

Client-side Technologies

Eng. Niveen Nasr El-Den
iTi

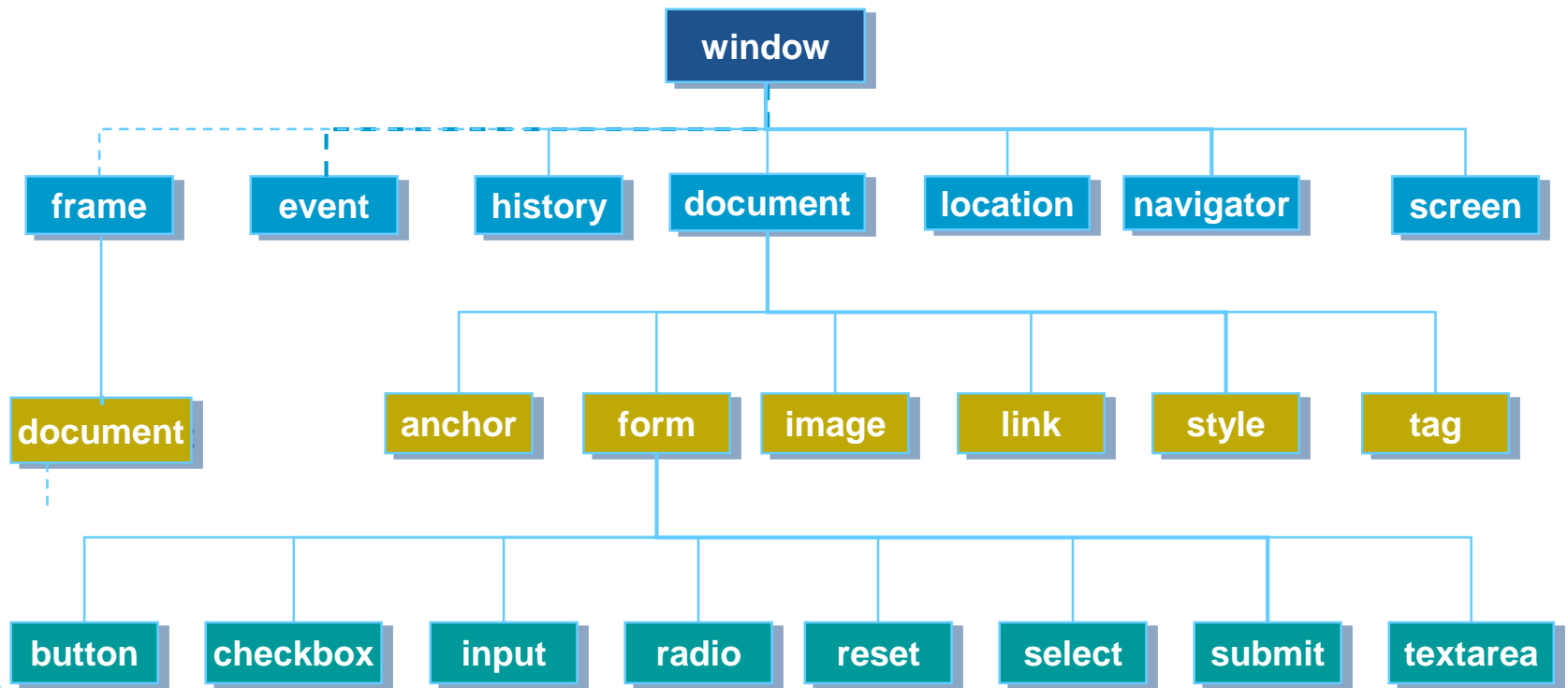
Day 6

Browser Object Model

DOM cont.

Model Hierarchy

BOM is a larger representation of everything provided by the browser including any other functionality the browser may expose to JavaScript.



Document Object

- The *document* object represents the entire HTML document and can be used to access all elements in a page.
- A *page* is what appears within the browser window.
- So, every window is associated with a document object.
- Document Object has its own set of Properties, Collections, Methods & Event handlers.

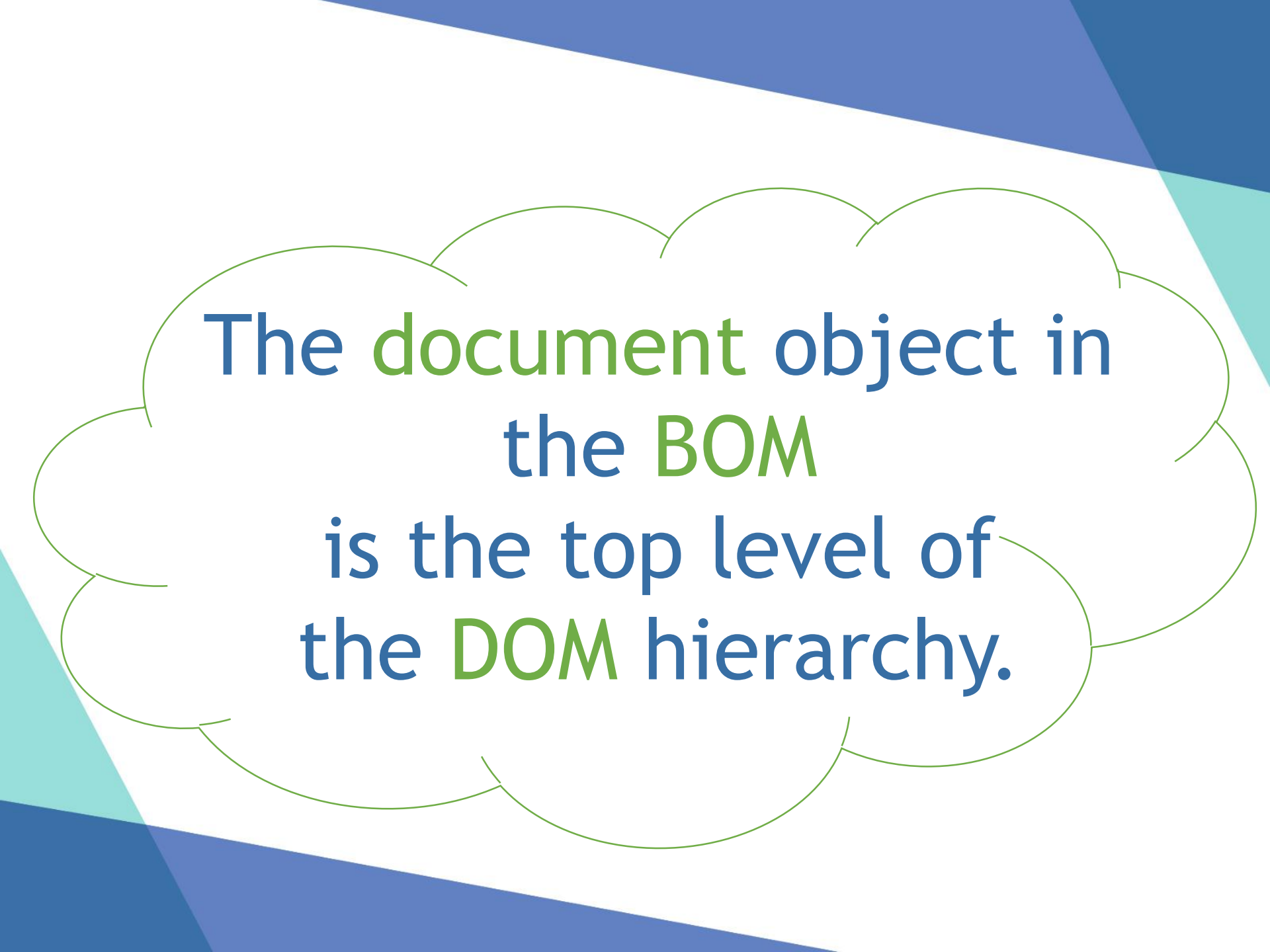
Document Properties

Property	Description
bgColor	A string that specifies the background color.
fgColor	A string that specifies the text color.
linkColor	The color of text for a link that the user has not yet visited.
vlinkColor	The color of text for a link that the user has already visited.
alinkColor	The color of text for a link that the user clicks.
title	A string that specifies the contents of the TITLE tag.
cookie	A string containing the name/value pair of cookies in the document.

Document Methods

Method	Description
write()	Writes one or more HTML expressions to a document in the specified window.
writeln()	Writes one or more HTML expressions to a document in the specified window and follows them with a new line character.

Example!



The document object in
the BOM
is the top level of
the DOM hierarchy.

DOM

methods and properties, allows
accessing any element on the page,
modify, **delete** or **remove** elements,
or **add** new ones.

Document Methods

for accessing document elements

Method	Description
<code>getElementById("id")</code>	For accessing any element on the page via its ID attribute
<code>getElementsByName("name")</code>	Returns a collection of objects with the specified name
<code>getElementsByTagName("tagName")</code>	Returns a collection of objects with the specified tagname

Example!

New HTML5 Selectors

- In HTML5 we can select elements by ClassName

```
var elements = document.getElementsByClassName('entry');
```

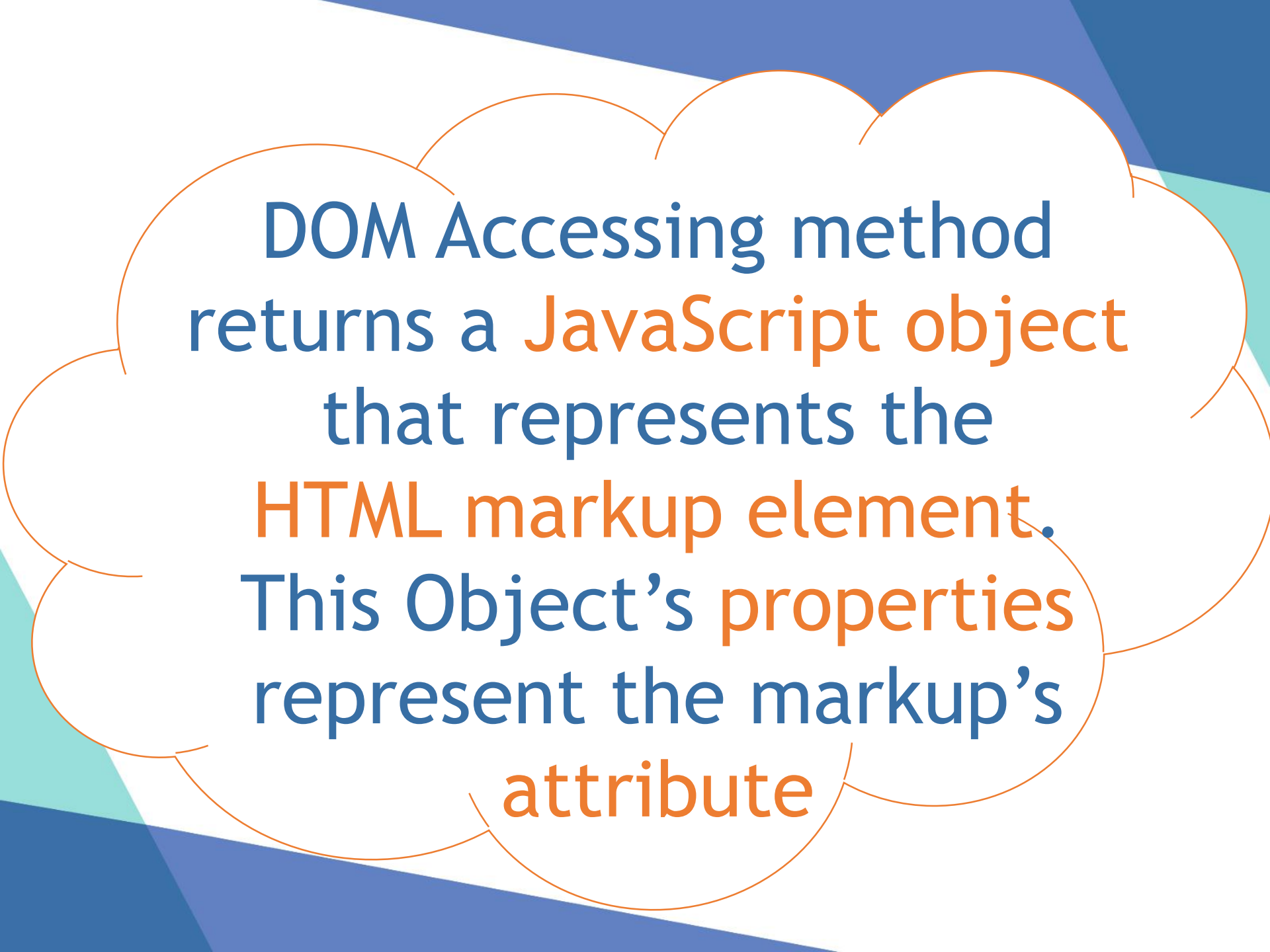
- Moreover there's now possibility to fetch elements that match provided CSS syntax

```
var elements = document.querySelectorAll(".someClasses");
```

```
var elements = document.querySelectorAll("div,p");
```

```
var elements = document.querySelector("#someID");
```

```
var first_td = document.querySelector("span");
```



DOM Accessing method
returns a JavaScript object
that represents the
HTML markup element.
This Object's properties
represent the markup's
attribute



We can access any
markup content via
`innerHTML`, `innerText`, or
`textContent`
properties

Document Collection

- links, anchors, images, forms are collection/array containing all occurrences of those objects within the document.
- Since they are treated as arrays, they have the *length* property which specifies the number of entries in the collection/array.
- To access a specific entry in any of these collections, we can use either their index or name.

Document Collection

Collection	Description
forms[]	An array containing an entry for each form in the document
images[]	An array containing an entry for each image in the document
anchors[]	An array containing an entry for each anchor in the document.
links[]	An array containing an entry for each link in the document.

Document Collection

- An item from an object collection can be referenced in one of the following ways:
 1. `collection[i]`
 2. `collection.item(i)`
 3. `collection.namedItem(id)`
 4. `collection["name"]`
 5. `collection["id"]`
 6. `collection.name`
- Example:
 - `document.forms[0]`
 - `document.forms["myForm"]`
 - `document.forms["frmId"]`
 - `document.myForm`

Document Event Handler

onclick
ondblclick
onkeydown
onkeypress
onkeyup
onmousedown
onmouseup

Image

- The Image object is an image on an HTML form, created by using the HTML '**IMG**' tag.
- Any images created in a document are then stored in an array in the **document.images** property.
- Image Object has its own set of Properties, Methods & Event handlers.

Image

- Object Model Reference:

[window.]document.imageName

[window.]document.imageID

[window.]document.images[i]

[window.]document.images[imageName]

document.img1.src="img1.jpg"
document.images[0].src="img1.jpg"

document.img2.src="img2.jpg"
document.images[1].src="img2.jpg"

```
<html>
<body>
  ....
  
  
  ...
</body>
</html>
```

Image Properties & Event handlers

Properties

name	id	src	height	width
------	----	-----	--------	-------

Event handlers

onabort	onload	onerror	onclick	ondblclick	onmouseover
---------	--------	---------	---------	------------	-------------

Example!

Form

- By using the form you have at your disposal; information about the elements in a form and their values.
- A separate instance of the form object is created for each form in a document.
- Objects within a form can be referred to by a numeric index or be referred to by name.
- Object Model Reference:
 - [window.]document.formname
 - [window.]document.forms[i]
 - [window.]document.forms["formNAME"]
 - [window.]document.forms["formID"]

Form

Properties

```
<form  
  [name="formName"]  
  [target="frameName or windowName"]  
  [onsubmit="handlerText Or Function"]  
  [onreset="handlerText Or Function"]  
>  
</form>
```

Events

Form Properties

Property	Description
elements[]	An array containing all of the elements of the form. Use it to loop through form easily.
length	The number of elements in the form.
name	The name of the form.
id	The id of the form.
target	The name of the target frame or window form is to be submitted to.

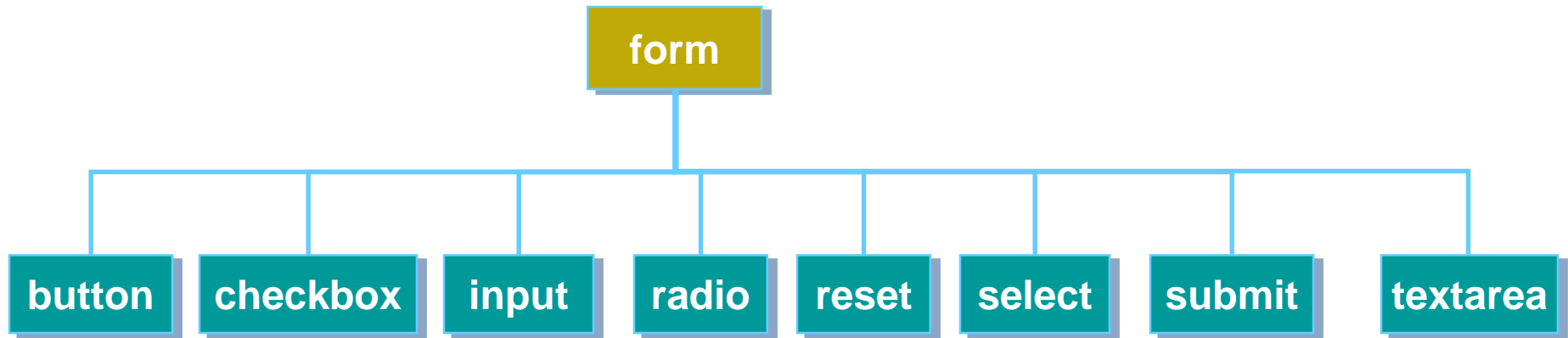
Form Methods

Method	Description
reset()	Resets the form. Clicking the reset button clears all contents that the user has made..
submit()	Submits a form. Clicking the submit button submits the content of the form to the server

Form Event Handler

Event	Description
onreset	Code is executed when the form is reset (by clicking on "reset" button)
onsubmit	Code is executed when form is submitted

Form



Text

```
<input type="text"  
      id="id"  
      value="string"  
      maxlength="n"  
      size="x"  
>
```

Text Event Handler

Event	Event Handler	Description
focus	onfocus	Fires when the field gains focus when the user tabs into or clicks inside the control.
blur	onblur	Fires when the field loses focus when the user tabs from or clicks outside the control.
change	onchange	Fires after changing the field value and losing being focused.
input	oninput	Fires every time the field value changes.
select	onselect	Fires when the content of the field being selected.

Text Methods

Method	Description
blur()	Removes focus from the field.
focus()	Assigns focus to the field; places the cursor in the control.
select()	Selects, or highlights, the content of the control.

Example!

Drop-down lists

```
<select  
  id="id"  
  multiple  
  size="n"  
>  
  <option value="string" selected > label </option>  
  <option value="string2" > label2 </option>  
  ...  
</select>
```

Drop-down lists Properties

Property	Description
length	The number of options in the list.
selectedIndex	The index number, beginning with 0, of the selected option.
options[]	An array of the options in the list. Used to reference properties associated with the options; e.g., options[1].value or options[2].text.
selected	A true or false value indicating whether an option is chosen.
value	The value associated with an option
text label	The text label associated with an option.

Drop-down lists Event Handler

Event Handler	Description
onfocus	The control gains focus.
onblur	The control loses focus.
onchange	A different option from the one currently displayed is chosen.

Example!

Radio Button

```
<input type="radio"  
  id="id"  
  name="name"  
  value="string"  
  checked  
>
```

Radio Button Properties

Property	Description
length	The number of radio buttons with the same name.
checked	A true or false value indicating whether a button is checked.
value	The value attribute coded for a button. A checked button with no assigned value is given a value of "on".

Radio Button Event Handler

Event Handler	Description
onfocus	The control gains focus.
onblur	The control loses focus.
onclick	The button is clicked.

Example!

Checkbox

```
<input type="checkbox"  
  id="id"  
  name="name"  
  value="string"  
  checked  
>
```

Button

```
<input type="button"  
  id="id"  
  value="string"  
>
```

Button Event Handler

Event Handler	Description
onclick	The mouse is clicked and released with the cursor positioned over the button.
ondblclick	The mouse is double-clicked with the cursor positioned over the button.
onmousedown	The mouse is pressed down with the cursor positioned over the button.
onmouseout	The mouse cursor is moved off the button.
onmouseover	The mouse cursor is moved on top of the button.
onmouseup	The mouse button is released with the cursor positioned over the button.

Reminder DOM References

- Use **this** keyword is used to refer to the current object.
 - ▷ e.g. the calling object in a method.

- Self reference to the object is used :

```
<input type="text"  
      onfocus = "this.value='You are in!'" />
```

- Passing current Object as a function parameter:

```
function myFunction(myObject) {  
    myObject.value = "In the function!!"  
}  
  
<input type="text" onclick="myFunction(this)" />
```



Events & Event Handlers

Events

- We have the ability to create dynamic web pages by using *events*.
- Events are **actions** that **respond** to user's specific actions.
- Events are controlled in JavaScript using **event handlers** that indicate what **actions** the browser takes in **response** to an event.
- Examples for different events:
 - ▷ A mouse click
 - ▷ A web page loading
 - ▷ Taking mouse over an element
 - ▷ Submitting an HTML form
 - ▷ A keystroke on your keyboard

Events

- Event handlers are created as **attributes** added to the HTML tags in which the event is triggered. (first way of binding an event handler)
- An Event handler adopts the event name and appends the word **“on”** in front of it.
< tag onevent = “JavaScript commands;” >
- Thus the **“click”** event becomes the **onclick** event handler.

Mouse Events

Event handler	Description
<code>onmousedown</code>	when pressing any of the mouse buttons.
<code>onmousemove</code>	when the user moves the mouse pointer within an element.
<code>onmouseout</code>	when moving the mouse pointer out of an element.
<code>onmouseup</code>	when the user releases any mouse button pressed
<code>onmouseover</code>	when the user moves the mouse pointer over an element.
<code>onclick</code>	when clicking the left mouse button on an element.
<code>ondblclick</code>	when Double-clicking the left mouse button on an element.
<code>ondragstart</code>	When the user has begun to select an element

Keyboard Events

Event handler	Description
onkeydown	When User presses a key
onkeypress	When User presses a key other than Modifiers (ctrl, shft, ..etc.)
onkeyup	When User releases the pressed a key

Other Events

Event handler	Description
onabort	The User interrupted the transfer of an image
onblur	when loosing focus
onfocus	when setting focus
onchange	when the element has lost the focus and the content of the element has changed
onload	a document or other external element has completed downloading all the data into the browser
onunload	a document is about to be unloaded from the window
onerror	When an error has occurred in a script.
onmove	when moving the browser window

Other Events

Event handler	Description
onreset	When the user clicks the form reset button
onsubmit	When the user clicks the form submit button
onscroll	When the user adjusts an element's scrollbar
onresize	When the user resizes a browser window
onhelp	When the user presses the F1 key
onselect	When selecting text in an input or a textarea element
onstart	When a marquee element loop begins
onfinish	When a marquee object finishes looping
onselectstart	When the user is beginning to select an element

Binding Events

- **Binding Event Handlers to Elements can be:**
 - 1.Event handlers as tag attribute**
 - 2.Event handlers as object property**

Event handlers as tag attribute

```
<input type=button value="click me" name=b1  
  onclick="alert('you have made a click')">
```

OR

```
<script>  
function showmsg(){  
    alert("you have made a click")  
}  
</script>
```

```
<input type=button value="click me" onclick="showmsg()" />
```

Example!

Event handlers as object property

```
<body>
  <form>
    <input type=button name='b1' value="Click ME" />
  </form>
</body>
```

```
<script>
function showAlert (){
  alert("you have clicked me")
}
document.forms[0].b1.onclick=showAlert
</script>
```

Note: there are no parentheses



Event handlers as object property

```
<body>
  <form>
    <input type=button name='b1' value="Click ME" />
  </form>
</body>
```

```
<script>
document.forms[0].b1.onclick=function showAlert (){
    alert("you have clicked me")
}
</script>
```

Event handlers return value

```
<a href="1.htm" onclick="myFunc(); return false">
```

This will make the browser ignore the action of href

- Another way that can also make the browser ignore the action of href is:

```
<a href="javascript:void(0)" onclick="alert('hi')" >  
    click me  
</a>
```

To avoid refresh action
if href is empty

Example!

void Operator

- **void** Operator
 - ▷ A unary operator used to explicitly return undefined.
 - ▷ It can be used as shown
void expression;
void(expression);
- Example:
var val= void "javascript";
typeof val; //undefined

Assignment