JQuery

Common JQuery Tasks

DOM manipulation	Find, alter, add, or remove DOM elements
User interaction	Respond to mouse clicks, mouse movement, or typing
Animation	Smoothly show, hide, or move web page elements
Widgets	Display and manage the interaction of complex GUI elements
Ajax	Issue asynchronous HTTP requests and handle responses

Including JQuery In Your Web Page

- From a local file: <**script src="jquery-3.5.1.min.js"**></**script**>
- From a content delivery network (CDN):

  <**script** src=https://code.jquery.com/jquery-3.5.1.min.js

  integrity="sha256-9/aliU8dGd2tb6OSsuzixeV4y/faTqgFtohetphbbj0="
  crossorigin="anonymous"></**script**>

  - The integrity and crossorigin attributes allow the browser to verify that the contents of the jquery file have not been altered by a 3rd party

Using JQuery

- Store JQuery objects in variables that start with $
- $() and JQuery() are identical. They perform a JQuery task and return a JQuery object

Selectors

- $(‘CSS selector’): returns all HTML DOM elements that match that CSS selector
- Basic JQuery selectors

Selector Type	Example	Explanation
Element	$("p")	Selects all <p> elements
ID	$("#hello")	Selects the element with id="hello"
Class	$(".important")	Selects all elements with class="important"

Add/RemoveClass

- addClass and removeClass add and remove a class from an HTML DOM object

  Example: $(“p”).addClass(“important”): adds the important class to all paragraph elements

Events

- Ways to register event handlers
  - Using the “on” function

    Example: $("#mybutton").on("click", myCallback);
  - Using the shortcut name for the method

    Example: $("#mybutton").click(myCallback);

Ready Event

- Ready event: Triggered when the browser has finished loading the web page's DOM
  - Similar to the load event on the body tag element

```
$(document).ready(function() {
    // DOM is ready to go
    $("button").addClass("big");
});
```

  - Shortcut technique: use $(function)

```
$(function() {
    // DOM is ready to go
    $("button").addClass("big");
});
```

Changing CSS Styles

- css() method adds/changes CSS styles for an object

```
$("body").css("background-color", "peachpuff");
$("body").css({ color: "green",  "font-size": "20pt" });
```

Animation Effects

Methods	Example	Description
show() hide() toggle()	$("h1").show("slow"); $("h1").hide("slow"); $("h1").toggle("slow");	Alters width, height, and opacity all at once
fadeIn() fadeOut() fadeToggle()	$("h1").fadeIn("normal"); $("h1").fadeOut("normal"); $("h1").fadeToggle("normal");	Alters opacity only
slideDown() slideUp() slideToggle()	$("h1").slideDown("fast"); $("h1").slideUp("fast"); $("h1").slideToggle("fast");	Alters height only

Speed Argument for Animation Methods

Argument	Example	Explanation
"slow"	$("p").show("slow");	0.6 seconds to show the paragraph
"normal"	$("p").show("normal");	0.4 seconds to show the paragraph
"fast"	$("p").show("fast");	0.2 seconds to show the paragraph
milliseconds	$("p").show(1500);	1.5 seconds to show the paragraph

Controlling Animation Order

- Code written after a call to an animation method will execute before the animation is complete.
- To execute code after the animation is complete, the jQuery animation methods can be passed a callback function

```
$("h1").fadeIn("slow", function() {
        $("h2").fadeIn("slow");
 });
```

- Animations on the same element are queued: The following two animations are performed sequentially

```
$("h1").fadeIn("slow");
$("h1").slideUp(1000);
```

Controlling Animation Order Using queue()

- queue(): queues code that should be executed after the previous animations complete.
- queue() can take the place of the callback function used with the animation methods.
- queue() takes a function argument that is passed a function parameter called next. The next function must be called so the next animation in the queue can be processed.

```
$("h1").hide("slow") .queue(function(next) {
        // Add the class after the <h1> hide completes
        $(this).addClass("important");
        // Process the next animation in the queue
        next();
}) .slideDown(1000);
```

Animate() method

- Animates the transition of any CSS property that has a numeric value
- animate() takes four arguments:
  - Dictionary of property:values to animate
  - Duration of the animation in ms (default is 400)
  - Easing function (default is "swing"): determines the speed at which an animation progresses.
    - “swing”: makes the animation start and end slowly, but the animation goes quicker in the middle, just like a person on a swing.
    - “linear”: makes the animation progress at a constant rate.
  - Optional callback to be executed when the animation is complete.

Animate method: Example

```
$("img").animate({left: "200px",
                  height: "-=60",
                  width: "200px" },
                 “slow”, “linear”,
                 function() {
                  $(this).css("border", "2px solid red");
                 });
```

- Moves image to pixel location 200, decrements height by 60, and sets width to 200 pixels
- Performs the animation linearly at a slow pace
- When the animation is complete the image’s border is set to solid red

Dom Manipulation-Adding DOM Objects

Methods	Example	Before	After
prepend()	$("ol").prepend(" New first ");	- A - B	- New first - A - B
append()	$("ol").append(" New last ");	- A - B	- A - B - New last
before()	$("h2").before(" Before ");	Test	Before Test
after()	$("h2").after(" After ");	Test	Test After

Removing DOM Objects

Methods	Example	Before	After
remove()	`$("li").remove();`	`<ol> <li>A</li> <li>B</li> </ol>`	`<ol> </ol>`
detach()	`let $listElems = $("li").detach();`	`<ol> <li>A</li> <li>B</li> </ol>`	`<ol> </ol>`

Modifying DOM Text

Methods	Example	Before	After
html()	`let s = $("p").html();` `$("div").html(s);`	`<p> A<b>B</b>C </p>` `<div> </div>`	`<p> A<b>B</b>C </p>` `<div> A<b>B</b>C </div>`
text()	`let s = $("p").text();` `$("div").text(s);`	`<p> A<b>B</b>C </p>` `<div> </div>`	`<p> A<b>B</b>C </p>` `<div> ABC </div>`

AJAX: Get and Post Requests

- $.get() and $.post() send HTTP requests and supply a callback function that executes when the full response is received from the web server.
- Parameters are
  - URL for server-side script
  - Data to be sent to the server in the form of a dictionary that can be encoded using json
  - A callback function that should be called when the server returns its response
    - The callback function takes a single argument which is the server's response
    - If the response is a JSON object, then the argument will be a javascript dictionary that the browser obtains by decoding the JSON object
  - A string that tells the browser how to parse the server's response. If you want json then the string value will be "json"

AJAX: Get and Post Requests

- $.get() and $.post() send HTTP requests and supply a callback function that executes when the full response is received from the web server. Example from 9.6.3 in book

```
<body>
   Title: <input type="text" id="title"><br>
   <button id="search">Search</button>
   <div id="movieinfo"> </div>
</body>
```

```
$("#search").click(function() {
   let requestData = { title: $("#title").val() };
            $.get("lookup.php", requestData, function(data) {
            $("#movieinfo").html("<cite>"
               + data.title + "</cite>: Rated " + data.rating
                +  ", released in " + data.year); }, "json");
});
```

Returned by server:

{"title":"Star Wars","rating":"PG","year":"1977"}

AJAX: Handling Errors

- Reasons for failure
  - The web server returns an empty response or a response with an error message because the requested data was not found (typically found in the data object returned by the script)
  - The web server returns a 40x or 500 response code.
- If script returns an error response, handle it in the callback function
- If server returns error code: $.get(), and $.post() return a jqXHR object, which is an abbreviation for **jQ**uery **X**ML**H**ttp**R**equest object. This object provides a fail method:

```
$.get("lookup.php", requestData, function(data) {
    $("#movieinfo").html("<cite>" + data.title + "</cite>: Rated " + data.rating + ", released in " + data.year); },
        "json").fail(function(jqXHR) {
            $("#movieinfo").html("There was a problem contacting the server: " + jqXHR.status + " " +
            jqXHR.responseText);
});
```