Recitation – Week 3

PRESENTED BY JEONGMIN LEE / SLIDE MADE BY JAIN PRANUT
Web storage
Using the Web Storage API

The Web Storage API provides mechanisms by which browsers can securely store key/value pairs, in a much more intuitive fashion than using cookies.
Using the Web Storage API

The Web Storage API provides mechanisms by which browsers can securely store key/value pairs, in a much more intuitive fashion than using cookies.

The keys and the values are always strings.

Question: What happens if the key is an integer?
Using the Web Storage API

The Web Storage API provides mechanisms by which browsers can securely store key/value pairs, in a much more intuitive fashion than using cookies.

The keys and the values are always strings.

**Question:** What happens if the key is an integer?

**A:** Integer key is automatically converted to string, like what objects do.
Types

**sessionStorage**
- maintains a separate storage area for each given origin
- available for the duration of the page session
- as long as the browser is open, including page reloads and restores

**localStorage**
- does the same thing
- *persists* even when the browser is closed and reopened

*SOURCE: MOZILLA DOCS*
Browser Support!

Newer versions of most browsers support Web Storage.

Although!!! You need to be sure.

Question: How would you check?
function storageAvailable(type) {
    try {
        var storage = window[type],
            x = '__storage_test__';
        storage.setItem(x, x);
        storage.removeItem(x);
        return true;
    } catch(e) {
        // Print Not supported!
        // Other reasons possible? YES!
        // Get error codes as e.code
        // Get error names as e.name
    }
}

window[type] ? What is this?
function storageAvailable(type) {
    try {
        var storage = window[type],
            x = '__storage_test__';
        storage.setItem(x, x);
        storage.removeItem(x);
        return true;
    } catch(e) {
        // Print Not supported!
        // Other reasons possible? YES!
        // Get error codes as e.code
        // Get error names as e.name
    }
}

window[type] ? What is this?
A: The type of storages; sessionStorage and localStorage
Getting values from storage

**Storage.getItem(key)** method is used to get a data item from storage.

```javascript
function setStyles() {
  var currentColor = localStorage.getItem('bgcolor');
  var currentFont = localStorage.getItem('font');
  var currentImage = localStorage.getItem('image');

  document.getElementById('bgcolor').value = currentColor;
  document.getElementById('font').value = currentFont;
  document.getElementById('image').value = currentImage;

  htmlElem.style.backgroundColor = '#' + currentColor;
  pElem.style.fontFamily = currentFont;
  imgElem.setAttribute('src', currentImage);
}
```
Setting values in storage

**Storage.setItem**(key, value) is used both to create new data items
- (if the data item already exists) update existing values.
- This takes two arguments
  — the key of the data item to create/modify
  — the value to store in it.

```javascript
function populateStorage() {
  localStorage.setItem('bgcolor', document.getElementById('bgcolor').value);
  localStorage.setItem('font', document.getElementById('font').value);
  localStorage.setItem('image', document.getElementById('image').value);
  setStyles();
}
```
Deleting data records

Web Storage also provides a couple of simple methods to remove data.

**Storage.removeItem**(key) takes a single argument

— the **key** of the data item you want to remove

— and removes it from the storage object for that domain.

**Storage.clear**() takes no arguments, and simply *empties the entire storage object* for that domain.