**Day 1: Popups and Window Methods**

A "window popup" typically refers to a new browser window or tab that is generated by a web application or website and appears on top of the main browser window. These popups serve various purposes, such as displaying additional content, forms, advertisements, or alerts to the user.

* popup is a separate window with its own independent JavaScript environment. So opening a popup with a third-party non-trusted site is safe.
* It’s very easy to open a popup.
* A popup can navigate (change URL) and send messages to the opener window.

The syntax to open a popup is: window.open(url, name, params):

**url**

An URL to load into the new window.

**name**

A name of the new window. Each window has a window.name, and here we can specify which window to use for the popup. If there’s already a window with such name – the given URL opens in it, otherwise a new window is opened.

**params**

The configuration string for the new window. It contains settings, delimited by a comma. There must be no spaces in params, for instance: width:200,height=100.

**Settings for params:**

**Position:**

left/top (numeric) – coordinates of the window top-left corner on the screen. There is a limitation: a new window cannot be positioned offscreen.

width/height (numeric) – width and height of a new window. There is a limit on minimal width/height, so it’s impossible to create an invisible window.

**Window features:**

* + menubar (yes/no) – shows or hides the browser menu on the new window.
  + toolbar (yes/no) – shows or hides the browser navigation bar (back, forward, reload etc) on the new window.
  + location (yes/no) – shows or hides the URL field in the new window. FF and IE don’t allow to hide it by default.
  + status (yes/no) – shows or hides the status bar. Again, most browsers force it to show.
  + resizable (yes/no) – allows to disable the resize for the new window. Not recommended.
  + scrollbars (yes/no) – allows to disable the scrollbars for the new window. Not recommended.

The open function typically expects a URL as its first argument, but you provided '/'. If you want to open a new window/tab with a specific set of features, you should provide a valid URL as the first argument.

The params string contains line breaks and spaces, which might cause issues in some JavaScript environments. It's better to remove unnecessary whitespace and line breaks to ensure your code works consistently across different platforms.

The open call returns a reference to the new window. It can be used to manipulate it’s properties, change location and even more.

A popup may access the “opener” window as well using window.opener reference. It is null for all windows except popups.

To close a window: win.close().

To check if a window is closed: win.closed.

Technically, the close() method is available for any window, but window.close() is ignored by most browsers if window is not created with window.open(). So it’ll only work on a popup.

The closed property is true if the window is closed. That’s useful to check if the popup (or the main window) is still open or not. A user can close it anytime, and our code should take that possibility into account.

**There are methods to move/resize a window:**

**win.moveBy(x,y)**

Move the window relative to current position x pixels to the right and y pixels down. Negative values are allowed (to move left/up).

**win.moveTo(x,y)**

Move the window to coordinates (x,y) on the screen.

**win.resizeBy(width,height)**

Resize the window by given width/height relative to the current size. Negative values are allowed.

**win.resizeTo(width,height)**

Resize the window to the given size.

There’s also window.onresize event.

**Only popups**

To prevent abuse, the browser usually blocks these methods. They only work reliably on popups that we opened, that have no additional tabs.

**No minification/maximization**

JavaScript has no way to minify or maximize a window. These OS-level functions are hidden from Frontend-developers.

**win.scrollBy(x,y)**

Scroll the window x pixels right and y down relative the current scroll. Negative values are allowed.

**win.scrollTo(x,y)**

Scroll the window to the given coordinates (x,y).

**elem.scrollIntoView(top = true)**

Scroll the window to make elem show up at the top (the default) or at the bottom for elem.scrollIntoView(false).

Move/resize methods do not work for maximized/minimized windows.

Theoretically, there are window.focus() and window.blur() methods to focus/unfocus on a window. Also there are focus/blur events that allow to focus a window and catch the moment when the visitor switches elsewhere.

In the past evil pages abused those. For instance, look at this code:

window.onblur = () => window.focus();

When a user attempts to switch out of the window (blur), it brings it back to focus. The intention is to “lock” the user within the window.

So, there are limitations that forbid the code like that. There are many limitations to protect the user from ads and evils pages. They depend on the browser.

For instance, a mobile browser usually ignores that call completely. Also focusing doesn’t work when a popup opens in a separate tab rather than a new window.

Still, there are some things that can be done.

**For instance:**

* When we open a popup, it’s might be a good idea to run a newWindow.focus() on it. Just in case, for some OS/browser combinations it ensures that the user is in the new window now.
* If we want to track when a visitor actually uses our web-app, we can track window.onfocus/onblur. That allows us to suspend/resume in-page activities, animations etc. But please note that the blur event means that the visitor switched out from the window, but they still may observe it. The window is in the background, but still may be visible.