Zmienne w JS

Nazewnictwo zmiennych

Nazwy zmiennych i stałych które deklarujemy nie mogą być byle jakie. Istnieją pewne zasady których musimy się trzymać. I tak:

- wielkość liter ma znaczenie. Zmienna myTXT to nie to samo co mytxt
- nazwa zmiennej nie może zaczynać się od cyfry,
- nazwa zmiennej nie może zawierać spacji, kropki, przecinka ani myślnika (można natomiast używać podkreślenia),
- nazwą zmiennej nie może być słowo kluczowe zarezerwowane przez JavaScript

Po prostu nie zaczynaj nazw od cyfr i pisz nazwy zmiennych po angielsku. Bardzo częstą praktyką jest stosowanie zapisu camelCase, czyli np. veryImportantThing, ale wiele osób stosuje też zapis z podłogą czyli very_important_thing.

Jest jednak ważniejsza sprawa, którą zapamiętaj.
Nazywaj swoje zmienne tak, by dało się zrozumieć do czego się odnoszą. Zmienna o nazwie **elementsCount** jest bardziej czytelna, niż **xxx**. W tym kursie czasami będę korzystał z mało czytelnych zmiennych (np. a), ale wynika to tylko z tego, że kod często jest bardzo, bardzo krótki, a i jestem leniem.

Let/var i const

Wraz z wprowadzeniem nowych słów let/const dostaliśmy rozróżnienie na zmienne i stałe. Zmienne deklarowane za pomocą **let/var** można w przyszłości zmienić - czyli podstawić im nową wartość. Zmiennym stworzonym za pomocą **const** nie jesteśmy w stanie podstawić nowej wartości.

```
var text = "ala";
text = "ala ma kota"; // wszystko ok, bo var to zmienna

let a = 0;
a = 10; //wszystko ok, bo let

const b = 0;
b = 10; //błąd - do stałej nie możemy przypisać nowej wartości

const name = "Ala";
name = "Monika"; //błąd
```

Różnice między var a let/const

Deklaracje let/const nie tylko wprowadziły stałe, ale także kilka różnic w stosunku do var.

Pierwsza i najważniejsza różnica między let/const a var to zasięg zmiennych.

W przypadku let/const zmienne mają zasięg blokowy, co w skrócie oznacza "od klamry do klamry":

```
let a = 20; //zmienna globalna

{
    let a = 30; //zmienna lokalna
    console.log(a); //30
}

console.log(a);    //20
```

```
{
    let a = "Ala";
    console.log(a); //Ala
}
{
    console.log(a); //error - nie ma takiej zmiennej
}
{
    let a = "Ola"; //zmienna lokalna w tym bloku
    console.log(a); //Ola
}
console.log(a); //error: a is not defined
```

Zmienne deklarowane za pomocą **var** mają natomiast zasięg funkcyjny, czyli ich zasięg określa ciało funkcji.

```
var a = 20; //zmienna globalna

function test() {
    var a = 30; //zmienna lokalna
    console.log(a); //30
}
test();

console.log(a);    //20
```

W nielicznych sytuacjach może to powodować niezamierzone działanie kodu.

```
if (true) {
    var myVar = 20;
}

console.log(myVar); //20
```

```
for (var i=0; i<10; i++) {
    console.log(i);
}

console.log(i); //10
```

Kolejna cecha rozróżniająca var od let/const jest taka, że zmienne var możemy ponownie deklarować, co jest niemożliwe w przypadku let i const:

```
var name = "Marcin";
var name = "Karol";
console.log(name); //Karol
```

```
let name = "Marcin";
let name = "Karol"; //błąd = Identifier "name" has already been declared
console.log(name);
```

Podsumowanie

Ok podsumujmy powyższe rozważania.

- Zmienne to rodzaje pudełek, w których możemy trzymać różne rzeczy.
- Zmienne możemy tworzyć za pomocą słów kluczowych var/let/const, przy czym zalecane są te dwa ostatnie
- Let/const różnią się od varów głównie zasięgiem oraz tym, że w jednym zasięgu (bloku) nie możemy ponownie tworzyć zmiennych o tej samej nazwie.
- Hoisting to zjawisko wynoszenia na początek skryptu zmiennych i deklaracji funkcji
- W naszych skryptach starajmy się używać jak najwięcej const - dzięki temu będziesz wyglądał jak pro. Jedynym wyjątkiem są liczniki oraz zmienne które wiemy, że zaraz zmienimy (np. toggleCatNightPartyMode)

I w zasadzie tyle. Cała reszta przyjdzie z praktyką. Nawet jeżeli na chwilę obecną powyższe rozważania wydają się dla ciebie dziwne, nie przejmuj się. Po 2-3 skryptach całość stanie się odruchem.