Mon Feb 17 19:03:08 EST 2020

Testing All Formats

test1.mdc

React

// hi

React Overview

Virtual DOM <- 0

* Reads
* Here is a super mega long one just to see what happens ..... la la la
  // how will this comment do ??? hmmm i
  wonder .... is this long enough?
* What if I do this
  // lovely
* Writes only if needed // this is a must

JSX <- 1

sefi er **I'm invalid** s--c cs

- Compiles into JavaScript

Happy

- Extends JavaScript* Creates React elements

Has <- 2

- name <- 3
  - child tags <- 4
    realated stuff

    like these <- 5

react.mdc

React Overview

Virtual DOM

* Reads
* Writes only if needed

JSX

- Compiles into JavaScript
- Extends JavaScript* Creates React elements

Has

- name
- attributes
- child tags

Isomorphic rendering

- renders components on client and server

Notes:

JavaScript-driven MVCs

- ie Angular, Backbone, Ember, they all render on DOM load and
  this can be bery lag
- Aren't indexable by search engines unless you are willing to pay

for a third party service
- These issues can be solved by rendering JavaScript on the server side
(aka. isomorphic rendering)

Unidirectional data flow
in two-way data binding, the view gets updated when the state changes and
the state changes when the view changes

Angular JS
ie. when you change a model in Angular JS, the view automatically reflects those changes
In addition, an input field in the view can cause the model to be mutated

Now in a well designed app, admittedly, this does work.
But it can cause cascading updates and changing a model could fire off even more event updates

Flux
Flux takes a functional approach with unidirectional data flow
The view is a function of the application state,
So when the application state changes the view automatically re-renders itself.

To put in another way, the same state resultes in the same view
and this leads to better predictability

Mutation of data is done via actions.

React and Flux
Now if you're going to use React and Flux you have to start thinking about rethinking best
practice.

So for example, we're going to be seeing JavaScript and markup in the same file.
Sometimes it just makes sense to avoid the separation and there are benefits.

With React, HTML is a projection of state

React
- Is a view layer
  - It's the 'V' in MVC
  - Well-built React components conver 'C' as well

- Has no opinion on
  - Routing
  - Data Flow
  - Buid & Deployment

Possible Stacks
1.

Node.js	server-side JavaScript
Browserify	to expose Node packages to the browser

React Components for our components
2.

React Router	for routing through pages
Facebook's Flux	for data flow
Glup	task runner to put everything together for deployment

Node JS

What is it?
* Server Side JavaScript
* Non-blocking I/O
* Uses Google V8 high performance engine
means that it is very light weight and efficient
* Provides package manager npm
  * thousands of packages avaliable
  * Easy maintenance and installation

FYI
* Node.j Foundation maintains the largest open source library in the world

NPM
  * Install and manage dependencies for an app
    - Like NuGet for .Net
    - PIP for Python

  * Quick and easy to use
  * Use NPM to obtain and maintain all out JS libraries

CommonJS Pattern
  - Node.js uses CommonJS for specifying modules
  - This mean creating a usable module form other modules like syncrinous modules like ATM
  - first we specify any modules to use by using require statement

  Specifying modules
  3 Parts:
    1. Require statement
    2. Module declaration
    3. Export module
       To expose out modules to other modules so they can reference the new module that we have created

Broswerify

Bundles up Node.js NPM modules
  - So we can use them in the browser
  - Easy managing and bundling dependencies

NPM has thousands of packages
  - bundles our npm packages up in one file
  - using browserify gives you acess to all of them in the broswer
    So does it download the modules to they are on the client side???
  - Organize our code
  - Browser can use all our packages
  - Uses same module system as Node.js
    So packages published to NPM that may have originally written for deployment in node, not for the browser, will work in the browser by using browserify

Because browserify bundles up all the dependencies into one file, it gives us a performance benefit because it reduces the HTTP requests to get out JavaScript modules, it’s going to lead to a faster loading application

react2.mdc

---------- React Notes ----------

React
  - Utilizes virtual DOM for unparalleled performance
  - DOM in HTML/XML
  - Virtual DOM as JS object (faster)

  When DOM updates, compares the old state and new state of DOM and updates in the least expensive way

Isomorphic rendering
  can render on the client or server side
  enabled by the virutal DOM

  - react doesn't read from the real DOM
  - it only writes to the real DOM if necessary

Flux
  Avoids traditional MVC pattern
    - problems occur when you have complex interactions with models and views

  Uses one way data flow
    - makes app logic easier to reason about

- avoids the complexities inherent in MVC
- avoids interactions between multiple views and models
- view is a function of the spplication state
- better predictability

Flux Data Flow

.. --> [Action] -> [Dispatcher] -> [store] -> [react view] -----> [Action] -> ..

^Isn't that the MVC????

Actions
- methods that enables the passing of data to dispatchers
- action creator creates actions
- view gets updated when action is fired

Dispatcher
- callback registary
- list of all the stores it will have to send actions to
- dispatchers collects actions and broadcast the actions and their data (payload)
  to all registered callbacks

Store
- Containers for application state and application logic
- The only entry to the store is a callback that's registered to the dispatcher
- once the store makes changes to state, it emites a change event and notifies the controller
  view that the state has changed

React Views
- react components that takes state from stores and transmit them through props to child
  components
- views takes state and renders it
- also accepts user input

test.mdc

Heading

Funtionality
* This is a second level bullet

Level 2 Heading
Level 2 Body
Level 2 Body / Level 3 Heading
Level 3 Body

Level 2 Body

- This is a dashed bullet
  - This is a nested circle bullet 1
    - This is a nested circle bullet 2
      * Nested bullets should not have colour

- This is an arrow section
-> Arrows must precedence with space\tabs and end with a space
-> "Quotes should be fine"

New Words

*GTM	Global Trade Management System	// This should not bullet
EODc	End of the Day Calculations	<- This is hard to do
	wrapping of text	
	very special and difficult	
	do later	

Notes:

```
    * Make sure to review everything

  Notes:
    * This is part of the new words body

  Beautiful Table
    Type A
                Col1        Col 2                   Col "3"
    ---------------------------------------------------------------------
                A           To Be ...               False      <- harder to do
                            Or not be                          <- smart tabing

                A           True

    Notes:
      Tables must have at least 2 columns
      All leaves do not have colouring

    Type B
                    Col1       |    Col2           Col3
    ---------------------------------------------------------------------
                      A              True            False
                      A              True            False
```

Sub Heading

https://www.google.ca

[lala land] https://www.google.ca

```
  This is a table
  [lala] https://www.google.ca
/*
  Notes for me ....
*/
```

```
  python -m pip install django
```

---

test3.mdc

React Overview

Virtual DOM

* Reads
* Writes only if needed

JSX

- Compiles into JavaScript
- Extends JavaScript* Creates React elements

Has

- name
  - child tags
    realated stuff

    like these
    ...
- id
  >>> blah

Isomorphic rendering

- renders components on client and server <- reseach this

Notes:

JavaScript-driven MVCs

```
/*
  - ie Angular, Backbone, Ember, they all render on DOM load and
                    this can be bery lag
*/
  - Aren't indexable by search engines unless you are willing to pay
                    for a third party service
```

installation

```
  npm install
```

tests.mdc

Heading

Funtionality

* This is a second level bullet

Level 2 Heading

Level 2 Body

Level 2 Body / Level 3 Heading

Level 3 Body

Level 2 Body

Type B

Col1	Col2	Col3
A	True	False
A	True	False

Level 2 Heading B <- This works

```
*GTM    Global Trade Management System          // This should not bullet
EODc    End of the Day Calculations             <- This is hard to do
        wrapping of text
        very special and difficult
        do later
```

React Overview

Virtual DOM

* Reads
* Writes only if needed

JSX

- Compiles into JavaScript
- Extends JavaScript* Creates React elements

Has

- name
  - child tags
    realated stuff

    like these
    ...
- id
  >>> blah

Isomorphic rendering

- renders components on client and server <- reseach this

Notes:

JavaScript-driven MVCs

```
  /*
    - ie Angular, Backbone, Ember, they all render on DOM load and
                        this can be bery lag
  */
    - Aren't indexable by search engines unless you are willing to pay
                        for a third party service
```

installation

```
  ```
    npm install
  ```
```