Najważniejsze (podstawe) informacje potrzebne do pracy z JS na stronie.

Dodawanie do strony

```
<script></script>
<script src="script.js"></script>
```

Najlepiej dodawaj na końcu strony, tuż przed znacznikiem zamykającym body.

```
<body>
        <!-- struktura strony -->
        <!-- js -->
</body>
```

Zmienne

Powszechne w językach programowania.

Zapisuje dane w pamięci, dzięki czemu te dane mogą być użyte na różnych etapach programu poprzez odwałenie się do nazwy zmiennej.

```
var nazwaZmiennej = "tekst";

console.log(nazwaZmiennej)
```

Typ danych (proste) - trzy główne

```
var name = "Jan"  //string
var age = 102 //number
var female = true //boolean
```

Nazwa zmiennej

```
let userName;
var productID;
const shopCategory;
```

Nazwa zmiennej jest jej identyfiaktorem.

Nazwa zmiennej - wielkość liter ma znaczenie. Bez cyfry na początku, nazwa ma odpowiadać temu co jest przechowywane. Istnieją słowa zastrzeżone na nazwy (true, function, class)

Deklaracja zmiennej i przypisanie wartości

```
var productID = 57190;
let userName = "Marysia"
const shopCategory = "Multimedia"
```

Deklaracja zmiennej i przypisanie wartości

```
var productID = 57190;
let userName = "Marysia"
const shopCategory = "Multimedia"

productID = 76221
userName = "Maria"
~~const shopCategory = "coś innego"~~
```

Deklaracja var

var nazwaZmiennej;

"Od zawsze" w JavaScript zmienne deklarowało się za pomocą słowa kluczowego var.

Nazwa zmiennej - wielkość liter ma znaczenie. Bez cyfry na początku, nazwa ma odpowiadać temu co jest przechowywane, istnieją słowa zastrzeżone na nazwy (true, function, class)

Deklaracja let

let to nowe var.

Usuwa różne "problemy" z var, ale na tym etapiue my tych problemów nie zbaczymy jeszcze i nas nie dotyczą, więc spokojnie możemy używać var. Tym bardziej, że stare przeglądarki mogę nie rozumieć "let".

Deklaracja const

Na tym etapie możemy myśleć o niej jako stała (choć bardziej prawidłowe jest określenie zmienna "stała". Nie można potem zmienić danych czy przypisanego obiektu.

```
const idUser = 1203;
let age = 20;
let username = "Kowalska";
```

Operator przypisania

```
let colorBike = "#34bad3"
```

Przypisz do lewej to co po prawej.

Przykłady innych operatorów

typeof //wyświetlenie typu danych

+ //dodawanie lub konkatenacja (łączenie)

++ //inkrementacja

-- //dekrementacja

Komentarz

```
// jednoliniowy

/*
wieloliniowy
wieloliniowy
*/
```

Wyrażenie

Zwraca coś

2 + 5 // zwróci 7
5 > 10 // zwróci false

Instrukcja powinna coś wnosić

2 + 5; //instrukcja nic nie wnosi

var add = 2 + 5; //instrukcja tworzy zmienne i przypisuje do niej wynik wyrażenie 2 + 5

Instrukcja - działanie, które coś robi, działanie które ma znaczenie dla programu.

Instrukcje dobrze zakończyć średnikiem, ale nie jest to konieczne w JS.

Instrukcje - kolejność

instrukcje są wykonywane od góry do dołu.

```
var a = 5;
var b = 6;
var c = a + b;
//c wynosi 11
```

notacja wielbłądzia

nazwaZmiennej

dodawanieElementow

onlyEnglish

DOCUMENT OBJECT MODEL

JavaScript

Bez przeglądarek nie byłoby JavaScript.

Bez JS internet nie byłby taki sam.

JavaScript - praca z DOM - to robimy tworząc slider, popup, menu hamburger itp.

Czym jest DOM?

Document Object Model - reprezentacja dokumentu (strony) html w przeglądarce.

DOM to zbiór obiektów (węzłów), tworzonych przede wszystkim przez znaczniki, atrybuty i zawartość tekstową. DOM ustawia węzły w strukturze typu drzewo co tworzy między nimi relacje i zagnieżdżenia.

Po co DOM?

Możliwy dostęp do elementów strony.

Możemy dzięki temu dodawać i edytować elementy i ich atrybuty (klasy, style) czy zawartość tekstową.

Manipulacja DOM w JavaScript

Wszystko co robimy ze stroną, robimy z DOM a nie z kodem HTML. Jeśli dokonasz jakichś zmian (dodasz element, zmienisz klasę), to te zmiany nie są widoczne w źródle strony (w html) tylko w obiekcie DOM.

Pobranie elementów

```
var redItem = document.querySelector('div.red');

var blueItems = document.querySelectorAll('.blue');

var greenElement = document.getElementById('green');

//Pamiętaj, że w metodach querySelector piszesz elektro jak w CSS a w getElementById piszesz nazwę identyfikatora (bez #) którą posiada dany element np. <div id="green"></div>
```

Ustawienie nasłuchiwania

```
const btn = document.querySelector('button');
btn.addEventListener('click', function () { });
```

Metoda addEventListener, którą możemy wykonać na różnych obiektach DOM.

Wskazanie zdarzenia

```
const btn = document.querySelector('buton');
btn.addEventListener('click', function () { });
```

Jako pierwszy argument metody addEventListener wskazujemy na co ma nasłuchiwać (jakiegow wydarzenia na danym obiekcie oczekujemy). Mamy listę takich zdarzeń np. "click", "scroll", "mousemove", "resize" i mnóstwo innych.

https://developer.mozilla.org/pl/docs/Web/Events

Określenie akcji

```
const btn = document.querySelector('buton');
btn.addEventListener('click', function () {
        //program
        console.log("kliknięte!");
});
```

Jako drugi argument metody addEventListener wskazujemy co ma się wydarzyć jeśli nastąpi zdarzenie. Określamy to za pomocą funkcji w której ciele umieszczamy instrukcje. Po każdym wykryciu zdarzenia nastąpi wywołanie funkcji.

Zadeklarowanie funkcji poza metodą i wskazanie jej w metodzie

```
function nazwaFunkcji(){
//ciało funkcji z instrukcjami. Program, który wykonuje się po każdym kliknięciu
}

window.addEventListener('scroll', nazwaFunkcji);
```

Nie wywołuj tej funkcji w metodzie!

```
function nazwaFunkcji(){
//ciało funkcji z instrukcjami. Program, który wykonuje się po każdym kliknięciu
}
nazwaFunkcji()

window.addEventListener('scroll', nazwaFunkcji());
```

TO PRZEJDŹMY DO PROJEKTÓW