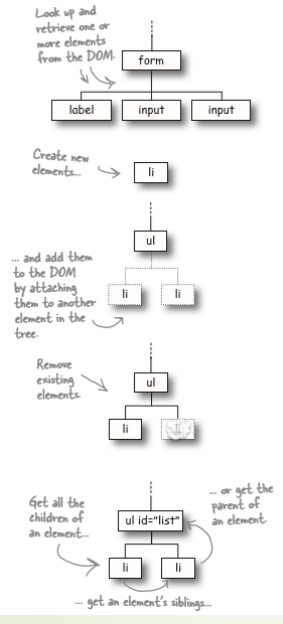
JavaScript DOM element



Get elements from the DOM.

Of course you already know this because we've been using document.getElementById, but there are other ways to get elements as well; in fact, you can use tag names, class names and attributes to retrieve not just one element, but a whole set of elements (say all elements in the class "on_sale"). And you can get form values the user has typed in, like the text of an input element.

Create and add elements to the DOM.

You can create new elements and you can also add those elements to the DOM. Of course, any changes you make to the DOM will show up immediately as the DOM is rendered by the browser (which is a good thing!).

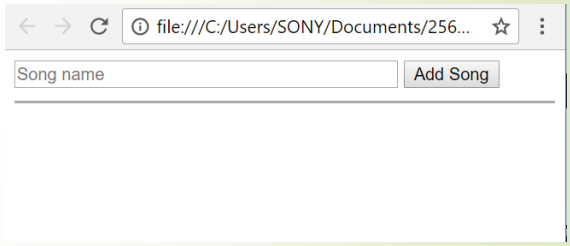
Remove elements from the DOM.

You can also remove elements from the DOM by taking a parent element and removing any of its children. Again, you'll see the element removed in your browser window as soon as it is deleted from the DOM.

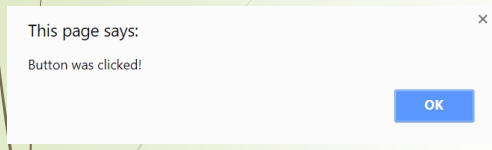
Traverse the elements in the DOM.

Once you have a handle to an element, you can find all its children, you can get its siblings (all the elements at the same level), and you can get its parent. The DOM is structured just like a family tree!

```html
<!doctype html>
<html lang="en">
<head>
<title>Webville Tunes</title>
<meta charset="utf-8">
<link rel="icon"
      type="image/ico"
      href="http://wickedlysmart.com/favicon.ico">
<script src="playlist.js"></script>
<script src="playlist_store.js"></script>
<link rel="stylesheet" href="playlist.css">
</head>
<body>

<form>
<input type="text" id="songTextInput" size="40" placeholder="Song name">
<input type="button" id="addButton" value="Add Song">
</form>

<ul id="playlist">

</ul>

</body>
</html>
```



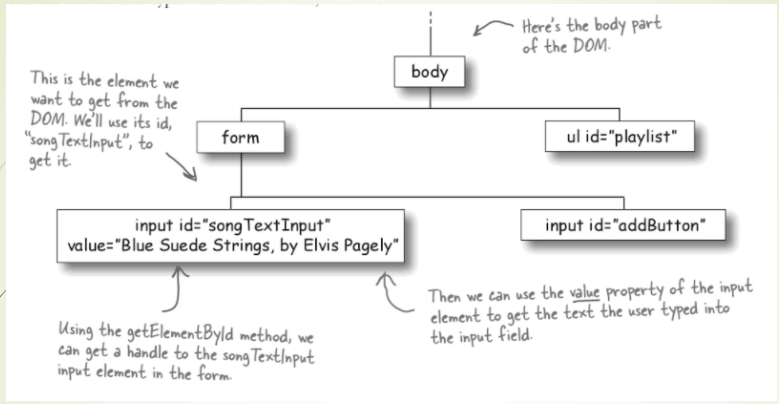

```
function handleButtonClick() {
    alert("Button was clicked!");
}
```




```
/* playlist.js */

window.onload = init;

function init() {
    var button = document.getElementById("addButton");
    button.onclick = handleButtonClick;


}
function handleButtonClick() {
    alert("Button was clicked!");
}
```

- A button click event is triggered when you click on a button in a web page.

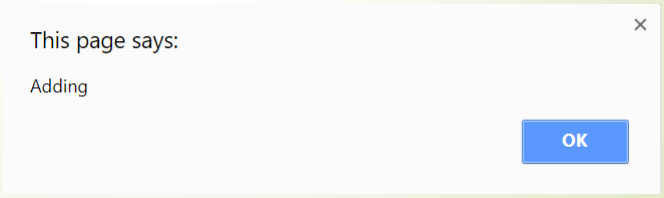
- You handle a button click event by registering a function to handle the event. You do this by writing a function, and setting the button's on click property to the function name.
-If a button click event handler is registered, that function will be called when you click on the button



- **To get the text a user has typed into a form input text field, you use the input's value property.**
- **If a user has not entered anything into a form input text field, the value of the field will be the empty string("").**
- **You can compare a variable to the empty string using an if test and ==**

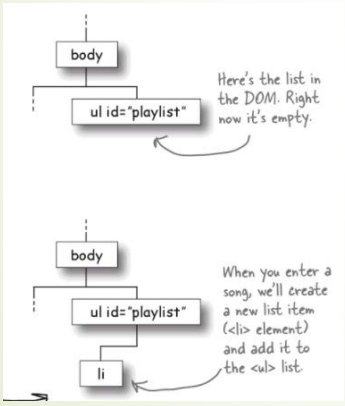
-You write code in the handler function to respond to the button clock event. You can alert the user or update the page or something else.

```
function handleButtonClick() {
    var textInput = document.getElementById("songTextInput");
    var songName = textInput.value;
    alert("Adding " + songName);
}
```

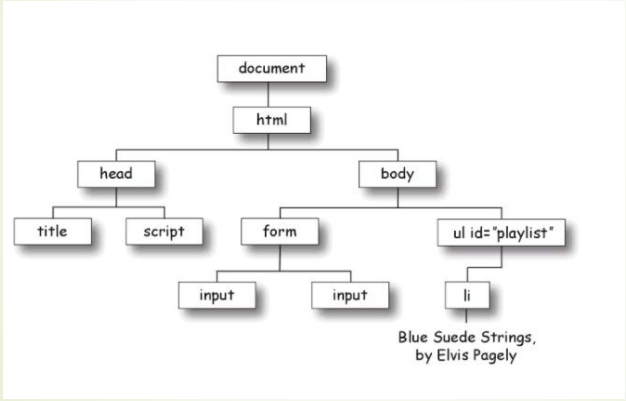


-You write code in the handler function to respond to the button clock event. You can alert the user or update the page or something else.

```
function handleButtonClick() {
    var textInput = document.getElementById("songTextInput");
    var songName = textInput.value;

    if (songName == "") {
        alert("Please enter a song");
    }
    else {
        alert("Adding " + songName);
    }
}
```



To add a new element to the DOM, you first need to create the element and then add it as a child of element



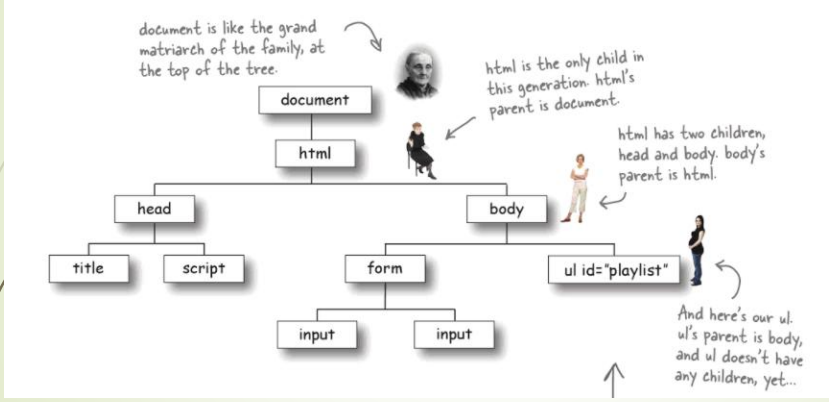
How to create element

Use document createElement to create a new element. Pass the tag name (e.g., "li") into the function call to indicate what element to create.

```
var li = document.createElement("li");
li.innerHTML = songName;
```

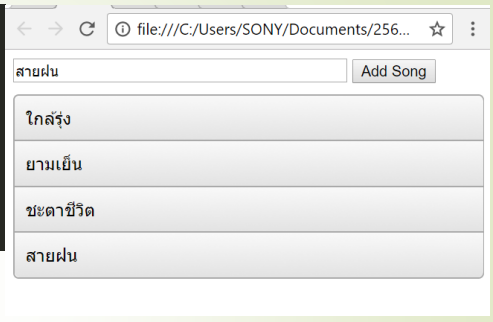
Adding an element to the DOM

To add element as a child of a parent element in the DOM, get a reference to the parent, and call appendChild on the parent, passing in the child element you're adding.




```
var ul = document.getElementById("playlist");
ul.appendChild(li);
```

```
function handleButtonClick() {
    var textInput = document.getElementById("songTextInput");
    var songName = textInput.value;
    var li = document.createElement("li");

        li.innerHTML = songName;

        var ul = document.getElementById("playlist");
        ul.appendChild(li);
}
```



If you add multiple children to a parent by using appendChild, each new child is appended after the other children, so they appear after or below the other children in the page (assuming you're not changing the layout with CSSS)

How to add class or id into child elements

- The setAttribute() method adds the specified attribute to an element, and gives it the specified value.

Syntax

```
element.setAttribute(attributename, attributevalue)
```