Eman Fathi

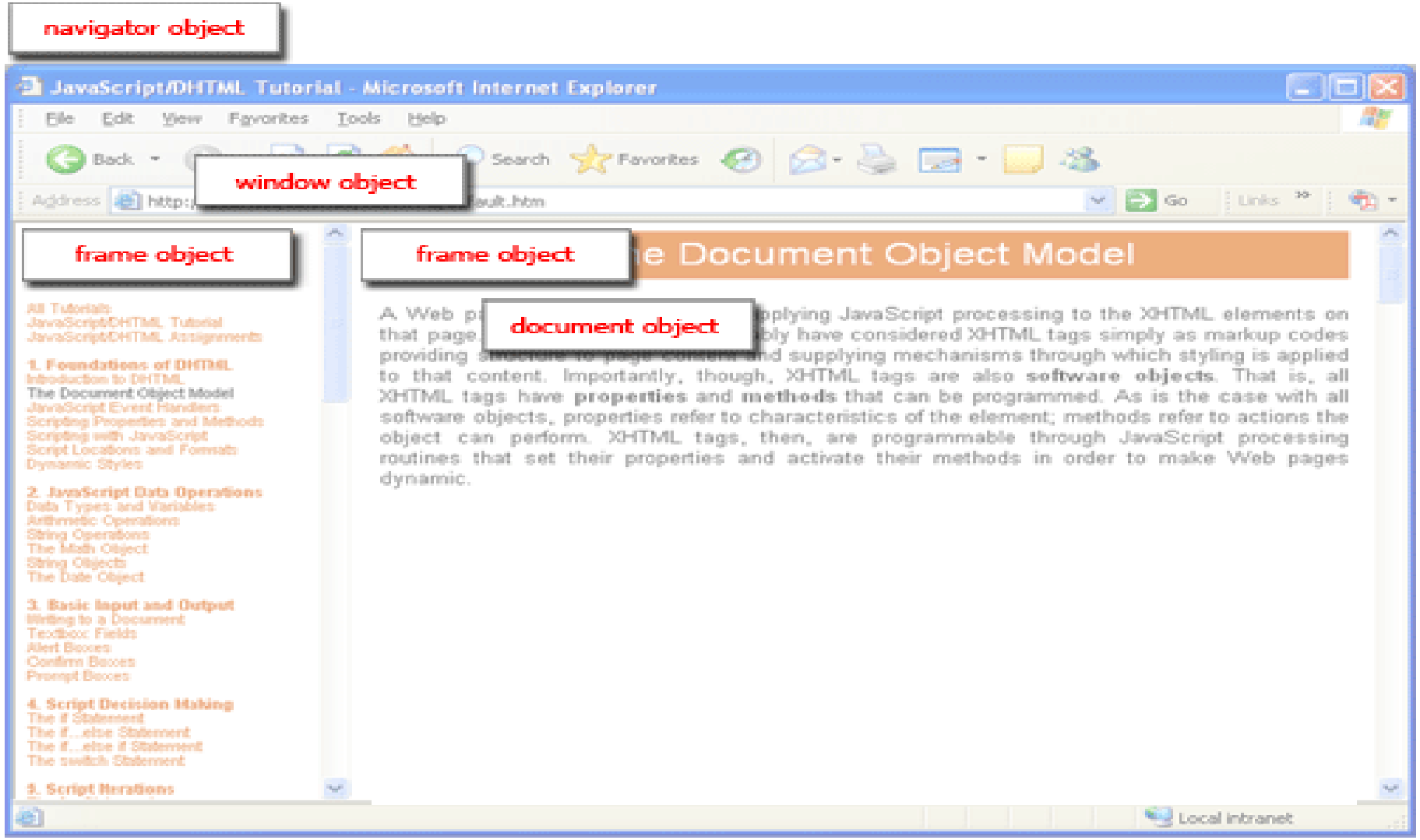
DOM

Document Object Model

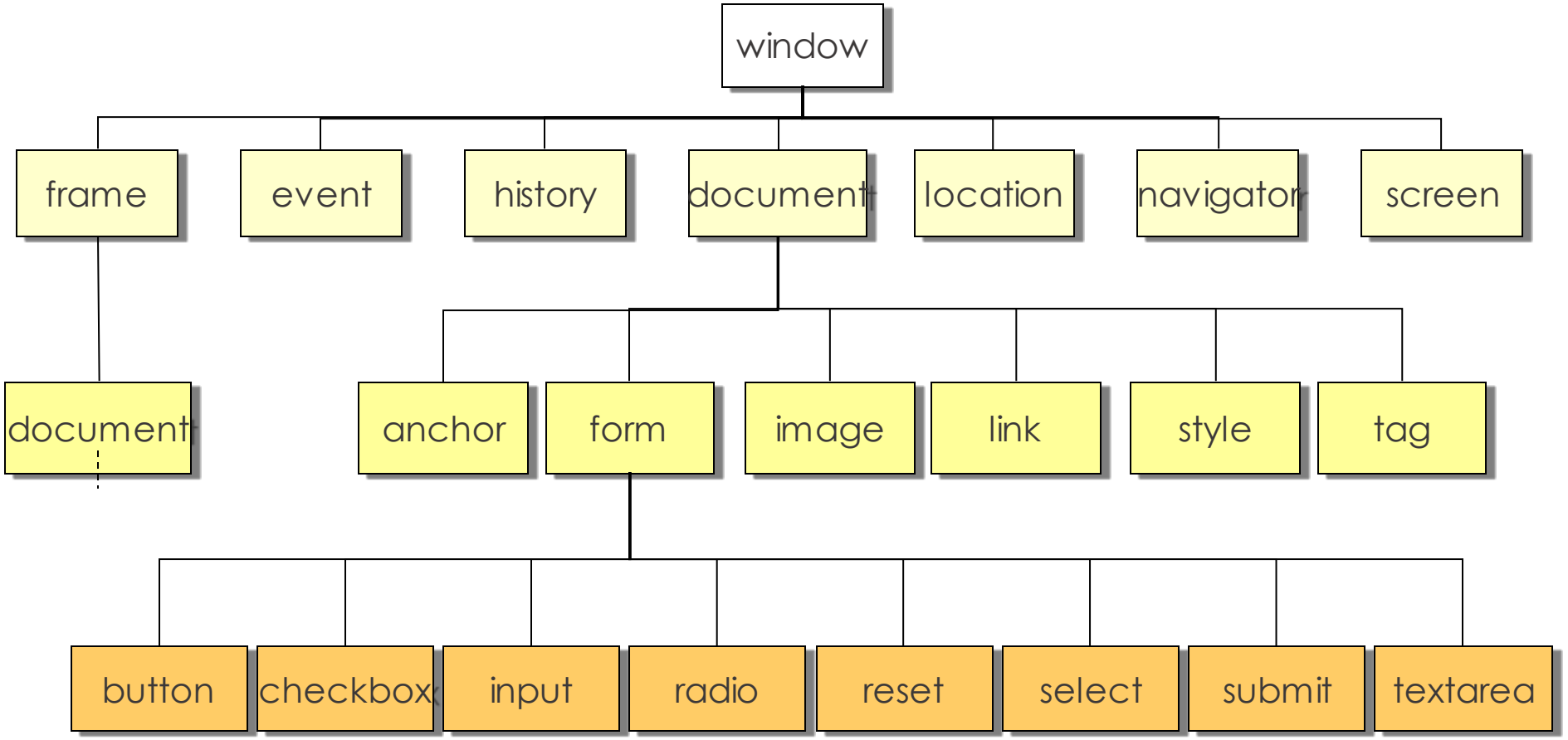
What is DOM?

- Document Object Model is the fundamental **API** for **representing** and **manipulating** the content of **HTML** and **XML** documents.
- W3C standard.
- Defines a standard way to access and manipulate HTML documents.
- Allows programmers generic access - **adding**, **deleting**, and **manipulating** - of all styles, **attributes**, and **elements** in a document.
- Platform independent.
- Language independent.

The DOM Model



DOM Tree


window
frame
event
history
document
location
navigator
screen
document
anchor
form
image
link
style
tag
button
checkbox
input
radio
reset
select
submit
textarea

JavaScript Objects Hierarchy

Every page has the following objects:

- **window**: the top-level object; has properties that apply to the entire window.
- **navigator**: has properties related to the name and version of the Navigator being used.
- **document**: contains properties based on the content of the document, such as title, background color, links, and forms.
- **location**: has properties based on the current URL.
- **history**: contains properties representing URLs the client has previously requested.

Window Object

Window

The Window object is the main entry point to all client-side JavaScript features and APIs. It represents a web browser window or frame, and you can refer to it with the identifier **window**.

```
 Set the location property to navigate to a new web page
window.location = "http:www.iti.gov.eg/";


location = "http:www.iti.gov.eg/";
optional use for window reference because it's global
```

Property	Description
Window.closed	This read-only property indicates whether the referenced window is closed or not.
Window.console	Read only Returns a reference to the console object which provides access to the browser's debugging console.
Window.document	Read only Returns a reference to the document that the window contains.
Window.frames	Read only Returns an array of the sub frames in the current window.
Window.fullScreen	Indicates whether the window is displayed in full screen or not.
Window.history	Read only Returns a reference to the history object.
Window.innerHeight	Gets the height of the content area of the browser window including, if rendered, the horizontal scrollbar.
Window.innerWidth	Gets the width of the content area of the browser window including, if rendered, the vertical scrollbar.
Window.length	Read only Returns the number of frames in the window.
Window.location	Read only Gets/sets the location, or current URL, of the window object.
Window.locationbar	Read only Returns the locationbar object, whose visibility can be toggled.
Window.**localStorage**	Read only Returns a reference to the local storage object used to store data that may only be accessed by the origin that created it.
Window.menubar	Read only Returns the menubar object, whose visibility can be toggled.

Property	Description
Window.name	//Gets/sets the name of the window.
Window.navigator	//Read only //Returns a reference to the navigator object.
Window.opener	//Returns a reference to the window that opened this current window.
Window.orientation	Read only Returns the orientation in degrees (in 90 degree increments) of the viewport relative to the device's natural orientation.
Window.outerHeight	Read only Gets the height of the outside of the browser window.
Window.outerWidth	Read only Gets the width of the outside of the browser window.
Window.pageXOffset	Read only An alias for window.scrollX.
Window.pageYOffset	Read only An alias for window.scrollY
Window.sessionStorage	Read only Returns a reference to the session storage object used to store data that may only be accessed by the origin that created it.
Window.parent	Read only Returns a reference to the parent of the current window or subframe.
Window.screen	Read only Returns a reference to the screen object.
Window.screenX	Read only Returns the horizontal distance of the left border of the user's browser from the left side of the screen.
Window.screenY	Read only Returns the vertical distance of the top border of the user's browser from the top side of the screen.

Property	Description
Window.scrollbars	Read only Returns the scrollbars object, whose visibility can be toggled in the window.
Window.scrollMaxX	Read only The maximum offset that the window can be scrolled to horizontally, that is the document width minus the viewport width.
Window.scrollMaxY	Read only The maximum offset that the window can be scrolled to vertically (i.e., the document height minus the viewport height).
Window.scrollX	Read only Returns the number of pixels that the document has already been scrolled horizontally.
Window.scrollY	Read only Returns the number of pixels that the document has already been scrolled vertically.
Window.self	Read only Returns an object reference to the window object itself.
Window.status	Gets/sets the text in the statusbar at the bottom of the browser.
Window.statusbar	Read only Returns the statusbar object, whose visibility can be toggled.
Window.toolbar	Read only Returns the toolbar object, whose visibility can be toggled.
Window.top	Read only Returns a reference to the topmost window in the window hierarchy. This property is read only.
Window.window	Read only Returns a reference to the current window

```
Window.alert();
//Displays an alert dialog.
Window.back()
//Moves back one in the window history.
Window.blur()
//Sets focus away from the window.
Window.close()
//Closes the current window.
Window.confirm()
//Displays a dialog with a message that the user needs to respond to.
Window.dispatchEvent()
//Used to trigger an event.
Window.dump()
//Writes a message to the console.
Window.find()
//Searches for a given string in a window.
Window.focus()
//Sets focus on the current window.
```

```
Window.forward()
//Moves the window one document forward in the history.
Window.home()
//Returns the browser to the home page.
Window.moveBy()
//Moves the current window by a specified amount.
Window.moveTo()
//Moves the window to the specified coordinates.
Window.open()
//Opens a new window.
Window.openDialog()
//Opens a new dialog window.
Window.print()
//Opens the Print Dialog to print the current document.
Window.prompt()
//Returns the text entered by the user in a prompt dialog.
```

```
Window.resizeBy()
//Resizes the current window by a certain amount.
Window.resizeTo()
//Dynamically resizes window.
Window.scroll()
//Scrolls the window to a particular place in the document.
Window.scrollBy()
//Scrolls the document in the window by the given amount.
Window.scrollByLines()
//Scrolls the document by the given number of lines.
Window.scrollTo()
//Scrolls to a particular set of coordinates in the document.
Window.setCursor()
//Changes the cursor for the current window
Window.showModalDialog()
//Displays a modal dialog.
Window.stop()
//This method stops window loading.
```

open

The open() method opens a new browser window.

Syntax

window.open(URL,name,specs,replace)

URL: Optional. Specifies the URL of the page to open.
If no URL is specified, a new window with about:blank is opened.

Name: Optional. Specifies the target attribute or the name of the window.
The following values are supported:

- _blank - URL is loaded into a new window. This is default
- _parent - URL is loaded into the parent frame
- _self - URL replaces the current page
- _top - URL replaces any framesets that may be loaded
- name - The name of the window

open

Specs: Optional. A comma-separated list of items, no whitespaces.
The following values are supported:

channelmode yes|no|1|0	Whether or not to display the window in theater mode. Default is no. IE only
directories yes|no|1|0	Obsolete. Whether or not to add directory buttons. Default is yes. IE only
fullscreen yes|no|1|0	Whether or not to display the browser in full-screen mode. Default is no. A window in full-screen mode must also be in theater mode. IE only
height=pixels	The height of the window. Min. value is 100
left=pixels	The left position of the window. Negative values not allowed
location yes|no|1|0	Whether or not to display the address field. Opera only
menubar yes|no|1|0	Whether or not to display the menu bar

open

Specs: Optional. A comma-separated list of items, no whitespaces.
The following values are supported:

resizable yes|no|1|0	Whether or not the window is resizable. IE only
scrollbars yes|no|1|0	Whether or not to display scroll bars. IE, Firefox & Opera only
status yes|no|1|0	Whether or not to add a status bar
titlebar yes|no|1|0	Whether or not to display the title bar. Ignored unless the calling application is an HTML Application or a trusted dialog box
toolbar yes|no|1|0	Whether or not to display the browser toolbar. IE and Firefox only
top=pixels	The top position of the window. Negative values not allowed
width=pixels	The width of the window. Min. value is 100

open

replace: Optional. Specifies whether the URL creates a new entry or replaces the current entry in the history list. The following values are supported:

- true - URL replaces the current document in the history list
- false - URL creates a new entry in the history list

Return Value: A reference to the newly created window, or null if the call failed.

```
var myWindow = window.open("", "", "width=200,height=100");
```

```
var myWindow = window.open("", "MsgWindow", "width=200,height=100");
myWindow.document.write("<p>This is 'MsgWindow'. I am 200px wide and 100px tall!</p>");
```

open

```
window.open("http:www.google.com", "_blank",
"toolbar=yes,scrollbars=yes,resizable=yes,top=500,left=500,width=400,height=400");

var mywindow = window.open("page2.html", "_blank");
mywindow.opener;     returns a reference to the window that created the window.
mywindow.close();   close the opened window
This method is often used together with the open() method.
mywindow.closed;    Returns a Boolean value indicating whether a window has been closed or not
```

Dialog Boxes

The Window object provides three methods for displaying simple dialog boxes to the user.

- alert
- confirm
- prompt

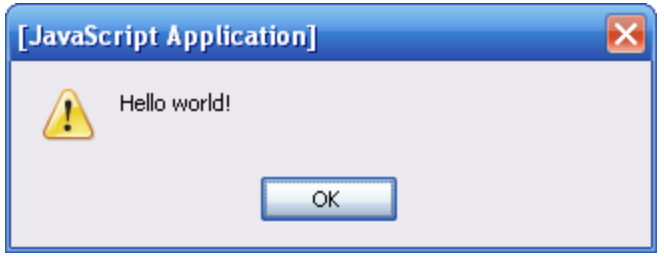
alert()

The **Window.alert()** method displays an alert dialog with the optional specified content and an OK button.

Syntax

```
window.alert(message);
```

message is an optional string of text you want to display in the alert dialog, or, alternatively, an object that is converted into a string and displayed.



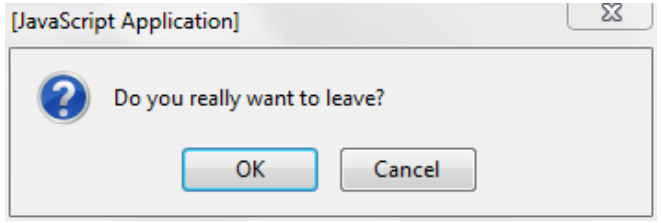
confirm()

The **Window.confirm()** method displays a modal dialog with an optional message and two buttons, **OK** and **Cancel**.

Syntax

result = window.confirm(*message*);

- **message** is the optional string to be displayed in the dialog.
- **result** is a boolean value indicating whether OK or Cancel was selected (true means OK).



JS

prompt()

The **Window.prompt()** displays a dialog with an optional message prompting the user to input some text.

Syntax

result = window.prompt(*message*, *default*);

- **result** is a string containing the text entered by the user, or null.
- **message** is a string of text to display to the user. This parameter is optional and can be omitted if there is nothing to show in the prompt window.
- **default** is a string containing the default value displayed in the text input field. It is an optional parameter.

Timers

Timers

setTimeout() and setInterval() allow you to register a function to be invoked once or repeatedly after a specified amount of time has elapsed.

These are important global functions of client-side JavaScript, and are therefore defined as methods of Window

setTimeout()

Schedules a function to run after a specified number of milliseconds elapses.

setTimeout() returns a value that can be passed to clearTimeout() to cancel the execution of the scheduled function.

Syntax:

```
var timeoutID = window.setTimeout(func, [delay, param1, param2, ...]);
var timeoutID = window.setTimeout(code, [delay]);
```

- **timeoutID** is the *numerical* ID of the timeout, which can be used later with clearTimeout().
- **func** is the function you want to execute after delay milliseconds.
- **delay** is the number of milliseconds that the function call should be delayed by. If omitted, it defaults to 0.
- **param1**, **param2**, and so forth are additional parameters which are passed through to the function specified by func.

Avoid passing string literals

```
 Recommended
window.setTimeout(function() {
    alert("Hello World!");
}, 500);

 Not recommended
window.setTimeout("alert('Hello World!');", 500);
```

setTimeout(), clearTimeout()

```
var timeoutID;    Note: Variable is visible for two functions

function delayedAlert() {
    timeoutID = window.setTimeout(slowAlert, 2000);
}

function slowAlert() {
    alert("That was really slow!");
}

function clearAlert() {
    window.clearTimeout(timeoutID);
}
```

setInterval()

setInterval() is like setTimeout() except that the specified function is invoked repeatedly at intervals of the specified number of milliseconds

Syntax

```
var intervalID = window.setInterval(func, delay[, param1, param2, ...]);
var intervalID = window.setInterval(code, delay);
```

- **intervalID** is a unique interval ID you can pass to clearInterval().
- **func** is the function you want to be called repeatedly.
- **delay** is the number of milliseconds that the setInterval() function should wait before each call to func.
- **param1**, **param2**, and so forth are additional parameters which are passed through to the function specified by func.

setInterval(), clearInterval()

```
var IntervalID;

IntervalID = setInterval(updateClock, 60000);  Call updateClock() every 60 seconds

function updateClock(){
    console.log(Date.now.toLocaleString());
}

function stopInterval(){
    clearInterval(IntervalID);
}
```

BOM

Browser Object Model

Location

Location

The location property of the Window object refers to a Location object, which represents the current URL of the document displayed in the window, and which also defines methods for making the window load a new document.

```
window.location.href returns the href (URL) of the current page
window.location.hostname returns the domain name of the web host
window.location.pathname returns the path and filename of the current page
window.location.protocol returns the web protocol used (http: or https:)
window.location.hash returns the part of the URL from the #
location.search return the query string part starts with ?
window.location.assign loads a new document
window.location.replace replaces the current document with a new one.
location.reload(true); Force reloading the current page from the server
```

Get Query String key value pairs

```
location.search return the query string part starts with ?
var args = {};  Start with an empty object
var query = location.search.substring(1);  Get query string, minus '?'
var pairs = query.split("&");  Split at ampersands
for(var i = 0; i < pairs.length; i++) {  For each fragment
    var pos = pairs[i].indexOf('=');  Look for "name=value"
    if (pos == -1) continue;  If not found, skip it
    var name = pairs[i].substring(0,pos);  Extract the name
    var value = pairs[i].substring(pos+1);  Extract the value
    value = decodeURIComponent(value);  Decode the value
    args[name] = value;  Store as a property
}
```

History

History

The History object models the browsing history of a window as a list of documents and document states.

```
history.back() same as clicking back in the browser
history.forward() same as clicking forward in the browser
history.go(-2);  Go back 2, like clicking the Back button twice
history.length returns the number of URLs in the history list of the current browser window
```

Navigator

Navigator

Navigator object contains browser vendor and version number information.

```
navigator.appName        The full name of the web browser.

navigator.appVersion     string that contains browser vendor and version information.

navigator.platform       A string that identifies the operating system (and possibly the
                 hardware) on which the browser is running.

navigator.onLine         Specifies whether the browser is currently connected to the network.

navigator.javaEnabled()  Return true if the browser can run Java applets.

navigator.cookieEnabled()Return true if the browser can store persistent cookies.
```

Screen

Screen

provides information about the size of the user's display and the number of colors available on it.

`screen.width`	returns the width of the visitor's screen in pixels.
`screen.height`	returns the height of the visitor's screen in pixels.
`screen.availWidth`	returns the width of the visitor's screen, in pixels, minus interface features like the Windows Taskbar.
`screen.availHeight`	returns the height of the visitor's screen, in pixels, minus interface features like the Windows Taskbar.
`screen.colorDepth`	returns the number of bits used to display one color.
`screen.pixelDepth`	returns the pixel depth of the screen.

Thank You