**Menerapkan localStorage pada Kalkulator**

Sebelum kita menuliskan sintaks pada ***storage.js***, pastikan kita sudah menghubungkan berkas tersebut dengan ***index.html*** seperti berikut ini:

1. ....
2. <script src="assets/storage.js"></script>
3. <script src="assets/kalkulator.js"></script>
4. </body>

Pastikan juga kita menghubungkannya sebelum ***kalkulator.js*** karena kita akan menggunakan ***storage.js*** pada berkas ***kalkulator.js***. Alhasil, berkas ***storage.js*** perlu dimuat terlebih dahulu.

Setelah kita menghubungkan berkas JavaScript dengan HTML, buka berkas ***storage.js***. Kemudian buat variabel CACHE\_KEY dengan nilai**“calculation\_history”.**

1. **const** CACHE\_KEY = "calculation\_history";

Variabel ini digunakan sebagai *key* untuk mengakses dan menyimpan data pada localStorage.

Selanjutnya kita buat fungsi checkForStorage() dengan mengembalikan nilai boolean dari pengecekan fitur Storage pada browser.

1. **function** **checkForStorage**() {
2. **return** **typeof**(Storage) !== "undefined"
3. }

Fungsi tersebut akan kita gunakan di dalam *if statement* setiap fungsi transaksi pada localStorage.

Lalu kita buat juga fungsi untuk menyimpan data riwayat kalkulasi pada localStorage. Fungsi tersebut memiliki satu argumen yang merupakan data dari hasil kalkulasi yang nantinya akan dimasukkan ke dalam localStorage.

1. function putHistory(data) {
2. if (checkForStorage()) {
3. let historyData = null;
4. if (localStorage.getItem(CACHE\_KEY) === null) {
5. historyData = [];
6. } else {
7. historyData = JSON.parse(localStorage.getItem(CACHE\_KEY));
8. }
10. historyData.unshift(data);
12. if (historyData.length > 5) {
13. historyData.pop();
14. }
16. localStorage.setItem(CACHE\_KEY, JSON.stringify(historyData));
17. }
18. }

Banyak istilah kode yang asing pada kode di atas. Mari kita lihat satu per satu.

Yang pertama fungsi JSON.parse() yang mana digunakan untuk mengubah nilai objek dalam bentuk string kembali pada bentuk objek JavaScript. Kemudian JSON.stringify() digunakan untuk mengubah objek JavaScript ke dalam bentuk String. Seperti yang kita tahu, bahwa localStorage hanya dapat menyimpan data primitif seperti string, sehingga kita perlu mengubah objek ke dalam bentuk string jika ingin menyimpan ke dalam localStorage.

JSON sendiri adalah singkatan dari *JavaScript Object Notation*. JSON merupakan format yang sering digunakan dalam pertukaran data. Saat ini JSON banyak diandalkan karena formatnya berbasis teks dan relatif mudah dibaca.

Lalu di sana juga ada fungsi unshift(), fungsi ini digunakan untuk menambahkan nilai baru pada array yang ditempatkan pada awal index. Fungsi ini juga mengembalikan nilai panjang array setelah ditambahkan dengan nilai baru.

Fungsi pop() di atas merupakan fungsi untuk menghapus nilai index terakhir pada array, sehingga ukuran array historyData tidak akan pernah lebih dari 5. Hal ini kita terapkan agar riwayat kalkulasi yang muncul adalah lima hasil kalkulasi terakhir oleh pengguna.

Dari sini kita bisa mengetahui bahwa data yang disimpan pada localStorage adalah array yang berisi beberapa objek hasil kalkulasi, kemudian array tersebut diubah menjadi string. Struktur data tersebut dalam bentuk string nampak seperti ini:

1. [
2. {
3. "firstNumber": "23",
4. "secondNumber": "15",
5. "operator": "-",
6. "result": 8
7. },
8. {
9. "firstNumber": "23",
10. "secondNumber": "6",
11. "operator": "-",
12. "result": 17
13. }
14. ]

Selanjutnya kita buat fungsi untuk mendapatkan data dari localStorage. Kita namakan fungsi tersebut “showHistory().”

1. function showHistory() {
2. if (checkForStorage()) {
3. return JSON.parse(localStorage.getItem(CACHE\_KEY)) || [];
4. } else {
5. return [];
6. }
7. }

Fungsi ini mengembalikan nilai array dari localStorage jika sudah memiliki nilai sebelumnya melalui JSON.parse(). Namun jika localStorage masih kosong, fungsi ini akan mengembalikan nilai array kosong.

Lalu yang terakhir, kita tambahkan fungsi untuk merender data riwayat kalkulasi pada tabel HTML. Fungsi ini diberi nama dengan renderHistory().

1. function renderHistory() {
2. const historyData = showHistory();
3. let historyList = document.querySelector("#historyList");

6. // selalu hapus konten HTML pada elemen historyList agar tidak menampilkan data ganda
7. historyList.innerHTML = "";

10. for (let history of historyData) {
11. let row = document.createElement('tr');
12. row.innerHTML = "<td>" + history.firstNumber + "</td>";
13. row.innerHTML += "<td>" + history.operator + "</td>";
14. row.innerHTML += "<td>" + history.secondNumber + "</td>";
15. row.innerHTML += "<td>" + history.result + "</td>";

18. historyList.appendChild(row);
19. }
20. }

Pada akhir berkas ***storage.js***, panggil fungsi renderHistory(); agar data history muncul ketika website pertama kali dijalankan.

1. renderHistory();

Sampai saat ini, struktur kode pada ***storage.js*** akan tampak seperti berikut:

1. const CACHE\_KEY = "calculation\_history";
3. function checkForStorage() {
4. return typeof(Storage) !== "undefined";
5. }
7. function putHistory(data) {
8. if (checkForStorage()) {
9. let historyData = null;
10. if (localStorage.getItem(CACHE\_KEY) === null) {
11. historyData = [];
12. } else {
13. historyData = JSON.parse(localStorage.getItem(CACHE\_KEY));
14. }
16. historyData.unshift(data);
18. if (historyData.length > 5) {
19. historyData.pop();
20. }
22. localStorage.setItem(CACHE\_KEY, JSON.stringify(historyData));
23. }
24. }
26. function showHistory() {
27. if (checkForStorage) {
28. return JSON.parse(localStorage.getItem(CACHE\_KEY)) || [];
29. } else {
30. return [];
31. }
32. }
34. function renderHistory() {
35. const historyData = showHistory();
36. let historyList = document.querySelector("#historyList");
37. historyList.innerHTML = "";
39. for (let history of historyData) {
40. let row = document.createElement('tr');
41. row.innerHTML = "<td>" + history.firstNumber + "</td>";
42. row.innerHTML += "<td>" + history.operator + "</td>";
43. row.innerHTML += "<td>" + history.secondNumber + "</td>";
44. row.innerHTML += "<td>" + history.result + "</td>";
46. historyList.appendChild(row);
47. }
48. }
50. renderHistory();

Terakhir kita gunakan fungsi putHistory() yang sudah kita buat ketika kalkulator melakukan proses kalkulasi, tepatnya pada fungsi performCalculation() berkas ***kalkulator.js***.

Sebelum memanggil fungsi putHistory(), tentu kita harus menyiapkan data yang akan dikirimkan sebagai argumen fungsi tersebut. Pada performCalculation() kita buat variabel baru dengan nama history yang merupakan objek dari data history yang akan dikirimkan. Struktur objeknya tampak seperti berikut:

1. const history = {
2. firstNumber: calculator.firstNumber,
3. secondNumber: calculator.displayNumber,
4. operator: calculator.operator,
5. result: result
6. }

Setelah menyiapkan datanya, barulah kita bisa memanggil fungsi putHistory() dengan mengirimkan variabel history sebagai argumen fungsinya. Jangan lupa juga panggil fungsi renderHistory() agar riwayat kalkulasi langsung tampil pada tabel setelah kalkulasi dilakukan.

Sehingga sekarang struktur kode dari fungsi performCalculation() akan tampak seperti berikut:

1. function performCalculation() {
2. if (calculator.firstNumber == null || calculator.operator == null) {
3. alert("Anda belum menetapkan operator");
4. return;
5. }
7. let result = 0;
8. if (calculator.operator === "+") {
9. result = parseInt(calculator.firstNumber) + parseInt(calculator.displayNumber);
10. } else {
11. result = parseInt(calculator.firstNumber) - parseInt(calculator.displayNumber)
12. }
14. // objek yang akan dikirimkan sebagai argumen fungsi putHistory()
15. const history = {
16. firstNumber: calculator.firstNumber,
17. secondNumber: calculator.displayNumber,
18. operator: calculator.operator,
19. result: result
20. }
21. putHistory(history);
22. calculator.displayNumber = result;
23. renderHistory();
24. }

Pada tahap ini seharusnya kalkulator kita sudah dapat menampilkan riwayat kalkulasi yang dilakukan pengguna.

Pict menambahkan riwayat kalkulasi pada web kalkulator – bagian 3

Pict menambahkan riwayat kalkulasi pada web kalkulator – bagian 3 satu

Selamat, Anda sudah berhasil membuat aplikasi kalkulator berbasis web dengan baik. Good Job!