8.9 Web storage

Web Storage API

The ***Web Storage API*** provides storage objects that allow JavaScript programs to securely store key/value pairs in the web browser. The Web Storage API supports two storage objects:

1. The ***sessionStorage*** object stores key/value pairs for an origin that are only available for the duration of the session. Closing the browser or browser tab ends the session.
2. The ***localStorage*** object stores key/value pairs for an origin that are stored indefinitely.

An ***origin*** is a combination of scheme, hostname, and port number in a URL. Each of the following are examples of different origins:

- http://example.com/
- http://www.example.com/
- https://www.example.com/
- http://www.example.com:8080/

The browser stores the data for each origin separately and does not share the data between origins.

PARTICIPATION ACTIVITY 8.9.1: Web storage.

1) Refreshing a webpage begins a new session.
   - ○ True
   - ◉ False

Correct

A new session is created when a browser is closed and restarted.

2) Data that should remain after the user closes a web browser should be stored in sessionStorage.
   - ○ True
   - ◉ False

Correct

Data in sessionStorage is removed when the browser is closed, but data stored in localStorage is saved when the browser is closed.

3) The webpage from http://google.com/ cannot access web storage data from https://google.com/.

- (●) True
- ( ) False

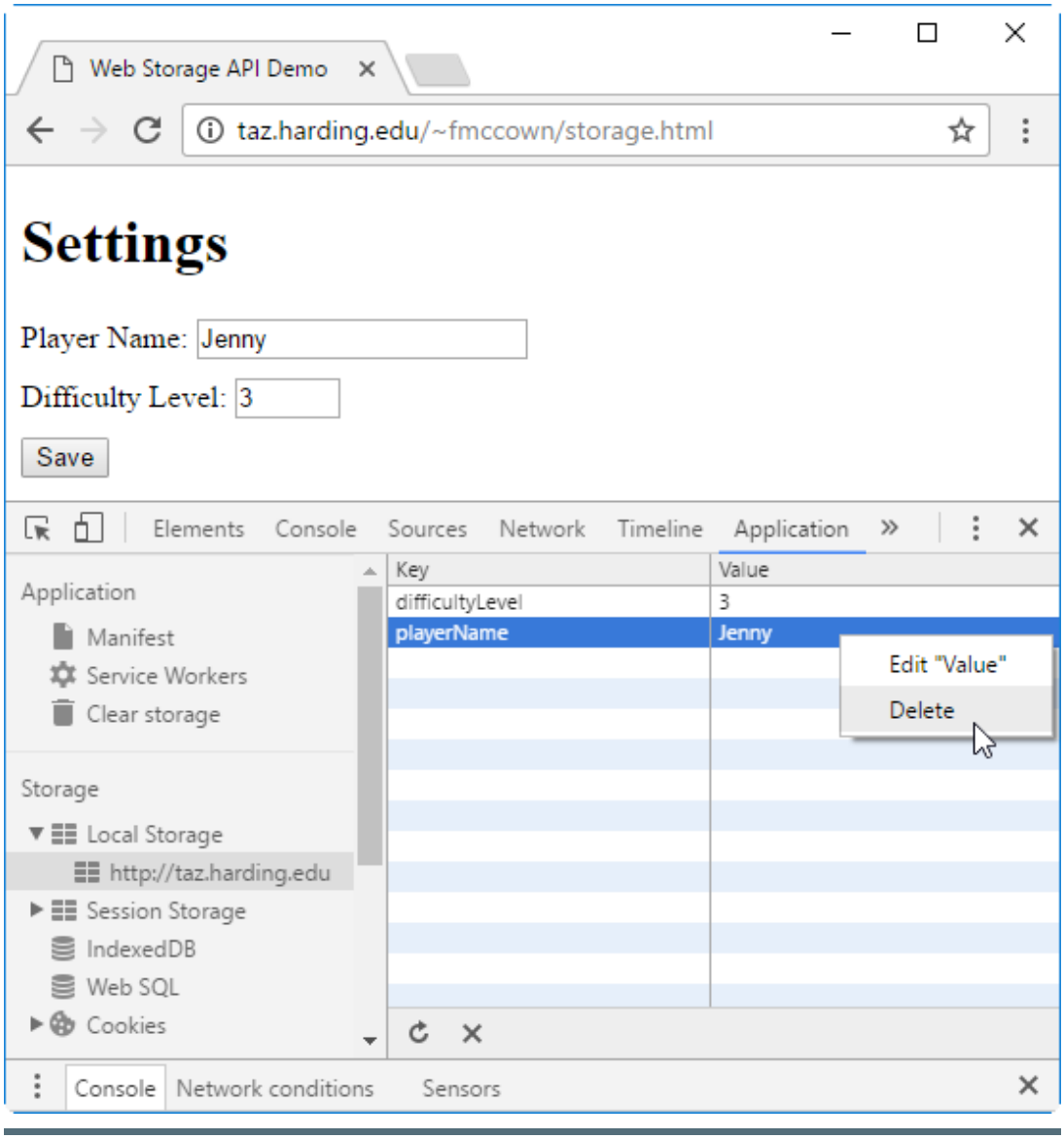
Correct

The scheme for the first webpage is `http`, and the second is `https`. Changes in schemes result in different origins, and different origins cannot share web storage data.

Feedback?

Chrome DevTools displays web storage

Most web browsers allow developers to see the key/value pairs stored in web storage. Chrome's DevTools displays the key/value pairs stored for the origin http://taz.harding.edu below and allows the developer to edit and delete key/value pairs. *Good practice is to avoid storing sensitive data like social security numbers, financial data, and passwords in web storage, because the values can be easily seen by others.*



Private browsing

Many web browsers allow users to browse the web privately using "incognito" mode or "private windows". When browsing privately, web storage may be disabled or may be cleared when the user stops browsing privately. See the Exploring further section for information on detecting the availability of localStorage.

Accessing web storage data

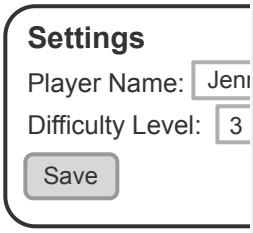
The `localStorage` and `sessionStorage` objects provide methods for storing data, retrieving data, and removing data:

- ***setItem(key, value)*** stores the `key` string and associated `value` string in storage.
- ***getItem(key)*** returns the value associated with the `key` in storage or `null` if the `key` does not exist.
- ***removeItem(key)*** removes the `key` and associated value from storage.
- ***clear()*** removes all keys and associated values from storage.

PARTICIPATION ACTIVITY 8.9.2: Storing and retrieving values from localStorage.

1 2 3 4 5 2x speed

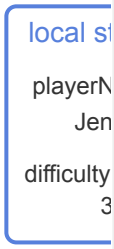
```
<h1>Settings</h1>
<label>Player Name:
   <input type="text" id="playerName" value="Player 1">
</label>
<label>Difficulty Level:
   <input type="number" min="1" max="3" id="diffLevel" value="1">
</label>
<input type="button" value="Save" id="saveBtn">
```




```
let playerNameWidget = document.getElementById("playerName");
let difficultyLevelWidget = document.getElementById("diffLevel");

if (localStorage.getItem("playerName")) {
   playerNameWidget.value = localStorage.getItem("playerName");
   difficultyLevelWidget.value = localStorage.getItem("difficultyLevel");
}

document.getElementById("saveBtn").addEventListener("click", function() {
   localStorage.setItem("playerName", playerNameWidget.value);
   localStorage.setItem("difficultyLevel", difficultyLevelWidget.value);
});
```



When the page is reloaded, the player name and difficulty level are loaded from localStorage using localStorage.getItem().

Captions

1. The form displays the default values "Player 1" and "1" in the browser.
2. localStorage.getItem() returns null because the playerName has not been previously saved in localStorage. The default HTML values remain in the browser.
3. The user changes the player name and difficulty level and clicks Save.
4. The Save button's click handler calls localStorage.setItem() to save the player name and difficulty level to local storage.
5. When the page is reloaded, the player name and difficulty level are loaded from localStorage using localStorage.getItem().

Feedback?

PARTICIPATION ACTIVITY 8.9.3: Web storage methods.

Refer to the animation above.

1) If local storage is empty and the Settings webpage is loaded into the browser, what does `localStorage.getItem("difficultyLevel")` return?

- "1"
- `null` (selected)
- `false`

Correct

Since nothing is stored in local storage when the webpage is first loaded, `null` is returned by `getItem()`.

2) Suppose localStorage is empty, and the Settings webpage is loaded in the browser. If the user clicks the Save button without changing the difficulty level, what does `localStorage.getItem("difficultyLevel")` return?

- "1" (selected)
- "3"
- `null`

Correct

The Save button's click handler stores the difficulty level widget's default value "1" for the key "difficultyLevel", so `getItem()` returns "1".

3) If the user sets the difficulty level to 2, clicks Save, closes the browser, re-opens the browser, and navigates to the Settings webpage, what difficulty level is displayed?

- 1
- 2 (selected)
- `null`

Correct

The Save button's click handler stored the difficulty level widget's value "2" in localStorage, so reloading the webpage displays the "2" stored in localStorage.

4) If localStorage stores the difficulty level "3" and then `localStorage.clear()` is called, what does `localStorage.getItem("difficultyLevel")` return?

- "1"
- "3"
- `null` (selected)

Correct

`null` is returned since `localStorage.clear()` removes all key/value pairs in localStorage.

Feedback?

CHALLENGE ACTIVITY 8.9.1: Web storage.

530096.4000608.qx3zqy7

Jump to level 1

1
2
3

Remove "bathroomColor" from sessionStorage.

```
1 sessionStorage.setItem("bathroomColor", "beige");
2
3 /* Your solution goes here */
4 sessionStorage.removeItem("bathroomColor");
```

1 2 **3**

Check Try again

Done. Click any level to practice more. Completion is preserved.

Testing value of "bathroomColor" on sessionStorage
Yours *Yours has no value*

Feedback?

Exploring further:

- Web Storage API from MDN
- Feature-detecting localStorage from MDN

How was this section? **Provide section feedback**