**jQuery Overview**

**Unleash the Power of jQuery**

**Table of Contents**

* What is jQuery?
* jQuery Fundamentals
* Selectors
* DOM Manipulation
* DOM Altering
* jQuery DOM elements
* AJAX
* jQuery AJAX Methods
* Executing AJAX Requests

**What is jQuery?**

**The world’s most popular JavaScript library**

* jQuery is a cross-browserJavaScript library
* Designed to simplify the client-side scripting of HTML
* The most popularJavaScript library in use today
* Free, open source software
* jQuery's syntax is designed to make it easier to
* Navigate a document and select DOM elements
* Create animations
* Handle events
* jQuery also provides capabilities for developers to create plugins for
* Low-level interaction and animation
* Advanced effects and high-level, theme-able widgets
* Creation of powerful and dynamic web pages
* Microsoft adopted jQuery withinVisual Studio
* Uses in Microsoft'sASP.NET AJAX Framework andASP.NET MVC Framework

**Why jQuery is So Popular?**

* Easy to learn

Fluent programming style

* Easy to extend

You create new jQuery plugins by creating new JavaScript functions

* Powerful DOM Selection

Powered by CSS 3.0

* Lightweight
* Community Support

Large community of developers and geeks

**How to Add jQuery to a Web Site?**

* Download jQuery files from http://www.jquery.com
* Self hosted
* You can choose to self host the .js file
* E.g. jquery-2.1.1.js or .min.js file
* Use it from CDN (content delivery network)
* Microsoft, jQuery, Google CDNs
* e.g. <http://code.jquery.com/jquery-2.1.1.min.js>,
* http://ajax.microsoft.com/ajax/jquery/jquery2.1.1.min.js

**Selectors and DOM Manipulation**

**Selectors**

Selection of DOM elements in jQuery is much like as in pure JavaScript

* Selection of elements using CSS selectors

$(selector)

* Like querySelectorAll

//by tag

$("div") //document.querySelectorAll("div");

//by class

$(".menu-item")

//document.querySelectorAll(".menu-item");

//by id

$("#navigation")

//by combination of selectors

$("ul.menu li")

**Selection with jQuery**

* Selecting items with jQuery
* Almost always returns a collection of the items

Even if there is only one item

* Can be stored in a variable or used right away
* The usage of the elements is always the same, no matter whether a single or many elements

// select the item

$("#something").hide();

$(".widgets").fade(1);

* More at: http://learn.jquery.com/using-jquerycore/selecting-elements/

*Selection with jQuery - Live Demo*

**DOM Traversal - Traversing the nodes of the DOM**

**DOM Traversal**

As with plain JavaScript, the DOM can be traversed with jQuery

* Properties for:
* Next and previous siblings
* Parents and children

**DOM Traversal: Next and Previous**

jQuery.next(), jQuery.prev()

* Returns the next/prev sibling
* Returns an HTML element
* Not a [text] node

<ul> var $first = $("li").first();

<li>Item 1</li> log($first); //logs "Item 1"

<li>Item 2</li> log($first.next()); //logs "Item 2"

<li>Item 3</li>

</ul>

*Next/PrevSiblings - Live Demo*

**DOM Traversal: Parent**

* jQuery.parent()
* Returns the parent of the element
* jQuery.parents(selector)
* Returns the first parent that matches the selector

<div id="wrapper">

<ul id="items-list">

<li>Item 1</li> <li>Item 2</li>

<li class="special">Item 3</li>

<li>Item 4</li>

</ul>

</div>

var $node = $(".special");

$node.parent().attr("id"); //logs "items-list"

$node.parents("div").attr("id"); //logs "wrapper"

$node.parents("#wrapper") .attr("id"); //logs "wrapper"

*Parent Element - Live Demo*

**Altering the DOM**

**Adding and removing DOM elements**

**Adding Elements**

* Adding elements can
* be done on the fly
* jQuery.appendTo()/prependTo()
* jQuery.append()/prepend()

$('<ul><li>Hello</li></ul>').appendTo('body');

$("body").prepend("<h1>header</h1>");

**Creating elements**

* Creating new elements is also easy

var $divElement = $('<div>');

var $anotherDivElement = $('<div />');

*Adding Elements to the DOM - Live Demo*

**Removing Elements**

* You can also remove elements from the DOM - Just as easy

// Before

<div>

<p>Red</p>

<p>Green</p>

</div>

// Removing elements

$('p').remove();

// After

<div>

</div>

*Removing Elements - Live Demo*

**jQuery Extended DOM Elements**

**jQuery Objects**

Selected with jQuery DOM elements are NOT pure DOM elements

* They are extended
* Have additional properties and methods
* addClass(), removeClass(), toogleClass()
* on(event, callback) for attaching events
* animate(), fade(), etc…

//Parsing a regular DOM element to jQuery Element

var content = document.createElement("div");

var $content = $(content);

**Properties of jQuery Elements**

* jQuery elements extend regular DOM elements
* Methods for altering the elements
* jQuery.css("color", "#f3f")
* jQuery.html() returns the innerHTML
* jQuery.html(content) sets the innerHTML
* jQuery.text(content) sets the innerHTML, by escaping the content

*Properties of jQuery Elements - Live Demo*

**jQuery Events (Cross-browser events)**

**jQuery Events**

* jQuery has a convenient way for attaching and detaching events
* Works cross-browser
* Using methods on() and off()

function onButtonClick(){

$(".selected").removeClass("selected");

$(this).addClass("selected");

}

$("a.button").on("click", onButtonClick);

* Optimize the event
* Add it on the parent element
* A bit different syntax

function onListItemClick(){

$(".selected").removeClass("selected");

$(this).addClass("selected");

}

$("ul").on("click", "li", onListItemClick);

*jQuery Event Handlers - Live Demo*

**jQuery Chaining Call (after call, after call…)**

**jQuery Chaining**

* The chaining paradigm is as follows:
* If a method should return result -> Ok, return it
* If a method should NOT return a result -> return this
* jQuery implements this paradigm, so methods can be chained to one another:

$('<button />')

.addClass('btn-success')

.html('Click me for success')

.on('click', onSuccessButtonClick)

.appendTo(document.body);

*jQuery Chaining - Live Demo*

**jQuery AJAX (Creating HTTP requests with jQuery)**

* AJAX stands for Asynchronous JavaScript and XML
* Meaning asynchronously get data from a remote place and render it dynamically
* jQuery provides some methods for AJAX
* jQuery.ajax(options) – HTTP request with full control (headers, data, method, etc…)
* jQuery.get(url) – HTTP GET request
* jQuery.post(url) – HTTP POST request
* jQuery(selector).load(url) – loads the contents from the url inside the selected node

*jQuery AJAX - Live Demo*

**Homework**

1. Create a slider control using jQuery

* The slider can have many slides
* Only one slide is visible at a time
* Each slide contains HTML code

i.e. it can contain images, forms, divs, headers, links, etc…

* Implement functionality for changing the visible slide after 5 seconds
* Create buttons for next and previous slide

2. Using jQuery implement functionality to insert a DOM element before or after another element

3. By given an array of students, generate a table that represents these students

* Each student has first name, last name and grade
* Use jQuery

4. Implement functionality to change the background color of a web page

* i.e. select a color from a color picker and set this color as the background color of the page

5. \*Implement a GridView control

* Rows can be added dynamically
* A header row can be added dynamically
* Each GridView can have at most one header row
* Each row can have a nested GridView
* Each GridView can have at most one nested GridView