Instructions

This test consists of a number of tasks that you will need to complete using the ezytest web application that we will provide to you. Once you have finished as many of the tasks as you can then zip up the project and email it back to us for review.

This is not an exam so the test is not marked to produce a total score but rather is subjectively assessed to provide us with an indication of your problem solving abilities.

The order that you complete the tasks is up to you. If there is a task that you cannot complete it should not impact your ability to proceed with subsequent or prior tasks.

You may or may not have had prior experience with some of the tasks. Attempt what you can and be prepared to answer questions later as to how you arrived at your solution and the steps you took to attempt/complete each task.

Just as important to us is the process that you undertook to solve each task than just seeing the completed task itself. Therefore if you cannot complete a task don't be discouraged because if you provide us with sufficient information on where you looked at and what you tried, due credit will be given.

You can supply supplementary information about each task either as comments in the code or as a separate file you add to the project. We don't mind how, as long as you make it clear to us where to look to find this information.

There is one section of the test unrelated to the web project but as mentioned above you can provide the answers in a file you add to the project or in an email.

The tasks can be completed using the tools and techniques of your choice unless explicitly stated otherwise.

Each task might have more than one possible solution. Unless specified otherwise you get to choose which way you solve the task but be prepared to answer questions about your approach.

Prerequisites

We will be testing your solutions to each task using the latest stable release of the following desktop browsers so you will need to have access to a PC with them installed:

- Chrome (v26 at time of writing)
- Firefox (v20 at time of writing)

One of the tasks does involve demonstrating your familiarity with source control management. If you choose to complete the task then you will need to have installed Git http://git-scm.com/ in your development environment.

You will need a web server in your development environment to host the ezytest web application. We will not be opening the index.html and testing your solution from the local file system.

EzyTest Application

The ezytest project is a simple web application that has been put together with some of the libraries and techniques commonly used by ezyinsights. You will be modifying this project to complete most of the tasks.

It is our expectation that you will need to start from the websites of these libraries to be able to become familiar with them:

- Knockout.js http://knockoutjs.com/
- jQuery http://jquery.com/
- Require.js http://requirejs.org/

You will also be responsible for extending the project to also include:

- LESS http://lesscss.org/
- and a javascript unit testing framework of your choice

The application consists of only one html file but several additional javascript files. The structure is loosely based on the example given by http://knockoutjs.com/documentation/amd-loading.html which shows using Knockout.js with the javascript module loader Require.js

The application allows you to maintain a list of names with the ability to create, edit or remove names from the view model. There is a short video (http://bit.ly/OhyTa3) that we recommend you watch to understand the functionality of the application with all the tasks complete.

Tasks

Complete as many as the following tasks as you can and when you have finished send us back the ezytest web application in a zip file within the time period specified. If you completed the SCM tasks make sure you include the Git repository in the zip.

SCM

- Create a local Git repository for the ezytest project and commit your work in progress after each task to the repository.
- Create a branch called "feature xyz" and create the following two files (leave them empty) tobeignored.txt and scmskills.txt. Add them to the branch and commit. Now remove the file tobeignored.txt from the branch and configure git to ignore files called tobeignored.txt in your repository. Commit these changes to the branch and merge the changes back into the HEAD.

Comment [1]: Paul Wistrand:
Use following to see tree:
git log --graph --oneline --all

CSS/HTML

- Implement the following CSS styling for the project using LESS in an external stylesheet file and alter the index.html to apply the style.

○ Typeface:	Helvetica, Arial, Sans-Serif
○ Font size:	15px
○ Font color:	#494949
○ Link color:	#DF3701
○ Link hover:	underline

- In chrome the cursor for the link anchor tag <a/> is not the hand cursor when the mouse hovers over the link. Make the necessary corrections to fix this.
- The number of names in the view model is currently displayed before the add button. Alter the HTML/CSS for this element to ensure that it is displayed in the top right hand corner of the screen and is always displayed even when forced to scroll the page to see the last name.

Unit Testing

- Using a javascript unit testing framework of your choice implement a test to ensure that the count property of the appViewModel is incremented when a new name is added to the names observable array. Incorporate the testing framework runner into the project.

JQuery/Knockout.js

- The following code snippet in init.js will never fire. What is the reason why?

```
$('.editBtn').on('click', function() {
        //TODO: why doesn't this event handler fire?
        alert('triggered this event handler');
});
```

- The remove function in appViewModel.js has a bug. Fix the bug and explain the reason for the bug.

- Alter the appViewModel.js so that it exposes an add function. Alter the addBtn event handler so that it invokes the add function on the view model.

- The editableRow custom binding handler in customHandlers.js needs to correctly implement the cancel behaviour for the cancel button. At the moment it does not revert the name back to its original state prior to the edit. Fix this bug.

- Why don’t we need any implementation for the update function of the editableRow custom binding handler and could we remove it?

- Alter the editableRow custom binding handler so that it is assigned using an IIFE. Is there any value in doing so in this example of the editableRow custom binding handler?

- Markup the appViewModel.js file with code comments using the JSDoc syntax.

Optimization

- Optimise the javascript files in the project by minification. Update the index.html and any unit test files to use the minified files.

Miscellaneous

- What are the benefits of a javascript script loader?

- What are the benefits of separating javascript that encapsulates a data model or controller from javascript that interacts with the DOM?

- What are the benefits of namespacing?