REVIEW of Front-End Web Development with React

6/23/2019 Rui Li

Bootstrap-帮助建立UI框架

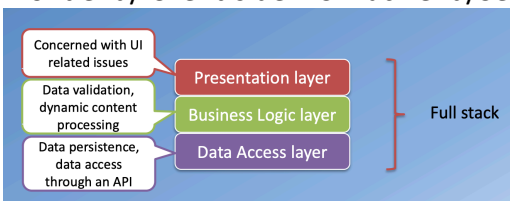
React解构调整框架，同时在框架中加入数据

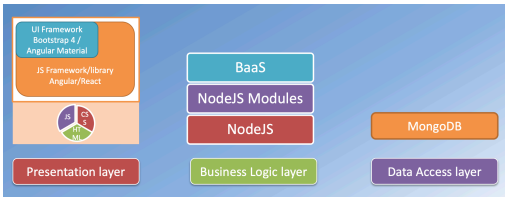
产生漩涡

Week1:

1. Introduction to Full Stack Web Development

Front end/ Client-side VS Back end/Server-side





2. Setting up your Development Environment: Git and Node

Git: Version Control 把所有写的程序版本都保存起来

NodeJS: JavaScript runtime （翻译软件，翻译 JS）可以让 JS 这种前端语言控制 server-side

NPM: 网上资源库的一个集合站，帮助从网上下载资源包

3. Introduction to React

Frameworks: 插 code 进去 [FLUX/REDUX]

Libraries: 调用现有的成熟的 behavior [React]

React: element, component, virtual DOM

JSX: extension to JavaScript, kind of a combination of JS & html in our case

4. React Components

Component: split UI into independent, reusable pieces

State: components used to store its own local information passed as PROPS

-declared in constructor

-use setState() to change state

Lifecycle: Mounting, updating, unmounting

Week2:

1. React component Types

Presentational: concerned with rendering view [do not maintain their own local state]

Container: data fetching & state updates [communicate with data sources]

Class component: extend react component/render()/can have local state

Functional component: simplest/receives props as param/no local state or lifecycle/return()

2. React Router

Virtual Dom: created completely from scratch on every setState [React Object/ light weighted/tree data structure/fast/use diffing algorithm to update DOM] state 改变和 DOM 改变的中间一个处理层

Router: collection of navigational components [use URL to enable navigation among views]

Link vs NavLink: NavLink 感觉是用来做链接的，导航栏里用 NavLink

3. Single Page Application

Single Page Application: web application or web site fits in a single page [No need to reload entire page] 一次性下载好大部分数据，后面和 server 只需要传递部分信息

React Router Parameters: [建立子网页 a list] 子网页的定义 <Link to{`/menu/${dish.id}`>

Path=”/menu/:dishId

Week3:

1. Controlled Forms

Form: entry (e.g.: login/ submit information)

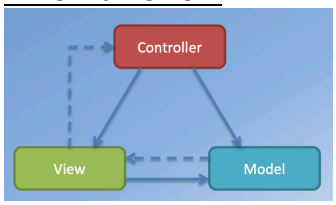
Controlled Forms: react component control the form (single source of truth/state tying with component state/**every state mutation have an associated handler function**)

2. Uncontrolled Forms

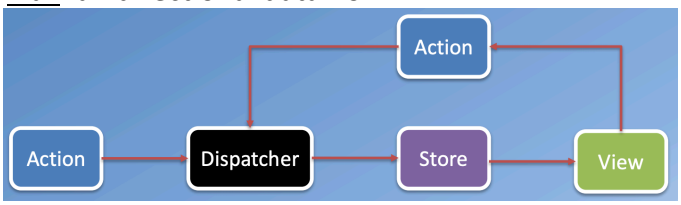
Uncontrolled Forms: easier, use a ref to get form values from DOM, instead of writing handler function (innerRef={(input)=>this.username.input})

3. Introduction to Redux

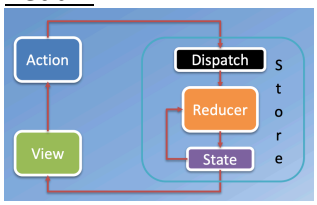
MVC Framework:



Flux: unidirectional data flow



Redux:



Redux: Framework/take previous state and action and return next state

4. React Redux Form

LocalForm: Map form model to local state of the component (when no need to form data persistence across component mounting/unmounting) (LocalForm/control.text)

Week4:

1. Redux Actions

Redux Action: type+payload

Actions: payloads of information (send data from your application to the store through dispatch())

Action Creators: Functions that create actions

Reducer: take previous state and action and return next state (splitting & combining redusers)

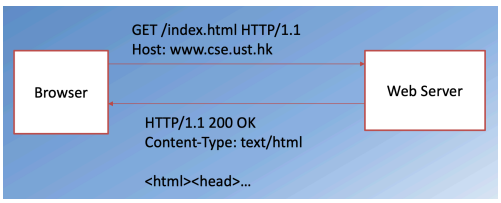
2. Redux Thunk

Redux Middleware: provides capability to run code after an action is dispatched, but before it reaches the reducer

Thunk: a subroutine used to inject an additional calculation in another subroutine

Redux Thunk: return **function** instead of an action **in action creators**, inner function receives dispatch() and getState() store methods

3. Client-Server Communication [Write applications recognizing the unexpected delays]



HTTP (Hypertext Transfer Protocol): a client-server communications protocol [response in XML or JSON]

JSON (JavaScript Object Notation): lightweight data interchange format

Web Services: a system designed to support interoperability of systems connected over a network (SOAP & REST)

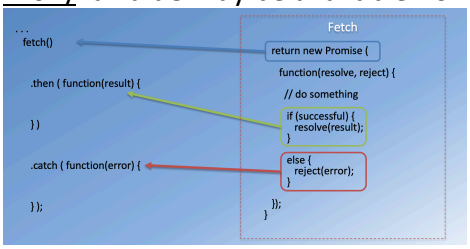
REST (Representational State Transfer): a style of software architecture for distributed hypermedia systems, outline how resources are defined and addressed

-HTTP methods explicitly
-stateless
-expose directory structure-like URIs
-transfer using XML/JSON

4. Fetch

Promise: Mechanism that supports asynchronous computation [先调用方法，但允许执行不成功]

Proxy: a value may be available now, or in the future, or never



Fetch Usage

```
fetch(baseUrl + 'dishes')
    .then(response => response.json())
    .then(data => console.log(data))
    .catch(error => console.log(error.message));
```

Fetch: interface for fetching resources, promise based (modern replacement for XMLHttpRequest())

5. React Animations