dependencies
- jQuery comes to the rescue **I**  jQuery

jQuery events

- on() is where it all starts
 - Binds a function to any event on any DOM element
 - off() can be used to unbind a function from an event
 - Replaces the bind() and unbind()

```
$('#myimg')  
  .on('click',  
      function(event){alert('Hello World!');}  
  );
```

- Works in any browser, jQuery hides the details for us
 - Possible to bind more than one event handler for an event on one element
- one() removes itself after event handler executed



Events

DEMO

A vibrant night scene at a theme park. In the center, a large pagoda is illuminated with red and white lights. To the left, a roller coaster track is visible, lit with purple and blue lights. The sky is filled with a large, bright blue and white firework explosion. In the foreground, a body of water reflects the lights from the pagoda, roller coaster, and fireworks. The overall atmosphere is festive and celebratory.

BUILT-IN ANIMATIONS AND EFFECTS

Animations and effects

- Core jQuery has some basic effects
 - More are available in jQuery UI
 - Should be used with caution!
- Most basic ‘animation’ is hiding/showing an element
 - `hide()`: sets `display:none` on the element
 - `show()`: sets `display` to `inline/block`
 - `toggle()`: sets visible is hidden and vice-versa
- Methods are overloaded, accepting
 - Speed
 - Callback

Animations and effects (2)

- Elements can also be gradually added/removed
 - `slideDown()` and `slideUp()`
- Fading in is supported as well
 - `fadeIn()` and `fadeOut()`
- `animate()` is mother of all animations
 - Using 'target values' for style properties, jQuery will animate the transition

```
$('.someClass').animate({opacity:0.25},'slow');
```

*Change the opacity
with a slow animation*

Animations

DEMO



TALKING TO THE SERVER WITH AJAX

Ajax in the past

- When we were all young (in 1998), Microsoft introduced the ability to perform asynchronous requests from script (ActiveX)
- Later, other browsers implemented a standard, the XMLHttpRequest
 - IE6 uses an ActiveX object
- Result is that we need to do checks

```
if(window.ActiveXObject) {  
    xhr = new ActiveXObject("Microsoft.XMLHTTP");  
}  
else if (window.XMLHttpRequest) {  
    xhr = new XMLHttpRequest();  
}
```

Looks ugly, doesn't it?

- Again... jQuery to the rescue! **I**  **jQuery**

Ajax with jQuery

- Basic functionality to load content from a server-side resource:
 - load()
 - url
 - parameters: data to be passed (string, object...)
 - If provided, a POST is executed, otherwise a GET
 - callback (optional)

```
$('#someDiv')  
    .load('test.html',  
        function() {  
            alert('Load was performed.');        });
```

← Loads the HTML page

- Next to load, we can also use \$.get()/\$.getJSON() or \$.post()

Basic Ajax request with load()

DEMO

Ajax with jQuery

- If we need all control over the Ajax request we can get:
 - \$.ajax()
 - options: defines an object containing all the properties for the Ajax request
- List of options is huge!
 - \$.ajaxSetup()
 - options: defines an object containing all the properties for the Ajax request, becoming the default for Ajax requests

```
$.ajaxSetup({  
  type: 'POST',  
  timeout: 5000,  
  dataType: 'html'  
});
```

Ajax with jQuery

- Throughout the Ajax request, we can get feedback
 - Local events from the \$.ajax() call (callbacks)
 - Global events
 - Are broadcast to every element within the DOM, can be attached on any element
 - ajaxStart
 - ajaxSend
 - ajaxSuccess
 - ajaxError
 - ajaxComplete

More control with ajax()

DEMO

```
< it's not that simple />
```



WORKING WITH WEBFORMS AND MVC

jQuery Ajax, ASP.NET MVC and WebForms

- jQuery can work in harmony with ASP.NET MVC and WebForms
 - Sample ajax() call for WebForms

```
$.ajax({  
    type: "post",  
    contentType: "application/json; charset=utf-8",  
    dataType: "json",  
    url: "/Default.aspx/AddTask",  
    data: JSON.stringify(dto)  
});
```



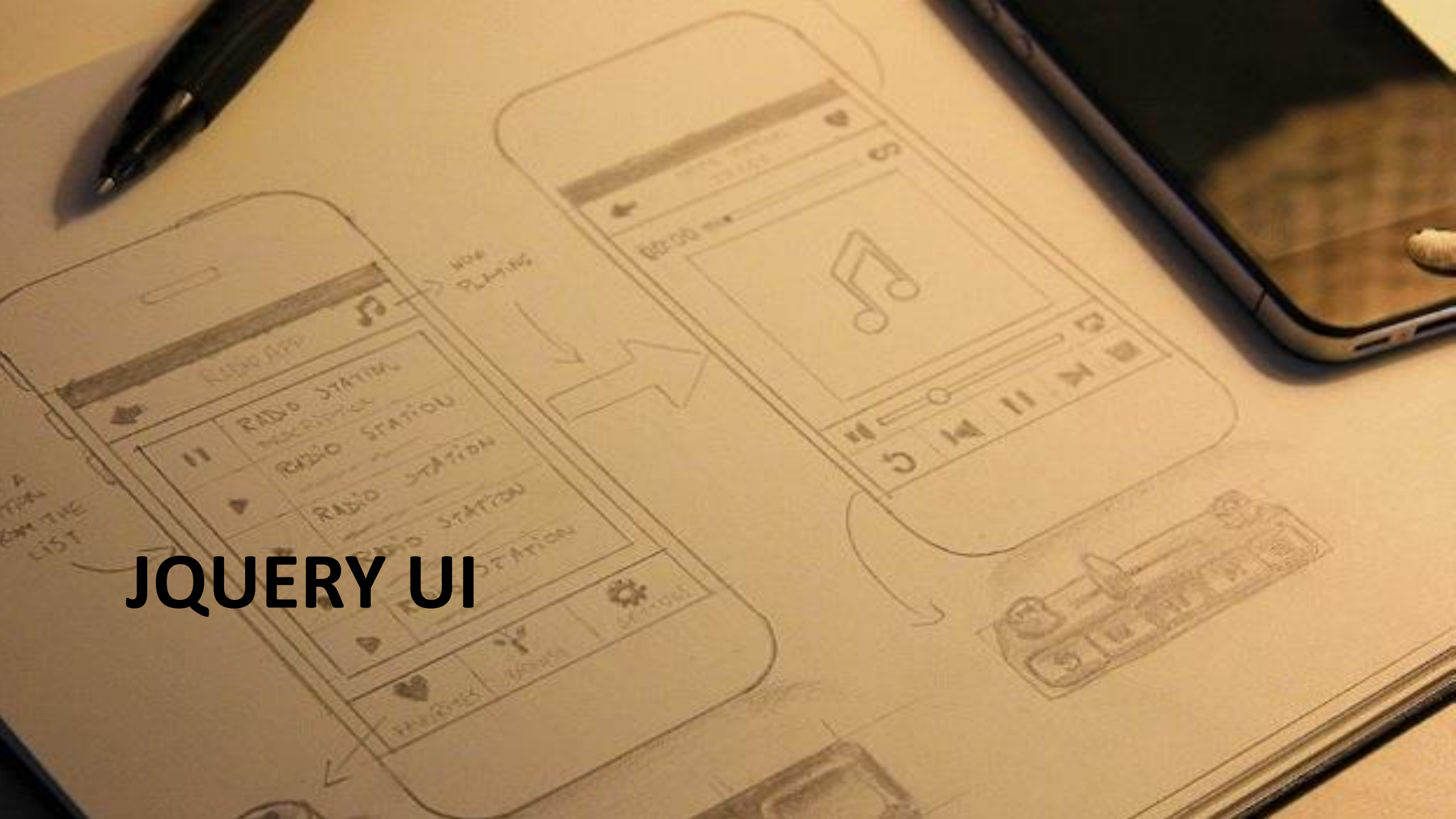
ASP.NET WebForms with jQuery

DEMO

ASP.NET MVC with jQuery

DEMO

JQUERY UI



jQuery UI

- Huge extension of jQuery, providing more UI capabilities
- Contains number of UI features we'd logically need
- Includes
 - Effects: more advanced than core effects
 - Interactions: drag and drop
 - Widgets (aka controls): date picker...
 - All can be themed
- Code included in jquery-ui.js

jQueryUI Themes

- Themes come with the download
 - It's **never** going to be OK for the marketing guys!
 - Options
 - Use it anyway
 - Use the ThemeRoller
 - Tweak a default or custom-created one
 - Create one yourself (Warning: the CSS is quite large)

Effects

- jQuery core contains some basic effects
- Based on the **effect**(type, options, speed, callback) method
 - Has several animation types such as puff, highlight and shake (even explode exists)
 - Also allows to do animations with colors (not possible with animate())
 - backgroundColor, color...
- Visibility methods (show()...) are extended
- Class methods (addClass()...) are extended
- position() method is added for advanced positioning

```
$('#someElement').position({  
  my: 'top center',  
  at: 'bottom right',  
  of: '#someOtherElement'});
```

Effects

DEMO

Interactions

- Interactions focus on allowing users to directly interact with elements, which isn't possible with standard HTML controls
 - They add advanced behaviors to our pages related to mouse interactions
- Available interactions:
 - Dragging
 - Dropping
 - Sorting
 - Resizing
 - Selecting

Dragging

- Easy-peasy (again) with jQuery
- `draggable()` is your friend (heavily overloaded once again)
 - Allows making elements draggable, possible with options (opacity...)

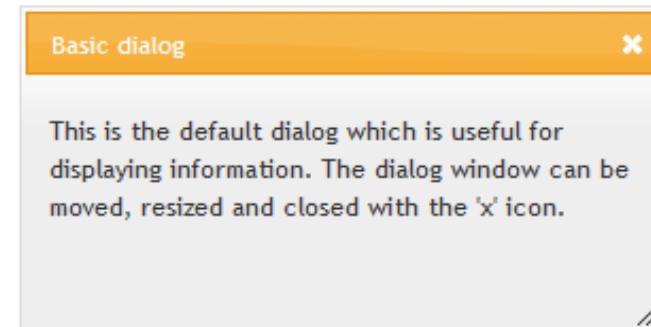
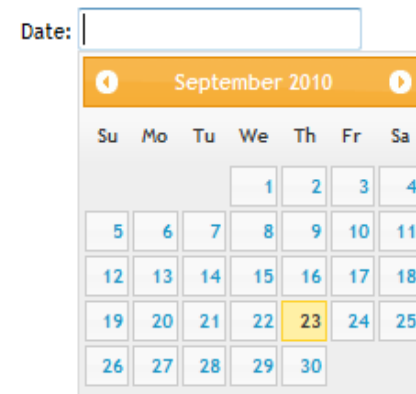
```
$('#someDiv').draggable();
```


Dragging and drop

DEMO

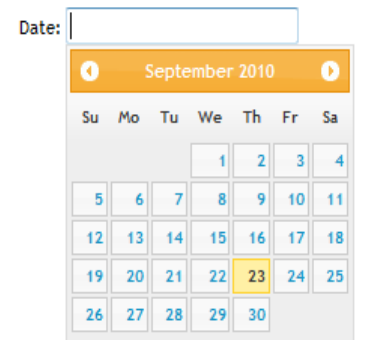
Widgets: controls on steroids

- New controls (based on existing ones)
- Contents
 - Buttons
 - Sliders
 - Progress bars
 - Autocompletion
 - Date picker
 - Tabs
 - Accordion
 - Dialog box



Date picker

- Have you noticed that entering dates is a difficult thing for end users? Some will always get it wrong!
- jQuery UI's DatePicker can help
 - datepicker() creates the control for you
 - Has numerous options, mostly defaults will do



Widgets in action

DEMO



JQUERY PLUGINS

Something missing in jQuery?

- 2 options:
 - Use an existing plugin
 - Google code (code.google.com): *going to be retired soon!*
 - GitHub
 - jQuery plugin (not active anymore)
 - Write a plugin yourself
 - Custom utility function
 - Create wrapper functions

Using a plugin

DEMO

Writing your own plugins

- Write a plugin to add it yourself!
 - Possible to write your own utility functions and wrapper methods
- Creating new wrapper methods:
 - Add the method as a property on the fn object in the \$ namespace

```
$.fn.wrapperFunctionName = function(params){function-body};
```

```
(function($){  
  $.fn.setToRed = function() {  
    return this.css('color','red');  
  };  
})(jQuery);
```

We are passing jQuery to a function that has # as parameter. This way, # will inside this function, always be the jQuery #

Writing a plugin

DEMO

A digital globe with a bright sunburst in the center and a network of glowing nodes and lines representing a CDN.

USING THE CDN

Where to get your stuff?

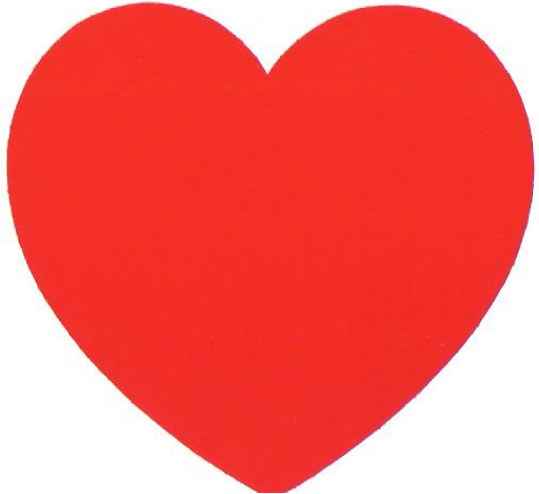
- Use a CDN?
 - Microsoft
 - Google
- Put scripts locally as well with a fallback mechanism

```
<script type="text/javascript"
    src="http://ajax.microsoft.com/ajax/jquery/jquery-1.4.2.min.js">
</script>
<script type="text/javascript">
if (typeof jQuery == 'undefined')
{
    document.write(unescape("%3Cscript src='/Scripts/jquery-1.4.2.min.js'
        type='text/javascript'%3E%3C/script%3E"));
}
</script>
```

Summary

- Where does all the (l) for jQuery come from?
 - Light-weight library that uses JavaScript as JavaScript, relying on CSS
 - Cross-browser compatible, hides the details (ready handler)
 - Easy eventing model
 - Can work with MVC & WebForms
 - Easily extensible to fit your needs, tons of plugins already available

So I hope you now say too...

I  **jQuery**



THANK

YOU

AND

HAVE

FUN

THANKS!

Q&A



A close-up photograph of a person's hands resting on a red running track. The hands are positioned on a white starting line, with fingers spread. The track surface is a reddish-brown granular material. The lighting is bright, casting shadows on the track.

Getting started with jQuery

Gill Cleeren
@gillcleeren