Tipe Web Storage

Web API menyediakan dua tipe Web Storage untuk kita gunakan, yakni sessionStorage dan localStorage.

Sebelum menggunakan Web Storage baik sessionStorage atau localStorage, kita perlu memastikan browser support terhadap fitur tersebut, dengan cara:

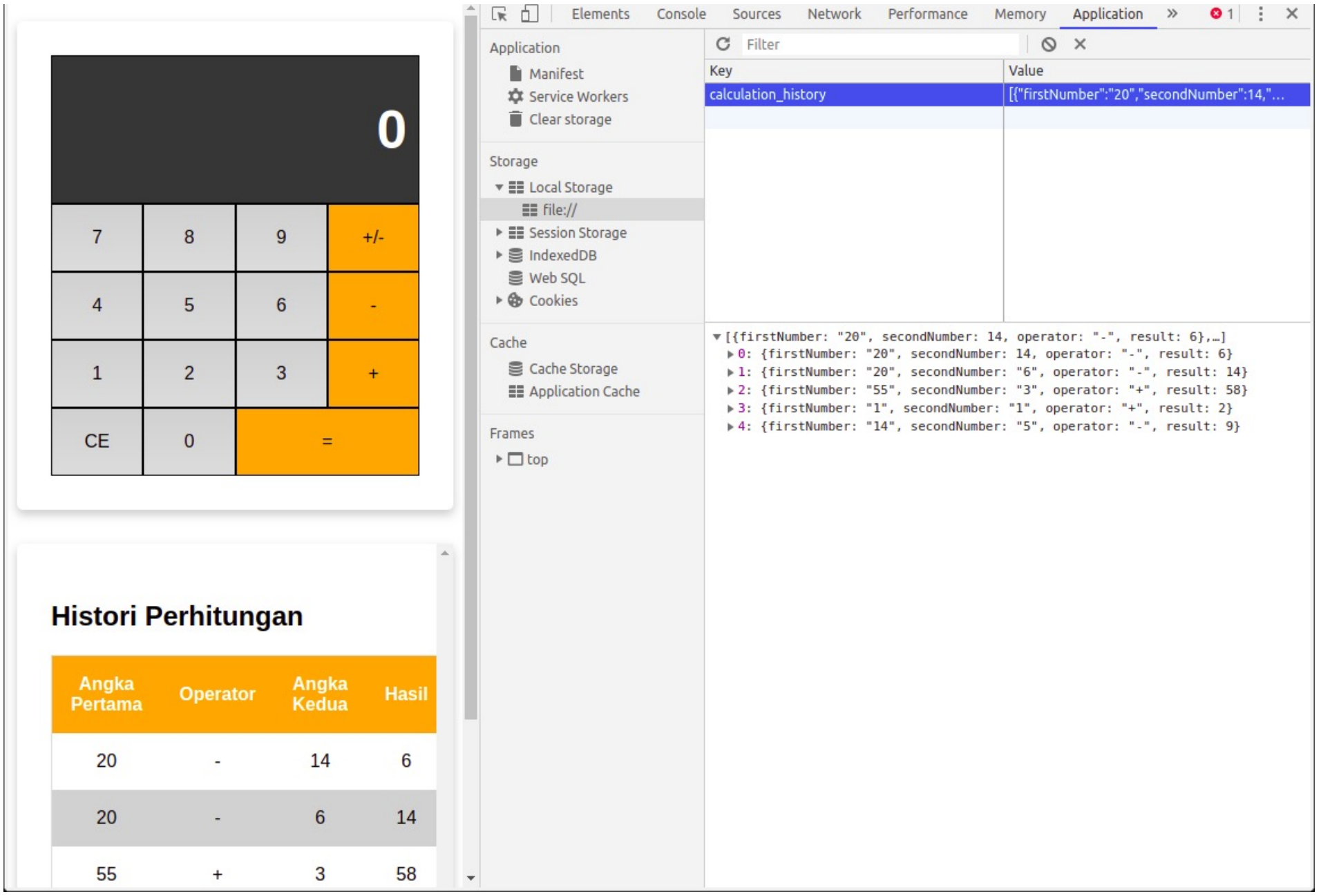
```
1. if (typeof(Storage) !== "undefined") {
2.   // Browser mendukung sessionStorage/localStorage.
3. } else {
4.   // Browser tidak mendukung sessionStorage/LocalStorage
5. }
```

Seharusnya browser yang ada pada saat ini sudah mendukung kedua fitur tersebut. Google Chrome dan Mozilla Firefox tentu memilikinya.

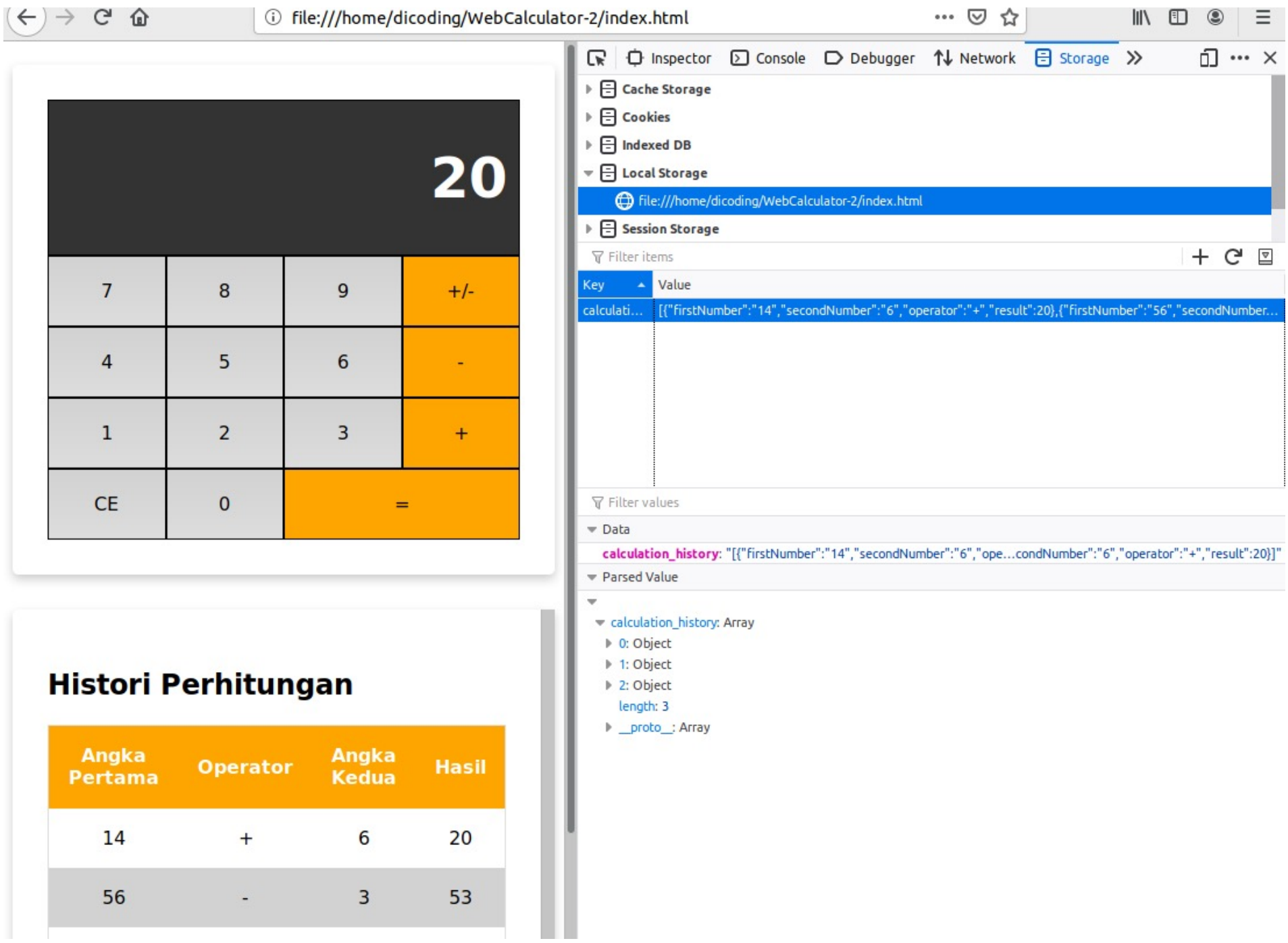
Pada sessionStorage atau localStorage data yang disimpan adalah nilai primitif seperti **number**, **boolean**, atau **string**. Namun kita juga dapat menyimpan data berbentuk objek dengan mengubahnya dalam bentuk **string**, begitu pula sebaliknya ketika kita ingin menggunakan datanya kembali.

Untuk menyimpan dan mengakses data pada storage, metode yang digunakan adalah *key-value*. Di sini *key* menjadi sebuah kunci untuk mengakses data pada storage.

Data yang disimpan pada Web Storage dapat kita lihat menggunakan DevTools pada *tab* **Application** (Google Chrome) atau *tab* **Storage** (Mozilla Firefox).



Data Local Storage pada Google Chrome



Data Local Storage pada Mozilla Firefox

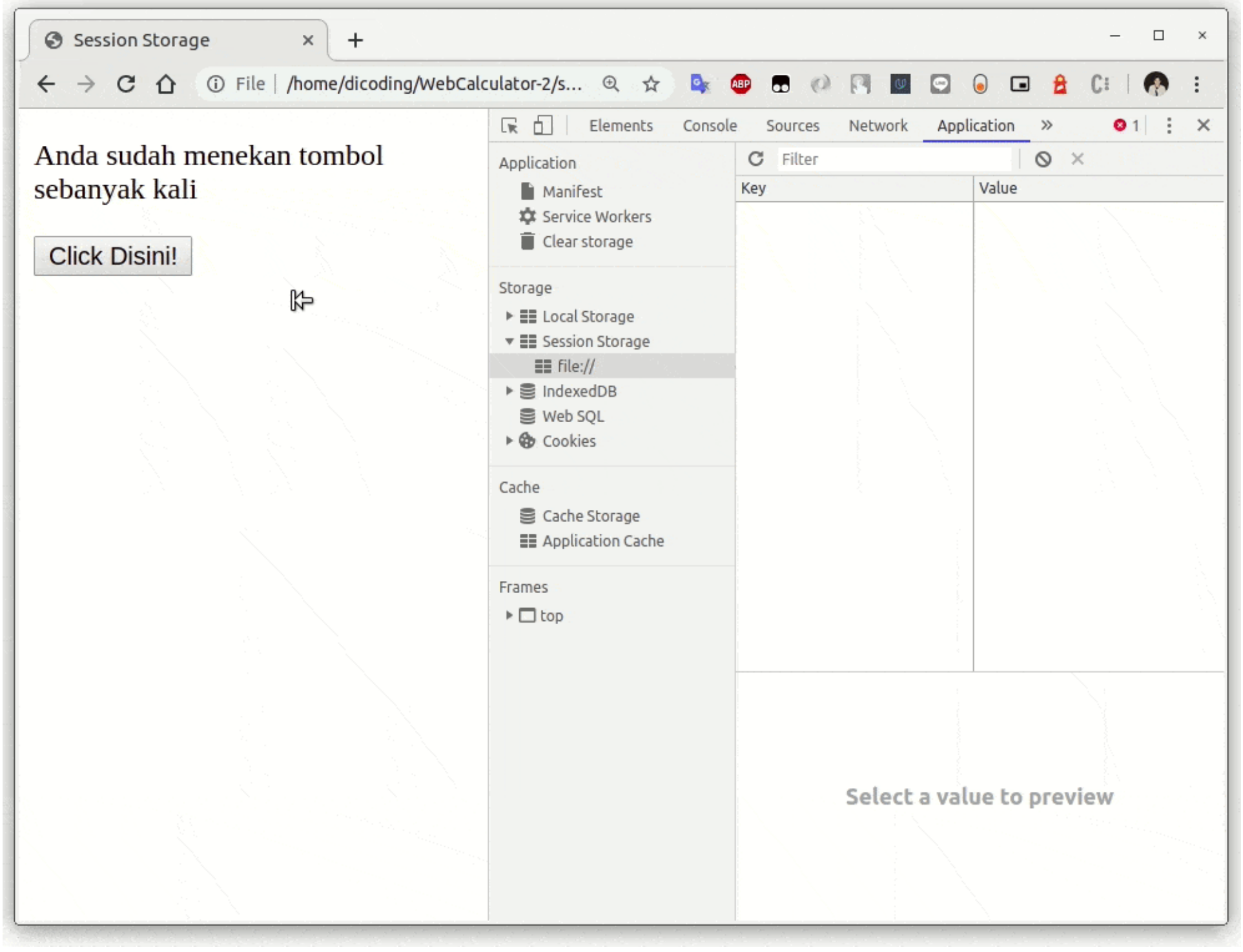
Session Storage

Tipe storage yang pertama adalah Session Storage yang mana digunakan untuk menyimpan data sementara pada browser. Data akan hilang ketika browser atau tab browser ditutup.

Untuk menggunakan Session Storage kita gunakan object `sessionStorage` , kemudian untuk menyimpan datanya gunakan method `setItem()` , method ini membutuhkan dua argumen yakni *key* dan nilai yang akan dimasukkan. Kemudian untuk mengakses data yang sudah dimasukkan kita gunakan method `getItem()` dan gunakan key sebagai argumen methodnya.

Berikut contoh penerapan dari session storage:

```
2.  <html>
3.
4.  <head>
5.      <title>Session Storage</title>
6.  </head>
7.
8.  <body>
9.      <p>Anda sudah menekan tombol sebanyak <span id="count"></span> kali</p>
10.     <button id="button">Click Disini!</button>
11.
12.     <script>
13.         const cacheKey = "NUMBER";
14.         if (typeof(Storage) !== "undefined") {
15.
16.             // pengecekkan apakah data sessionStorage dengan key NUMBER tersedia atau belum
17.             if (sessionStorage.getItem(cacheKey) === "undefined") {
18.                 // Jika belum maka akan atur dengan nilai awal yakni 0
```



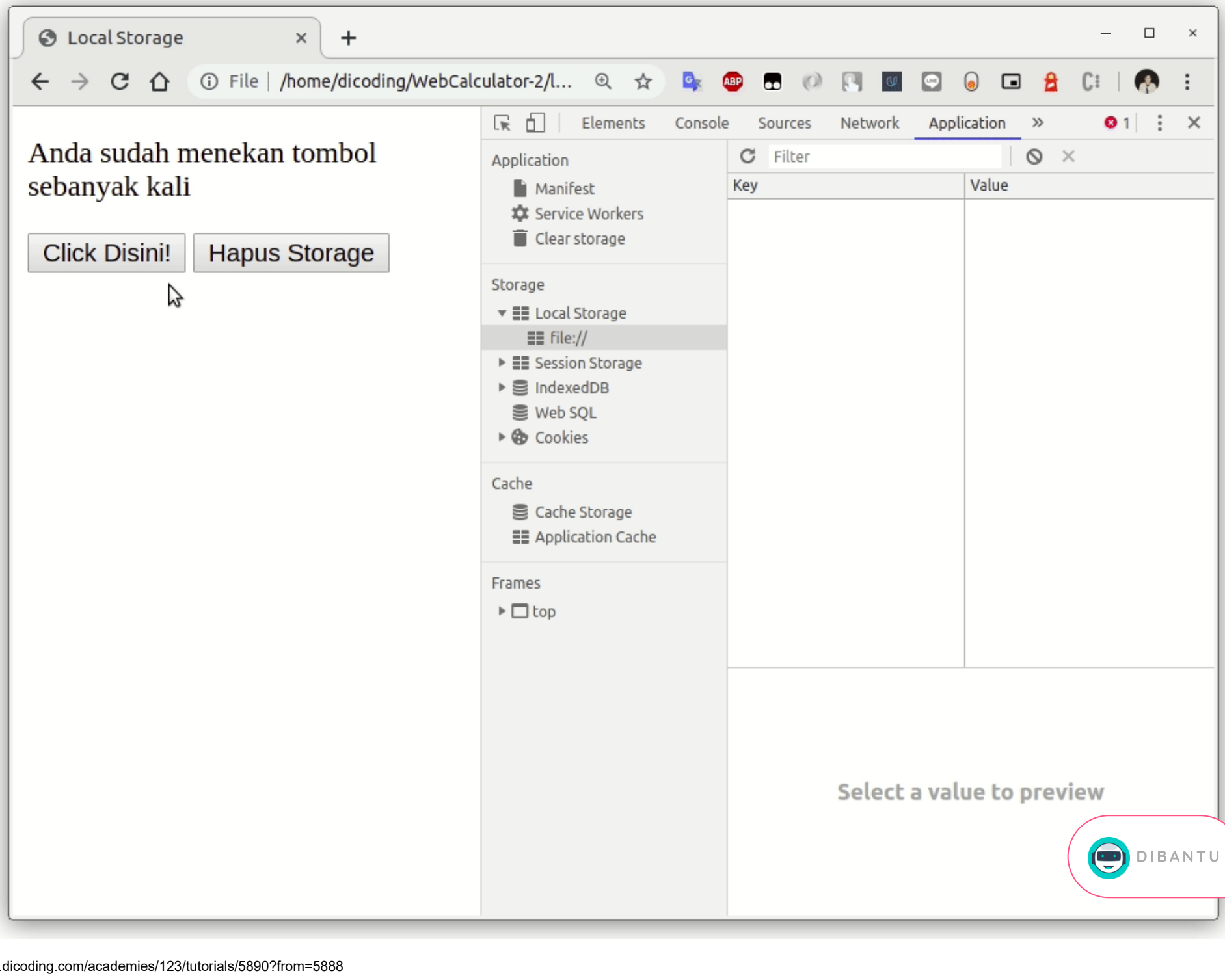
Bisa kita lihat bahwa data yang disimpan pada sessionStorage akan bertahan jika terjadi reload pada browser, namun data tersebut akan hilang apabila tab browser atau browser itu sendiri ditutup.

Local Storage

Untuk penggunaan localStorage identik dengan sessionStorage , perbedaanya storage ini diakses melalui object localStorage .

Berikut contoh penerapan dari local storage:

```
1.  <!DOCTYPE html>
2.  <html>
3.
4.  <head>
5.      <title>Local Storage</title>
6.  </head>
7.
8.  <body>
9.      <p>Anda sudah menekan tombol sebanyak <span id="count"></span> kali</p>
10.     <button id="button">Click Disini!</button>
11.     <button id="clear">Hapus Storage</button>
12.
13.     <script>
14.         const cacheKey = "NUMBER";
15.         if (typeof(Storage) !== "undefined") {
16.
17.             // pengecekkan apakah data sessionStorage dengan key NUMBER tersedia atau belum
18.             if (localStorage.getItem(cacheKey) === "undefined") {
```



memanggil method `localStorage.removeItem()`.

Bagaimana? Cukup mudah kan untuk menggunakan Web Storage? Selanjutnya kita akan coba terapkan `localStorage` pada Web Kalkulator yang kita buat untuk menampung riwayat kalkulasi yang pengguna lakukan.

Sebelumnya

Selanjutnya