jQuery

User Interfaces

420-WC4-AB

jQuery

- jQuery is ideal because it can create impressive animations and interactions.
- jQuery is simple to understand and easy to use, which means the learning curve is small, while the possibilities are (almost) infinite.
- One of main benefits of jQuery is that it tries to hide browser differences
  - and it mostly succeeds.
- The full jQuery documentation can be found at: https://jquery.com/

Accomplish a lot with less code

- **DOM manipulation** – easier to select DOM elements
- **Event handling** – eases the way to capture a wide variety of events
- **AJAX Support** – it helps you a lot to develop a site using AJAX technology
- **Animations** – it comes with a lot of built-in animation effects
- **Lightweight** – loads faster and takes less space if hosted localy
- **Cross Browser Support** – compatible with all major modern browsers (Chrome, Firefox, Safari, Internet Explorer, etc.)

Review JavaScript

- Traversing our DOM to access HTML elements

```
let myString = "Example of a string";
let myInteger = 7;
let myBoolean = true;
let myArray = ['A String', 11, myString, true];
let myObject = {
   name: 'Shadow',
   age: 14,
   breed: ['Mini Pincher', 'Jack Russel']
}
```

Review DOM

- Traversing our DOM to access HTML elements

```
document.getElementById('txtName');
document.getElementsByClassName('txtBlue');
document.getElementsByTagName('h1');
document.querySelectorAll('a');
document.querySelector('img');
```

JavaScript vs. jQuery

- JavaScript

```
let listItems = document.getElementsByTagName("li");
for (let i = 0; i < listItems.length; i++){
   listItems[i].style.display = "none"
}
```

- jQuery

```
$("li").hide();
```

jQuery Library

- Library is a collection of functions/methods to make our lives easier
- You can use the jQuery library locally or use the CDN Based Version.
  - As with many third-party library you'll end up using in JavaScript, there are both pros and cons to local installation and CDN based version.
- Once downloaded you'll notice the .js extension, (because the jQuery is just a JavaScript library), so include the file in your HTML file like you include normal JavaScript files.
- Always include the jQuery file **before** your custom scripts

Sample jQuery

- Example of a simple jQuery operation by changing the color of the heading text from the default black color to red

```
<script src="js/jquery.js"></script>
<script type="text/javascript">
   $(document).ready(function(){
      $("h2").css("color", "#0088ff");
   });
</script>
```

Ready

- The ready event is used to make sure that the DOM is ready and loaded.
- Most common .ready() structure

```
$(document).ready(function(){
  // Code executed once DOM is loaded...
  alert("Hello World!");
});
```

The `$()` functions

- The jQuery selectors usually starts with the dollar sign **$** followed by parentheses **()** .
- We can select different elements using the jQuery selector (similar to css)
- It is most common to use `$()`
    - this is equivalent to using `jQuery()`.

Selecting Elements

- You'll notice that selectors in jQuery are like those used in CSS
- `$("p")` → all paragraph <*p*> elements
- `$(".className")` → all the elements that have the class *className*
- `$("#myId")` → the element that has the id *myId*
- `$(".className, #myId")` → all the elements in your document that have the class *className* and the element that has the id myId
- `$(".className li")` → all the list items <li> elements that are descendant of elements with the class *className*

Events

- Wait for a button to be click before changing the text

```
$(document).ready(function(){
    $("button").on("click", function(){
        $("p").text("Hello World!");
    });
});
```

- Now our will wait for any button element to be clicked before changing the text of all our paragraphs

Prevent Default Events

```
//default event for clicking on link is to go to new page
$('a').on('click', function (event) {
    event.preventDefault();
    console.log('Not going there!');
});

//default event is to submit form and reload page
$('form').on('submit', function (event) {
    event.preventDefault();
    console.log('Not submitting, time to validate!');
});
```

Sample – Hover Event

- The hover() method expects one or two event handler.
  - First for when the pointer enters and the other for when it leaves the elements.

```
$(document).ready(function(){
  $("p").on("mouseenter", function(){
    $(this).addClass("highlight");
  }).on("mouseleave", function(){
    $(this).removeClass("highlight");
  });
});
```

- The **this** keyword inside an event handler function is a reference to the element where the event is currently being executed.

Sample – Change Event

- Change is executed when a value to change
- For selects, checkboxes and radio buttons, the event is executed as soon as the user makes a selection
- For text input and textarea the event is fired after the element loses focus

```
$(document).ready(function(){
    $("select").on("change", function(){
        var selectedOption = $(this).find(":selected").val();
        alert("You have selected – " + selectedOption);
    });
});
```

```
– var checkedOption = $(this).find(":checked").val();
```

Looping Elements

- Instead of creating a loop, we can have jQuery handle the iterations for all matched elements in the selector.
- Using the .each() method allows us to affect every element individually

```
$( "li" ).each(function( index ) {
   console.log( index + ": " + $( this ).text() );
});
```

Create new Elements

- Use any HTML string as jQuery selector to create the DOM element

```
let newElem = $('<p class="intro">Lorem ipsum!</p>');
newElem.css("color", "orange");
$("body").append( newElem );

let newInput = $("<input>").attr('name', "uAge").attr('type', "number").val(2);

$('<div/>',{ text: 'Div text', class: 'className'}).appendTo('#parentDiv');
```

Create new Elements

- Use any HTML string as jQuery selector to create the DOM element

```
$('<div/>',{
    text: 'Div text',
    class: 'className'
}).appendTo('#parentDiv');
```

Common Effects

`.show(...)`

`.hide(...)`

`.toggle(...)`

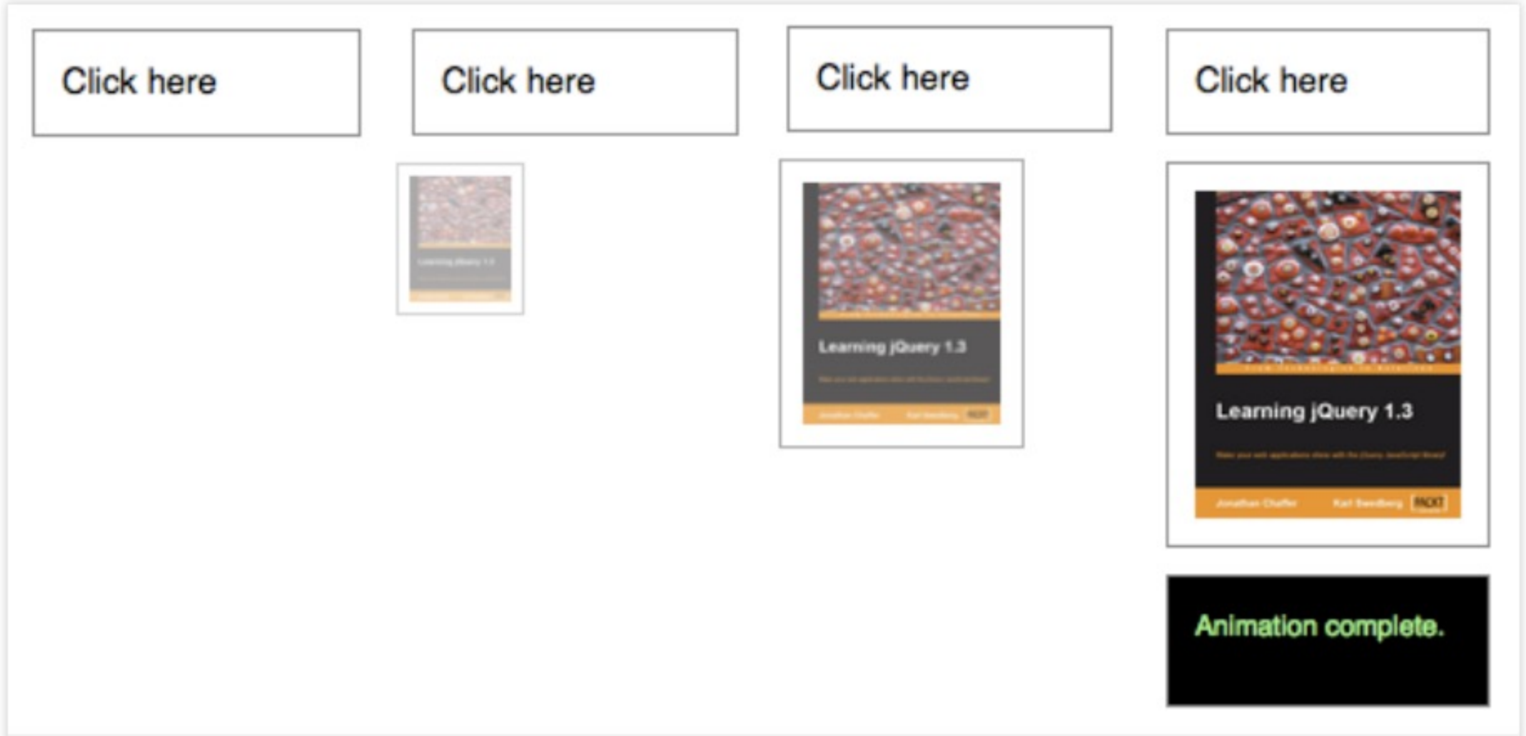
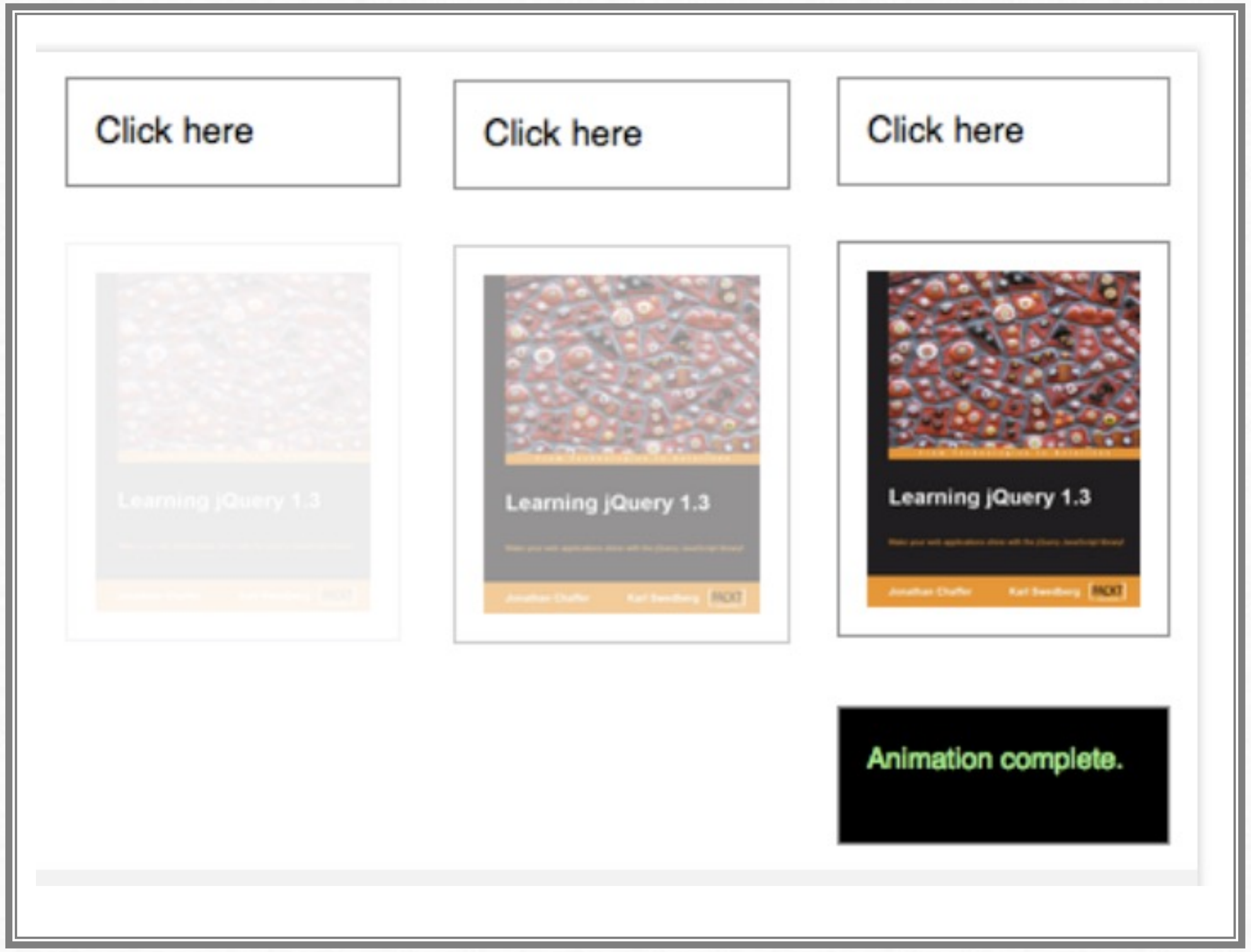


Figure 1 - Illustration of the `show()` *effect*

Image credit: https://api.jquery.com/show/



Common Effects

`.fadeIn(...)`

`.fadeOut(...)`

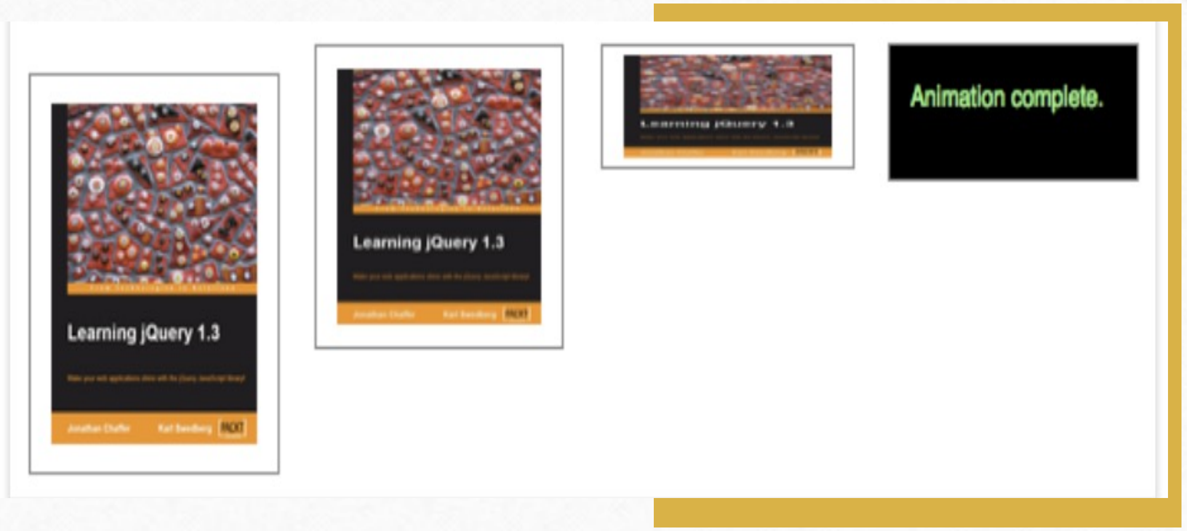
`.fadeToggle(...)`

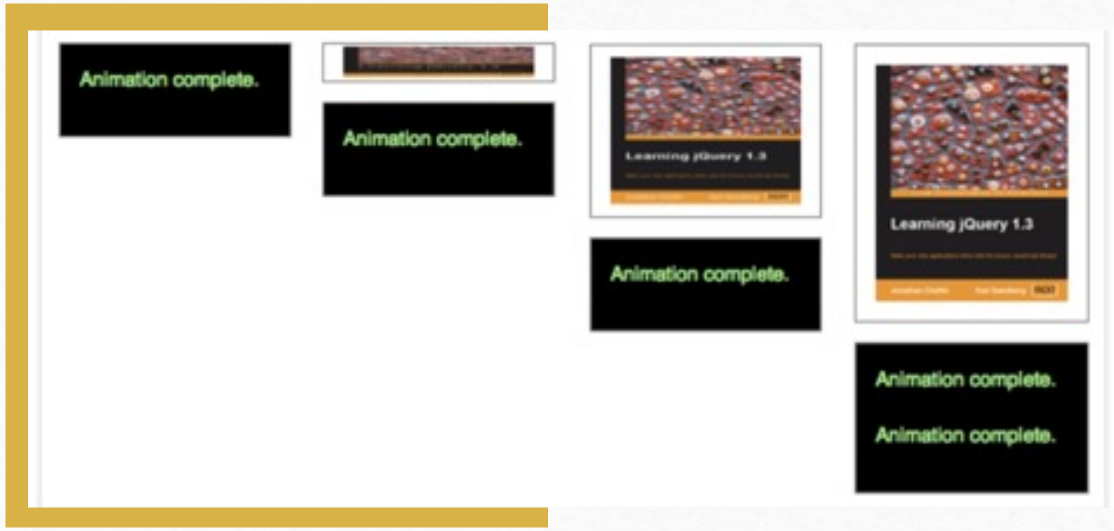
`.fadeTo(...)`

Image credit: https://api.jquery.com/fadeIn/

Common Effects

```
.slideUp(...)
.slideDown(...)
.slideToggle(...)
```





Sample – Show/Hide Effects

- You can show and hide HTML elements using .show() and .hide()
- You can automatically switch between then using .toggle()

```
$(document).ready(function(){
    $("h2").show(); // Show h2 elements
    $("h3").hide(); // Hide h3 elements

    // on click button toggle visibility
    $(".show-btn").click(function(){
        $("p").toggle();
    });
});
```

Effects - Animate

- The `.animate()` method is usually used to animate numeric CSS properties
  - non-numeric properties such as color, background-color cannot be animated

```
$("img").animate({
   width: "300px",
   height: "300px",
});
```

- You can add speed and a callback function as the 2nd and 3rd parameters

Effects – Durations

- You can control the duration of effects by adding a parameter
- Possible values : *slow, fast* or in a number of milliseconds

```
$("p.normal").hide();
$("p.fast").hide("fast");
$("p.slow").show("slow");
$("p.very-fast").fadeOut(50);
$("p.very-slow").fadeIn(2000);
```

Effects – Callback Function

- JavaScript is executed line by line, but since jQuery effects take time to complete, we implement callback functions that are executed once the method finishes.
- The callback function is passed as the 2nd parameter of an effect

```
$("p").slideToggle("slow", function(){
    // executed once effect is complete
    alert("The slide toggle effect has completed.");
});
```

Chaining

- Chaining allows us to perform multiple action on the same set of elements, all within a single line of code.

```
let banner = $("#myBanner");

banner.css('color', 'red');
banner.html('Welcome!');
banner.show();
```

- Same as:

```
let banner = $("#myBanner");
banner.css('color', 'red').html('Welcome!').show();
```

Content and Values

`.text() / .text(newText)`

- Get or set the text content of the given element (removes any markup)

`.html() / .html(newHTML)`

- Get or set the HTML content of a given element

`.val() / .val(newVal)`

- Get or set the value of a form control

```
.attr(attribute) / .attr(attribute, newValue)
.removeAttr(attribute)
```

- Get, set or remove the value of the attribute

Classes

`.addClass(className)`

`.removeClass(className)`

`.toggleClass(className)`

- Add, remove or toggle the specified *className*

`.css(property, value)`

- Change a single CSS property

Questions ?

The first step to receiving an answer is being brave enough to ask a question