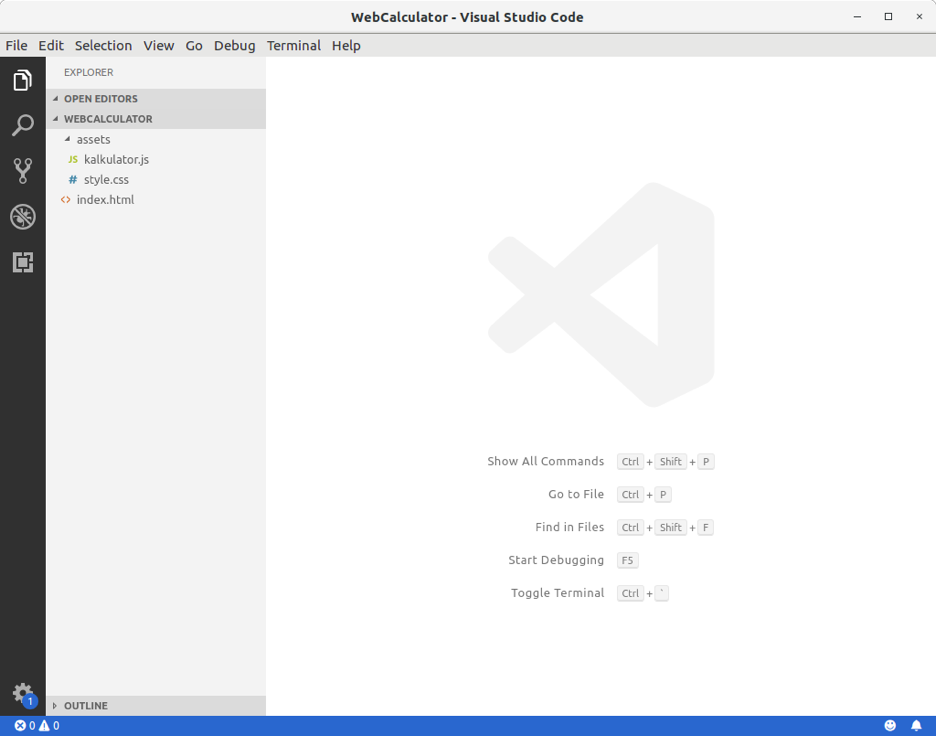
**Menerapkan JavaScript pada Web Kalkulator**

Sejauh ini kita sudah mempelajari bagaimana cara menggunakan JavaScript pada HTML, bagaimana menuliskan sintaks JavaScript, mengenal tipe data dan fungsi sintaks yang ada, hingga akhirnya memanipulasi dan memberikan event pada elemen HTML melalui DOM *Object*.

Dengan komponen yang sudah disebutkan tadi, sepertinya sekarang kita sudah cukup bekal untuk memberikan “nyawa” pada Web Kalkulator kita sehingga dapat berfungsi layaknya sebuah kalkulator pada umumnya. Let’s do it!

Silakan buka project Web Kalkulator dengan editor favorit kita.



Jika Anda mengikuti seluruh latihan sebelumnya, maka struktur projek tampak seperti gambar di atas. Kita juga sudah mencoba menghubungkan berkas JavaScript (***kalkulator.js***) dengan berkas HTML (***index.html***) lalu menuliskan sintaks dasar untuk menampilkan pesan pada console browser.

Selanjutnya kita akan bekerja full pada ***kalkulator.js***. Silakan hapus sintaks yang sudah kita buat sebelumnya.

1. console.log("Selamat Anda berhasil menggunakan JavaScript pada Website")

Langkah pertama adalah buatlah sebuah objek dengan nama calculator. Di dalamnya terdapat properti yang menggambarkan data dan kondisi dari kalkulatornya, seperti displayNumber, operator, firstNumber, dan waitingForSecondNumber. Sehingga kodenya akan nampak seperti ini:

1. const calculator = {
2. displayNumber: '0',
3. operator: null,
4. firstNumber: null,
5. waitingForSecondNumber: false
6. };

Kita gunakan objek ini sebagai tempat menyimpan data dan kondisi pada calculator, di mana angka yang muncul pada layar kalkulator selalu diambil dari data calculator.displayNumber.

Properti operator dan firstNumber kita gunakan nilai null terlebih dahulu karena properti tersebut akan diberikan nilai ketika pengguna melakukan aksi.

Dan properti waitingForSecondNumber merupakan kondisi di mana kalkulator sedang menunggu pengguna menentukkan angka kedua dalam melakukan perhitungan.

Setelah membuat *object* calculator, selanjutnya kita buat fungsi - fungsi umum yang dilakukan kalkulator seperti meng-*update* angka pada layar dan menghapus data pada kalkulator.

1. function updateDisplay() {
2. document.querySelector("#displayNumber").innerText = calculator.displayNumber;
3. }
5. function clearCalculator() {
6. calculator.displayNumber = '0';
7. calculator.operator = null;
8. calculator.firstNumber = null;
9. calculator.waitingForSecondNumber = false;
10. }

Lalu kita buat juga fungsi untuk memasukkan angka ke dalam nilai displayNumber kalkulator.

1. function inputDigit(digit) {
2. calculator.displayNumber += digit;
3. }

Kemudian kita buat variabel buttons dengan menginisialisasikan nilai seluruh elemen button yang ada, dan berikan event **click** pada tiap elemennya.

Untuk mendapatkan nilai seluruh elemen button kita gunakan querySelectorAll(“#button”) kemudian kita *looping* nilainya dan berikan event **click** pada tiap itemnya.

1. const buttons = document.querySelectorAll(".button");
2. for (let button of buttons) {
3. button.addEventListener('click', function(event) {
5. // mendapatkan objek elemen yang diklik
6. const target = event.target;
8. inputDigit(target.innerText);
9. updateDisplay()
10. });
11. }

Sehingga keseluruhan kode pada ***kalkulator.js*** akan tampak seperti ini:

1. const calculator = {
2. displayNumber: '0',
3. operator: null,
4. firstNumber: null,
5. waitingForSecondNumber: false
6. };
8. function updateDisplay() {
9. document.querySelector("#displayNumber").innerText = calculator.displayNumber;
10. }
12. function clearCalculator() {
13. calculator.displayNumber = '0';
14. calculator.operator = null;
15. calculator.firstNumber = null;
16. calculator.waitingForSecondNumber = false;
17. }
19. function inputDigit(digit) {
20. calculator.displayNumber += digit;
21. }

24. const buttons = document.querySelectorAll(".button");
25. for (let button of buttons) {
26. button.addEventListener('click', function(event) {
28. // mendapatkan objek elemen yang diklik
29. const target = event.target;
31. inputDigit(target.innerText);
32. updateDisplay()
33. });
34. }

Sekarang kita coba jalankan ***index.html*** pada browser, dan lihat fungsi kalkulator untuk yang pertama kali.

Pict menerapkan javascript pada web calculator 1

Uh sangat aneh bukan? Siapa yang ingin gunakan kalkulator seperti itu? Untuk saat ini tak apa. Setidanya kode yang kita tuliskan sudah berhasil berjalan dengan baik. Selanjutnya kita akan memperbaiki keanehan-keanehan yang ada satu per satu.

Saat ini kalkulator masih dapat menampilkan angka 0 di awal bilangan, hal itu tentu aneh dan tidak pernah terjadi pada kalkulator manapun terkecuali dalam menampilkan bilangan desimal. Untuk memperbaikinya, pada fungsi inputDigit(), tambahkan sebuah kondisi dimana jika displayNumber bernilai ‘0’, maka angka yang pertama dimasukkan pengguna akan menggantikan keseluruhan nilai displayNumber selain itu, lakukan seperti biasanya. Untuk melakukannya kita gunakan if-else statement.

1. function inputDigit(digit) {
2. if(calculator.displayNumber === '0') {
3. calculator.displayNumber = digit;
4. } else {
5. calculator.displayNumber += digit;
6. }
7. }

Dengan begitu kalkulator tidak akan menampilkan angka 0 diawal bilangan lagi.

Pict menerapkan javascript pada web calculatator 2

Kemudian kita akan membuat fungsi clear pada kalkulator berjalan dengan semestinya sehingga dalam mereset kalkulator kita tidak perlu melakukan reload pada browser seperti yang ditunjukkan pada gif di atas.

Pada event handler, kita tambahkan kondisi dimana ketika event target merupakan elemen yang menerapkan class clear maka kita akan panggil fungsi clearCalculator().

1. button.addEventListener('click', function(event) {
3. // mendapatkan objek elemen yang diklik
4. const target = event.target;
6. if(target.classList.contains('clear')) {
7. clearCalculator();
8. updateDisplay();
9. return;
10. }
12. inputDigit(target.innerText);
13. updateDisplay()
14. });

Kita bisa memanfaatkan event.classList untuk melihat nilai class apa saja dalam bentuk array yang ada pada element target, kemudian menggunakan contains() yang merupakan method dari array yang berguna untuk memastikan nilai yang terkandung di dalam array tersebut.

Jika kondisi if terpenuhi sudah dipastikan tombol tersebut adalah tombol clear, sehingga kita perlu memanggil fungsi clearCalculator() dan mengupdate display kalkulator. Jangan lupa juga untuk gunakan return statement agar fungsi event handler terhenti sehingga kode yang ada di bawahnya tidak ikut tereksekusi.

Pict menerapkan javascript pada web calculator 3

Setelah menerapkan kondisi tersebut maka seluruh kode pada ***kalkulator.js*** akan tampak seperti berikut:

1. const calculator = {
2. displayNumber: '0',
3. operator: null,
4. firstNumber: null,
5. waitingForSecondNumber: false
6. };
8. function updateDisplay() {
9. document.querySelector("#displayNumber").innerText = calculator.displayNumber;
10. }
12. function clearCalculator() {
13. calculator.displayNumber = '0';
14. calculator.operator = null;
15. calculator.firstNumber = null;
16. calculator.waitingForSecondNumber = false;
17. }
19. function inputDigit(digit) {
20. if (calculator.displayNumber === '0') {
21. calculator.displayNumber = digit;
22. } else {
23. calculator.displayNumber += digit;
24. }
25. }

28. const buttons = document.querySelectorAll(".button");
29. for (let button of buttons) {
30. button.addEventListener('click', function(event) {
32. *// mendapatkan objek elemen yang diklik*
33. const target = event.target;
35. if (target.classList.contains('clear')) {
36. clearCalculator();
37. updateDisplay();
38. return;
39. }
41. inputDigit(target.innerText);
42. updateDisplay()
43. });
44. }

Selanjutnya kita akan melengkapi beberapa fungsi lainnya yang ada pada kalkulator, yakni fungsi negative, operator, dan equals. Untuk itu, buat kondisi lainnya pada event handler, sehingga kode tampak seperti berikut:

1. button.addEventListener('click', function(event) {
3. *// mendapatkan objek elemen yang diklik*
4. const target = event.target;
6. if (target.classList.contains('clear')) {
7. clearCalculator();
8. updateDisplay();
9. return;
10. }
12. if(target.classList.contains('negative')) {
13. inverseNumber();
14. updateDisplay();
15. return;
16. }
18. if(target.classList.contains('equals')) {
19. performCalculation();
20. updateDisplay();
21. return;
22. }
24. if(target.classList.contains('operator')) {
25. handleOperator(target.innerText)
26. updateDisplay();
27. return;
28. }
30. inputDigit(target.innerText);
31. updateDisplay()
32. });

Jika kita membukanya sekarang, maka eror akan muncul ketika tombol - tombol fungsi dan operatornya ditekan.

Pict menerapkan javascript paa web calculator 4

Hal tersebut wajar karena kita belum mendefinisikan seluruh fungsi - fungsi yang kita dituliskan di atas. Dengan begitu mari kita buat fungsinya mulai dari inverseNumber()

1. function inverseNumber() {
2. if (calculator.displayNumber === '0') {
3. return;
4. }
5. calculator.displayNumber = calculator.displayNumber \* -1;
6. }

Fungsi inverseNumber() cukuplah simple karena kita hanya perlu melakukan perkalian displayNumber dengan -1, terkecuali jika displayNumber masih bernilai ‘0’ maka perkalian tidak akan dilakukan.

Pict menerapkan javascript pada web calculator 5

Sekarang tombol **"+/-"** sudah berfungsi dengan baik, selanjutnya kita akan membuat fungsi untuk menetapkan sebuah operator, baik itu + atau - pada kalkulator. Tuliskan fungsi berikut:

1. function handleOperator(operator) {
2. if (!calculator.waitingForSecondNumber) {
3. calculator.operator = operator;
4. calculator.waitingForSecondNumber = true;
5. calculator.firstNumber = calculator.displayNumber;
6. } else {
7. alert('Operator sudah ditetapkan')
8. }
9. }

Fungsi tersebut membutuhkan satu buah argument yang merupakan sebuah operator. Nilai operator tersebut bersumber dari innerText tombol operator yang menjadi event target. Secara prinsip fungsi ini bertujuan untuk menyimpan operator dan firstNumber dengan nilai displayNumber saat ini pada object calculator, hanya jika properti waitingForSecondNumber bernilai false. Namun jika waitingForSecondNumber bernilai true, browser akan menampilkan alert() dengan pesan **“Operator sudah ditetapkan”**

Selain itu, kita ubah juga logika yang terdapat pada fungsi inputDigit() dengan menambahkan kondisi pengecekkan terhadap nilai operator, dan displayNumber. Sehingga fungsi inputDigit() akan tampak seperti berikut:

1. function inputDigit(digit) {
2. if (calculator.waitingForSecondNumber && calculator.firstNumber === calculator.displayNumber) {
3. calculator.displayNumber = digit;
4. } else {
5. if (calculator.displayNumber === '0') {
6. calculator.displayNumber = digit;
7. } else {
8. calculator.displayNumber += digit;
9. }
10. }
11. }

Perubahan kode bertujuan untuk mengevaluasi "apakah operator baru saja ditetapkan?". Jika hasil evaluasi benar maka angka pada display sebelumnya akan dihapus ketika tombol angka pada kalkulator kembali ditekan. Lalu nilai angka pada display sekarang akan digunakan untuk bilangan kedua dalam melakukan kalkulasi.

Voila! Sekarang tombol operator sudah dapat menetapkan nilai operator pada object calculator.

Pict menerapkan javascript pada web calculator 6

Kita bisa lihat pada console bahwa sekarang nilai properti operator dan firstNumber tidak lagi null, begitu pula dengan properti waitingForSecondNumber yang sudah berubah menjadi true.

Kita buat fungsi terakhir yakni performCalculation(). Fungsi ini digunakan untuk melakukan kalkulasi terhadap nilai - nilai yang terdapat pada objek calculator, sehingga pastikan kalkulator sudah memiliki nilai operator dan firstNumber ketika fungsi ini dijalankan.

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. calculator.displayNumber = result;
15. }

Fungsi tersebut diawali dengan pengecekan nilai-nilai yang dibutuhkan untuk melakukan kalkulasi. Jika tidak terpenuhi maka proses akan dihentikan. Namun jika terpenuhi kalkulasi akan dilakukan.

Dalam melakukan kalkulasi terdapat pengecekan tipe operator apa yang akan dilakukan. Kita juga menggunakan parseInt() untuk mengubah nilai **string** menjadi **number**. Mengapa konversi tipe data dibutuhkan? Sejatinya kita menggunakan string dalam menampilkan nilai pada jendela browser, namun untuk proses kalkulasi kita membutuhkan number.

Setelah menambahkan fungsi tersebut, maka seluruh struktur kode pada berkas kalkulator.js akan tampak seperti ini:

1. const calculator = {
2. displayNumber: '0',
3. operator: null,
4. firstNumber: null,
5. waitingForSecondNumber: false
6. };
8. function updateDisplay() {
9. document.querySelector("#displayNumber").innerText = calculator.displayNumber;
10. }
12. function clearCalculator() {
13. calculator.displayNumber = '0';
14. calculator.operator = null;
15. calculator.firstNumber = null;
16. calculator.waitingForSecondNumber = false;
17. }
19. function inputDigit(digit) {
20. if (calculator.waitingForSecondNumber && calculator.firstNumber === calculator.displayNumber) {
21. calculator.displayNumber = digit;
22. } else {
23. if (calculator.displayNumber === '0') {
24. calculator.displayNumber = digit;
25. } else {
26. calculator.displayNumber += digit;
27. }
28. }
29. }
31. function inverseNumber() {
32. if (calculator.displayNumber === '0') {
33. return;
34. }
35. calculator.displayNumber = calculator.displayNumber \* -1;
36. }
38. function handleOperator(operator) {
39. if (!calculator.waitingForSecondNumber) {
40. calculator.operator = operator;
41. calculator.waitingForSecondNumber = true;
42. calculator.firstNumber = calculator.displayNumber;
43. } else {
44. alert('Operator sudah ditetapkan')
45. }
46. }
48. function performCalculation() {
49. if (calculator.firstNumber == null || calculator.operator == null) {
50. alert("Anda belum menetapkan operator");
51. return;
52. }
54. let result = 0;
55. if (calculator.operator === "+") {
56. result = parseInt(calculator.firstNumber) + parseInt(calculator.displayNumber);
57. } else {
58. result = parseInt(calculator.firstNumber) - parseInt(calculator.displayNumber)
59. }
61. calculator.displayNumber = result;
62. }

65. const buttons = document.querySelectorAll(".button");
66. for (let button of buttons) {
67. button.addEventListener('click', function(event) {
69. // mendapatkan objek elemen yang diklik
70. const target = event.target;
72. if (target.classList.contains('clear')) {
73. clearCalculator();
74. updateDisplay();
75. return;
76. }
78. if (target.classList.contains('negative')) {
79. inverseNumber();
80. updateDisplay();
81. return;
82. }
84. if (target.classList.contains('equals')) {
85. performCalculation();
86. updateDisplay();
87. return;
88. }
90. if (target.classList.contains('operator')) {
91. handleOperator(target.innerText)
92. updateDisplay();
93. return;
94. }
96. inputDigit(target.innerText);
97. updateDisplay()
98. });
99. }

Jika kita buka ***index.html*** sekarang, seharusnya kalkulator sudah dapat melakukan kalkulasi.

Pict menerapkan javascript pada web calculator 7

Selamat! Sejauh ini kita sudah bisa membuat aplikasi web dengan memanfaatkan pengetahuan yang sudah kita pelajari. Selanjutnya kita akan mengenal salah satu Web API yang menarik untuk diterapkan pada project kalkulator kita.