Methods of the Windows Object

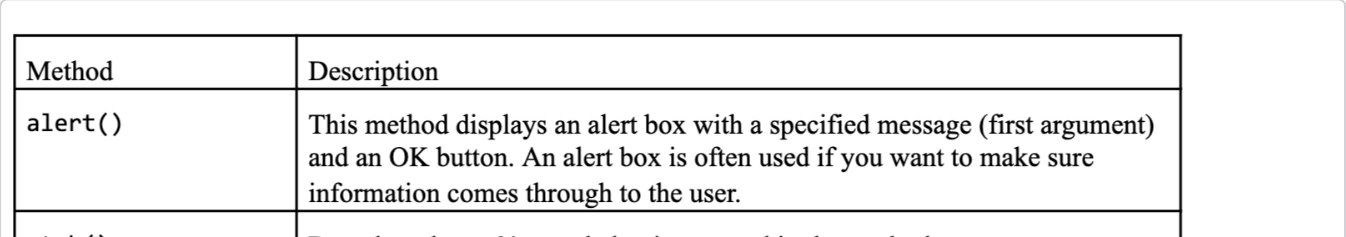
In this lesson, we will discover some methods to work with the windows object.
Let's begin!

WE'LL COVER THE FOLLOWING

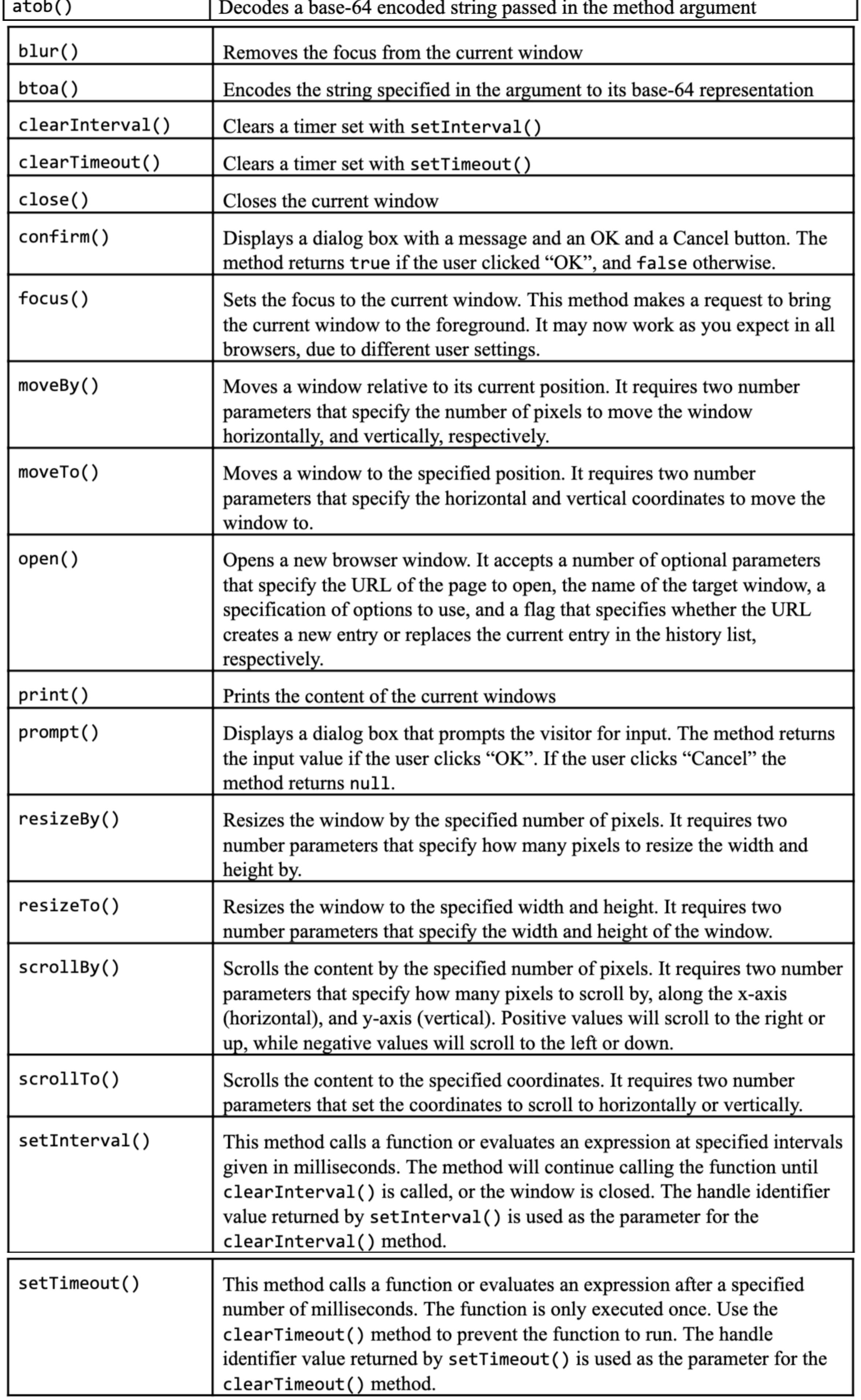
- The methods of the window object
- Exercise-08-22

You already know document from Chapter 7, and you will learn about the other objects in this section. The table below summarizes the methods of window.

The methods of the window object



Method	Description
`alert()`	This method displays an alert box with a specified message (first argument) and an OK button. An alert box is often used if you want to make sure information comes through to the user.



`atob()`	Decodes a base-64 encoded string passed in the method argument
`blur()`	Removes the focus from the current window
`btoa()`	Encodes the string specified in the argument to its base-64 representation
`clearInterval()`	Clears a timer set with `setInterval()`
`clearTimeout()`	Clears a timer set with `setTimeout()`
`close()`	Closes the current window
`confirm()`	Displays a dialog box with a message and an OK and a Cancel button. The method returns `true` if the user clicked "OK", and `false` otherwise.
`focus()`	Sets the focus to the current window. This method makes a request to bring the current window to the foreground. It may now work as you expect in all browsers, due to different user settings.
`moveBy()`	Moves a window relative to its current position. It requires two number parameters that specify the number of pixels to move the window horizontally, and vertically, respectively.
`moveTo()`	Moves a window to the specified position. It requires two number parameters that specify the horizontal and vertical coordinates to move the window to.
`open()`	Opens a new browser window. It accepts a number of optional parameters that specify the URL of the page to open, the name of the target window, a specification of options to use, and a flag that specifies whether the URL creates a new entry or replaces the current entry in the history list, respectively.
`print()`	Prints the content of the current windows
`prompt()`	Displays a dialog box that prompts the visitor for input. The method returns the input value if the user clicks "OK". If the user clicks "Cancel" the method returns `null`.
`resizeBy()`	Resizes the window by the specified number of pixels. It requires two number parameters that specify how many pixels to resize the width and height by.
`resizeTo()`	Resizes the window to the specified width and height. It requires two number parameters that specify the width and height of the window.
`scrollBy()`	Scrolls the content by the specified number of pixels. It requires two number parameters that specify how many pixels to scroll by, along the x-axis (horizontal), and y-axis (vertical). Positive values will scroll to the right or up, while negative values will scroll to the left or down.
`scrollTo()`	Scrolls the content to the specified coordinates. It requires two number parameters that set the coordinates to scroll to horizontally or vertically.
`setInterval()`	This method calls a function or evaluates an expression at specified intervals given in milliseconds. The method will continue calling the function until `clearInterval()` is called, or the window is closed. The handle identifier value returned by `setInterval()` is used as the parameter for the `clearInterval()` method.
`setTimeout()`	This method calls a function or evaluates an expression after a specified number of milliseconds. The function is only executed once. Use the `clearTimeout()` method to prevent the function to run. The handle identifier value returned by `setTimeout()` is used as the parameter for the `clearTimeout()` method.

In the source code below, you will find a sample project in the Exercise-08-22 folder. Examine the **index.html** and **smallwindow.html** files that demonstrate most of the properties and methods of the window object.

Exercise-08-22

```
ï»¿<!DOCTYPE html>
<html>
<head>
  <title>Using the window object</title>
  <style>
    body {
      font-family: "Verdana", "Arial", sans-serif;
    }

    button {
      border: 1px dotted dimgray;
      border-radius: 4px;
      padding: 4px 8px;
    }

    button:hover {
      background-color: aliceblue;
    }
  </style>
  <script>
    function queryWindow() {
      var label = document.getElementById("props");
      var out = "";

      function addText(text, value) {
        out += text + ": " + value + "<br/>";
      }

      addText("Inner width", innerWidth);
      addText("Inner height", innerHeight);
      addText("Outer height", outerHeight);
      addText("Outer height", outerHeight);
      addText("Screen X", screenX);
      addText("Screen Y", screenY);
      addText("Screen left", screenLeft);
      addText("Screen top", screenTop);
      addText("Page X offset", pageXOffset);
      addText("Page Y offset", pageYOffset);
      label.innerHTML = out;
    }

    function alertSample() {
      alert("This is an alert dialog");
    }

    function confirmSample() {
      var result = confirm("Is this OK?");
      alert("The result is " + result);
    }

    function promptSample() {
      var result =
        prompt("Your input", "value");
      alert("The result is " + result);
    }
    }

    function openSample1() {
      open("http://developer.mozilla.org",
        "Sample window",
        "width=200,height=100," +
        "menubar=no,top=200,left=300");
    }
    function openSample2() {
      open("smallwindow.html",
        "Sample window",
        "width=400,height=200," +
        "menubar=no,top=200,left=300");
    }

    var timer;
    var counter;
    function startTimer() {
      counter = 0;
      timer = setInterval(function () {
        counter++;
        var label1 = document.getElementById("label1");
        label1.innerText = "Timer loops: " + counter;
      }, 200);
    }

    function stopTimer() {
      if (timer) {
        clearInterval(timer);
        timer = null;
        var label1 = document.getElementById("label1");
        label1.innerText = "Timer stopped.";
      }
    }

    var timeout;
    function startTimeout() {
      var label2 = document.getElementById("label2");
      label2.innerText = "Timeout set to 3 seconds...";
      timeout = setTimeout(function () {
        counter++;
        label2.innerText = "Timeout event fired.";
        timeout = null;
      }, 3000);
    }

    function stopTimeout() {
      if (timeout) {
        clearTimeout(timeout);
        timeout = null;
        var label2 = document.getElementById("label2");
        label2.innerText = "Timeout event cancelled.";
      }
    }
  </script>
</head>
<body>
  <h3>Dialogs</h3>
  <button onclick="alertSample()">
    alert()
  </button>
  <button onclick="confirmSample()">
    confirm()
  </button>
  <button onclick="promptSample()">
    prompt()
  </button>
  <hr />
  <h3>New window</h3>
  <button onclick="openSample1()">
    open() #1
  </button>
  <button onclick="openSample2()">
    open() #2
  </button>
  <hr />
  <h3>Window properties</h3>
  <button onclick="queryWindow()">
    Get window properties
  </button>
  <p id="props"></p>
  <h3>Timers</h3>
  <button onclick="startTimer()">
    setInterval();
  </button>
  <button onclick="stopTimer()">
    clearInterval();
  </button>
  <p id="label1">No interval set.</p>
  <hr />
  <button onclick="startTimeout()">
    setTimeout();
  </button>
  <button onclick="stopTimeout()">
    clearTimeout();
  </button>
  <p id="label2">No timeout set.</p>
</body>
</html>
```

In the *next lesson*, we will meet the history object.

See you there! :)