



Scripting and Web Applications EE1081

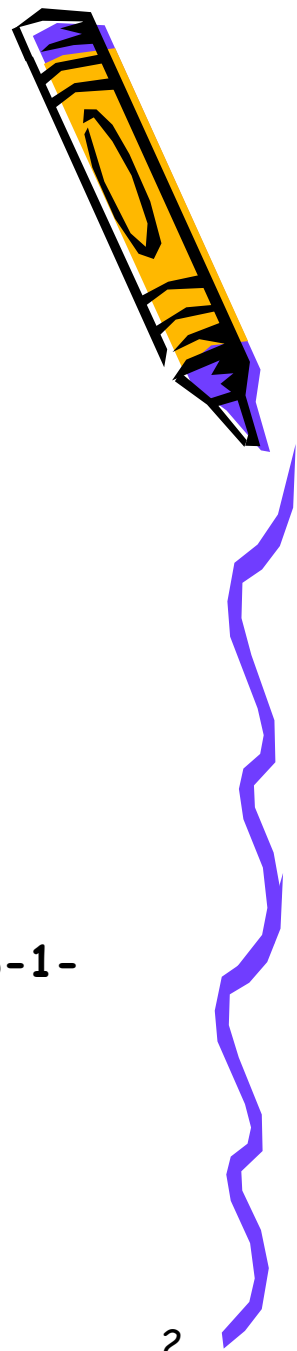
Lecture 4 JavaScript

By: A. Mousavi

SERG, School of Engineering Design, Brunel University, UK



Document Object Model (DOM)

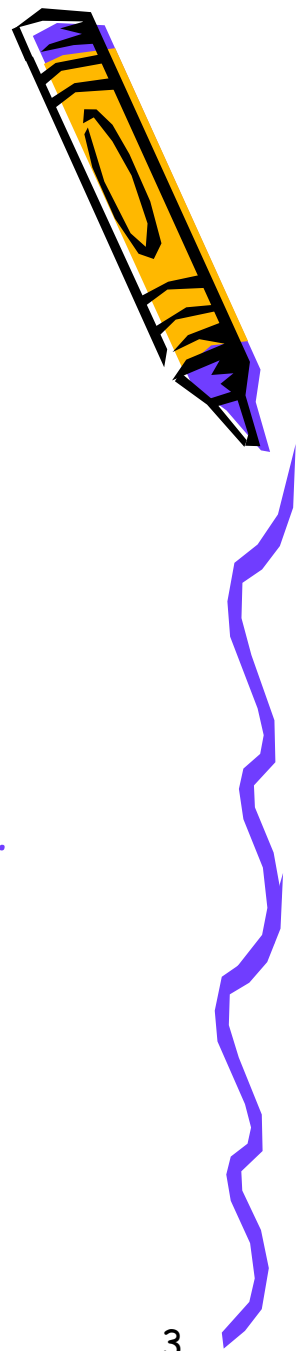


1. Nodes
2. Objects
3. DOM methods

Source: DOM Scripting, Web Design with JavaScript and the document object model, Jeremy Keith, Apress, 2005 - ISBN: 978-1-59059-533-6

also see examples in: <http://www.w3schools.com/>

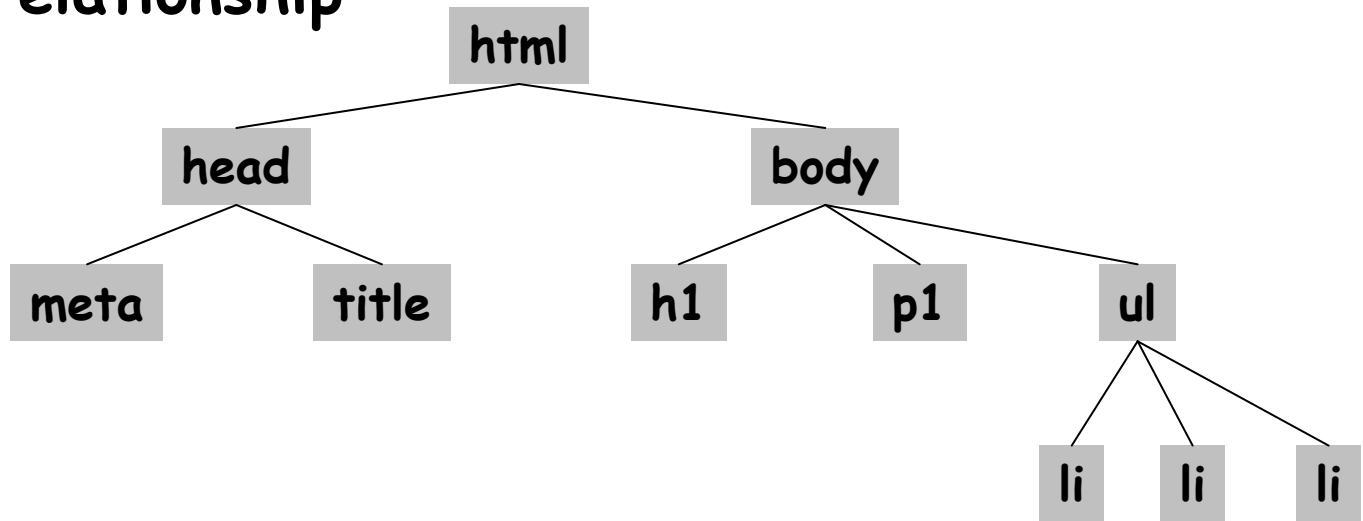
DOM



- DOM turns a **d**ocument into **o**bjects
- **O**bjects are self contained data items which have properties and method
 1. Host objects - provided by browsers e.g. window objects
 2. Native objects - built in JavaScript e.g. Math, Date...
 3. User defined - created by programmer
- **M**odel describes the relationship between different elements within the document

DOM Cont.

DOM interprets the elements of an html document as a tree of familial (*parent* and *sibling* relationship)

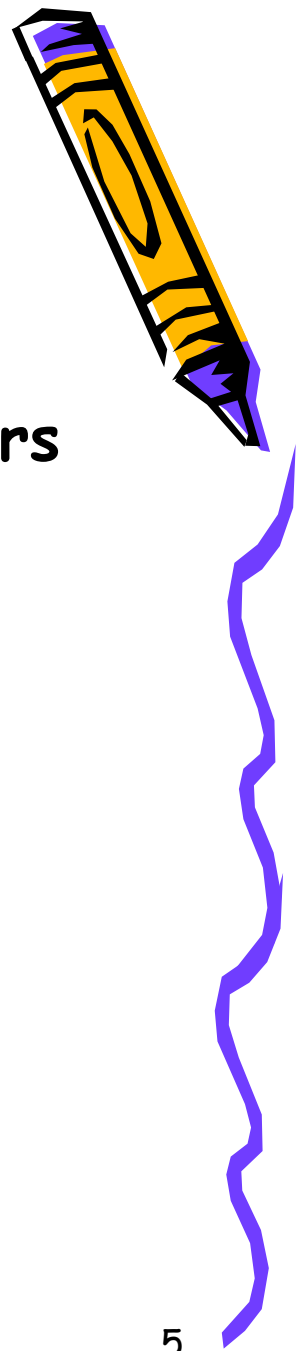


Source of figure: DOM Scripting, Web Design with JavaScript and the document object model, Jeremy Keith, Apress, 2005 - ISBN: 978-1-59059-533-6, p. 44

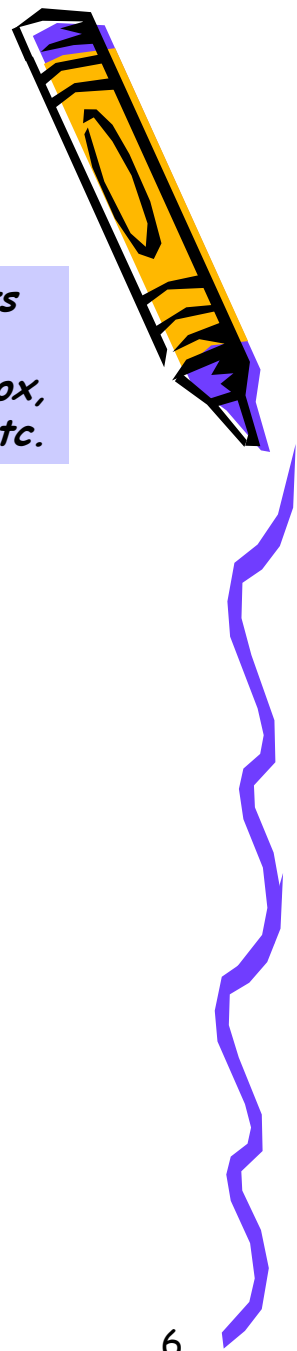
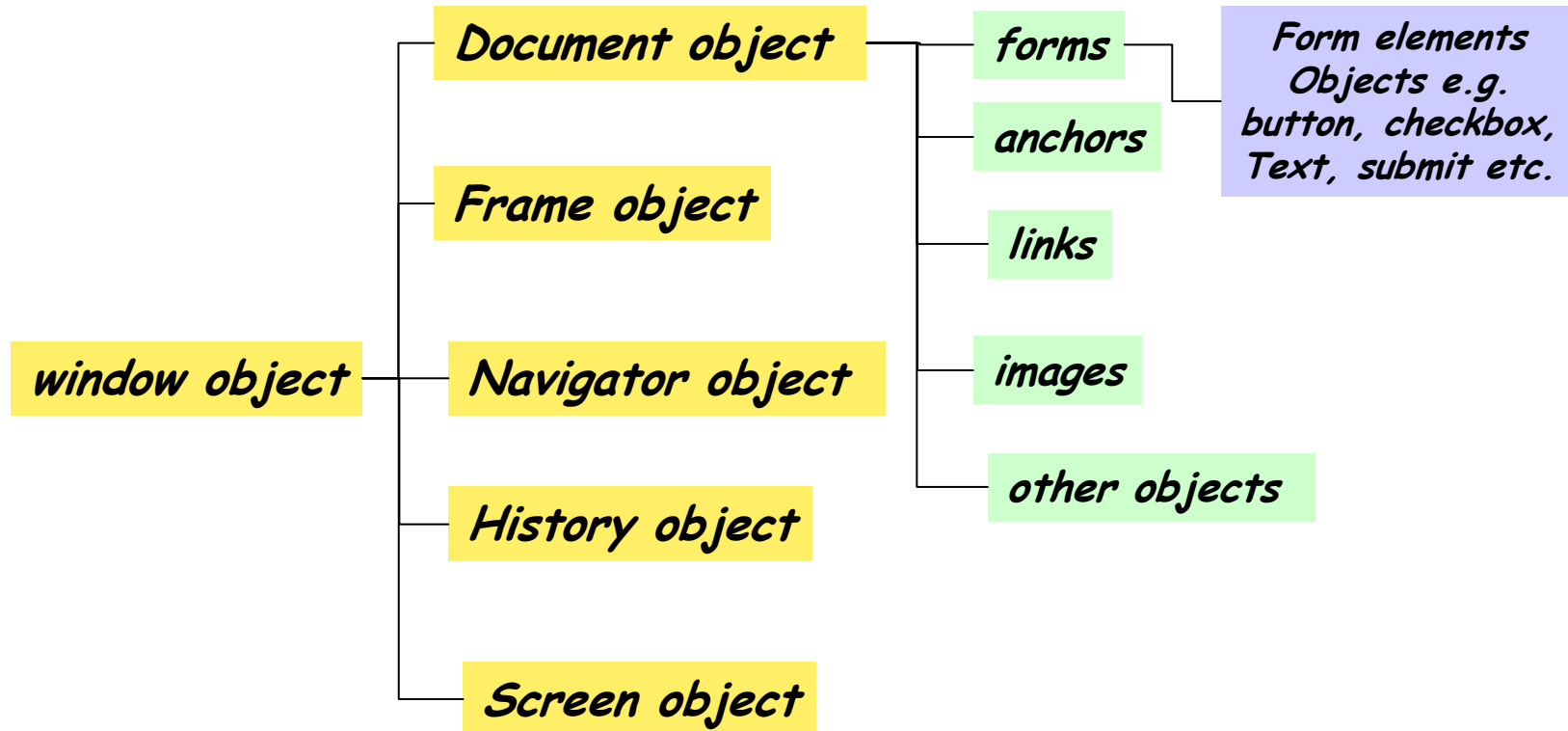
Browser Object Model

The window object contains:

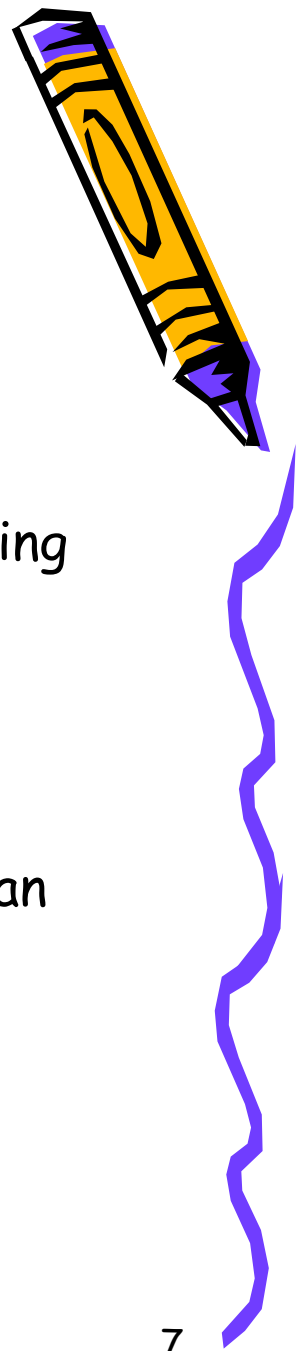
- Document object (contains form array, anchors array, images array, and links array)
- Frame array (array of windows)
- Navigator
- History
- Screen



Browser Object Model



Nodes



Consider nodes as the building blocks of a document

1. **element nodes:** elements are the building blocks of documents on the web such as `<body>`, `<p>` and ``.
Elements can contain other elements like `` containing ``
2. **text nodes:** contain the textual information of the documents. For example `<body> <p> text
abcdefg...</p></body>`
3. **attribute nodes:** provide further information about an element. For example: `<ul id = "List of Players">`
` Jack`
` John`
` Jill`

Quick Reminder JavaScript objects



- Objects are an important part of the client side
- Objects have:
 - *Properties: variables within an object*
 - *Methods: defined functions*
- **Built-in objects:** retrieve or give instructions to the browser

*Examples: **window, document, Date, and Math** are objects*

***window.alert()** // a method in the window object*

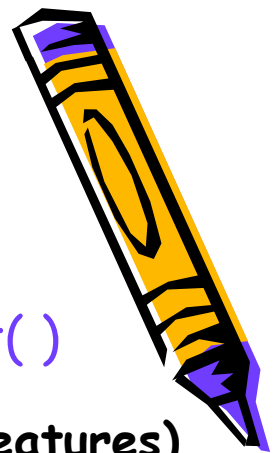
***document.write()** // method in document object*

***Math.sin()** or **Math.random** // methods in Math object*

***document.bgcolor** // a property of document object*

Note that methods are enclosed in () and properties are not.

Some useful object *methods* and *properties*



1. window object

- methods: `alert()`, `confirm()`, `open()`, `close()`, `prompt()`

For example:

MsgWin = open("Message.html", Message, 'width = 450, height= 600')

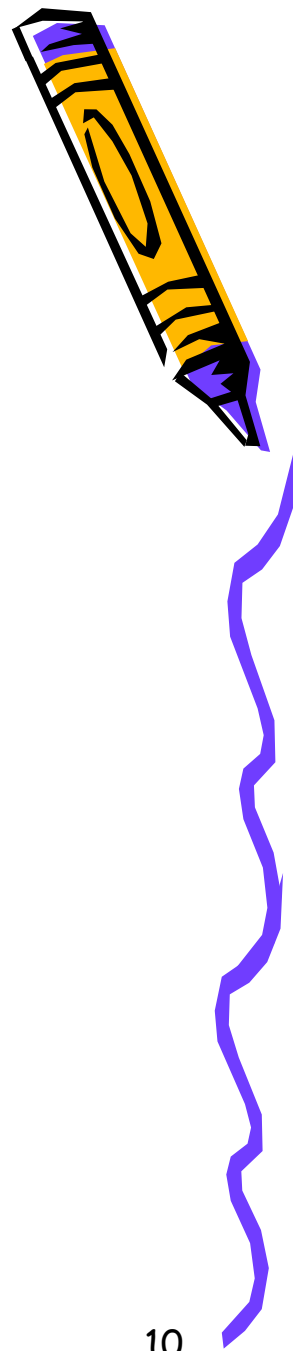
URL name Size (features)

The diagram shows three parameters of the `open()` method: "Message.html", "Message", and "width = 450, height= 600". Each parameter is enclosed in a dotted oval. Arrows point from the labels "URL", "name", and "Size (features)" to their respective ovals.

MsgWin.close() // to close the window object

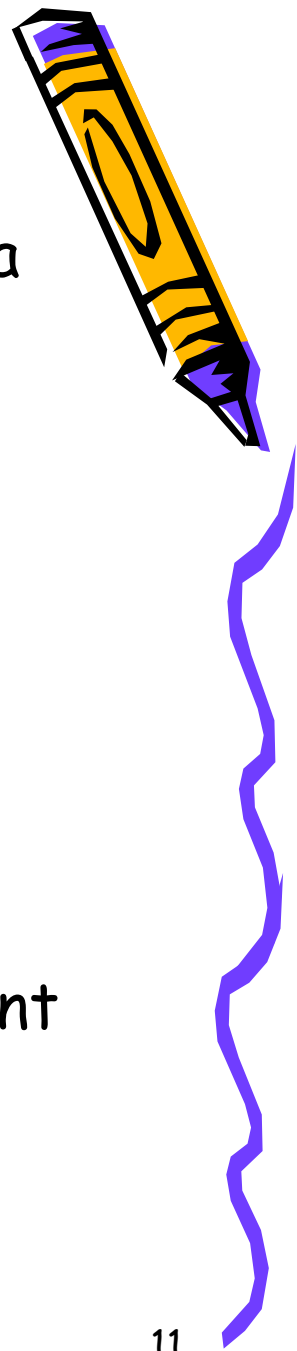
- Properties: `status`, `defaultstatus` // is shown in status bar for example "done" is the default but you can set any other text you wish

Exercise 4-1



Write a small application to open a message window with a set of features.

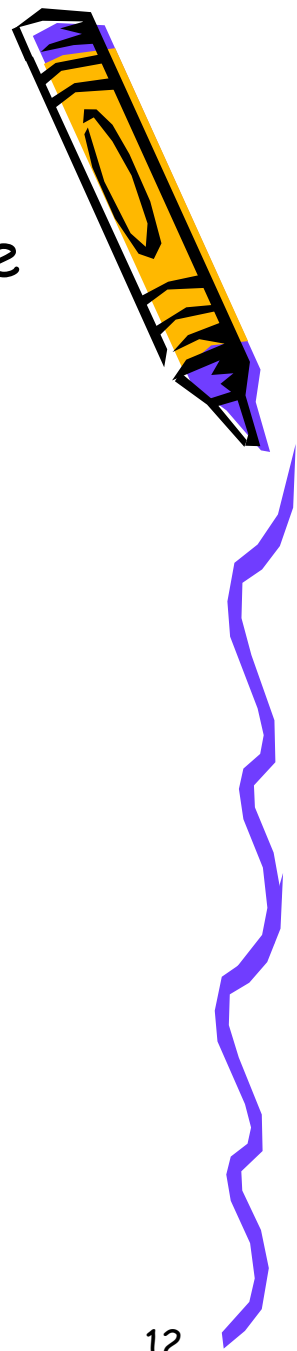
document object



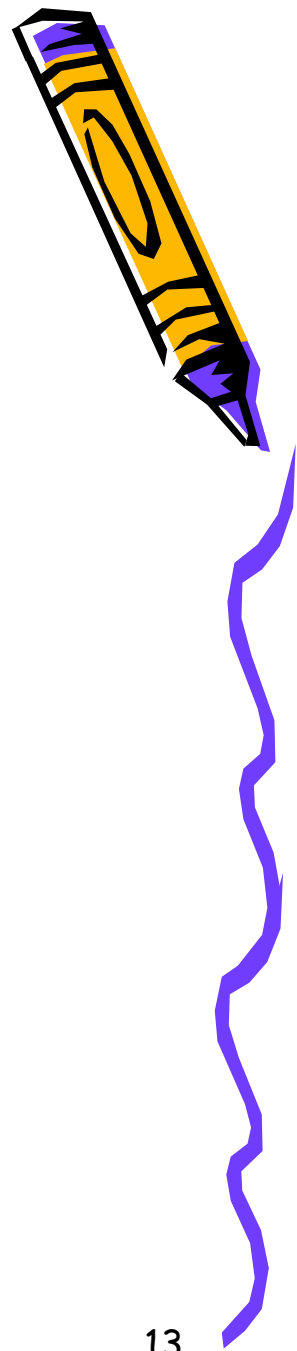
- **forms []**: refers to a number of forms within a document
- **anchors []**: refers to the document anchors
**
- **links []**: refers to a series of links in the document
* ...*
- **images []**: refers to the images in the document
use image tags for embedded images

history and location object

- **location object:** provides information about the location of the document
 - methods: `reload()` and `replace(url)`
 - properties: `hostname`, `href`, `port`, ...
- **history object:** provides browser history (security issues)
 - methods: `back()`, `forward()` and `go(index)`
 - properties: `next`, `previous`, ...



Exercise 4-2

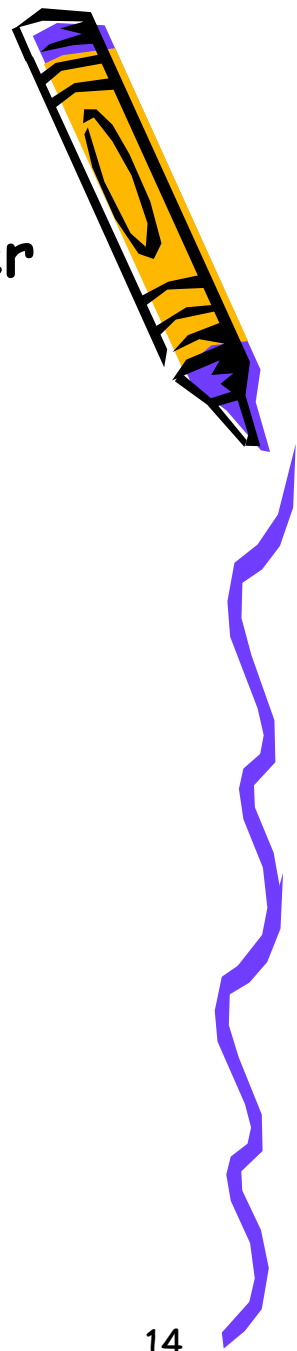


A short example demonstrating the *location* object

navigator object

- Mainly provides information about the browser such as type and version
 - methods: `javaEnabled()`, `plugin.refresh()`
 - properties: `appName`, `appVersion`, ...

Exercise 4-3 a navigator object.

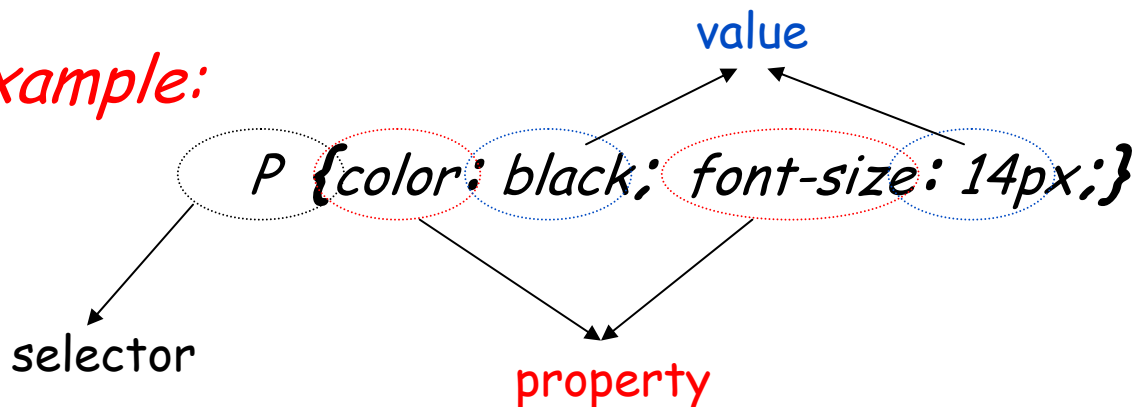


Cascading Style Sheets (CSS)

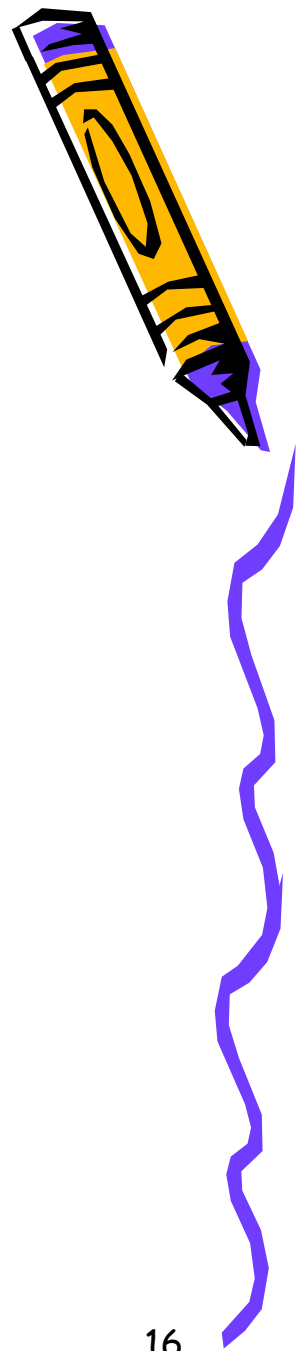


- CSS can be used to instruct browsers on how to display a document
- Styles can be declared either in the <head> by using the <style> tag in a document or as a separate (external) style sheet
- CSS has two parts - **selector** and **declarations**

Example:



Exercise 4-4 & 4-5



Simple examples of style sheets in action

CSS cont.



- At occasions when applying styles you may want to select a specific element in the document without changing other parts of the style - you can use two ways to do this:

1. class

```
<p class = "special"> A special class paragraph</p>
```

```
<h1 class = "special"> This headline also is special</h1>
```

/ the styles can now be applied to the relevant elements of this class*/*

```
.special {  
    font-style: italic;}
```

// and specifically take an element

```
h1.special{ text-transform: uppercase}
```

2. Id

```
<ul id = "team">
```

```
# team li {font-style: italic; }
```

DOM methods



- ***getElementById ()***: returns a reference to the first object with the specified ID

document.getElementById(id)

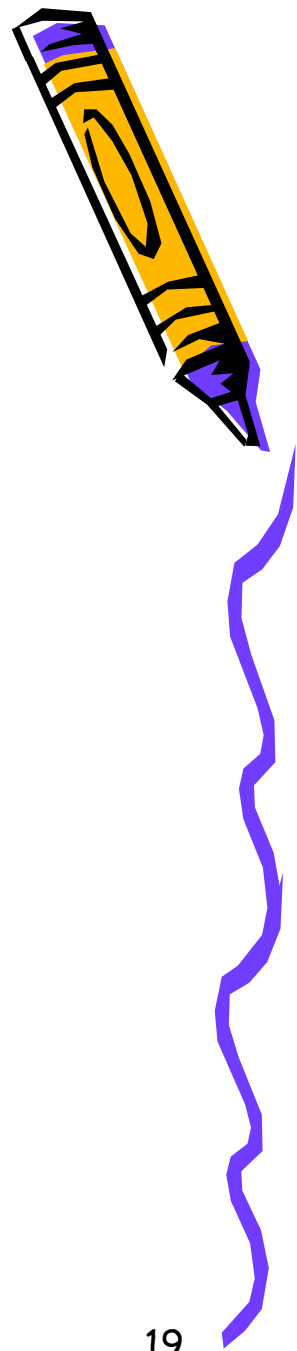
example: *document.getElementById("Team")*

- ***getElementsByTagName ()***: returns a collection of objects with the specified TagName

document.getElementsByTagName(tag)

Example: *document.getElementsByTagName("input")*

Exercise 4-6 & 4-7



**getElementById and getElementsByTagName
methods**

Quick reminder

- DOM : turns document into objects
- Interpretation of a document to Familial Tree
- Browser Object Models
- Cascading Style Sheets
- Methods & Properties

