talentica

DREAM · BUILD · DISRUPT

TechWatch Report – Javascript Libraries and Frameworks

Date: February 2018

Created By: Prateek Vijan, Sanjeevan Biswas

Contributors: Vrushali Malushte, Sridatta Pasumarthy, Mayank Kansal, Arindam Nayak

Contents

1 Analysis

What we are plotting – Capabilities Vs Ease of Adoption

What we mean by Capabilities and Ease of Adoption

2 Capabilities

2.1 Feature completeness

- **What we mean by this?**
  List of primary features supported by the library.

- **How we measured the feature completeness?**
  Number of basic Features supported (out of the box):

  - Routing
  - Data Binding
  - Model/Data management
  - Dependency Injection
  - Ajax Support
  - Web Sockets
  - History Management
  - Form Validation
  - Server Side Rendering
  - Mobile Support
  - Visual Designer Tools
  - Reusable Components Development
  - Deep Linking
  - I18N (Internationalization Support)
  - Source code Available
  - Logging
  - SEO (Search Engine Optimization)

- **How we rated each library – basis for rating**
  All Ratings are on the scale of 0 to 5 where 0 is worst and 5 is the best.

Rating	Criteria
5	Feature is directly supported by library itself.
3 / 4	Feature is supported by a stable / standard third party library. Stable means having good community support and adoption.
1 / 2	Feature is supported by an unstable third party library. Means not having good community support and adoption.
0	Feature is not supported at all.

- **Weightage**
  Feature completeness has a weightage of 30%.

2.2 Library Footprint

- **What we mean by this?**
  The actual size of the library to be included in the code and gets loaded on to the browser on page load.

- **How we rated each library?**
  All Ratings are on the scale of 0 to 5 where 0 is worst and 5 is the best.

Rating	Criteria
5	Size less than or equals 100 KB
4	Size less than or equals 200 KB
3	Size less than or equals 300 KB
2	Size less than or equals 400 KB
1	Size less than or equals 500 KB
0	Size less than or equals 1 MB

- **Weightage**
  It has a weightage of 5%.

2.3 Scalability

- **What we mean by this?**
  Scalability is the ability of an application or product to continue to function well even if it (or its context) is changed in size or volume. Typically, the rescaling is to a larger size or volume.

- **How we measure this?**
  We designed a sample application for each library and evaluated them for rendering pages containing bulk amount of data.

  - Average memory usage for loading 100, 200, 500, 100 and 2500 records.
  - Measured memory leakages for constructing and destroying pages with 100, 200, 500, 100 and 2500 records.

- **How we rate each library?**
  This parameter was evaluated using the profile logs of Chrome browser's console window. The libraries were rated on a scale of 1 to 5.

- **Weightage**
  It has a weightage of 25%.

2.4 Performance

- **What we mean by this?**
  Performance is the degree to which a software system or component meets its objectives for timeliness.

- **How we measured this?**
  We designed a sample application for each library and evaluated them for rendering pages containing bulk amount of data. We then measured:
  - Time taken for initial loading and rendering
  - Average time taken for rendering pages containing 100, 200, 500, 100 and 2500 records.

- **How we rated each library?**
  This parameter was evaluated using the timeline logs of Chrome browser's console window. The libraries were rated on a scale of 1 to 5.

- **Weightage**
  It has a weightage of 30%.

2.5 Security and Vulnerability

- **What we mean by this?**
  Security is the measure of the system's ability to resist unauthorized usage while still providing its service to legitimate users.

- **How we measured this?**
  Since front-end frameworks mostly rely on back-end to return validated data only for the current user, the only thing that we can measure is the use of any vulnerable libraries or code by the library. Retire.js is one such tool that does this sort of analysis.

- **How we rated each library?**
  We rated each library based on the score evaluated by Retire.js.

- **Weightage**
  It has a weightage of 5%.

2.6 Interoperability

- **What we mean by this?**
  The ability of the system to provide services to and accept services from other systems, and to use the exchanged services to enable them to operate effectively together. Most of the JavaScript libraries work independently and don't cause interoperability issues. Two things that can cause an issue (i.e.

that are shared between libraries) are the page DOM and the global namespace.

- **How we measured this?**
    - Number of variables added to the global namespace by the library.
    - If any API updates DOM which is out of scope of the passed element (for example, adding a parent/sibling).
    - If API allows undoing of the DOM changes. For example, most of the jQuery plugins provide a destroy method to revert whatever DOM changes were made. We looked for 'destroy' like APIs or whether the library architecture provides a destructor interface, i.e. a way to revert DOM changes. For example, componentWillUnmount method for ReactJS.

- **How we rated each library?**
  We evaluated each library against each measureable point and rated them on a scale of 1 to 5.

- **Weightage**
  It has a weightage of 5%.

3 Ease of Adoption

3.1 Ease of development

- **What we mean by this?**
  The time any new technology will take to learn, adapt, implement and maintain.

- **How we measured this?**
  We measured it based on the following points:

  - Availability of developer tools for the library
  - Learning curve required for the library
  - Testability
    How we measured Testability?
    - Number of test files against number of source files
    - Use analysis tool such as coveralls.io

- **How we rated each library – basis for rating**
  We evaluated each library against each measureable point and rated them on a scale of 1 to 5.

- **Weightage**
  It has a weightage of 50%.

3.2 Community Support

- **What we mean by this?**
  The extent of help and support you can get for a particular library. It is also a measure of how effectively the library is being used in the JS developer world.

- **How we measured this?**
  We measured it based on the following points:

  - Q & A in Stackoverflow
  - Availability of open source plugins and components
  - Ease of integration of 3rd party libraries
  - Number of issues open/closed in GitHub

- **How we rated each library?**
  We evaluated each library against each measureable point and rated them on a scale of 1 to 5.

- **Weightage**
  It has a weightage of 30%.

3.3 Paid / Open Source

- **What we mean by this?**
  Whether the tool is proprietary or open source.

- **How we rated each library?**
  - We rated 5 if the library is free, open-source software licensed under the MIT License.
  - We rated 2, 3, 4 based on license agreement.
  - We rated 1 if the library is paid.

- **Weightage**
  It has a weightage of 20%.

4 JS Libraries and Frameworks

List of JS libraries and frameworks we considered.

4.1 AngularJS 2.0:

This is the second version of the AngularJS web framework. Angular 2 takes a web component-based approach to building powerful applications for the web. It is used along with TypeScript which provides support for ECMAScript 5, ECMAScript 6, and ECMAScript 7.

Capabilities

Feature completeness	4.22
Library Footprint	2
Scalability	2.5
Performance	4
Security/Vulnerability	3
Interoperability	5

Ease of Adoption

Ease of development	2.5
Community support	2
Paid / Open source	5

Ratings

Capabilities : 3.6

Ease of Adoption : 2.85

4.2 AngularJS 5.0:

AngularJS5 focused on making the framework smaller and faster to use. Emphasis has been provided on making it easier to build progressive web apps, which are browser-based apps that offer superior, native-like experience.

Capabilities

Feature completeness	4.55
Library Footprint	4
Scalability	3.1
Performance	3
Security/Vulnerability	4
Interoperability	5

Ease of Adoption

Ease of development	4
Community support	3
Paid / Open source	5

Ratings

Capabilities	*: 3.75*
Ease of Adoption	*: 4*

4.3 ReactJS:

Is an open-source JavaScript library providing a view for data rendered as HTML. React views are typically rendered using components that contain additional components specified as custom HTML tags. React promises programmers a model in which sub-components cannot directly affect enclosing components ("data flows down"); efficient updating of the HTML document when data changes; and a clean separation between components on a modern single page application.

Capabilities

Feature completeness	4.28
Library Footprint	4
Scalability	4
Performance	4
Security/Vulnerability	5
Interoperability	4

Ease of Adoption

Ease of development	3.5
Community support	3.5
Paid / Open source	5

Ratings
Capabilities : 4.14
Ease of Adoption: 3.8

4.4 ReactJS16:

ReactJS16 comes with a new core architecture named 'Fiber'. With 'Fiber', there have been significant changes to the internals of the framework while keeping the public API essentially the same. It supports new rendered return types: fragments & strings. The error handling & server side rendering have been enhanced and the library footprint is smaller than the earlier versions.

Capabilities

Feature completeness	4.44
Library Footprint	4
Scalability	4.1
Performance	4
Security/Vulnerability	5
Interoperability	4

Ease of Adoption

Ease of development	3.5
Community support	4
Paid / Open source	5

Ratings
Capabilities : 4.18
Ease of Adoption : 3.95

4.5 VueJS:

VueJs is a progressive framework for building user interfaces. Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects.

Capabilities

Feature completeness	4.3
Library Footprint	5
Scalability	4.25
Performance	4.5
Security/Vulnerability	4
Interoperability	5

Ease of Adoption

Ease of development	2.25
Community support	0.5
Paid / Open source	5

Ratings

Capabilities	*: 4.42*
Ease of Adoption	*: 3.09*

4.6 ExtJS:

Ext JS is a pure JavaScript application framework for building interactive cross platform web applications using techniques such as Ajax, DHTML and DOM scripting. Originally built as an add-on library extension of YUI by Jack Slocum in April 15, 2007, **Ext JS** includes interoperability with jQuery and Prototype.

Capabilities

Feature completeness	3.67
Library Footprint	1
Scalability	5
Performance	5
Security/Vulnerability	4
Interoperability	3

Ease of Adoption

Ease of development	2
Community support	2.5
Paid / Open source	1

Ratings

Capabilities	*: 4*
Ease of Adoption	*: 1.95*

4.7 BackboneJS:

Backbone.js gives structure to web applications by providing **models** with key-value binding and custom events, **collections** with a rich API of enumerable functions, **views** with declarative event handling, and connects it all to your existing API over a RESTful JSON interface.

Capabilities

Feature completeness	3.33
Library Footprint	5
Scalability	3.5
Performance	1
Security/Vulnerability	2
Interoperability	2

Ease of Adoption

Ease of development	3.5
Community support	4.5
Paid / Open source	5

Ratings

Capabilities	*: 2.63*
Ease of Adoption	*: 4.1*

4.8 jQuery:

jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery is the most popular JavaScript library in use today, with installation on 65% of the top 10 million highest-trafficked sites on the Web. jQuery is a free, open-source software licensed under the MIT License.

Capabilities

Feature completeness	2.27
Library Footprint	5
Scalability	3
Performance	1
Security/Vulnerability	1
Interoperability	4

Ease of Adoption

Ease of development	3.5
Community support	5
Paid / Open source	5

Ratings

Capabilities : 2.24
Ease of Adoption : 4.25

5 Our recommendation

Go with React JS. It is an Open source, easy to learn and maintain, and most importantly provides you a component-based design. Using ES6 and JSX you can write compact and smart code with lesser number of lines as compared to the traditional way of coding. You can go with React-native too if you are planning to design a cross-platform mobile based application.
Instead of using Redux for react, you can create your own flux based architecture which provides you a unidirectional flow between Views -> Actions -> Models. It is because redux is quite bulky and loads your application.

AngularJS 2.0 also provides you a component-based design but if we compare performance and ease of adoption, it is slightly below par compared to ReactJS.

VueJS is another framework that's gaining popularity due to the ease of integration with other libraries. In terms of ease of adoption, it ranks lower than ReactJS.

6 References

- https://facebook.github.io/react/
- https://angular.io/docs/ts/latest/
- http://backbonejs.org/
- https://www.sencha.com/products/extjs/
- https://jquery.com/
- https://github.com
- http://stackoverflow.com/
- https://vuejs.org/

Additional Details:

https://talenticaall-my.sharepoint.com/personal/sanjeevan_biswas_talentica_com/_layouts/15/guestaccess.aspx?guestaccesstoken=boHZGOz4fbAigbdoqJpia6cPkljixpoL+90yE9j5yc0=&docid=0c69a635522dd4a1a95a8909b4510a5cc&rev=1

You can download or fork the sample applications used for the analysis of JS libraries:

https://github.com/Talentica/TechWatchJS