

WSTĘP DO JAVASCRIPT

Aplikacja kalkulator z zaawansowanym GUI w postaci HTML-a.

Wykorzystanie zewnętrznego CSS

- Jak rozdzielić kolejne warstwy aplikacji, kaskadowe arkusze stylów, kod HTML oraz JS

Należy osadzić skrypty oraz style wewnątrz pliku HTML.

```
<html>  
  <head>  
    <link rel="stylesheet" href="..." />  
    <script src="..." async></script>  
  </head>  
  <body>  
  </body>  
</html>
```

Jak należy zacząć organizować pliki, nazywać katalogi.

Wykorzystanie gotowych stylów, framework do CSS – Bootstrap 4.0

- **Pobranie odpowiednich plików, podpięcie ich do pliku index.html**
 - <https://getbootstrap.com/docs/4.5/getting-started/download/>
- **Wykorzystanie „grid system” do stworzenia układu zawartości strony:**
- **Wykorzystanie stylów do uzyskania atrakcyjnego wyglądu.**
 - `<input type="button" class="btn btn-outline-primary" />`

```
<div class="container">
  <div class="row">
    <div class="col-sm">
      One of three columns
    </div>
    <div class="col-sm">
      One of three columns
    </div>
    <div class="col-sm">
      One of three columns
    </div>
  </div>
</div>
```

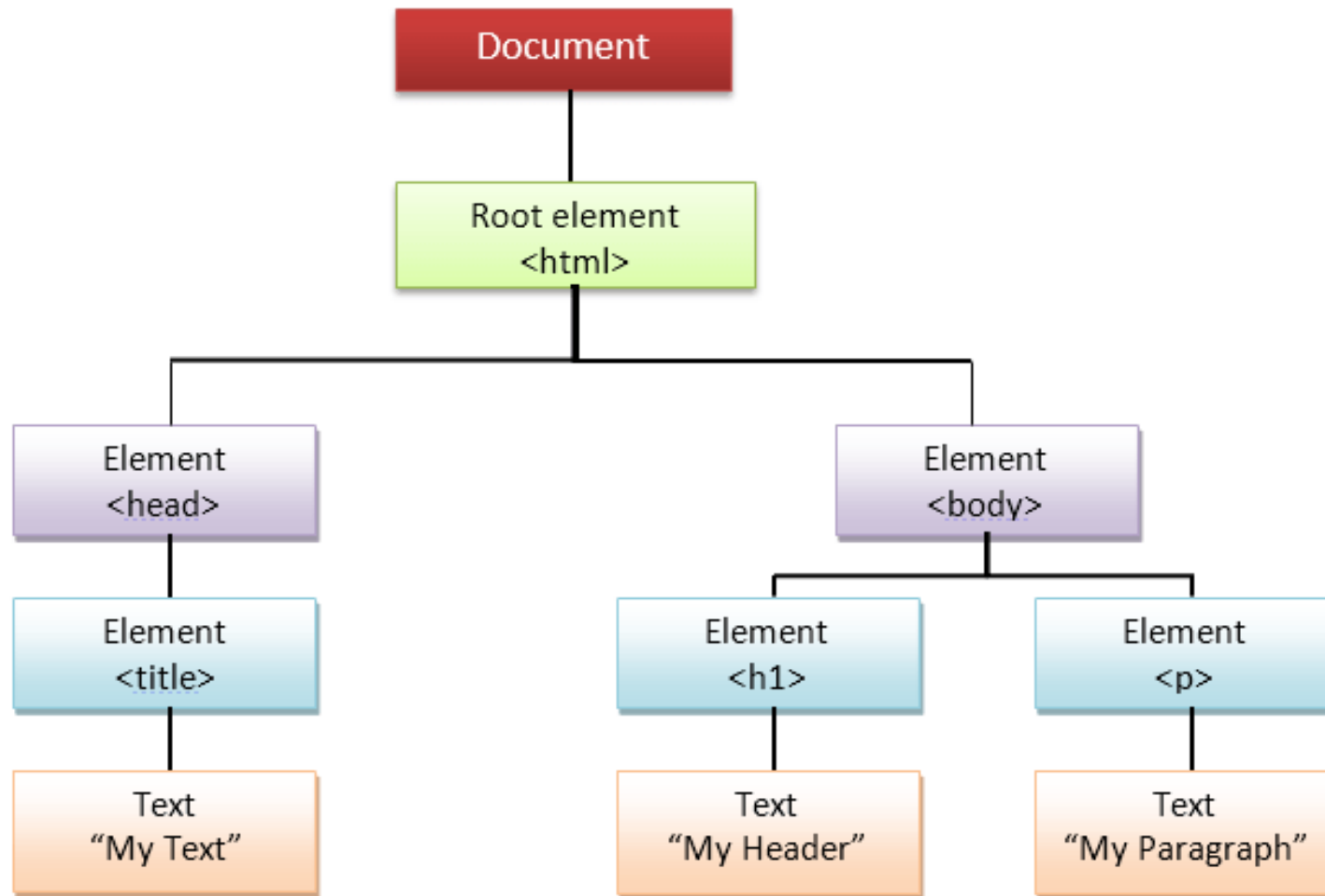
Wymagania aplikacji CALC 0.3

- Aplikacja wykorzystuje GUI w postaci HTML-a.
- Skrypt programu zostaje podpięty do pliku index.html
- **Dodatkowe style CSS zostają podpięte do pliku index.html**
 - Wykorzystany zostaje framework Bootstrap 4
- Program pozwala na korzystanie z czterech podstawowych operacji arytmetycznych.
- Aplikacja pobiera argumenty od użytkownika po czym zwraca wynik w przeglądarce.
- **Obsługa aplikacji odbywa się poprzez interfejs graficzny przypominający klasyczny kalkulator**
 - Dane wprowadzane są przy pomocy przycisków z odpowiadającymi im liczbami
 - Zdeklarowanie operacji arytmetycznych odbywa się poprzez przyciski.
 - Dane zostają wyświetlone w elemencie typu input pełniącego rolę wyświetlacza.

Instrukcje i mechanizmy z których należy skorzystać

- **Obsługa zdarzeń: reakcja na naciśnięty przycisk wywołuje konkretną instrukcję kodu.**
 - `button.onclick = function() {`
 ...
 }
• `button.addEventListener('click',`
 `function() {`
 ...
 }
);
- **Tworzenie uchwytów, identyfikowanie elementów DOM**
 - **HTML:** `<input type="button" id="send" />`
 - **JS:** `var x = document.getElementById('send');`

DOM (Document Object Model)



Lepsza organizacja kodu: tablice i funkcje

- Tablice, mogą posłużyć do przechowanie wielu nazw poszczególnych identyfikatorów (https://devdocs.io/javascript/global_objects/array):

```
const buttonIdNumber = ['one', 'two', 'three', 'four', 'five',  
  'six', 'seven', 'eight', 'nine', 'zero', 'dot'];
```

- Funkcje służą do grupowania powtarzalnych elementów kodu - **zasada DRY** (<https://devdocs.io/javascript/functions>):

```
function f(arg1, arg2) {  
  return arg1 + arg2;  
}
```

- Elementy z tablicy można odczytać używając funkcji **forEach**:

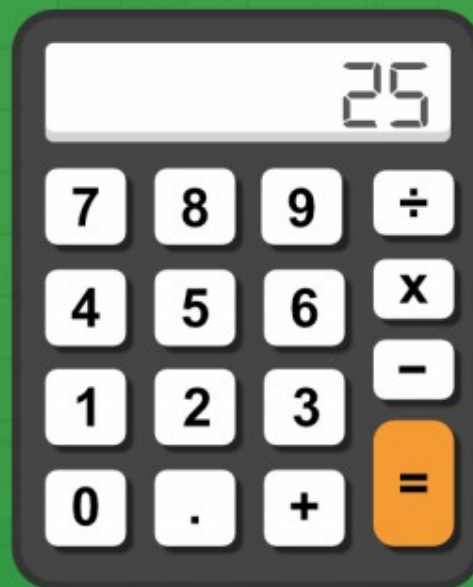
```
buttonIdNumber.forEach(element => {  
  callfunction(element);  
});
```

DOCUMENTACJA

- <https://devdocs.io/dom/document/getelementbyid>
- <https://devdocs.io/dom/location/reload>
- https://devdocs.io/dom/document_object_model/events
- <https://devdocs.io/javascript/>
- https://devdocs.io/dom_events/

Projekt wyglądu

Program to make a simple Calculator



Rozwój aplikacji

- **Gdzie dostrzegasz błędy programu ze źródła?**
- **Zachowanie elementów GUI:**
 - Czy zachowanie elementów interfejsu jest intuicyjne dla użytkownika?
 - Na błędy ze strony użytkownika powinna odpowiadać aplikacja?
 - Co stanowi największą trudność w projektowaniu takiego GUI?
- **Nowe funkcjonalności:**
 - Jakie operacje matematyczne może jeszcze obsługiwać aplikacja?
 - W co można wyposażać stronę www, a czego nie ma kalkulator?