"소극적인 노력은 고생이다. 적극적인 노력은 훈련이다."

Lecture 3
Events, Handlers and All that Jazz: A Little Interaction

Samkeun Kim skim@hknu.ac.kr

http://cyber.hknu.ac.kr/

국립한경대학교
HANKYONG NATIONAL UNIV.

Get ready for Webville Tunes

How about a **playlist manager**

Calls it something original, like... hmm, say **Webville Tunes**

Getting started...

Let's just create an **HTML5 document** with a form and a list element to hold the playlist:

```
<!doctype html>
<html lang="en">
<head>
  <title>Webville Tunes</title>
  <meta charset="utf-8">
  <script src="playlist.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>
```

Just your standard HTML5 head and body.

We're going to be putting all our JavaScript in the playlist.js file.

We've included a stylesheet to give our playlist app a nice look & feel.*

All we need is a simple form. Here it is with a text field to type in your songs. We're using the HTML5 placeholder attribute that shows an example of what to type in the input field.

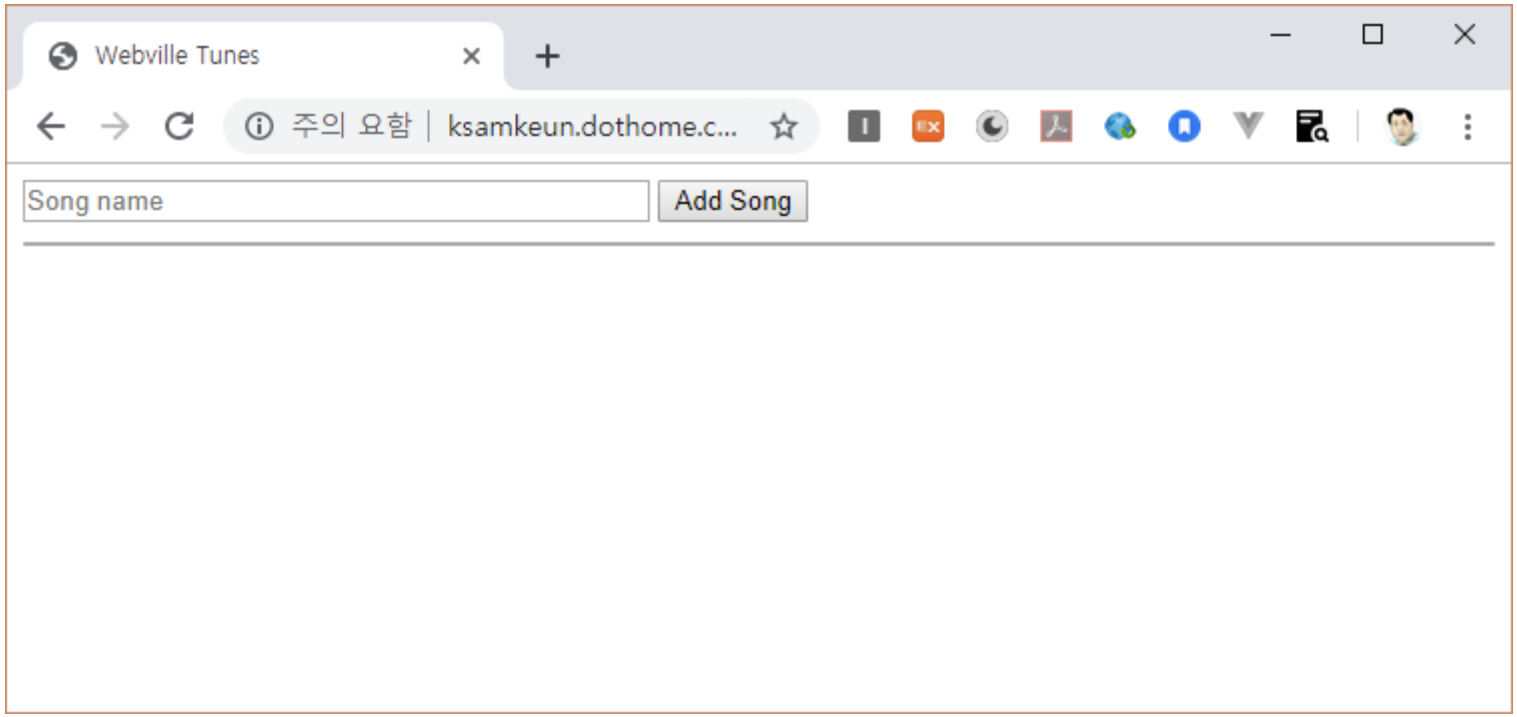
And we've got a button with an id of "addButton" to submit your new additions to the playlist.

We're going to use a list for the songs. For now it's empty, but we'll change that with JavaScript code in a sec...

실습과제 3-1 Give it a test drive

Go ahead and type in the code above, load it into your favorite browser and give it a spin before moving on to the next page.

Download: http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch3/playlist.css



http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch3/_playlist.html

But nothing happens when I click “Add Song”

버튼을 클릭 할 때 JavaScript code가 어떻게 호출되게 할 것인가?

We need two things:

1. 사용자가 "**Add Song**" 버튼을 클릭하면 수행될 약간의 자바스크립트 코드
2. 버튼이 클릭되었을 때 자바스크립트가 "Add Song" 코드를 실행해야 하는 걸 알 수 있도록 연결해 주는 방법 필요

Handling Events

웹 페이지가 디스플레이 되는 동안 Browser에서는 많은 일들이 벌어지고 있다:

- 버튼이 클릭되었다.
- 네트워크상으로부터 요청한 데이터가 도착되었다.
- 타이머가 시작되었다.

. . .

⇒ 위의 모든 행위는 **이벤트**(event)를 발생시킨다:
a button click event, a data available event, a time expired event, and so on (there are many more).

⇒ 이벤트가 발생할 때마다 자바스크립트 코드는 그 이벤트를 제어할 기회를 갖는다.
즉, **이벤트가 발생하면 호출될 코드를 제공할 기회를 갖게 된다.**

Handling Events

이벤트가 발생할 때 뭔가 흥미로운 것이 발생하기를 원한다면 그것을 **제어할 코드**가 필요하다:

- 버튼 클릭 이벤트가 발생했을 때 playlist에 새로운 song을 추가하고 싶다
- 새로운 데이터가 도착했을 때 그것을 처리하여 웹 페이지에 디스플레이하고 싶다 등

Making a Plan...

목적: "Add Song"을 클릭하면 playlist에 노래 추가

1. Set up a handler to handle the user's click on the "**Add Song**" button.
2. Write the handler to get the song name the user typed in, and then...
3. Create a new element to hold the new song, and...
4. Add the element to the page's DOM.

이제 **playlist.js** 코드를 작성해 보자.

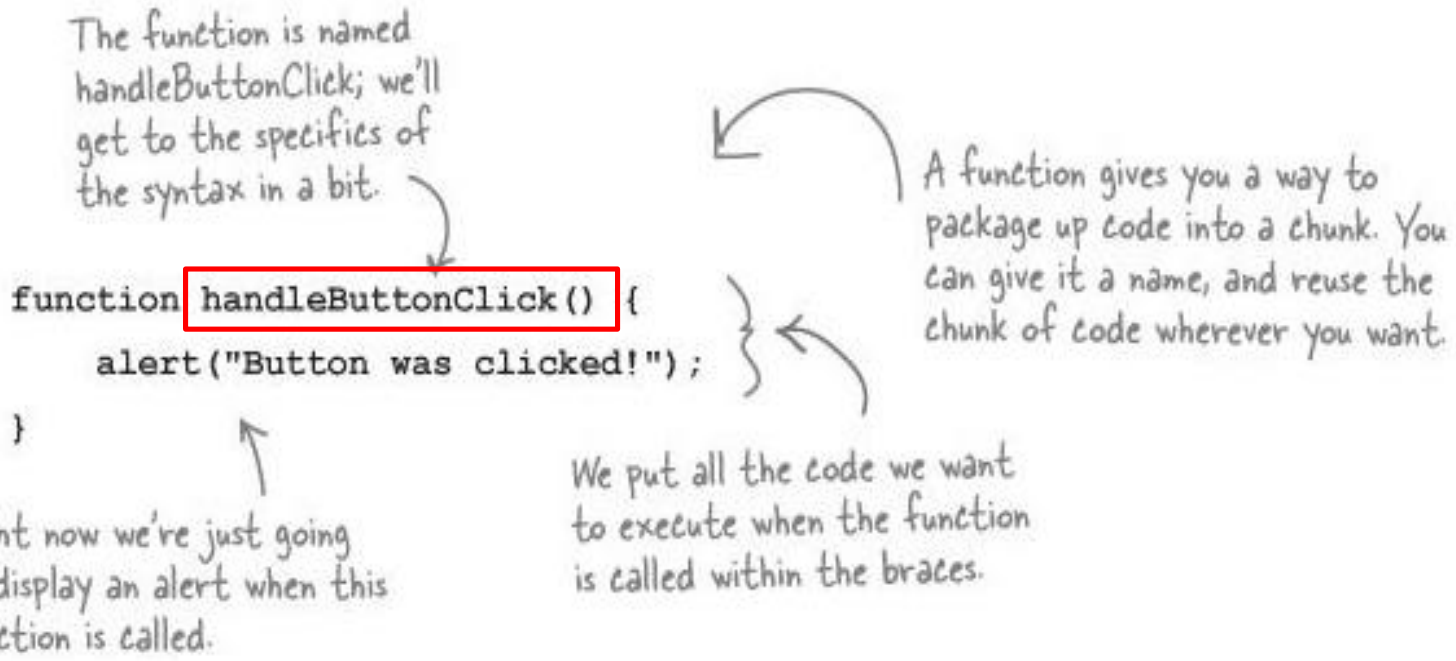
Getting access to the "Add Song" button

버튼에게 클릭 이벤트가 발생되면 알려 달라고 요청하기 위해서는 먼저 버튼에 **접근**(access)할 필요가 있다.

- 버튼에 대한 레퍼런스(주소)를 얻기 위해 **getElementById**를 사용한다:

  var button = document.**getElementById**("addButton");

- 이제 버튼에게 **클릭이 발생했을 때 호출할 코드**를 제공해야 한다:
  이를 위해 이벤트를 처리할 handleButtonClick이라는 함수(function) 생성!



Giving the button a click handler

1. Set up a handler to handle the user's click
2. Write the handler to get the song name
3. Create a new element to hold the new song
4. Add the element to the page's DOM

버튼과 이벤트 핸들러 함수(handleButtonClick)를 얻었다.

이들을 함께 두기 위해서는 onclick이라는 버튼의 프로퍼티를 사용한다.

Sets the onclick property like this:

```
var button = document.getElementById("addButton");
button.onclick = handleButtonClick;
```

With a button in hand, after calling getElementById, we set the onclick property to the function we want called when a click event occurs.

window.onload 프로퍼티와 유사! 버튼이 클릭되면 함수를 호출한다.

Now let's put all of this together:

Just like we did in the last chapter, we're using an init function that won't be called and executed until the page is fully loaded.

```
window.onload = init;
function init() {
    var button = document.getElementById("addButton");
    button.onclick = handleButtonClick;
}
function handleButtonClick() {
    alert("Button was clicked!");
}
```

After the page loads we'll grab the button and set up its onclick handler.

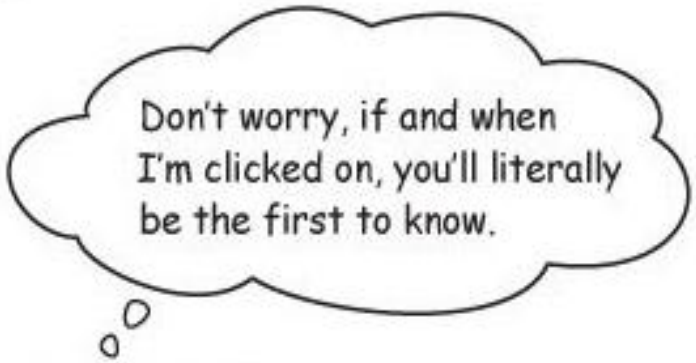
And the click handler will display an alert when we click on the button.

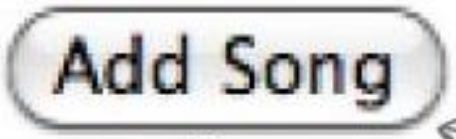
A closer look at what just happened...

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

We set up the button click event handler in the init function (that is, after the page has finished loading).

The button object has an onclick property that we set to the handleButtonClick function.





When the user clicks on the button, the "click" event is triggered, and the handleButtonClick function is called.

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


Time to wake up, there's a click from the user.
I see I have a handler for this, better let him know.
Add Song
Yes! Somone clicked on the button. I get to run the handleButtonClick function.
function handleButtonClick() {
alert("Button was clicked!");
}
I was asked to alert you that the button was clicked... I know, for an alert dialog that's a little underwhelming, but anyway, just doing my job.
http://localhost
Button was clicked!
OK

Getting the song name

1. Set up a handler to handle the user's click
2. Write the handler to get the song name
3. Create a new element to hold the new song
4. Add the element to the page's DOM

The **second** step of our task:

사용자가 입력한 노래명을 얻어 온다.

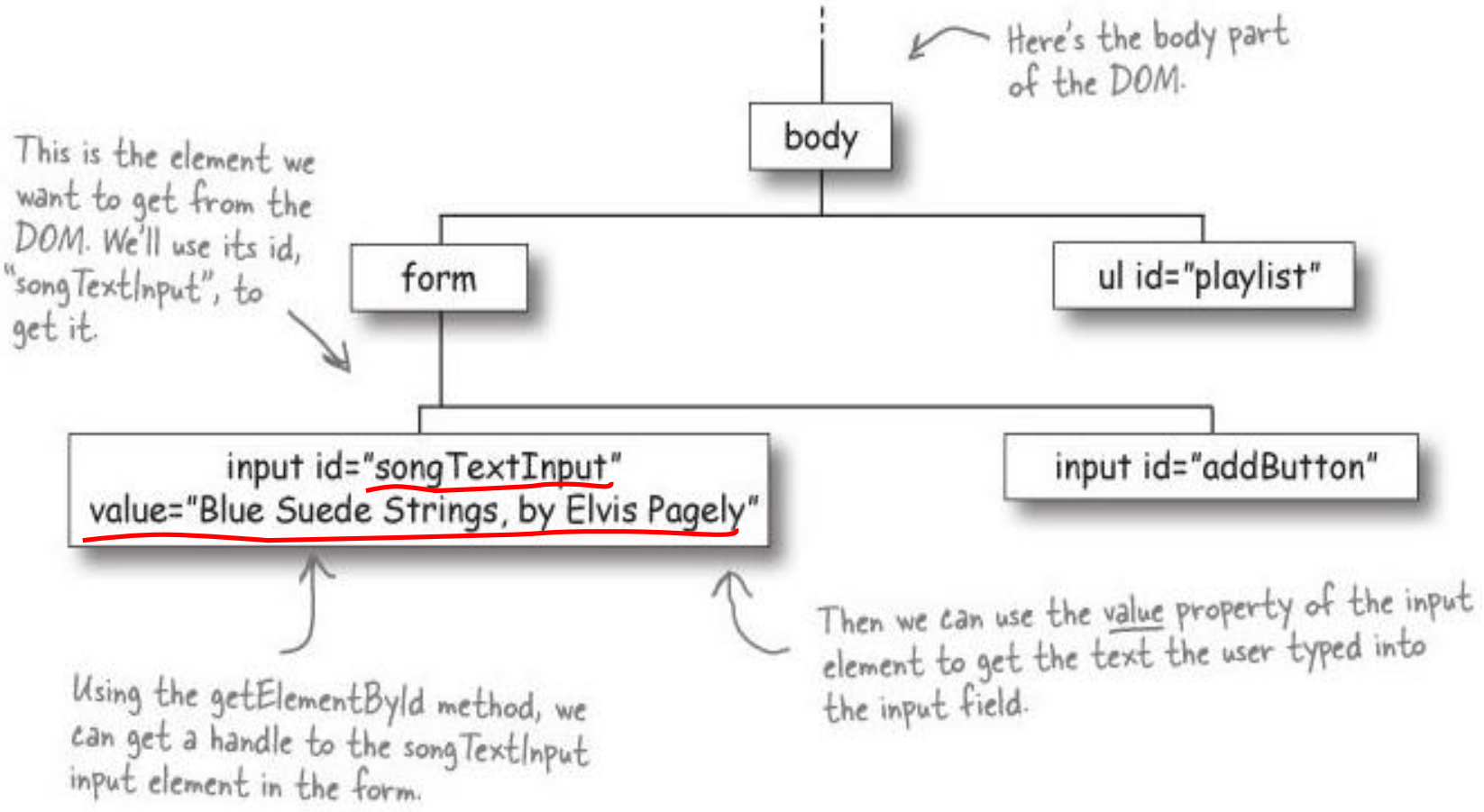
- 노래명을 어떻게 얻어올 것인가?
- 노래명은 브라우저 폼에 사용자가 타이핑한 것이다

웹 페이지에서 발생한 모든 일은 DOM에 반영되고,

따라서 사용자가 타이핑한 내용 또한 웹 페이지에 나타나게 된다.

웹 페이지의 input element로부터 텍스트를 얻어 오기 위해서는 먼저 DOM으로부터 input element를 얻어와야 한다: getElementById

사용자가 폼 필드에 타이핑한 텍스트에 접근하기 위해 input element의 **value 프로퍼티**를 사용할 수 있다:



Sharpen Your Pencil

Rework the **handleButtonClick** function below to obtain the name of the song the user has typed into the form input element.

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

How do we add a song to the page?

1. Set up a handler to handle the user's click
2. Write the handler to get the song name
3. Create a new element to hold the new song
4. Add the element to the page's DOM

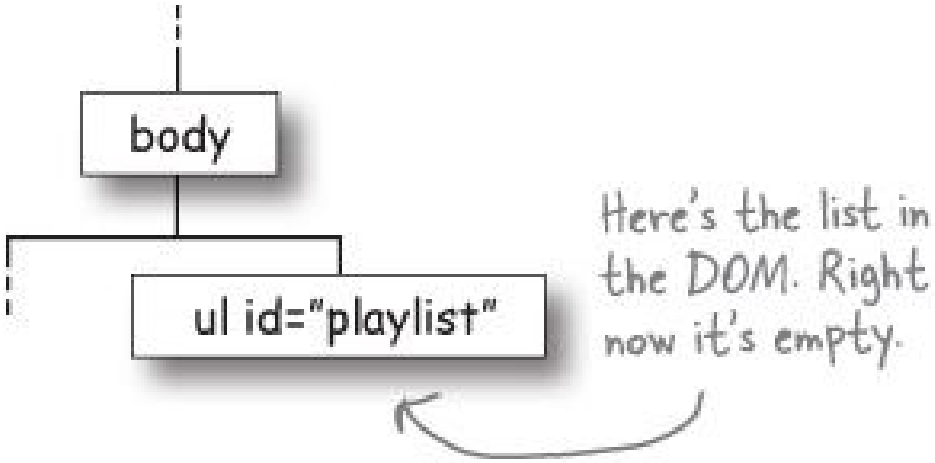
Can **type** a song name into a form, **click** the Add Song button and **get** the text you typed into the form, all within your code

Now we're going to **display** the playlist on the page itself. Here's what it's going to look like:

Here's what we need to do:

1. **Put an empty list in the HTML markup**
   (an empty <ul> element to be exact) back when we first typed it in.

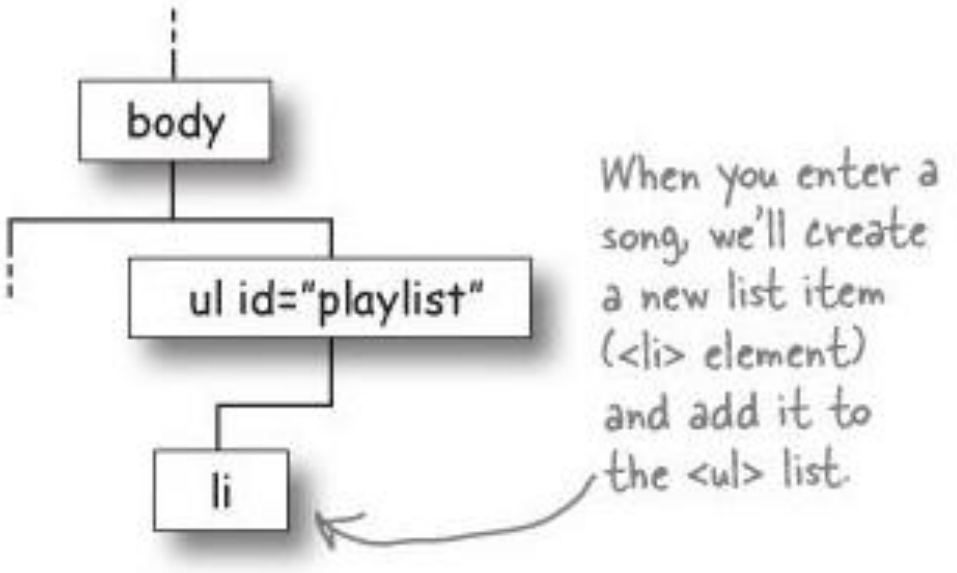
   DOM looks like:



Here's what we need to do:

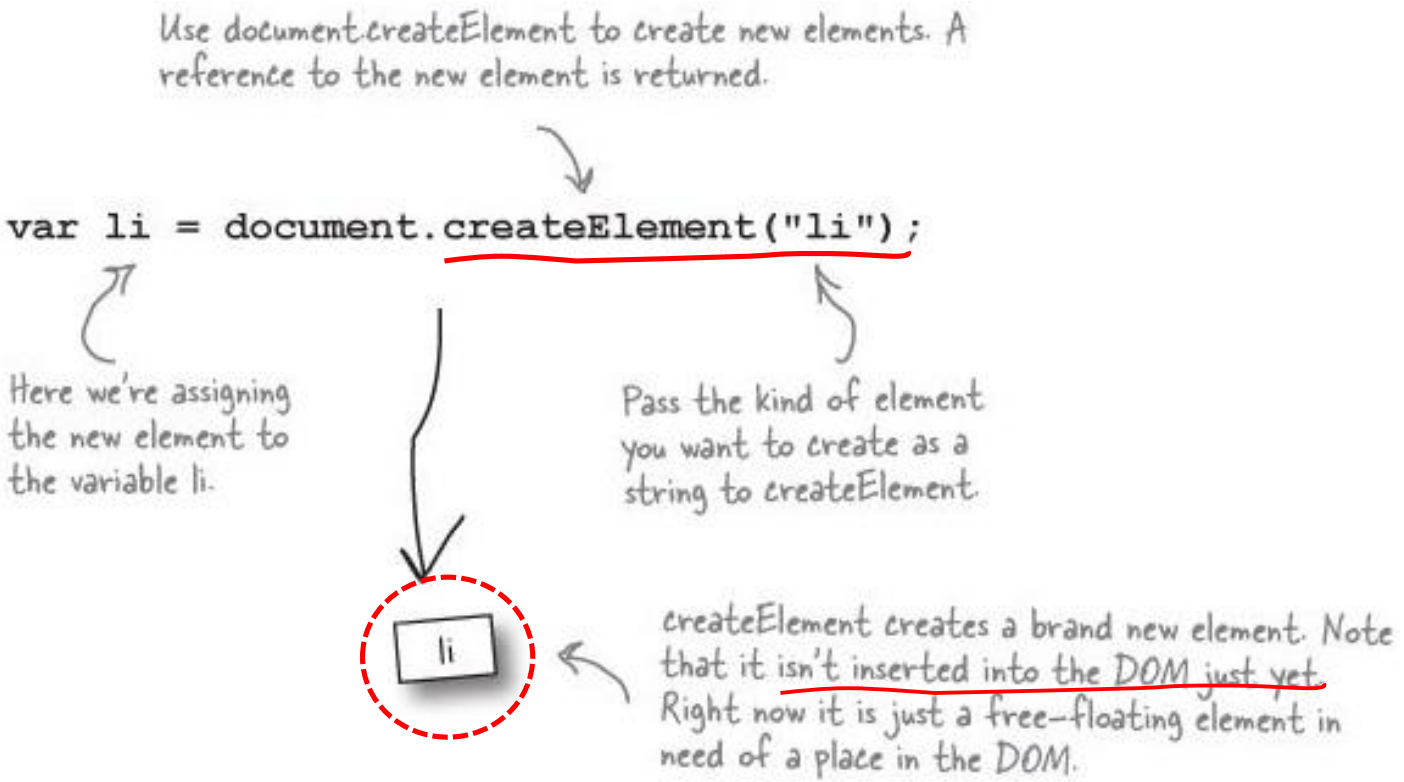
2. **Every time we enter a new song, we want to add a new item to the unordered list.**

   - ✓ Create a new **<li>** element that will hold the song name
   - ✓ Take the new **<li>** element and add it to the **<ul>** in the DOM



How to create a new element

Create a <li> element



How to create a new element

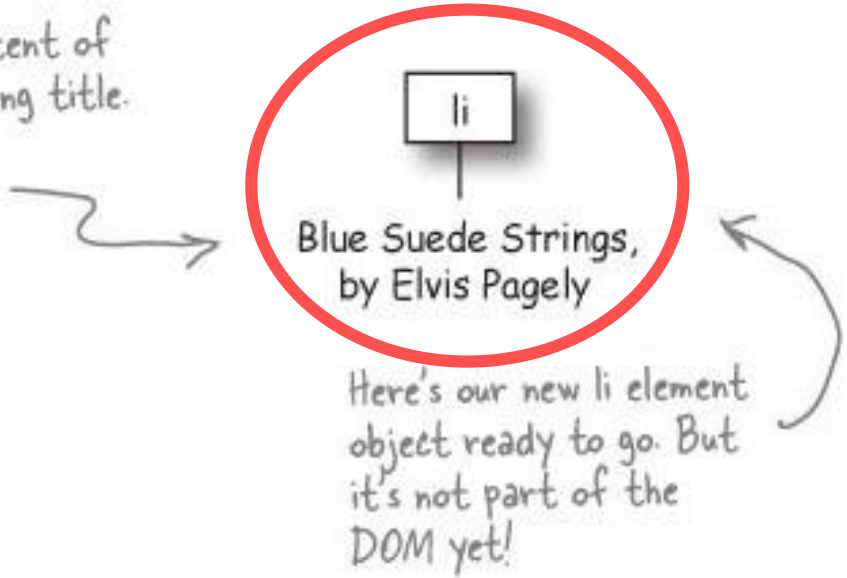
Have a **new <li> element** with nothing in it.

Knows one way to get text into an element:

```
li.innerHTML = songName;
```

Our li variable.

This sets the content of the <li> to the song title.



Adding an element to the DOM

1. **Set up a handler to handle the user's click**
2. **Write the handler to get the song name**
3. **Create a new element to hold the new song**
4. **Add the element to the page's DOM**

To **add** a new element to the DOM:
have to know where you want to put it.

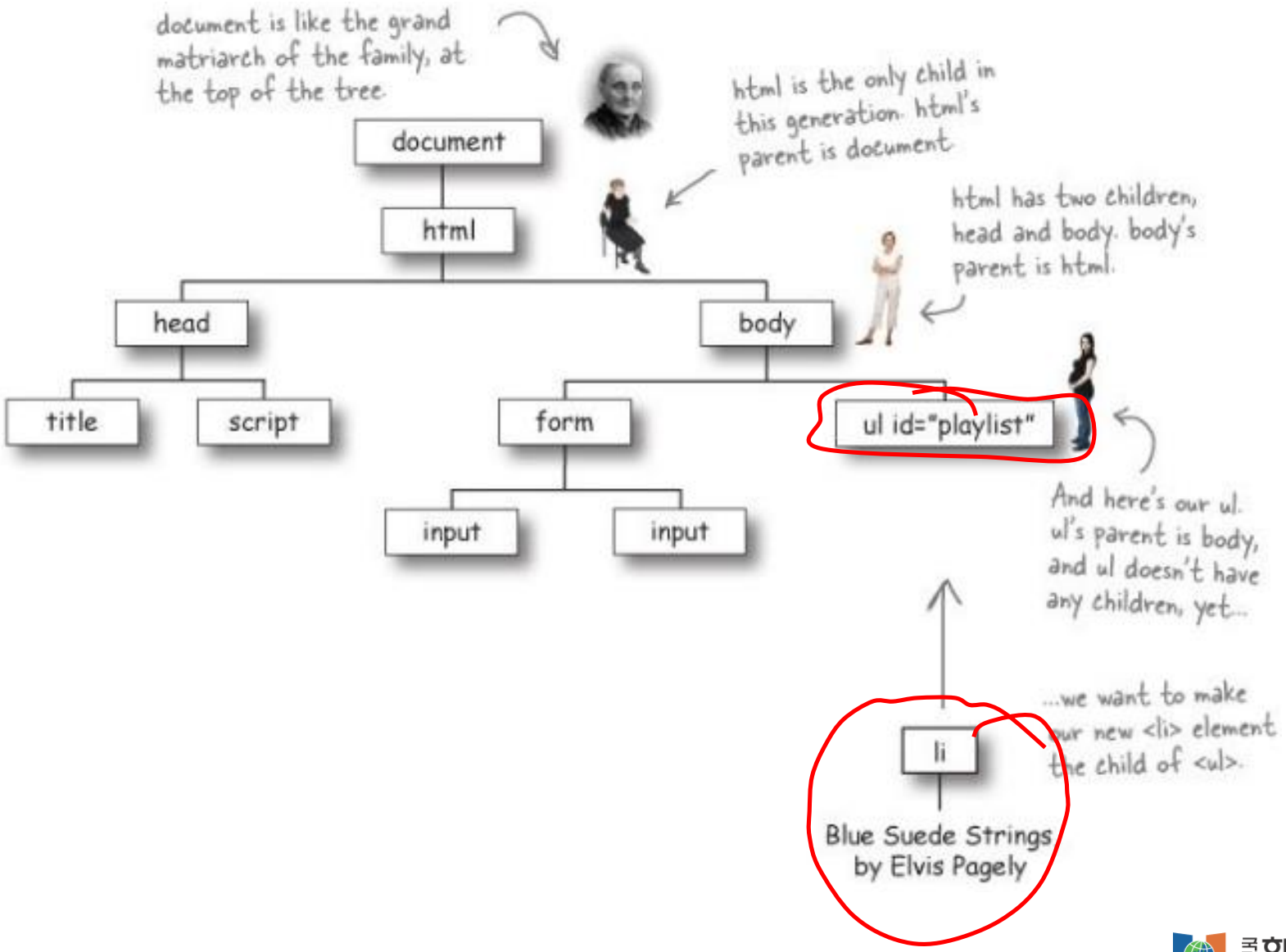
Put the <li> element in the <ul> element.

But how do we do that?

Let's take another look at the DOM.
Remember how we said it was like a tree?

Think family tree:

Adding an element to the DOM

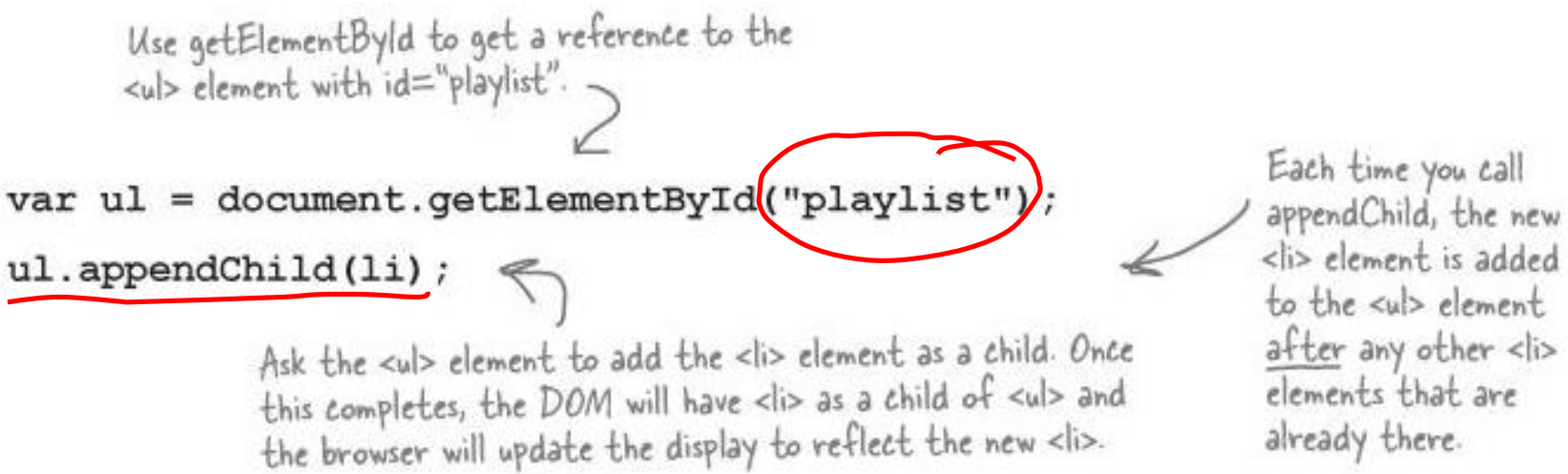


Adding an element to the DOM

 엘리먼트를 트리에 추가하기 위해서:
   엘리먼트의 자식 노드로 만들면 된다

To do that, we first need to **find** the <ul> element in the tree and then to **add** the <li>, we **tell** the <ul> element to add a new child to itself.

Here's how we do that:



Put it all together...

Let's put all that code together and **add it to the handleButtonClick function.**

Go ahead and type it in if you haven't already so you can test it.

```
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);

}
```

First, create the new <li> element where the song name is going to go.

Then, set the content of that element to the song name.

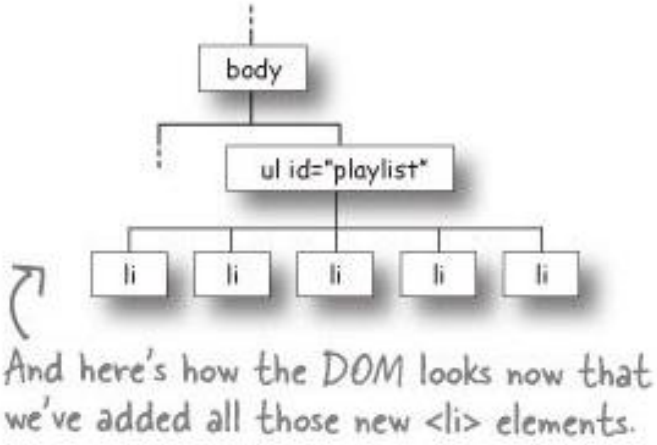
The <ul> with the id "playlist" is the parent element for our new <li>. So we get that next.

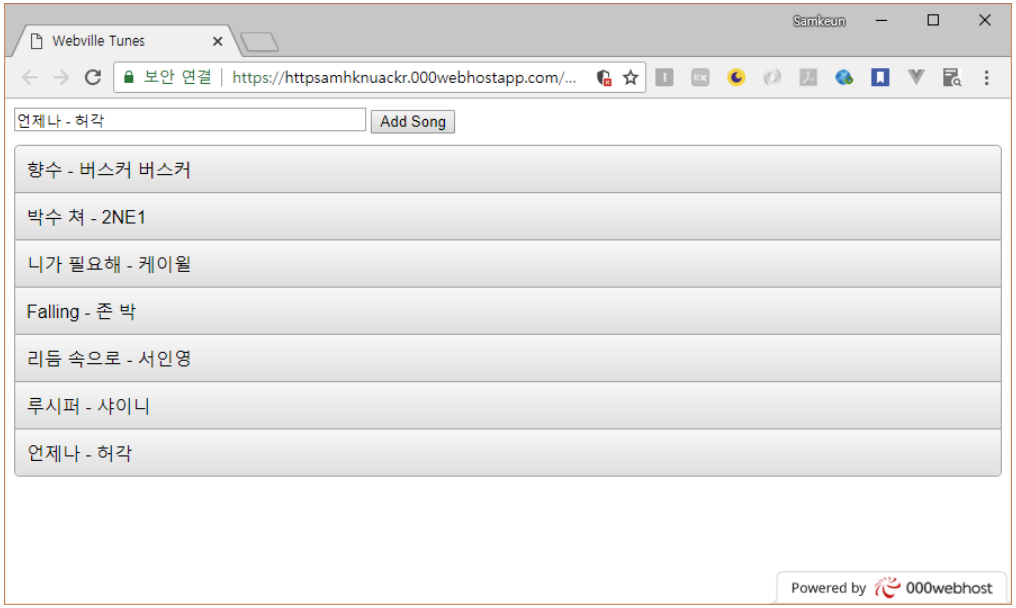
Then we add the li object to the ul using appendChild.

Notice that we ask the parent element, ul, to add li as a new child.

실습과제 3-2

playlist.html 을 완성하시오.





http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch3/playlist.html

Wait a sec, I get we're interacting with the DOM and all, but how is this a real web App? If I close my browser, all my songs are gone. Shouldn't my playlist items stick around if this is really an application?

Samkeun

Webville Tunes

skimok77.dothome.co.kr/hfhtml5/ch3/playlist.htr

Song name | Add Song

F5 click!!

힘들게 입력했던 노래 제목들은 어디에?

READY BAKE CODE

How to add the Ready Bake Code...

All you have to do is make a new file, **playlist_store.js**, type in the code below, and then make a couple of changes to your existing code.

```
function save(item) {
    var playlistArray = getStoreArray("playlist");
    playlistArray.push(item);
    localStorage.setItem("playlist", JSON.stringify(playlistArray));
}

function loadPlaylist() {
    var playlistArray = getSavedSongs();
    var ul = document.getElementById("playlist");
    if (playlistArray != null) {
        for (var i = 0; i < playlistArray.length; i++) {
            var li = document.createElement("li");
            li.innerHTML = playlistArray[i];
            ul.appendChild(li);
        }
    }
}

function getSavedSongs() {
    return getStoreArray("playlist");
}

function getStoreArray(key) {
    var playlistArray = localStorage.getItem(key);
    if (playlistArray == null || playlistArray == "") {
        playlistArray = new Array();
    }
    else {
        playlistArray = JSON.parse(playlistArray);
    }
    return playlistArray;
}
```

Type this into "playlist_store.js".

Integrating your Ready Bake Code

First, add a reference to **playlist_store.js** in your <head> element in **playlist.html**:

```
<script src="playlist_store.js"></script>
<script src="playlist.js"></script>
```

Add this just above your link to playlist.js. It loads the Ready Bake code.

Add two lines to your code, in **playlist.js**, that will load and save the playlist:

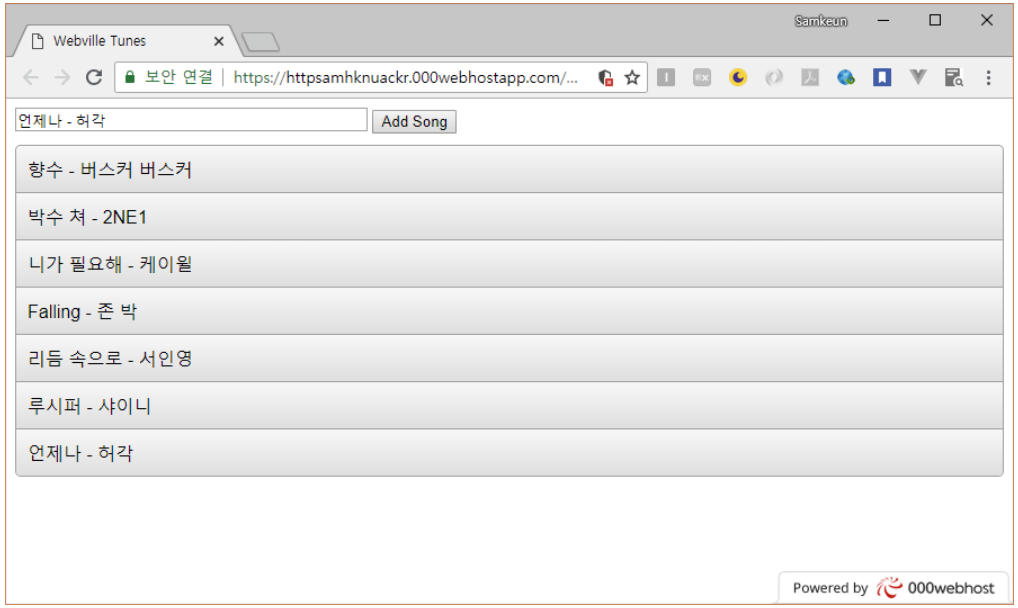
```
function init() {
    var button = document.getElementById("addButton");
    button.onclick = handleButtonClick;
    loadPlaylist();
}

function handleButtonClick() {
    var textInput = document.getElementById("songTextInput");
    var songName = textInput.value;
    var li = document.createElement("li");
    li.innerHTML = songName;
    var ul = document.getElementById("list");
    ul.appendChild(li);
    save(songName);
}
```

This loads the saved songs from localStorage when you load your page, so you see your saved songs.

And this saves your song each time you add one to the playlist

Test drive the saved songs

Okay, reload the page and type in some songs. Quit the browser. Open the browser and load the page again.



http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch3/playlist2.html

주의)

저장된 내용이 보이지 않을 경우 **F12** 키를 클릭하면 하단에 브라우저 모드 선택 창 Open -> 브라우저 선택을 '호환성 보기'가 아닌 것으로 선택한다.

실습과제 3-3

Ready Bake code **(playlist_store.js)**를 이용하는 playlist2.html을 완성하고 자신이 좋아하는 노래 10곡 이상을 입력한 후 브라우저 창에 나타내기

실습과제 3-4

playlist2.html에서 Clear시키는 버튼(예: Clear All)을 추가하고,

playlist_clear.js에 Clear All 버튼이 클릭되었을 때 입력된 모든 노래 항목을 삭제하는 기능을 추가하시오.

실습과제 #3부터는 실행화면 캡처 시 브라우저 URL이 아래 처럼 표시되도록 dothome 서버에 업로드하여 실행하시오:

http://YOUR-ID.dothome.co.kr/~

Q & A