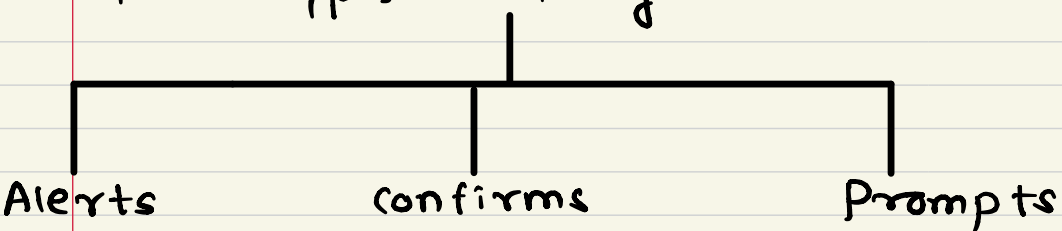
6. Using Javascripts

A. Naik.

* Dialogs:

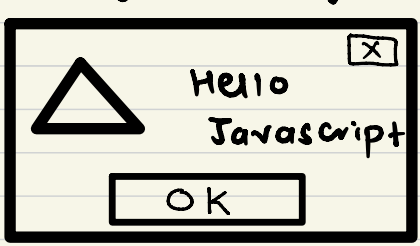
- A dialog box or simply dialog is a small window in GUI that pops up requesting some action from a user.

- Three types of dialog box:



1. Alerts:

- The window's object alert() method creates special small window with a short string message and ok button



- Syntax:

```
window.alert(`string');
```

Shorthand:

```
alert(string);
```

- String passed to alert may be variable

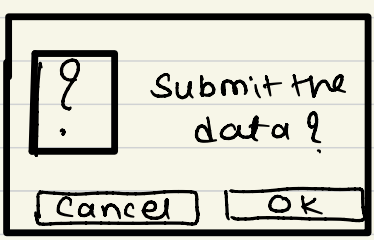
valid alert() method:

- alert(" Hi ");

- var msg = "Hi";
  alert(msg)

2. Confirm:

- The confirm() method for the window object creates a window that displays a message for user to respond to it by clicking either an OK button to agree or cancel button to disagree.



- Syntax :

```
window.confirm(string);
```

or

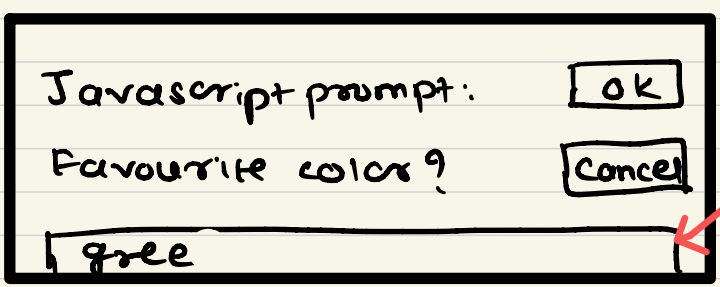
simply

```
confirm(string);
```

- It returns boolean value that indicates whether or not the information was confirmed,
  true → ok
  false → cancel.

3. Prompt( ):

- A prompt window is a small data collection dialog that prompts the user to enter a short line of data.



- Syntax:

```
window.prompt(prompt string, default
                value string);
```

- prompt() method takes two arguments. The first string that displays the prompt value and second is the default value to put in prompt window.
- This method returns a string value that contains the value entered by user in prompt.

- The shorthand prompt() is almost used instead of window.prompt().

```
result = prompt("what's your name?", " ");
```

empty string so value shouldn't be undefined.

- Need to check if user clicked cancel button, the value returned will be null.

* Opening and closing Generic windows:

- The window object methods open() and close() are used to create and destroy a window, resp.

1. Open() :

- when you open a window, we can set its URL, name, size, buttons and other attributes.

- Basic syntax :

```
window.open(url, name, features, replace)
```

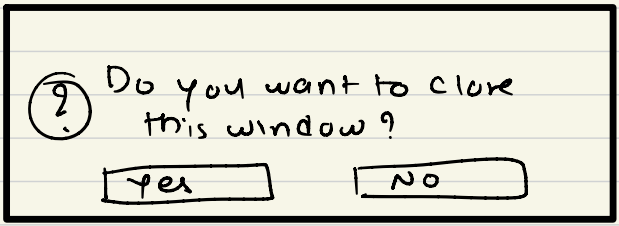
url - URL indicates the document to load into window.
name - name for window.
features - string that lists the features of window.
replace - optional Boolean value that indicates if URL specified should replace the window's content or not.

2. Close() :

Syntax:

```
window.close().
```

Can see the below window in some browser :



Others may actually close down the main window without warning. browsers.

* Controlling windows

1. Moving windows:

- Moving windows around screen is possible using two methods
  1. window.moveBy()
  2. window.moveTo()

- Syntax for moveBy():

```
windowname.moveBy(horizontal pixels, vertical pixels)
```

windowname — name of window to move
horizontalpixels — no. of. horizontal pixels to move window.
verticalpixels — no. of. vertical pixels to move window

e.g. myWindow.moveBy(100,100);
would move window up 100 pixels and right 100 pixels.

- Syntax for moveTo():

```
windowname.moveTo(x-coord, y-coord);
```

windowname - name of window to move
x-coord - screen coordinate on x axis to move window to.
y-coord - screen coordinate on y axis to move window to.

2. Resizing windows:

- The method window.resizeBy(horizontal, vertical) resizes a window by values given in horizental and vertical.
- Negative values makes window smaller and positive values make windows biggen.

`myWindow.resizeBy(10, 10);` ⇒ makes window 10 pixels taller and wider.

- window.resizeTo(width, height) resizes the window to the specified width and height indicated.

`myWindow.resizeTo(100, 100)` ⇒ makes window 100x100.

3. Scrolling windows:

- Window objects scrollBy() and scrollTo() methods to scroll window by certain number of pixels or to a particular pixels location.

- `Mywindow.scrollBy(100, 100)` ⇒ scroll 100 pixels to right and down.

- `myWindow.scrollTo(1, 1)` ⇒ scroll to 1, 1 the origin.

4. Accessing window's History:

- Javascript provides the History object a way to access the history list for particular browser window.
- History object is array of URL strings that show where user has been recently.

`window.history.forward()` ⇒ gives forward history.

`window.history.backward()` ⇒ gives backward history.

- Possible to access a particular item in history list by history.go() method. Negative values gives history of previous and positive will give forward in history list.

`window.history.go(-2)` ⇒ Back two times

`window.history.go(3)` ⇒ forward three times

5. Controlling Windows Status Bar:

- Status bar is small text area in lower-left corner of browser window where messages like downloading progress can be seen.
- `window.status = 'Hello world';` ⇒ Displayed only for short time
- `default status = 'Javascript';` ⇒ Displayed any time, nothing else going on browser window

* Window Events:

- Windows objects supports many events.
- The safe-cross browser window events are onblur, onerror, onload, onfocus, onunload, onresize.
- Below is the table showing:

Event	Description
onblur	Fires when the window loses focus.
onerror	Rudimentary error handling event fired when a JavaScript error occurs.
onfocus	Fires when the window gains focus.
onload	Fires when the document is completely loaded into the window. Warning: Timing of this event is not always exact.
onresize	Event triggered as user resizes the window.
onunload	Triggered when the document is unloaded, such as following an outside link or closing the window.

- Window events handlers can be set through HTML event attributes on body element.

```
<body  onload="alert('Hello');"
       onunload="alert('world');">.
```

- Use of window object:

```
function sayHi() { alert('hi'); }
window.onload = sayHi;
```

- Internet Explorer add numerous events to window object.

- Useful window object:

Event	Description
onafterprint	Event triggered after the window is printed.
onbeforeprint	Fires just before the window is printed or print previewed.
onbeforeunload	The event is triggered just before the window unloads. Should happen before the **onunload** event.
ondragdrop	Is triggered when a document is dragged onto a window. (Netscape only.)
onhelp	Fires when the Help key, generally F1, is clicked.
onresizeend	Fires when the resize process ends—usually the user has stopped dragging the corner of a window.
onresizestart	Fires when the resize process begins—usually the user has started dragging the corner of a window.
onscroll	Fires when the window is scrolled in either direction.

* Frames : Special Case of Windows

- Frames are a special case in Javascript - where each frame behaves like separate document.
- To modify anything in another frame, we need to gain control of that frame.
- Frames are a HTML technique which allow to split one page into number of contained windows, each of which hold a separate HTML file.
- We can access separate window objects through window.frames[].
- Properties of frame:

Window Property	Description
frames[]	An array of all the frame objects contained by the current window.
length	The number of frames in the window. Should be the same value as **window.frames.length**.
name	The current name of the window. This is both readable and settable since JavaScript 1.1.
parent	A reference to the parent window.
self	A reference to the current window.
top	A reference to the top window. Often the top and the parent will be one and the same unless the <**frame**> tag loads documents containing more frames.
window	Another reference to the current window.

- To access a particular frame, we can use both its name and its position in array, so

parent.frames[0].name

⇓

print out the name of first frame.

- Frame :

```
<frameset>
  <frame src="frame1.html" name="frame1" id="frame_1">
  <frame src="frame2.html" name="frame2" id="frame_2">
  <frame src="frame3.html" name="frame3" id="frame_3">
</frameset>
```

- With iframes, we can add frame directly into document without using frameset.

```
<iframe src="google.com"> </iframe>.
```