Mobile

Lecture 5 – Jquery

Semester I

- **Startup**

- Selectors

- Modifying the DOM

- Events

- Ajax

What's Jquery

- JavaScript library

- Cross-browser support

- Simple selecting of HTML elements

- Simple event handling

- Simple animation

- Ajax calls to server

- A lot of plugins available

Jquery file and vsdoc

- https://jquery.com/download/

- Using DCN:

  - <script src="//code.jquery.com/jquery-1.11.3.min.js"></script>

  - <script src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>

- Uncompressed version :

  - http://code.jquery.com/jquery-[version].js

- Minified version for release:

  - http://code.jquery.com/jquery-[version].min.js

- IntelliSense :build in since VS2013

- Do not use the CDN links as you sometimes don't have an internet link

```
<head>
    <script src="//code.jquery.com/jquery-1.11.3.min.js"></script>
    <script src="//code.jquery.com/jquery-migrate-1.2.1.min.js"></script>
</head>
```

- Using the script at the end of the body, after the DOM is build

```
<body>
    <div id="a"></div>
    <script type="text/javascript">
        $('#a').text('nir');
    </script>
</body>
```

$ - the jquery function

5

©nir chen

$(document).ready

- Ready – when the DOM is loaded and ready to use (before the images are loaded)

```
<head>
    <script type="text/javascript" src="scripts/jquery
    <script>
        $(document).ready(func1); //callback function

        function func1() {
            $('#a').text('nir');
        }
    </script>
</head>
```

- Or simpler…

```
<head>
    <title>jqeury</title>
    <script type="text/javascript" src="scripts/jquery-1.9.1
    <script>
        $(document).ready(function () { //anonymous function
            $('#a').text('nir');
        });
    </script>
</head>
```
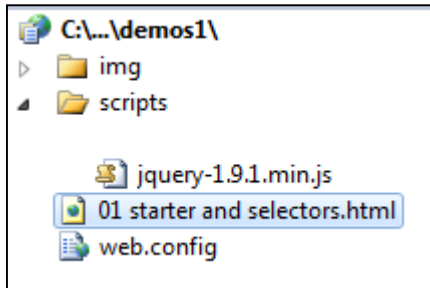
Agenda

- Startup

- **Selectors**

- Modifying the DOM

- Events

- Ajax

Selectors

C:\...\demos1\
- img
- scripts

    jquery-1.9.1.min.js
    01 starter and selectors.html
    web.config

- Selectors enables you to select one or more html element\tags in order to perform a manipulation on it.

- Can be done using JS but not as simple as with Jquery

- $(selctorExpression) = jQuery(selctorExpression)

©nir chen

Selecting by Tag Name

```
 alert($('div').text());
…
<div id="firstName">Dexter</div>
<div id="lastName">Morgan </div>
```

DexterMorgan

- Returns a <u>collection</u> of all the divs

- .text() : returns the text value of the element

Selecting multiple elements

```
alert($('div,p,span').text());
…
 <div id="firstName">Dexter</div>
<div id="lastName">Morgan</div>
<p>Debra</p>
<p>Morgan</p>
<span>Harry</span>
<span>Morgan</span>
```
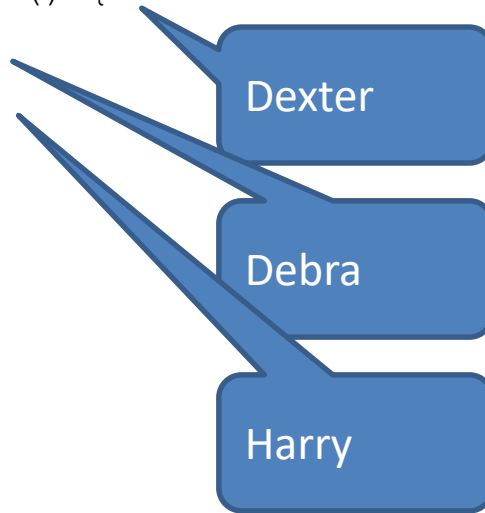
DexterMorganDebraMorganHarryMorgan

- Use , to select more than one element. (not necessary by tag name)

- Here: all the divs, ps and spans

Selecting Descendants elements

```
alert($('div p,p span').text());
…
<div id="firstName">
    <p>Dexter</p>
</div>
<div id="lastName">Morgan</div>
<p>
    <a>
        <span>Debra</span>
    </a>
</p>
<p>Morgan</p>
<p>
    <span>Harry</span>
</p>
<span>Morgan</span>
```

DexterDebraHarry

- $(ancestor descendant) select the descendant of this ancestor. (not necessarily by tag name)

- Not necessarily immediate descendant

- Here: all the p within a div and all span within a p

11 ©nir chen

.each()

```
$('div p,p span').each(function () {
    alert($(this).text());
});
…
<div id="firstName">
    <p>Dexter</p>
</div>
<div id="lastName">Morgan</div>
<p>
    <a>
        <span>Debra</span>
    </a>
</p>
<p>Morgan</p>
<p>
    <span>Harry</span>
</p>
<span>Morgan</span>
```

Dexter

Debra

Harry

- Will iterate through all the elements and perform the function on each of them. 3 separate alerts!

- *this* – is the element selected. Here we use it as jquery object to get the .text() function

©nir chen

Selecting by Id

```
$('#lastName').css('color', 'red');
…
<div id="firstName">
    <p>Dexter</p>
</div>
<div id="lastName">Morgan</div>
<p>
    <a>
        <span>Debra</span>
    </a>
</p>
<p>Morgan</p>
<p>
    <span>Harry</span>
</p>
<span>Morgan</span>
```

Change the style

Dexter
Morgan
Debra
Morgan
Harry
Morgan

- # - Returns the element with the specified ID

- The fastest way to find an element!

Selecting start from a specific place

```javascript
$('p', '#firstName').each(function () {
    $(this).css('color', 'yellow');
});
…
 <div id="firstName">
     <p>Dexter</p>
 </div>
 <div id="lastName">Morgan</div>
 <p>
     <a>
         <span>Debra</span>
     </a>
 </p>
 <p>Morgan</p>
 <p>
     <span>Harry</span>
 </p>
 <span>Morgan</span>
```
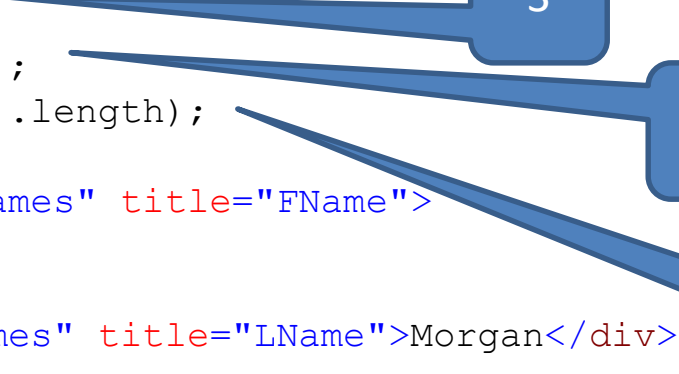
- *$('element1','element2')= $('element2 element1')* - Returns *element1* within the *element2*

©nir chen

Selecting by class

```javascript
$('.names').each(function () {
    alert($(this).text());
});

//much more complicated
//not all browser supports
//only HTML5
//dont have intellisense
var namesJS = document.getElementsByClassName('names');
for (var i = 0; i < namesJS.length; i++) {
    alert(document.getElementsByClassName('names')[i].innerText);
}
…
<div id="firstName" class="names">
        <p>Dexter</p>
    </div>
    <div id="lastName" class="names">Morgan</div>
    <p>
        <a><span class="spans">Debra</span> </a>
    </p>
    <p class="names">Morgan</p>
    <p>
        <span class="spans">Harry</span>
    </p>
    <span class="spans">Morgan</span>
```

Dexter

Morgan

Morgan

Dexter

Morgan

Morgan

©nir chen

- **.** - Returns the element with the specified class name

- You can see how simple and convenient is Jquery comparing to JS!!!

- .Length : returns how many elements are in the collection

```
alert($('div.names').length);
…
…
<div id="firstName" class="names">
        <p>Dexter</p>
    </div>
    <div id="lastName" class="names">Morgan</div>
    <p>
        <a><span class="spans">Debra</span> </a>
    </p>
    <p class="names">Morgan</p>
    <p>
        <span class="spans">Harry</span>
    </p>
    <span class="spans">Morgan</span>
```

2

Selecting by attribute

```
alert($('[title]').length);                      3
alert($('div[title]').length);
alert($('div[title="LName"']').length);          2
…
<div id="firstName" class="names" title="FName">
    <p>Dexter</p>
</div>                                            1
<div id="lastName" class="names" title="LName">Morgan</div>
<p>
    <a><span class="spans">Debra</span> </a>
</p>
<p class="names" title="pt">Morgan</p>

…
```

- *[attribute]* - Returns the element with the specified attribute

- *[attribute="value"]* - Returns the element with the specified attribute=value

©nir chen

```
alert($('input[type="text"]').length);

…
<p>first name:<input type="text" id="txtFName"/></p>
<p>last name:<input type="text" id="txtLName"/></p>
<p><input type="button" id="btn1" value="push"/></p>
<p><input type="checkbox" id="chk1"/>choose</p>

…
```

2

- Only the textboxes

Selecting by input

```
alert($(':input').length);
alert($(':input[type="text"]').length);
alert($('select').val());
$('textarea').val($('textarea').val() + " is great!");

…
<p>first name:<input type="text" id="txtFName"/></p>
<p>last name:<input type="text" id="txtLName"/></p>
<p><input type="button" id="btn1" value="push"/></p>
<p><input type="checkbox" id="chk1"/>choose</p>
<p><textarea rows="3" cols="10">season 3</textarea></p>
<p><select>
    <option value=""></option>
    <option value="LaGuerta">LaGuerta</option>
    <option value="Batista ">Batista </option>
</select></p>
<p><img src="img/dex.jpg"/></p>
```

6

2

LaGuerta

Will add the "is graet!" to the textarea

- `':input'` - returns all the input elements like: text, button, checkbox, radio, select, image, textarea, …

- `':input[type="value"]'` - Returns the element with the specified type=value. Value = text\radio\checkbox. But not textarea, img,…

- Pay attention that the alerts are fired before the img is shown because the ready() is before the images are loaded to the DOM! This is way only 6 inputs and not 7

- Acctually `'input[type="text"]'` is more efficient than `':input[type="text"]'` because it starts with 4 to filter from and not 6!!

Ohad chen

.contains()

```
alert($('div:contains("Mor")').length);
…
<div id="Div1"  >Hello Mor</div>
<div id="firstName" class="names" title="FName">
    <p>Dexter</p>
</div>
<div id="lastName" class="names" title="LName">Morgan</div>

…
```
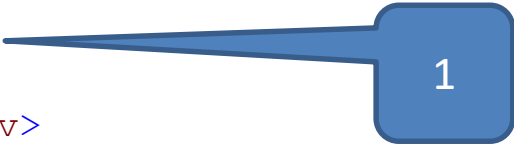
2

- *element:contains(string)* - Returns the element that contains a certain string in it's text

©nir chen

:even \ :odd

```
$('div:even').each(function () {
    $(this).css('background-color', 'red');
});
alert($('div:odd').length);
…
<div id="Div1"  >Hello Mor</div>
<div id="firstName" class="names" title="FName">
    <p>Dexter</p>
</div>
<div id="lastName" class="names" title="LName">Morgan</div>

…
```

1

| Hello Mor |
| Dexter |
| Morgan |
| Debra |

- *element:even* - Returns the even appearance of that elements. 0,2,4…

- *element:odd* - Returns the odd appearance of that elements. 1,3,5,…

©nir chen

:first-child

```
alert($('span:first-child').length);
…
<p>
    <a><span class="spans">Debra</span> </a>
</p>
<p class="names" title="pt">Morgan</p>
<p>
    <span class="spans">Harry</span>
    <span class="spans">Carry</span>
</p>
<span class="spans">Morgan</span>
```

2

- *element:first-child* - Returns the element if it is a first child of another element

:eq()

```
alert($('span:eq(0)').text());
alert($('span:eq(2)').text());
…
<p>
    <a><span class="spans">Debra</span> </a>
</p>
<p class="names" title="pt">Morgan</p>
<p>
    <span class="spans">Harry</span>
    <span class="spans">Carry</span>
</p>
<span class="spans">Morgan</span>
```

Debra

Carry

- *element:eq(number)* - Returns the element which is number in the order in the whole DOM. Returns only one element.

- Starts from 0.

- Cant use n variable

:nth-child()

```
alert($('span:nth-child(1)').text());
alert($('span:nth-child(1)').length);
alert($('span:nth-child(2)').length);

…
<p>
    <a><span class="spans">Debra</span> </a>
</p>
<p class="names" title="pt">Morgan</p>
<p>
    <span class="spans">Harry</span>
    <span class="spans">Carry</span>
</p>
<span class="spans">Morgan</span>
```

DebraHarry

2

1

- *element:nth-child(number)* - Returns the element which is the nth-child of some parent. Can return more than one element.

- Starts from 1.

©nir chen

:nth-child()

```javascript
$('span:nth-child(2n)').each(function () {
    alert($(this).text());
});
$('span:nth-child(2n+1)').each(function () {
    alert($(this).text());
});
…
<p>
    <a><span class="spans">Debra</span> </a>
</p>
<p class="names" title="pt">Morgan</p>
<p>
    <span class="spans">Harry</span>
    <span class="spans">Carry</span>
</p>
<span class="spans">Morgan</span>
```

Carry

Morgan

Debra

Harry

- Can use n variable like: 2n, 2n+1, 3n…

```
alert($('div[id^="fi"]').length);
…
<div id="fid"  >Start</div>
<div id="Div1"  >Hello Mor</div>
<div id="firstName" class="names" title="FName">
    <p>Dexter</p>
</div>
<div id="lastName" class="names" title="LName">Morgan</div>
```

2

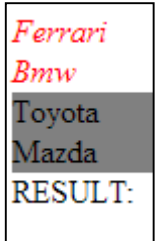- *element[attribute^="value"]* - Returns elements the their attribute's value begins with "value"

©nir chen

$= ends with

```
alert($('input[id$="1"]').length);              2
…
<p>first name:<input type="text" id="txtFName"/></p>
<p>last name:<input type="text" id="txtLName"/></p>
<p><input type="button" id="btn1" value="push"/></p>
<p><input type="checkbox" id="chk1"/>choose</p>
```

- *element[attribute$="value"]* - Returns elements the their attribute's value ends with "value"

©nir chen

*= contains

```
alert($('div[id*="Na"]').length);
…
<div id="fid"  >Start</div>
<div id="Div1"  >Hello Mor</div>
<div id="firstName" class="names" title="FName">
    <p>Dexter</p>
</div>
<div id="lastName" class="names" title="LName">Morgan</div>
```

2

- *element[attribute*="value"]* - Returns elements the their attribute's value contains "value"

Agenda

- Startup

- Selectors

- **Modifying the DOM**

- Events

- Ajax

©nir chen

The example

```
<style>
    .sportCar
    {
        color: red;
        font-style: italic;
    }
    .familyCar
    {
        background-color: Gray;
        font-family: @Batang;
    }
</style>
…

<div class="sportCar">Ferrari</div>
<div class="sportCar">Bmw</div>
<div class="familyCar">Toyota</div>
<div class="familyCar">Mazda</div>
<label>RESULT:</label><div id="result"></div>
```

```
$(document).ready(function () {
    $('.sportCar, .familyCar').each(function (index) {
        $('#result').html($('#result').html() + "</br>" + index +": "+
$(this).html());
    });
});
…
<div class="sportCar">Ferrari</div>
<div class="sportCar">Bmw</div>
<div class="familyCar">Toyota</div>
<div class="familyCar">Mazda</div>
<label>RESULT:</label><div id="result"></div>
```

Ferrari
Bmw
Toyota
Mazda
RESULT:

0: Ferrari
1: Bmw
2: Toyota
3: Mazda

- Will iterate through all the chosen elements and perform the function on each of them.

- `this` – is the element selected.

- `Index` – the current iteration's index. Starting with 0.

- Instead of using `this` we can use the second parameter - `element`

```
$(document).ready(function () {
    $('.sportCar, .familyCar').each(function (index, element) {
        $('#result').html($('#result').html() + "</br>" + index + ": " +
$(element).html());
    });
});
```

©nir chen

each() –efficiency consideration!!!

```
$(document).ready(function () {
    $('.sportCar, .familyCar').each(function (index) {
        $('#result').html($('#result').html() + "</br>" + index +": "+
$(this).html());
    });
});
```

- Better to cache the `$('#result')` in order to prevent the <u>searching</u> of it in every loop iteration!

```
$(document).ready(function () {
    var output = $('#result');
    $('.sportCar, .familyCar').each(function (index) {
        output.html(output.html() + "</br>" + index + ": " + $(this).html());
    });
});
```

- Now we <u>update</u> the DOM every iteration so to do better we need to use a local variable and update the DOM at the end just once!

```
var output = '';
$('.sportCar, .familyCar').each(function (index ) {
    output += "</br>" + index + ": " + $(this).html();
});
$('#result').html(output);
```

©nir chen

Attribute

```
$('div').each(function () {
    this.title = 'Hello';
});
…
<div class="sportCar">Ferrari</div>
<div class="sportCar">Bmw</div>
<div class="familyCar">Toyota</div>
<div class="familyCar">Mazda</div>
<label>RESULT:</label><div id="result"></div>
```
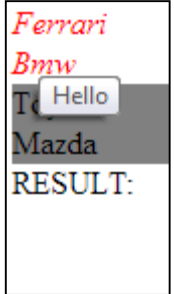
- Here we use the this in order to get to the attribute BUT there is no dynamic (on the $()) attribute, instead we can use the `.attr()` function as follows (attribute not properties like innerText)

```
$('div').each(function () {
    $(this).attr('title', 'Hello');
});
```

- $(element).attr('attribute', 'value'); //to set

- $(element).attr('attribute'); //to get

©nir chen

Attribute cont'

- We could shorten the last modification because the `attr()` will work on all the collection selected any way:

```
$('div').attr('title', 'Hello');
$('div').attr('style', 'color:green');
```

- Multiple modifications are done using JSON

- The style is one of the attributes

```
$('div').attr(
{
    title: 'Hello',
    style: 'color:green;'
});
```

JSON object looks like this:

```
{
    property: 'value',
    property: 'value'
}
```
JSON has also nested object (not here)
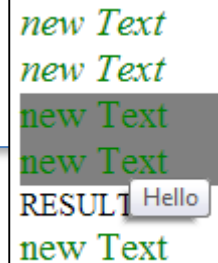
Ferrari
Bmw
Toyota
Mazda
RES Hello

34   ©nir chen

Chaining functions

- After calling a jquery function on an element we can chain another function or more than one like this:

```
$('div').attr('title', 'Hello')
        .attr('style', 'color:green')
        .css('font-size', '20px')
        .text('new Text');
```

new Text
new Text
new Text
new Text
RESULT Hello
new Text
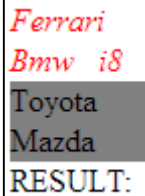
append()

- We can append elements using the *append()* function <u>after</u> a specific element

```
$('#bmw').append('&nbsp&nbsp&nbsp<span>i8</span>');

...
<div class="sportCar" id="ferrari">Ferrari</div>
<div class="sportCar" id="bmw">Bmw</div>
<div class="familyCar">Toyota</div>
<div class="familyCar">Mazda</div>
<label>RESULT:</label><div id="result"></div>
```

Ferrari
Bmw i8
Toyota
Mazda
RESULT:

©nir chen

prepend()

- We can prepend elements using the *prepend()* function <u>before</u> a specific element

```
$('#ferrari').prepend('<p>Sport Cars</p>');

...
<div class="sportCar" id="ferrari">Ferrari</div>
<div class="sportCar" id="bmw">Bmw</div>
<div class="familyCar">Toyota</div>
<div class="familyCar">Mazda</div>
<label>RESULT:</label><div id="result"></div>
```

©nir chen

wrap()

- We can wrap elements using the *wrap()* function <u>arround</u> a specific element

```
$('.sportCar,.familyCar').wrap('<p>');

...
<div class="sportCar" id="ferrari">Ferrari</div>
<div class="sportCar" id="bmw">Bmw</div>
<div class="familyCar">Toyota</div>
<div class="familyCar">Mazda</div>
<label>RESULT:</label><div id="result"></div>
```


Ferrari

Bmw

Toyota

Mazda

RESULT:

remove()

- We can remove elements using the *remove()* function

```
$('.sportCar').remove();

...
<div class="sportCar" id="ferrari">Ferrari</div>
<div class="sportCar" id="bmw">Bmw</div>
<div class="familyCar">Toyota</div>
<div class="familyCar">Mazda</div>
<label>RESULT:</label><div id="result"></div>
```

Toyota
Mazda
RESULT:

©nir chen

.css()

- The $css()$ function (already saw it) helps us to modify the style.

```
$('div').css("color", "yellow");
```

Ferrari
Bmw
Toyota
Mazda
RESULT:

- We can use here the JSON object again

```
$('div').css(
    {
        'font-size': '20px',
        'border': '2px solid black',
        'margin': '5px'
    }
);
```

Ferrari
Bmw
Toyota
Mazda
RESULT:

©nir chen

addClass()

- We can add a new class or more (using spaces) than just one into an element using the *addClass()* function

```
$('.sportCar').addClass('Highlight');

...
<div class="sportCar" id="ferrari">Ferrari</div>
<div class="sportCar" id="bmw">Bmw</div>
<div class="familyCar">Toyota</div>
<div class="familyCar">Mazda</div>
<label>RESULT:</label><div id="result"></div>
```
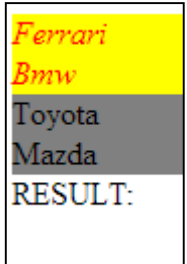
removeClass()

- We can remove a class or more (using spaces) than just one from an element using the *removeClass('className')* function

```
$('.familyCar').removeClass('familyCar');

...
<div class="sportCar" id="ferrari">Ferrari</div>
<div class="sportCar" id="bmw">Bmw</div>
<div class="familyCar">Toyota</div>
<div class="familyCar">Mazda</div>
<label>RESULT:</label><div id="result"></div>
```

Ferrari
Bmw
Toyota
Mazda
RESULT:

- *removeClass()* without a specific className will remove all the classes of the element

©nir chen

hasClass()

- We can ask about a class or more (using spaces) than just if it is a part of an element using the *hasClass()* function

```
.Highlight
{
    background-color:Yellow;
}
…
$(document).ready(function () {
…
});

function anchorClick(div) {
    if ($(div).hasClass('Highlight')) {
        $(div).removeClass('Highlight');
    }
    else {
        $(div).addClass('Highlight');
    }
}

...
<a  href="#" onclick="anchorClick(this)">just an anchor</a>
```

©nir chen
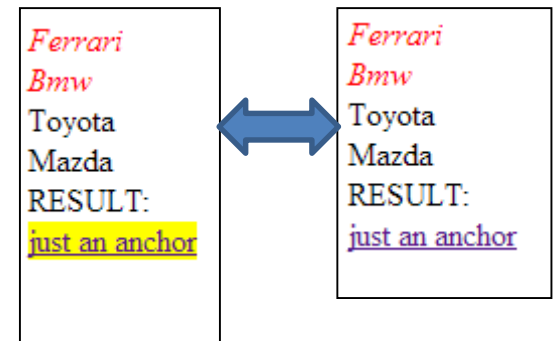
toggleClass()

- We can toggle (switch on and off repeatedly) a class or more (using spaces) than just one on an element using the *toggleClass()* function

```css
.Highlight
{
    background-color:Yellow;
}
```

```javascript
$(document).ready(function () {
…
});


function anchorClickToggle(div) {
    $(div).toggleClass('Highlight');
}


...
<a  id="a" href="#" onclick="anchorClickToggle(this)">just another anchor</a>
```
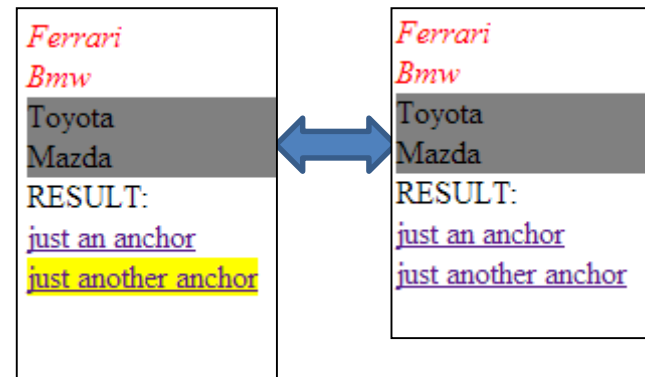
Agenda

- Startup

- Selectors

- Modifying the DOM

- Events

- Ajax

JavaScript events

- In JS we would have to script a different code for IE v.8 and earlier vs. all the other browsers. Jquery is cross platform and much more compact and simple.

```
<input id="myButton" type="button" value="PUSH" />
<script type="text/javascript">
    //JS event
    var button = document.getElementById('myButton');
    if (document.addEventListener) { //all the other browsers!
        button.addEventListener('click', function () { alert('clicked!'); },
false);
    }
    else { //IE v8 or earlier !
        button.attachEvent('onclick', function () { alert('IE v8 or earlier
clicked!'); });
    }
 </script>
```

Wiring the old fashion

- You can wire the events using the markup, but then its more difficult to find it on the page. Wiring all through the jquery makes it easy to find all in the same place!

```
<input id="myButton" type="button" value="PUSH" onclick="clickFunction()"/>
…
<script type="text/javascript">
        function clickFunction() {
            alert('click function');
        }
</script>
```

click()
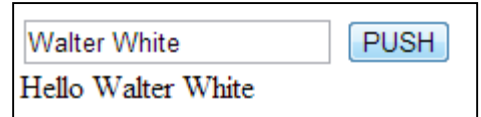
- click() – a shortcut for *.on("click", handler)*

```
$(document).ready(function () {
    wireEvents();
});

function wireEvents(){
    $('#myButton').click(function(){
        $('#res').text("Hello " + $('#txtName').val());
    });
}
…
<input id="txtName" type="text" />
<input id="myButton" type="button" value="PUSH"/>
<div id="res"></div>
```

| Walter White | PUSH |

Hello Walter White

©nir chen
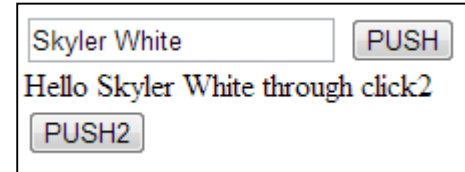
- Can be used (normally) as definition for the action taken when we click an element but can be also used to call another click function

```
$('#myButton2').click(function () {
    $('#myButton').click();
    $('#res').text($('#res').text() + " through click2");
});

…
<input id="txtName" type="text" />
<input id="myButton" type="button" value="PUSH"/>
<div id="res"></div>
<input id="myButton2" type="button" value="PUSH2"/>
```

Skyler White   PUSH
Hello Skyler White through click2
PUSH2

©nir chen

change()

- change() – a shortcut for `.on( "change", handler )`

- Works on textbox,textarea and select

```
$('#selBB').change(function () {
    $('#res').text($(this).val());
});
$('#txtName').change(function () {
    $('#res').text($(this).val());
});
$('#txtarea').change(function () {
    $('#res').text($(this).val());
});

…
<input id="txtName" type="text" />
<input id="myButton" type="button" value="PUSH"/>
<input id="myButton2" type="button" value="PUSH2"/>
<p><select id="selBB">
    <option>Jesse Pinkman</option>
    <option>Hank Schrader</option>
</select></p>
<p><textarea id="txtarea" rows="2" cols="22"></textarea></p>
<div id="res"></div>
```

mouseenter\mouseleave\mouseup\mousedown

- mouseenter() – a shortcut for *.on("mouseenter", handler)*

- mouseleave() – a shortcut for *.on("mouseleave", handler)*

- mouseup() – a shortcut for *.on("mouseup", handler)*

- mousedown() – a shortcut for *.on("mousedown", handler)*

```
$('#res').mouseenter(function () {
    $('#res').text("mouse has entered the div!");
    $(this).css('cursor', 'crosshair');
}).mouseleave(function () {
    $('#res').text("mouse has leaved the div!");
});

…
<input id="txtName" type="text" />
<input id="myButton" type="button" value="PUSH"/>
<input id="myButton2" type="button" value="PUSH2"/>
<p><select id="selBB">
    <option>Jesse Pinkman</option>
    <option>Hank Schrader</option>
</select></p>
<p><textarea id="txtarea" rows="2" cols="22"></textarea
<div id="res"></div>
```

©nir chen
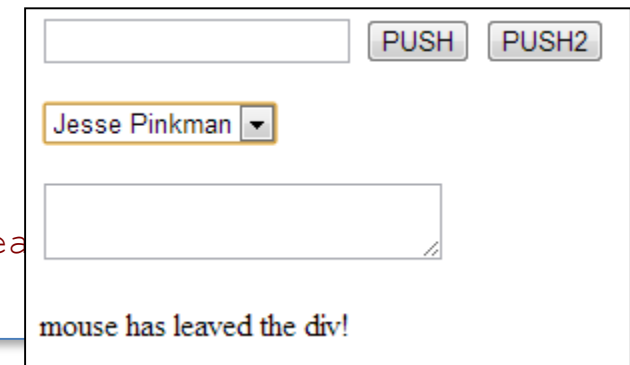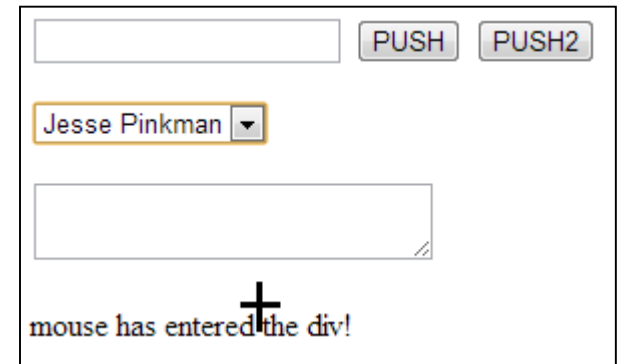
- We can use the event argument to get some information like:

  - `pageX`

  - `pageY`

  - `The id of the element chosen`

```
$('#res').mouseenter(function () {
    $(this).text("mouse has entered the div!");
    $(this).css('cursor', 'crosshair');
}).mouseleave(function () {
    $(this).text("mouse has leaved the div!");
}).mousedown(function (e) {
    $(this).text("mouse was pushed down on id=" + $(e.target).attr('id') +
                 " in x="+ e.pageX +" and y=" + e.pageY);
});

…
<p><select id="selBB">
    <option>Jesse Pinkman</option>
    <option>Hank Schrader</option>
</select></p>
<p><textarea id="txtarea" rows="2" cols="22"></textarea></p>
<div id="res"></div>
```

mouse was pushed down on id=res in x=127 and y=159

©nir chen

on() and multiple bindings

- We can use the *.on* function to bind several events together

- A bit more efficient that calling the shortcuts – one function call less

- This way we can bind an event dynamically

- Downside is that we can misspell the string

- We can use the event args to manipulate a specific event separately

```javascript
$('#res').on('mouseenter mouseleave mousedown', function (e) {
    $(this).text("mouse has entered or leaved!");
    if(e.type=='mousedown')
     $(this).text($(this).text() + " mouse was pushed down on id=" +
$(e.target).attr('id') +
                    " in x="+ e.pageX +" and y=" + e.pageY);
});

…
<p><select id="selBB">
    <option>Jesse Pinkman</option>
    <option>Hank Schrader</option>
</select></p>
<p><textarea id="txtarea" rows="2" cols="22"></textarea></p>
<div id="res"></div>
```

mouse has entered or leaved!

mouse has entered or leaved! mouse was pushed down on id=res in x=174 and y=157

©nir chen

off()

- We can use the `.off` function to unbind a specific event

```javascript
$('#res').on('mouseenter mouseleave mousedown', function (e) {
    $(this).text("mouse has entered or leaved!");
    if(e.type=='mousedown')
     $(this).text($(this).text() + " mouse was pushed down on id=" +
$(e.target).attr('id') +
                    " in x="+ e.pageX +" and y=" + e.pageY);
});
$('#myButton').click(function () {
    $('#res').off('mousedown');
});
…
<input id="myButton" type="button" value="PUSH"/>
…
<p><select id="selBB">
    <option>Jesse Pinkman</option>
    <option>Hank Schrader</option>
</select></p>
<p><textarea id="txtarea" rows="2" cols="22"></textarea></p>
<div id="res"></div>
```

On mousedown

mouse has entered or leaved!

mouse has entered or leaved! mouse was pushed down on id=res in x=174 and y=157

After myButton was clicked and on mousedown

mouse has entered or leaved!

- We can use the .*off* function to unbind all events

```javascript
$('#res').on('mouseenter mouseleave mousedown', function (e) {
    $(this).text("mouse has entered or leaved!");
    if(e.type=='mousedown')
     $(this).text($(this).text() + " mouse was pushed down on id=" +
$(e.target).attr('id') +
                    " in x="+ e.pageX +" and y=" + e.pageY);
});
$('#myButton').click(function () {
    $('#res').text("Hello");
    $('#res').off();
});
…
<input id="myButton" type="button" value="PUSH"/>
…
<p><select id="selBB">
    <option>Jesse Pinkman</option>
    <option>Hank Schrader</option>
</select></p>
<p><textarea id="txtarea" rows="2" cols="22"></textarea></p>
<div id="res"></div>
```

On mousedown

mouse has entered or leaved!

mouse has entered or leaved! mouse was pushed down on id=res in x=174 and y=157

After myButton was clicked and on mousedown mouseenter or mouseleave

Hello

on\off

- *On* and *off* replace the old:
  - *Bind\unbind*
  - *Live\die*
  - *Delegate*
- don't use them any more!

hover()

- *.hover(mouseInHandler, mouseOutHandler)*

```
<style>
    .Highlight
    {
        background-color: Yellow;
    }
</style>

…
$('#res').hover(
    function () { //mouseenter
        $(this).addClass('Highlight');
    },
    function () { //mouseleave
        $(this).removeClass('Highlight');
    }
);

…
<div id="res"></div>
```
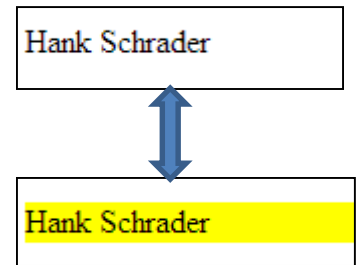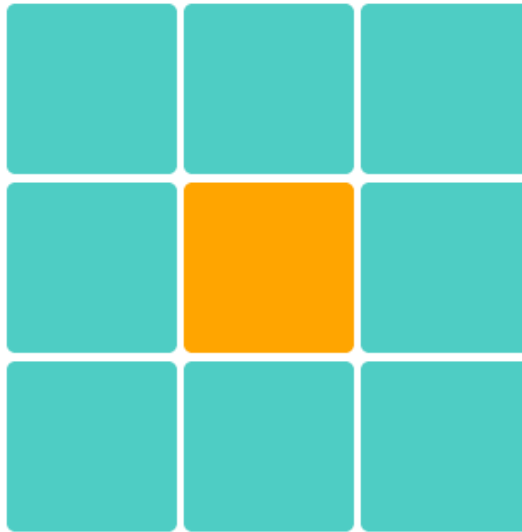
©nir chen
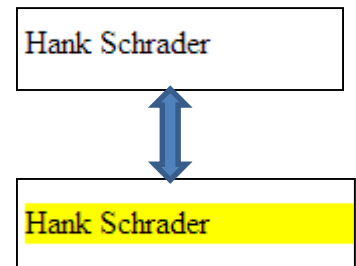
- *.hover(mouseInOutHandler)*

```
<style>
    .Highlight
    {
        background-color: Yellow;
    }
</style>

…
$('#res').hover(
function () { //mouseInOut
    $(this).toggleClass('Highlight');
});

…
<div id="res"></div>
```
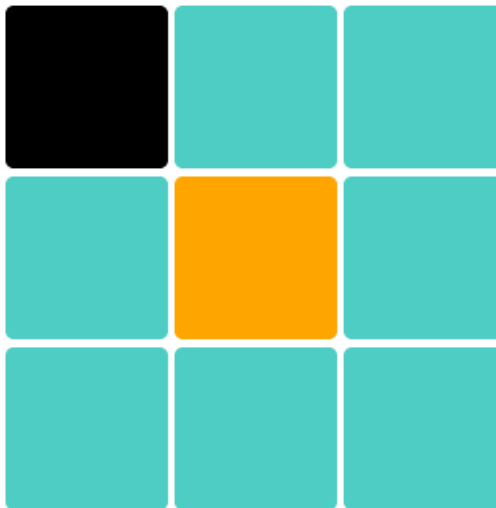
X Mix Drix

The Orange Turn

New Game

The Black Turn
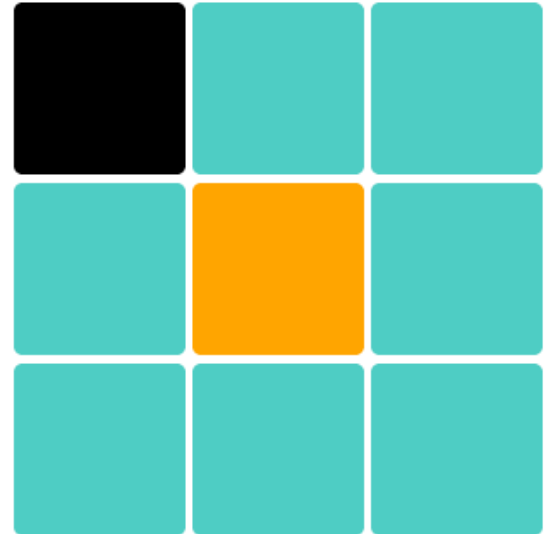
The Orange Turn

The Black Turn

The Orange Turn

The Black Turn

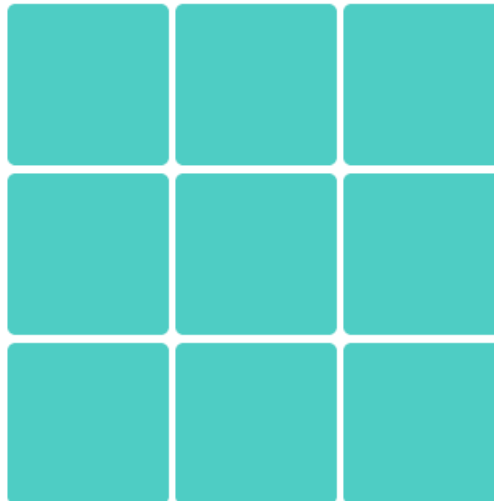The Orange Wins through DIAGONAL!!!

The Orange Wins through ROWS!!!

The Black Wins through COLS!!!
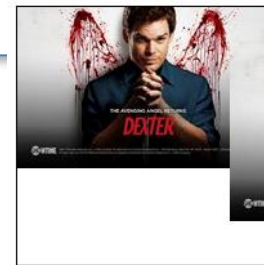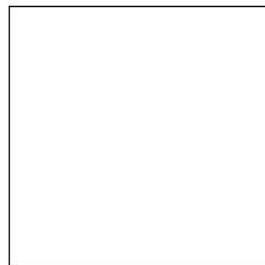
60

There is a Tie Score!

New Game

Drag and Drop

- *.draggable() – enables the element to be dragged*

```
<link href="styles/jquery-ui%20v1.10.3.css" rel="stylesheet" type="text/css" />
<script type="text/javascript" src="scripts/jquery-1.9.1.min.js"></script>
<script src="scripts/jquery-ui%20v1.10.3.js" type="text/javascript"></script>

…
$('img').draggable({
    helper: 'clone' //with this property can be dropped only on a droppable!!!
});

…
<div  id="src">
    <img src="img/dex.jpg" alt="dex" width="200px"/>
</div>
<div  id="target"></div>
```

2 libraries needed

©nir chen

Drag and Drop

- *.droppable() – enables the element to be dropped into*

```
<link href="styles/jquery-ui%20v1.10.3.css" rel="stylesheet" type="text/css" />
<script type="text/javascript" src="scripts/jquery-1.9.1.min.js"></script>
<script src="scripts/jquery-ui%20v1.10.3.js" type="text/javascript"></script>
…
$('#target').droppable({
    drop: function (event, ui) {
        $(this).append(ui.draggable);
    }
});
…
<div  id="src">
    <img src="img/dex.jpg" alt="dex" width="200px"/>
</div>
<div  id="target"></div>
```
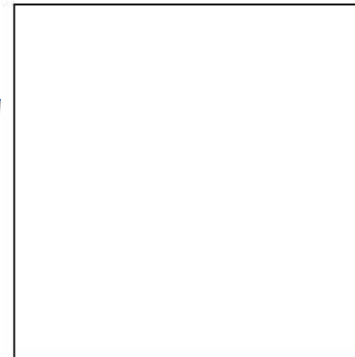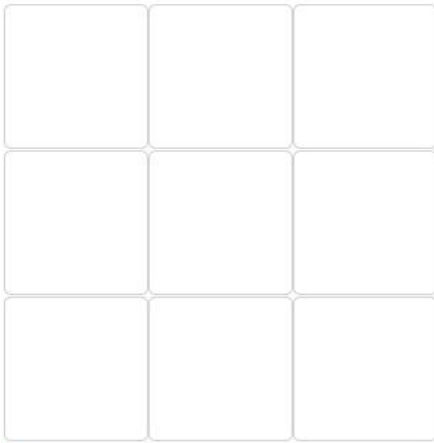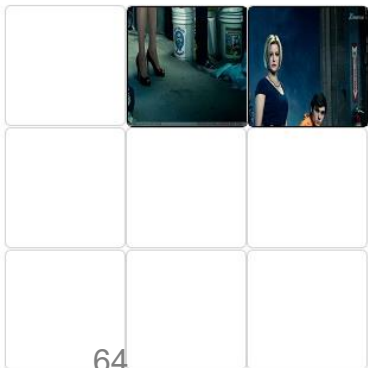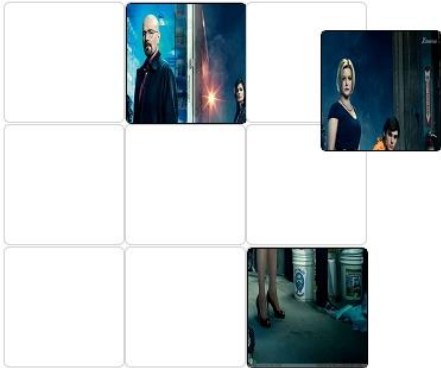
On drop…

The element dragged

©nir chen

**BrBa Puzzle**

New Game

©nir chen

nir chen

- Startup

- Selectors

- Modifying the DOM

- Events

- Ajax

Ajax

- Ajax - *Asynchronous JavaScript and XML*

- A way to refresh only <u>part</u> of the page (DOM), so it is more efficient than refreshing whole of it.

- We will use it to send and receive data from the server.

- The data can be of type: JSON, XML, HTML… we will use the JSON format (very efficient)

- The server side will be written in Asp.Net C# using the Web Service technique

Web Service

- A web service is function that can be called from another computer over the internet.

- We need to know the url and the functions signature
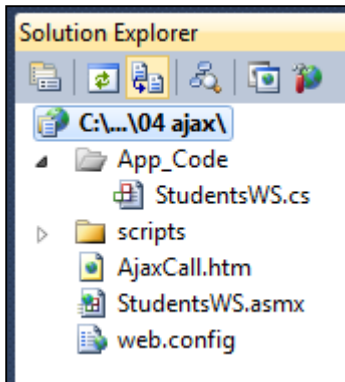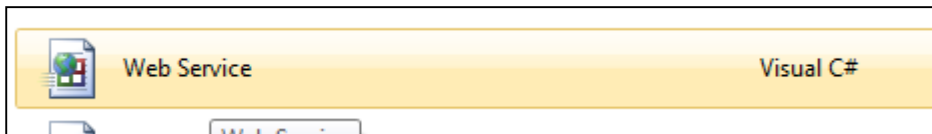
- This is just a student class that we would use in the example

```csharp
class Student
{
    static int count=0;
    public Student()
    {
        ID = count;
        count++;
        Name = "avi" + ID;
        Grade = 80 + ID;
    }
    public int ID { get; set; }
    public string Name { get; set; }
    public double Grade { get; set; }
}
```

Web Service cont'

attribute

Could be any name

uncomment

JSON

```csharp
[WebService(Namespace = "http://nir.org/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
// To allow this Web Service to be called from script, using ASP.NET AJAX,
uncomment the following line.
[System.Web.Script.Services.ScriptService]
public class StudentsWS : System.Web.Services.WebService {

    public StudentsWS () {
        //Uncomment the following line if using designed components
        //InitializeComponent();
    }


    [WebMethod]
    public string GetStudents(int num)
    {
        Student[] students = new Student[num];
        for (int i = 0; i < num; i++)
        {
            students[i] = new Student();
        }
        JavaScriptSerializer serializer = new JavaScriptSerializer();
        string jsonString = serializer.Serialize(students);
        return jsonString;
    }
}
```

©nir chen

Web config

- In order to be able to call and **<u>INVOKE</u>** the function from another domain (cross domain) let's say your laptop, we need to add the following to the web.config

```
<system.web>
    <webServices>
      <protocols>
        <add name="HttpGet"/>
        <add name="HttpPost"/>
      </protocols>
    </webServices>
```

©nir chen

ajax()

- the $\$.ajax()$ function has some parameters to set using the JSON object:
  - url: the url of the Web Service

  - dataType: the type of the data returned xml, json, script, or html, we use JSON

  - type: Get\POST, we use POST

  - data: the data send to the server

  - contentType: the data type sent to the server, we use "application/json; charset=utf-8"

  - error: function callback

  - success:function callback

ajax()

```javascript
var WebServiceURL = "StudentsWS.asmx"; //the same as above. only with…

$.ajax({
    url: WebServiceURL + "/GetStudents",
    dataType: "json",
    type: "POST",    //use only POST!
    data: "{'num':'" + num + "' }",
    contentType: "application/json; charset=utf-8",
    error: function (jqXHR, exception) {
        //alert("errornir: " + JSON.stringify(jqXHR)); //all the erro…
        alert( formatErrorMessage(jqXHR, exception));
    },
    success: function (data) {
        var str = "";
        studentsObj = JSON.parse(data.d); //or data["d"]
        for (var i = 0; i < studentsObj.length; i++) {
            str += studentsObj[i].ID + "<span class='space'></span>"…
        }
        $("#studentsTable").html(str);
    }
});
```
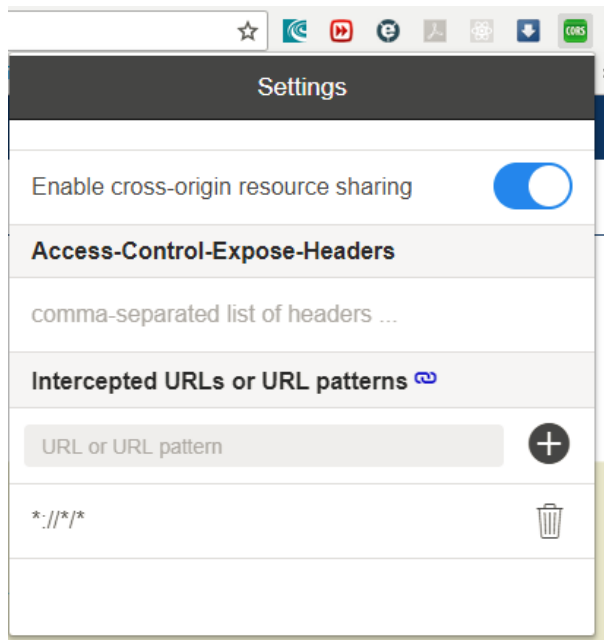
©nir chen

ajax() cont'

```javascript
function formatErrorMessage(jqXHR, exception) {
    if (jqXHR.status === 0) {
        return ('Not connected.\nPlease verify your network connection.');
    } else if (jqXHR.status == 404) {
        return ('The requested page not found. [404]');
    } else if (jqXHR.status == 500) {
        return ('Internal Server Error [500].');
    } else if (exception === 'parsererror') {
        return ('Requested JSON parse failed.');
    } else if (exception === 'timeout') {
        return ('Time out error.');
    } else if (exception === 'abort') {
        return ('Ajax request aborted.');
    } else {
        return ('Uncaught Error.\n' + jqXHR.responseText);
    }
}
```

©nir chen

Ajax

- The code we saw here will only work *localhost*

- In order to use it on other machine, cross domain we need to use the *JSONP* (JSON with padding) technique. (a bit complicated)

- Luckily, we wont need to do anything while using the *PhoneGap Build*. Because it will do everything for us even with the code we saw here!

Add extension <u>or</u> add to web.config!!!


```xml
<configuration>

  <system.webServer>
    <httpProtocol>
      <customHeaders>
        <add name="Access-Control-Allow-Headers" value="accept, content-type" />
        <add name="Access-Control-Allow-Origin" value="*" />
        <add name="Access-Control-Allow-Methods" value="POST, GET, OPTIONS" />
      </customHeaders>
    </httpProtocol>
  </system.webServer>
```