JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 7 (2018) 158-161
Wysłane: 2017-11-29
Przyjęte: 2017-12-04

Porównanie efektywności szkieletów AngularJS i Meteor

Oleksandr Chornyi*, Marek Miłosz
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule przedstawiono rezultaty porównania szybkości tworzenia kodu i ładowania projektów w różnych frameworkach w celu zbadania, który z nich wydaje się być najlepszym wyborem w dłuższej perspektywie czasu. Analizę przeprowadzono pod kątem badania eksperymentalnego, wykonania oprogramowania przykładowego projektu oraz badania literaturowego. Przeprowadzone porównanie pozwoli wskazać framework o lepszej wydajności i prostszy w wykorzystaniu.

Słowa kluczowe: AngularJS; Meteor; framework; porównanie

*Autor do korespondencji.

Adres e-mail: oleksandr.chornyi@pollub.edu.pl

Effectiveness Comparison of the AngularJS and Meteor frameworks

Oleksandr Chornyi*, Marek Miłosz
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This article presents the results of comparing code developing speeds and project loads across different frameworks to explore which of them seems to be the best choice in the long time. The analysis was carried out in terms of the exploratory study, the design of the sample project and the literature review. Comparison will make it possible to point to a better performance framework and prolong its use.

Keywords: AngularJS; Meteor; framework; comparison;

*Corresponding author.

E-mail address: oleksandr.chornyi@pollub.edu.pl

1. Wstęp

Szkielety JavaScript nabrały ogromnej popularności wśród web-developerów, ponieważ pomagają one efektywniej i szybciej tworzyć strony internetowe i pisać ładny, czysty kod nie wykorzystując metod które zajmują dużo czasu. Struktura Javascript frameworków pozwala zmniejszyć czas opracowania aplikacji dzięki automatyzacji wielu modeli dla konkretnie wykorzystywanych celi. Pomimo tego frameworki pozwalają projektantowi pisać bezbłędny i czytelny kod. Framework implikuje operacje i dzięki temu framework pozwala programistom pracować mniej, a czynić więcej.

Najczęściej frameworki pracują szybko dzięki temu, że wykonują tylko to, co wymaga od ich użytkownik, podczas gdy CMS wykonuje o wiele więcej obliczeń i zapytań takich jak "on/off moduł", "if on" i inne podobne do tego funkcje.

Frameworki JavaScript dają programiście wiele plusów i przewag odpowiadających postawionemu zadaniu, lecz jednocześnie sprawiają niedogodność swoim mnogim wyborem, ponieważ już w chwili obecnej istnieje duża liczba różnych frameworków i każdy z nich ma swoje plusy jak i minusy. Co można zrobić w jedym frameworku, tego już się nie da w drugim i na odwrót. A więc należy dobrze pomyśleć i wybrać ten który odpowiada nam najlepiej.

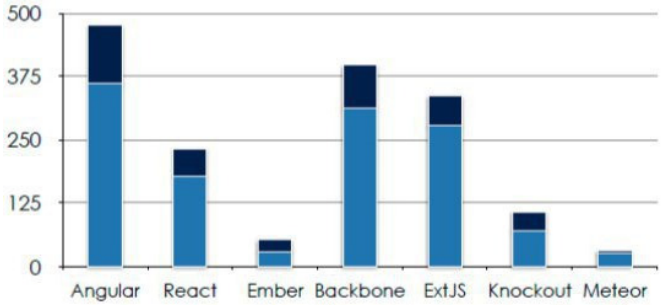
2. Framework AngularJS i framework MeteorJS

AngularJS został stworzony i opracowany przez dwóch programistów – Miszko Heweri i Adam Abronson w 2009 roku w celu zapewnienia usługi danych JSON i ułatwienia tworzenia aplikacji [1, 2]. Przed wydaniem AngularJS jako biblioteki open source dla użytkowników, był to pomysł na biznes umieszczony w domenie "GetAngular.com" [3]. Kolejna wersja Angular2 została wydana siedem lat później w 2016 r. I dopiero rok później, w kwietniu 2017 r. ukazał się nowy Angular4 [3].

Z kolei MeteorJS został stworzony 2 lata później, czyli w 2011 roku pod nazwą Skybreak, lecz dopiero na początku stycznia 2012 roku został użyty przez wszystkich programistów w zwykłym wyglądzie [4-6]. Oprócz tego MeteorJS można nazwać full - stuck frameworkim - to oznacza, że może on pracować tak samo po stronie klienta, jak i po stronie serwera.

Różnice pomiędzy frameworkami będą przedstawione w tablicy 1, ale nie warto ich dosłownie porównywać jako w całości jednakowe frameworki, ponieważ MeteorJS - to full - stack framework, a AngularJS - to mvc po stronie klienta. Jednak główne aspekty ich pracy można porównać i spróbować wyznaczyć lidera [7, 8-10].

Na rynku pracy AngularJS jest bardzo poszukiwanym językiem programowania. Skalę ofert pracy ukazuje poniższy rysunek, który został sporządzony i zaprezentowany na stronie internetowej freelancerów HeadHunter w 2016 r. [11].



Rys. 1. Wykres popytu i podaży na języki programowania w firmach zajmujących się wytwarzeniem oprogramowania. [11]

Jasnoniebieski kolor przyporządkowuje ilość osób poszukujących pracy, a granatowy kolor oznacza ilość zaproponowanych stanowisk pracy w poszczególnych językach programowania. Jak widać Meteor nie cieszy się dużym zainteresowaniem pracodawcór jak również przyszłych pracowników.

W tabeli 1 widać, że w niektórych momentach AngularJS przegrywa z MeteorJS, lecz nie można zapominać o tym, że dane Framework niezupełnie są równorzędne.

Tabela 1. Przewagi AngularJS i Meteor [3]

	angular js	**meteor js**
Full-stack framework	Nie, wyłącznie MVC po stronie klienta	Tak
Backend	Obojętnie	Node.js
Console Utility	Niema	Jest
Dynamiczny związek html z danymi po stronie klienta	Tak	Tak
Render html po stronie serwera	Nie	Tak, nie natywny
npm pakiety	Jest możliwość połączyć w browserify	Przez pakiety własnego menegera pakietów
Wielokrotne użycie między stroną klienta i serwerem	Niskie	Wysokie
REST API	Nie, ale jest możliwość dodania	Jest
DB	Jakakolwiek	Jakakolwiek
Odnowa dodatku bez uruchamiania ponownie	Nie	Tak html, css, js
Strona internetowa	Angular.io	Meteor.com

3. Metodyka badawcza

Podstawą metodologii badań tej pracy dyplomowej będzie opracowania naukowe na temat opracowania i napisania web-aplikacji. Zostaną rozpatrzone plusy i minusy każdego z frameworków względem siebie.

Niemałe znaczenie w praktycznej części tego badania ma uogólnianie zasad napisania kodu i szybkości pracy każdego z frameworków w równorzędnych warunkach.

Metodą porównania danych frameworków AngularJS i MeteorJS będą następujące punkty:

- Podłączanie frameworków do projektu
- Struktura napisania kodu(jednakowej funkcji)
- Szybkość ładowania strony
- Szybkość wykonania tej funkcji.

Porównanie szybkości ładowania strony będzie wykonane przy pomocy wzoru:

$$S = \frac{Z1 + Z2 + Z3 + \cdots + Z4}{K} \quad (1)$$

gdzie: Z – czas ładowania strony, K – liczba eksperymentów.

Średnia liczba napisanych linii kodu przedstawia się następująco:

$$E = \frac{\frac{S_{js}}{F_{js}} + \frac{S_{ts}}{F_{ts}} + \frac{S_{html}}{F_{html}}}{X} \quad (2)$$

gdzie: S_{js}- liczba napisanych linii, *js* - rozszerzenie pliku, F_{ts} - liczba plików z rozszerzeniem .ts, X – liczba plików.

Badania wykonane zostały na komputerze o następujących parametrach technicznych:

- Processor Intel Core i5-650 @ 3.20 GHz (4 Cores);
- RAM: 6 Gb.
- Windows 10

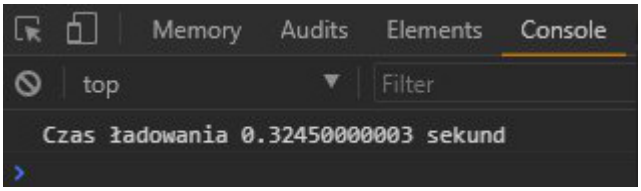
W celu wykonania eksperymentu będzie wykonane ładowanie projektów już ze statyczne dodanymi zadaniami w liczbie 5000 i 15000. Rezultaty będą zapisane do tablic i po wszystkich testach będą obliczone średnie szybkości ładowania stron.

Szybkość ładowania stron będzie realizowana za pomocą funkcji JavaScript:

Przykład 1. Funkcja licząca szybkość ładowania stron

```
var startTime = new Date();
function showElapsedTime() {
var testSiteUrl = location.href;
var testSiteString = String(testSiteUrl).slice
(testSiteUrl.indexOf("www"));
var endTime = new Date();
var platform = navigator.platform;
var msgString = "Время загрузки „ +
Number(elapsedTime/1000) + „секунд ";
document.getElementById(„vremia").innerHTML =
msgString;}onload = function() {showElapsedTime();}
```

Rezultaty powyższego kodu bedą wyświetlane w konsoli, którą włączamy w przeglądarce. Powyższy kod wykona się po załadowaniu strony.



Rys. 2. Wynik wykonanego skryptu w konsoli Google Chrome.

4. Realizacja badań i rezultaty

W tym rozdziale będzie opisane stworzenie projektu na AngularJS i Meteor. Celem projektu było stworzenie listy spraw do załatwienia - "To do list". Do stworzenia aplikacji zostanie wykorzystany AngularJS i dodatkowe utility-y takie jak stworzenie samego DOM poprzez zaprogramowanie HTML CSS (Cascading Style Sheets) oraz MeteorJS.

Przykład 2. Funkcje dodawania, usunięcia, tasku AngularJS

```
var todo_list = angular.module('TodoListApp'), [])
.controller('ListController', function($scope){
$scope.addItem = function (){
$scope.item.push({detauk: $scope.newItem.detail,
stat: 'Active'});
}
$scope.deleteItem = function(item){
var index = $scope.item.indexOf(item);
$scope.item.splice(index, 1);
}
$scope.completeItem = function (item){
item.stat = "Complete";
}
```

Przykład 3. Funkcje dodawania, usunięcia taska Meteor

```
'submit from': function(event){
event.preventDefault();
var todoName = $('[name="todoName"').val();
Todos.insert({
name: todoName,
completed: false,
createdAt: new Date()
});
}
'click .delete-todo': function(event){
event.preventDefault();
var documentId = this._id;
}
```

Powyższe listingi, pomimo róznic w składni wykonują się tak samo i mają identyczną funkcjonalność.

Rys. 2. Zrzut ekranowy prezentuje wykonanie obydwu funkcij.

Budowanie aplikacji w Meteorze wymaga krótszych linii kodu, ale działąca aplikacja wykorzystuje więcej tworzących ją plików czego nie można powiedzieć o Angularze.

Za pomocą frameworków AngularJS i MeteorJS istnieje możliwość stworzenia praktycznie identycznych web - aplikacji, typu landing page. Frameworki te są potężnymi narzędziami do tworzenia web - aplikacji, które ułatwiają programowanie jeśli porównujemy je z innymi tradycyjnymi frameworkami. Wybór frameworków najczęściej zależy od postawionych zadań i wymogów zleceniodawcy projektu. Należy patrzeć na oddzielne, konkretne przypadki i wybierać spośród nich najlepszy. Kiedy musimy szybko wykonać web-opracowanie gdzie trzeba użyć frameworków JavaScript i nie ma potrzeby wykorzystywania funkcji back-end-owych oraz pokazujących front-end. W takim wypadku oczywisty wybór pada na AngularJS.

AngularJS dość często nazywają MVW (Model - View - Whatever) frameworkiem i jego zaletą jest szybkie tworzenia kodu i natychmiastowe odzwierciedlenie przemian na stronie back - end po stronie klienta. Dzięki jego zasłużeniu w technice nazywają go najbardziej popularnym frameworkiem JavaScript dla opracowania jednostronicowych stosowań (SPA Single - Page - Applications) i może on pochlubić się największym wsparciem społeczności.

Tabela 2. Ilość plików i linii kodu wykorzystanych przez AngularJS

Rozszerzenie plika	Liczba plików	Liczba linii
.js	4	576
.ts	3	201
HTML	1	179
Suma	8	956

Tabela 3. Ilość plików i linii kodu wykorzystanych przez Meteor

Rozszerzenie plika	Liczba plików	Liczba linii
.js	3	423
.ts	2	152
HTML	1	115
Suma	6	670

Tabela 3. Porównanie liczby linii kodu AngularJS i MeteorJS

Rozszerzenie plika	Angular	Meteor
.js	144	141
.ts	67	76
HTML	179	115
Średnia	390	332

W tabeli 3 są zawarte wyniki liczby napisanych linii kodu dla każdego rozszerzenia plików i za pomocą formuły opisanej wyżej uzyskujemy ich średnią. Możemy zauważyć, że w Angularze wykorzystano więcej linii kodu i w perspektywie zajmowało to więcej czasu przy tworzeniu aplikacji.

W eksperymencie z wymiarami szybkości ładowania stron zrobionych zostało kilka pomiarów. Zadanie wykonane było na różnych przeglądarkach, ale na tym samym komputerze.

Tabela 4. Szybkość ładowania projektu bez statycznych tasków

Framework/Przeglądarka	Chrome	FireFox	Safari
AngularJS	0.323s	0.340s	0.331s
Meteor	0.396s	0.401s	0.339s

Ponieważ aplikacje nie mają określonego obciążenia, różnica czasu ładowania jest nieco inna. Jeśli spojrzymy na wyniki w kategoriach perspektywy możemy dostrzec, że struktura AngularJS jest ładowana szybciej niż framework Meteor. Wniosek jest taki, że przy dużych aplikacjach zaoszczędzimy czas ładowania stron korzystając z Angulara, a to wpłynie na ogólną wydajność.

W tabeli 5 pokazane są wyniki z dodaniem 5000 statycznych zadań a w tabeli 5 pokazane są wyniki z dodaniem 15000 statycznych zadań.

Tabela 5 Szybkość ładowania projektu z 5000 statycznych zadań

Framework/Przeglądarka	Chrome	FireFox	Safari
AngularJS	1.6s	1.89s	1.55s
Meteor	1.9s	2.03s	1.82

Tabela 6 Szybkość ładowania projektu z 15000 statycznych zadań

Framework/Przeglądarka	Chrome	FireFox	Safari
AngularJS	3.11s	3.30s	3.15s
Meteor	4.02s	4.10s	3.97s

Tabela 7 Szybkość ładowania projektu - średnia

Framework/Przeglądarka	Chrome	FireFox	Safari
AngularJS	3.11s	3.30s	3.15s
Meteor	4.02s	4.10s	3.97s

W tabeli 7 pokazane są średnie czasy ładowania stron biorąc pod uwagę wszystkie eksperymenty.

Dzięki temu eksperymentowi widać, że liczba linii napisanych w aplikacjach nie różni się w ogólnym zakresie, ale w przyszłości różnica ta będzie rosnąć i stawać się znacząca. Angular wymaga więcej linii kodu, aby stworzyć aplikację, ale wygrywa prędkością ładowania stron. Ponieważ głównym kryterium była szybkość pobierania aplikacji, Angular jest wyraźnym liderem, aczkolwiek wymaga więcej pracy w postaci pisania większej ilości kodu. AngularJS ładuje strony szybciej i w perspektywie większych projektów będzie pracował efektywniej.

5. Porównanie efektywności AngularJS i MeteorJS

Porównanie Meteor i Angular jest trochę skomplikowane, ponieważ w rzeczywistości są to bardzo różne frameworki. Cechą wspólną jest to, że są napisane w JavaScript. Meteor jest pełną strukturą, która pracuje na serwerze Node.js. Angular działa tylko po stronie klienta w przeglądarce. Są one odrębnymi szkieletami, dlatego opisano osobisty pogląd na każdy z ich i to, co u każdego z nich jest "rodzynkiem".

AngularJS i Meteor nie należą do tej samej kategorii. Dlatego też nie można uczyć się jednego, polegając na podobieństwie drugiego.

Obie struktury mają koncepcję tego, co Meteor nazywa "reaktywnymi źródłami danych". W Angularze mamy obiekt obwodu. Stwarza on hierarchię, w której możemy przypisać wartości i funkcje. Później mamy możliwość otrzymania dostępu do tych wartości ze swoich szablonów. Zmiany zostaną automatycznie zaktualizowane po zmianie wartości w obszarze, dając aplikacji pojedynczy punkt w swoim modelu. Angular umożliwia także monitorowanie tych źródeł danych i wykonywanie niestandardowych działań podczas zmiany wartości. Meteor zawiera kilka źródeł reaktywnych danych. Najprostszym przykładem jest sesja. Obiekt sesji umożliwia nam przechowywanie dowolnych par klucz-wartość w czasie rzeczywistym w pamięci przeglądarki. Kursory są kolejnym ważnym źródłem danych reaktywnych w Meteor.

Kursor jest zwracany przy zapytaniu bazy danych. Połączenie reaktywnych źródeł danych i podkategorii - to właśnie sprawia, że Meteor jest tak niezwykły. Gdy zmiany dotyczą danych na jednym kliencie, zmiany te są natychmiast odzwierciedlane w widokach i wysyłane do serwera, a następnie strumienie zmian są zwracane do wszystkich innych połączonych klientów. Kursory umożliwiają również śledzenie i obserwowanie zmian w danych.

6. Dyskusja

Angular2 dostarczany jest z ogromnym spisem funkcji, które pozwolą opracować wszystko, zaczynając od web-dodatków do desktop-owych i mobilnych zastosowań. Framework jest zbudowany na TypeScript od Microsoft którego celem jest stworzenie JavaScript bardziej wyrafinowanym i atrakcyjnym dla dużych projektów. Funkcje Angular2 posiadają architekturę na podstawie komponentów, udoskonalony DI (dependency injection - wprowadzenie zależności), związek między komponentami i tym podobne.

Porównanie AngularJS z Meteorem nie jest, być może, wyczerpujące z tej racji, że Meteor jest full-stuck frameworkiem, a AngularJS jest frameworkiem MVC. Badania nad frameworkami pokazują, że ze względu na prędkość wykonania kodu wygrywa AngularJS, a biorąc pod uwagę konieczną ilość tworzonych linii kodu zwycięzcą jest Meteor.

Na dziszejszy dzień główne kryterium oceny jest czas wykonania aplikacji i z tego wynika że AngularJS jest bardzej przyszłościowy.

Literatura

[1] AngularJS Web Application Development Blueprints. Vinci Rufus, 2014.

[2] Mastering MeteorJS Application Development. Jebin B V, 2015

[3] Mastering Web Application Development with AngularJS. Pawel Kozlowki, 2013.

[4] Node.js, MongoDB, and AngularJS Web Development. Brad Dayley, 2014.

[5] Meteor Design Patterns. Marcelo Reyna,.

[6] Learning AngularJS for .NET Developers. Alex Pop, 2015.

[7] Instant Meteor JavaScript Framework Starter. Gabriel Manricks, 2013.

[8] 2015Pro Angular – Edition 2. Adam Freeman, 2017.

[9] Getting Started with Gulp. Travis Maynard, 2016.

[10] Meteor: Full-Stack Web Application Development. Fabian Vogelsteller Isaac Strack Marcelo Reyna, 2016.

[11] https://hh.ru/ [18.03.2017]