

Web Dev Bootcamp 2024/25

JavaScript - Lab 06

09.11.2024

Sponsored by:



Agenda

1. Diskusi tentang tugas minggu lalu
2. Document Object Model (DOM)
3. Built-in form validation
4. Prep & To-Do untuk next session
5. Organisasi dan feedback

Diskusi tentang tugas dari minggu lalu

- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/A_first_splash#example_%E2%80%94_guess_the_number_game
 - Variables ('let' vs 'const')
 - Functions
 - Operators
 - Arithmetic operators
 - Comparison operators
 - Conditionals
 - Events
 - [Event reference](#) / Cheat sheet
 - [Introduction to events - Learn web development | MDN](#)
 - Loops
 - Objects

Document Object Model (DOM)

DOM adalah objek/representasi dari HTML yang bisa digunakan oleh suatu web application programming language (in our case: JavaScript) untuk mengakses/mengontrol HTML elements.

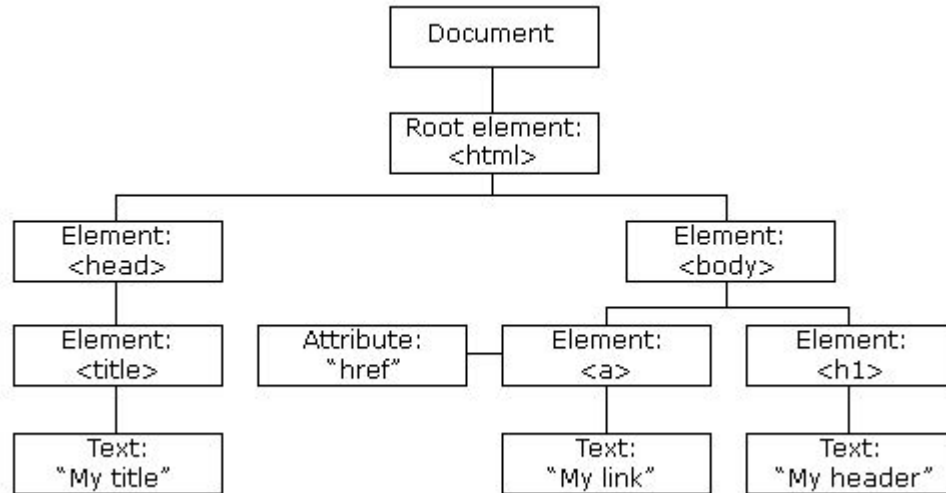
Suatu HTML element akan selalu memiliki 1:1 mapping dengan sebuah **node**.

```
92     function setGameOver() {  
93         guessField.disabled = true;  
94         guessSubmit.disabled = true;  
95         resetButton = document.createElement('button');  
96         resetButton.textContent = 'Start new game';  
97         document.body.appendChild(resetButton);  
98         resetButton.addEventListener('click', resetGame);  
99     }
```

- guessField
- guessSubmit
- resetButton
- document

Source: <https://github.com/mdn/learning-area/blob/main/javascript/introduction-to-js-1/first-splash/number-guessing-game.html#L92-L99>

Document Object Model (DOM) – DOM Tree



Source: https://www.w3schools.com/js/pic_htmltree.gif

Mirip seperti HTML elements, DOM nodes ini juga memiliki:

- Struktur Parent-Child
- Properties
 - Own properties
 - Inherited properties

More on this:

- [Chrome DevTools DOM](#)
- [MDN: Introduction to DOM](#)

Document Object Model (DOM) – DOM Tree

The screenshot displays a web browser window with the URL `mdn.github.io/learning-area/javascript/introduction-to-js-1/first-splash/number-guessing-game.html`. The page content includes the heading "Number guessing game", a paragraph "We have selected a random number between 1 and 100. See if you can guess it in 10 turns or fewer. High or too low.", and an input field with the placeholder text "Enter a guess:". The browser's developer tools are open, showing the DOM tree on the left and the Properties panel on the right. The DOM tree shows the root `<html>` element with attributes `lang="en-US"` and `scroll`. The `<body>` element contains a heading `<h1>`, a paragraph `<p>`, a form `<div class="form">`, a result display `<div class="resultParas">`, and a script `<script>`. The Properties panel shows various attributes and styles for the selected `<p>` element, including `accessKey`, `align`, `attributeStyleMap`, `attributes`, `autocapitalize`, `autofocus`, `baseURI`, `childNodes`, `children`, `classList`, `className`, `clientHeight`, `clientLeft`, `clientTop`, `clientWidth`, `contentEditable`, `currentCSSZoom`, `dataset`, `dir`, `draggable`, `elementTiming`, `enterKeyHint`, `firstChild`, `hidden`, `id`, `inert`, `innerHTML`, `innerText`, `inputMode`, and `isConnected`.

Number guessing game

We have selected a random number between 1 and 100. See if you can guess it in 10 turns or fewer. High or too low.

Enter a guess:

```
<!DOCTYPE html>
<html lang="en-US" scroll>
  <head>
  </head>
  <body>
    <h1>Number guessing game</h1>
    <p>== $0</p>
    <div class="form"></div>
    <div class="resultParas"></div>
    <script></script>
    <div id="janus-extension-installed" style="display: none;">
    </div>
  </body>
</html>
```

Styles

Filter

Show all

accessKey: ""

align: ""

attributeStyleMap: StylePropertyMap {size: 0}

attributes: NamedNodeMap {length: 0}

autocapitalize: ""

autofocus: false

baseURI: "https://mdn.github.io/learning-area/javascript/introduction-to-js-1/first-splash/number-guessing-game.html"

childNodes: NodeList [text]

children: HTMLCollection []

classList: DOMTokenList [value: '']

className: ""

clientHeight: 54

clientLeft: 0

clientTop: 0

clientWidth: 480

contentEditable: "inherit"

currentCSSZoom: 1

dataset: DOMStringMap {}

dir: ""

draggable: false

elementTiming: ""

enterKeyHint: ""

firstChild: text

hidden: false

id: ""

inert: false

innerHTML: "We have selected a random number between 1 and 100. See if you can guess it in 10 turns or fewer. High or too low."

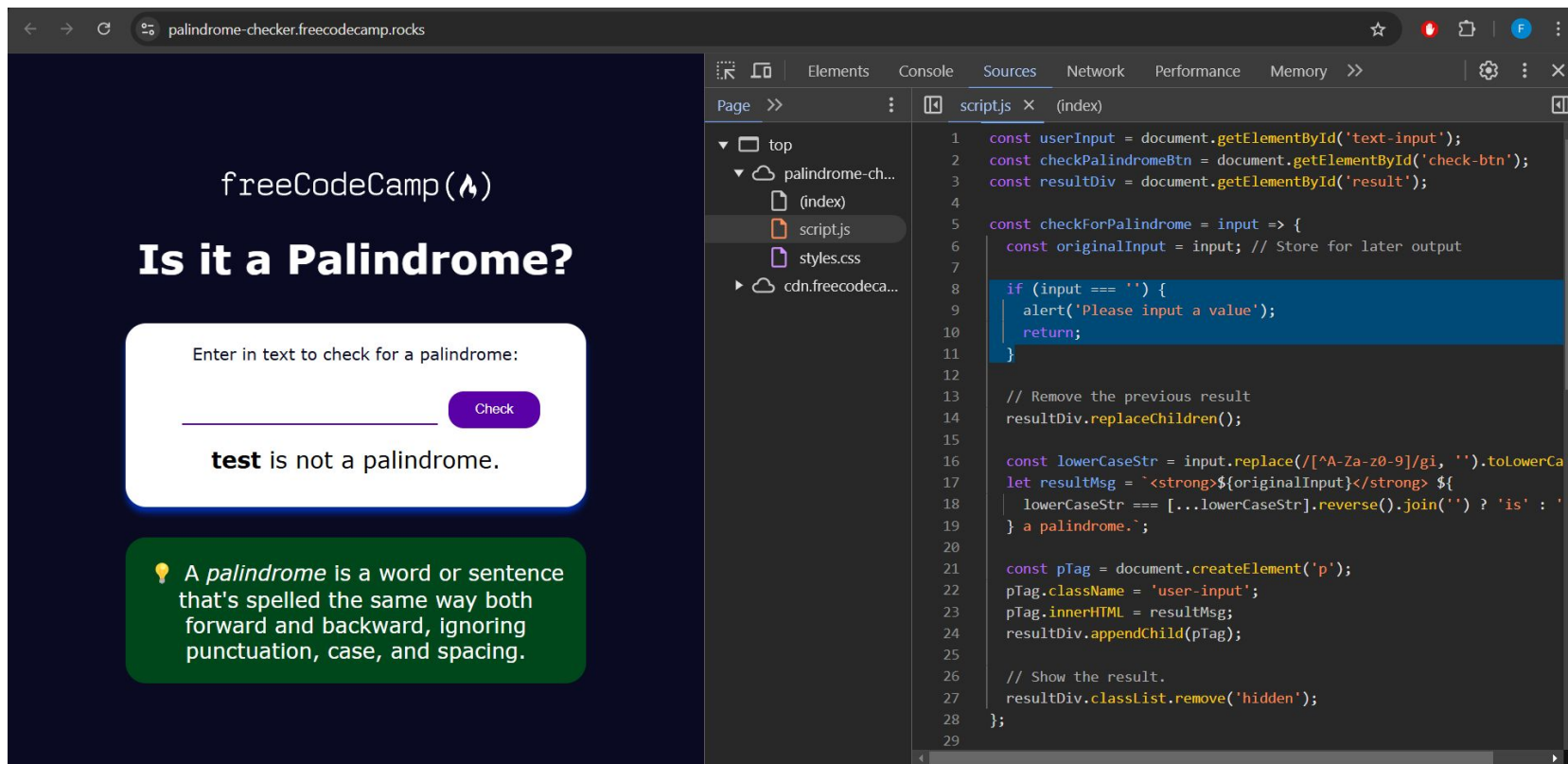
innerText: "We have selected a random number between 1 and 100. See if you can guess it in 10 turns or fewer. High or too low."

inputMode: ""

isConnected: true

Source: <https://mdn.github.io/learning-area/javascript/introduction-to-js-1/first-splash/number-guessing-game.html>

Form Validation



The screenshot displays a web browser window at the URL `palindrome-checker.freecodecamp.rocks`. The page features a dark blue background with the text "freeCodeCamp (🔥)" and "Is it a Palindrome?". Below this, there is a white input field with the placeholder text "Enter in text to check for a palindrome:" and a "Check" button. The input field contains the text "test", and the output below it states "test is not a palindrome.". A green box at the bottom provides a definition: "A *palindrome* is a word or sentence that's spelled the same way both forward and backward, ignoring punctuation, case, and spacing."

The browser's developer tools are open, showing the "Sources" tab. The file `script.js` is selected, and the following JavaScript code is visible:

```
1 const userInput = document.getElementById('text-input');
2 const checkPalindromeBtn = document.getElementById('check-btn');
3 const resultDiv = document.getElementById('result');
4
5 const checkForPalindrome = input => {
6   const originalInput = input; // Store for later output
7
8   if (input === '') {
9     alert('Please input a value');
10    return;
11  }
12
13  // Remove the previous result
14  resultDiv.replaceChildren();
15
16  const lowerCaseStr = input.replace(/[^A-Za-z0-9]/gi, '').toLowerCase();
17  let resultMsg = `<strong>${originalInput}</strong> ${
18    lowerCaseStr === [...lowerCaseStr].reverse().join('') ? 'is' : '
19  } a palindrome.`;
20
21  const pTag = document.createElement('p');
22  pTag.className = 'user-input';
23  pTag.innerHTML = resultMsg;
24  resultDiv.appendChild(pTag);
25
26  // Show the result.
27  resultDiv.classList.remove('hidden');
28
29  };
```

Form Validation

Di contoh sebelumnya tadi, itu hanya memvalidasi dari 1 value saja.

Bagaimana kalau misalnya ada banyak values/constraints yang perlu dipenuhi? 🤔

- **Client-side validation (Frontend)**
 - **Built-in form validations**
 - **Custom form validations**
- Server-side validation (Backend)

10 mins. break :)

Form Validation: RegEx

Regular Expression adalah format pattern yang bisa digunakan sebagai constraint untuk suatu input teks/String.

Contoh penggunaan:

- Password harus memiliki angka dan huruf kapital
- Nama tidak boleh memiliki special character
- Email yang valid harus berformatkan seperti '[abcd@email.com](#)'

More on this: [MDN Regular Expression](#)

Coba-coba Regex: [regex101.com](#)

Built-in Form Validation: HTML

- **required**: Menspesifikasikan apabila suatu form input field harus terisi sebelum bisa disubmit.
- **minlength** dan **maxlength**: Menspesifikasikan panjang/length dari suatu input teks.
- **min**, **max**, dan **step**: Menspesifikasikan value min/max/increment (++) dari suatu input numerical.
- **type**: Menspesifikasikan tipe dari input yang dimasukkan ([Link](#)).
- **pattern**: Menspesifikasikan pattern regular expression (regex) yang harus dipenuhi.

More on this: [MDN Client-side Form Validation](#)

Built-in Form Validation: JavaScript

- Menggunakan conditions (`if ... else ...`)
- Menggunakan [Constraint Validation API](#)
 - Bisa digunakan untuk beberapa HTML elements (e.g. `<buttons>`, `<input>`, `<textarea>`, dll.)
 - Ekspansi dari built-in HTML form validations
 - Customize warning messages/toasts

Let's try it out:

- [MDN Validating Forms using JavaScript](#) - [CodePen.io](#)
- [MDN Form Validation Tests](#)

To-Do untuk next session

1. Masing-masing team member fork repo team mereka ([Link](#)).
2. Mengerjakan tugas pertama untuk certificate freeCodeCamp JavaScript Algorithms & Data Structures ([Link](#)):
 - a. Pass semua requirements dari freeCodeCamp.
 - b. Juga commit dan push codes nya ke forked repo team kalian.
3. Familiarkan diri dengan cara debugging JavaScript langsung di terminal IDE, atau lewat browser ([Link](#)).

Organisasi dan feedback

1. Feedback: <https://forms.gle/YQxtfUdkX5Ej6yxD9>
2. **Session untuk Sabtu, 16 November 2024 akan dicancel,** tapi recording akan diberikan lebih awal sebagai penggantinya. Kalau ada pertanyaan, silahkan ping Sari langsung di Discord :)
3. Untuk tugasnya, apa mau build from scratch, atau mau disiapin dulu basic HTML dan CSS nya oleh Sari?