

n Number() = 42

PROPERTIES

- n**.POSITIVE_INFINITY +∞ equivalent
- n**.NEGATIVE_INFINITY -∞ equivalent
- n**.MAX_VALUE largest positive value
- n**.MIN_VALUE smallest positive value
- n**.EPSILON diff between 1 & smallest >1
- n**.NaN not-a-number value

METHODS

- s**.toExponential(**dec**) exp. notation
- s**.toFixed(**dec**) fixed-point notation
- s**.toPrecision(**p**) change precision
- b**.isFinite(**n**) check if number is finite
- b**.isInteger(**n**) check if number is int.
- b**.isNaN(**n**) check if number is NaN
- n**.parseInt(**s**, **radix**) string to integer
- n**.parseFloat(**s**, **radix**) string to float

r Regexp() = /.+/ig

PROPERTIES

- n**.lastIndex index to start global regexp
- s**.flags active flags of current regexp
- b**.global flag g (search all matches)
- b**.ignoreCase flag i (match lower/upper)
- b**.multiline flag m (match multiple lines)
- b**.sticky flag y (search from lastIndex)
- b**.unicode flag u (enable unicode feat.)
- s**.source current regexp (w/o slashes)

METHODS

- a**.exec(**str**) exec search for a match
- b**.test(**str**) check if regexp match w/str

CLASSES

- . any character \t tabulator
- \d digit [0-9] \r carriage return
- \D no digit [^0-9] \n line feed
- \w any alphanumeric char [A-Za-z0-9_]
- \W no alphanumeric char [^A-Za-z0-9_]
- \s any space char (space, tab, enter...)
- \S no space char (space, tab, enter...)
- \xN char with code N [b] backspace
- \uN char with unicode N \0 NUL char

CHARACTER SETS OR ALTERNATION

- [abc] match any character set
- ^[abc] match any char. set not enclosed
- a|b match a or b

BOUNDARIES

- ^ begin of input \$ end of input
- \b zero-width word boundary
- \B zero-width non-word boundary

GROUPING

- (x) capture group (?x) no capture group
- \n reference to group n captured

QUANTIFIERS

- x* preceding x 0 or more times {0,}
- x+ preceding x 1 or more times {1,}
- x? preceding x 0 or 1 times {0,1}
- x{n} n occurrences of x
- x{n,} at least n occurrences of x
- x{n,m} between n & m occurrences of x

ASSERTIONS

- x(=?y) x (only if x is followed by y)
- x(?!y) x (only if x is not followed by y)

s String() = 'text'

PROPERTIES

- n**.length string size

METHODS

- s**.charAt(**index**) char at position [i]
- n**.charCodeAt(**index**) unicode at pos.
- s**.fromCharCode(**n1**, **n2**...) code to char
- s**.concat(**str1**, **str2**...) combine text +
- b**.startsWith(**str**, **size**) check beginning
- b**.endsWith(**str**, **size**) check ending
- b**.includes(**str**, **from**) include substring?
- n**.indexOf(**str**, **from**) find substr index
- n**.lastIndexOf(**str**, **from**) find from end
- n**.search(**regex**) search & return index
- n**.localeCompare(**str**, **locale**, **options**)
- a**.match(**regex**) matches against string
- s**.repeat(**n**) repeat string n times
- s**.replace(**str**|**regex**, **newstr**|**func**)
- s**.slice(**ini**, **end**) str between ini/end
- s**.substr(**ini**, **len**) substr of len length
- s**.substring(**ini**, **end**) substr fragment
- a**.split(**sepl**|**regex**, **limit**) divide string
- s**.toLowerCase() string to lowercase
- s**.toUpperCase() string to uppercase
- s**.trim() remove space from begin/end
- s**.raw`` template strings with \${vars}

d Date()

METHODS

- n**.UTC(**y**, **m**, **d**, **h**, **i**, **s**, **ms**) timestamp
- n**.now() timestamp of current time
- n**.parse(**str**) convert str to timestamp
- n**.setTime(**ts**) set UNIX timestamp
- n**.getTime() return UNIX timestamp

UNIT SETTERS (ALSO .setUTC*() methods)

- n**.setFullYear(**y**, **m**, **d**) set year (yyyy)
- n**.setMonth(**m**, **d**) set month (0-11)
- n**.setDate(**d**) set day (1-31)
- n**.setHours(**h**, **m**, **s**, **ms**) set hour (0-23)
- n**.setMinutes(**m**, **s**, **ms**) set min (0-59)
- n**.setSeconds(**s**, **ms**) set sec (0-59)
- n**.setMilliseconds(**ms**) set ms (0-999)

UNIT GETTERS (ALSO .getUTC*() methods)

- n**.getDate() return day (1-31)
- n**.getDay() return day of week (0-6)
- n**.getMonth() return month (0-11)
- n**.getFullYear() return year (yyyy)
- n**.getHours() return hour (0-23)
- n**.getMinutes() return minutes (0-59)
- n**.getSeconds() return seconds (0-59)
- n**.getMilliseconds() return ms (0-999)

LOCALE & TIMEZONE METHODS

- n**.getTimezoneOffset() offset in mins
- s**.toLocaleDateString(**locale**, **options**)
- s**.toLocaleTimeString(**locale**, **options**)
- s**.toLocaleString(**locale**, **options**)
- s**.toUTCString() return UTC date
- s**.toDate() return American date
- s**.toISOString() return ISO8601 date
- s**.toJSON() return date ready for JSON

a Array() = [1, 2, 3]

PROPERTIES

- n**.length number of elements

METHODS

- b**.isArray(**obj**) check if obj is array
- b**.includes(**obj**, **from**) include element?
- n**.indexOf(**obj**, **from**) find elem. index
- n**.lastIndexOf(**obj**, **from**) find from end
- s**.join(**sep**) join elements w/separator
- a**.slice(**ini**, **end**) return array portion
- a**.concat(**obj1**, **obj2**...) return joined array

MODIFY SOURCE ARRAY METHODS

- a**.copyWithin(**pos**, **ini**, **end**) copy elems
- a**.fill(**obj**, **ini**, **end**) fill array with obj
- a**.reverse() reverse array & return it
- a**.sort(**cf(a,b)**) sort array (unicode sort)
- a**.splice(**ini**, **del**, **o1**, **o2**...) del&add elem

ITERATION METHODS

- a**.entries() iterate key/value pair array
- a**.keys() iterate only keys array
- a**.values() iterate only values array

CALLBACK FOR EACH METHODS

- b**.every(**cb(e,i,a)**, **arg**) test until false
- b**.some(**cb(e,i,a)**, **arg**) test until true
- a**.map(**cb(e,i,a)**, **arg**) make array
- a**.filter(**cb(e,i,a)**, **arg**) make array w/true
- o**.find(**cb(e,i,a)**, **arg**) return elem w/true
- n**.findIndex(**cb(e,i,a)**, **arg**) return index
- o**.forEach(**cb(e,i,a)**, **arg**) exec for each
- o**.reduce(**cb(p,e,i,a)**, **arg**) accumulative
- o**.reduceRight(**cb(p,e,i,a)**, **arg**) from end

ADD/REMOVE METHODS

- o**.pop() remove & return last element
- n**.push(**o1**, **o2**...) add element & return length
- o**.shift() remove & return first element
- n**.unshift(**o1**, **o2**...) add element & return len

b Boolean() = true / false

no own properties or methods

f Function() = function(a, b) { ... }

PROPERTIES

- o**.length return number of arguments
- s**.name return name of function
- o**.prototype prototype object

METHODS

- o**.call(**newthis**, **arg1**, **arg2**...) change this
- o**.apply(**newthis**, **arg1**) with args array
- o**.bind(**newthis**, **arg1**, **arg2**...) bound func

n number**NaN** (not-a-number)**s** string**b** boolean (true/false)**a** array**d** date**r** regular expression**f** function**o** object**u** undefined

only available on ECMAScript 6

n static (ex: Math.random())**n** non-static (ex: new Date().getDate())**argument** required**argument** optional

Emezeta.com

Math

PROPERTIES

- n**.E Euler's constant
- n**.LN2 natural logarithm of 2
- n**.LN10 natural logarithm of 10
- n**.LOG2E base 2 logarithm of E
- n**.LOG10E base 10 logarithm of E
- n**.PI ratio circumference/diameter
- n**.SQRT1_2 square root of 1/2
- n**.SQRT2 square root of 2

METHODS

- n**.abs(**x**) absolute value
- n**.cbrt(**x**) cube root
- n**.clz32(**x**) return leading zero bits (32)
- n**.exp(**x**) return e^x
- n**.expm1(**x**) return $e^x - 1$
- n**.hypot(**x1**, **x2**...) length of hypotenuse
- n**.imul(**a**, **b**) signed multiply
- n**.log(**x**) natural logarithm (base e)
- n**.log1p(**x**) natural logarithm (1+x)
- n**.log10(**x**) base 10 logarithm
- n**.log2(**x**) base 2 logarithm
- n**.max(**x1**, **x2**...) return max number
- n**.min(**x1**, **x2**...) return min number
- n**.pow(**base**, **exp**) return $base^{exp}$
- n**.random() float random number [0,1)
- n**.sign(**x**) return sign of number
- n**.sqrt(**x**) square root of number

ROUND METHODS

- n**.ceil(**x**) superior round (smallest)
- n**.floor(**x**) inferior round (largest)
- n**.fround(**x**) nearest single precision
- n**.round(**x**) round (nearest integer)
- n**.trunc(**x**) remove fractional digits

TRIGONOMETRIC METHODS

- n**.acos(**x**) arccosine
- n**.acosh(**x**) hyperbolic arccosine
- n**.asin(**x**) arcsine
- n**.asinh(**x**) hyperbolic arcsine
- n**.atan(**x**) arctangent
- n**.atan2(**x**, **y**) arctangent of quotient x/y
- n**.atanh(**x**) hyperbolic arctangent
- n**.cos(**x**) cosine
- n**.cosh(**x**) hyperbolic cosine
- n**.sin(**x**) sine
- n**.sinh(**x**) hyperbolic sine
- n**.tan(**x**) tangent
- n**.tanh(**x**) hyperbolic tangent

JSON

METHODS

- n**.parse(**str**, **tf**(**k**,**v**)) parse string to object
- n**.stringify(**obj**, **repf**[**wl**, **sp**]) convert to str

Error()

PROPERTIES

- s**.name return name of error
- s**.message return description of error

EvalError(), InternalError(), RangeError(), URIError(), ReferenceError(), SyntaxError(), TypeError()

Object() = {key: value, key2: value2}

PROPERTIES

- o**.constructor return ref. to object func.

METHODS

- o**.assign(**dst**, **src1**, **src2**...) copy values
- o**.create(**proto**, **prop**) create obj w/prop
- o**.defineProperties(**obj**, **prop**)
- o**.defineProperty(**obj**, **prop**, **desc**)
- o**.freeze(**obj**) avoid properties changes
- o**.getOwnPropertyDescriptor(**obj**, **prop**)
- a**.getOwnPropertyNames(**obj**)
- a**.getOwnPropertySymbols(**obj**)
- o**.getPrototypeOf(**obj**) return prototype
- b**.is(**val1**, **val2**) check if are same value
- b**.isExtensible(**obj**) check if can add prop
- b**.isFrozen(**obj**) check if obj is frozen
- b**.isSealed(**obj**) check if obj is sealed
- a**.keys(**obj**) return only keys of object
- o**.preventExtensions(**obj**) avoid extend
- o**.seal(**obj**) prop are non-configurable
- o**.setPrototypeOf(**obj**, **prot**) change prot

INSTANCE METHODS

- b**.hasOwnProperty(**prop**) check if exist
- b**.isPrototypeOf(**obj**) test in another obj
- b**.propertyIsEnumerable(**prop**)
- s**.toString() return equivalent string
- s**.toLocaleString() return locale version
- o**.valueOf() return primitive value

Promise()

METHODS

- p**.all(**obj**) return promise
- p**.catch(**onRejected**(**s**)) = .then(**undef**,**s**)
- p**.then(**onFulfilled**(**v**), **onRejected**(**s**))
- p**.race(**obj**) return greedy promise (res/rej)
- p**.resolve(**obj**) return resolved promise
- p**.reject(**reason**) return rejected promise

Proxy() Reflect same methods (not func)

METHODS

- o**.apply(**obj**, **arg**, **arglist**) trap function call
- o**.construct(**obj**, **arglist**) trap new oper
- o**.defineProperty(**obj**, **prop**, **desc**)
- o**.deleteProperty(**obj**, **prop**) trap delete
- o**.enumerate(**obj**) trap for...in
- o**.get(**obj**, **prop**, **rec**) trap get property
- o**.getOwnPropertyDescriptor(**obj**, **prop**)
- o**.getPrototypeOf(**obj**)
- o**.has(**obj**, **prop**) trap in operator
- o**.ownKeys(**obj**)
- o**.preventExtensions(**obj**)
- o**.set(**obj**, **prop**, **value**) trap set property
- o**.setPrototypeOf(**obj**, **proto**)

globals

METHODS

- o** eval(**str**) evaluate javascript code
- b** isFinite(**obj**) check if is a finite number
- b** isNaN(**obj**) check if is not a number
- n** parseInt(**s**, **radix**) string to integer
- n** parseFloat(**s**, **radix**) string to float
- s** encodeURIComponent(**URI**) = to %3D
- s** decodeURIComponent(**URI**) %3D to =

Set()

WeakSet only obj as items

PROPERTIES

- n**.size return number of items

METHODS

- s**.add(**item**) add item to set **ws**
- b**.has(**item**) check if item exists **ws**
- b**.delete(**item**) del item & return if del **ws**
- o**.clear() remove all items from set

ITERATION METHODS

- si**.entries() iterate items
- si**.values() iterate only value of items

CALLBACK FOR EACH METHODS

- o**.forEach(**cb**(**e**,**i**,**a**), **arg**) exec for each

Map()

WeakMap only obj as keys

PROPERTIES

- n**.size return number of elements

METHODS

- m**.set(**key**, **value**) add pair key=value **wm**
- o**.get(**key**) return value of key **wm**
- b**.has(**key**) check if key exist **wm**
- b**.delete(**key**) del elem. & return if ok **wm**
- o**.clear() remove all elements from map

ITERATION METHODS

- m**.entries() iterate elements
- m**.keys() iterate only keys
- m**.values() iterate only values

CALLBACK FOR EACH METHODS

- o**.forEach(**cb**(**e**,**i**,**a**), **arg**) exec for each

Symbol()

PROPERTIES

- s**.iterator specifies default iterator
- s**.match specifies match of regexp
- s**.species specifies constructor function

METHODS

- s**.for(**key**) search existing symbols
- s**.keyFor(**sym**) return key from global reg

Generator() = function* () { ... }

METHODS

- o**.next(**value**) return obj w/{value,done}
- o**.return(**value**) return value & true done
- o**.throw(**except**) throw an error

Others

FAST TIPS

- var declare variable
- let declare block scope local variable
- const declare constant (read-only)
- func(**a**=1) default parameter value
- func(...**a**) rest argument (spread operator)
- (**a**) => { ... } function equivalent (fat arrow)
- `string \${a}` template with variables
- 0**n** binary (2) number **n** to decimal
- 0**o**n octal (8) number **n** to decimal
- 0**x**n hexadecimal (16) number **n** to decimal
- for (**i** in **array**) { ... } iterate array, **i** = index
- for (**e** of **array**) { ... } iterate array, **e** = value
- class **B** extends **A** { } class sugar syntax

window = Browser global object

PROPERTIES

b.closed check if window is closed
n.devicePixelRatio ratio vertical size pix
b.fullScreen check if window is fullscreen
n.innerWidth width size (incl. scrollbar)
n.innerHeight height size (incl. scrollbar)
n.outerWidth width size (incl. browser)
n.outerHeight height size (incl. browser)
n.length number of frames
s.name inner name of window
s.status bottom statusbar text

API/OBJECTS PROPERTIES

o.applicationCache offline resources API
o.console console browser API
o.crypto cryptographic API
o.history session page history API
o.location information about URL API
o.localStorage storage for site domain
o.sessionStorage storage until closed
o.navigator information about browser
o.performance data about performance

SCREEN PROPERTIES

o.screen information about screen
n.screenX horizontal pos browser/screen
n.screenY vertical pos browser/screen
n.pageXOffset horizontal pixels scrolled
n.pageYOffset vertical pixels scrolled

WINDOW PROPERTIES

o.opener window that opened this window
o.parent parent of current window/frame
o.self this window (equal to .window)
o.top top window of current win/frame

METHODS

s.btoa(**str**) encode string to base64
s.atob(**str**) decode base64 string to text
z.focus() request send window to front
z.blur() remove focus from window
o.getSelection(**id**) return Selection object
z.postMessage(**msg**, **dst**, **transf**) send
o.open(**url**, **name**, **options**) open popup
z.stop() stop window loading
b.find(**str**, **case**, **back**, **wrap**, **word**, **fr**, **d**)
z.print() open print document window

ANIMATION METHODS

n.requestAnimationFrame(**cb**(**n**))
z.cancelAnimationFrame(**reqID**)

TIMER METHODS

n.setTimeout(**f**(**a...**), **ms**, **a...**) delay&run
z.clearTimeout(**id**) remove timeout
n.setInterval(**f**(**a...**), **ms**, **a...**) run every
z.clearInterval(**id**) remove interval

SCREEN METHODS

z.scrollBy(**x**, **y**) scroll x,y pixels (relative)
z.scrollTo(**x**, **y**) scroll x,y pixels (absolute)
z.moveBy(**x**, **y**) move window by x,y (rel)
z.moveTo(**x**, **y**) move window to x,y (abs)
z.resizeBy(**x**, **y**) resize win by x,y (rel)
z.resizeTo(**w**, **h**) resize win to WxH (abs)

STYLESHEET METHODS

o.getComputedStyle(**elem**, **pseudelem**)
a.matchMedia(**mediaq**) match CSSMQ

screen = info about screen / resolution

PROPERTIES

n.availTop top-from space available
n.availLeft left-from space available
n.availWidth width space available
n.availHeight height space available
n.width screen width resolution
n.height screen height resolution
n.colorDepth screen color depth (bits)
n.pixelDepth screen pixel depth (bits)

METHODS

b.lockOrientation(**mode**|**modearray**)
b.unlockOrientation() remove locks

console = unofficial console browser API

METHODS

z.assert(**cond**, **str1**|**obj1**...) set a assert
z.count(**str**) count (show number times)
z.dir(**obj**) show object (expanded debug)
z.group() open new message group
z.groupCollapsed() open new group coll.
z.groupEnd() close previous group
z.table(**array**|**obj**, **colnames**) show table
z.trace() show code trace
z.timeStamp(**str**) put time on timeline

PERFORMANCE METHODS

z.profile(**name**) start performance profile
z.profileEnd(**name**) stop perf. profile
z.time(**name**) start performance timer
z.timeEnd(**name**) stop perf. timer

LOG LEVEL METHODS

z.log(**str1**|**obj1**...) output message
z.info(**str1**|**obj1**...) output information
z.warn(**str1**|**obj1**...) output warning
z.error(**str1**|**obj1**...) output error

window = global interaction func.

METHODS

USER INTERACTION METHODS

z.alert(**str**) show message (ok button)
s.prompt(**str**, **def**) ask answer to user
b.confirm(**str**) show message (ok, cancel)

history = page history on tab

PROPERTIES

n.length number of pages in historytab
n.state return state top history stack

METHODS

z.back() go prev page (same as .go(-1))
z.forward() go next page (same as .go(1))
z.go(**n**) go **n** page (positive or negative)
z.pushState(**obj**, **title**, **url**) insert state
z.replaceState(**obj**, **title**, **url**) repl. state

storage localStorage / sessionStorage

PROPERTIES

n.length number of items in storage

METHODS

s.key(**n**) return key name on position **n**
s.getItem(**key**) return value of item key
z.setItem(**key**, **value**) set or update key
z.removeItem(**key**) delete item with key
z.clear() delete all items for current site

performance = info about performance

PROPERTIES

o.navigation info about redir/type nav.
o.timing info about latency-load perf.

METHODS

n.now() high precision timestamp

navigator = info about browser

PROPERTIES

b.cookieEnabled browser cookies on?
n.doNotTrack DNT privacy enabled?
o.geolocation user-info geolocation
s.language language in browser
n.maxTouchPoints max on device
b.onLine browser work in online mode?
s.userAgent identify browser of user

METHODS

n.vibrate(**n**|**pattern**) use device vibration

location = info about current URL

PROPERTIES

s.href full document url
s.protocol <https://www.emezeta.com/>
s.username <https://user:pass@www>
s.password <https://user:pass@www>
s.host <https://emezeta.com:81/>
s.hostname <https://emezeta.com:81/>
s.port <https://emezeta.com:81/>
s.pathname <http://emezeta.com/42/>
s.hash <http://emezeta.com/#contacto>
s.search <http://google.com/?q=emezeta>
o.searchParams search params object
s.origin source origin of document url

onClick="..." (HTML) .onclick = (JS func) 'click' (Listener)

e events (only popular events)

MOUSE EVENTS

e.onClick **e**.onDbClick
e.onMouseDown **e**.onMouseUp
e.onMouseEnter **e**.onMouseLeave
e.onMouseMove **e**.onMouseOver
e.onMouseOut **e**.onWheel

KEYBOARD EVENTS

e.onKeyDown **e**.onKeyUp
e.onKeyPress

LOAD/OBJECT EVENTS

e.onDOMContentLoaded **e**.onLoad
e.onAbort **e**.onError
e.onResize **e**.onScroll
e.onBeforeUnload **e**.onUnload

FORM/FIELDS EVENTS

e.onBlur **e**.onFocus
e.onChange **e**.onInput
e.onInvalid **e**.onSelect
e.onReset **e**.onSubmit

ANIMATION/TRANSITION EVENTS

e.onDragEnter **e**.onDragLeave
e.onDragStart **e**.onDragEnd
e.onDragOver **e**.onDrag **e**.onDrop

ANIMATION/TRANSITION EVENTS

e.onAnimationStart **e**.onAnimationEnd
e.onAnimationIteration **e**.transitionEnd

document = Document object

PROPERTIES

s.characterSet document charset
s.compatMode quirks or standard mode
s.cookie return all cookies doc string
s.designMode return design mode status
s.dir return direction text: "rtl" or "ltr"
s.doctype return document type (DTD)
s.domain return document domain
s.documentURI return document URL
s.lastModified return date/time modific.
s.origin return document's origin
s.readyState return current load status
s.referrer return previous page (referrer)
s.title return document title
s.URL return HTML document URL
o.location information about URL

ELEMENTS PROPERTIES

o.activeElement focused element
o.body return body element
o.currentScript return active script
o.defaultView return window element
o.documentElement first element (root)
o.head return head element
o.scrollingElement first scrollable elem.

DOCUMENT ARRAY PROPERTIES

a.anchors array of images elements
a.applets array of applets elements
a.embeds array of embeds elements
a.forms array of forms elements
a.images array of images elements
a.links array of links elements
a.plugins array of plugins elements
a.scripts array of scripts elements

STYLESHEET PROPERTIES

a.styleSheets array of style files elem
o.preferredStyleSheetSet preferred css
o.selectedStyleSheetSet selected css

METHODS

o.adoptNode(**node**) adopt from ext doc
o.createAttribute(**name**) create Attr obj
o.createDocumentFragment()
o.createElement(**tag**) create Element obj
o.createEvent(**type**) create Event object
o.createRange() create Range object
o.createTextNode(**text**) create TextNode
o.enableStyleSheetsForSet(**name**)
o.importNode(**node**, **desc**) import copy
o.getElementById(**id**) find elem with id
a.getElementsByName(**name**) w/ name
o.getSelection(**id**) return Selection object

r ClientRect() = Coords of element

PROPERTIES

n.top top coord of surrounding rect
n.right right coord of surrounding rect
n.bottom bottom coord of surrounding r.
n.left left coord of surrounding rect
n.width width coord of surrounding rect
n.height height coord of surrounding r.

e Element() = Element object

PROPERTIES

s.accessKey if exist, shortcut key
o.attributes array of Attr objects
o.classList DOMTokenList of classes
s.className classes list to string
s.id id string of element
s.name name string of element
s.tagName HTML tag of element

POSITION, SIZE AND SCROLL PROPERTIES

n.clientTop top border width element
n.clientLeft left border width element
n.clientWidth inner width element
n.clientHeight inner height element
n.scrollTop top-position in document
n.scrollLeft left-position in document
n.scrollWidth width of element
n.scrollHeight height of element

GET/SET HTML CODE PROPERTIES

s.innerHTML get/set HTML inside elem
s.outerHTML get/set HTML (incl. elem)

METHODS

o.closest(**selec**) closest ancestor
a.getElementsByClassName(**class**)
a.getElementsByTagName(**tag**)
o.querySelector(**selec**) return first elem
a.querySelectorAll(**selec**) return elems
b.matches(**selec**) match with this elem?
o.insertAdjacentHTML(**posstr**, **html**)

ATTRIBUTE METHODS

b.hasAttributes() exists attributes?
b.hasAttribute(**name**) exist attribute?
s.getAttribute(**name**) return value
o.removeAttribute(**name**) del attribute
o.setAttribute(**name**, **value**) set attrib.

CLIENTRECT (POSITION AND SIZES) METHODS

o.getBoundingClientRect() return pos.
a.getClientRects() return pos/size array

e Event() = Event on action

PROPERTIES

b.bubbles true=bubble, false=captures
b.cancelable event is cancelable?
o.currentTarget current element
b.defaultPrevented preventDefault() call
n.detail additional event info
n.eventPhase current stage (0-3)
b.isTrusted user action or dispatched
o.target reference to dispatched object
n.timeStamp time when was created
s.type type of event

METHODS

o.preventDefault() cancel event
o.stopImmediatePropagation()
o.stopPropagation() prevent being called

t EventTarget (use over elements)

METHODS

o.addEventListener(**ev**, **cb(ev)**, **capt**)
o.removeEventListener(**ev**, **cb(ev)**, **capt**)
b.dispatchEvent(**ev**)

a Attr() = Attribute object

PROPERTIES

s.name name of element attribute
s.value value of element attribute

t DOMTokenList() = List of classes

PROPERTIES

n.length number of items

METHODS

b.contains(**item**) check if item exists
o.add(**item**) add item to list
s.item(**n**) return item number **n**
o.remove(**item**) del item from list
b.toggle(**item**) del item if exist, add else

n Node() = Minor element (elem. or text)

PROPERTIES

s.baseURI absolute base URL of node
s.namespaceURI namespace of node
s.nodeName name of node
s.nodeType 1=element, 2=text, 9=doc
s.nodeValue value of node
s.prefix namespace prefix of node
s.textContent text of node and children

NAVIGATION PROPERTIES

o.childNodes children nodes collection
o.firstChild first children (include text)
o.lastChild last children (include text)
o.nextSibling immediate next node
o.previousSibling immediate prev node
o.parentElement immediate parent elem
o.parentNode immediate parent node
o.ownerDocument return document

METHODS

o.appendChild(**node**) add node to end
o.cloneNode(**child**) duplicate node
o.compareDocumentPosition(**node**)
b.contains(**node**) node is descendant?
b.hasChildNodes() node has childs?
o.insertBefore(**newnode**, **node**)
b.isDefaultNamespace(**nsURI**)
b.isEqualNode(**node**) check if are equal
s.lookupNamespaceURI() ret namesp.
s.lookupPrefix() return prefix for a ns
o.normalize() normalize-form children
o.removeChild(**node**) del node & return
o.replaceChild(**newnode**, **oldnode**)

c ChildNode()

METHODS

o.remove() remove specified node

p ParentNode()

PROPERTIES

n.childElementCount number of children
o.children children elements
o.firstElementChild first children elem.
o.lastElementChild last children elem.

n NonDocumentTypeChildNode()

PROPERTIES

o.nextElementSibling next element
o.previousElementSibling prev element