

WEB APP DESIGN AND DEVELOPMENT

ASSIGNMENT WEEK 4

TOP 10 JAVASCRIPT FRAMEWORKS

(SUBMISSION DATE: December 7, 2017)

SUBMITTED BY - HARIPRASATH PERIYASAMY

**GUIDED BY - PROF. MR. ANIRUDDHATREYA RAJAGOPALAN**

**Libraries**

A library is an organized collection of useful functionality. A typical library could include functions to handle strings, dates, HTML DOM elements, events, cookies, animations, network requests, and more. Each function returns values to the calling application which can be implemented however you choose. Think of it like a selection of car components: you’re free to use any to help construct a working vehicle but you must build the engine yourself.

Libraries normally provide a higher level of abstraction which smooths over implementation details and inconsistencies. For example, Ajax normally relies on the XMLHttpRequest API but this requires several lines of code and there are subtle differences across browsers. A library may provide a simpler ajax() function so you’re free to concentrate on higher-level business logic.

A library could cut development time by 20% because you don’t have to worry about the finer details. The downsides:

a bug within a library can be difficult to locate and fix

there’s no guarantee the development team will release a patch quickly

a patch could change the API and incur significant changes to your code.

**Frameworks**

A framework is an application skeleton. It requires you to approach software design in a specific way and insert your own logic at certain points. Functionality such as events, storage, and data binding are normally provided for you. Using the car analogy, a framework provides a working chassis, body, and engine. You can add, remove or tinker with some components presuming the vehicle remains operational.

A framework normally provides a higher level of abstraction than a library and will help you rapidly build the first 80% of your project.

The downsides:

* the last 20% can be tough going if your application moves beyond the confines of the framework
* framework updates can be difficult – if not impossible
* core framework code and concepts rarely age well. Developers will always discover a better way to do the same thing.

Top frameworks are discussed below based on the popularity:

### **jQuery**

| **jQuery** | |
| --- | --- |
| type | library |
| website | [jquery.com](http://jquery.com/) |
| repository | [github.com/jquery/jquery](https://github.com/jquery/jquery) |
| current version | 3.2.1 |
| developer | jQuery team |
| launch date | August 2006 |
| typical size | 30kb min |
| typical use | general purpose |
| usage | [72.4% of all websites](https://w3techs.com/technologies/details/js-jquery/all/all) |

jQuery remains the most-used JavaScript library ever created and is distributed with WordPress, ASP.NET and several other frameworks. It revolutionized client-side development by introducing CSS selector to DOM node retrieval plus chaining to apply event handlers, animations, and Ajax calls.jQuery has fallen from favor in recent years but remains a viable option for projects which require a sprinkling of JavaScript functionality.

**Pros:**

* small distribution size
* shallow learning curve, considerable online help
* concise syntax
* easy to extend

**Cons:**

* adds a speed overhead to native APIs
* less essential now that browser compatibility has improved
* usage has flat-lined
* some [industry](https://www.sitepoint.com/do-you-really-need-jquery/?aref=cbuckler) [backlash](http://youmightnotneedjquery.com/) against unnecessary use.

### **React**

| **React** | |
| --- | --- |
| type | library |
| website | [facebook.github.io/react/](https://facebook.github.io/react/) |
| repository | [github.com/facebook/react](https://github.com/facebook/react) |
| current version | 15.5.4 |
| developer | Facebook and contributors |
| launch date | March 2013 |
| typical size | 21kb min |
| typical use | single-page applications |
| usage | [low](https://w3techs.com/technologies/details/js-react/all/all) |

Perhaps the most-talked about library of the past year, React claims to be a JavaScript library for building user interfaces. It focuses on the “View” part of Model-View-Controller (MVC) development and makes it easy to create UI components which retain state. It was one of the first libraries to implement a virtual DOM; the in-memory structure computes the differences and updates the page efficiently.

React usage appears low in statistics perhaps because it’s used in applications rather than websites. Almost [38% of developers claim to be using the library](https://www.sitepoint.com/front-end-tooling-trends-2017/#librariesandframeworks).

**Pros:**

* small, efficient, fast and flexible
* simple component model
* good documentation and online resources
* server-side rendering is possible
* currently popular and experiencing rapid growth

**Cons:**

* New concepts and syntaxes to learn
* build tools are essential
* can require other libraries or frameworks to provide the model and controller aspects
* can be incompatible with code and other libraries which modify the DOM

### **Lodash**

| **Lodash** | |
| --- | --- |
| type | library |
| website | [lodash.com/](https://lodash.com/) |
| repository | [github.com/lodash/lodash/](https://github.com/lodash/lodash/) |
| current version | 4.17.4 |
| developer | John-David Dalton |
| launch date | April 2012 |
| typical size | 4kb – 24kb min |
| typical use | general purpose |
| usage | [low](https://w3techs.com/technologies/details/js-lodash/all/all) |

Lodash are the functional JavaScript utilities to supplement native string, number, array and other primitive object methods. There is some overlap so you are unlikely to require both libraries in a single project.

Client-side usage appears low but either library can be adopted for server-side Node.js applications.

**Pros:**

* small and simple
* easy to learn with good documentation
* compatible with most libraries and frameworks
* does not extend built-in objects
* can be used on the client or server

**Cons**:

* some methods are available in ES2015 and later editions of JavaScript.

### **Angular 2.x (now 4.x)**

|  |  |
| --- | --- |
| Type | Framework |
| Website | Angular.io |
| Repository | github.com/angular/angular.js |
| Current Version | 4.1 |
| Developer | Google |
| Launchdate | Sept, 2016 |
| Typical Size | 450 Kb |
| usage | High |

Angular 2.0 was released in September 2016. It was a complete rewrite which introduced a modular component-based model created with TypeScript (which is compiled to JavaScript). To add to the confusion, version 4.0 was released in March 2017 (v3 was skipped to avoid semantic version issues).

Angular 2+ is radically different to v1. Neither is compatible with the other – perhaps Google should have given the project a different name!

**Pros:**

* a single solution to produce modern web applications
* still part of the MEAN stack although fewer [Angular 2+ tutorials](https://www.sitepoint.com/angular-2-tutorial/) are available
* TypeScript provides some advantages to those familiar with statically-typed languages such as C# and Java.

**Cons:**

* steeper learning curve than some alternatives
* large code base
* impossible to upgrade from Angular 1.x
* Angular 2.x has experienced relatively low uptake compared to 1.x
* despite being a Google project, Google do not appear to use it?

### **Vue.js**

| [**Vue.js**](http://vue.js/) | |
| --- | --- |
| type | framework |
| website | [vuejs.org](https://vuejs.org/) |
| repository | [github.com/vuejs/vue](https://github.com/vuejs/vue) |
| current version | 2.0 |
| developer | Evan You |
| launch date | February 2014 |
| typical size | 19kb min |
| typical use | single-page applications |
| usage | [low](https://w3techs.com/technologies/details/js-vuejs/all/all) |

Vue.js is a lightweight progressive framework for building user interfaces. The core offers a React-like virtual DOM-powered view layer which can be integrated with other libraries but it is also capable of powering single-page applications. The framework was created by Evan You who previously worked on AngularJS but wanted to extract the parts he liked.

Vue.js uses an HTML template syntax to bind the DOM to instance data.

Models are plain JavaScript objects which update the view when data is changed. Additional tools provide facilities for scaffolding, routing, state management, animations and more.

**Pros:**

* rapid adoption and increasing popularity
* simple to get started with high level of developer satisfaction
* small dependency and good performance

**Cons:**

* a newer project – risks may be greater
* some reliance on a single developer for updates
* fewer resources than alternatives

### **Backbone.js**

| [**Backbone.js**](http://backbone.js/) | |
| --- | --- |
| type | framework |
| website | [backbonejs.org](http://backbonejs.org/) |
| repository | [github.com/jashkenas/backbone/](https://github.com/jashkenas/backbone/) |
| current version | 1.3.3 |
| developer | Jeremy Ashkenas |
| launch date | October 2010 |
| typical size | 8kb min |
| typical use | single-page applications |
| usage | [low](https://w3techs.com/technologies/details/js-backbone/all/all) |

Backbone.js was one of the earliest client-side options to provide an MVC structure commonly found in server-side frameworks. Its only dependency is Underscore.js which was created by the same developer.

Backbone.js claims to be a library because it can be integrated with other projects. I suspect most developers consider it to be a framework, albeit less opinionated than some others.

Pros:

* small, lightweight and less complicated
* does not add logic to HTML
* great documentation
* adopted for many applications including Trello, WordPress.com, LinkedIn and Groupon

Cons:

* a lower level of abstraction than alternatives such as AngularJS (although that could be considered a benefit)
* requires additional components to implement features such as data binding
* more recent frameworks have moved away from MVC architectures

### **Gulp.js**

| [**Gulp.js**](http://gulp.js/) | |
| --- | --- |
| website | [gulpjs.com](http://gulpjs.com/) |
| repository | [github.com/gulpjs/gulp](https://github.com/gulpjs/gulp) |
| current version | 3.9.1 |
| monthly downloads | 3 million |

While it was not the first task runner, Gulp quickly became the most popular and is [my personal favorite](https://www.sitepoint.com/introduction-gulp-js/?aref=cbuckler). Gulp uses easy-to-read JavaScript code which loads source files into a stream and pipes the data through various plugins before they are output to a build folder. It’s simple, fast and fun – examine Gulp.js before any other option.

### **NPM**

| **npm** | |
| --- | --- |
| website | [npmjs.com](https://www.npmjs.com/) |
| repository | [github.com/npm/npm](https://github.com/npm/npm) |
| current version | 4.5.0 |
| monthly downloads | 3 million |

npm is the Node.js package manager but its scripts facility can be used for [general-purpose task running](https://www.sitepoint.com/guide-to-npm-as-a-build-tool/?aref=cbuckler). It’s an attractive option for simpler projects with few dependencies. However, more complex tasks can rapidly become impractical.

### **Grunt**

| **Grunt** | |
| --- | --- |
| website | [gruntjs.com](https://gruntjs.com/) |
| repository | [github.com/gruntjs/grunt](https://github.com/gruntjs/grunt) |
| monthly downloads | 2 million |

Grunt was one of the first JavaScript task runners to achieve mass adoption but the speed and complicated JSON configuration led to the rise of Gulp. The worst issues have now been resolved and Grunt remains a popular option.

**Tools: Module Bundlers**

Managing more than a few JavaScript files quickly becomes a chore. By default, browser files are not compiled so dependencies must be loaded or concatenated in an appropriate order. There are various options such as [ES6 Modules](https://www.sitepoint.com/understanding-es6-modules/?aref=cbuckler) and [CommonJS](http://wiki.commonjs.org/wiki/Modules) but browser support is limited so a module bundler becomes essential.

### **Webpack**

| **Webpack** | |
| --- | --- |
| website | [webpack.js.org](https://webpack.js.org/) |
| repository | [github.com/webpack/webpack](https://github.com/webpack/webpack) |
| current version | 2.5.1 |
| monthly downloads | 6 million |

Webpack supports all popular module options and has become synonymous with React development. Although claiming to be a module bundler, Webpack can be used as a general-purpose task runner.

### **Browserify**

| **Browserify** | |
| --- | --- |
| website | [browserify.org](http://browserify.org/) |
| repository | [github.com/substack/node-browserify](https://github.com/substack/node-browserify) |
| current version | 14.3.0 |
| monthly downloads | 2.6 million |

Browserify supports CommonJS modules as used by Node.js to compile all modules into a single browser-compatible file.

**Reference:**

* Craig Buckler (April 10, 2016) Top JavaScript frameworks Retrieved from <https://www.sitepoint.com/top-javascript-frameworks-libraries-tools-use/>
* Sviatoslav A.(October 17, 2015) Best front end JavaScript frameworks Retrieved from <https://rubygarage.org/blog/best-javascript-frameworks-for-front-end>
* Margarita Shilova(August 24, 2017) JS framework you should know in 2017Retrieved from <https://apiumhub.com/tech-blog-barcelona/top-javascript-frameworks/>
* Noeticsutil (August 02, 2017) Retrieved from http://noeticforce.com/best-Javascript-frameworks-for-single-page-modern-web-applications