

Quizard.js

It's cool!

DOCs

*["Solving problems is not about the language; it is about seeing
patterns and knowing how to apply them to solve complex problems.]"*
-Mr. Oluwatosin .M. Adesanya

Version 0.2.4

This page is intentionally left blank.

Contents

Introduction.....	4
Overview	5
Guide for users	6
Guide for developers	8
Folders in the dev-version.....	8
Core Files	9
Conclusion	10
Thanks to	10
About me.....	10
Find me on	10

Introduction

The need for solutions to reoccurring problems drives some individuals to develop frameworks or solutions to this recurring pattern. They tend to come up with all kinds of technical tools in other to prove that their methods are worthy. However, I am not playing the role of a technical evangelist neither am I a devil's advocate for any technology or practice. What I want we to understand here is that any technology can be use to produce solutions provided we could see these recurring themes and apply them to solve simple to complex problems.

-Damian Simon Peter

Overview

The file extension beside the name Quizard may have got you thinking this is yet another JavaScript framework for something other than manipulating the DOM. Well, if you thought of that then you are definitely right.

Quizard.js is both an application and a framework depending on the perspective from which you see it. It is built with vanilla JavaScript.

I built this application from the ground up without sourcing Google for solutions after deciding that it was time to stop reading and start coding. I tend to develop and release subsequent versions overtime and your contribution is highly welcomed.

From a user perspective, Quizard.js is just a quiz application. It could be a tool for practicing questions for exams, simulating e-exams, and just for whiling away time. However, this is far from the truth, as we would see in future sections of this documentation.

From a developer's perspective, Quizard.js is a framework, which can be extended or reconfigured to your own taste and style. The pattern I adhere to when building this application is what I like to call ***“the mind your own business”*** pattern. This technique is formally called ***“separation of concerns”*** or ***unobtrusive JavaScript***.

While building Quizard.js I adhere to the unobtrusive JavaScript principle but not to graceful degradation or feature detection so you will not find any form of JavaScript in my HTML or vice versa. Each technology minds their own business. In fact, this application works well on Firefox and Chrome. For IE and Safari, since I did not follow the pattern of feature detection some stuffs may not work since I mainly used the ``addEventListener`` method which in older versions of IE is called ``attachEvent``.

Therefore, I believe we all have chrome, or better still Firefox as Quizard.js works well on both of them.

Quizard.js

Guide for users

The best way to get started if you just want to use the Quizard.js app is to download the minified version from [here](#). After downloading, your folder structure would look like this:

Name	Date modified
js	10/1/2015 9:07 AM
lib	9/28/2015 9:54 AM
template	10/1/2015 2:50 AM
index	9/30/2015 5:21 PM
package	9/28/2015 6:38 PM

Open up `index.html` page in your web browser either Firefox or Chrome. Create an account and start using the app.

After playing around the app, you will find out that the questions are fixed. This is not a problem as Quizard.js enables you to add more questions to the application. To do this, open up the `package.json` file, the one closer to the `index.html` here you will find the list of questions as [JSON](#) string, carefully observe the structure and add your own question by duplicating any questions and change their values.

```
*C:\xampp\htdocs\quizard\package.json - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
PrimeChecker.js new 3 new 7 new 8 api.php app.js ApiController.php tbl_users.json wp-config.php api.php new 9 scripts.js package.json
1 [
2   {"question":"Living organism that dwell high above the ground, especially on trees are
3   called?","options":["Acquatic Habitat","Aboreal Habitat","Terrestrial Habitat"],"answer":1},
4   {"question":"Isaac Newton is a man famous for the discovery of?","options":["Radioactivity","Law of
5   floatation","Laws of Motion"],"answer":2},
6   {"question":"Billa Jean, was a song by?","options":["Micheal Jackson","Chris Brown","Elvis Presly"],"answer":0},
7   {"question":"Who among these men is Renaissance?","options":["Leonardo Da Vinci","Vasco Da Gama","Leonardo
8   Capucino","Vasco Ancelloti"],"answer":0},
9   {"question":"How many surface dose a cube has?","options":["16","36","4","none of the options"],"answer":3},
10  {"question":"Cancer (the crab) is a zodiac sign for anyone born
11  on?","options":["January","July","June","September"],"answer":1},
12  {"question":"The name Steve Job is synonymous with which of
13  these?","options":["Microsoft","Android","Apple","Google"],"answer":2}
14  ]
15
16
```

If I were to duplicate the question and add a new one, I will enter a new question as follow:

```
{"question":"Newton's first law of motion, is also regarded as the law of? ","options":["Relativity","Cause & Effect","Karma","Inertia"],"answer":3}
```

In the above JSON string, you can see the question, its options and the correct answer but why is the answer 3? Just count from the first item in your options starting at 0 which is the first item and 3 which is the last item and the correct answer. In computer programming when dealing with certain data structures, counting starts at 0, so when you do a literal count of 6, in computer programming (when working with the array data structure) you have an index from 0 through 5.

After adding as much question as desired, save and refresh the application. You will find out that your questions are now available. The questions do not show in any particular order, rather, they pop out at random this is due to the algorithm I created for reshuffling questions after a page refresh or reload.










User wise the method for adding questions is not encouraged, I will fix this in the next release of the application. However, ensure you type the above JSON string in just one line.

When opening the`package.json` file ensure you open in with a plain text editor and not Microsoft Word. Notepad or Notepad++ is fine.

Guide for developers

Now for the geeks, the best way to hack Quizard.js is to download the [dev-version](#) or git-clone into your local directory. The dev-version is heavily documented, and it contains documentation for all APIs used in the framework. Quizard.js also has an [online demo](#) and [online documentation](#).

After git-cloning or downloading, you file structure should look like this:

Name	Date modified
 .git	10/2/2015 1:21 AM
 doc	10/2/2015 1:21 AM
 js	10/1/2015 11:26 PM
 lib	10/1/2015 11:29 PM
 template	10/1/2015 11:28 PM
 index	10/1/2015 7:02 PM
 LICENSE	10/1/2015 11:02 PM
 package	10/1/2015 4:38 PM
 README	10/1/2015 11:02 PM

For the APIs docs, open the docs folder and run the HTML pages in your web browser. Quizard.js docs were automagically (*my term*) generated by JSDOC-toolkit.

Folders in the dev-version

- 1. doc:** This folder contains Quizard.js API docs. The API docs are just plain html pages you can run on your browser.
- 2. js:** This folder contains Quizard.js core scripts which are `config.js`, `dom-core.js`, `FetchAjax.js`, `globals.js`, `main.js`, `PostAjax.js`, `user-manager.js` and `utility.js`
- 3. lib:** This folder contains dependent library not for Quizard.js functionality, but for its User Interface. The absence of this library does not affect the functionality of Quizard.js, there are only present for UI-sake.
- 4. template:** This folder contains another folder called `js` and three (3) HTML files `dashboard.html`, `quiz.html` and

`register.html` respectively. Each of this file have their corresponding counterpart in the `js` folder of this directory with `login.js` being an exception but it corresponds to the `index.html` page outside this directory.

5. **index.html:** This is not a folder just the entrance to Quizard.js
6. **package.json:** This is not a folder just a JSON file containing the quizzes that will be displayed when you are logged in.

As a developer, the `package.json` file should not be your data source. Instead, consider building a backend service say in PHP/MYSQL and make Quizard.js a consumer of your question service. To do this is very easy with Quizard.js All you need do is open your `config.js` file and change the value in the constant labeled `QUIZ_SOURCE` to point to your question service URL but ensure you adhere to the data structure format in the `package.json` file.

Example

```
const QUIZ_SOURCE = "http://www.mydomain.com/api/questions-service";
```

This service should return JSON string in the exact format as seen in `package.json`

Core Files

All scripts in the root `js` folder are Quizard.js core scripts and their API docs should be read carefully before editing them, the dev-version is heavily commented so the docs is just a repetition of the comment.

Scripts in **template/js** extend scripts in the core. Here you will appreciate the beauty of APIs and use them to your benefits.

Conclusion

Conclusion

In my own view, Quizard.js is a fantastic framework and application but can become better with feedbacks and criticism. Feel free to mail me damiansimonpeter@gmail.com

Thanks to

- God
- Mum & Dad
- Mr. Tosin
- Sirolad
- Sunday Goodman
- Quincy Larson
- @andela
- @freecodecamp
- @bincomICT
- MiMi
- All class 12 boot campers
- You

About me

Programmer, Entrepreneur and Public Speaker. Loves cooking, music, Greek mythology and motivational literature.

Find me on

[Stackoverflow](#)

[Twitter](#)

[GitHub](#)

Thanks for reading.