Modern JavaScript frameworks: A Survey Study

Sanja Delčev, Dražen Drašković
Department of Computer Engineering and Informatics
University of Belgrade School of Electrical Engineering
Belgrade, Serbia
sanja.delcev@etf.bg.ac.rs, drazen.draskovic@etf.bg.ac.rs

***Abstract*— With the increasing popularity of the web, some new web technologies emerged and introduced dynamics to web applications, in comparison to HTML, as a static programming language. JavaScript is the language that provided a dynamic web site which actively communicates with users. JavaScript is used in today's web applications as a client script language and on the server side. The JavaScript language supports the Model View Controller (MVC) architecture that maintains a readable code and clearly separates parts of the program code. The topic of this research is to compare the popular JavaScript frameworks: AngularJS, Ember, Knockout, Backbone. All four frameworks are based on MVC or similar architecture. In this paper, the advantages and disadvantages of each framework, the impact on application speed, the ways of testing such JS applications and ways to improve code security are presented.**

Keywords – client server applications; MVC architecture; JavaScript; AngularJS; Ember; Knockout; Backbone;

I. Introduction

Applications that have been realized as desktop standalone are increasingly shifting to web and mobile platforms. Out of that need a lot of new client-server and mobile technologies and frameworks have been developed, and design patterns are actively used. Among the most popular architectures for improving web applications are: *Model View Controller* (MVC), *Model View Presenter* (MVP), *Model View ViewModel* (MVVM) and many others [1].

The aim of this research is to present four popular JavaScript frameworks – AngularJS, Ember, Knockout, and Backbone, to show their shortcomings and advantages, to give a side-to-side analysis according to certain criteria, and to conclude the analysis.

Chapter II includes a brief description of each of these frameworks, chapters III, IV, and V provide their comparative overview, advantages, and disadvantages of each of them. At the end, a conclusion is given.

II. Popular JavaScript frameworks

This chapter provides a short overview of the frameworks analyzed.

A. AngularJS

AngularJS is a frontend web application framework whose first version was designed by Google in 2010, and currently active is the version 1.7 from 2018 [2]. AngularJS is part of the MEAN stack consisting of MongoDB database, Express.js web application server framework, AngularJS and Node.js as server runtime environment.

It is based on MV* (*Model View Whatever*) architecture, which is slightly different from the MVC architecture. In MVC we differentiate model component, which is responsible for data management, view component, responsible for displaying the data to the user, and controller component, which controls the interaction between model and view and represents the business logic of the application. In MV* architecture, what connects model and view is not relevant, but it is important that every model change affects the view, and vice versa. In MVC, the controller is controlling the situation and consists of attributes and functions. It is defined using the ng-controller directive. Each controller accepts a $scope object as a parameter, or an object that refers the application. Scope object represents the medium between the controller and view and it is an object to which we add variables and functions. This object demonstrates a technique called dependency injection. In this way, the function does not depend on the variable in any way. Other important AngularJS directives are: ng-app, ng-init, ng-include, ng-model, ng-bind, ng-repeat, ng-show, ng-hide, ng-click.

B. BackboneJS

BackboneJS is a JavaScript library which provides a flexible, minimalistic solution for separating business logic from the user interface. It is based on an architecture similar to MVC, which abstracts data using models, DOM views and binds model and view with events. Backbone is dependent on JavaScript file underscore.js, which must be included together with the backbone.js file. The Backbone architecture contains the following: 1) HTTP Request – for sending requests from the client to the server; 2) Router – serves for connecting the application to actions and events using URL; 3) View – responsible for the presentational layer; 4) Events – triggering the appropriate function bound to the object over which the event occurred; 5) Model – contains application data and data logic; 6) Collection – list of models; 7) Data Source – connection to the database.

C. EmberJS

EmberJS is a JavaScript framework for developing web applications. It uses MVC architecture and has the following parts: 1) Model – each route has an associated model and showing the data from the model is the route's assignment; 2) View – they are created very rarely due to the existence of the

Authorized licensed use limited to: University of Glasgow. Downloaded on November 23,2021 at 23:09:39 UTC from IEEE Xplore. Restrictions apply.

Handlebar template; 3) Template – a standard template which has a header, footer, and at least one {{outlet}}, an expression which will load the specified content depending on the current URL; 4) Controller – manages the display logic and controls the operations between the route, model, and view; 5) Route – defines how the application will be represented to the user, loads the templates and necessary data, while the routers determine which routes should be loaded depending on the given URL. Relevant features of Ember are: computed properties, observers, bindings.

D. KnockoutJS

KnockoutJS is a JavaScript library based on the MVVM architecture, derived from the MVC pattern. It provides us the support for building rich applications, with which we can interact. View represents the user interface built with HTML. View binds to ViewModel, so the changes occurred in ViewModel are automatically propagated to the View using a concept of two-way data binding. ViewModel is a class which binds the Model and View, i.e. binds real data with their user representation. Three important concepts on which this framework is based are: observables, data binding, and templates. Components structure applications and provide code reusability.

III. Analysis of General Information

In the first step of this study, we compared the frameworks' representation on the Internet, framework's size, and their influence on the application speed, as shown in Table I.

Figure 1 shows the graph of interest over a period from January 2010 until March 2018. AngularJS is the most common framework on the Internet. It offers plenty of written and video tutorials, which make learning it easier and shorten the time necessary to familiarize yourself with the framework. Backbone had a great interest from the users in a period between May 2012 and August 2013, but since then the interest for this framework is declining. AngularJS experienced a sudden expansion, starting in 2013, and to this day this framework left the other three behind.

TABLE I. Representation on the Internet

Criteria of comparison	Framework			
	AngularJS	***Backbone***	***Ember***	***Knockout***
Google results	~ 105 million	~ 45.5 million	~ 6 million	~ 49.4 million
Stack Overflow results	252 317	20 505	22 507	19 316
GitHub projects/code	380 084	24 191	29 912	5 693
Number of stars on GitHub	36 669	27 192	19 003	8 895
YouTube results	~ 800 thousand	~ 29 thousand	~ 42 thousand	~ 17 thousand
Basic size of FW [KB]	39.5	6.5	90	55
Size with all dependent libraries [KB]	39.5	43.5 (jQuery +Undersc.) 20.6 (Zepro +Undersc.)	132.2 (jQuery + Handle-bars)	55

Another important factor in comparing these frameworks is their size. Since the users do not have a lot of patience when it comes to the application loading time, this factor is important when it comes to web applications. Minimized version size comparison is also shown in Table I.

Backbone is more a library than a framework and research has shown that it is good for making fast applications that respond quickly to user actions. There are many plugins for Backbone which can be integrated with it to get the desired functionalities. Compared to AngularJS, Ember and Knockout, which have larger libraries, make it easier for us to do things by doing them themselves, such as automatic refreshment of the view component when changing the model, it is not the case with Backbone, and working with it produces more code. Less code means easier testing and less room for errors.

Ember has the largest library, but it can be an advantage as it comes with a lot of built-in support. Ember can be the right choice for large projects.

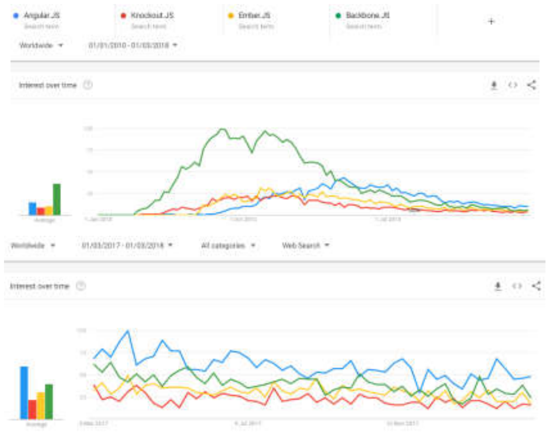


Fig. 1. Graph of user interest in the period 2010-2018 and in the period 1.3.2017 - 1.3.2018.

IV. Analysis of Testing and Security

All four frameworks support different types of testing, both functional (black box) and structural (white box). The most famous JavaScript testing modules are supported in these technologies, such as Karma and Jasmine for AngularJS, Mocha, Jasmine and Sinon for Backbone, and many others.

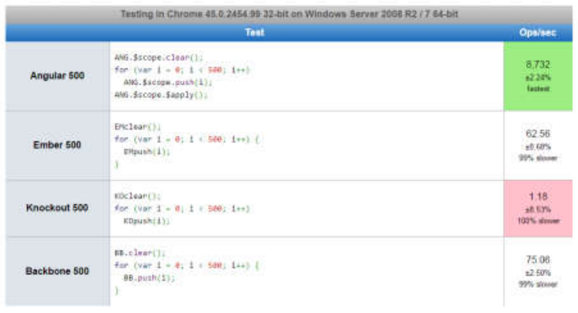
In AngularJS, dependency injection is fully supported, which is a great advantage when it comes to software testing. There are different extensions in web browsers such as Google Chrome to help with debugging, developed by teams that have developed the analyzed frameworks. Debugging AngularJS can be achieved using Angular Batarang. When Batarang is loaded, the scope and concrete element that interests us can be chosen. In addition to reading the value of the elements, Batarang provides the ability to test performance, visualize the dependency graph, etc. A plugin that works with Ember is Ember Inspector. This tool provides a view of the various parts of the Ember application, model handling, and a detailed view of the route. Knockout Context Debugger adds sections with elements that show current binding for any selected element.

Authorized licensed use limited to: University of Glasgow. Downloaded on November 23,2021 at 23:09:39 UTC from IEEE Xplore. Restrictions apply.

Backbone Debugger displays in real-time all the models, collections, views, and roles in the application.

A. *Performance*

The speed of the application is influenced by other factors besides the size, so we cannot with certainty determine which framework is faster. Application speed also depends on the structure of the source code of our application. For comparison purposes, attached are the results of a short test, which prints 500 numbers, launched in all four analyzed technologies [6]. The test shows the number of operations per second in each of the technologies. In this test example, shown in Fig. 2, AngularJS has shown the best performance.



Testing in Chrome 45.0.2454.99 32-bit on Windows Server 2008 R2 / 7 64-bit

	Test	Ops/sec

Angular 500	`ANG.$scope.clear();`
	`for (var i = 0; i < 500; i++)`
	`  ANG.$scope.push(i);`
	`ANG.$scope.$apply();`
		8,732
		±2.24%
		fastest

Ember 500	`EMclear();`
	`for (var i = 0; i < 500; i++) {`
	`  EMpush(i);`
	`}`
		62.56
		±0.60%
		99% slower

Knockout 500	`KOclear();`
	`for (var i = 0; i < 500; i++)`
	`  KOpush(i);`
		1.18
		±8.63%
		100% slower

Backbone 500	`BB.clear();`
	`for (var i = 0; i < 500; i++) {`
	`  BB.push(i);`
	`}`
		75.08
		±2.50%
		99% slower

Fig. 2. Performance testing of the analyzed frameworks

B. *Security*

The tool Retire.JS helps in discovering the vulnerabilities of our JS applications. Retire.JS recommends improving five different versions of AngularJS based on found vulnerabilities, one version of Backbone and fifteen versions of Ember. Both AngularJS and Backbone do not have Common Vulnerabilities and Exposures (CVE), however Ember has five public CVE documents.

The most vulnerable places in the application are the templates. They are the places where the DOM is most frequently attacked. While Backbone and Ember do not use their own template manipulation library, AngularJS uses its own template system. Knockout allows execution of an arbitrary JavaScript that is injected into HTML5 data attributes, so this is the most attacked part. Every JS program code located within the databind attribute will be executed once when the ko.applyBindings function is called. All four libraries are vulnerable regarding to this factor to a varying degree, as studied in the project Mustache-security from 2013 [7]. The ranking system for determining how secure is the template system is shown with the following characteristics:

- {}SEC-A Are template expressions executed without using eval or Function? (yes = pass)
- {}SEC-B Is the execution scope well isolated or sand-boxed? (yes = pass)
- {}SEC-C Can only script elements serve as template containers? (yes = pass)
- {}SEC-D Does the framework allow, encourage or even enforce separation of code and content? (yes = pass)
- {}SEC-E Does the framework maintainer have a security response program? (yes = pass)
- {}SEC-F Does the framework allow or encourage safe Content Security Policy (CSP) rules to be used (yes = pass)

The results for analyzed frameworks are given in Table II. Instead of Backbone, underscore.js is used, because Backbone uses its templates.

TABLE II. TESTING THE SAFETY OF THE ANALYZED FRAMEWORKS

Test case	**Framework**				
	Angular JS 1.2	***Angular JS 1.4***	***Under-score.js***	***Ember***	***Knockout***
{} SEC-A	Fail	Fail	Fail	Fail	Fail
{} SEC-B	Pass	Pass	Fail	Pass	Fail
{} SEC-C	Fail	Fail	Pass	Pass	Fail
{} SEC-D	Fail	Pass	Fail	Fail	Fail
{} SEC-E	Pass	Pass	Fail	Pass	Fail
{} SEC-F	Pass	Pass	Fail	Not defined	Fail

V. PROS AND CONS OF ANALYZED JAVASCRIPT FRAMEWORKS

Backbone represents only the core of the application, or the way the code is edited. We can enrich Backbone with various plugins; without them it's a small library that does not bring many new things. AngularJS is the fastest growing and fastest-changing, and for that reason IT companies still do not largely opt for that technology.

Ember is similar to AngularJS in terms of the functionality it offers, but a lot of content available on the Internet about it does not work in practice.

Knockout does not have routing integrated, which is available in AngularJS. AngularJS also includes dependency injection concepts, provides many services ($http, $log, ...), modules, filtering, form validation, etc. For example, these concepts are not implemented in Knockout. Knockout can be used as framework in less complex applications, which do not require great user interface display control.

Table III shows the comparison from the architecture side of the system and the technical possibilities. As the first criterion for comparison, the architecture of the frameworks was analyzed. All four technologies are based on variations of MVC architecture. Backbone is assigned to the MVP architecture, where the HTML and DOM represent the View component, and the Backbone View represents the Presenter component.

The second criteria for the analysis was the method for using templates. Backbone uses the library underscore.js for handling templates, Ember uses handlebars.js, while Knockout has its own integrated system for template handling. Then, the view and models binding and the method of refreshing the web pages were analyzed. AngularJS, Ember and Knockout offer an automatic refresh view when the model changes, as opposed to Backbone, in which the render function must be

Authorized licensed use limited to: University of Glasgow. Downloaded on November 23,2021 at 23:09:39 UTC from IEEE Xplore. Restrictions apply.

called every time we want to refresh the View, which leaves a lot of control to the programmer.

One of the problems encountered in the research was the problem of memory leak when working with Backbone. Therefore, if the user is not familiar enough with JavaScript, it is better to choose one of the remaining three frameworks.

TABLE III. ARCHITECTURE AND TECHNICAL POSSIBILITIES

Criteria of comparison	**Framework**			
	AngularJS	***Backbone***	***Ember***	***Knockout***
Architecture	MV*	MVP	MVC	MVVM
Templates	Built-in template system	Uses underscore.js library	Uses handlebars .js library	Built-in template system
Built-in validation	yes	no	no	no
Building of views and models	automatic	manual	automatic	automatic
Compatibility with other libraries	excellent	good	good	good

Table IV contains the following criteria of comparison: technical documentation and web browsers support, or compatibility with certain web browsers on mobile platforms. Technical documentation is displayed on a scale from 1 (worst) to 5 (best), where the assessment included the documentation of the official websites of these frameworks, sites with official tutorials like w3schools and others, and official forums with the support of active users.

TABLE IV. DOCUMENTATION AND WEB BROWSER COMPATIBILITY

Frame-work	**Criteria of comparison**	
	Technical documentation	***Web browser support***
Angular JS	3 / 5	Opera, Firefox, Chrome, IE6/IE7 (before v1.2), IE8 (before v1.3), Chrome Mobile, iOS Safari, Google AndroidOS browser
Backbone	hoo5 / 5	Opera, Firefox, Chrome, IE7+, Safari, iOS Safari, Google AndroidOS browser
Ember	2.5 / 5	Opera, Firefox, Chrome, Safari, IE8 (between versions 1.3 and 2.0), IE9, Opera Mini, iOS Safari, Google AndroidOS browser
Knockout	5 /5	Opera, Firefox, Chrome, IE6 – IE11, Apple Safari for Mac, iOS Safari, Google AndroidOS browser, Opera Mini

An analysis of the time required for learning the technologies in this paper was carried out and the results are shown in Table V. The shortest time to get familiar with the framework was with Knockout, and the longest with Backbone. AngularJS is the most complicated out of the frameworks analyzed, but it has a large number of good tutorials that make learning easier. Also, the syntax of these JavaScript frameworks was analyzed. AngularJS syntax is the easiest, because it requires the least amount of additional code, Knockout is slightly more difficult, and Backbone and Ember require a lot of additional code that is necessary to write. The ratings are shown in Table V, on a scale from 1 (worst) to 5 (best). Table V also shows some of the well-known client companies using these frameworks, and according to this criterion, Backbone stands out the most.

TABLE V. LEARNING TIME AND COMPLEXITY

Framework	**Criteria of comparison**		
	Learn time	***Complexity of the syntax***	***Companies/applications that are clients***
AngularJS	4 days	5 / 5	YouTube, Google, yap. TV Facebook Application, Nike, Huffington Post
Backbone	5 days	3 / 5	Twitter, Foursquare, LinkedIn Mobile, Soundcloud
Ember	3 days	3 / 5	Apple Music, imgix Sandbox, Fitbot, Solid, Microsoft, Netflix
Knockout	1 day	4.5 / 5	Azure, Childrensplace, Vivaaerobus, Dotnetnuke

VI. CONCLUSION

In this research paper the current top-level frameworks based on JavaScript technology analysis is given. There are many factors influencing the selection of a framework, and primarily the web application structure needs to be known and what kind of help should be obtained from the framework.

The analysis showed that Backbone is significantly different from AngularJS, Ember and Knockout, because it leaves a lot of decision making to developers. The remaining three frameworks provide much more finished functionalities and take a lot of the decision making themselves. Functionalities provided by AngularJS, Ember and Knockout are automatic content updates, data management through the program, relatively simple filtering of content based on a certain criterion, list sorting, etc.

All four frameworks separate the structure of the program into different parts, therefore creating a more readable code, which is very important for the development of large web systems. The job division into well-defined parts is provided by their architecture, which is based on the variation of the MVC. Each of the frameworks also has some bad sides, which are related to application security, testing, execution speed and technical documentation.

REFERENCES

[1] P. Thung, C. Ng; S. Thung, S. Sulaiman, “Improving a Web Application Using Design Patterns: A Case Study,” International Symposium in Information Technology 2010, IEEE, Kuala Lumpur, Malaysia, Sept. 2010

[2] AngularJS documentation, https://docs.angularjs.org

[3] Backbone documentation, http://backbonejs.org/

[4] Ember documentation, http://emberjs.com/

[5] Knockout documentation, http://knockoutjs.com/documentation/

[6] JavaScript performance playground, https://jsperf.com/

[7] Mustache security, https://code.google.com/archive/p/mustache-security

[8] D. Sofer “What Should Beginners Choose: AngularJS, Ember.js, or Backbone.js?”, https://www.codementor.io/proloser/what-should-beginners-choose-angularjs-ember-js-or-backbone-js-8r7xkdved

Authorized licensed use limited to: University of Glasgow. Downloaded on November 23,2021 at 23:09:39 UTC from IEEE Xplore. Restrictions apply.