bootcamp online

JAVASCRIPT

Witaj w piątym tygodniu Bootcampu

Przed Tobą przedostatni tydzień nauki w Bootcampie JavaScript. Mam nadzieję, że poprzedni tydzień był dla Ciebie ciekawy oraz zawierał wiele cennych informacji. Ten oraz kolejny tydzień, w całości poświęcone są najnowszej edycji języka JavaScript, która powstaje na bazie specyfikacji EcmaScript. Potocznie tę wersję określa się mianem **ES6**. Mając już solidne podstawy z zakresu JavaScriptu w wersji ES5, zobaczysz jak najnowsza edycja ułatwia wykonywanie wielu zadań, a także jakie nowe funkcjonalności dodaje do tego języka.

Piotr Palarz

Piotr Palarz

--- Zadania na tydzień 5 ---

*Zadania domowe spakuj ZIPem i umieść na kanale **#prace_domowe** na Slacku.*
W przypadku zadań, gdzie niezbędny jest dodatkowy kod HTML i CSS, nie musisz się skupiać na tym, aby strona wyglądała pięknie. Jeśli masz czas i chęci, dopracuj swój projekt, ale najważniejszy pozostaje i tak kod JavaScript.

1. Dekompozycja obiektu z danych JSON

Pamiętasz funkcję `getJSON`, którą stworzyłeś w tygodniu trzecim? Za jej pomocą pobierz dane JSON z tego adresu: http://code.eduweb.pl/bootcamp/json/. Następnie w funkcji callback, gdzie te dane będą już zamienione na obiekt JavaScript, wykorzystaj dekompozycję (*destructuring*), aby utworzyć za pomocą zapisu **ES6** nowe zmienne, które przechowywać będą dane spod kluczy: `name`, `username`, `email`, `address.geo[0]`, `address.geo[1]`, `website` i `company.name`. Powyższe dane wstaw do template stringu, dodając odpowiednie etykiety jak np. *Imię*, *Firma* czy *Adres e-mail* wraz z niezbędnym kodem HTML, np. w formie linku dla `website`. W przypadku współrzędnych geograficznych, wstaw je do takiego linku: `<a href="http://bing.com/maps/default.aspx?cp=LAT~LON">Pokaż na mapie</a>`, gdzie `LAT` i `LON` zastąpisz kolejno przez `address.geo[0]` i `address.geo[1]`, które na tym etapie powinny być już w zmiennych. Powyższą operację wykonaj oczywiście dla **wszystkich** obiektów z tablicy. Cały sformatowany ciąg wraz ze wstawkami HTML wstaw na stronę. Sam proces pobierania danych Ajaxem i dalszego ich formatowania, możesz wywołać za pomocą kliknięcia jakiegoś przycisku.

2. Funkcja tagująca do formatowania cen

Utwórz funkcję tagującą, która użyta na tzw. *template stringu* w **ES6**, sformatuje podane w nim ceny za pomocą kodu: `n.toFixed(2).replace(/(\d)(?=(\d{3})+\.)/g, '$1,');` (pożyczamy ciekawe rozwiązanie z tego postu). Zanim jednak dokonasz takiego formatowania, przelicz cenę przez kurs podanej przy wywołaniu funkcji tagującej waluty. Zakładamy, że ceny bazowe są podane w złotówkach, a nazwa i kurs waluty dostępna będzie pod `window.currencies` (mogłaby być w ten sposób dodana np. podczas renderowania strony przez system CMS, a dzięki temu mamy dostęp do tych danych w

kodzie JavaScript. My jednak wpiszemy to sobie na sztywno). Przykładowe użycie tego kodu powinno wyglądać następująco: https://pastebin.com/6A3WZF6h. Zauważ, że przed samym template stringiem nie jest podana wyłącznie nazwa funkcji `formatPrice`, ale jest ona wywoływana z argumentem "`GBP`". Jak być może się domyślasz, oznacza to, że ta funkcja ma zwrócić inną funkcję, która zostanie użyta jako *tag function*. Argument jest przekazywany po to, aby można go było zmienić np. na "`USD`" i wówczas funkcja powinna przeliczyć cenę po kursie dolara, a także dodać przyrostki USD do ceny w sformatowanym ciągu. Zwracana funkcja, która posłuży jako funkcja tagująca, powinna mieć zatem dostęp do zmiennej przechowującej **kurs** oraz **nazwę waluty** dla podanego argumentu. Zrealizujesz to za pomocą domknięcia.

3. Przepisanie konstruktorów z ES5 na klasy w ES6

Wróćmy pamięcią do zadania 4 z tygodnia drugiego. Uzupełniłeś w nim kod tak, aby dziedziczenie prototypowe działało poprawnie. Tutaj znajdziesz kod, od którego należało wtedy zacząć: https://pastebin.com/YEBncx0d. Teraz Twoim zadaniem jest przepisanie tego działającego już kodu tak, aby korzystał z klas w **ES6**. Przepisz zarówno klasę `EventEmitter` jak i klasę `Database`, wykorzystując oczywiście dziedziczenie. Samo użycie tego kodu, a więc tworzenie nowych instancji obiektów i korzystanie z ich metod, pozostaje bez zmian.

4. Iterator dla klasy losującej liczby

W lekcjach drugiego tygodnia, w sekcji *Praktyczne projekty* znajduje się lekcja pt. *Losowanie Dużego Lotka*. Utwórz z użyciem zapisu **ES6** klasę `Lotek`, która zawierać będzie metodę `getNumbers`, a ta z kolei wylosuje liczby według logiki ze wspomnianej lekcji. Metoda ta powinna być wywoływana od razu przy tworzeniu nowej instancji obiektu tej klasy. Wylosowane liczby musisz zatem zapisać wewnątrz obiektu. Dodatkowo stwórz metodę `checkNumbers`, która pozwoli podać **6 liczb** jako osobne argumenty i zwróci obiekt, który zawierał będzie właściwości `numbers`, gdzie będzie tablica z trafionymi liczbami oraz `count`, gdzie będzie liczba trafionych numerów.

To jednak dopiero początek, gdyż najważniejszym celem tego zadania jest dodanie iteratora do tej klasy. Po poprawnym jego dodaniu, kiedy na obiekcie utworzonym z tej klasy użyjemy operatora `…spread` lub pętli `for…of`, powinien on zwrócić kolejne

wylosowane liczby. Przykładowe użycie tego kodu powinno wyglądać następująco: https://pastebin.com/0zygc3hk.

5. Preloader obrazów z użyciem Promise

W lekcjach czwartego tygodnia, w sekcji *Ajax w Praktyce oraz model Pub/Sub* znajduje się lekcja pt. *Praktyczny przykład: Preloader obrazów*. Utwórz podobne rozwiązanie, ale nie korzystając z jQuery, a z natywnych metod **DOM API**. Zamiast obiektu `Deferred` z jQuery, wykorzystaj `Promise` dostępną w **ES6**. Załadowane w tle obrazy zamień następnie na elementy `<img>` i wstaw na stronę. Przykładowe użycie tego kodu powinno wyglądać następująco: https://pastebin.com/jM7N6NTn

EcmaScript 2015

Język JavaScript opiera się na specyfikacji **EcmaScript**. Choć jest ona ciągle rozwijana, to przez wiele lat nie obserwowaliśmy przełomowych zmian w tym języku. Aż do **czerwca 2015** roku kiedy pojawiła się nowa wersja EcmaScript oznaczona numerem 2015. Poprzednia edycja nazywana była **EcmaScript 5**, a więc nowa szybko zyskała miano **ES6** i pod tymi dwoma nazwami (**ES6 i EcmaScript 2015**) kryje się dziś dokładnie to samo.

I choć komitet **TC-39**, który pracuje nieustannie nad nowym standardem języka, zapowiedział, że nowa wersja będzie wydawana co roku, to nie należy się spodziewać aż tak wielkich, jednoczesnych zmian, jak miało to miejsce w przypadku ES6. Co zatem nowego wnosi ta edycja?

Przede wszystkim kilka rozwiązań, które określane są w żargonie programistycznym jako tzw. "*syntactic sugar*". Oznacza to, że dodany został pewien nowy zapis (*syntax*), który pozwala zrealizować to samo zadanie, ale w inny sposób. I tak do JavaScriptu zawitały **klasy**, choć wciąż opierają się one na dziedziczeniu prototypowym i brakuje im prywatności. I choć sam ich zapis łudząco przypomina rozwiązania znane z innych języków programowania, to wciąż nie można określić JavaScriptu językiem, który implementuje klasyczny model programowania zorientowanego obiektowo.

Kolejną nowością są tzw. "*arrow functions*", znane m. in. z języka CoffeScript. Pozwalają one na tworzenie funkcji anonimowych bez użycia słowa kluczowego `function`, a także na zwracanie wartości bez użycia słowa `return`. Rozwiązują one również pewne problemy z kontekstem wykonania, które nierzadko musiały być obchodzone w standardowych funkcjach.

Istotnych zmian doczekały się również łańcuchy znaków, które notorycznie łączone były za pomocą operatora `+` by "skleić" statyczny tekst z dynamicznymi danymi. Od teraz możliwa jest tzw. interpolacja, a więc umieszczanie wyrażeń JavaScriptowych bezpośrednio wśród innych znaków stringu. Nowy zapis pozwala także na tworzenie tekstu rozciągającego się na wiele linijek, co nie było możliwe wcześniej bez konkatenacji.

Warto wspomnieć również o tym, że słowo kluczowe `var`, służące do tworzenia zmiennych, doczekało się nowych przyjaciół. Pojawiły się dwa nowe słowa - `let` oraz `const`. To

pierwsze pozwala tworzyć zmienne “widoczne” wyłącznie w danym bloku kodu. Oznacza to, że zmienna może być utworzona w bloku instrukcji warunkowej `if` i nie będzie dostępna poza tym blokiem. Słowo `const` pozwala natomiast tworzyć stałe. Można o nich myśleć tak samo jak o zmiennych z tą różnicą, że nie można zmienić ich wartości. Nie oznacza to jednak, że gdy do stałej przypiszemy jakiś obiekt, to staje się on niemodyfikowalny. W praktyce oznacza to jedynie, że nie można nadpisać wartości takiej stałej w dalszej części kodu, gdyż próba takiego działania wygeneruje błąd.

Jeszcze jednym z ciekawych rozwiązań, którego od zawsze brakowało w języku JavaScript, jest możliwość przypisywania domyślnych wartości parametrów. Dzięki temu definiując funkcję, możemy ustawić wybrany parametr na jakąś domyślną wartość, która zostanie użyta, jeśli argument nie zostanie przekazany przy wywołaniu tej funkcji.

To tylko niektóre z wielkich zmian, jakie zawitały do języka JavaScript dzięki ES6. I choć wsparcie dla tej specyfikacji w najnowszych przeglądarkach internetowych jest bardzo dobre, to aby w pełni cieszyć się poprawnym działaniem aplikacji w jak największej liczbie przeglądarek i środowisk, kod napisany w użyciem ES6 transpiluje się do kodu w standardzie ES5. I choć na pierwszy rzut oka to zadanie może się wydawać bezsensownym, to po wypróbowaniu nowych możliwości języka, obraz ten się zmienia. Dzięki takim narzędziom jak np. Babel, możemy cieszyć się możliwością pisania w ES6 i nie martwić się o to czy nasz kod zadziała w środowiskach, które nie obsługują w pełni (lub w ogóle) tego standardu.

Z solidnymi podstawami języka JavaScript w wersji ES5, z łatwością wskoczysz na kolejny poziom i wykorzystasz ułatwienia oraz nowości, płynące z nowego standardu ES6. Do dzieła!