# *jQuery

Ahmed Elashry
aelashry@outlook.com

# *What is jQuery

- jQuery is a fast and concise JavaScript Library created by John Resig in 2006 with a nice motto: Write less, do more. jQuery simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is a JavaScript toolkit designed to simplify various tasks by writing less code.

# jQuery library contains

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

# Why jQuery?

- DOM manipulation – The jQuery made it easy to select DOM elements, negotiate them and modifying their content by using cross-browser open source selector engine called Sizzle.
- Event handling – The jQuery offers an elegant way to capture a wide variety of events, such as a user clicking on a link, without the need to clutter the HTML code itself with event handlers.
- AJAX Support – The jQuery helps you a lot to develop a responsive site using AJAX technology.
- Animations – The jQuery comes with plenty of built-in animation effects which you can use in your websites.
- Lightweight – The jQuery is very lightweight library - about 60KB in size (Minified and gzipped).
- Cross Browser Support – The jQuery has cross-browser support, and works well in IE 6.0+, FF 2.0+, Safari 3.0+, Chrome and Opera 9.0+
- Latest Technology – The jQuery supports CSS3 selectors .

# *How to use jQuery?

- *Local Installation – You can download jQuery library on your local machine and include it in your HTML code.

- *CDN Based Version – You can include jQuery library into your HTML code directly from Content Delivery Network (CDN).

- 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)

# *Local JQuery

# JQUERY CDN

- To use jQuery from Google or Microsoft, use one of the Following
- Google CDN:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"
></script>
```

- Microsoft CDN:

```
<script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-
3.1.1.min.js"></script>
```

# *jQuery Syntax

- Basic syntax is: $(selector).action()
- $ sign to define/access jQuery
- (selector) to "query (or find)" HTML elements
- jQuery action() to be performed on the element(s)

# How to Call a jQuery Library Functions?

- As almost everything, we do when using jQuery reads or manipulates the document object model (DOM), we need to make sure that we start adding events etc. as soon as the DOM is ready.
- If you want an event to work on your page, you should call it inside the $(document).ready() function. Everything inside it will load as soon as the DOM is loaded and before the page contents are loaded.

```
$(document).ready(function() {
$("div").click(function() {
alert("Hello, world!");});
});
```

- The jQuery team has also created an even shorter method for the document ready event:

```
$(function(){

   // jQuery methods go here...

});
```

- Tag Name Represents a tag name available in the DOM. For example $('p') selects all paragraphs <p> in the document.
- Tag ID Represents a tag available with the given ID in the DOM. For example $('#some-id') selects the single element in the document that has an ID of some-id.
- Tag Class Represents a tag available with the given class in the DOM. For example $('.some-class') selects all elements in the document that have a class of some-class.

# jQuery Selectors

- The element selector selects all the elements that have a tag name of T.
- this selector also returns an array filled with the found elements.
- **$('p')** – Selects all elements with a tag name of **p** in the document.
- **$('div')** – Selects all elements with a tag name of **div** in the document.

# jQuery - Tag Name Selector

# Our Mission

“Hide all <div>s”

```
(1)  var divs = document.getElementsByTagName(‘div’);

(2)  for(var i=0; i < divs.length ; i++){
                 divs[i].style.display=“none”;
      };
```

*Let’s do some magic

# * Let's do some magic

**Our Mission**

“Hide all <div>s”

```
$(“div”).hide();
```

- The element ID selector selects a single element with the given id attribute.
- **$('#myid')** – Selects a single element with the given id myid.
- **$('div#yourid')** – Selects a single division with the given id yourid.

# jQuery - Element ID Selector

# jQuery - Element Class Selector

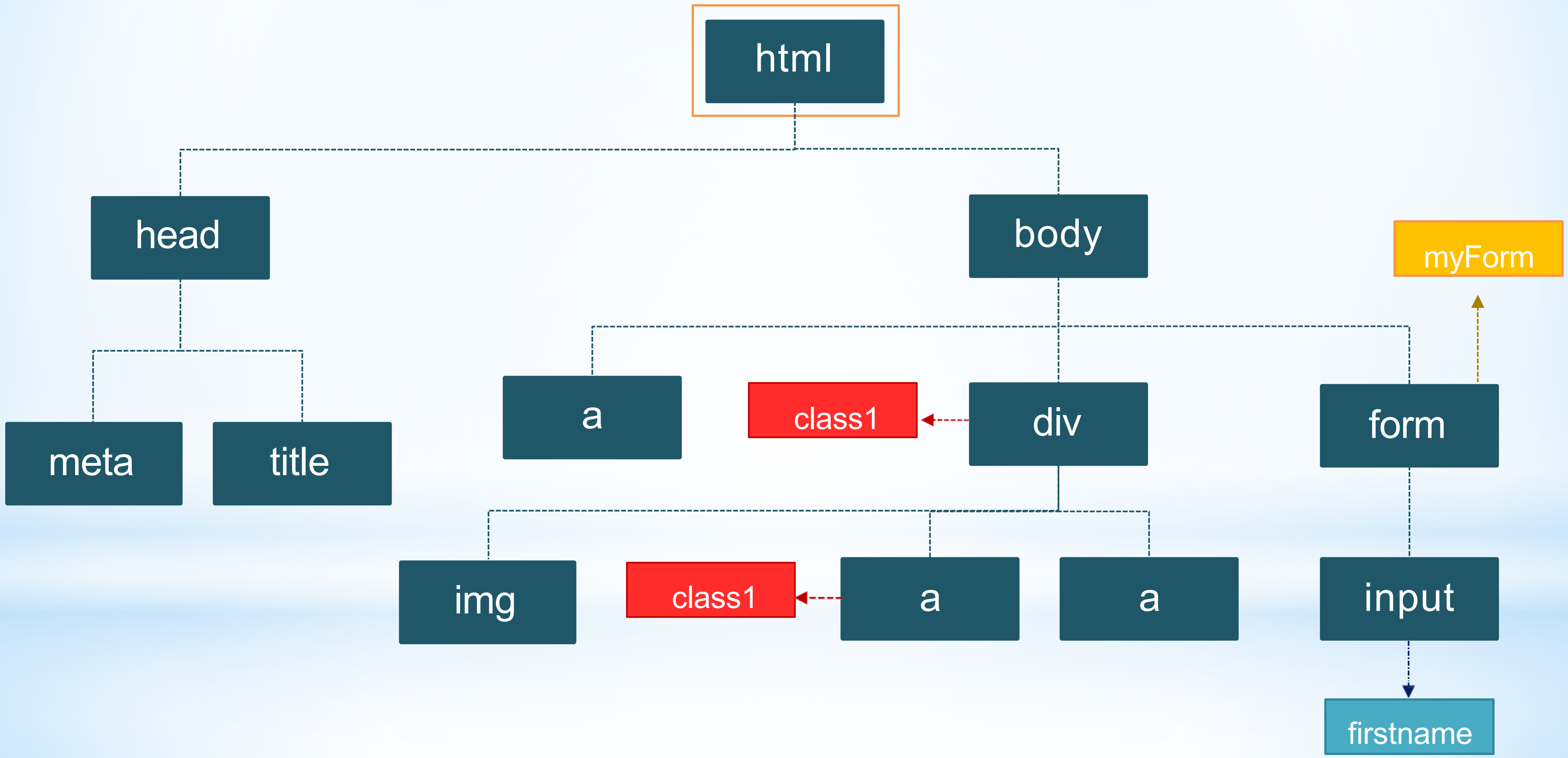
- The element class selector selects all the elements which match with the given class of the elements.
- this selector also returns an array filled with the found elements.
- **$('.big')** – Selects all the elements with the given class ID **big**.
- **$('p.small')** – Selects all the paragraphs with the given class ID **small**.
- **$('.big.small')** – Selects all the elements with a class of **big** and **small**.

# jQuery - Universal Selector

- The universal selector selects all the elements available in the document.
- this selector also returns an array filled with the found elements.
- **$('*')** selects all the elements available in the document.

# * Using CSSSelectors

$ ("*")

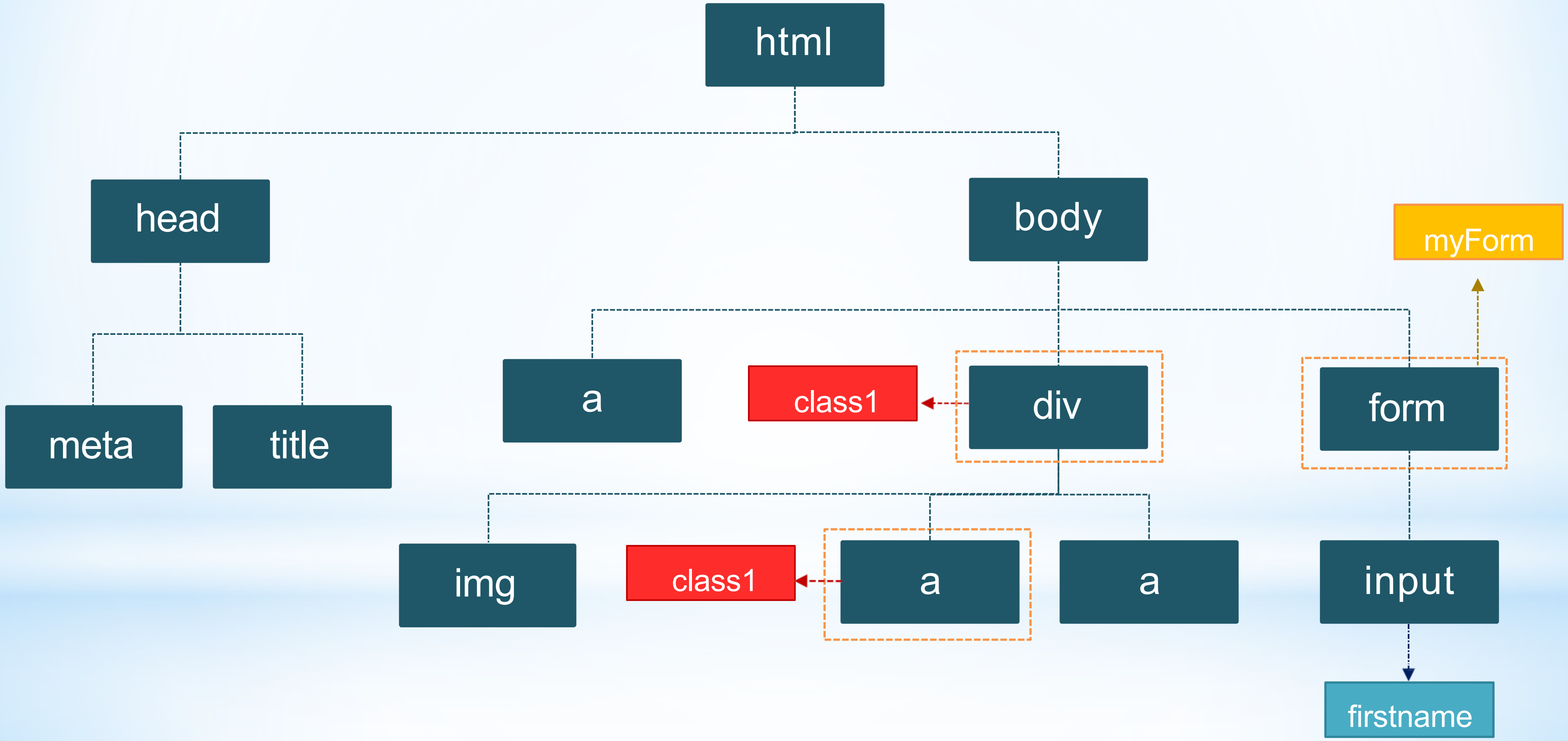


- This Multiple Elements selector selects the combined results of all the specified selectors E, F or G.
- You can specify any number of selectors to combine into a single result. Here order of the DOM elements in the jQuery object aren't necessarily identical.
- **$('div, p')** – selects all the elements matched by **div** or **p.**
- **$('p strong, .myclass')** – selects all elements matched by **strong** that are descendants of an element matched by **p** as well as all elements that have a class of **myclass**.
- **$('p strong, #myid')** – selects a single elements matched by **strong** that is descendant of an element matched by **p** as well as element whose id is **myid**.

# *Multiple Elements Selector

# * Using CSSSelectors

```
$("#myForm,.class1,div")
```



# Selectors Examples

- $("p.intro") Selects all <p> elements with class="intro“
- $("p:first") Selects the first <p> element
- $("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 NOT equal to "_blank“
- $(":button") Selects all <button> elements and <input> elements of type="button“

- $(':checked')
- $(':disabled')

- $('[title]') has the attribute
- $('[attr="val"]') attr equals val
- $('[attr!="val"]') attr does not equal val
- $('[attr^="val"]') attr begins with val
- $('[attr$="val"]') attr ends with val
- $('[attr*="val"]') attr has val anywhere

# Selectors Examples

- JQuery contains powerful methods for changing and manipulating HTML elements and attributes.
- Most of these attributes are available through JavaScript as DOM node properties. Some of the more common properties are −
- className – tagName – id – href – title -src

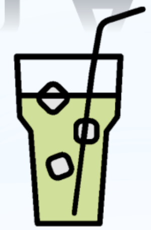
# Attributes

- *The attr() method can be used to either fetch the value of an attribute from the first element in the matched set or set attribute values onto all matched elements.

# *Get Attribute Value

# *Treating with Attributes

Do



**1** GET Attribute Value of an Element

```
$(“img”).attr(“src”);
```

Do

# *Treating with Attributes

## 1 GET Attribute Value of an Element

```
$(“img”).attr(“src”);
```

## 2 Set Attribute value

```
$(“img”).attr(“src”, “orange.png”);
```

# TEXT(), HTML(), AND VAL()

- Three simple, but useful, jQuery methods for DOM manipulation are:
- 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

Do

# Getting Contents

## 1 GET HTML Content

```
$(“div”).html();
```

## 2 GET Text Content

```
$(“p”).text();
```

## 3 GET Input Value

```
$(“input”).val();
```

# Setting Contents

## 4 Set HTML Content

```
$(“div”).html(“<p>some html</p>”);
```

## 5 Set Text Content

```
$(“p”).text(“some text”);
```

## 6 Set Input Value

```
$(“input”).val(“Hello There!”);
```

# *CHECK

.hasClass(class)

.is(selector)

** returns a boolean

Implicit Iteration

Explicit Iteration (Looping)

```
$('li').removeClass('myclass');       //implicit

$('li').each(function(index) {       //explicit
  $(this).append( ' #' + (index+1) );
});
```

# *LOOPING

# each

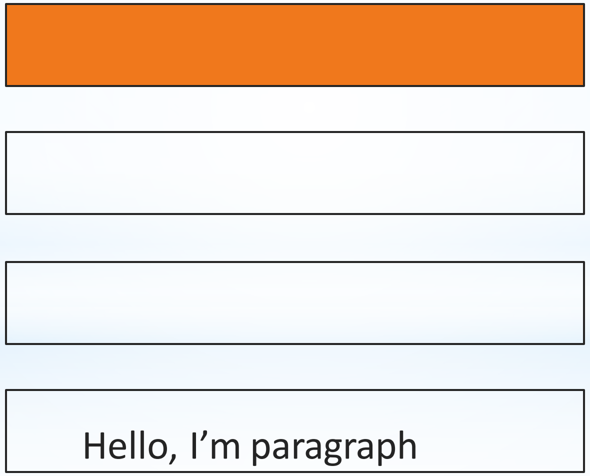
```
var colorArr = [‘orange’, ‘red’, ‘blue’, ‘green’];
```

- $(‘p’).each(function(index){
    - $(this).css(‘background’, colorArr[index]);
- });

- Hello, I’m paragraph

  Hello, I’m paragraph

  Hello, I’m paragraph



Hello, I’m paragraph

# each

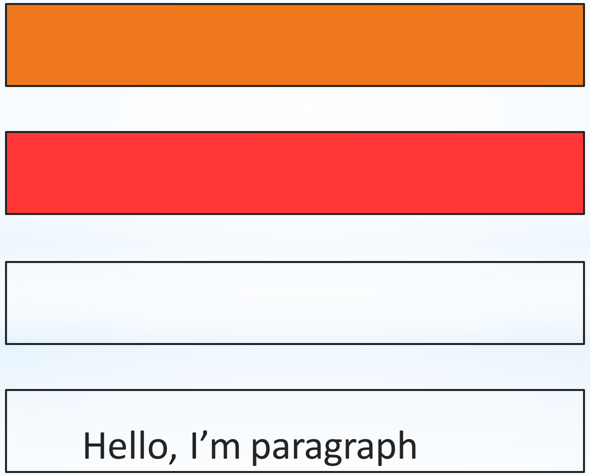
```
var colorArr = [‘orange’, ‘red’, ‘blue’, ‘green’];
```

```
$(‘p’).each(function(index){
    $(this).css(‘background’, colorArr[index]);
});
```

- Hello, I’m paragraph

Hello, I’m paragraph

Hello, I’m paragraph



Hello, I’m paragraph

# each

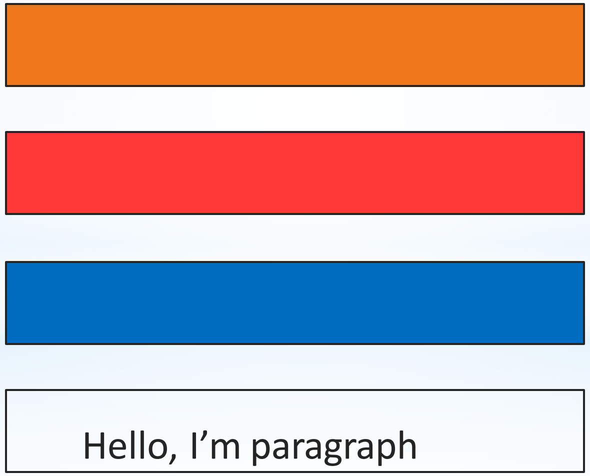
```
var colorArr = [‘orange’, ‘red’, ‘blue’, ‘green’];
```

- $(‘p’).each(function(index){
  - $(this).css(‘background’, colorArr[index]);
- });

- Hello, I’m paragraph

Hello, I’m paragraph

Hello, I’m paragraph


Hello, I’m paragraph

# each

```
var colorArr = [‘orange’, ‘red’, ‘blue’, ‘green’];
```

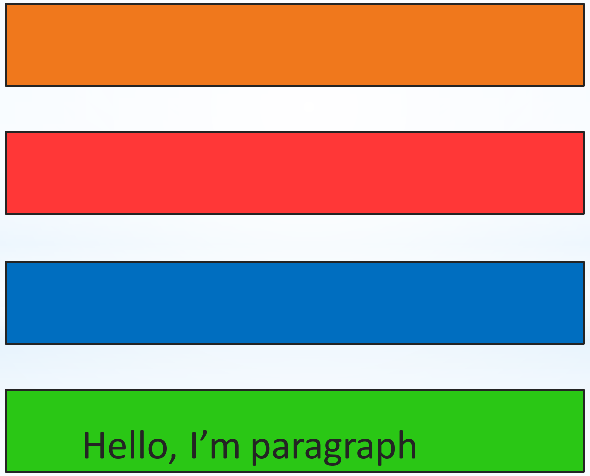
```
$(‘p’).each(function(index){
    $(this).css(‘background’, colorArr[index]);

});
```

- Hello, I’m paragraph

  Hello, I’m paragraph

  Hello, I’m paragraph


Hello, I’m paragraph