Cairo University
Faculty of Engineering
Computer Engineering Department

# WEB BUILDING

# Agenda

**Jquery (Java Script Library):**

1. Introduction

2. Syntax

3. Selectors

4. Event Methods

5. Effects

6. HTML DOM Manipulation

# JQuery

## Introduction

# What is jQuery?

- jQuery is a fast, small, and feature-rich JavaScript library.

- The jQuery library contains the following features:
  - HTML/DOM manipulation
  - CSS manipulation
  - HTML event methods
  - Effects and animations
  - AJAX
  - Utilities

# Why jQuery?

□ There are lots of other JavaScript frameworks out there, but jQuery seems to be the most popular, and also the most extendable.

□ Many of the biggest companies on the Web use jQuery, such as:

- Google
- Microsoft
- IBM
- Netflix

# jQuery Install

☐ There are two versions of jQuery available for downloading:

  ■ **Production version** - this is for your live website because it has been minified and compressed

  ■ **Development version** - this is for testing and development (uncompressed and readable code)

☐ Both versions can be downloaded from jQuery.com

# JQuery

## Syntax

# jQuery Syntax

□ The jQuery syntax is tailor made for **selecting** HTML elements and performing some **action** on the element(s).

□ Basic syntax is: **$(*selector*).*action*( )**

  ▪ A **$** sign to define/access jQuery

  ▪ A **(*selector*)** to "query (or find)" HTML elements

  ▪ A jQuery ***action*( )** to be performed on the selected element(s)

# Examples on jQuery Syntax

**Examples:**

- **$(this).hide()** - hides the current element.
- **$("p").hide()** - hides all <p> elements.
- **$(".test").hide()** - hides all elements with class="test".
- **$("#test").hide()** - hides the element with id="test".

# JQuery

# Selectors

# jQuery Selectors

□ All selectors in jQuery start with the dollar sign and parentheses: **$( )**

1. The element Selector

   The jQuery element selector selects all elements based on the element name. Ex. **$("p")**

2. The #id Selector

   The jQuery #id selector uses the id attribute of an HTML tag to find the specific element. Ex. **$("#test")**

3. The .class Selector

   The jQuery class selector finds elements with a specific class. Ex. **$(".test")**

# More Examples of jQuery Selectors

| Syntax | Description |
|---|---|
| $("*") | Selects **all** elements |
| $(this) | Selects the **current** HTML element |
| $("p.intro") | Selects all <p> elements with class="intro" |
| $("p:first") | Selects the first <p> element |
| $("ul li:first") | Selects the first <li> element of the first <ul> |
| $("ul li:first-child") | Selects the first <li> element of every <ul> |
| $("[href]") | Selects all elements with an href **attribute** |
| $("a[target='_blank']") | Selects all <a> elements with a target attribute value equal to "_blank" |
| $("a[target!='_blank']") | Selects all <a> elements with a target attribute value NOT equal to "_blank" |
| $(":button") | Selects all <button> elements and <input> elements of type="button" |
| $("tr:even") | Selects all even <tr> elements |
| $("tr:odd") | Selects all odd <tr> elements |

# JQuery
## Event Methods

# jQuery Event Methods

□ Here are some common JQuert events:

| Mouse Events | Keyboard Events | Form Events | Document/ Window Events |
|---|---|---|---|
| click | keypress | submit | load |
| dblclick | keydown | change | resize |
| mouseenter | keyup | focus | scroll |
| mouseleave | | blur | unload |
| mousedown | | | ready |
| mouseup | | | |
| hover | | | |

# jQuery Syntax For Event Methods

□ To assign a click event to all paragraphs on a page, you can do this:

```
$("p").click(  );
```

□ The next step is to define what should happen when the event fires. You must pass a function to the event:

```
$("p").click( function ( ) {
    // action goes here!!
    $(this).hide();
} );
```

Any other event method goes here

# JQuery
# Effects

Hide, Show, Toggle, Slide, Fade, and Animate.

# 1. Hide and Show

☐ With jQuery, you can hide and show HTML elements with the hide() and show() methods:

| hide() | $(selector).hide(speed,callback); |
|---|---|
| show() | $(selector).show(speed,callback); |
| toggle() | $(selector).toggle(speed,callback); |

☐ The optional **speed** parameter specifies the speed of the hiding/showing, and can take the following values: **"slow"**, **"fast"**, or **milliseconds**.

☐ The optional **callback** parameter is a function to be executed <u>after</u> the hide() or show() method completes.

# 1. Hide and Show

## Callback

- JavaScript statements are executed line by line. However, with effects, **the next line of code can be run even though the effect is not finished.** This can create errors.

- To prevent this, you can create a callback function.

- A callback function is executed **after** the current effect is finished.

```
$("button").click( function ( ) {
        $("p").hide("slow", function ( ) {
                alert("The paragraph is now hidden");
        } ) ;
} ) ;
```

It will show the alert with each "p" element is being hidden

# Callback

- Without using callback.. Observe when alert box is shown?


- $("button").click( function ( ) {
        $("p").hide( 1000 );
        alert( "The paragraph is now hidden" );
} ) ;

It will show the alert once, and usually the alert will be shown before the hiding.

CMPN 425-Fall 2022

# 2. Fading

☐ JQuery has the following fade methods:

| | |
|---|---|
| **fadeIn()** | $(selector).fadeIn(speed,callback); |
| **fadeOut()** | $(selector).fadeOut(speed,callback); |

To make an element fade in" appear" or fade out "disappear" of visibility.

```
$("button").click( function ( ) {
        $("#div1").fadeOut( );
        $("#div2").fadeOut( "slow" );
        $("#div1").fadeIn( 3000 ); //speed. The higher number, the slower
} ) ;
```

# Example (Fading)

```html
<html> <head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js">
   </script>

<script>

      $(document).ready ( function ( ) {

            $("#b1").click ( function ( ) {

                  $("p").fadeOut( );

            } ) ;

            $("#b2").click ( function ( ) {

                  $("p").fadeIn( );

            } ) ;

      } ) ;

</script>  </head>
```

# Example cont.

**&lt;body&gt;**

 **&lt;p&gt; A paragraph to be faded! &lt;/p&gt;**

 **&lt;button type="button" id="b1"&gt; Fade Out the paragraph!**

 **&lt;/button&gt;**

 **&lt;button type="button" id="b2"&gt; Fade In the paragraph!**

 **&lt;/button&gt;**

**&lt;/body&gt;**

**&lt;/html&gt;**

# 3. Sliding

- The jQuery slide methods slides elements up and down.

| slideDown() | $(selector).slideDown(speed,callback); |
|---|---|
| slideUp() | $(selector).slideUp(speed,callback); |
| slideToggle() | toggles between the slideDown() and the slideUp() methods |

# 4. Animation

- The jQuery animate() method lets you create custom animations.

  **$(selector).animate(** {params} , speed , callback **);**

- The required **params** parameter defines the CSS properties to be animated.

```
$("button").click ( function ( ) {
        $("div").animate ( {
                left:'250px',
                opacity:'0.5',
                height:'150px',
                width:'150px'
        } ) ;
} ) ;
```

# 4. Animation

## Using Pre-defined Values

□ You can even specify a property's animation value as **"show"**, **"hide"**, or **"toggle"**:

```
$("button").click ( function ( ) {
        $("div").animate ( {
                height:'toggle'
        } ) ;
} ) ;
```

# 4. Animation

## Uses Queue Functionality

☐ If you write multiple animate() calls after each other, jQuery creates an "internal" queue with these method calls. Then it runs the animate calls ONE by ONE.

```
$("button").click( function ( ) {
    var div = $("div");
    div.animate( {height:'300px',opacity:'0.4'} , "slow" );
    div.animate( {width:'300px',opacity:'0.8'} , "slow" );
    div.animate( {height:'100px',opacity:'0.4'} ,"slow" );
    div.animate( {width:'100px',opacity:'0.8'} , "slow" );
} ) ;
```

# 4. Animation

## Using Relative Values

□ It is also possible to define **relative values** (the value is then relative to the element's current value). This is done by putting **+=** or **-=** in front of the value:

```
$("button").click ( function ( ) {
        $("div").animate ( {
                left:'250px',
                height:'+=150px',
                width:'+=150px'
        } );
} );
```

# 5. Stop Animations

- The jQuery **stop( )** method is used to stop an animation or effect before it is finished

$(selector).stop( stopAll , goToEnd );

- The optional **stopAll** parameter specifies whether also the animation queue should be cleared or not (Default is **false**)

- The optional **goToEnd** parameter specifies whether or not to complete the current animation immediately. Default is **false**.

# 6. Chaining

☐ Until now we have been writing jQuery statements one at a ==time== (one after the other).

☐ However, there is a technique called ==**chaining**==, that allows us to run **multiple jQuery commands**, one after the other, **on the ==same element(s).==**

☐ **Tip:** This way, ==browsers do not have to find the same element(s) more than once==.

```
$("#p1").css("color","red").slideUp(2000).slideDown(2000);
```
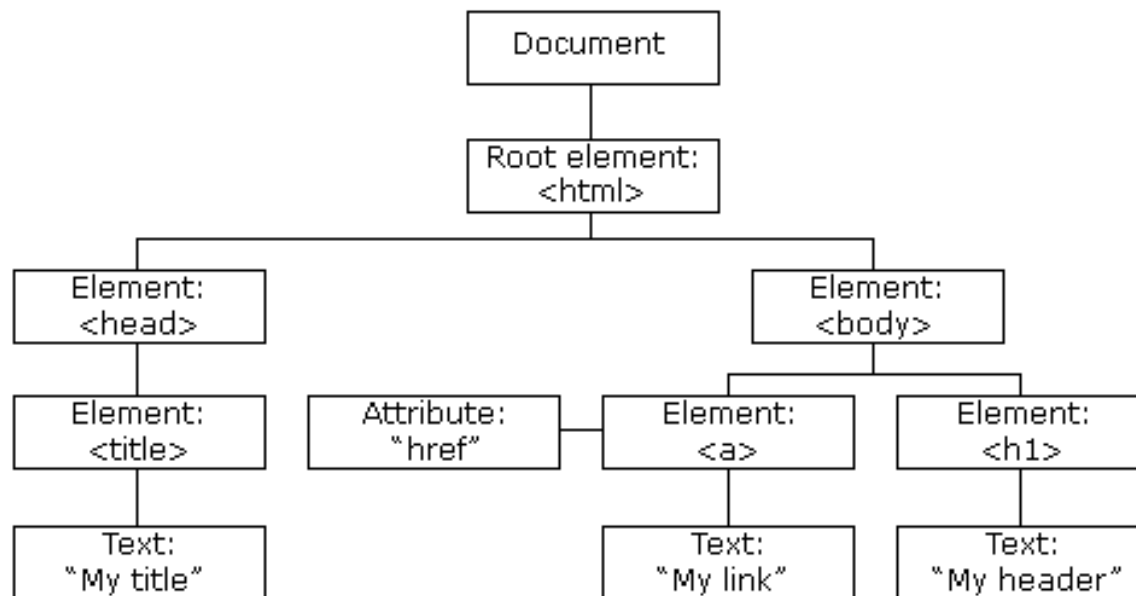
# JQuery
## HTML DOM Manipulation

# The HTML DOM (Document Object Model)

☐ DOM is an interface that allows to access and update the content, structure, and style of a document dynamically .

# The HTML DOM (Document Object Model)

□ All HTML elements are defined as objects that have properties and methods.

⬚ A property is a value that you can get or set (like changing the content of an HTML element).

⬚ A method is an action you can do (like add or deleting an HTML element).

□ **document.getElementById("demo").innerHTML = "Hello World!";**

| This is a method | This is a selector | This is a property |
|------------------|--------------------|--------------------|
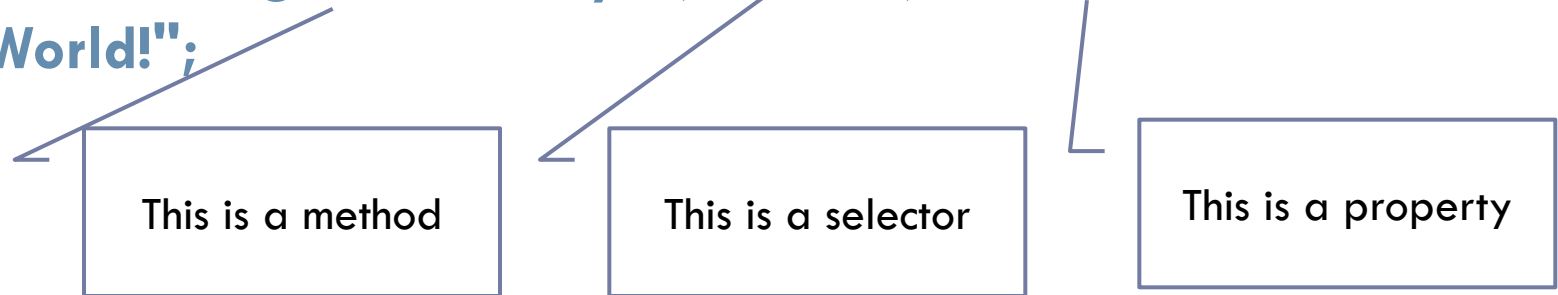
# jQuery HTML DOM Manipulation

□ JQuery comes with a bunch of DOM related methods that make it easy to access and manipulate elements and attributes.

□ For example,

1. Set and Get Content
2. Add Elements
3. Remove Elements/Content
4. Set and Get CSS Classes

# 1. Set and Get Content

| | |
|---|---|
| **text( )** | Sets or returns the **text content** of selected elements |
| **html( )** | Sets or returns **the content of selected elements (including HTML markup)** |
| **val( )** | Sets or returns **the value of form fields** |
| **attr()** | Sets or returns the attribute values |

# Example on Set Content

```
<html> <head>

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
   </script>

<script>

        $(document).ready( function ( ) {
                $("#btn1").click( function ( ) {
                        $("#test1").text( "Hello world! " );
                } ) ;
        } ) ;

</script>

</head> <body>

    <p id="test1"> This is a paragraph. </p>

    <button id="btn1"> Set Text </button>

</body> </html>
```

# Example on Get Content

```
<html> <head>

<script src=http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js>
    </script>

<script>
        $(document).ready( function ( ) {
                $("#btn1").click( function ( ) {
                        alert("Text: " + $("#test1").text( ) );
                } ) ;
        } ) ;
</script>

</head> <body>
    <p id="test1"> This is some text in a paragraph. </p>
    <button id="btn1"> Show Text </button>

</body></html>
```

# 2. Add Elements

□ With jQuery, it is easy to add new elements/content.

| append() | Inserts elements/content **at the end of** the selected elements |
|----------|------------------------------------------------------------------|
| prepend() | Inserts elements/content **at the beginning of** the selected elements |

# 2. Add Elements

```
function appendText()
{
        var txt1 = "<p>Text.</p>"; // Create element with HTML
        var txt2 = $("<p></p>").text( "Text." ); // Create with jQuery
        var txt3 = document.createElement ( "p" );// Create with DOM
        txt3.innerHTML = "Text.";

        $("p").append( txt1 , txt2 , txt3 ); // Append the new elements
}
```

# 3. Remove Elements/Content

□ To remove elements and content, there are mainly two jQuery methods:

| | |
|---|---|
| **remove()** | Removes the selected element **(and its child elements)** |
| **empty()** | **Removes the child elements** from the selected element |

# 3. Remove Elements/Content

```html
<html>
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">
</script>
<script>
        $(document).ready( function ( ) {
                $("button").click ( function ( ) {
                        $("#div1").remove();
                } );
        } );
</script>
</head>
```

# 3. Remove Elements/Content

**&lt;body&gt;**

  **&lt;div id="div1" &gt;**

    **This is some text in the div.**

    **&lt;p&gt; This is a paragraph in the div. &lt;/p&gt;**

  **&lt;/div&gt;**

  **&lt;br&gt;**

  **&lt;button&gt; Remove div element &lt;/button&gt;**

**&lt;/body&gt;**

**&lt;/html&gt;**

# 3. Remove Elements/Content

□ The jQuery **remove( )** method also accepts one parameter, which allows you to filter the elements to be removed.

□ Example

$$\text{\$("p").remove( ".class1" );}$$

# 4. Get and Set CSS Classes

□ With jQuery, it is easy to manipulate the CSS of elements.

| | |
|---|---|
| **addClass()** | Adds one or more classes to the selected elements |
| **removeClass()** | Removes one or more classes from the selected elements |
| **toggleClass()** | Toggles between adding/removing classes from the selected elements |
| **css()** | Sets or returns the style attribute `css("propertyname");` |

# 4. Get and Set CSS Classes

```
.important
{
    font-weight : bold;
    font-size : xx-large;
}
```

-----------------------------------------------------------

```
$("button").click( function ( ) {
        $("h1,h2,p").addClass( "important" );
} ) ;
```

# jQuery Traversing

☐ With jQuery traversing, you can easily **move up (ancestors), down (descendants)** and **sideways (siblings)** in the family tree, starting from <u>the selected (current) element</u>.

☐ This movement is called **traversing** - or **moving through -** the **DOM.**

# jQuery Traversing

- DOM represents parent ,child and siblings relationships.

- Using the DOM model, we can traverse (reach any relative) using the following methods:
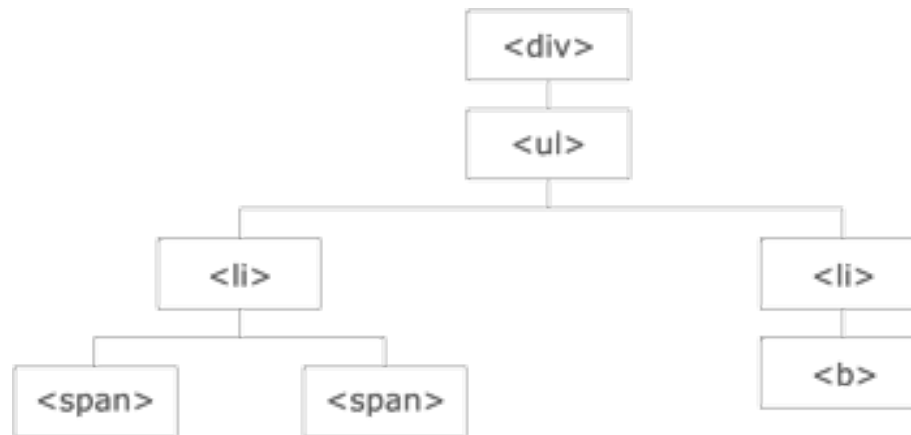
# Traversing the DOM Tree

- **parent ( )**

   It returns the <u>direct</u> parent element of the selected element.

- **children ( )**

   It returns all direct children of the selected element "one level down only"

   **next ( )**

   It returns the next <mark>sibling</mark> element of the selected element.

- **first ( )**

   It returns the first element of the specified elements.

- **Example:**

-       **$(document).ready( function ( ) {**
            **$("div").children();**
        **} ) ;**