

Moje Kursy

Kursy > WRO_FER_S_24 > Nauka > Sass i RWD > Egzamin próbny - rozwiązanie > Egzamin próbny - rozwiązanie > Zadanie 2

Artykuł

Zadanie 2

Career Lab Front End

sygnatura: WAR_CFE_D_06

JavaScript Developer: React

sygnatura: WRO_FER_S_24

0. Prework

1. Zaawansowany HTML i CSS

2. Sass i RWD

Praca samodzielna - Podstawy NodeJS i npm

Dzień 1 - Konfiguracja środowiska i Sass

Dzień 1 - Praca domowa

Dzień 2 - Sass i RWD

Dzień 2 - Praca domowa

Dzień 3 - RWD

Dzień 3 - Praca domowa

Dzień 4 - RWD i Animacje

Dzień 4 - Praca domowa

Dzień 5 - Warsztat

Egzamin próbny

Egzamin próbny - rozwiązanie

- Egzamin próbny - rozwiązanie

3. Praca własna 1

4. JavaScript

5. ES6 i React Podstawy

6. Praca własna 2

7. React

Scrum Lab Front End

sygnatura: WRO_SFE_S_33

Idź w górę

Poprzedni

Następny

Pierwsza część zadania drugiego mówiła o stworzeniu mapy z konkretnymi kolorami. Kolory podane były w formacie HEX obok nazw pod którymi miały znaleźć się w mapie:

```
1 | $colors: (  
2 |   gold: #f9c00c,  
3 |   blue: #00b9f1,  
4 |   purple: #7200da,  
5 |   red: #f9320c,  
6 |   green: #75D701  
7 | );
```

Kopiuuj

W kolejnym kroku, mieliśmy dzięki interpolacji i pętli ustawić kolory tła elementom `el-1`, `el-2` itd. w klasie `task-02` zgodnie z wartościami w naszej mapie.

Potrzebowaliśmy tutaj licznika (aby wytworzyć odpowiednie selektory) a także konkretnego koloru który się pod nim kryje. Jednak mamy do czynienia tutaj z mapą, której nie możemy iterować przy użyciu pętli `@for` ze względu na to, że nie jest ona indeksowana od liczb całkowitych.

Aby skorzystać z pętli `@for`, potrzebujemy użyć funkcji `map.values()`, do której prześlemy naszą mapę a w zwrocie otrzymamy listę kolorów. **Należy również użyć reguły `@use` aby zaimportować odpowiedni moduł!**

```
1 | // Na początku pliku  
2 | @use "sass:map";  
3 |  
4 | // ... mapa $colors  
5 |  
6 | $colors-list: map.values($colors);
```

Kopiuuj

Mając już listę kolorów, stwórzmy pętlę która po niej przeiteruje, stworzy odpowiednie klasy i ustawi kolory. Aby wyciągnąć wartość z listy, użyjemy metody `list.nth` z modułu `sass:list`.

```
1 | // Na początku pliku  
2 | @use "sass:list"  
3 |  
4 | // ... mapa $colors  
5 | // ... lista $colors-list  
6 |  
7 | @for $i from 1 through list.length($colors-list) {  
8 |   .el-#{ $i } {  
9 |     background-color: list.nth($colors-list, $i);  
10 |   }  
11 | }
```

Kopiuuj

Dodatkowo elementy miały mieć `200px` wysokości i być ustawione obok siebie przy użyciu `flex`. Dlatego dla klasy `task-04` dodamy właściwość `display: flex`, a dla każdego elementu:

- `height: 200px`
- `flex-grow: 1` - elementy nie mają treści i chcemy żeby zajmowały równą szerokość

Pełne rozwiązanie

```
1 | @use "sass:map";  
2 | @use "sass:list";  
3 |  
4 | $colors: (  
5 |   gold: #f9c00c,  
6 |   blue: #00b9f1,  
7 |   purple: #7200da,  
8 |   red: #f9320c,  
9 |   green: #75D701  
10 | );  
11 |  
12 | .task-02 {  
13 |   display: flex;  
14 |  
15 |   $colors-list: map.values($colors);  
16 |   @for $i from 1 through list.length($colors-list) {  
17 |     .el-#{ $i } {  
18 |       background-color: list.nth($colors-list, $i);  
19 |       height: 200px;  
20 |       flex-grow: 1;  
21 |     }  
22 |   }  
23 | }
```

Kopiuuj

Jak oceniasz artykuł ?

