Internet Programming

DOM & Dynamic HTML

Dan Brandon, Ph.D., PMP

Standard Object Types, Methods and Properties

- JS has a number of standard object types (including the browser itself) which have methods and properties
- There are some standard properties (i.e. “name’) that are common to many objects
- There are some standard methods (i.e. ‘click”) that are common to many objects
- The appendix of this lesson contains a description of the commonly used JS GUI objects with their properties and methods

Commonly Used Methods

alert(string)	shows string in modal window
anchor(string)	create anchor from string
back()	go back one page in history
blur()	remove focus from object
clearTimeout(id)	clears timeout id
click()	simulate object click
close()	close window or document
confirm(string)	presents confirmation window
escape(character)	returns hex value for character
eval(arith exp)	evaluates arithmetic expression
focus()	moves focus to the object

- forward() moves forward in history
- go(val | string) go to a history page
- link(URL string) make string into link object
- open(URL | MIME) open window or document
- parseInt(string, base) returns integer to base
- prompt(string, default) outputs prompt
- select() highlights an object (not same as focus)
- setTimeout(func, msec) invoke function after ms
- submit() simulate submit button
- unEscape(string) returns ascii character for string(%nn)
- write(string) outputs string to browser (w/o lf)
- writeln(string) outputs string with line feed

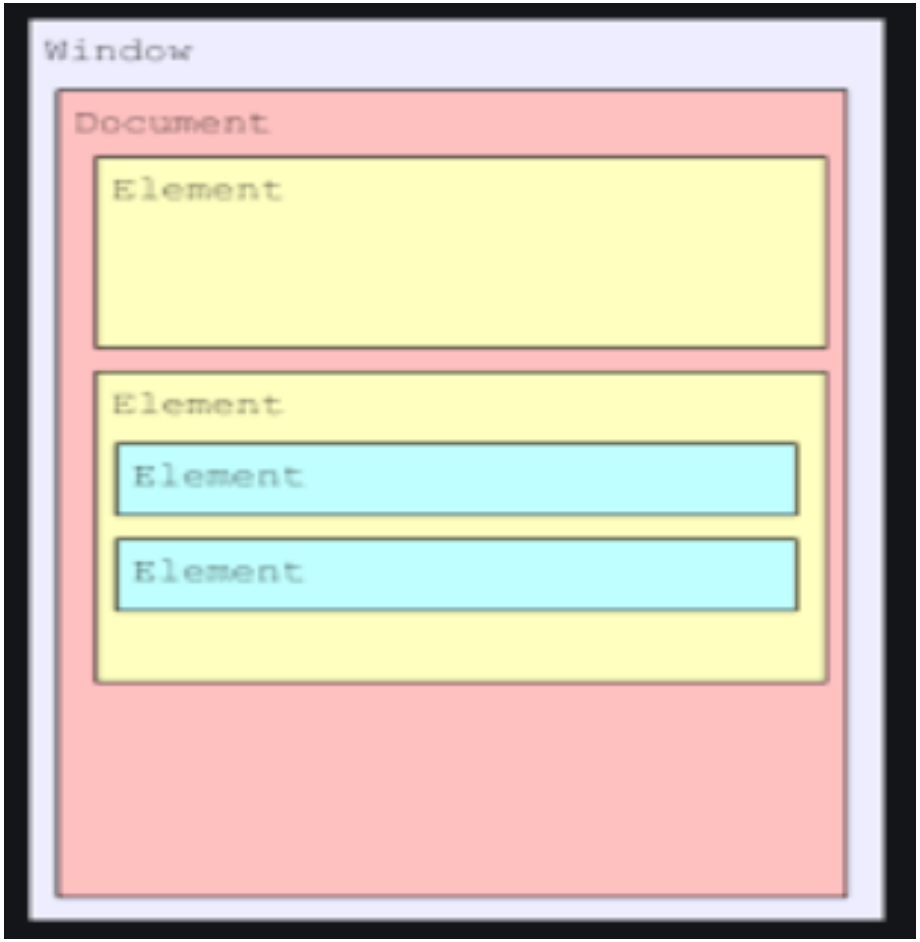
Dynamic Object Model (DOM)

- The standard objects defined in a browser having to do with HTML are defined in a “Document Object Model”
- The document object model has only recently become a standard and earlier versions (level 4 and below) of browsers have a somewhat different version of their own “Dynamic HTML Object Model”
- **A web page author is able to retrieve and modify the properties of each “object” of a web page**

DOM (con't)

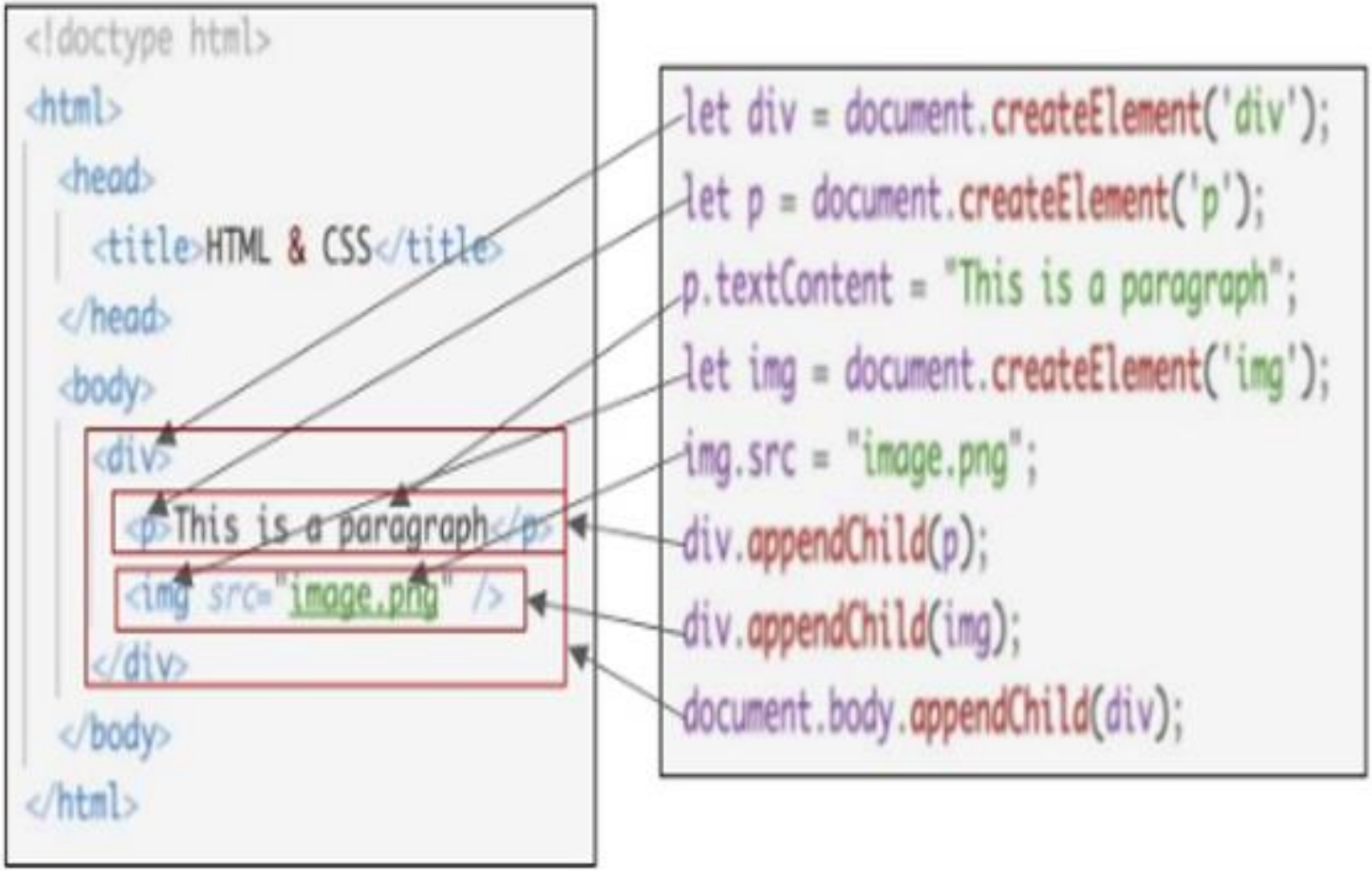
- The DOM is not part of JS [ECMA-262 (ISO-16262)]; it is standardized thru the World Wide Web Consortium (W3C)
- W3C has released multiple levels of "standard" DOM specs:
  - Level 1 deals with the document object and how to navigate within it
  - Level 2 deals with the event model and interfaces thereto
  - Level 3 deals with document loading and saving

Generic DOM Hierarchy

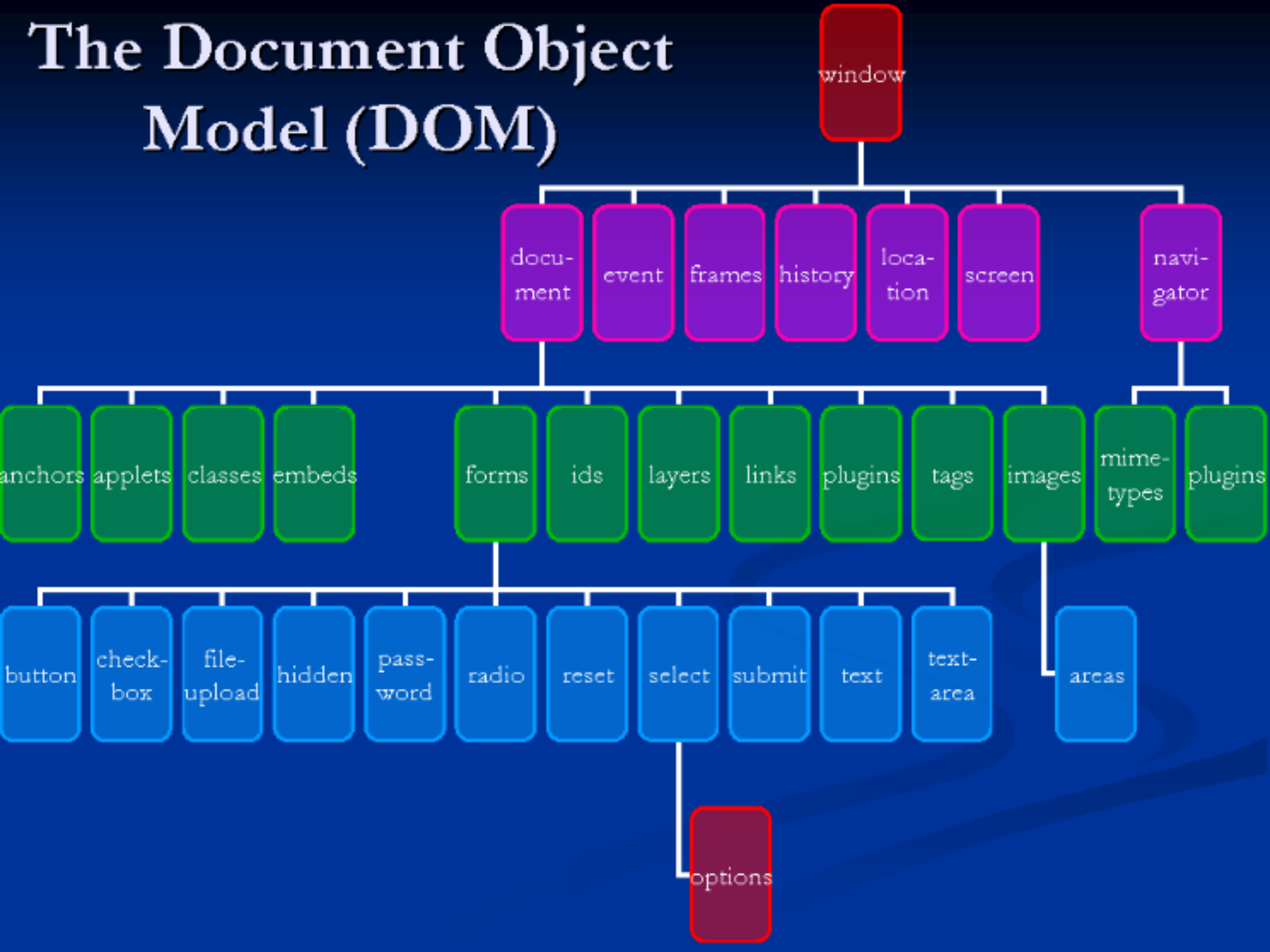


Nested Elements

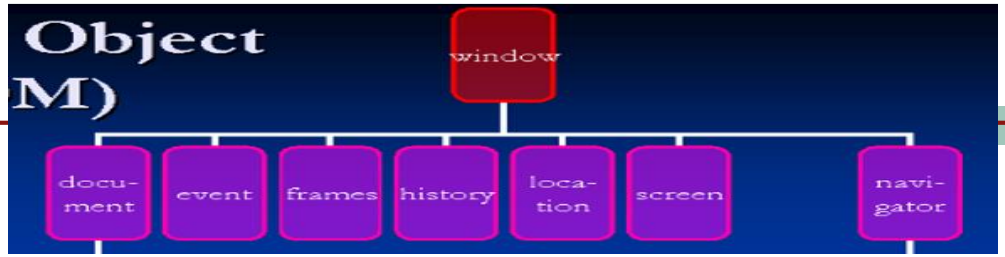
DOM and the JavaScript API



The Document Object Model (DOM)

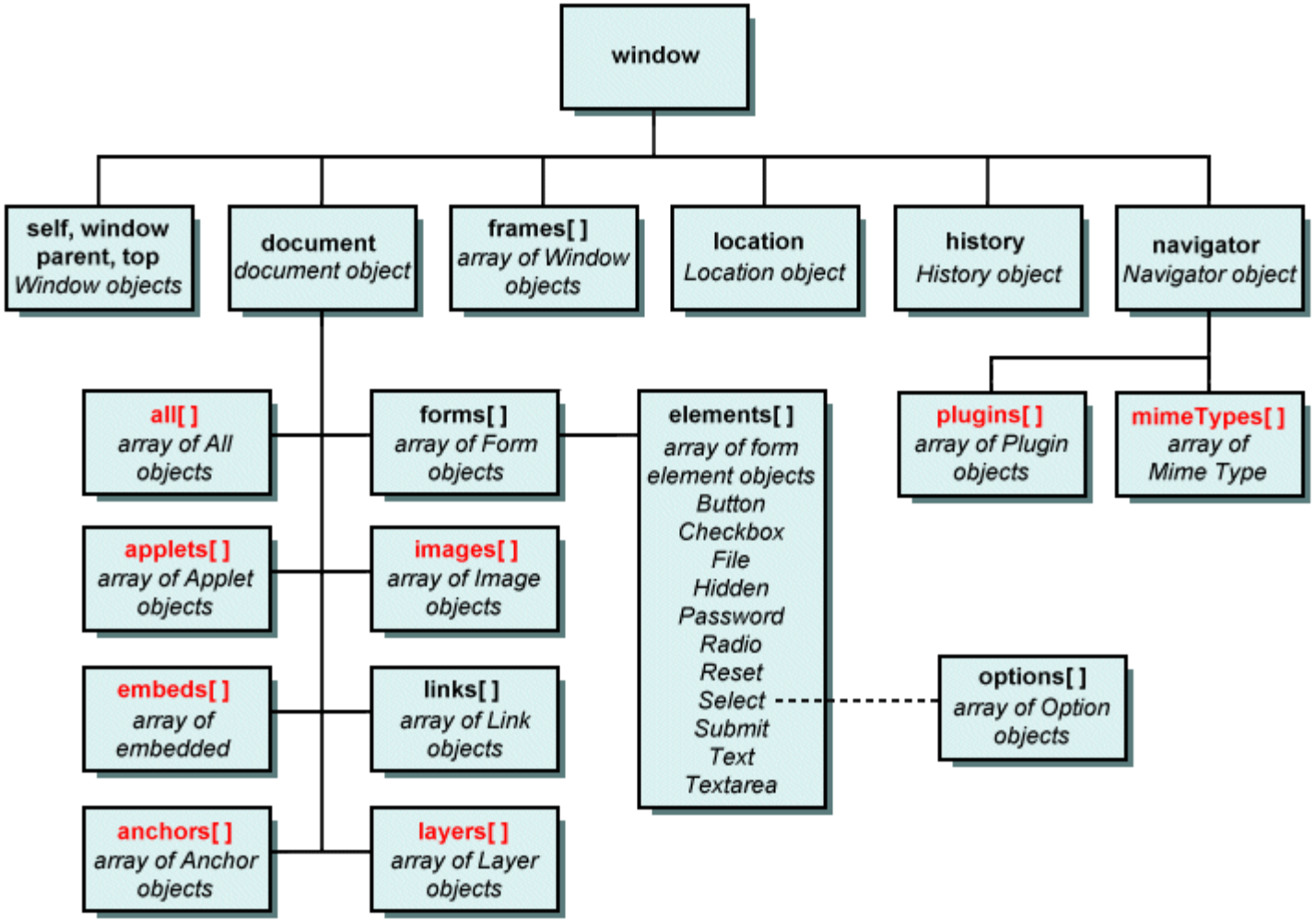


Top Level (window) DOM Definitions



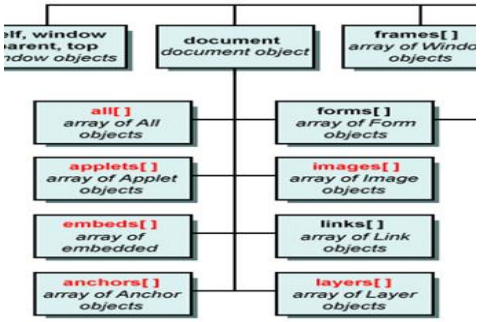
- Location – URL of current page
- Frames – A collection of the windows in the current window
- History – Collection of visited URL's
- Navigator – Browser information
- Event – The event currently being processed
- Screen – Screen and display setup info
- Document – The page currently in the browser window

Window Detailed DOM (items in "[]" are arrays)

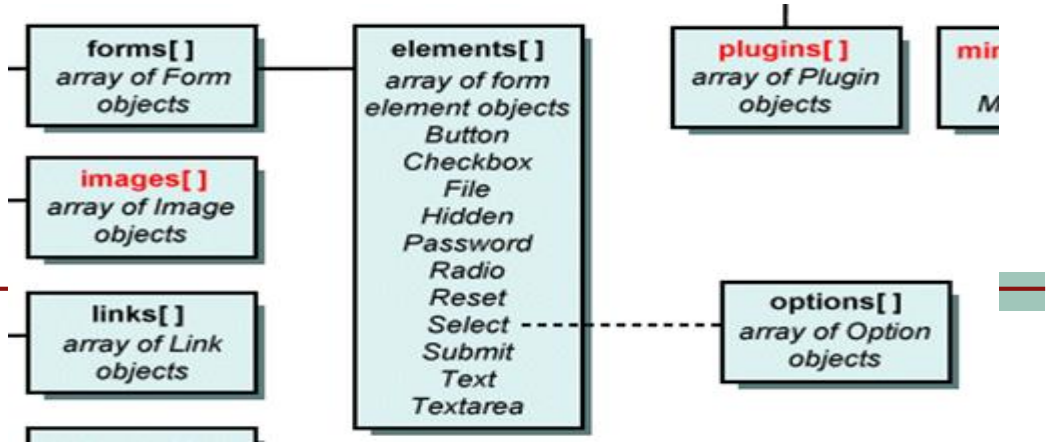


Document Level DOM Definitions

- Links – All of the A tags with an HREF attribute
- Anchors - All of the A tags with an ID or Name attribute
- Images – All of the IMG tags
- Forms – All of the FORM tags
- Applets – All of the Java APPLET tags
- Embeds – All of the EMBED tags
- Plug-ins – Same as EMBED
- Frames – All of the window objects for the frames in a document
- Scripts – All of the SCRIPT tags
- All – All of the elements in a document section (ie BODY)
- Stylesheets – All of the STYLE tags



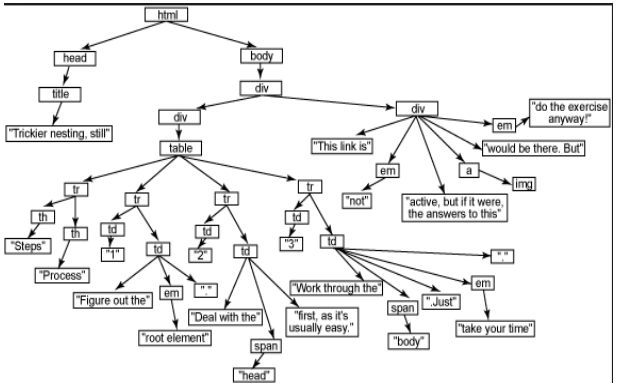
Forms



- Button — Used to access <input type="button">
- Checkbox — Used to access <input type="checkbox">
- Password — Used to access <input type="password">
- Radio — Used to access <input type="radio">
- Reset — Used to access <input type="reset">
- Select — Used to access an item in <select>
- Submit — Used to access <input type="submit">
- Text — Used to access <input type="text">
- TextArea — Used to access <textarea>

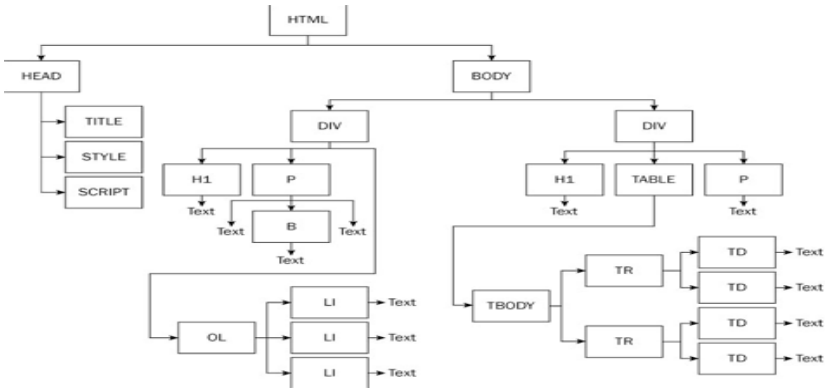
User Objects

- For other objects you create the names when you write the HTML code and specify a name:
  - <INPUT name="nameField" type = "text" size=20>
- "nameField" is an object, an instance of the object type "text"
- You use properties and methods with the dot notation:
  - nameField.someMethod();
  - nameField.someProperty=x;
  - x=nameField.someProperty;



User Objects (con't)

- Or specify an ID (not compatible with some earlier browsers, but prefered today) :
  - <P ID="paragraph1">…</P>
- Some properties are arrays
- Object properties are available (to get or change) only after the body of the document has been loaded



- Example:
  - <html><head><title>My Document</title></head>
  - <body bgcolor="#0000bb" fgcolor="#ffff00">
  - <form name="myForm">
    - <input type="text" name = "Person" size=35>
    -
  - </form>
  - </body></html>
- Some properties are:
  - document.title = "My Document"
  - document.fgColor=#ffff00
  - document.bgColor=#0000bb
  - document.href="http://www.somehost.com/somedoc.html
  - x = document.myForm.Person.value // value user keys into field

Element Identification

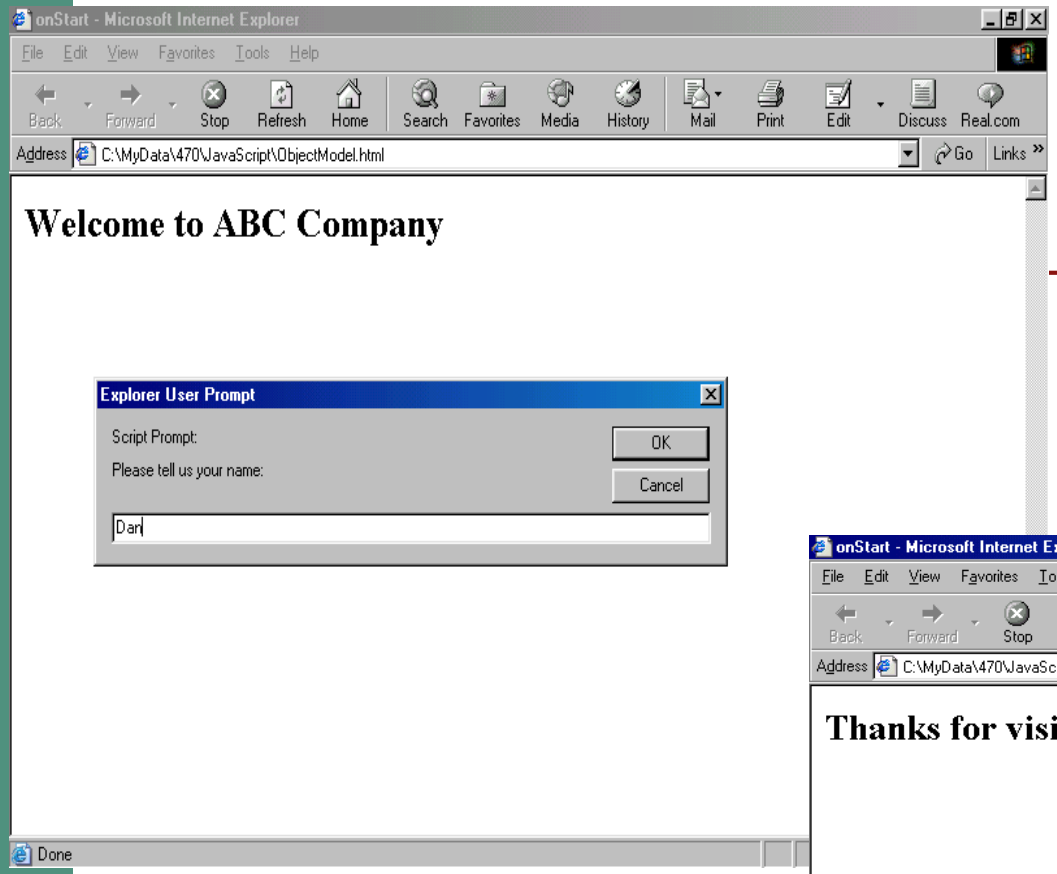
Try this !

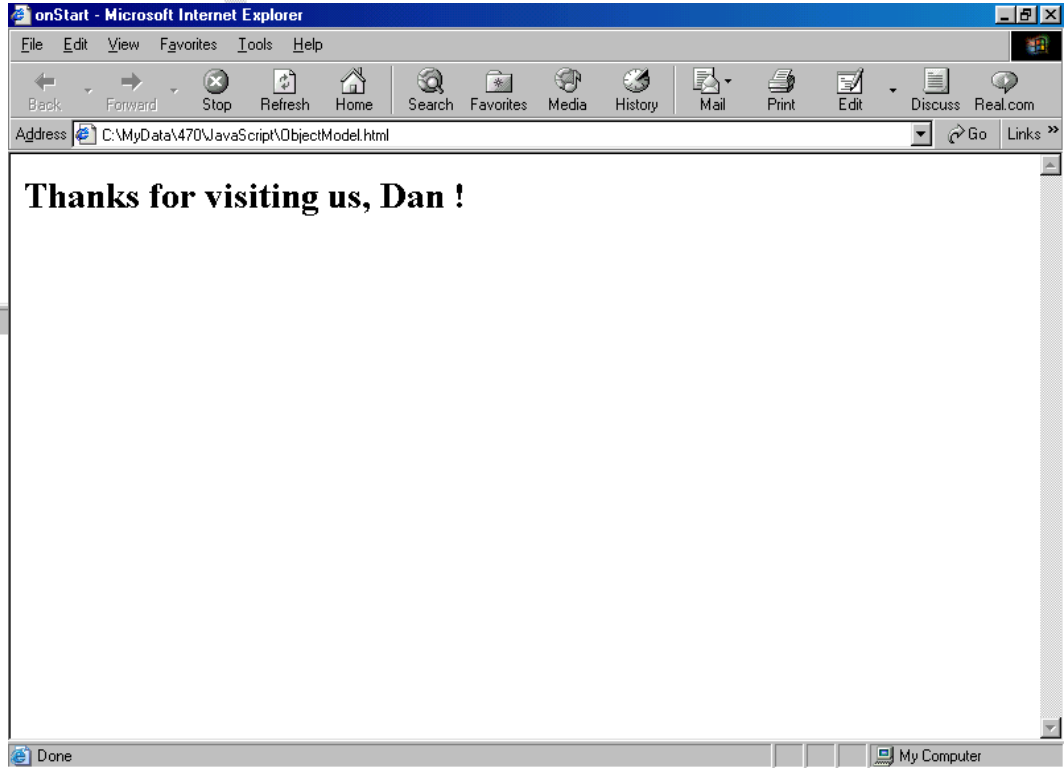
```
<HTML>
<HEAD><TITLE>onStart</TITLE>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
      var visitorName;
      function start() {
        visitorName = prompt ("Please tell us your name:","");
        mesg1.innerText = "Thanks for visiting us, " + visitorName + "
!";
      }

// -->
</SCRIPT>
<BODY ONLOAD="start()">
<H1 ID="mesg1">Welcome to ABC Company</H1>
</BODY></HTML>
```

May have problems with earlier browsers !

Can also use textContent

- mesg1.textContent = "Thanks for visiting us, " + visitorName + " !";


onStart - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites Media History Mail Print Edit Discuss Real.com
Address C:\MyData\470\JavaScript\ObjectModel.html
Go Links
Welcome to ABC Company
Explorer User Prompt
Script Prompt:
Please tell us your name:
OK
Cancel
Dan
Done


onStart - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites Media History Mail Print Edit Discuss Real.com
Address C:\MyData\470\JavaScript\ObjectModel.html
Go Links
Thanks for visiting us, Dan !
Done
My Computer

getElementById

- The getElementById() method returns the element that has the ID attribute with the specified value
- This method is one of the most common methods in the HTML DOM → getting an element and changing a property
- Returns *null* if no elements with the specified ID exists
- An ID should be unique within a page
  - However, if more than one element with the specified ID exists, the getElementById() method returns the first element in the source code
- Example:
  - var x = document.getElementById("demo"); // Get the element with id="demo"
    x.style.color = "red"; // Change the color of the element

getElementById (con’t)

- <html>
  - <body>
  - <script>
    - var aMessage = document.getElementById("aaa").innerHTML;
    - alert(aMessage);
  - </script>
  - Hello World!
  - </body>
- </html>

Tag Content

Property or Method	Description
element.innerHTML	Returns the HTML code within *element*
element.outerHTML	Returns the HTML code within *element* as well as the HTML code of *element* itself
element.textContent	Returns the text within *element* disregarding any HTML tags
element.insertAdjacentHTML (*position*, *text*)	Inserts HTML code defined by *text* into *element* at *position*, where *position* is one of the following: 'beforeBegin' (before the element's opening tag), 'afterBegin' (right after the element's opening tag), 'beforeEnd' (just before the element's closing tag), or 'afterEnd' (after the element's closing tag)

Standard Properties

[apply to one or more standard objects]

alinkColor	active link color
anchors	array of links in document
bgColor	background color
checked	checkbox or radio status
defaultChecked	is this default button ?
defaultSelected	default selection made ?
defaultStatus	default status bar message
defaultValue	default value of object

Standard Properties (con't)

elements	form elements array
fgColor	foreground color
forms	array of forms in document
frames	array of frames in window
hash	anchor name following #
host	host name and port
hostname	host name
href	URL
innerHTML	HTML inner content of tag
innerText	text content
lastModified	date of last modification

Standard Properties (con't)

length	length or number in
linkColor	link color
links	array of links in document
location	full URL
method	form posting method
name	object name
options	array of options in select
outerHTML	HTML of tag

Standard Properties (con't)

parent	window parent
pathname	pathname pat of URL
port	URL port number
protocol	protocol access method
selectedIndex	number value of option selected
self	current window
status	status bar message
target	targeted form response window
text	text after option tag
title	document title
value	value of an object's field
vlinkColor	visited link color
window	current window

Browser Incompatibilities

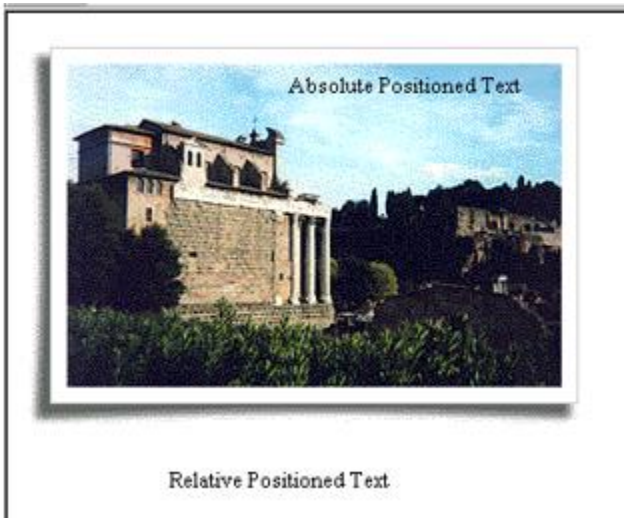
- Older browsers use somewhat different DOM's
- Also some vendors have more and extended versions of the built-in object functions
- It is difficult to get some JS and CSS features to work in all browsers
- To get your dynamic HTML to work in many browsers you have three choices:
  - ***Only use features that work in latest browsers !***
  - ***Use only the features that work in all browsers !***
  - ***Dynamically test to see which browser is being used, and then employ the objects and methods for that specific browser !***

Browser Incompatibilities (con't)

- Ideally you should be able to manipulate any web page element
- However because of the browser differences, you should create a "container" for all content (paragraphs, table cells, etc.) you wish to dynamically manipulate, and then manipulate the container
- Recommended containers are the DIV and SPAN tags: <DIV ID="…">…</DIV>
  - DIV is for content that starts on a new line, SPAN is for content on the same line
- Finally you need to test your HTML and JS code in each of the browsers (and also the different versions of each)

Item Identification and Positioning that Works in Most Browsers (with DIV container)

```
<HTML><HEAD>
<STYLE TYPE="text/css">
        #abs {position: absolute;
                top: 35px; left: 165px; z-index:2}
        #rel {position: relative;
                left: 85px; z-index: 2}
</STYLE></HEAD>
<BODY>
<IMG  SRC="castle.gif">
<DIV ID="abs">
        <P>Absolute Positioned Text</P>
</DIV>
<DIV ID="rel">
        <P>Relative Positioned Text</P>
</DIV>
</BODY></HTML>
```



Finding the Type of Browser

- You may need to examine the type of browser and take action accordingly due to browser's differences on implementation of JS features
- You can use the "navigator" object to find the type of browser:
  - navigator.appName and navigator.appVersion
- You can also test to see if a browser supports a feature:
  - var isIE = (document.all) ? 1:0; // check for IE
  - var isNS4 = (document.layers) ? 1:0; // check for Netscape
  - Var isDOM = (document.getElementById) ?1:0; // check for DOM

Navigator Object

Properties

Syntax: object.property_name

Property	Description	N	IE	W3C
appCodeName	Returns the code name of the browser	2	3.02	No
appMinorVersion	Returns the minor version of the browser		4	No
appName	Returns the name of the browser	2	3.02	No
appVersion	Returns the platform and version of the browser	2	3.02	No
browserLanguage	Returns the current browser language		4	No
cookieEnabled	Returns a Boolean value that specifies whether cookies are enabled in the browser	6	4	No
cpuClass	Returns the CPU class of the browser's system		4	No
onLine	Returns a Boolean value that specifies whether the system is in offline mode (In IE 4+ the user can choose to work offline by selecting File/Work Offline in the browser. When Work Offline is selected, the system enters an offline state, and content is read from the cache)		4	No
platform	Returns the operating system platform	4	4	No
systemLanguage	Returns the default language used by the OS		4	No
userAgent	Returns the value of the user-agent header sent by the client to the server	2	3.02	No
userLanguage	Returns the OS' natural language setting		4	No

Methods

Syntax: object.method_name()

Method	Description	N	IE	W3C
javaEnabled()	A Boolean value that specifies whether the browser has Java enabled	3	4 (MO)	No
taintEnabled()	A Boolean value that specifies whether the browser has data tainting enabled	3	4	No

Lab

Printing Browser Identification [browserID.html]

```
<HTML>
<HEAD><TITLE>Browser Identification</TITLE>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
     function start() {
          document.writeln("<P><H2>" + navigator.appName + "</H2></P>");
          document.writeln("<P><H2>" + navigator.appVersion + "</H2></P>");
     }
// -->
</SCRIPT>
<BODY ONLOAD="start()">
<H1>Browser ID Example</H1>
</BODY></HTML>
```

Try this in all browsers …

- Try to work class exercises without looking ahead !

Results of Browser ID Page

- In IE6:
  - **Microsoft Internet Explorer**
  - **4.0 (compatible; MSIE 6.0; Windows NT; DigExt)**
- **In IE7**
  - **4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 1.1.4322; IEMB3; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30; InfoPath.2; IEMB3)**
- **In IE9**
  - **5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/5.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; MDDR; .NET4.0C; InfoPath.3; yie9)**
- **In IE 11**
  - **5.0 (Windows NT 6.1; Trident/7.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; MDDR; .NET4.0C; .NET4.0E; Tablet PC 2.0; InfoPath.3; GWX:QUALIFIED; yie9; rv:11.0) like Gecko**
- In NAV 4.5
  - Netscape
  - 4.5 [en] (Win98; I)
- In NAV 6 & 7:
  - Netscape
  - 5.0 (Windows; en-US)
- In FireFox
  - Netscape
  - 5.0 (Windows; en-US)
- Safari
  - **Netscape**
  - **5.0 (Windows NT 6.1) AppleWebKit/534.52.7 (KHTML, like Gecko) Version/5.1.2 Safari/534.52.7**
- Opera
  - Opera
  - 9.80 (Windows NT 6.1; U; en)

Browser Properties

- Most of these properties come from one of three browser objects: Navigator, Screen, and Document
- When a web page loads, the browser sets several useful properties that describe the user's environment
- Some of these properties include the user's screen size, their operating system (platform), type of browser, and even the page that linked them to the current page

```
<script language="JavaScript">
function WriteFormProps(name,value)
{
        window.alert(name + ":" + value);
}

WriteFormProps("userAgent", navigator.userAgent);
WriteFormProps("browser", navigator.appName);
WriteFormProps("browseVersion", navigator.appVersion);
WriteFormProps("platform", navigator.platform);
WriteFormProps("browserCodename", navigator.appCodeName);
WriteFormProps("cookieEnabled", navigator.cookieEnabled);
WriteFormProps("language", (navigator.language ? navigator.language : navigator.userLanguage));
WriteFormProps("javaEnabled", navigator.javaEnabled());
WriteFormProps("screenWidth", window.screen.width);
WriteFormProps("screenHeight", window.screen.height);
WriteFormProps("screenAvailWidth", window.screen.availWidth);
WriteFormProps("screenAvailHeight", window.screen.availHeight);
WriteFormProps("screenColorDepth", window.screen.colorDepth);
WriteFormProps("referrer", document.referrer);
WriteFormProps("title", document.title);
WriteFormProps("URL", document.URL);
WriteFormProps("lastModified", document.lastModified);
</script>
```

JS to test for Browsers and Platforms {browserDetect.js}

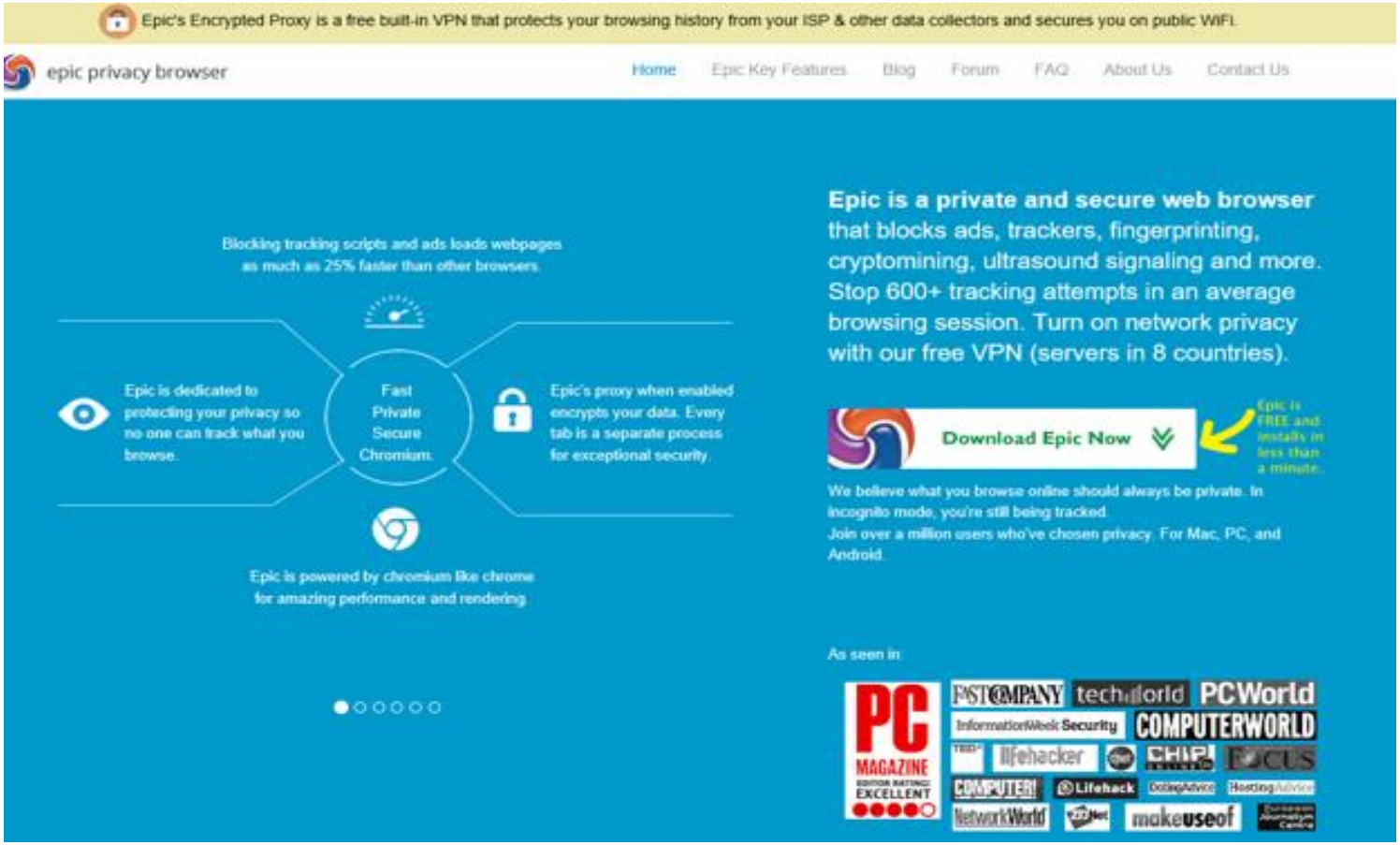
```
// code to detect browser and platform
        var isWin = false;
        var isMac = false;
        if (navigator.platform.indexOf("Win") != -1) isWin = true;
        if (navigator.platform.indexOf("Mac") != -1) isMac = true;
        var isNav = false;
        var isNav4 = false;
        var isNav5 = false;
        var isIE = false;
        var isIE4 = false;
        var isIE5 = false;
        if (navigator.appName == "Netscape")
        {
                  isNav = true;
                  if (parseInt(navigator.appVersion) == 4) isNav4 = true;
                  if (parseInt(navigator.appVersion) == 5) isNav5 = true;
        }
        if (navigator.appName == "Microsoft Internet Explorer")
        {
                  isIE = true;
                  if (navigator.appVersion.indexOf("MSIE 4") !=-1) isIE4 = true;
                  if (navigator.appVersion.indexOf("MSIE 5") !=-1) isIE5 = true;
        }
```

Browser & PC Information

- Information about your browser and your PC can be read by the web sites you visit (and by your organization's routers); see:
  - http://www.privacy.net/analyze
  - http://gemal.dk/browserspy/
- Javascript and Java Applets are the mechanisms within web pages that read browser info, cookies, and other info on one's hard drive
- To avoid having your information read and used, one needs to special browsers (EPIC or TOR - http://tor.eff.org/) or use proxy servers (Cloak - http://www.the-cloak.com/anonymous-surfing-home.html)

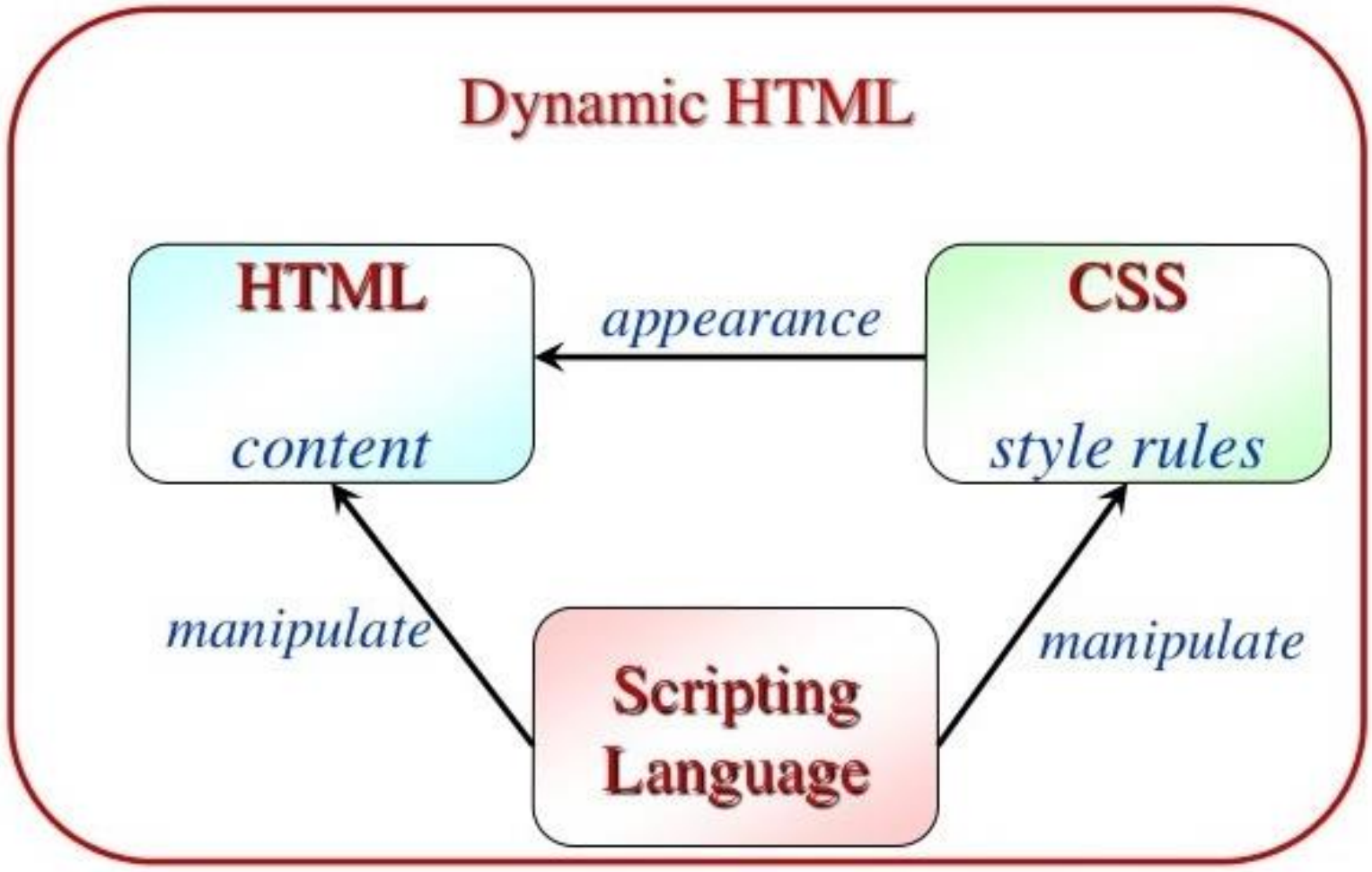
Anonymous Web Surfing - TOR

Anonymous Web Surfing - EPIC



Free Anonymous Proxy Server

Dynamic HTML



Opening and Writing Documents

- document.open(MIME)
- MIME (Multipurpose Internet Mail Extensions) is optional:
  - text/html - default
  - text/plain
  - text/css
  - image/gif
  - image/jpeg

Document.open (con't)

- Other MIME's can be used if appropriate "helper" processor (plug-in) is available
- Procedure:
  - document.open()
  - write stream (to MIME processor):
    - document.write() [Or document.writeln()]
  - document.close()

Dynamic HTML

- To write a table in the current document:
  - <script language="JavaScript">
    - document.write("<dl><dd><table border><tr><td>Cell 1</td>");
    - document.write("<td>Cell 2</td></tr>");
    - document.write(<tr rowspan=2><td>Cell 3</td></tr>");
    - document.write("</table></d1><p>");
  - </script>

“Inline JS” to Write HTML

```
<HTML>
<HEAD><TITLE>Inline Writing</TITLE>
</HEAD>
<BODY>
<H1>Hello …</H1>
<SCRIPT LANGUAGE="JavaScript">
<!--
      today = new Date();
      if ((today.getHours() >= 4) && (today.getHours() <= 12))
            document.write("<H2>Good Morning</H2>");
      if ((today.getHours() > 12) && (today.getHours() <= 18))
            document.write("<H2>Good Afternoon</H2>");
      if ((today.getHours() >= 18) && (today.getHours() <= 23))
            document.write("<H2>Good Evening</H2>");
      if ((today.getHours() >= 0) && (today.getHours() < 4))
            document.write("<H2>Why are you here at this hour
?</H2>");

// -->
</SCRIPT>
</BODY></HTML>
```

Changing Page in Current Window

- One way:
  - <HTML>…
  - <BODY ONCLICK="window.location.href='myNextPage.html'">
  - …
  - <H2>Click anywhere to go to next page</H2>
  - …
  - </BODY></HTML>
- Or with a function:
  - <HTML>…<HEAD>…
    - function changeSite() {
      - window.location.href = "myNextPage.html";
    - }
  - </HEAD>
  - <BODY>…
  - …ONCLICK="changeSite()"…
  - </BODY></HTML>

Events covered in more detail later !

Opening Other Windows

- **window.open(URL, “windowName”, [Options]);**
  - URL is the URL as a string; if blank, a blank window will be open
  - Options is a comma separated list of options with yes/no (or 1/0):
    - (toolbar, location, directories, status, menubar, scrollbars, resizable, copyhistory, width, height)
    - default is yes for all, except last two which default to current window size)

To open a window and place ‘myDoc’ in it:

- <A HREF=“myDoc.html” TARGET=“newWindow” onClick=“window.open(‘myDoc.html’, ‘newWindow’, ‘width=200, height=200, toolbar.resizable.status’)”>
- Click Here to Open myDoc</A>

Note: second argument on ‘open’ function must be valid window name [no spaces]

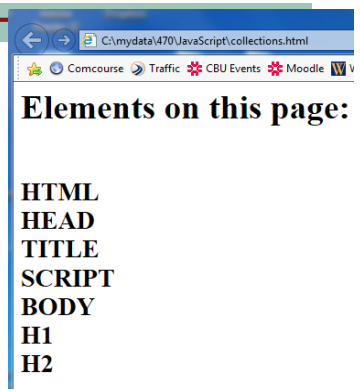
Open (con't)

- Function to open a new empty window and write a new document:
  - function newWin () {
    - content = "Some HTML formatted text";
    - props = "width=400,height=200,scrollbars=0,toolbar=0, location=0,directories=0,status=0,menubar=0";
    - w = window.open("", "HTMLWindow", props);
    - w.document.open();
    - w.document.write("<hr>");
    - w.document.write(content);
    - w.document.write("<hr>");
    - w.document.close();
  - }
- In a similar manner, a child window can modify content in a parent window

Collections

- A DOM collection is a JS array of objects
- One can process all the elements in a collection by walking thru the array with a JS “for” loop:
  - for (var i = 0; i < document.all.length, i++) {
    - // do something with document.all[i]
  - }
- One can process just the direct “children” of an element with object.children notation:
  - for (var i = 0; i < object.children.length, i++) {
    - // do something with object.children[i]
  - }

Example of Collection Processing [collections.html]

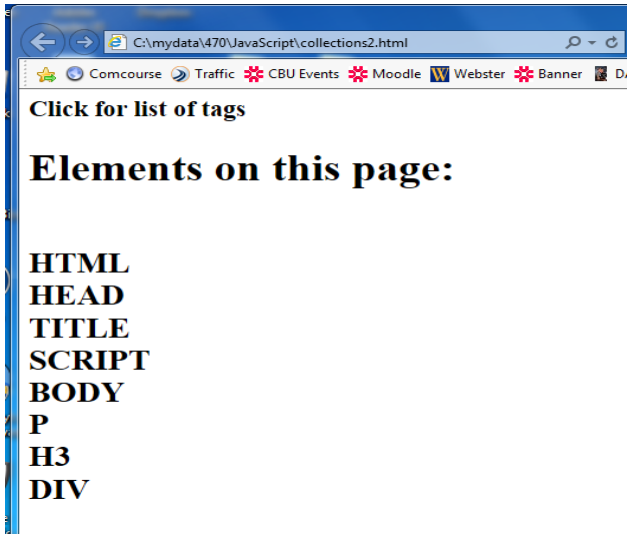



```
<HTML>
<HEAD><TITLE>Collections</TITLE>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
     var tags = "";
     document.writeln("<H1>Elements on this page:</H1><H2>");

     for (var i = 0; i < document.all.length; i++) {
          tags += "<BR>" + document.all[i].tagName;
     }
     document.writeln(tags);
// -->
</SCRIPT>
<BODY>
</BODY></HTML>
```

Try this !

Using “innerHTML” and Collections [collections2.html]

```
<HTML>
<HEAD><TITLE>Collections</TITLE>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
        function findTags() {
                var tags = "<H1>Elements on this page:</H1><H2>";
                for (var i = 0; i < document.all.length; i++) {
                        tags += "<BR>" + document.all[i].tagName;
                }
                d1.innerHTML = tags;
        }
// -->
</SCRIPT>
<BODY onClick="findTags();">
<P><H3>Click for list of tags</H3>
<DIV ID="d1">...</DIV>
</BODY></HTML>
```



References

- DOM Scripting: Web Design With Javascript and the Document Object Model by Jeremy Keith
- Learning JavaScript: The non-boring beginner's guide to modern (ES6+) JavaScript programming Vol 2: DOM manipulation by Marco Emrich and Christin Marit
- JavaScript & DOM Tips, Tricks, and Techniques (Volume 3) (JS Tips) by Louis Lazaris

Homework

- Hybrid → Appendix on GUI objects
- Re-do the “please tell us your name” web page using the getElementById function

```
<HTML>
<HEAD><TITLE>onStart</TITLE>
</HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
      var visitorName;
      function start() {
        visitorName = prompt ("Please tell us your name:","");
        mesg1.innerText = "Thanks for visiting us, " + visitorName + " !";
      }

// -->
</SCRIPT>
<BODY ONLOAD="start()">
<H1 ID="mesg1">Welcome to ABC Company</H1>
</BODY></HTML>
```

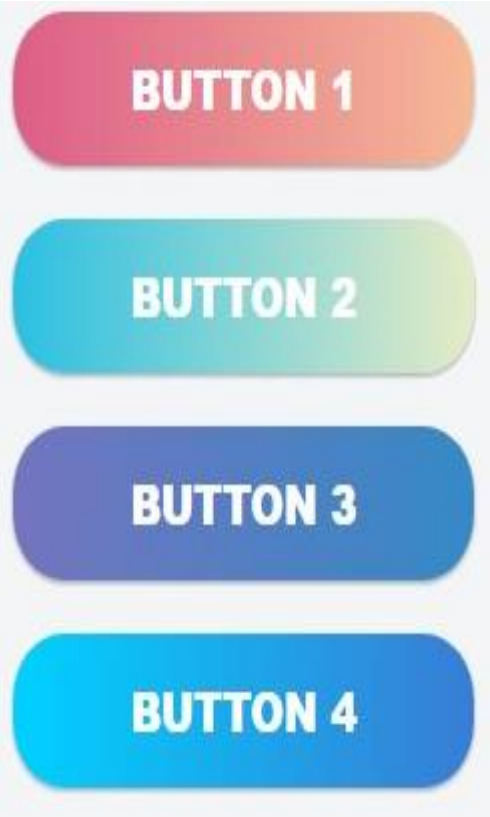
JAVASCRIPT'S STANDARD GUI OBJECTS & COMMONLY USED PROPERTIES/METHODS (ALPHA ORDER)

Anchor

- Properties:
  - name
- Methods:
- Event handlers:

Button

- Properties:
  - name
  - value
- Methods:
  - click()
- Event handlers:
  - onClick



Checkbox

- Properties:
  - name
  - checked
  - defaultChecked
  - value
- Methods:
  - click()
- Event handlers:
  - onClick

Document

- Properties:
  - alinkColor
  - anchors
  - bgColor
  - cookie
  - fgColor
  - form
  - forms
  - lastModified
  - linkColor
  - links
  - loadedDate
  - location

Document (con’t)

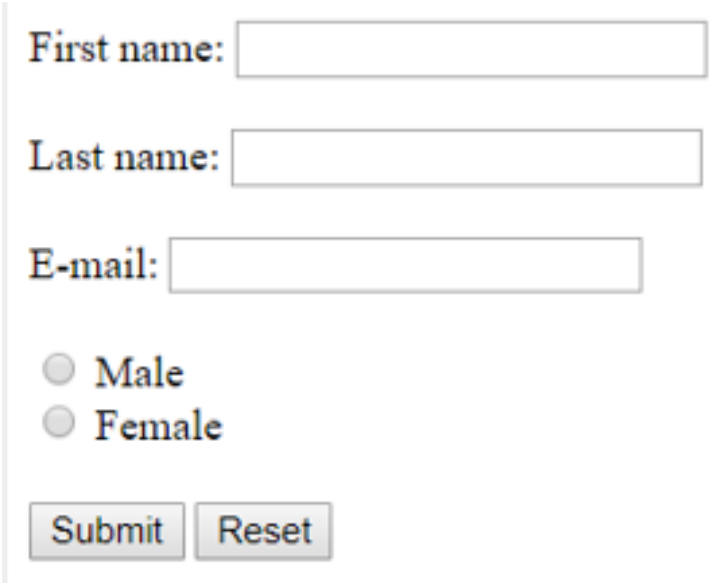
- referrer
- title
- userAgent
- vlinkColor

- Methods:
  - clear()
  - close(0
  - open()
  - write()
  - writeln()

- Event handlers:
  - onLoad
  - onUnLoad

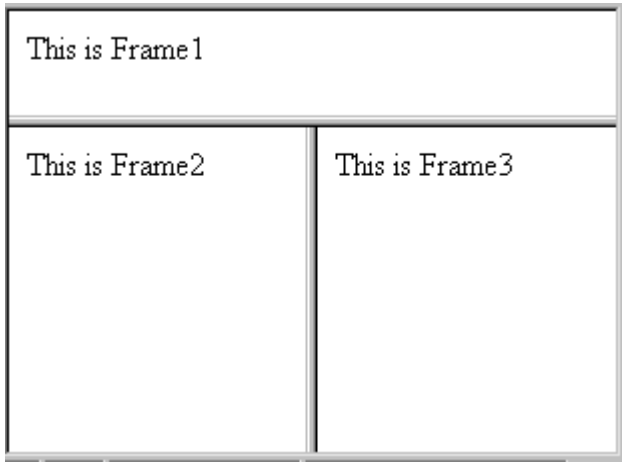
Form

- Properties:
  - action
  - elements
  - encoding
  - method
  - name
  - target
- Methods:
  - submit()
- Event handlers:
- onSubmit



Frame

- Properties
  - frames
  - parent
  - self
  - top
  - window
- Methods
  - alert(message)
  - clearTimeout(name)
  - close()
  - confirm(message)
  - prompt(message, defaulttext)
  - setTimeout(expression/function, time)



Hidden

- Properties
  - name
  - value

History

- Properties:
  - current
  - length
- Methods:
  - back(0
  - forward()
  - go(location)
- Event handlers:

Image

- Properties
  - border
  - complete
  - height
  - hspace
  - lowsrc
  - src
  - vspace
  - width

Link

- Properties:
  - target
- Methods:
- Event handlers:
  - onClick
  - onMouseOver

```
<body>
    <a href="http://www.google.com">click here</a>
    <br>
    <a href="http://www.fb.com">click here</a>
</body>
```

Location (URL)

- Properties:
  - hash
  - host
  - hostname
  - href
  - pathname
  - port
  - protocol
  - search

Location (con't)

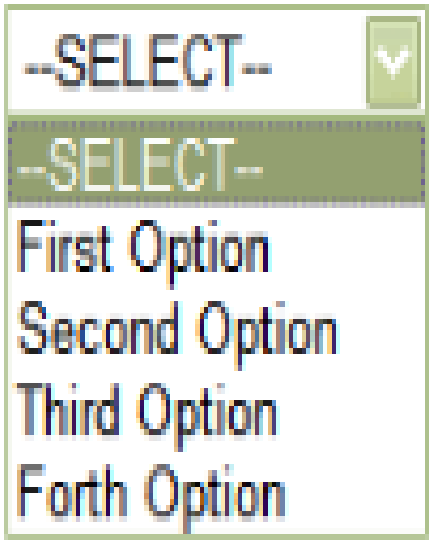
- Methods:
  - assign()
  - toString()
- Event handlers:

Navigator

- Properties
  - appCodeName
  - appName
  - appVersion
  - userAgent

Option

- Properties
  - defaultSelected
  - index
  - selected
  - text
  - value



Password

- Properties:
  - defaultValue
  - name
  - value
- Methods:
  - focus()
  - blur()
  - select()
- Event handlers:

Radio

- Properties:
  - checked
  - defaultChecked
  - index
  - length
  - name
  - value
- Methods:
  - click()
- Event handlers:
  - onClick

Checkbox and Radio Button

- The checkbox is presented as a small square box on the screen; it has two states: checked and unchecked
- When checked, the square will be filled with a check mark
- Checkboxes are used to present a range of options, from which the user may select any number of options to complete their task
- Within a group of checkboxes, each operates individually, so the user can check or uncheck options independently
- The radio button is presented as a small circle on the screen
- It also has two states, and when selected, the circle is filled with a solid dot
- In contrast to checkbox groups, radio button groups behave as a single control and limit the user's choice to just one option from the range provided

Reset

- Properties:
  - name
  - value
- Methods:
  - click()
- Event handlers:
  - onClick

Select

- Properties:
  - defaultSelected
  - index
  - length
  - Name
  - options
  - selectedIndex
  - text
  - value

Select (con't)

- Methods:
  - click()
- Event handlers:
  - onChange
  - onBlur
  - onFocus

HTML SELECT & OPTION

- HTML <select> tag is used to create drop down list of options, which appears when the user clicks on form element, and it allows to choose one of the options
- The <option> tag is used to define the possible options to choose from. The tag is put into the <select> tag
- The first option from the list of options is selected by default; to change a predefined option, the selected attribute is used
- The <optgroup> tag is used to group several options into one group; the content of <optgroup> looks like heading in bold
- The look of the list depends on the size attribute, which defines the height of the list, the width of the list depends on the length of the text inside <option>; the width can also be regulated with CSS styles

Submit

- Properties:
  - name
  - value
- Methods:
  - click()
- Event handlers:
  - onClick

Text

- Properties:
  - defaultValue
  - name
  - value
- Methods:
  - blur()
  - focus()
  - select()

Text (con't)

- Event handlers:
  - onBlur
  - onChange
  - onFocus
  - onSelect

TextArea

- Properties:
  - defaultValue
  - name
  - value
- Methods:
  - blur()
  - focus()
  - select()

Text

TextArea

TextArea (con't)

- Event handlers:
  - onBlur
  - onChange
  - onFocus
  - onSelect

Window

- Properties:
  - defaultStatus
  - frames
  - Length
  - name
  - parent
  - self
  - status
  - top
  - window

Window (con't)

- Methods:
  - alert(message)
  - clearTimeout(name)
  - close()
  - confirm(message)
  - open()
  - prompt(message, defaultText)
  - setTimeout(expression/function, time-ms)
- Event handlers:
  - onLoad
  - onUnLoad