Lecture 15
Storing Things Locally: Web Storage

Samkeun Kim <skim@hknu.ac.kr>

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

국립한경대학교
HANKYONG NATIONAL UNIV.

How browser storage works (1995 – 2010)

쇼핑 카트를 만들고 있는가? 그렇다면 사용자 선호도(user preferences)를 **저장**할 필요가 있을 것이다.

또는 단순히 각 사용자와 연관된 데이터를 잘 보관해 둘 필요가 있을 수 있다.

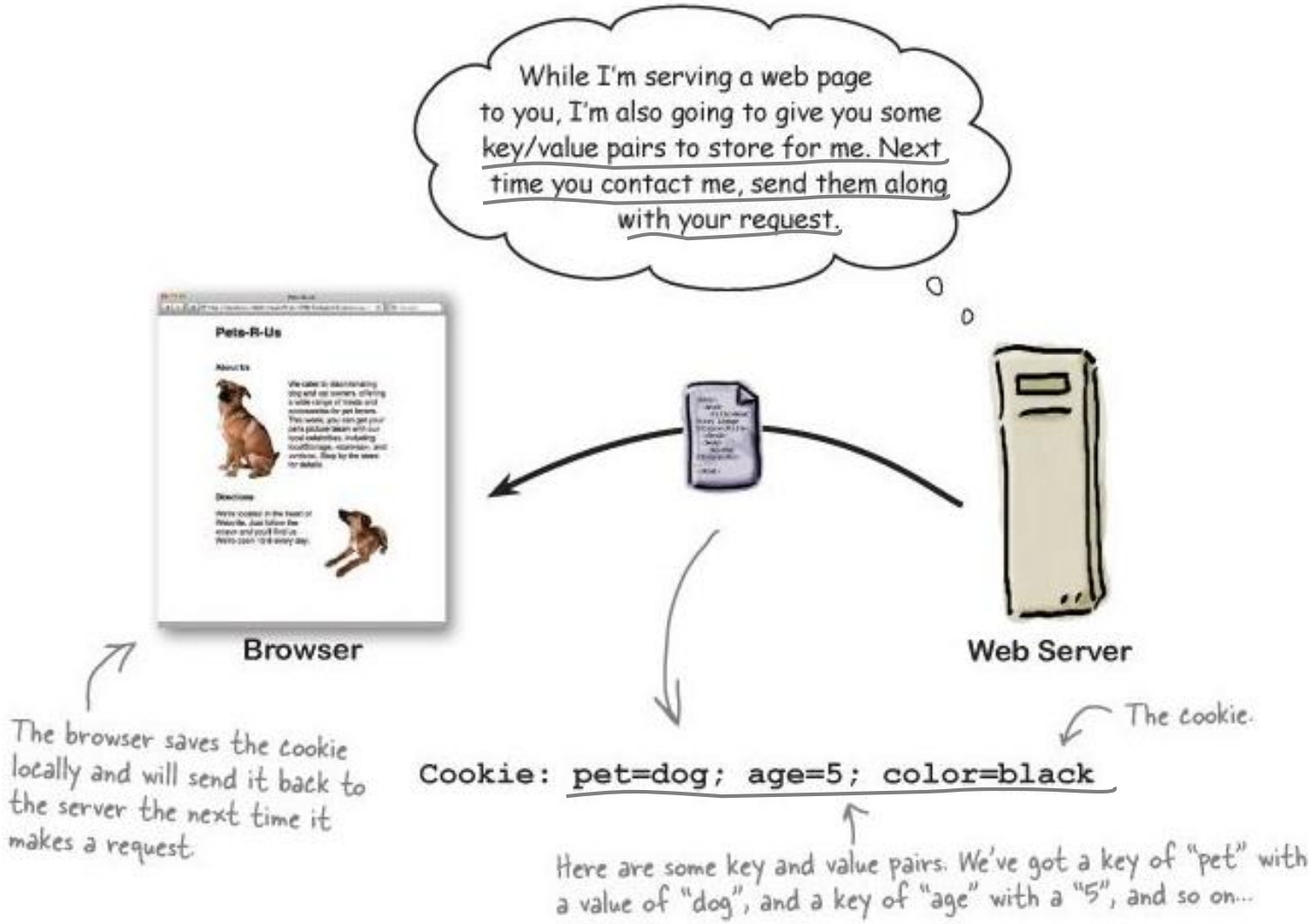
이것이 바로 브라우저의 **저장공간이 필요한 이유**다.

브라우저 스토리지는 **웹 경험**(web experience)을 구축하는데 사용할 수 있는 데이터를 지속적으로 저장시킬 방법을 제공한다.

Behind the Scenes

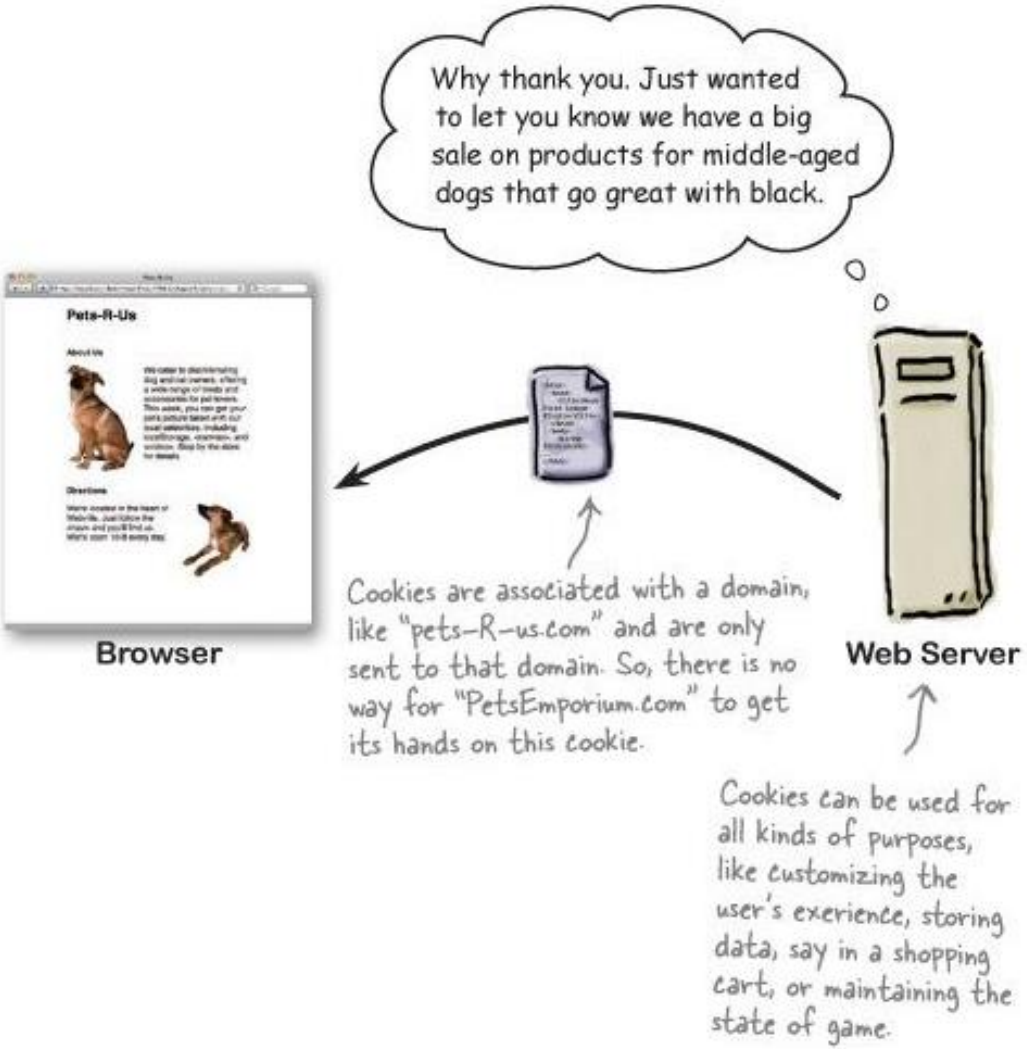
지금까지는 마을에 **브라우저 쿠키**(browser cookie)라는 하나의 게임만 있었다.
먼저 쿠키가 어떻게 동작하는 지 살펴보자.

1. 브라우저가 웹 페이지를 검색할 때 **서버**(예: pets-R-us.com)는 응답과 함께 쿠키를 보낼 수 있다: 쿠키는 하나 이상의 **key/value 쌍**을 포함하고 있다.

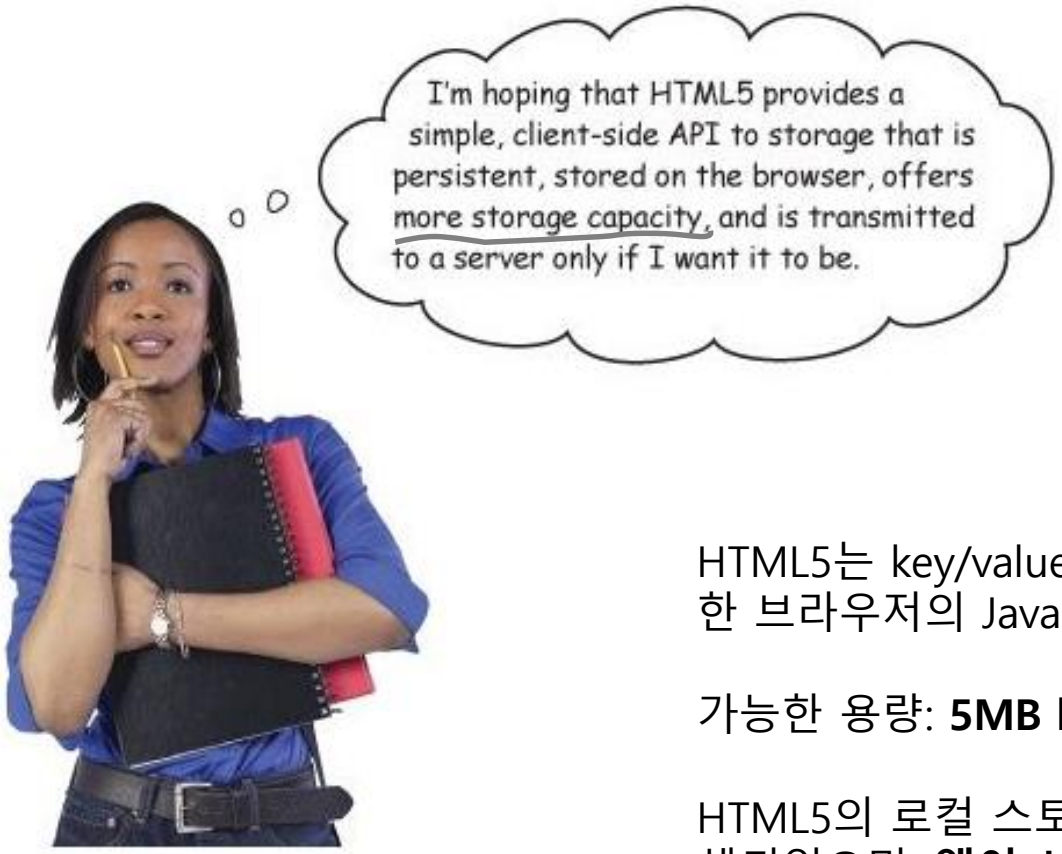


2. 다음 번에 **브라우저가** 같은 서버(pets-R-us.com)에 다시 요청할 때 이전에 보내졌던 쿠키와 함께 보낸다.

3. 서버는 쿠키를 이용하여 **사용자 경험을 개인화**시킬 수 있다: 예를 들어, 관련된 항목들을 **프로모션** 한다든지 등의 다양한 방법으로 **사용자경험을 제공**할 수 있다.



How HTML5 Web Storage works



HTML5는 key/value 쌍을 지속적으로 저장하기 위한 브라우저의 JavaScript API를 제공한다.

가능한 용량: **5MB limit**

HTML5의 로컬 스토리지는 웹앱을 염두에 두고 탄생되었으며, **앱이 서버와의 통신을 줄일 수 있도록** 데이터를 저장할 수 있게 해준다.

1 A page can store one or more key/value pairs in the browser's local storage.

2 And then later use a key to retrieve its corresponding value.

Note to self…

지금까지는 "해야 할 일"을 포스트잇에 적어서 벽에 붙여 두었다가 그 일이 완료되면 그 포스트잇(일명 **stickies**)을 떼어내어 쓰레기통에 버리는 방법을 사용해 왔다.

How about we build one using HTML?

그러기 위해서는 **모든 stickies를 저장할 방법**이 필요하다.

그래서 **서버**가 필요해지고, 약간의 **쿠키**도 . . .

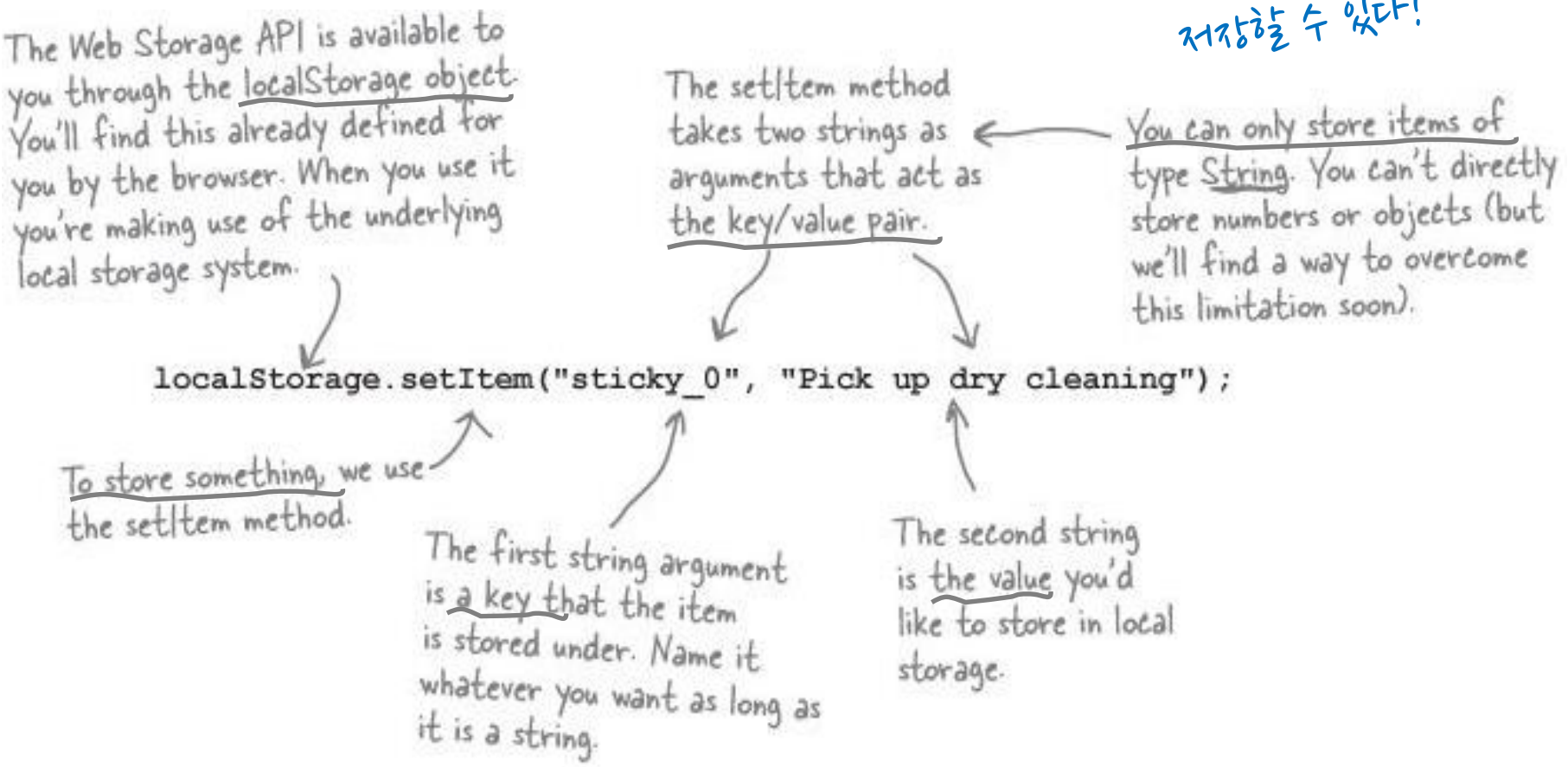
⇒ Can do this with the HTML5 Web Storage API!

Exercise

Local store에 대해 살펴보자:

1. 작성할 내용이 그렇게 많지는 않다.

String 타입의 항목들만 저장할 수 있다!



2. 너무 쉽다. 두 번째 항목을 추가해보자.

```
localStorage.setItem("sticky_1", "Cancel cable tv, who needs it now?");
```

Another key. Like we said already, you can use any key you like as long as it is a string, but you can only store one value per key.

A value to go with our new key.

3. 이제 **key**를 이용하여 **localStorage**로부터 해당되는 값을 꺼내올 수 있다:

We're getting the value associated with the key "sticky_0" from the local store...

...and assigning it to the variable named sticky.

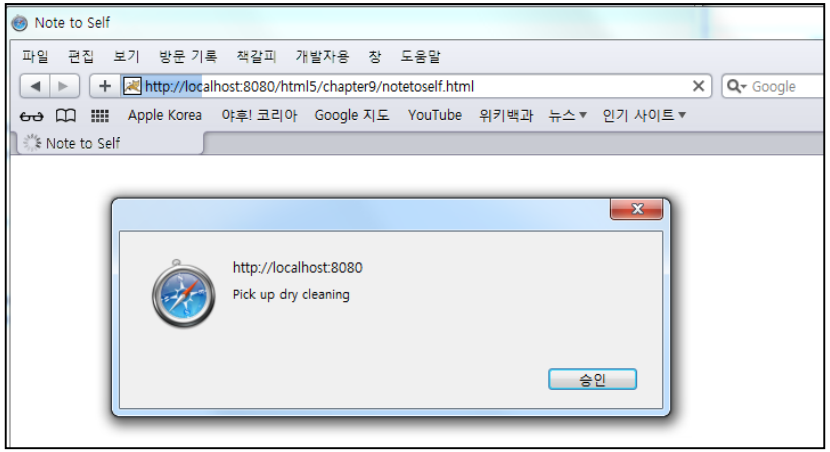
```
var sticky = localStorage.getItem("sticky_0");

alert(sticky);
```

And to make this a little more interesting, let's use the alert function to pop the sticky note's value up on the screen.

실습과제 15-1 Time for a test drive!

notetoself.html

```
1  <!doctype html>
2  <html>
3  <head>
4  <title>Note to Self</title>
5  <meta charset="utf-8">
6  <link rel="stylesheet" href="data.css">
7  <script>
8      localStorage.setItem("sticky_0", "Pick up dry cleaning");
9      localStorage.setItem("sticky_1", "Cancel cable tv, who needs it now?");
10     var stickyValue = localStorage.getItem("sticky_0");
11     alert(stickyValue);
12 </script>
13 </head>
14 <body>
15
16 </body>
17 </html>
```

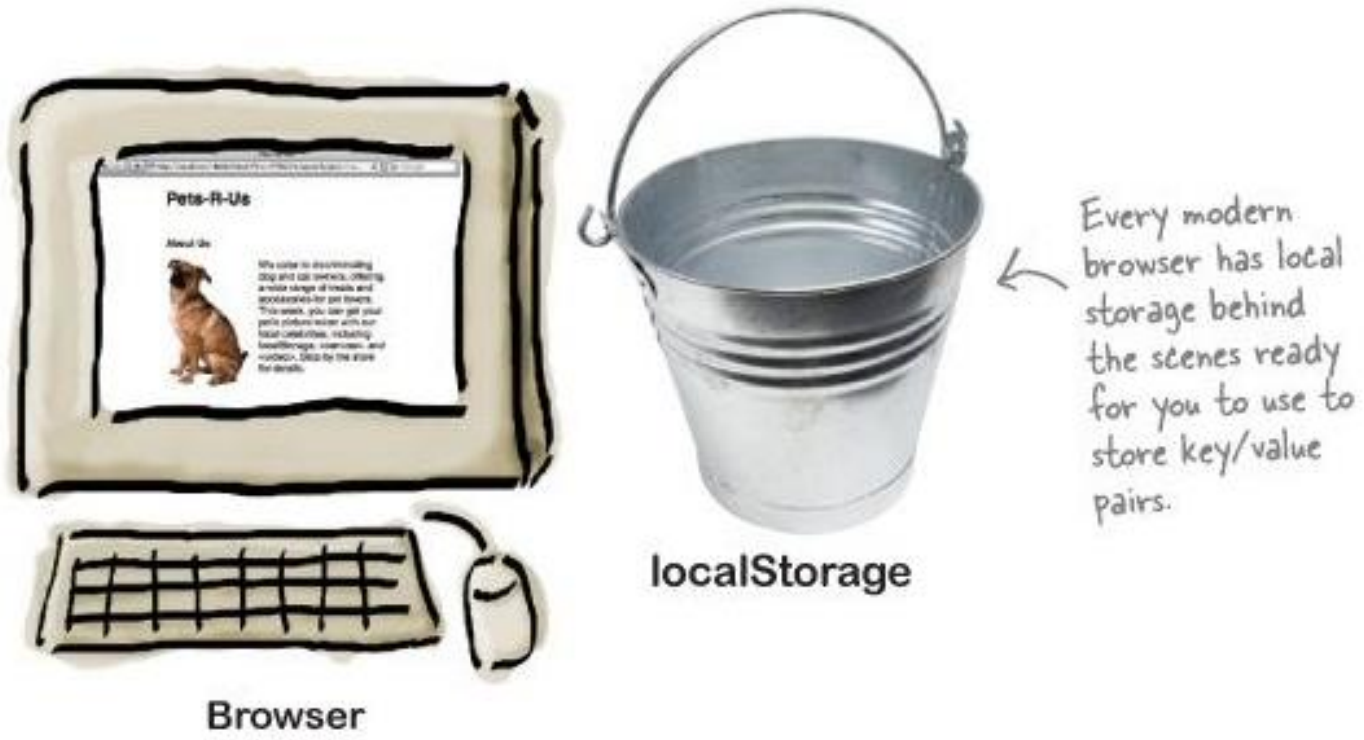


http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch9/notetoself.html

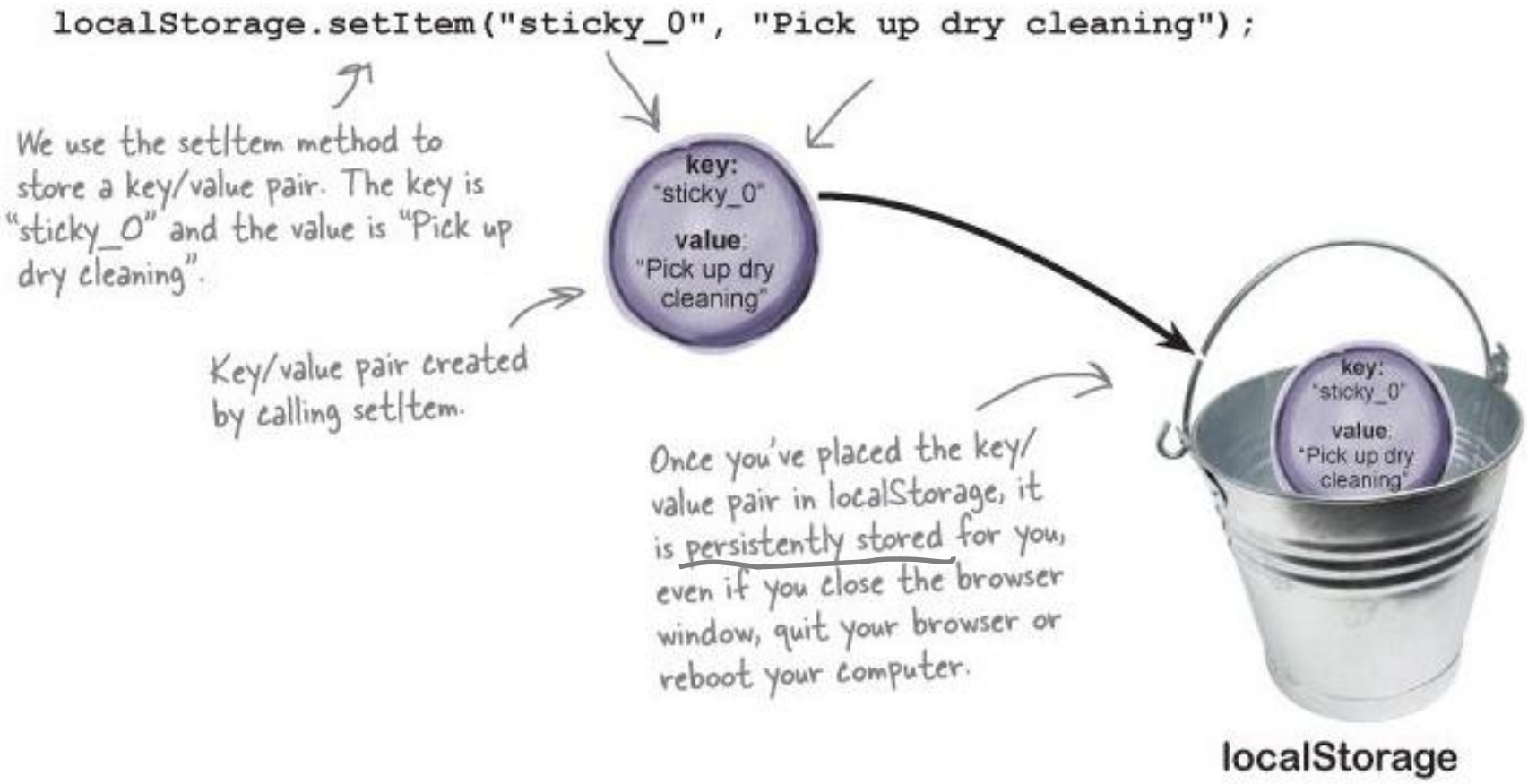
Time for a test drive!

지금까지 개략적인 내용을 파악했다면 이제 세부적으로 파헤쳐보자:

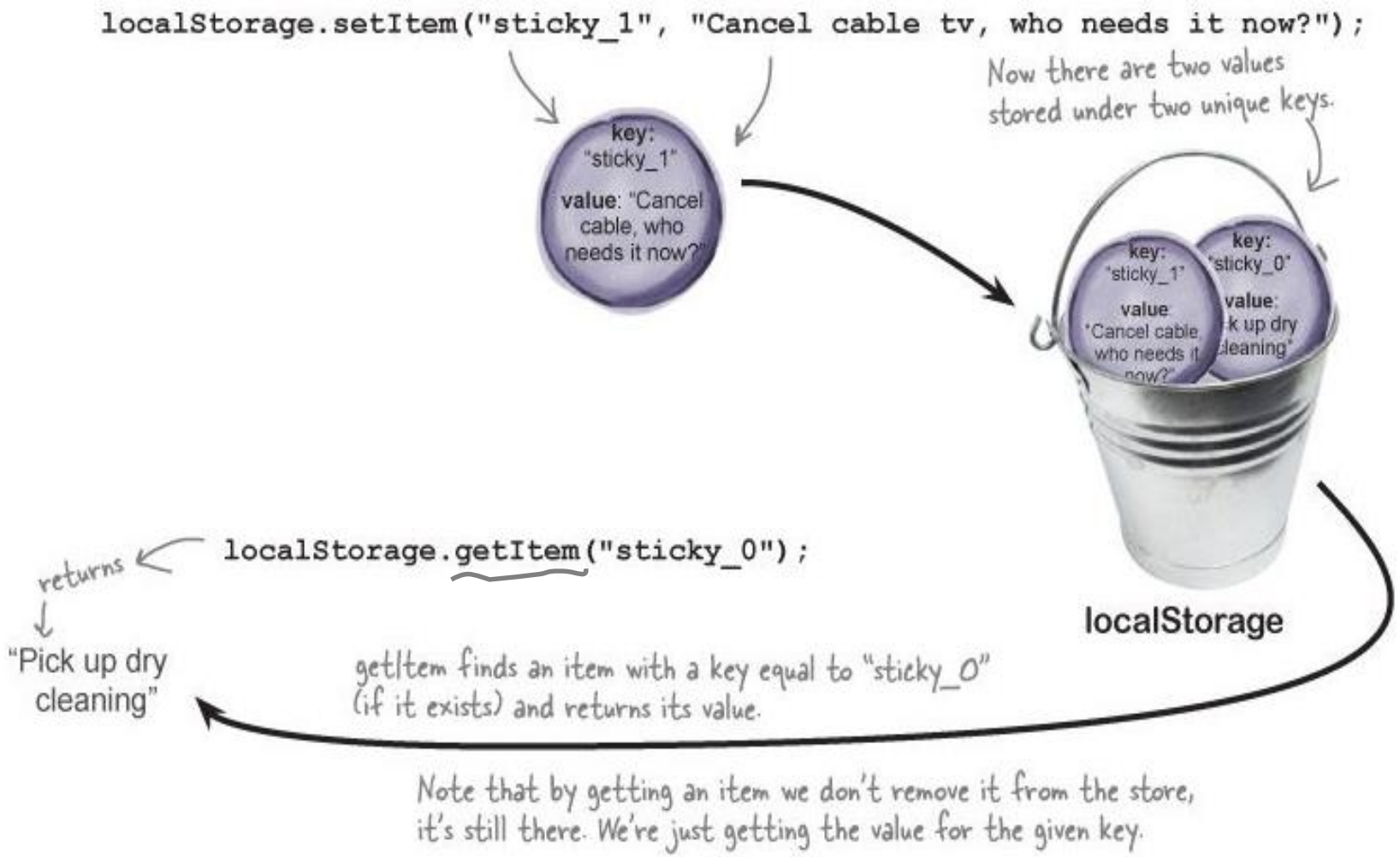
1. 모든 브라우저는 **key/value 쌍**을 저장하는데 사용할 수 있는 약간의 **로컬 스토어**를 가지고 있다.



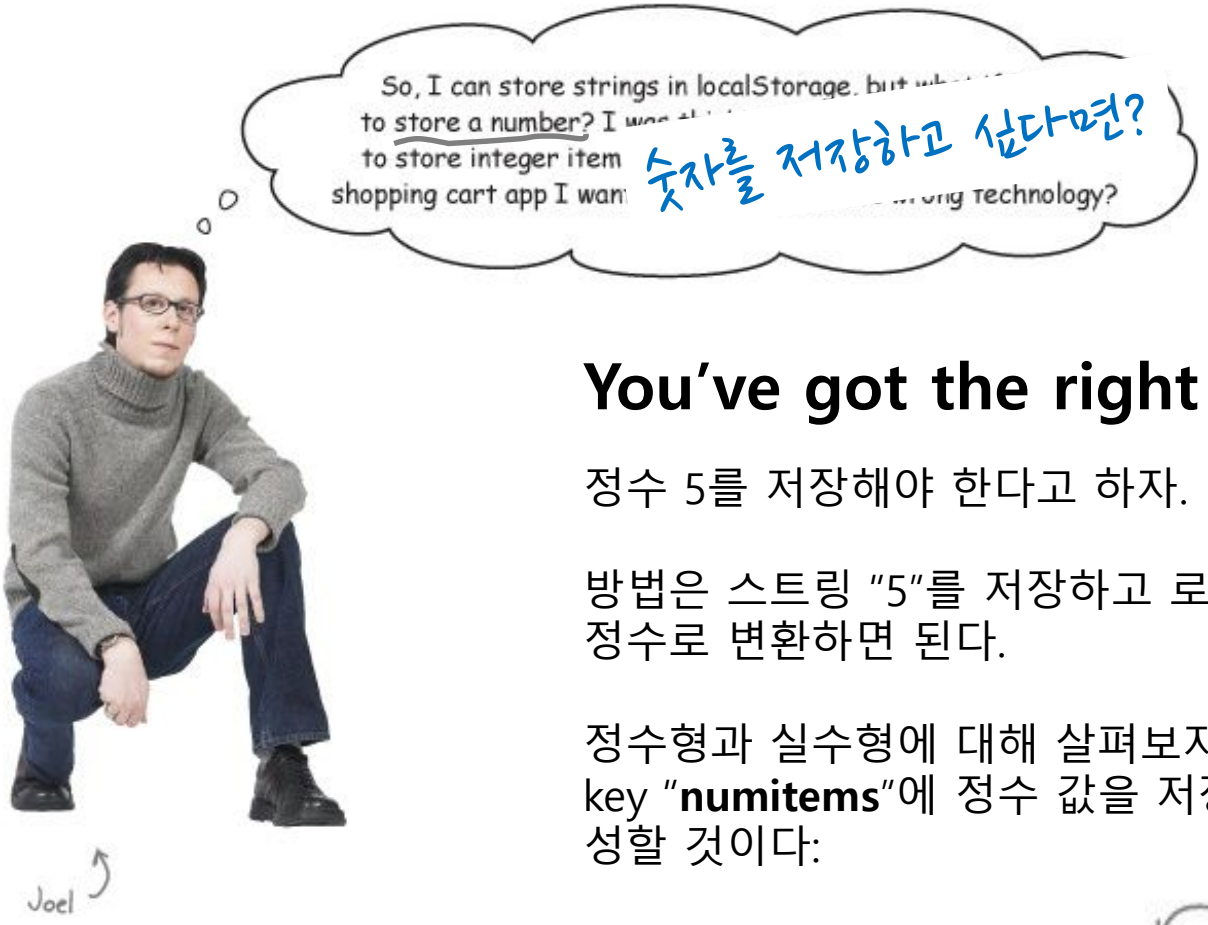
2. 로컬 스토리지에 **key/value 쌍**(둘 다 스트링)을 저장할 수 있다.



3. **setItem**을 다시 호출하여 **두 번째 key/value쌍**을 저장한다.



4. "sticky_0"의 key로 **getItem**을 호출하면 key/value 쌍의 value 값이 리턴된다.



You've got the right technology.

정수 5를 저장해야 한다고 하자.

방법은 스트링 "5"를 저장하고 로컬 스토어에서 꺼내올 때 정수로 변환하면 된다.

정수형과 실수형에 대해 살펴보자:
key "**numitems**"에 정수 값을 저장하기 위해 다음과 같이 작성할 것이다:

```
localStorage.setItem("numitems", 1);
```

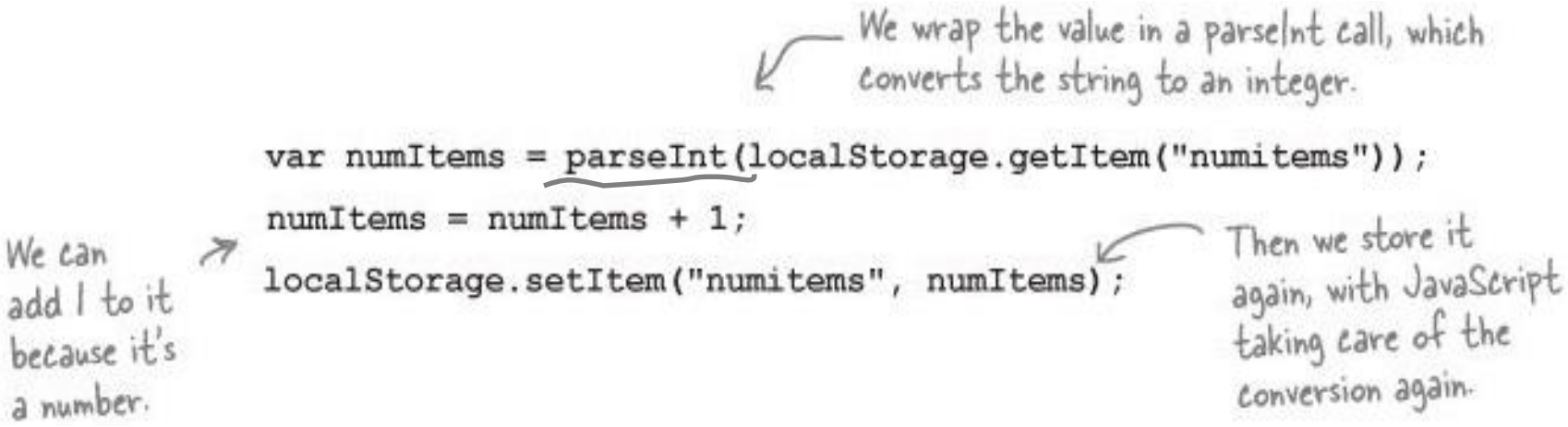
What? Didn't we just say we couldn't store integers?

위에서 **setItem** 메소드가 실제로 보는 것은 정수 값 1이 아니라 스트링 "1" 이다.

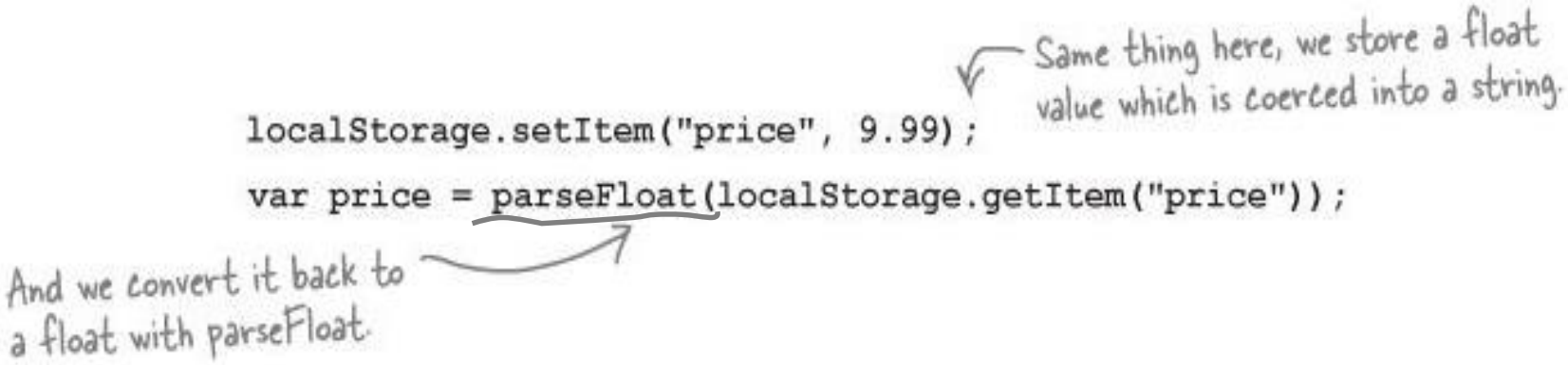
한편, 자바스크립트의 **getItem**은 값을 꺼내 올 때도 그렇게 스마트하지 않다.

```
var numItems = localStorage.getItem("numitems");
```

"numItems"를 숫자로 변환하기 위해서는 JavaScript 함수 **parseInt**를 이용하여 스트링을 정수로 변환해야 한다:



실수형으로 변환하기 위해서는 **parseFloat** 함수를 사용하면 된다:



We're Local Storage and the Array separated at birth?

localStorage는 게터/세터 메소드를 제공할 뿐만 아니라 localStorage 객체를 **연상배열** (associative array)로 취급할 수 있도록 해준다.

즉 **setItem** 메소드를 사용하는 대신에 다음처럼 **key를 배열의 index처럼 사용**할 수 있다:

```
localStorage["sticky_0"] = "Pick up dry cleaning";
```

Here, the key looks like an index for the storage array.

And here's our value sitting over here on the righthand side of an assignment statement.

또한 같은 방식으로 key에 저장된 값을 꺼내올 수 있다:

```
var sticky = localStorage["sticky_0"];
```

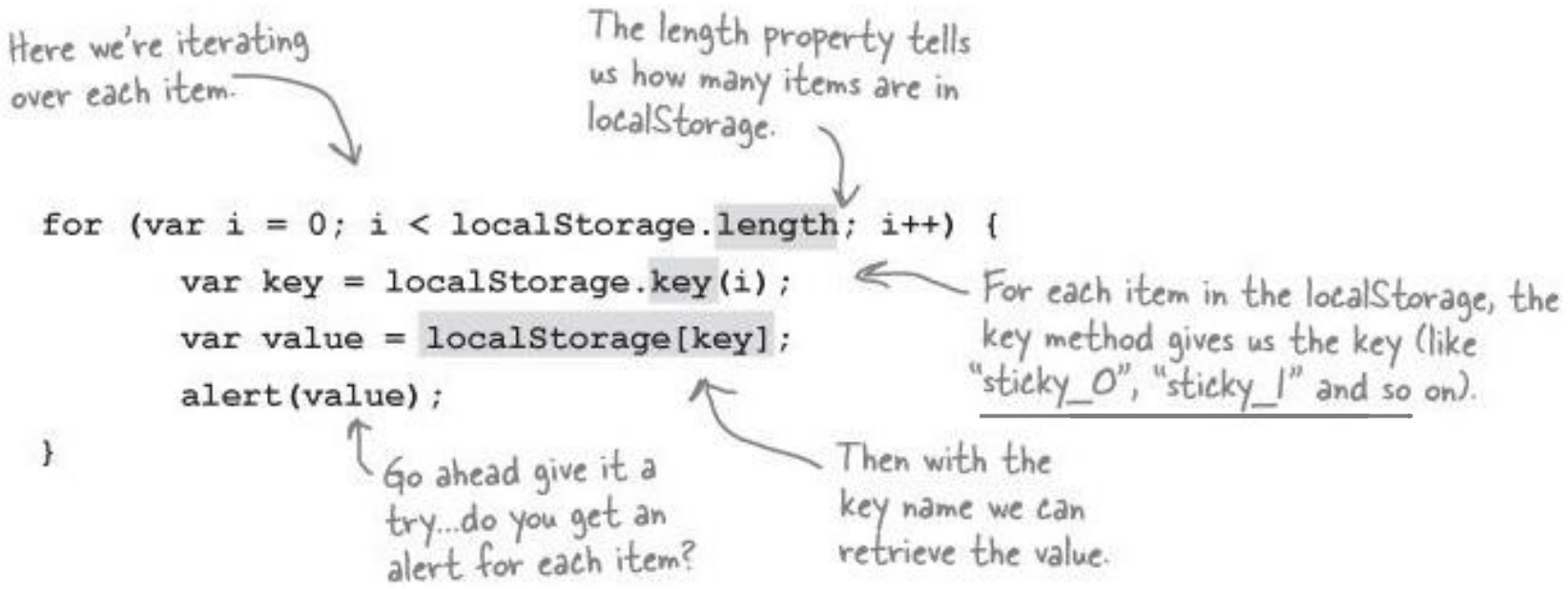
This works exactly like using the call to the getItem method.

Here we assign our variable sticky to...

...the value of the key "sticky_0" in the local store.

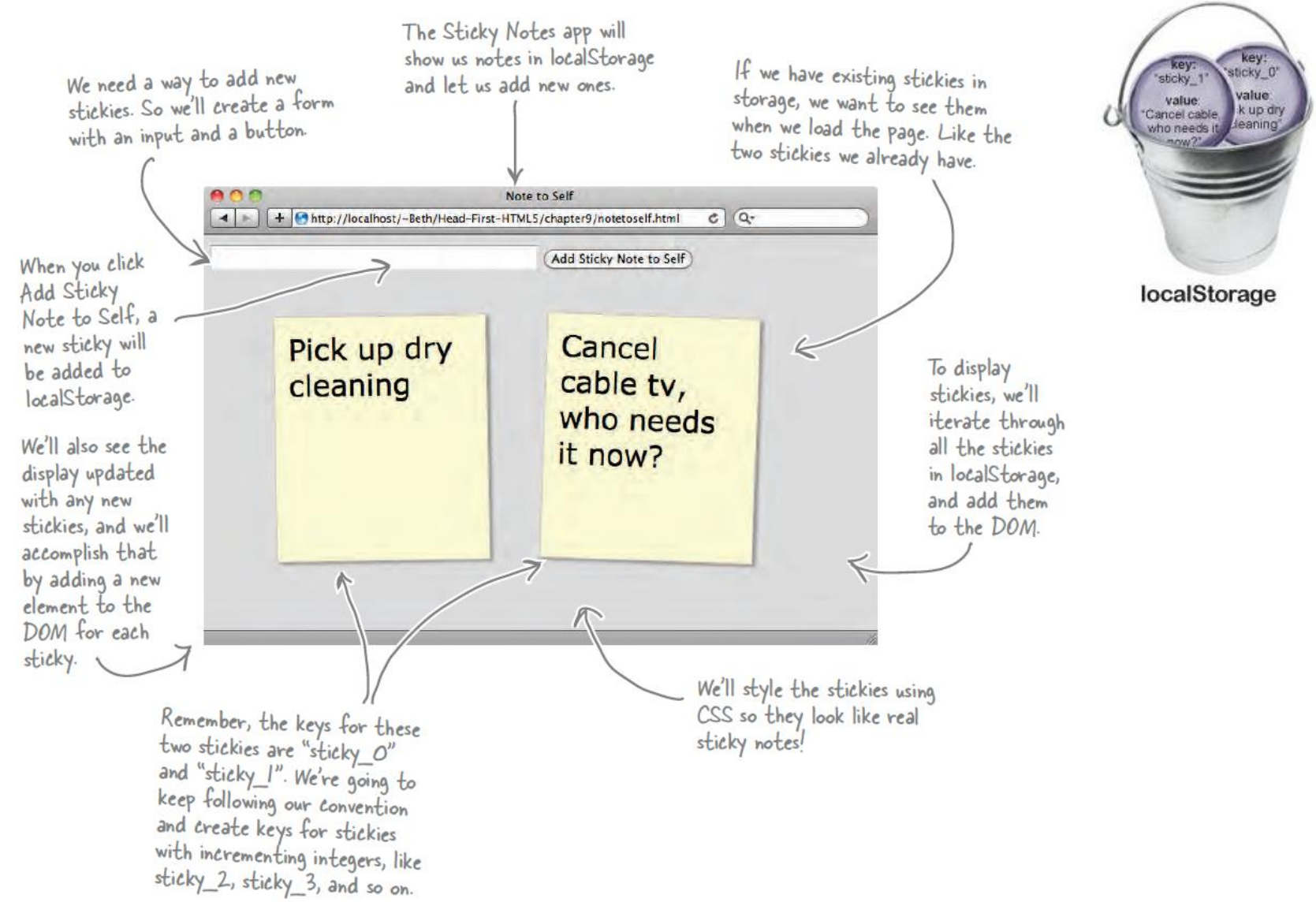
But wait, there's more!

localStorage API는 또 다른 두 가지 특징을 제공한다: 프로퍼티 **length**와 메소드 **key**



Getting serious about stickies

Let's take a peek at what we're going to build before we build it.



Creating the interface

```
<!doctype html>
<html>
<head>
<title>Note to Self</title>
<meta charset="utf-8">
<link rel="stylesheet" href="notetoself.css">
<script src="notetoself.js"></script>
</head>
<body>
    <form>
        <input type="text" id="note_text">
        <input type="button" id="add_button" value="Add Sticky Note to Self">
    </form>

    <ul id="stickies">
    </ul>
</body>
</html>
```

We've added a form as a user interface to enter new stickies.

And we've got to have somewhere to place our stickies in the interface, so we're going to put them in a unordered list.

The CSS handles making each list item look a little more like a Post-it note.

notetoself.css:
http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch9/notetoself.css

Now let's add the JavaScript

현재 localStorage에 한 쌍의 Sticky 노트가 들어있다.
먼저 localStorage로부터 노트들을 읽어 들여서 ** 엘리먼트**에 두는 일을 해보자:

When the page is loaded we're going to call the init function...

```
window.onload = init;


function init() {
    for (var i = 0; i < localStorage.length; i++) {
        var key = localStorage.key(i);
        if (key.substring(0, 6) == "sticky") {
            var value = localStorage.getItem(key);
            addStickyToDOM(value);
        }
    }
}
```

...which reads all the existing stickies from localStorage and adds them to the <ul> through the DOM.

To do that we iterate over all items in the store.

Grab each key.

And then we make sure this item is a sticky by testing to see if its key begins with "sticky". Why do we do that? Well, there might be other items stored in localStorage other than our stickies (more on this in a bit).

If it's a sticky, then grab its value and add it to our page (via the DOM).

다음은 **addStickyToDOM** 함수를 작성해보자.

⇒ 이 함수는 노트들을 **<ul>** 엘리먼트에 추가한다:

```
function addStickyToDOM(value) {
    var stickies = document.getElementById("stickies");
    var sticky = document.createElement("li");
    var span = document.createElement("span");
    span.setAttribute("class", "sticky");
    span.innerHTML = value;
    sticky.appendChild(span);
    stickies.appendChild(sticky);
}
```

We're being passed the text of the sticky note. We need to create a list item for the unordered list and then insert it.

So, let's get the "stickies" list element.

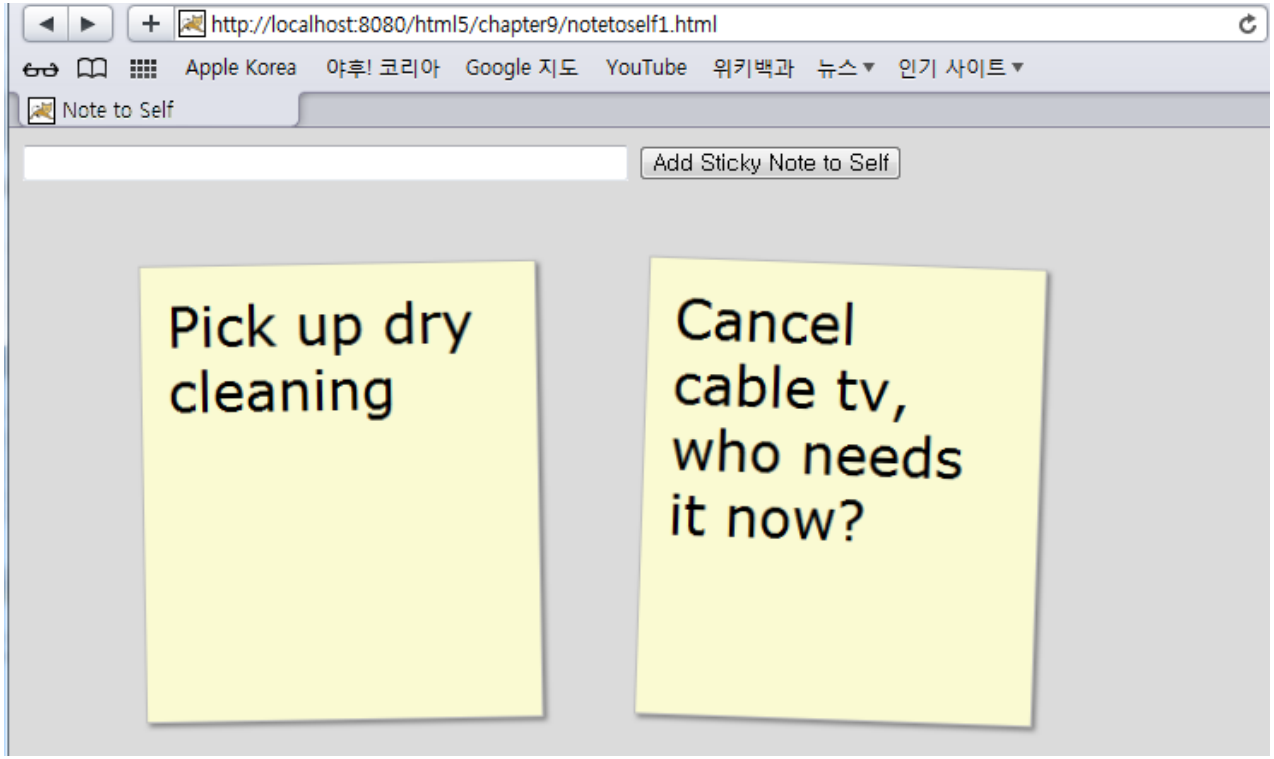
Create a list element, and give it a class name of "sticky" (so we can style it).

Set the content of the span holding the text of the sticky note.

And add the span to the "sticky" li, and the li to the "stickies" list.

실습과제 15-2

Go ahead and get this code into your script element and load it into your browser. Here's what we got when we loaded the page in our browser:



http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch9/notetoself2.html

Completing the user interface

이제 남아있는 일은 폼에 새로운 노트를 추가하는 방법을 작성하는 것이다.

"Add Sticky Note to Self" 버튼이 클릭될 때의 핸들러를 추가한다.

즉 새로운 Sticky를 생성한다.

Add this new code to your init function:

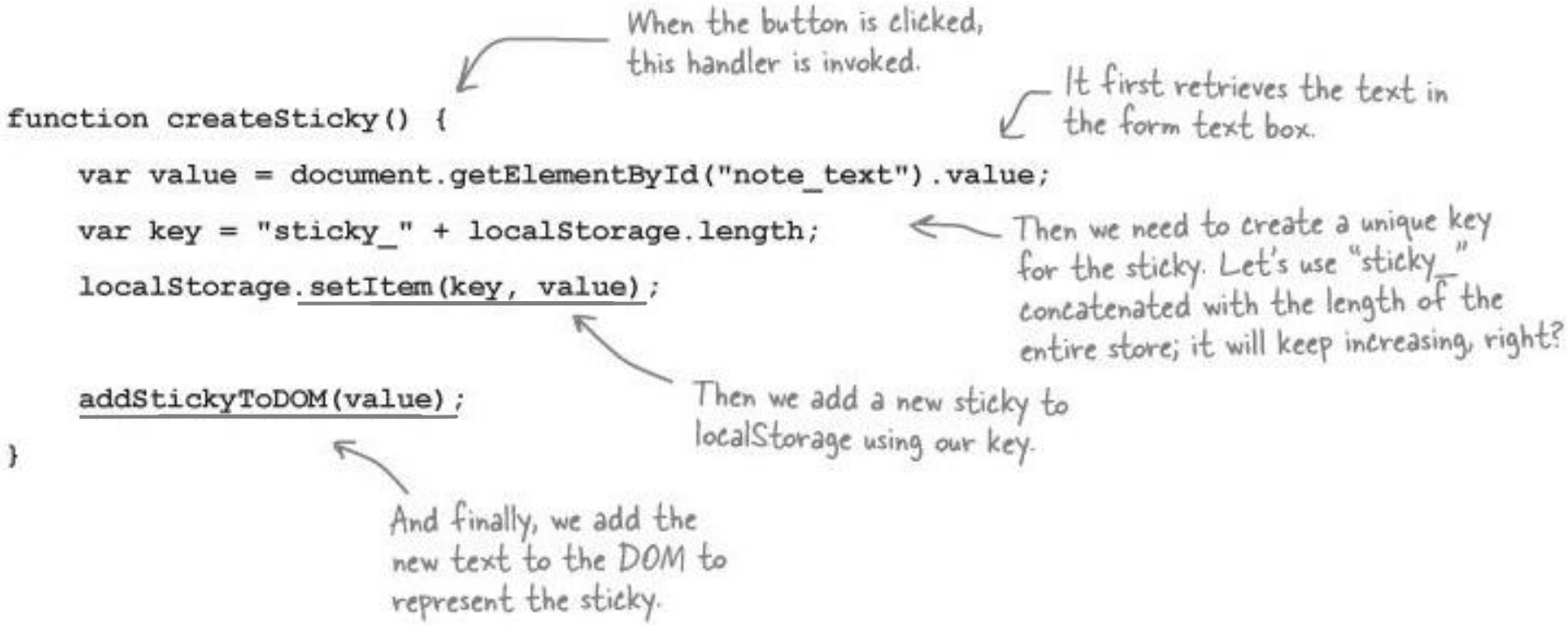
```
function init() {
    var button = document.getElementById("add_button");
    button.onclick = createSticky;

    // for loop goes here
}
```

Let's grab a reference to the "Add Sticky Note to Self" button.

And add a handler for when it is clicked. Let's call the handler createSticky.

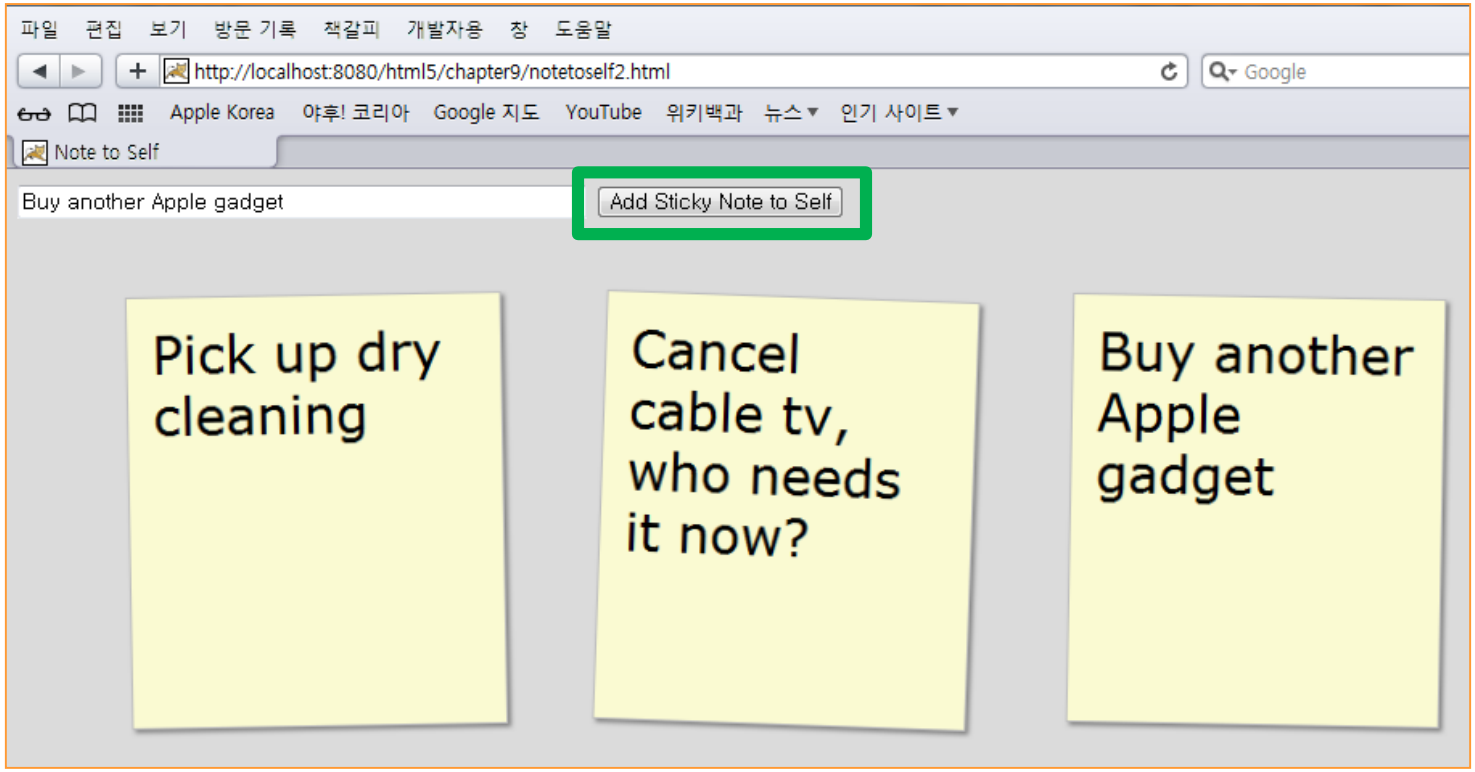
The rest of the code in init stays the same, we're saving a few trees by not repeating it here.

And the code to create a **new sticky** note:



실습과제 15-3 Add Sticky Note to Self

Now we're truly interactive! Load this new code in your browser, enter a new "sticky note to self" and click or tap the "**Add Sticky Note to Self**" button. You should see the new sticky note appear in your list of stickies. Here's what we see:



http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch9/notetoself3.html

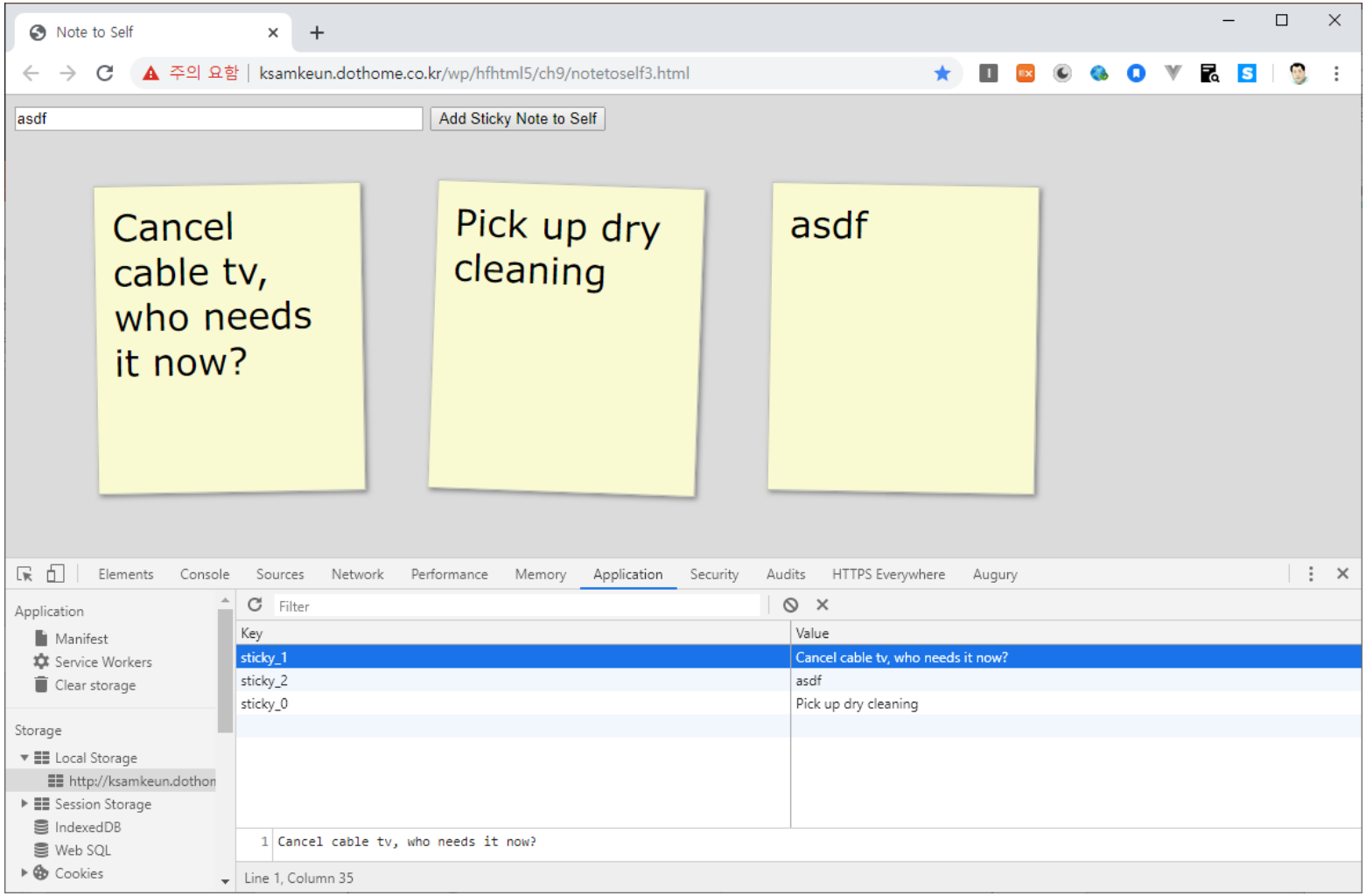
We need to stop for a little scheduled service

localStorage에 있는 항목들을 직접 볼 수 있는 툴이 있다면 좋지 않을까?

현대의 모든 브라우저는 로컬 스토어를 직접 검사할 수 있는 **개발자 도구**를 지원한다.
개발자 도구는 브라우저마다 다르다.

Developer Tools for HTML5 Programming

개발자용 > 웹속성보기 클릭!



Do-It-Yourself maintenance

localStorage API는 로컬 스토어에 있는 모든 항목을 삭제해 버리는 **clear** 메소드를 포함하고 있다.
JavaScript에서 어떻게 사용될 수 있나를 살펴보자: **maintenance.html**

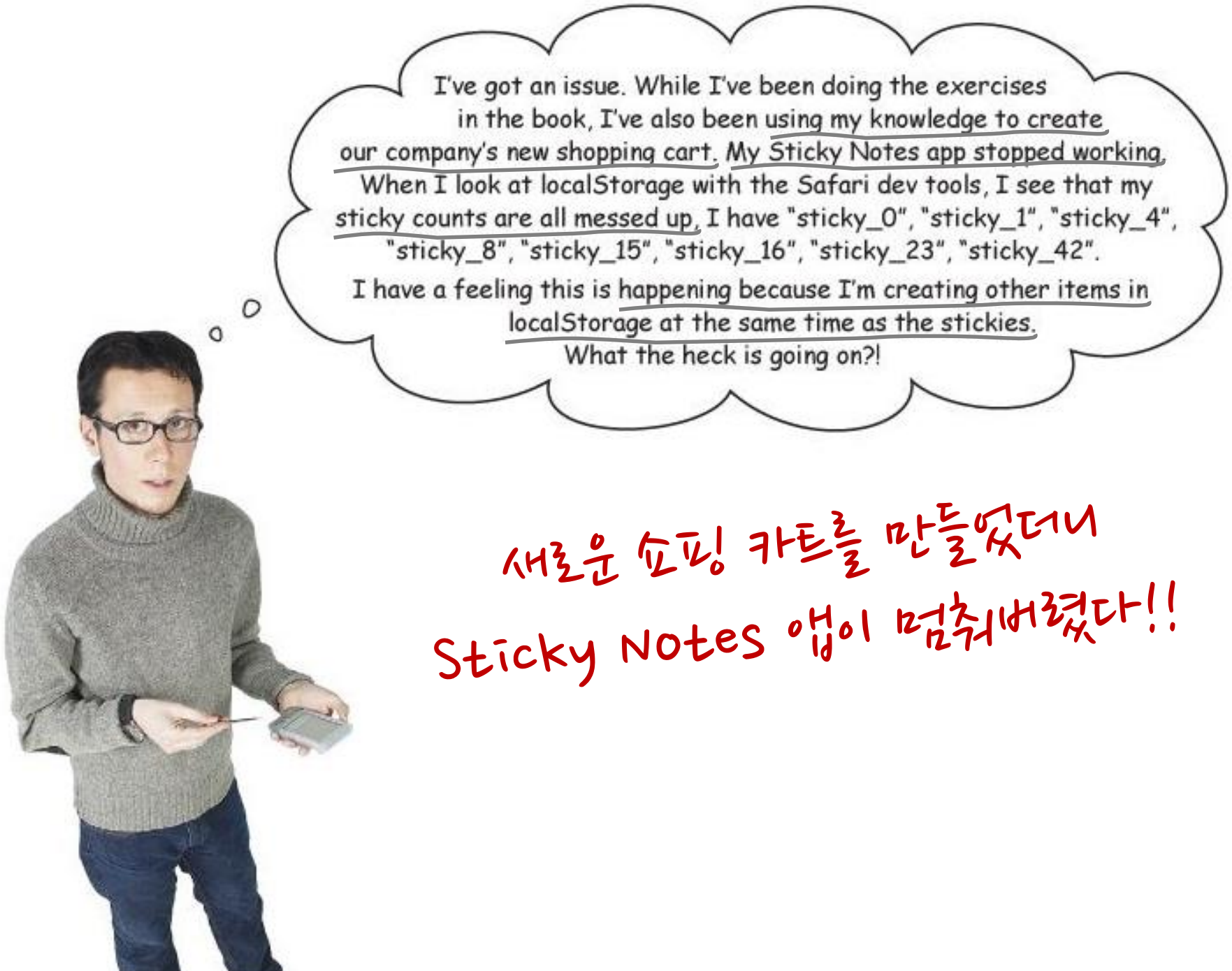
```
<!doctype html>
<html>
<head>
<title>Maintenance</title>
<meta charset="utf-8">
<script>
window.onload = function() {
    var clearButton = document.getElementById("clear_button");
    clearButton.onclick = clearStorage;
}

function clearStorage() {
    localStorage.clear();
}
</script>
</head>
<body>
    <form>
        <input type="button" id="clear_button" value="Clear storage" />
    </form>
</body>
</html>
```

We've added one button to the page, and this code adds a click handler for the button.

When you click the button, the clearStorage function is called.

All this function does is call the localStorage.clear method. Use with caution as it will delete all the items associated with the origin of this maintenance page!

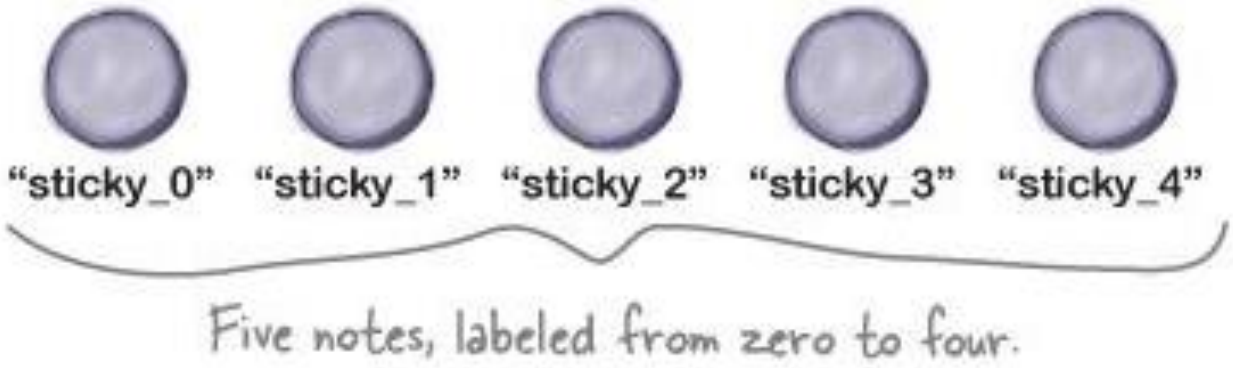
And here's our button. Use this file whenever you need to erase everything in localStorage (good for testing).


I've got an issue. While I've been doing the exercises in the book, I've also been using my knowledge to create our company's new shopping cart. My Sticky Notes app stopped working. When I look at localStorage with the Safari dev tools, I see that my sticky counts are all messed up, I have "sticky_0", "sticky_1", "sticky_4", "sticky_8", "sticky_15", "sticky_16", "sticky_23", "sticky_42".
I have a feeling this is happening because I'm creating other items in localStorage at the same time as the stickies.
What the heck is going on?!
새로운 쇼핑 카트를 만들었더니
Sticky Notes 앱이 멈춰버렸다!!

Ah, you've discovered a major design flaw.

맞다. 어떤 **다른 항목을 localStorage에 추가**하게 되면 더 이상 앱은 동작하지 않게 된다!

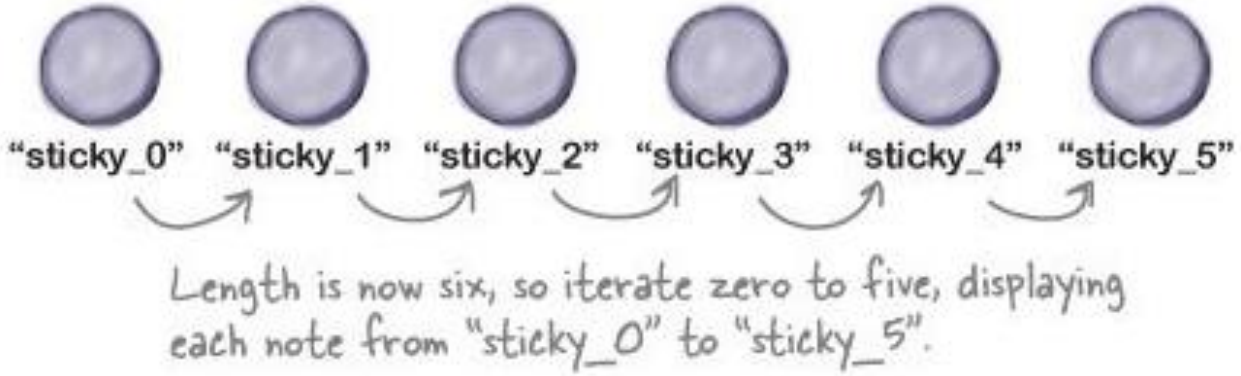
첫 번째 이유 => Sticky 노트가 0부터 시작하여 Sticky 개수만큼 번호가 붙여진다는 것이다:



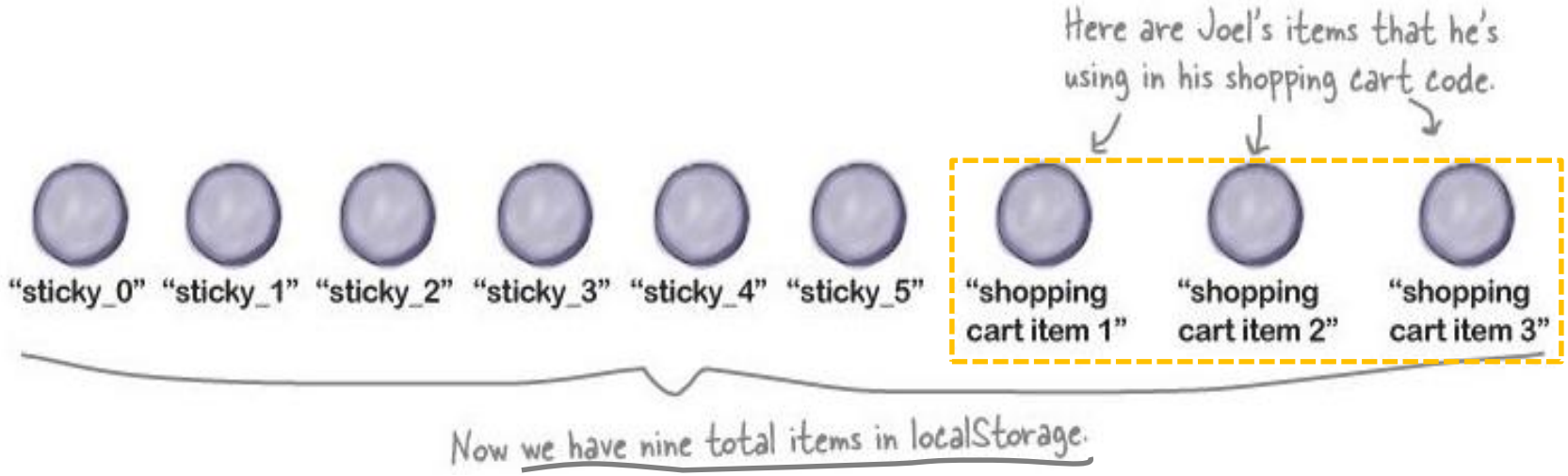
새로운 Sticky는 로컬 스토어의 맨 끝 항목으로 추가된다:

```
var key = "sticky_" + localStorage.length;
```

모든 Sticky를 디스플레이하기 위해 0부터 **[로컬 스토어 길이 – 1]** 만큼 반복한다:

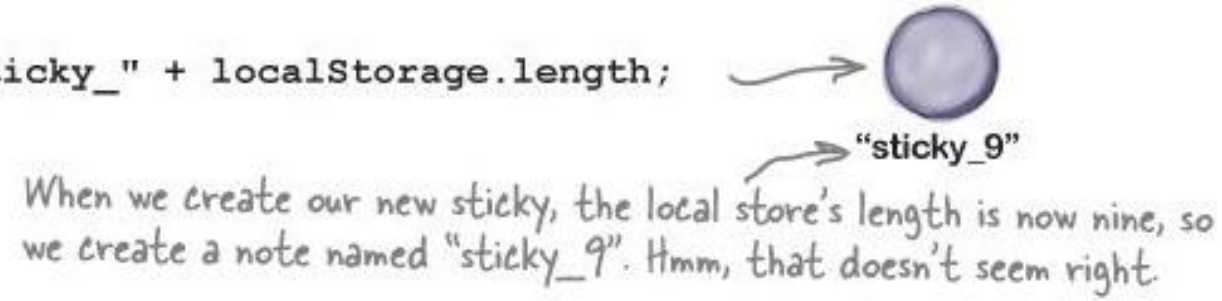


이제 localStorage에 **쇼핑 카트 항목들을 추가**한다:

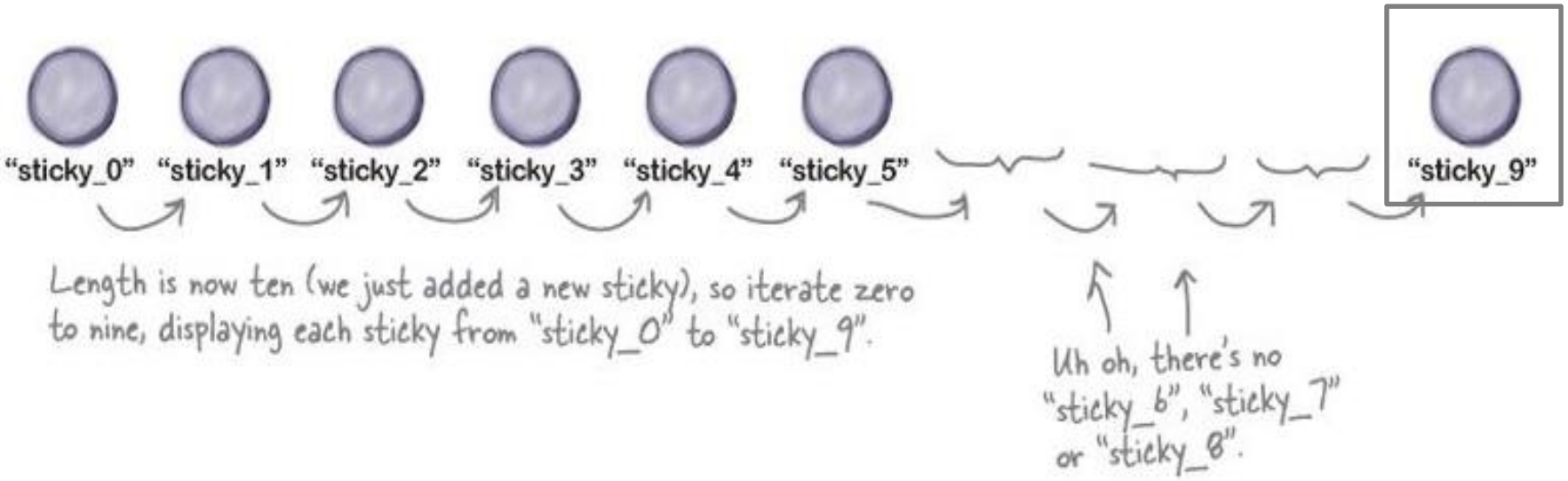


다음은 새로운 Sticky를 생성해 보자:

```
var key = "sticky_" + localStorage.length;
```



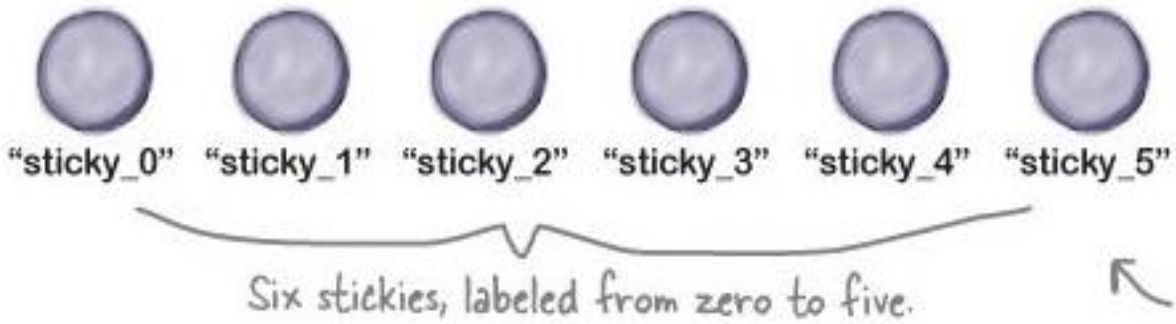
이제 sticky 들을 순서대로 디스플레이하고 싶을 때 문제가 발생한다:



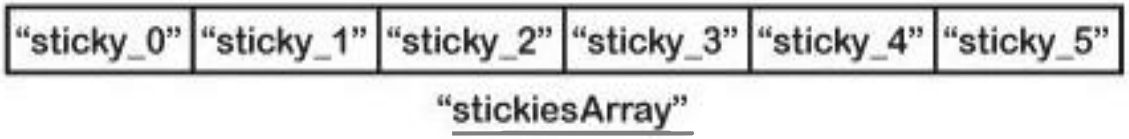
localStorage에 배열을 저장할 수 있다면?

We have the technology...

다시 현재 localStorage에 여섯 개의 Sticky를 가졌다고 해보자:



Stickies 와 stickies 배열 둘 다 localStorage에 저장된다!



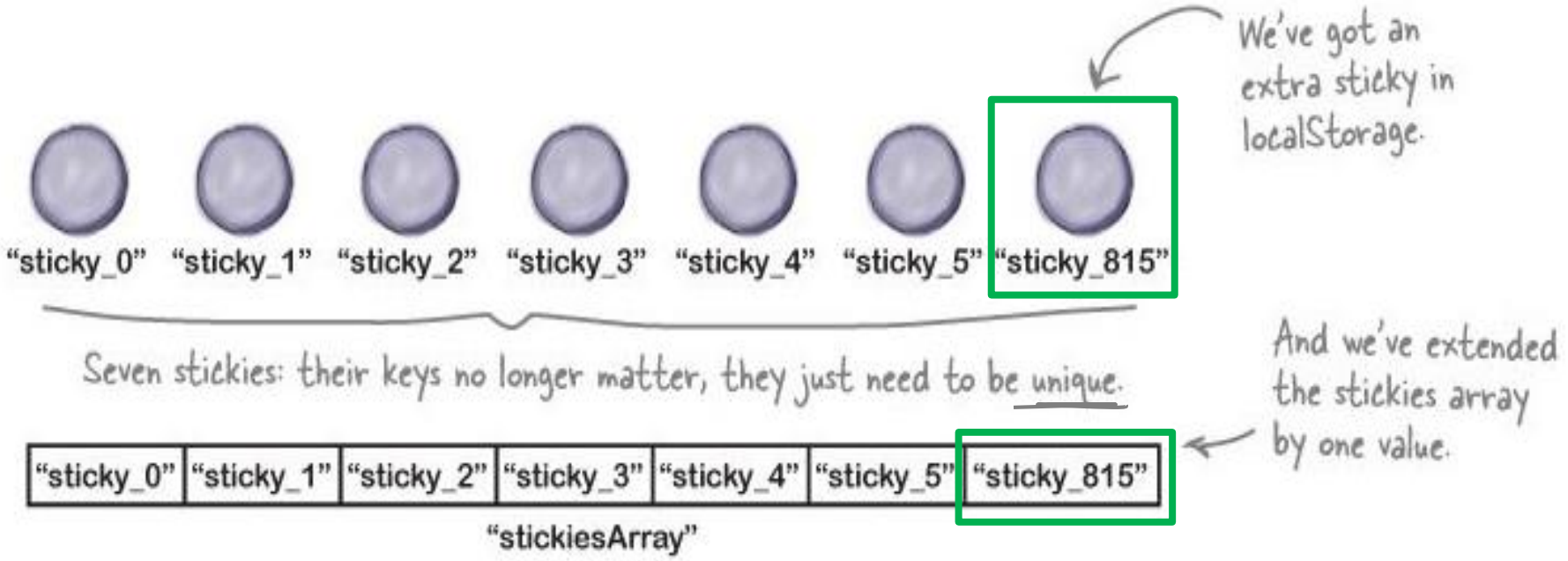
Stickies 배열의 각 원소는 localStorage의 sticky에 대한 key가 된다.

그리고 localStorage에 "**stickiesArray**"라는 이름의 배열을 만들었다고 하자.

이제 새로운 "**sticky_815**"라는 Sticky를 추가해보자.

어떤 숫자인지는 무의미하다는 것을 보여주기 위해서 815같은 숫자를 사용했다.

"**sticky_815**"를 배열에 추가하고 항목을 localStorage에 넣기만 하면 된다:



Reworking our app to use an array

Before...

```
function init() {
    // button code here...
    for (var i = 0; i < localStorage.length; i++) {
        var key = localStorage.key(i);
        if (key.substr(0, 6) == "sticky") {
            var value = localStorage.getItem(key);
            addStickyToDOM(value);
        }
    }
}
```

Here's our old code that relies on the stickies having specific names, sticky_0, sticky_1, and so on..

Wow, this was messy, come to think of it.

As we now know, this might break because we can't depend on all stickies to be there if we're naming them based on the count of the items in localStorage.

New and improved

```
function init() {
    // button code here...
    var stickiesArray = localStorage["stickiesArray"];
    if (!stickiesArray) {
        stickiesArray = [];
        localStorage.setItem("stickiesArray", stickiesArray);
    }

    for (var i = 0; i < stickiesArray.length; i++) {
        var key = stickiesArray[i];
        var value = localStorage[key];
        addStickyToDOM(value);
    }
}
```

We're starting by grabbing the stickiesArray out of localStorage.

We need to make sure there is an array in localStorage. If there isn't one, then let's create an empty one.

We're iterating here through the array.

Each element of the array is the key of a sticky, so we're using that to retrieve the corresponding item from localStorage.

And then we add that value to the DOM just like we have been.

Converting createSticky to use an array

앱을 거의 완성했다 => **createSticky** 메소드만 다시 수정하면 된다.

변경하기 전에 현재의 구현 상태를 살펴보자:

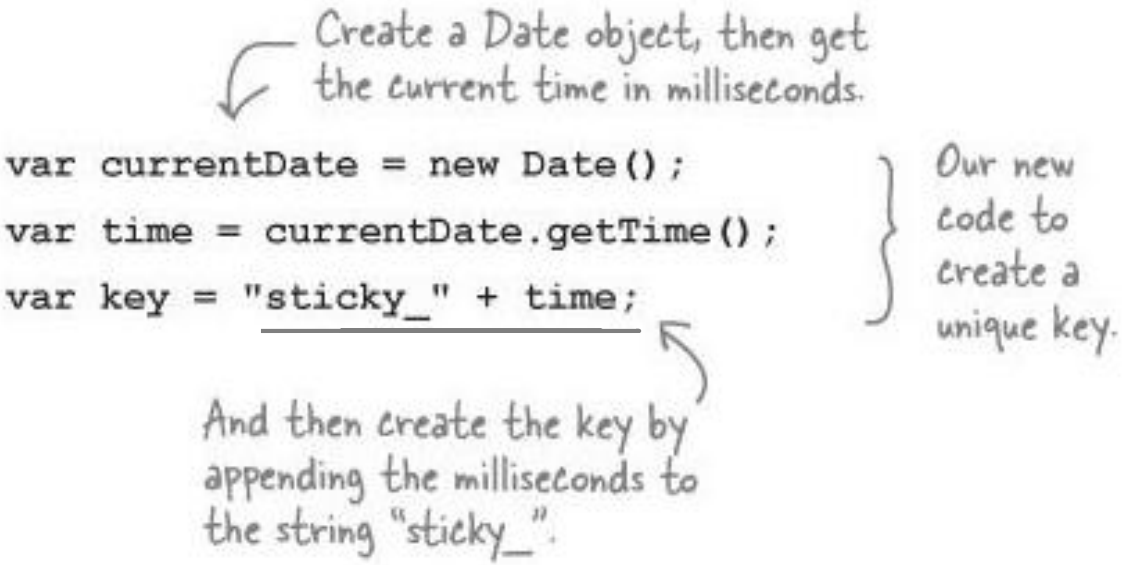
```
function createSticky() {
    var value = document.getElementById("note_text").value;
    var key = "sticky_" + localStorage.length;
    localStorage.setItem(key, value);

    addStickyToDOM(value);
}
```

Rather than using the localStorage length to create a key, which we've seen can cause problems, we're going to need to create a more unique key.

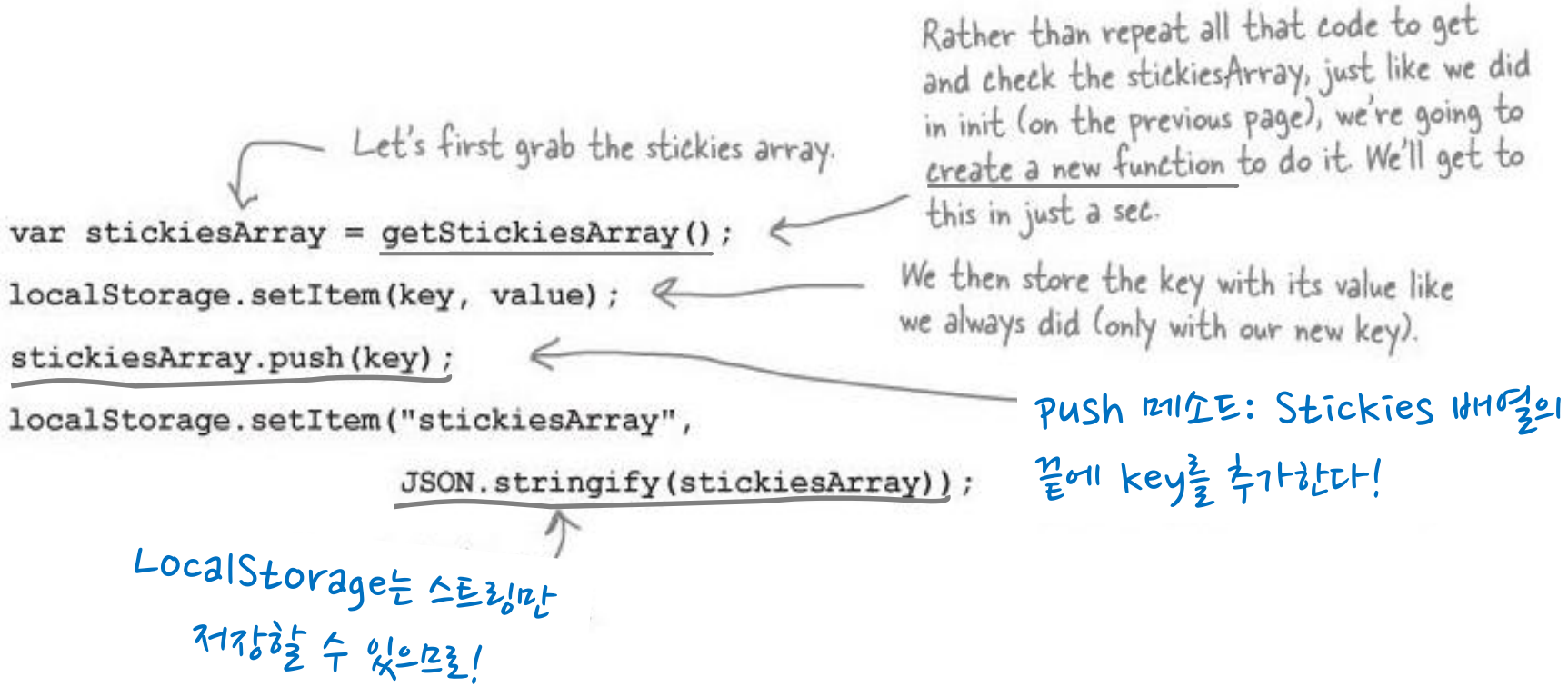
Stickies 배열에 sticky를 추가하고,
localStorage에 그 배열을 저장한다.

What needs to change?

Sticky에 대한 유일한 키를 생성할 필요가 있다:



배열에 새로운 Sticky를 저장할 필요가 있다:




Excellent, once I've got this working
I'm going to rework my shopping cart the same
way and these two apps are going to be able to work
from the same origin without any problems.
I also love using an array; it makes everything much
simpler to keep track of!

Putting it all together

새로운 배열 기반 코드를 모두 함께 통합

그렇게 하기 전에 **localStorage**로부터 **stickies** 배열을 검색해주는 코드를 작성하자:

```
function getStickiesArray() {

    var stickiesArray = localStorage.getItem("stickiesArray");

    if (!stickiesArray) {

        stickiesArray = [];

        localStorage.setItem("stickiesArray", JSON.stringify(stickiesArray));

    } else {

        stickiesArray = JSON.parse(stickiesArray);

    }

    return stickiesArray;

}
```

First we get the item "stickiesArray" out of localStorage.

If this is the first time we've loaded this app, there might not be a "stickiesArray" item.

And if there isn't an array yet we create an empty array, and then store it back in localStorage.

Don't forget to stringify it first!

그렇지 않고 localStorage 에서 배열을 발견했다면, 파싱을 위해 JavaScript 배열로 변환한다!

어느 경우이든 배열을 반환한다!

init과 **createSticky** 함수의 최종 버전:

```
function init() {
    var button = document.getElementById("add_button");
    button.onclick = createSticky;

    var stickiesArray = getStickiesArray();

    for (var i = 0; i < stickiesArray.length; i++) {
        var key = stickiesArray[i];
        var value = localStorage[key];
        addStickyToDOM(value);
    }
}
```

Remember we also set up the button events here in the init method.

Next we grab the array with the stickies' keys in it.

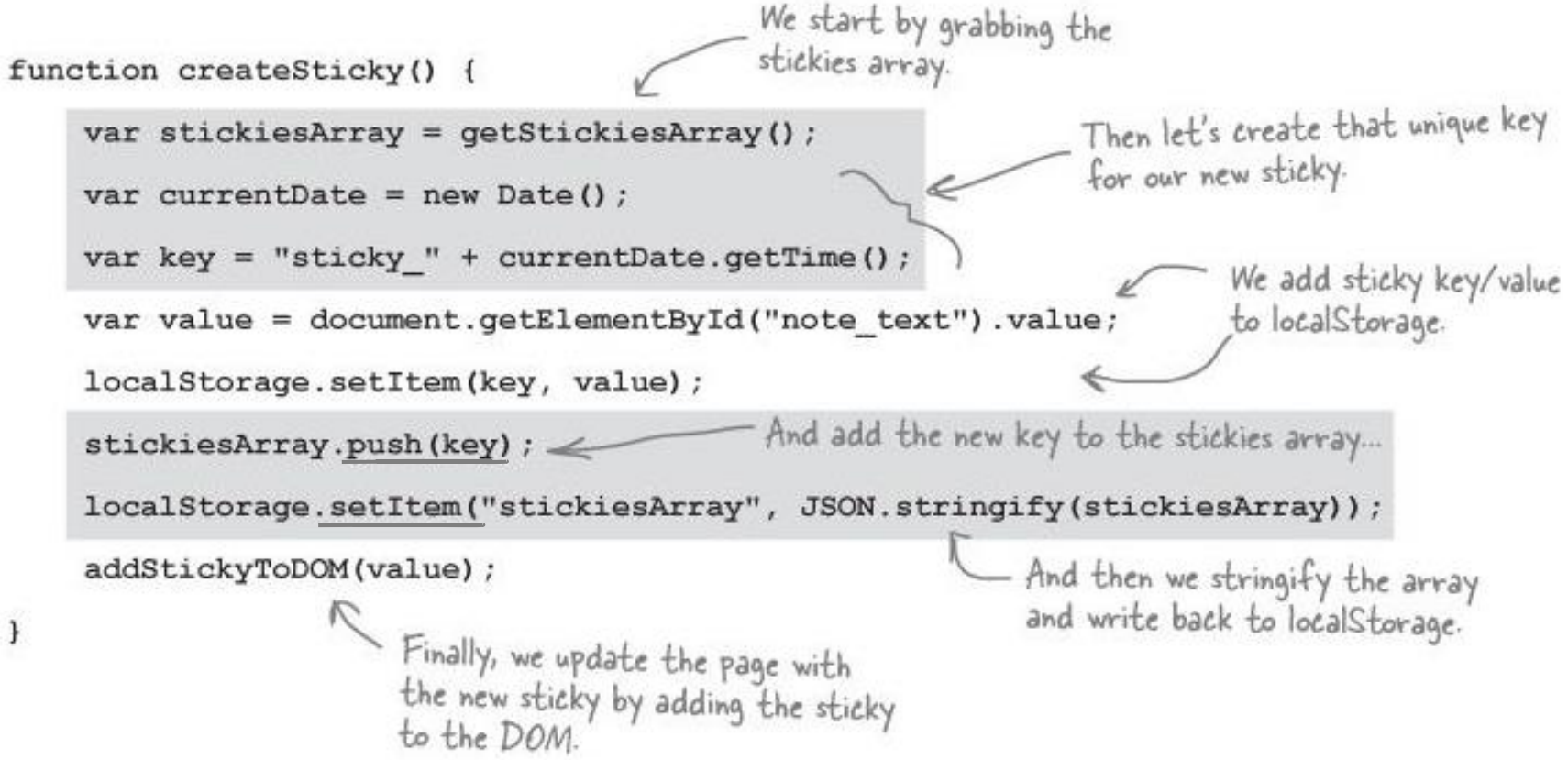
Now we're going to iterate through the stickies array (not the localStorage items!).

Each item in the array is a key to a sticky. Let's grab each one.

And grab its value from localStorage.

And add it to the DOM just like we've been doing.

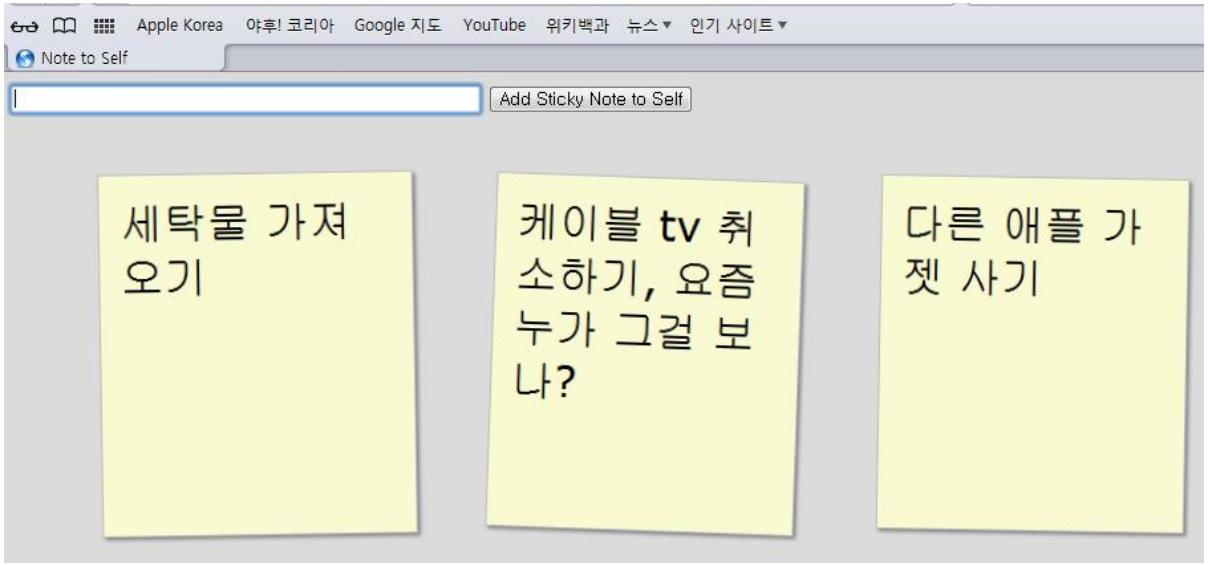
createSticky:



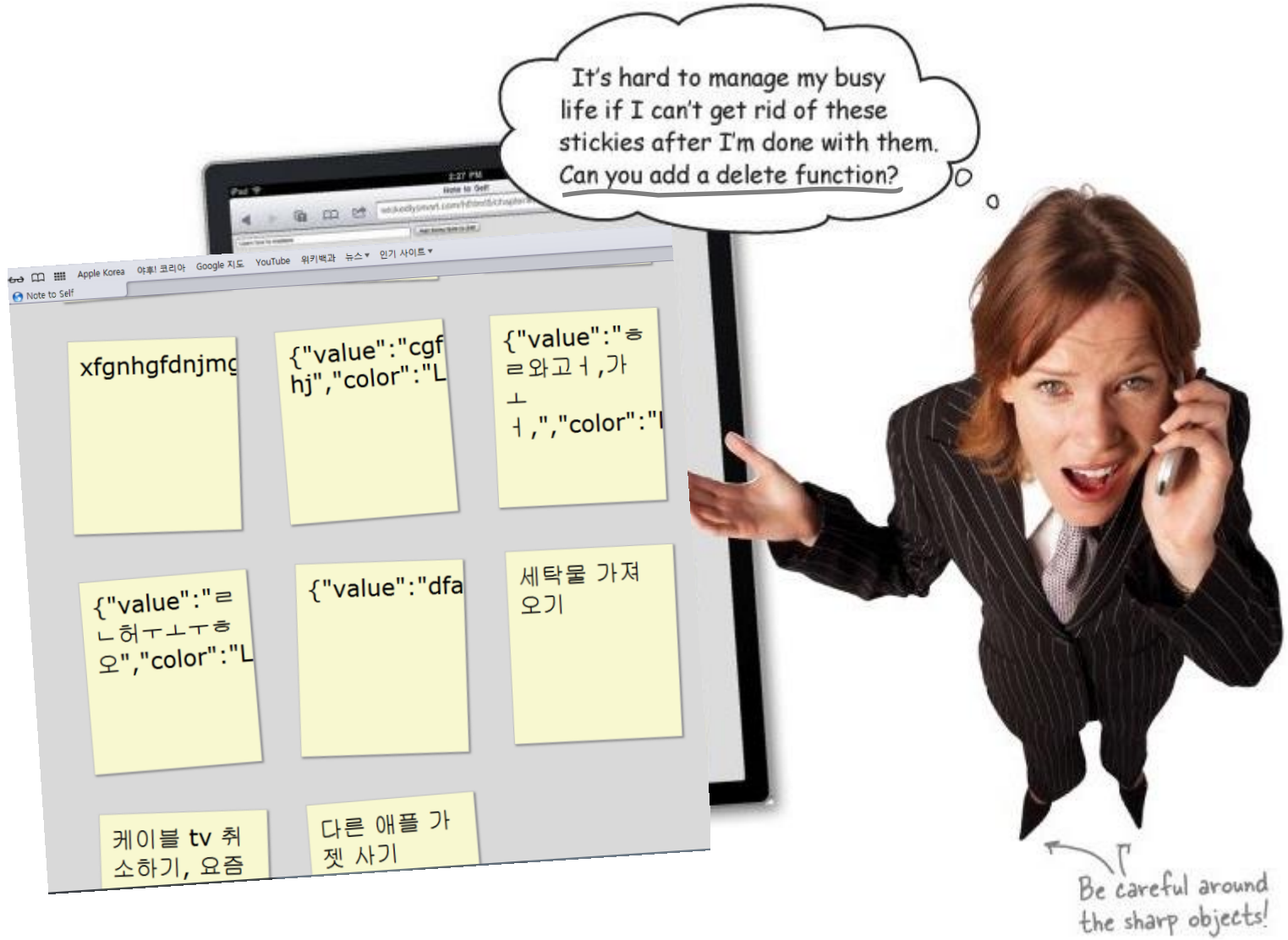

```
function createSticky() {
    var stickiesArray = getStickiesArray();
    var currentDate = new Date();
    var key = "sticky_" + currentDate.getTime();
    var value = document.getElementById("note_text").value;
    localStorage.setItem(key, value);
    stickiesArray.push(key);
    localStorage.setItem("stickiesArray", JSON.stringify(stickiesArray));
    addStickyToDOM(value);
}
```

실습과제 15-4 Test Drive!

Get all this code in and clear out your localStorage to make a nice clean start. Load this code, and you should see exactly the same behavior as last time. Joel, you'll see your code working correctly now!



http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch9/notetoself4.html


It's hard to manage my busy life if I can't get rid of these stickies after I'm done with them. Can you add a delete function?
Note to Self
xfgnhgfdnjmg
{"value":"cgf hj","color":"L
{"value":"ㅎ ㄹ와고ㅓ,가 ㅗ ㅓ,","color":"
{"value":"ㄹ ㄴ허ㅜㅗㅜㅎ 오","color":"L
{"value":"dfa
세탁물 가져 오기
케이블 tv 취소하기, 요즘
다른 애플 가젯 사기
Be careful around the sharp objects!

Deleting sticky notes

앱이 Sticky를 제거할 수 없다면 결코 유용하지 않다.

앞에서 이미 **localStorage.removeItem** 메소드에 대해 언급했었다.

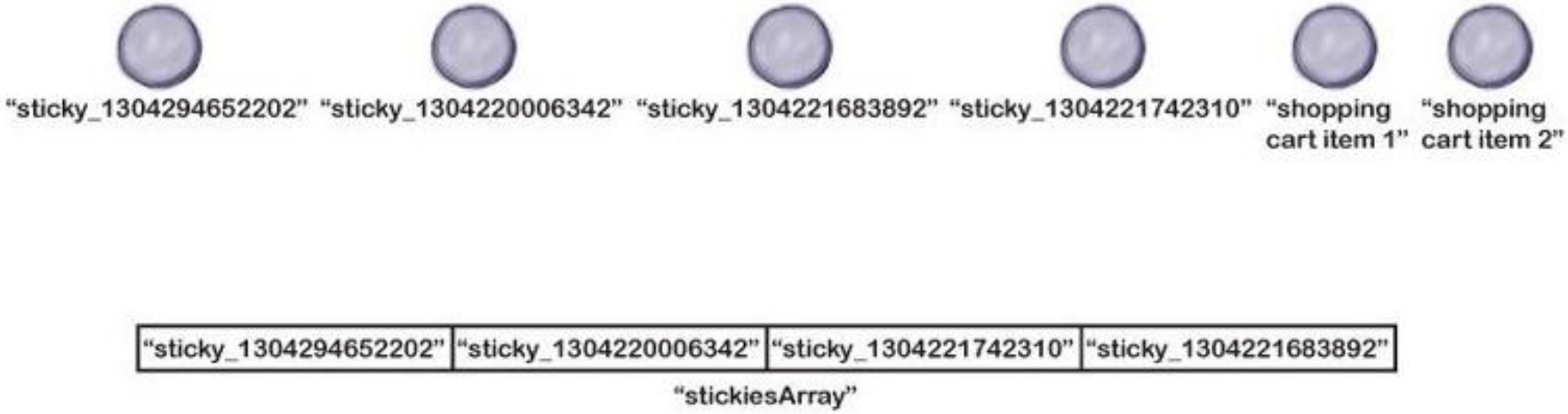
removeItem 메소드

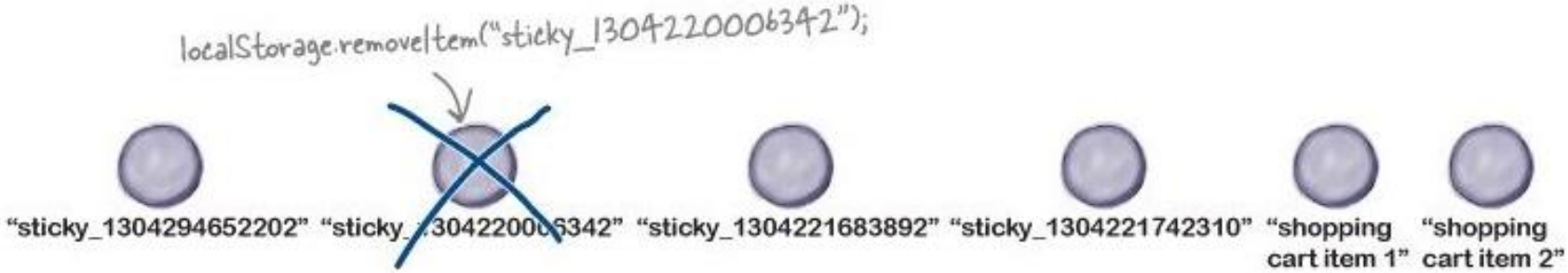
⇒ **항목의 키를 이용**하여 localStorage로부터 해당 항목을 제거한다:

충분히 완벽한 방법이라고 생각들지 모르겠지만 아직 해결할 문제가 남아있다.

stickiesArray를 다루어야 한다 . . .

Let’s delete a sticky!





(1) Remove the sticky with the key "sticky_1304220006342" from localStorage using the localStorage.removeItem method.

(2) Get the stickiesArray.

(3) Remove element with key="sticky_1304220006342" from the stickiesArray.

(4) Write stickiesArray back into localStorage (stringifying it first).

(5) Find "sticky_1304220006342" in the DOM and remove it.

The deleteSticky function

```
function deleteSticky(key) {
    localStorage.removeItem(key);
    var stickiesArray = getStickiesArray();
    if (stickiesArray) {
        for (var i = 0; i < stickiesArray.length; i++) {
            if (key == stickiesArray[i]) {
                stickiesArray.splice(i,1);
            }
        }

        localStorage.setItem("stickiesArray", JSON.stringify(stickiesArray));
    }
}
```

We're using the getStickiesArray function to get the stickiesArray from localStorage.

We make sure we have a stickiesArray (just in case), and then iterate through the array looking for the key we want to delete.

When we find the right key, we delete it from the array using splice.

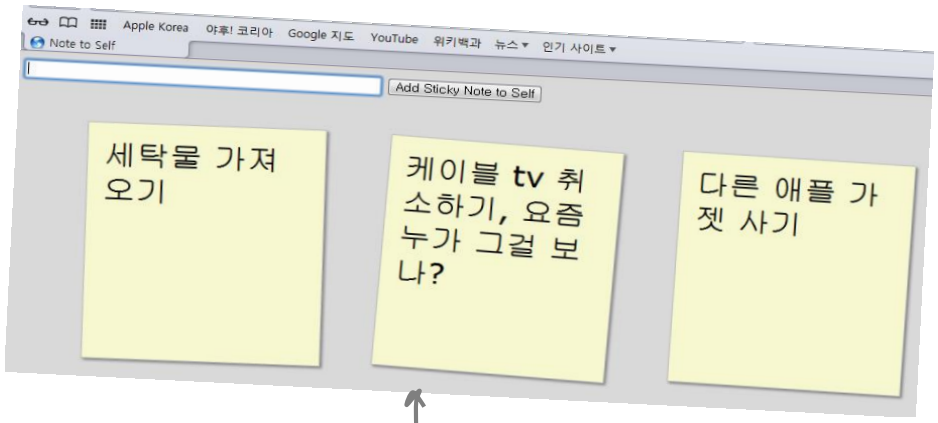
splice removes elements from an array starting at the location given by the first argument (i), for as many elements as are specified in the second argument (1).

Finally, we save the stickiesArray (with the key removed) back to localStorage.

splice(i, 1): 주어진 배열에서 첫 번째 인자(i)에 의해 주어진 위치에서 시작하여 두 번째 인자(1)에 지정된 개수만큼의 요소를 삭제한다.

How do you select a sticky to delete?

먼저 사용자가 **삭제할 sticky를 선택하게 하기 위한 방법**이 필요하다.




```
function deleteSticky(key) {
    localStorage.removeItem(key);
```

먼저 Sticky가 클릭되었을 때를 알아 내어, 그것을 **deleteSticky** 함수에 전달할 수 있도록 Sticky들을 변경해야 한다.

addStickyToDOM 함수:

Big picture: we're going to use the key of the sticky note, which, remember, is "sticky_" + time, to uniquely identify the note. We'll pass in this key whenever we call addStickyToDOM.

```
function addStickyToDOM(key, value) {
    var stickies = document.getElementById("stickies");
    var sticky = document.createElement("li");
    sticky.setAttribute("id", key);
    var span = document.createElement("span");
    span.setAttribute("class", "sticky");
    span.innerHTML = stickyObj.value;
    sticky.appendChild(span);
    stickies.appendChild(sticky);
    sticky.onclick = deleteSticky;
}
```

We're adding a unique id to the <li> element that represents the sticky in the DOM. We're doing this so deleteSticky will know which sticky you clicked on. Since we already know the sticky's key is unique, we're just using that as the id.

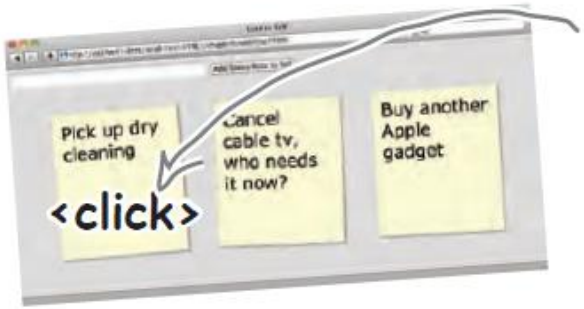
We're also adding click handler to every sticky. When you click on a sticky, deleteSticky will be called.

How to get the sticky to delete from the event

앞에서 각 Sticky 노트에 **이벤트 핸들러**를 만들었다.

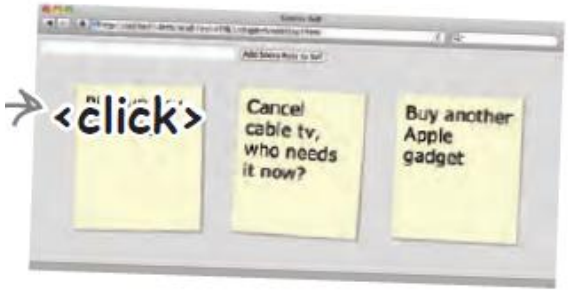
Sticky를 클릭하면 deleteSticky 함수가 호출 => 이벤트 객체가 deleteSticky 함수에 전달

어느 Sticky가 호출되었는지는 **event.target**에서 찾을 수 있다.



Sticky의 노란색 부분을 클릭한다면 event.target은 <li> 엘리먼트가 된다. 는 이제 sticky 노트의 키가 된다.

텍스트를 클릭한다면 event.target은 <li> 엘리먼트 안의 이 된다. 이것은 우리가 원하는 것이 아니다.




```
<li id="sticky_1304270008375">
    <span class="sticky">Pick up dry cleaning</span>
</li>
```

This is the HTML for the sticky note that we create in addStickyToDOM.

사용자의 클릭에 의해 생성된 이벤트가 deleteSticky에 전달된다.

```
function deleteSticky(e) {
    var key = e.target.id;
    if (e.target.tagName.toLowerCase() == "span") {
        key = e.target.parentNode.id;
    }
    localStorage.removeItem(key);
    var stickiesArray = getStickiesArray();
    if (stickiesArray) {
        for (var i = 0; i < stickiesArray.length; i++) {
            if (key == stickiesArray[i]) {
                stickiesArray.splice(i,1);
            }
        }
        localStorage.setItem("stickiesArray", JSON.stringify(stickiesArray));
        removeStickyFromDOM(key);
    }
}
```

The target is the element you clicked on that generated the event, and we can get the id of that element from the target property. If the target is <li>, we're set.

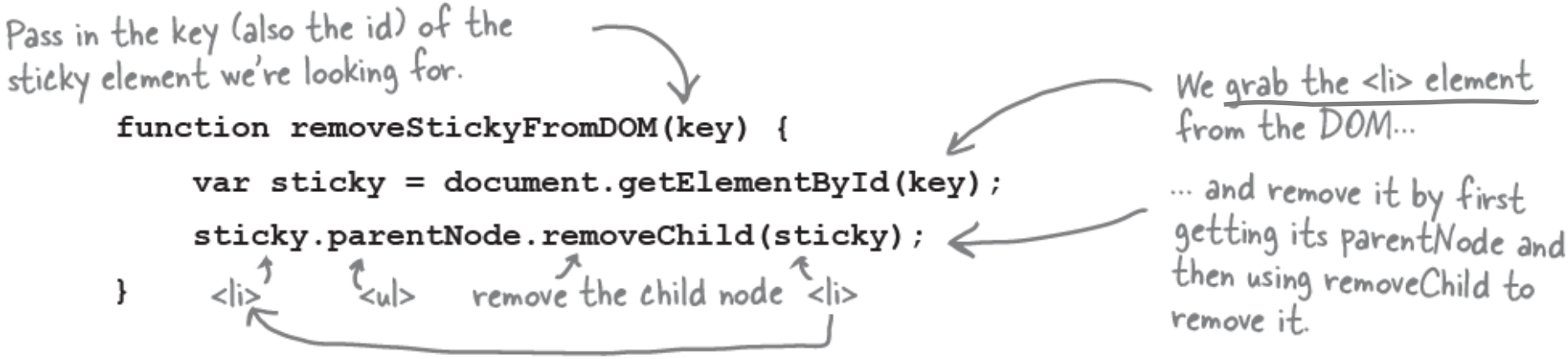
If the target is the , then we need to get the id of the parent element, the <li>. The <li> is the element with the id that is the key we need.

Now we can use the key to remove the item from localStorage, and from the stickiesArray.

페이지에서 Sticky를 붙잡고 있는 를 제거해야 한다. 이렇게 해야 클릭했을 때 페이지에서 사라진다.

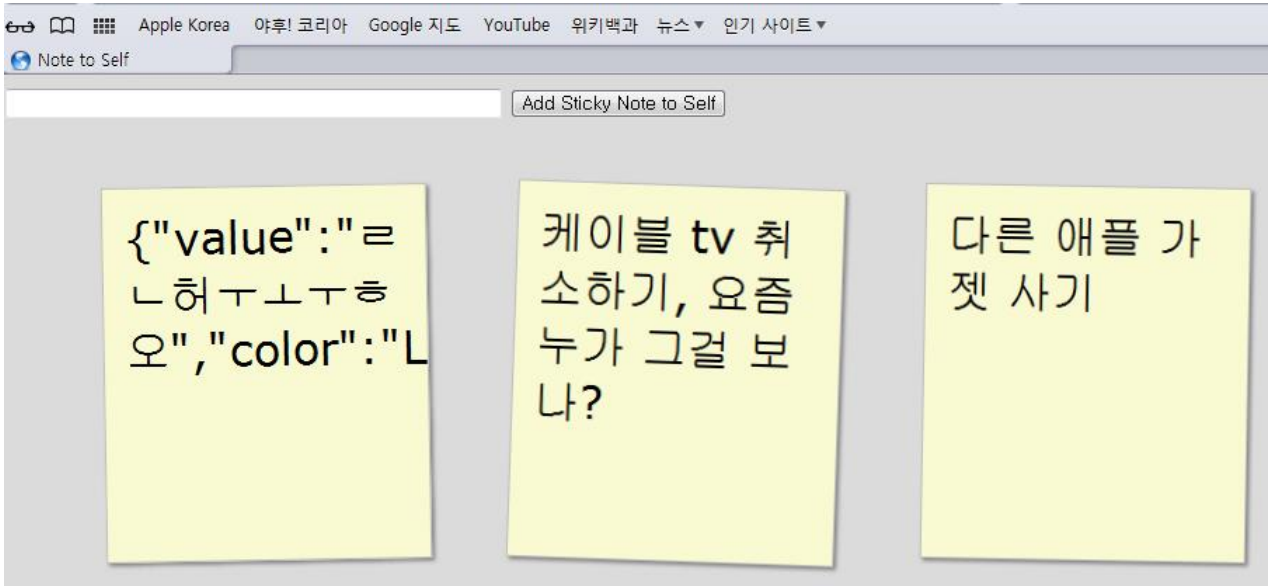
Delete the sticky from the DOM, too

삭제 방법을 완성하기 위해 **removeStickyFromDOM** 함수를 구현할 필요가 있다.

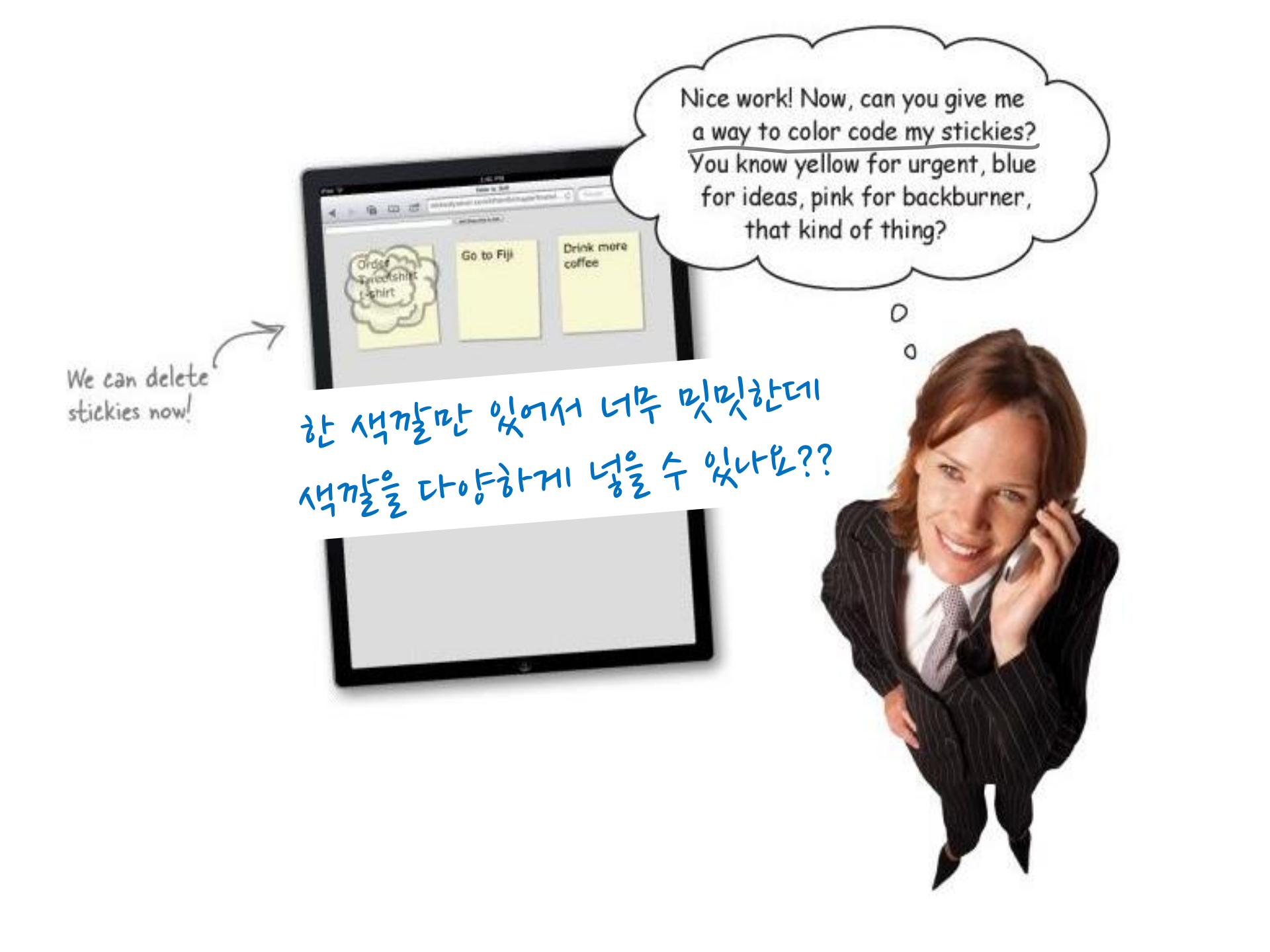


실습과제 15-5 Deleting stickies

Get all that code in, load the page, add and delete some stickies. Quit your browser, load it again, and give it a real run through!



http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch9/notetoself5.html


Nice work! Now, can you give me a way to color code my stickies? You know yellow for urgent, blue for ideas, pink for backburner, that kind of thing?
We can delete stickies now!
Go to Fiji
Drink more coffee
한 색깔만 있어서 너무 밋밋한데
색깔을 다양하게 넣을 수 있나요??

But of course we can!

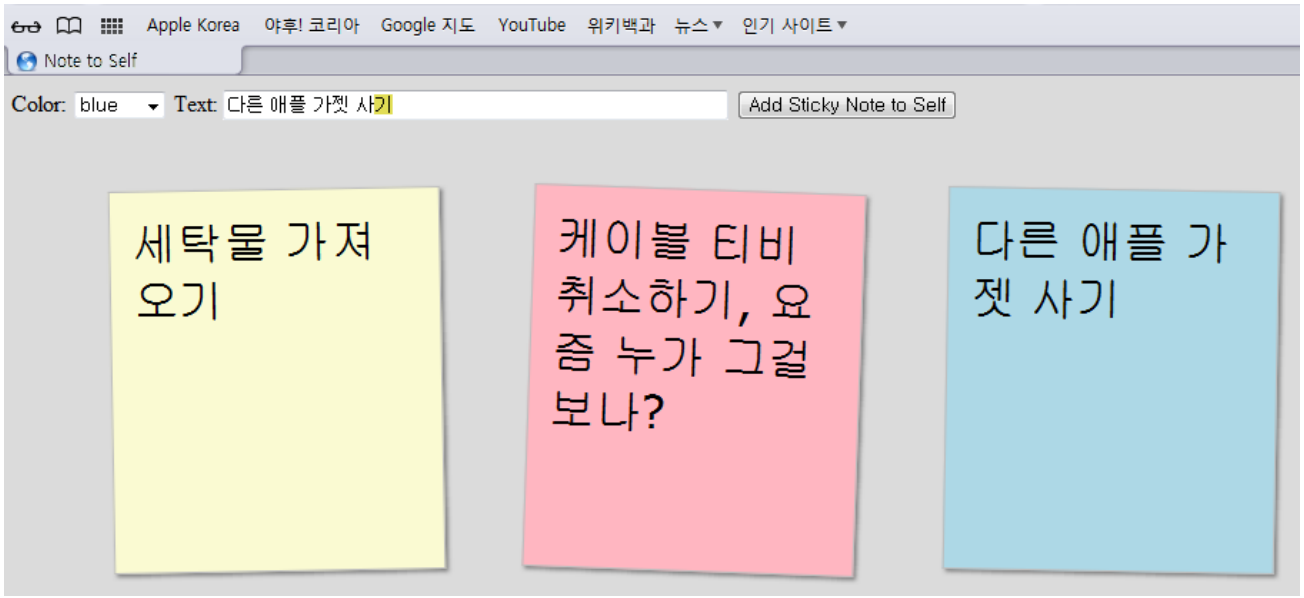
Come on, given your level of experience with this we're going to be able to knock this out.

How do we do it?

먼저, 노트의 텍스트를 저장할 객체를 생성한 다음, **JSON.stringify**를 사용하여 그것을 스트링으로 변환하여, Sticky item의 값으로 저장하면 된다.

Update the user interface so we can specify a color

목표:



먼저 쉬운 부분부터 하자: 색깔을 선택하는 메뉴 작성 (**notetoself.html**)

```
<html>
...
    <form>
        <label for="note_color">Color: </label>
        <select id="note_color">
            <option value="LightGoldenRodYellow">yellow</option>
            <option value="PaleGreen">green</option>
            <option value="LightPink">pink</option>
            <option value="LightBlue">blue</option>
        </select>
        <label for="note_text">Text:</label> <input type="text" id="note_text">
        <input type="button" id="add_button" value="Add Sticky Note to Self">
    </form>
...
</html>
```

We're only changing the form, the rest stays the same.

Notice the id of the <select>; we'll need that to grab the value of the selected option in the JavaScript.

We've added four sticky colors to choose from.

We'll add a label for the sticky text so the user knows what that field is for.

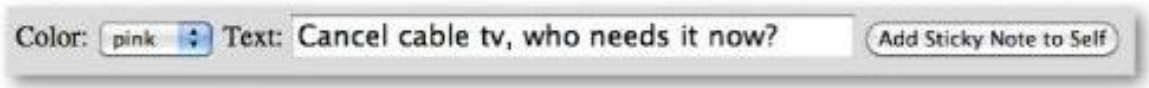
The value of each option is the name of a color we can just plug right into the style for our stickies.

And the rest of the form is the same.

JSON.stringify, it's not just for Arrays

Sticky의 텍스트와 함께 색깔을 저장하기 위해 stickiesArray에서 사용했던 것과 같은 기법을 사용할 수 있다:

즉 **텍스트와 색깔을 포함하는 객체**를 localStorage안에 Sticky에 대한 값으로써 저장할 수 있다.



We're going to take the values the user enters for the color and the sticky note text and package them up into a simple object.

And we'll store that in localStorage with the sticky's key.

```
var stickyObj = {
    "value": "Cancel cable tv, who needs it now?",
    "color": "LightPink"
};
```

Just like stickiesArray, we'll have to call JSON.stringify on the sticky value before we call localStorage.setItem to save the value.

Sticky 노트의 색깔을 저장하기 위한 **createSticky** 함수를 재작성해보자:

```
function createSticky() {
    var stickiesArray = getStickiesArray();
    var currentDate = new Date();
    var colorSelectObj = document.getElementById("note_color");
    var index = colorSelectObj.selectedIndex;
    var color = colorSelectObj[index].value;
    var key = "sticky_" + currentDate.getTime();
    var value = document.getElementById("note_text").value;
    var stickyObj = {
            "value": value,
            "color": color
    };
    localStorage.setItem(key, JSON.stringify(stickyObj));
    stickiesArray.push(key);
    localStorage.setItem("stickiesArray", JSON.stringify(stickiesArray));
    addStickyToDOM(key, stickyObj);
}
```

We do the usual thing to grab the value of the selected color option.

Then we use that color to create stickyObj: an object that contains two properties, the text of the sticky, and the color the user selected.

And, we JSON.stringify the stickyObj before we put it in localStorage.

이제 addStickyToDOM에 텍스트 스트링 대신에 객체를 전달한다. 이것은 addStickyToDOM도 변경해야 한다는 것을 의미한다. 그렇죠??

Using the new stickyObj

이제 **stickyObj**를 **addStickyToDOM**에 전달할 수 있다면 스트링 대신 **객체를 사용하도록** 함수를 변경할 필요가 있다:

```
function addStickyToDOM(key, stickyObj) {
    var stickies = document.getElementById("stickies");
    var sticky = document.createElement("li");
    sticky.setAttribute("id", key);

    sticky.style.backgroundColor = stickyObj.color;

    var span = document.createElement("span");
    span.setAttribute("class", "sticky");
    span.innerHTML = stickyObj.value;
    sticky.appendChild(span);
    stickies.appendChild(sticky);
    sticky.onclick = deleteSticky;
}
```

We need to change the parameter here to be the stickyObj rather than the text value of the sticky.

We get the color from the stickyObj we're passing into addStickyToDOM.

Notice that when we set the background color property in JavaScript, we specify it as backgroundColor, NOT background-color, like in CSS.

HTML element objects have a style property you can use to access the style of that element

And then we need to get the text value we're going to use in the sticky note from the object

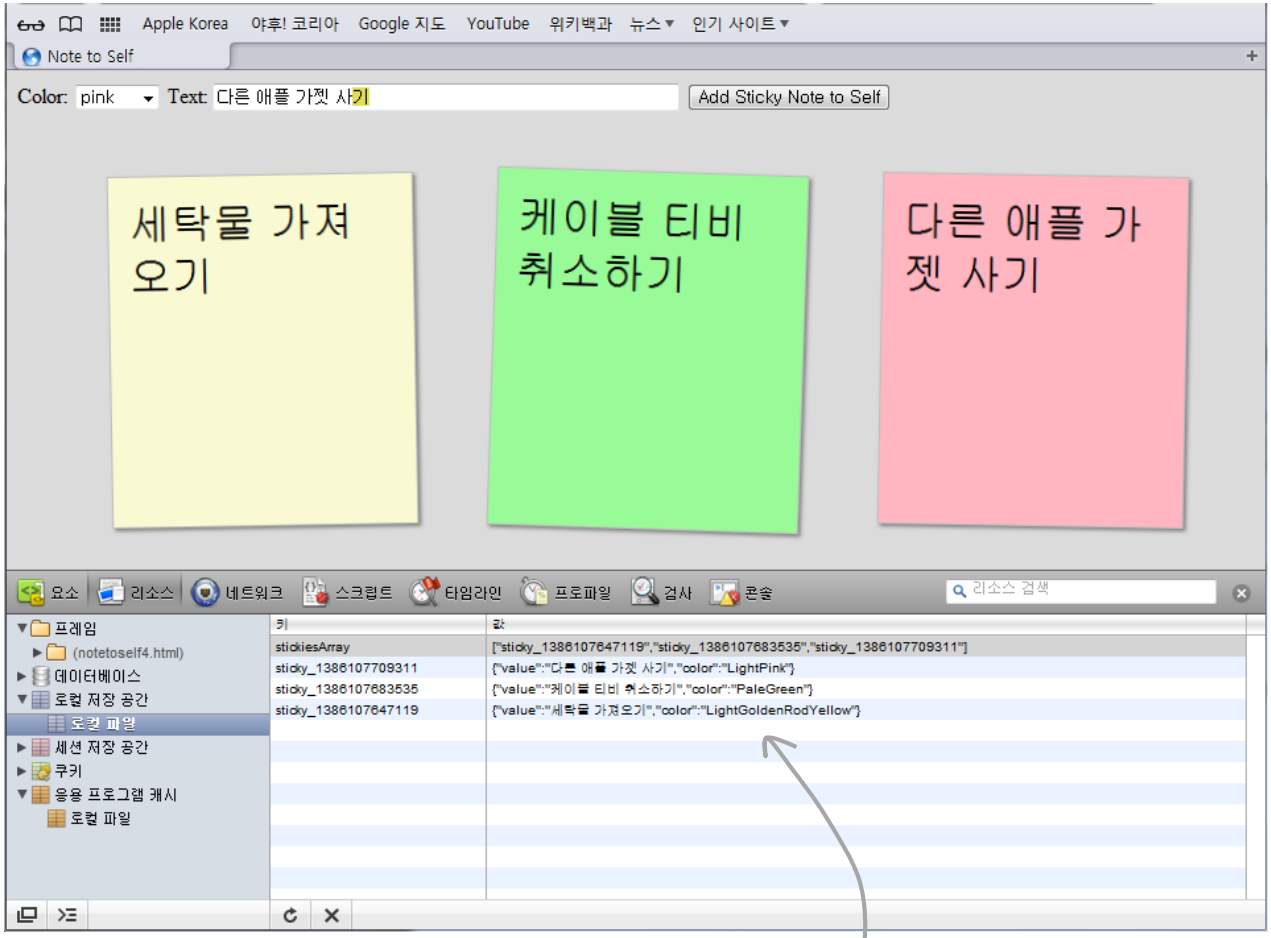
localStorage로부터 Sticky를 얻어와서 페이지가 처음 로드될 때 **addStickyToDOM** 함수에 Sticky를 전달해주는 **init 함수도 갱신**해야 한다:

```
function init() {
    var button = document.getElementById("add_button");
    button.onclick = createSticky;

    var stickiesAray = getStickiesArray();

    for (var i = 0; i < stickiesArray.length; i++) {
        var key = stickiesArray[i];
        var value = JSON.parse(localStorage[key]);
        addStickyToDOM(key, value);
    }
}
```

이제 localStorage로부터 Sticky 노트의 값을 가져올 때, 그것을 JSON.parse 해야 한다. 왜냐하면 더 이상 스트링이 아니라 객체이기 때문에.

그리고 스트링 대신에 그 객체를 addStickyToDOM에게 전달한다. 코드는 같지만 전달하는 내용물이 다르다!

실습과제 15-6



http://ksamkeun.dothome.co.kr/wp/hfhtml5/ch9/notetoself6.html

Try this at Home (or Blowing up your 5 Megabytes)


```
<html>
<head>
<script>

localStorage.setItem("fuse", "-");
while(true) {
    var fuse = localStorage.getItem("fuse");
    try {
        localStorage.setItem("fuse", fuse + fuse);
    } catch(e) {
        alert("Your browser blew up at" + fuse.length + " with exception: " + e);
        break;
    }
}
localStorage.removeItem("fuse");
</script>
</head>
<body>
</body>
</html>
```

Let start with a one-character string, with the key "fuse".

And just keep increasing its size...

...by doubling the string (by concatenating it with itself).

Then we'll try to write it back to localStorage.

If it blows up, we're done! We'll alert the user and get out of this loop.

And let's not leave a mess, so remove the item from localStorage.

Q & A