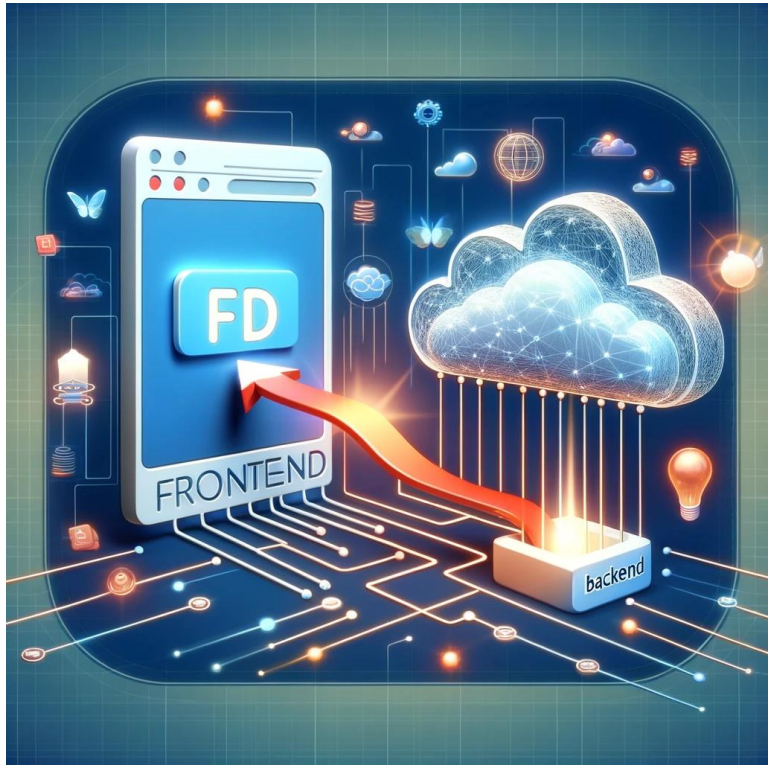
FRONTEND

VANILLA JS, / HTML5 / BOOTSTRAP

Aleksy Dąda, Szymon Domagała, Piotr Stasiak

CZYM JEST FRONTEND?

- Wszystko to co użytkownik widzi i z czym wchodzi w interakcję



PODSTAWA FRONTENDU

- HTML
- CSS
- JS

+ Biblioteki i frameworki

OPTYMALIZACJA WYDAJNOŚCI

- **Lazy Loading**
- **Optymalizacja obrazów**
- **Minifikacja zasobów**

HTML

```
1    <h1> TYTUŁ </h1>
2    <h2> podtytuł </h2>
3    <a href=""> link </a>
4    <p> paragraf </p>
5    <button> przycisk </button>
```

TYTUŁ

podtytuł

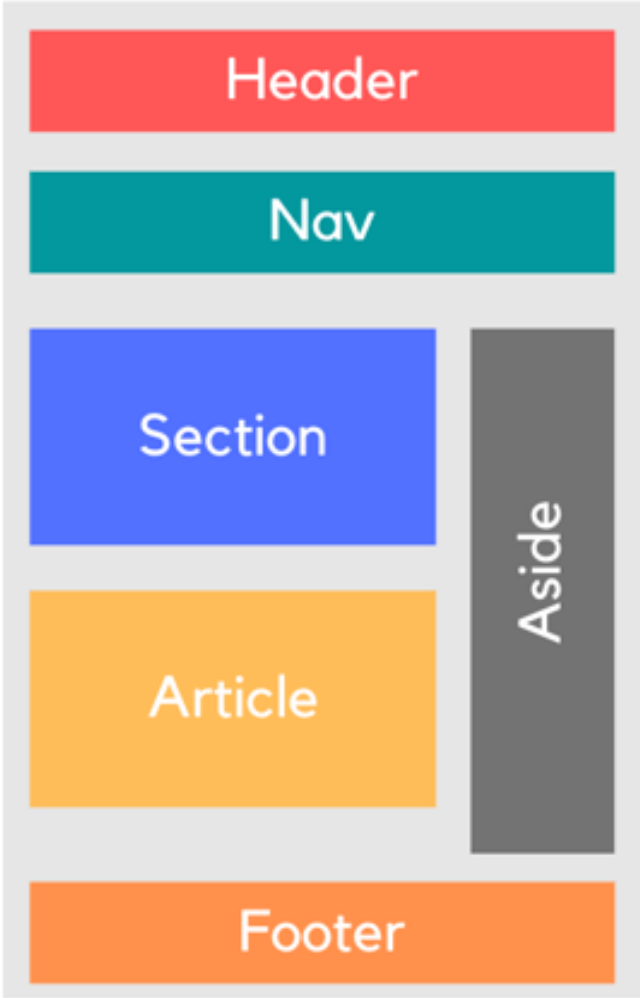
link

paragraf

przycisk

HTML5

- Semantyczne znaczniki
- Wsparcie dla multimediów
- Formularze i walidacja



HTML5 – FORMULARZE I WALIDACJA

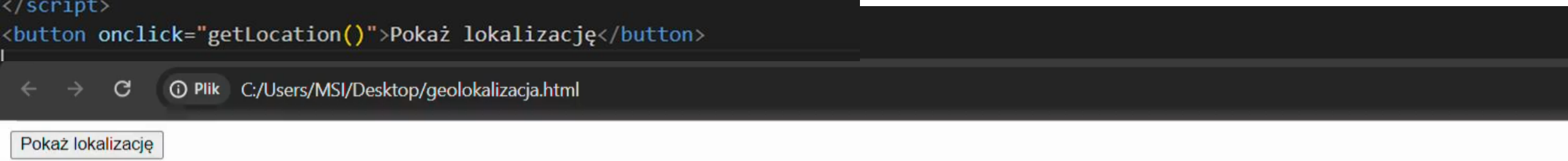
```html
<form>
    <label for="email">Email:</label>
    <input type="email" id="email" placeholder="Wpisz swój email" required>

    <label for="birthdate">Data urodzenia:</label>
    <input type="date" id="birthdate" required>

    <input type="submit" value="Wyślij">
</form>
```

HTML5

- Semantyczne znaczniki
- Wsparcie dla multimediów
- Formularze i walidacja
- Grafika i efekty wizualne
- Offline i magazynowanie
- Nowe API - np. Geolocation API

HTML5 – GEOLOCATION API

```
<script>
    function getLocation() {
            navigator.geolocation.getCurrentPosition(showPosition);
    }
    function showPosition(position) {
        alert("Szerokość geograficzna: " + position.coords.latitude +
        "\nDługość geograficzna: " + position.coords.longitude);
    }
</script>
<button onclick="getLocation()">Pokaż lokalizację</button>
```



KRÓTKA HISTORIA VANILLA JS?

Vanilla JS jest to nazwa na tzw. "czysty" JavaScript. W pierwszej połowie lat 90-tych stał się podstawowym językiem skryptowym. Umożliwił on tworzenie interaktywnych stron internetowych. Aktualnie jego popularność ponownie wzrasta z uwagi na nowe standardy (bardziej wydajny i oszczędny kod). Do dziś Vanilla JS pozostaje podstawową umiejętnością front-end developerów.

Zastosowanie JS w technikach webowych

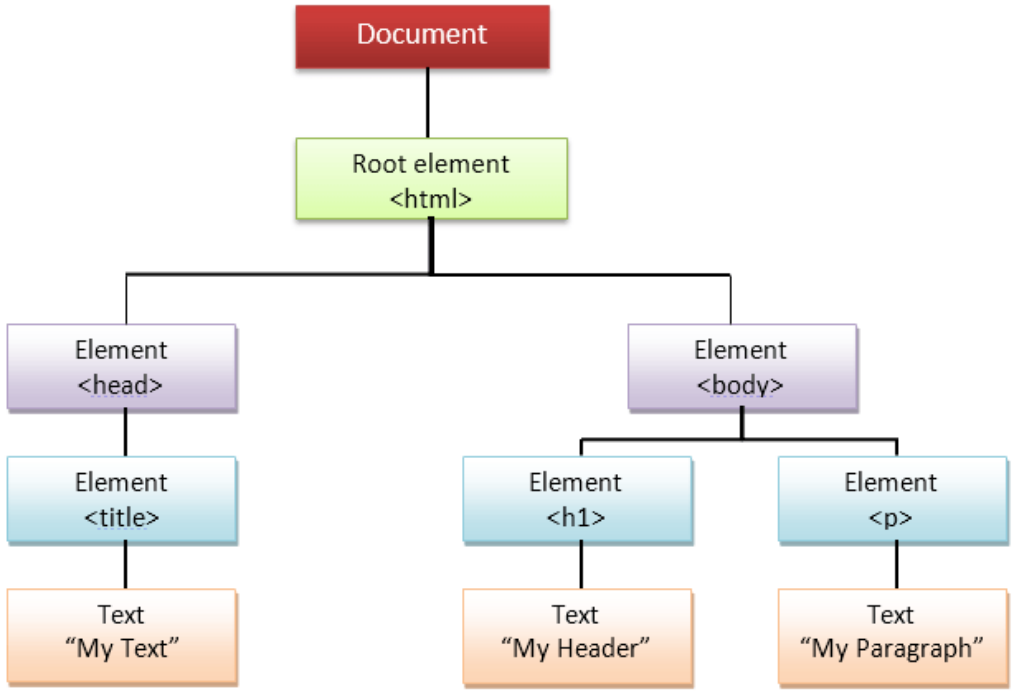
JavaScript jest bardzo szeroko stosowany w technikach webowych głównie do stworzenia interaktywnych i dynamicznych elementów.

Zastosowania:

- Tworzenie aplikacji webowych
- Animacje i efekty wizualne
- Manipulacja DOM
- Walidacja formularzy
- Tworzenie gier w przeglądarce (frameworkPhaser)
- Tworzenie wieloplatformowych gier i aplikacji dzięki Node.js i silnika przeglądarki Chromium
- Dzięki Node.js można używać JS w backend

Manipulacja DOM – Document-Object-Model

Jest to reprezentacja naszej strony internetowej w postaci obiektowego drzewa logicznego.




```
document.getElementsByClassName('nazwaKlasy');
//Zwraca tablicę obiektów które maja przypisaną daną klasę

document.activeElement ;
//Aktywnie wybrany element

document.getElementsByName("nazwa");
//Zwraca element który posiada atrybut name

document.getElementsByTagName("nazwa tagu HTML")
//Zwraca listę elementów

document.querySelector("dowolny selektor");
//Zwraca pierwszy element który spełnia dany selektor

document.querySelectorAll("dowolny selektor");
//Zwraca wszystkie elementy, które spełniają dany selektor
```

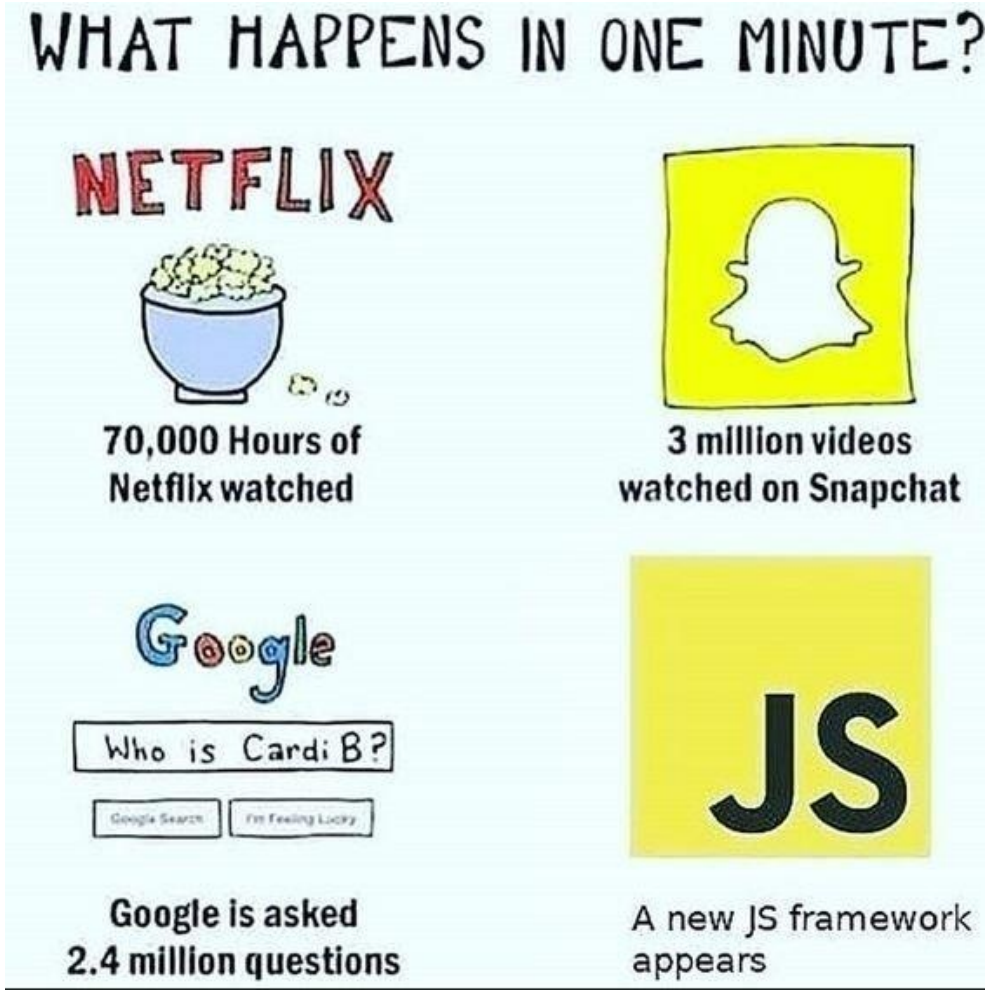
Główne wady i zalety JS

ZALETY

- Prostota (bardzo prosta i intuicyjna składnia)
- Uniwersalność (pozwala na wykonanie niemalże każdego działania)
- Popularność (mnóstwo ogólnodostępnych poradników)
- Asynchroniczność (wykonywanie operacji bez czekania na zakończenie poprzednich)

WADY

- Niejednoznaczna interpretacja kodu przez przeglądarki
- Nieustanny rozwój
- Bezpieczeństwo (może być podatny na ataki takie jak wstrzykiwanie kodu czy ataki XSS- Cross Site Scripting)



RÓŻNICE W SKŁADNI POMIĘDZY INNYMI JĘZYKAMI PROGRAMOWANIA (JAVA / PYTHON / C++)

```javascript
let a = 5;
a = "Witaj";
console.log(a);
```

```python
a = 5
a = "Siemano"
print(a)
```

```java
public class Main {
    public static void main(String[] args) {
        int a = 5;
        String str = "Hello";
        System.out.println(str);
    }
```

```cpp
int main() {
    int a = 5;
    std::string s = "Hello";
    std::cout << s << std::endl;
    return 0;
```

Javascript oraz Python są jedynymi językami dynamicznie typowanymi w tym zestawieniu.

RÓŻNICE W SKŁADNI POMIĘDZY INNYMI JĘZYKAMI PROGRAMOWANIA (JAVA / PYTHON / C++)

```javascript
// Funkcja anonimowa
let anonim = function (name = "foo") {
  console.log("Anonimowa");
};

// Funkcja strzałkowa
let arrow = (x, y) => x * y;
```

```
interface calcSquare }
    new *
    int calculate(int x);
}
        ...
calcSquare square = (int x) -> x * x;
int result = square.calculate(5);
```

RÓŻNICE W SKŁADNI POMIĘDZY INNYMI JĘZYKAMI PROGRAMOWANIA (JAVA / PYTHON / C++)

```javascript
// Funkcja anonimowa
let anonim = function (name = "foo") {
  console.log("Anonimowa");
};

// Funkcja strzałkowa
let arrow = (x, y) => x * y;
```

```cpp
auto square = [](int x) { return x * x; };
   int result = square(5);
```

```python
AddOne = lambda x: x + 1;
print(AddOne(5)) # 6
```

RÓŻNICE W SKŁADNI POMIĘDZY INNYMI JĘZYKAMI PROGRAMOWANIA (JAVA / PYTHON / C++)

```javascript
// Operator REST
function test(...args) {
  console.log(args);
}

test(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
// [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```cpp
template <typename ... args>
void print(args ... a) {
    (std::cout << ... << a);
}
print(1, 2, 3, "Hello", 4.5); // 123Hello4.5
```

```python
def func(*args):
    print(args)
func(1,2,3,4,5,6,7,8,9)
#(1, 2, 3, 4, 5, 6, 7, 8, 9)
```

W C++ nie mamy stricte odpowiedników operatora "..." znanych z .js - natomiast można osiągnąć podobny efekt przy użyciu operatora FOLD (C++ 17)

RÓŻNICE W SKŁADNI POMIĘDZY INNYMI JĘZYKAMI PROGRAMOWANIA (JAVA / PYTHON / C++)

```javascript
// Operator REST
function test(...args) {
  console.log(args);
}

test(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
// [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```java
public static void main(String[] args) {
    printArgs("a", "b", "c");
}
1 usage  new *
public static void printArgs(String ...args) {
    for(String arg : args) {
        System.out.println(arg);
    };
}
```

Podobne działanie można uzyskać także w Javie

RÓŻNICE W SKŁADNI POMIĘDZY INNYMI JĘZYKAMI PROGRAMOWANIA (JAVA / PYTHON / C++)

```javascript
// Klasy
class Car {
  constructor(brand) {
    this.carname = brand;
  }
  present() {
    return "I have a " + this.carname;
  }
}
const user1 = new Car("Ford");

Car.prototype.hello = function () {
  return "Hello";
};
```

```java
public class Example {
    2 usages
    private int num;
    no usages  new *
    public Example(int x) {this.num = x;}
    no usages  new *
    public int getX() {return num;}
}
```

Brak prototypów w Java.

RÓŻNICE W SKŁADNI POMIĘDZY INNYMI JĘZYKAMI PROGRAMOWANIA (JAVA / PYTHON / C++)

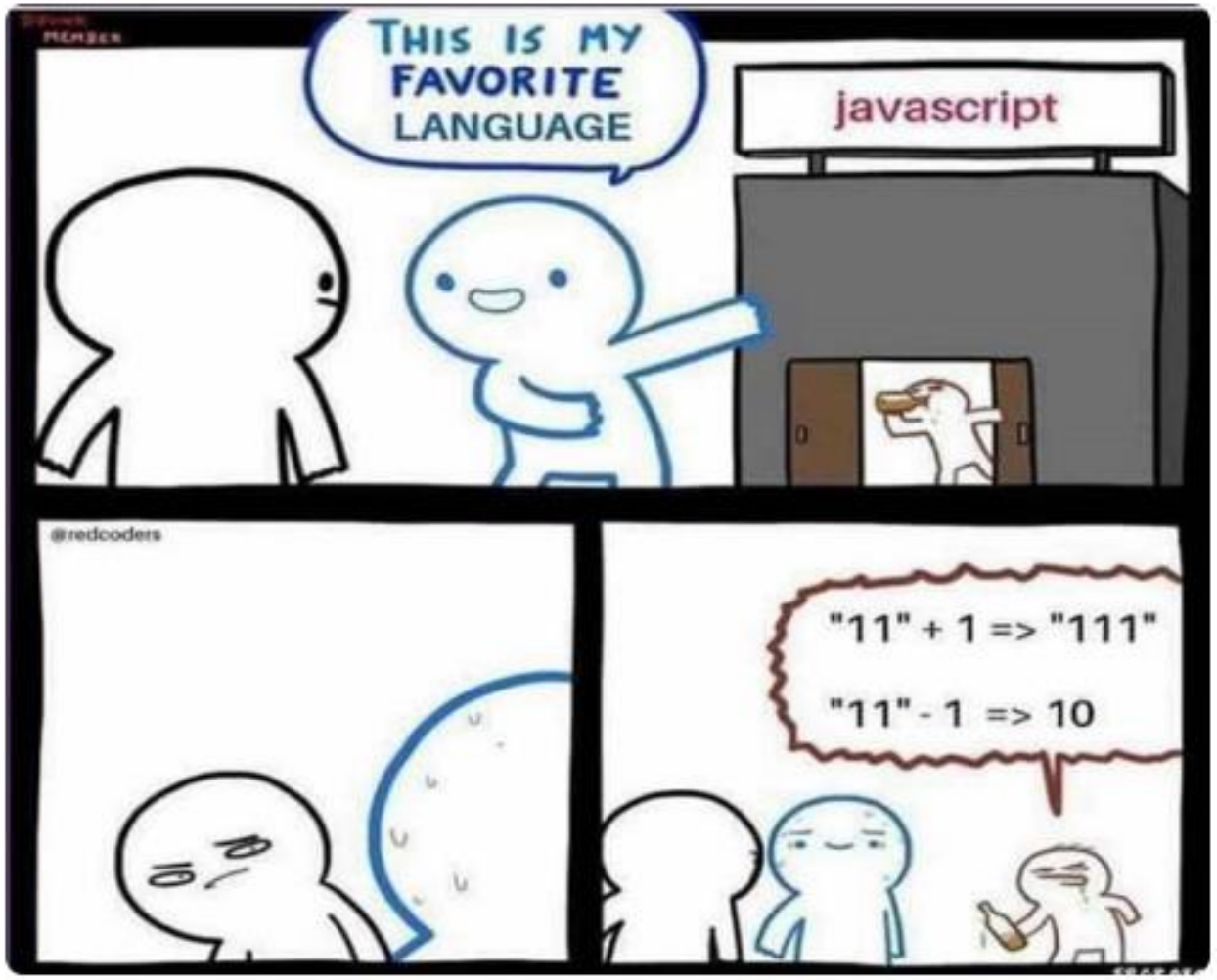
```javascript
// Klasy
class Car {
  constructor(brand) {
    this.carname = brand;
  }
  present() {
    return "I have a " + this.carname;
  }
}
const user1 = new Car("Ford");

Car.prototype.hello = function () {
  return "Hello";
};
```

```cpp
class Example {
public:
    Example(int x) : num(x) {}
    int getX() const { return x; }
private:
    int num;
};
```

```python
class MyClass:
    def __init__(self):
        self.data = 5
    def printData(self):
        print(self.data)

obj = MyClass()
obj.printData() # 5
```

W przypadku gdy obiekty mają wiele wspólnych metod, przechowywanie ich w prototypie zamiast w każdej instancji pozwala znacznie zaoszczędzić pamięć. Każda instancja ma dostęp do prototypu i jego metod, ale nie przechowuje ich kopii.

DZIWNE ZACHOWANIA JĘZYKA



DZIWNE ELEMENTY SKŁADNI

```
typeof NaN; // number

NaN === NaN; // false
NaN == NaN; // false
```

NaN jest wartością która nie jest równa żadnej innej wartości, nawet samej sobie.

DZIWNE ELEMENTY SKŁADNI

```
[10,9,8,3,2,1,0].sort();  // [0, 1, 10, 2, 3, 8, 9]
```

JavaScript porównuje liczby alfabetycznie tzn. jako string. Alfabetycznie „10” jest mniejsze od „2” ponieważ pierwszy znak w „10” czyli „1” jest porównywany z pierwszym znakiem w „2”.

DZIWNE ELEMENTY SKŁADNI

```
typeof null; // object
typeof undefined; // undefined

null === undefined; // false
null == undefined; // true
```

Null oraz Undefined są traktowane jako równoważne przez operator „luźnego” porównania. Analogiczne porównanie to : 0 == false - zwróci „true”.

DZIWNE ELEMENTY SKŁADNI

```
Math.max(); // ?
Math.min(); // ?
Math.max() > Math.min(); // false
```

Math.max() oraz Math.min() służą do porównywania przekazanych wartości. Jednakże jeśli z jakiegoś powodu nie podamy żadnych wartości, to zwracane wartości będą odpowiednio równe: -Inf oraz Inf.

DZIWNE ELEMENTY SKŁADNI

```
[] % 2 == 0; // true

typeof []; // object
```

Próba konwersji typu object na prymitywną wartość. W tym przypadku metodą toNumber(). A w niej:

1. Pusta tablica jest rzutowana na string. A jak jest pusta to otrzymamy - ' '
2. Pusty string rzucony na liczbę to : 0

DZIWNE ELEMENTY SKŁADNI

```
[] + [] == '';  // true

[] + {} == '[object Object]'; // true -> zgodnie z przewidywaniem

{} + [] == 0; // true, ale czego ?

({} + []) == '[object Object]'; // true
```

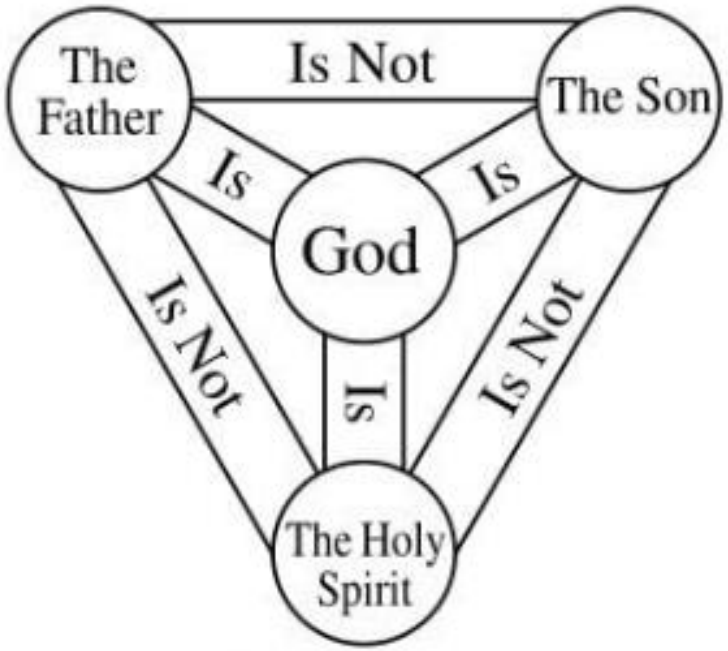
Przedostatni : Pusty obiekt ({ }) jest traktowany jako blok kodu. A ponieważ jest pusty to nie mamy wykonywanego dodawania tylko wymuszamy rzutowanie na liczbę : „+[ ]”
Ostatni : Otaczając nawiasami, wymuszamy faktyczne dodawanie (łączenie dwóch stringów) *

DZIWNE ELEMENTY SKŁADNI

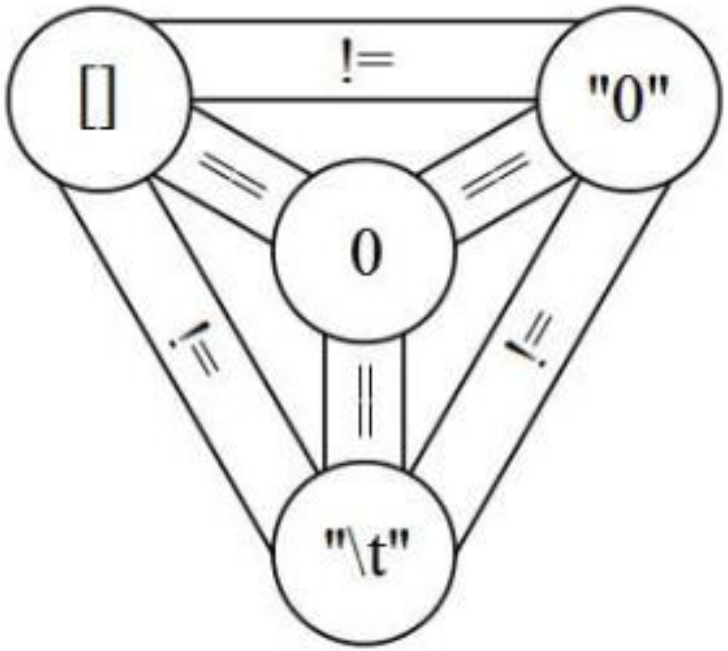
```
[] == 0; // true
"0" == 0; // true
"" == 0; // true
"\t" == 0; // true -> konwersja tabulatora na liczbę
[] == "0"; // false
[] == "\t"; // false
"0" != "\t"; // true -> brak konwersji, zwykłe porównanie ciągów znaków
```

Pusta tablice po konwersji do stringa (' ') jest porównywana z wartością tego samego typu – a te są różne.

DZIWNE ELEMENTY SKŁADNI



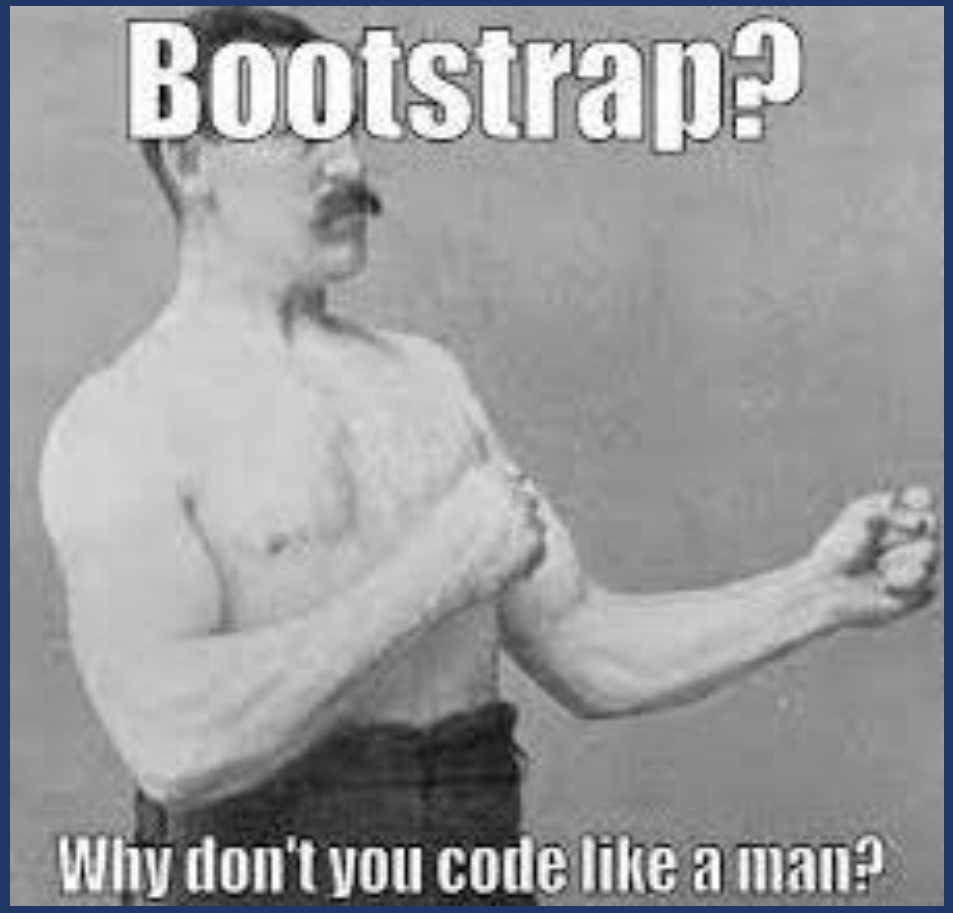
Christianity



JavaScript

@hsjoihs

BOOTSTRAP



BOOTSTRAP

- Jest to narzędzie ułatwiające tworzenie responsywnych stron internetowych (w teorii ☺) .
- Zabawa polega na wykorzystywaniu gotowych komponentów, lub klas dostępnych i dokładnie opisanych w dokumentacji Bootstrapa.
- Nastawiony na workflow typu „mobile-first”.
- Używając predefiniowanych przez Bootstrapa klas mamy pewność że strona będzie odpowiednio skalowalna zarówno na ekranach mobilnych jak i tych większych – desktopowych.

BOOTSTRAP

ZALETY ?

WADY ?

Bibliografia

HTTPS://HOSTCLUB.PL/JAVASCRIPT/

HTTPS://BORINGOWL.IO/BLOG/VANILLA-JS-CZYM-JEST-I-JAK-TO-DZIALA

HTTPS://GEEK.JUSTJOIN.IT/POWINNISMY-PRZESTAC-UZYWAC-JAVASCRIPTU-DEVDEBATA/#2_ZALETY_JS_–_CO_UWAZASZ_ZA_NAJWIEKSZA_ZALETE_JAVASCRIPTU_I_DLACZEGO

HTTPS://JAKI-JEZYK-PROGRAMOWANIA.PL/TECHNOLOGIE/JAVASCRIPT/

HTTPS://SDACADEMY.PL/FRONTEND-CO-TO-JEST/

HTTPS://IMAKEABLE.COM/NASZ-BLOG/OPTYMALIZACJA-APLIKACJI-WEBOWYCH-NAJLEPSZE-PRAKTYKI-DLA-LEPSZEJ-WYDAJNOSCI

HTTPS://BLOG.HUBSPOT.COM/BLOG/TABID/6307/BID/5847/A-MARKETER-S

HTTPS://INFORMATYK.EDU.PL/JAVASCRIPT-MANIPULACJA-DOM/

DZIĘKUJEMY! ZA UWAGĘ