To include javascript code on a page, the syntax is –

<**script** type = “text/javascript”>   
// all the code  
</**script**>

To create separate file, use **extension** .js and include the file on the page as –

<**script** src="myjsfile.js"></**script**>

|  |  |
| --- | --- |
| **Comments** Single-line Multiple-line | There are two types of comments:   // this is a single line comment /\* this is a multiple line comment when you have to write a lot of things \*/ |
| **Variables** – values that hold data to perform calculations or other operations | * var – most widely used. can be accessed within the function where declared. can be reassigned. * const – constant value i.e. cannot be reassigned * let – can be used only within the block its declared, can be reassigned |
| **Data types** | Can be of different types –   * Number, eg. var id = 20 * Unassigned variable, eg. var x * String, eg. var company = “hackr” * Boolean, eg. var windowopen = true * Constants. eg. const counter = 1 * Operations, eg. var sum = 20 + 20 * Objects, eg. var student = |
| **Objects** | Contains single object of various data types –  Eg, var student = ; |

**Arrays**

Arrays group similar kinds of data together. Eg, var subjectlist = [“math”, “science”, “history”, “computer”]; Arrays can perform the following functions:

|  |  |
| --- | --- |
| **Functions** | **Description** |
| concat() | Concatenate different arrays into one. |
| join() | Joins all the elements of one array as a string |
| indexof() | Returns the index (first position) of an element in the array |
| lastindexof() | Returns the last position of an element in the array |
| sort() | Alphabetic sort of array elements |
| reverse() | Sort elements in descending order |
| valueof() | Primitive value of the element specified |
| slice() | Cut a portion of one array and put it in a new array |
| splice() | Add elements to an array in a specific manner and position |
| unshift() | Add new element to the array in the beginning |
| shift() | Remove first element of the array |
| pop() | Remove the last element of the array |
| push() | Add new element to the array as the last one |
| tostring() | Prints the string value of the elements of the array |

**Operators**

|  |  |
| --- | --- |
| **Basic** | * Addition (+) * Subtraction (-) * Multiply (\*) * Divide (/) * Remainder (%) * Increment (++) * Decrement (--) * Execute brackets first (…) |
| **Logical** | * And (&&) * Or (||) * Not (|) |
| **Comparison** | * Equal to (==) * Equal value and type (===) * Not equal (!=) * Not equal value or type (!==) * Greater than (>) * Less than (<) * Greater than or equal to (>=) * Less than or equal to (<=) * Ternary operator (?) |
| **Bitwise** | * AND (&) * OR (|) * NOT (~) * XOR (^) * Left shift (<<) * Right shift (>>) * Zero fill right shift (>>>) |

**Function**

A group of tasks can be performed in a single function. Eg,

**function** **add**(a, b){// code}

**Outputting the Data**

|  |  |
| --- | --- |
| alert() | Show some output in a small pop up window (alert box) |
| document.write() | Write output to the html document |
| console.log() | Mainly used for debugging, write output on the browser console |
| prompt() | Prompt for user input using dialog box |
| confirm() | Open dialog with yes/no and return true/false based on user click |

**Global Functions**

|  |  |  |
| --- | --- | --- |
| encodeURI() | Encodes a URI into UTF-8 | **var** uri = “hackr.io/blog”; **var** enc = encodeURI(uri); |
| encodeURIComponent() | Encoding for URI components | **var** uri = “hackr.io/blog”; **var** enccomp = encodeURIComponent(uri); |
| decodeURI() | Decodes a [Uniform Resource Identifier (URI)](https://en.wikipedia.org/wiki/Uniform_Resource_Identifier) created by encodeURI or similar | **var** dec = decodeURI(enc); |
| decodeURIComponent() | Decodes a URI component | **var** decomp = decodeURIComponent(enccomp); |
| parseInt() | Parses the input returns an integer | **var** a = parseInt(“2003 monday”); |
| parseFloat() | Parses the input and returns a floating-point number | **var** b = parseFloat(“23.333”); |
| eval() | Evaluates JavaScript code represented as a string | **var** x = eval(“2 \* 2”); |
| Number() | Returns a number converted from its initial value | **var** y = **new** Date(); **var** z = Number(y); |
| isNaN() | Determines whether a value is NaN or not | isNan(25); |
| isFinite() | Determines whether a passed value is a finite number | isFinite(-245); |

**Loops**

|  |  |  |
| --- | --- | --- |
| for | looping in javascript | **var** i; **for** (i = 0; i < 5; i++ { // code} |
| while | execute a block of code while some condition is true | **while** (product.length > 5) {// some code} |
| do… while | similar to while, but executes at least as the condition is applied after the code is executed | **do** { // code }**while** (condition){ } |
| break | break and exit the cycle based on some conditions | **if** (i <10)     break; |
| continue | continue next iteration if some conditions are met | **if** (j>10)   continue; |

**if-else statements**

if-else lets you set various conditions –

**if** (condition 1)  
{  
 //execute **this** code  
} **else** **if** (condition 2)  
{  
 // execute **new** code  
} **else**  
{  
 // execute **if** **no** other condition **is** **true**  
}

**String Methods**

|  |  |  |
| --- | --- | --- |
| **Method** | **Meaning** | **Example** |
| length | determines length of string | **var** a = “hackr.io”; a.length; |
| indexof() | finds position of the first occurrence of a character or text in the string | **var** a = “hackr.io **is** nice website”; **var** b = a.indexof(“nice”); |
| lastindexof() | returns last occurrence of text in a string | **var** a = “hackr.io **is** nice website”; **var** b = a.indexof(“nice”, 6); |
| search() | searches and returns position of a specified value in string | var a = “hackr.io is nice website”; var b = a.search(“nice”); |
| slice() | extracts and returns part of a string as another new string | **var** a = “hackr.io **is** nice website”; **var** b = a.slice(13); will **return** nice website. |
| substring() | substring returns part of the string from start index to the end index specified. cannot take negative values unlike slice() | **var** a = “hackr.io **is** nice website”; **var** b = a.substring(0, 7); |
| substr() | returns the sliced out portion of a string, the second parameter being the length of the final string. | **var** a = “hackr.io **is** nice website”; **var** b = a.substr(13, 8); |
| replace() | replaces a particular value with another | **var** a = “hackr.io **is** nice website”; **var** b = a.replace(“nice”, “good”); |
| touppercase() | changes all characters into uppercase | **var** a = “hackr.io **is** nice website”; **var** b = a.touppercase (a); |
| tolowercase() | changes all characters into lowercase | **var** a = “hackr.io **is** nice website”; **var** b = a.tolowercase(a); |
| concat() | joins two or more strings together into another string | **var** a = “my name **is**”; **var** b = “john”; **var** c = a.concat(“: ”, b); |
| trim() | removes white spaces from a string | **var** a = “       hi, there!         ”; a.trim(); |
| charat() | finds character at a specified position | **var** a = “hackr.io”;  a.charat(1) will return a |
| charcodeat() | returns the unicode of character at the specified position | “**hackr**”.charcodeat(0);  will return 72 |
| split() | convert a string into array based on special character | **var** a = “hackr.io”; **var** arr = a.split(“”);  will return an array of characters h,a,c,k,r and so on.. |
| accessing characters using [] | access a character of string using its index (doesn’t work on some versions of ie) | **var** a = “hackr.io”; a[2] will **return** c |

**Escape characters**

|  |  |
| --- | --- |
| \' | Single quote |
| \" | Double quote |
| \\ | Single backslash |
| \b | Backspace |
| \f | Form feed |
| \n | New line |
| \t | Horizontal tab |
| \v | Vertical tab |
| \r | Carriage return |

**Regular Expressions**

Regular expressions can be in the form of pattern modifiers, metacharacters, quantifiers and brackets. **Pattern modifiers**

|  |  |
| --- | --- |
| e | evaluate replacement |
| i | case-insensitive matching |
| g | global matching – find all matches |
| m | multiple line matching |
| s | treat strings as a single line |
| x | allow comments and whitespace in the pattern |
| u | ungreedy pattern |

**Brackets**

|  |  |
| --- | --- |
| [abc] | Find any of the characters between the brackets |
| [^abc] | Find any character which are not in the brackets |
| [0-9] | Used to find any digit from 0 to 9 |
| [A-z] | Find any character from uppercase A to lowercase z |
| (a|b|c) | Find any of the alternatives separated with | |

**Metacharacters**

|  |  |
| --- | --- |
| . | Find a single character, except newline or line terminator |
| \w | Word character |
| \W | Non-word character |
| \d | A digit |
| \D | A non-digit character |
| \s | Whitespace character |
| \S | Non-whitespace character |
| \b | Find a match at the beginning/end of a word |
| \B | A match not at the beginning/end of a word |
| \0 | NULL character |
| \n | A new line character |
| \f | Form feed character |
| \r | Carriage return character |
| \t | Tab character |
| \v | Vertical tab character |
| \xxx | The character specified by an octal number xxx |
| \xdd | Character specified by a hexadecimal number dd |
| \uxxxx | The Unicode character specified by a hexadecimal number xxxx |

**Quantifiers**

|  |  |
| --- | --- |
| n+ | Matches string that contains at least one ‘n’ |
| n\* | Any string containing zero or more occurrences of n |
| n? | A string that has no or one occurrence of n |
| n | String that contains a sequence of X n’s |
| n | Strings that contain a sequence of X to Y n’s |
| n | Matches string that has a sequence of at least X n’s |
| n$ | Any string with n at the end of it |
| ^n | String with n at the beginning of it |
| ?=n | Any string that is followed by the string n |
| ?!n | String that is not followed by the string n |

**Numbers**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Number properties** | |  |  | | --- | --- | | MAX\_VALUE | The maximum numeric value that can be represented in JavaScript | | MIN\_VALUE | Smallest positive numeric value possible in JavaScript | | NaN | Not-a-Number | | NEGATIVE\_INFINITY | The negative Infinity value | | POSITIVE\_INFINITY | Positive Infinity value | |
| **Number methods**   |  |  |  | | --- | --- | --- | | **Method** | **Meaning** | **Example** | | toExponential() | Returns the string with a number rounded to and written in exponential form | **var** a = 3.1417; a.toExponential(2); will give 3.14e+0 | | toFixed() | Returns the string of a number with specific number of decimals | **var** a = 3.1417; a.toFixed(2); will **return** 3.14 | | toPrecision() | Returns string to the precision of the specified decimal | **var** a = 3.46; a.to{recision(2); returns 3.5 | | valueOf() | Converts number object to primitive type | **var** x = 23; x.valueOf(); | | |
| **Math properties** | |  |  | | --- | --- | | E | Euler’s number | | LN2 | The natural logarithm with base 2 | | LN10 | Natural logarithm with base 10 | | LOG2E | Base 2 logarithm of E | | LOG10E | Base 10 logarithm of E | | PI | The number PI (3.14…) | | SQRT1\_2 | Square root of 1/2 | | SQRT2 | Square root of 2 | |
| **Math methods** | |  |  | | --- | --- | | *All angle values are in radian* | | | abs(x) | Returns the absolute (positive) value of x | | acos(x) | The arccosine of x | | asin(x) | Arcsine of x | | atan(x) | The arctangent of x (numeric) | | atan2(y,x) | Arctangent of the quotient of its arguments | | sin(x) | The sine of x | | cos(x) | The cosine of x | | tan(x) | The tangent of an angle | | exp(x) | Value of Ex | | ceil(x) | Value of x rounded up to its nearest integer | | floor(x) | The value of x rounded down to its nearest integer | | log(x) | The natural logarithm (base E) of x | | max(x,y,z,...,n) | Returns the number with the highest value | | min(x,y,z,...,n) | Same for the number with the lowest value | | pow(x,y) | X to the power of y | | round(x) | The value of x rounded to its nearest integer | | sqrt(x) | Square root of x | | random() | Returns a random number between 0 and 1 | |

**Dates**

|  |  |
| --- | --- |
| Date() | Creates a new date object with current date and time |
| Date(2019, 10, 21, 12, 24, 58, 13) | Create a custom date object. Format – (yyyy, mm, dd, hh, min, s, ms). Except for year and month, all parameters are optional. |
| Date("2019-10-21") | Date declaration as a string |
| getDate() | Get the day of the month as a number (1-31) |
| getDay() | The weekday as a number (0-6) |
| getFullYear() | Year as a four-digit number (yyyy) |
| getHours() | Get the hour (0-23) |
| getMilliseconds() | Get the millisecond (0-999) |
| getMinutes() | Get the minute (0-59) |
| getMonth() | Month as a number (0-11) |
| getSeconds() | Get the second (0-59) |
| getTime() | Get the milliseconds since January 1, 1970 |
| getUTCDate() | The day (date) of the month in the specified date according to universal time (also available for day, month, full year, hours, minutes etc.) |
| parse | Parses a string representation of a date and returns the number |
| setDate() | Set the day as a number (1-31) |
| setFullYear() | Sets the year (optionally month and day) |
| setHours() | Set the hour (0-23) |
| setMilliseconds() | Set milliseconds (0-999) |
| setMinutes() | Sets the minutes (0-59) |
| setMonth() | Set the month (0-11) |
| setSeconds() | Sets the seconds (0-59) |
| setTime() | Set the time (milliseconds since January 1, 1970) |
| setUTCDate() | Sets the day of the month for a specified date according to universal time (also available for day, month, full year, hours, minutes etc.) |

**DOM mode**

**D**ocument **O**bject **M**odel) is the code of the page structure. HTML elements (called as nodes) can be easily manipulated using JavaScript.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Node properties** | |  |  | | --- | --- | | attributes | Returns all attributes registered to an element | | baseURI | Provides the absolute base URL of an HTML element | | nodeName | the name of a node | | nodeType | type of a node | | nodeValue | sets or gets value of a node | | parentNode | parent node of an element | | childNodes | all child nodes of an element | | firstChild | first child node of an element | | lastChild | last child node of an element | | ownerDocument | top-level document object for this (current) node | | previousSibling | node immediately preceding the current one | | nextSibling | next node in the same node tree level | | textContent | Sets or returns the textual content of a node and its descendants | |
| **Node methods** | |  |  | | --- | --- | | cloneNode() | Clones an HTML element | | compareDocumentPosition() | Compares the document position of two elements | | isDefaultNamespace() | Returns true if the specified namespaceURI is the default | | lookupNamespaceURI() | Returns the namespace URI associated with the given node | | getFeature() | Returns an object which implements the APIs of a specified feature | | isSupported() | Returns true if a specified feature is supported on the element | | hasAttributes() | Returns true if an element has any attributes | | insertBefore() | Inserts a new child node before a specified, existing child node | | isEqualNode() | Checks if two elements are equal | | isSameNode() | Checks if two elements are the same node | | hasChildNodes() | Returns true if an element has any child nodes | | lookupPrefix() | Returns a DOMString containing the prefix for a given namespace URI, if present | | normalize() | Joins adjacent text nodes and removes empty text nodes in an element | | removeChild() | Removes a child node from an element | | replaceChild() | Replaces a child node in an element | | appendChild() | Adds a new child node to an element as the last child node | |
| **Element methods** | |  |  | | --- | --- | | getAttribute() | Returns the specified attribute value of an element node | | getAttributeNS() | Returns string value of the attribute with the specified namespace and name | | getAttributeNode() | Gets the specified attribute node | | getAttributeNodeNS() | Returns the node for the attribute with the given namespace and name | | getElementsByTagName() | Provides a collection of all child elements within the specified tag name | | getElementsByTagNameNS() | Returns HTML elements with particular tag name with the given namespace | | hasAttribute() | Returns true if an element has any attributes, otherwise false | | hasAttributeNS() | Provides a true/false value indicating whether the current element in a given namespace has the specified attribute | | setAttribute() | Sets or changes the specified attribute to the specified value | | setAttributeNS() | Adds a new attribute or changes the value of an existing attribute with the given namespace and name | | setAttributeNode() | Sets or modifies the specified attribute node | | setAttributeNodeNS() | Adds a new name spaced attribute node to an element | | removeAttribute() | Removes a specified attribute from an element | | removeAttributeNS() | Removes and returns the specified attribute node within a certain namespace | | removeAttributeNode() | Removes and returns the specified attribute node | |

**Browser actions**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Window properties** | |  |  | | --- | --- | | closed | Checks if a window has been closed | | defaultStatus | Sets or gets the default text in the windows status bar | | self | the current window | | top | topmost browser window | | parent | parent window of the current window | | document | Returns the window document object | | frames | Returns all <iframe> elements in the current window | | history | History object for the window | | innerHeight | The inner height of window’s content area | | innerWidth | The inner width of content area | | length | number of  <iframe> elements in the window | | location | location object for the window | | name | Sets or gets the window name | | navigator | Returns the Navigator object for the window | | opener | reference to the window that created the window | | outerHeight | outer height of a window, including toolbars/scrollbars | | outerWidth | outer width of a window, including toolbars/scrollbars | | pageXOffset | Number of pixels the current document has been scrolled horizontally | | pageYOffset | Number of pixels the current document has been scrolled vertically | | screen | Returns the Screen object for the window | | screenLeft | The horizontal coordinate of the window | | screenTop | The vertical coordinate of the window | | screenX | Same function as screenLeft (for some browsers) | | screenY | Same function as screenTop (for some browsers) | | status | Sets or gets the text in the status bar of a window | |
| **Window methods** | |  |  | | --- | --- | | alert() | Displays an alert box with a message and an OK button | | blur() | Removes focus from the current window | | clearTimeout() | Clears a timer set with setTimeout() | | clearInterval() | Clears a timer set with setInterval() | | close() | Closes the current window | | open() | Opens a new browser window | | stop() | Stops the window from loading | | confirm() | Displays a dialogue box with a message and an OK and Cancel button | | focus() | Sets focus to the current window | | moveBy() | Moves a window relative to its current position | | moveTo() | Moves a window to a specified position | | print() | Prints the content of the current window | | prompt() | Displays a dialogue box that prompts the visitor for input | | resizeBy() | Resizes the window by the specified number of pixels | | resizeTo() | Resizes the window to a specified width and height | | scrollBy() | Scrolls the document by a specified number of pixels | | scrollTo() | Scrolls the document to specified coordinates | | setInterval() | Calls a function or evaluates an expression at specified intervals | | setTimeout() | Calls a function or evaluates an expression after a specified interval | |
| **Screen properties** | |  |  | | --- | --- | | availHeight | Returns the height of the screen (excluding the Windows Taskbar) | | availWidth | Returns the width of the screen (excluding the Windows Taskbar) | | colorDepth | Returns the bit depth of the color palette for displaying images | | height | The total height of the screen | | pixelDepth | The color resolution of the screen in bits per pixel | | width | The total width of the screen | |

**User Events**

**1. Mouse**

|  |  |
| --- | --- |
| onclick | event that happens when user clicks on an element |
| onmouseover | when the mouse is moved over some element or its children |
| onmouseout | User moves the mouse pointer out of an element or one of its children |
| onmouseup | when user releases a mouse button while over an element |
| onmousedown | when user presses a mouse button over an element |
| onmouseenter | pointer moves onto an element |
| onmouseleave | Pointer moves out of an element |
| onmousemove | pointer is moving when it is over an element |
| oncontextmenu | User right-clicks on an element to open a context menu |
| ondblclick | The user double-clicks on an element |

**2. Keyboard**

|  |  |
| --- | --- |
| onkeydown | When the user is pressing a key down |
| onkeypress | The moment the user starts pressing a key |
| onkeyup | The user releases a key |

**3. Frame**

|  |  |
| --- | --- |
| onabort | The loading of a media is aborted |
| onbeforeunload | Event that occurs before a document is to be unloaded |
| onunload | Event occurs when a page has unloaded |
| onerror | When an error occurs while loading an external file |
| onhashchange | There have been changes to the anchor part of a URL |
| onload | When an object has loaded |
| onpagehide | The user navigates away from a webpage |
| onpageshow | the user navigates to a webpage |
| onresize | The document view is resized |
| onscroll | An element’s scrollbar is being scrolled |

**4. Form**

|  |  |
| --- | --- |
| onblur | When an element loses focus |
| onchange | when content of a form element like <input>, <select> and <textarea> changes |
| onfocus | An element gets focus |
| onfocusin | When an element is about to get focus |
| onfocusout | When element is about to lose focus |
| oninput | User input on an element |
| oninvalid | An element is invalid |
| onreset | form reset |
| onsearch | The user writes something in the input type search |
| onselect | The user selects some text (<input> and <textarea>) |
| onsubmit | event that happens upon submitting the form |

**5. Drag**

|  |  |
| --- | --- |
| ondrag | An element is dragged |
| ondrop | Dragged element is dropped on the drop target |
| ondragstart | User starts to drag an element |
| ondragend | The user has finished dragging the element |
| ondragenter | The dragged element enters a drop target |
| ondragleave | A dragged element leaves the drop target |
| ondragover | The dragged element is on top of the drop target |

**6. Clipboard**

|  |  |
| --- | --- |
| oncut | event that happens when user cuts content of an element |
| oncopy | event that happens when user copies content of an element |
| onpaste | event that happens when user pastes content of an element |

**7. Media**

|  |  |
| --- | --- |
| onabort | Media loading is aborted |
| onended | The media ended |
| onerror | Happens when an error occurs while loading an external file |
| oncanplay | The browser can start playing media |
| oncanplaythrough | The browser can play through media without stopping |
| ondurationchange | change in the duration of the media |
| onloadeddata | Media data loaded |
| onloadedmetadata | Metadata (e.g. dimensions, duration) are loaded |
| onloadstart | The browser starts looking for specified media |
| onpause | Media is paused either by the user or automatically |
| onplay | The media started to play or is no longer paused |
| onplaying | Media is playing after being paused or stopped for buffering |
| onprogress | The browser is in the process of downloading the media |
| onratechange | The playing speed of the media changes |
| onseeked | User is finished moving/skipping to a new position in the media |
| onseeking | The user starts moving/skipping |
| onstalled | The browser is trying to load the media but it is unavailable |
| onwaiting | Media paused but expected to resume (like in buffering) |
| onsuspend | The browser is intentionally not loading media |
| ontimeupdate | The playing position has changed (like in case of fast forward) |
| onvolumechange | Media volume has increased or reduced |

**8. Animation**

|  |  |
| --- | --- |
| animationstart | CSS animation started |
| animationend | CSS animation ended |
| animationiteration | CSS animation plays over |

**9. Other**

|  |  |
| --- | --- |
| transitionend | event triggered when a CSS transition has completed |
| onmessage | A message is received through the event source |
| ononline | The browser starts to work online |
| onoffline | The browser starts to work offline |
| ontoggle | The user opens or closes the <details> element |
| onpopstate | When the window’s history changes |
| onshow | A <menu> element is shown as a context menu |
| onstorage | A Web Storage area is updated |
| onwheel | Mouse wheel rolls up or down over an element |
| ontouchstart | A finger is placed on the touch-screen |
| ontouchend | User’s finger is removed from a touch-screen |
| ontouchcancel | Screen-touch is interrupted |
| ontouchmove | User finger is dragged across the screen |

**10. Errors**

|  |  |
| --- | --- |
| try | block of code to execute in case of no errors |
| catch | block of code to execute in case of an error |
| throw | Create custom error messages rather than standard JavaScript errors |
| finally | block that is always executed whether there is error in execution or not |

**Error-values**

Each error has a name and message property that define it.

* **name:**Sets or gets the error name
* **message:**Sets or gets error in an understandable string format

|  |  |
| --- | --- |
| EvalError | error occurred in the eval() function |
| RangeError | number out of range |
| ReferenceError | illegal reference occurred |
| SyntaxError | syntax error |
| TypeError | type error |
| URIError | encodeURI() error |