react

# The Complete Web Developer in 2019 Zero to Mastery

## Javascript basic:

Resource:

<https://github.com/zero-to-mastery/complete-web-developer-manual>

<https://github.com/zero-to-mastery/zero-to-mastery-website/>

Start from js

Javascript type:

Có 6 kiểu bao gồm:

1. Number
2. String
3. Boolean
4. Undefined
5. Null
6. Symbol( new in ECMAScript 6)
7. Object

Ở màn hình console muốn clear thì gọi hàm clear(), toán tử % dùng để chia lấy dư

Đối với String có thể sử dụng dấu nháy đơn hay nháy kép

Ex: ‘It\’s me!’

Toán tử trong js

5 + "34" // "534"

5 - "4" // 1

10 % 5 // 0

5 % 10 // 5

"Java" + "Script" // "JavaScript"

" " + " " // " "

" " + 0 // " 0"

true + true // 2

true + false // 1

false + true // 1

false - true // -1

3 - 4 // -1

"Bob" - "bill" // NAN

5 >= 1 // true

0 === 1 // false

4 <= 1 // false

1 != 1 // false

"A" > "B" // false

"B" < "C" // true

"a" > "A" // true

"b" < "A" // false

true === false // false

true != true // false

// Make the string: "HI There! It's sunny out" by using the + sign.

"Hi There! " + "It\'s \"sunny\" out"

Biến

Var x = 123;

Hàm prompt(‘What is your name?’); để hiển thị hộp thoại nhập tên vào

// Make a Calculator! using prompt(), and variables, make a program that does the following:

var firstNumber = prompt("Please provide the first number");

var secondNumber = prompt("Please provide the second number");

alert(Number(firstNumber) + Number(secondNumber));

Nếu biến chưa được khai báo mà sử dụng sẽ có kiểu là undefined

Control follow

if (Number(age) < 18) {

alert("Sorry, you are too yound to drive this car. Powering off");

} else if (Number(age) > 18) {

alert("Powering On. Enjoy the ride!");

} else if (Number(age) === 18) {

alert("Congratulations on your first year of driving. Enjoy the ride!");

}

Để in ra console dùng lệnh : console.log() ;

Sừ dụng toán tử 3 ngôi hoặc switch

Nhúng file js vào file html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<link rel="stylesheet" type="text/css" href="">

</head>

<body>

<script type="text/javascript" src="abc.js"></script>

</body>

</html>

Hàm

Cú pháp:

function checkDriverAge1() {

}

var checkDriverAge2 = function() {

}

Data structure

Array

*// using this array,*

var array = ["Banana", "Apples", "Oranges", "Blueberries"];

*// 1. Remove the Banana from the array.*

array.shift(); # pop() lấy pt cuối

*// 2. Sort the array in order.*

array.sort();

*// 3. Put "Kiwi" at the end of the array.*

array.push("Kiwi");

*// 4. Remove "Apples" from the array.*

array.splice(0, 1);

*// 5. Sort the array in reverse order.*

array.reverse();

array.concat([‘beer’]);

*// using this array,*

*// var array2 = ["Banana", ["Apples", ["Oranges"], "Blueberries"]];*

*// access "Oranges".*

array2[1][1][0];

Object

Loop

Có đầy đủ for, while, do while, và có thêm cả forEach() (new in ECMAScript 5)

array.forEach(function(value) {

console.log(value);

});

For(item of arr)

For (item in arr)

const basket = ['apples', 'oranges', 'grapes'];

const detailedBasket = {

apples: 5,

oranges: 10,

grapes: 1000

}

*//1*

for (let i = 0; i < basket.length; i++) {

console.log(basket[i]);

}

*//2*

basket.forEach(item => {

console.log(item);

})

for (item in detailedBasket) {

console.log(item);

}

for (item of basket) {

console.log(item);

}

*// Question #1:*

*// create a function called biggestNumberInArray() that takes*

*// an array as a parameter and returns the biggest number.*

*// biggestNumberInArray([-1,0,3,100, 99, 2, 99]) should return 100;*

*// Use at least 3 different types of javascript loops to write this:*

const array = [-1,0,3,100, 99, 2, 99] *// should return 100*

const array2 = ['a', 3, 4, 2] *// should return 4*

const array3 = [] *// should return 0*

function biggestNumberInArray(arr) {

let highest = 0;

for (let i = 0; i < arr.length; i++) {

if (highest < arr[i]) {

highest = arr[i];

}

}

return highest

}

function biggestNumberInArray2(arr) {

let highest = 0;

arr.forEach(item => {

if (highest < item) {

highest = item;

}

})

return highest;

}

function biggestNumberInArray3(arr) {

let highest = 0;

for (item of arr) {

if (highest < item) {

highest = item;

}

}

return highest;

}

biggestNumberInArray3(array3)

*// Question #2:*

*// Write a function checkBasket() that lets you know if the item is in the basket or not*

amazonBasket = {

glasses: 1,

books: 2,

floss: 100

}

function checkBasket(basket, lookingFor) {

for (item in basket) {

if (item === lookingFor) {

return `${lookingFor} is in your basket`

}

}

return 'that does not exist in your basket'

}

Call stack: FILO, đơn luồng dễ xử lý k deadlock

Scope

Advanced array

<https://sdras.github.io/array-explorer/>

babel: <https://babeljs.io/repl/>

*//Create an array using forEach that has all the usernames with a "!" to each of the usernames*

let newArray = []

array.forEach(user => {

    let { username } = user;

    username = username + "!";

    newArray.push(username);

})

console.log(newArray);

*//Create an array using map that has all the usernames with a "?" at the end of usernames*

const mapArray = array.map(user => {

    let { username } = user;

    return username + "?";

})

console.log(mapArray);

*//Filter the array to only include users who are on team: red*

const filterArray = array.filter(user => {

    return user.team === "red";

})

console.log(filterArray);

*//Find out the total score of all users using reduce*

const total = array.reduce((acc, user) => {

    return acc + user.score

}, 0);

console.log(total);

*// (1), what is the value of i? INDEX of the array.*

*// (2), Make this map function pure:*

const arrayNum = [1, 2, 4, 5, 8, 9];

const newArray = arrayNum.map((num, i) => {

    return num \* 2;

})

*//BONUS: create a new list with all user information, but add "!" to the end of each items they own.*

const answer = array.map(user => {

    user.items = user.items.map(item => {

        return item + "!"

    });

    return user;

})

console.log(answer);

Advanced object

Var obj1 = {value: 10};

Var obj2 = obj1 ; // === true

Var obj3 = {value: 10};

Var obj3 === obj1 ; // === false

Từ khóa this

<https://www.w3schools.com/js/js_classes.asp>

class Car {

constructor(brand) {

this.carname = brand;

}

present() {

return 'I have a ' + this.carname;

}

}

class Model extends Car {

constructor(brand, mod) {

super(brand);

this.model = mod;

}

show() {

return this.present() + ', it is a ' + this.model;

}

}

mycar = new Model("Ford", "Mustang");

pass by value and references

Đối với object, array mới truyền tham chiếu, khi thay đổi value => affect

Const clone = Object.assign({}, obj);

Objclone2 = {…obj}

// shallow clone first level tạo bản sao nên không ảnh hưởng khi thay đổi giá trị

// deep clone

Let superclone = JSON.parse(JSON.stringify(obj));

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Equality_comparisons_and_sameness>

<https://www.ecma-international.org/ecma-262/5.1/#sec-11.9.3>

<https://dorey.github.io/JavaScript-Equality-Table/>

ES7

‘hello’.includes(‘o’); // true có thể AD cho arrays

*// #1) Check if this array includes the name "John".*

const dragons = ['Tim', 'Johnathan', 'Sandy', 'Sarah'];

dragons.includes('John') *// false*

*// #2) Check if this array includes any name that has "John" inside of it. If it does, return that*

*// name or names in an array.*

const dragons = ['Tim', 'Johnathan', 'Sandy', 'Sarah'];

dragons.filter(name => name.includes('John')) *// ['Johnathan']*

*// #3) Create a function that calulates the power of 100 of a number entered as a parameter*

const power100 = (num) => num\*\*100;

*// #4) Useing your function from #3, put in the paramter 10000. What is the result?*

*// Research for yourself why you get this result.*

power100(10000) *// Infinity*

ES8

*// #1) Line up the Turtle and the Rabbit at the start line:*

const startLine = ' ||<- Start line';

let turtle = '🐢';

let rabbit = '🐇';

*// it should look like this:*

' ||<- Start line'

' 🐢'

' 🐇'

*// when you do:*

console.log(startLine);

console.log(turtle);

console.log(rabbit);

turtle = turtle.padStart(8);

rabbit = rabbit.padStart(8);

*// #2) What happens when you run turtle.trim().padEnd(9, '=') on the turtle variable*

*// Read about what the second parameter does in padEnd and padStart*

turtle = turtle.trim().padEnd(9, '=');

' ||<- Start line'

'🐢======='

' 🐇'

*// #3) Get the below object to go from:*

let obj = {

my: 'name',

is: 'Rudolf',

the: 'raindeer'

}

*// to this:*

'my name is Rudolf the raindeer'

Object.entries(obj).map(value => value.join(" ")).join(' ')

Object.values

Object.keys

Kết thúc truyền tham số trong hàm để lại dấu , ở cuối để tránh lỗi khi truyền k đủ tham số

Modules

<https://medium.com/sungthecoder/javascript-module-module-loader-module-bundler-es6-module-confused-yet-6343510e7bde>

<https://hacks.mozilla.org/2018/03/es-modules-a-cartoon-deep-dive/>

<https://github.com/getify/You-Dont-Know-JS>

<http://javascript.info/>

<http://dmitrysoshnikov.com/ecmascript/javascript-the-core-2nd-edition/>

xem lại

## DOM manipulation

DOM is document object model

<https://www.w3schools.com/js/js_htmldom.asp>

document.write();

document.getElementById(‘a’);

window

DOM Selectors

--------------

getElementsByTagName

getElementsByClassName

getElementById

querySelector

querySelectorAll

getAttribute

setAttribute

##Changing Styles

style.{property} //ok

className //best

classList //best

classList.add

classList.remove

classList.toggle

##Bonus

innerHTML //DANGEROUS

parentElement

children

##It is important to CACHE selectors in variables

document.querySelector('li').getAttribute('id');

lấy ra thẻ li đầu tiên có id là 12

document.querySelector('li').setAttribute('id', ‘123’);

có thể thay đổi thuộc tính trong class

document.querySelector('li').style.background = ….

document.querySelector('li').className

document.querySelector('li').classList.add(‘cool’);

document.querySelectorAll('li')[1].parentElement.children;

<https://developer.mozilla.org/en-US/docs/Web/Events>

<https://www.cambiaresearch.com/articles/15/javascript-char-codes-key-codes>

tham khảo: 6.1 script.js.js

jQuery

<http://youmightnotneedjquery.com/>

ES5 và ES6

Babel: convert ES6 => change for all browser

Let sử dụng cho scope nhỏ, trong khi var sẽ lướn hơn rất nhìu

Const dừng cho hằng số

Đối với Object thì thay đổi thuộc tính nó sẽ thay đổi dù sử dụng từ khóa nào

*// Destructuring*

const person = {

firstName : "John",

lastName : "Doe",

age : 50,

eyeColor : "blue"

};

const { firstName, lastName, age, eyeColor } = person;

*// Object properties*

const a = 'test';

const b = true;

const c = 789;

const okObj = {a, b, c};

*// Template strings*

const message = `Hello ${firstName} have I met you before? I think we met in ${city}. last summer no???`;

*// default arguments*

*// default age to 10;*

const isValidAge = (age = 10) => age;

*// Symbol*

*// Create a symbol: "This is my first Symbol"*

const sym = Symbol('This is my first Symbol');

*// Arrow functions*

const whereAmI = (username, location) => {

if (username && location) {

return "I am not lost";

} else {

return "I am totally lost!";

}

}

var name = "ddd";

const obj = {

[name]: "hello",

[1 + 2]: "three"

};

Advanced func

*//Solve these problems:*

*//#1 Create a one line function that adds adds two parameters*

const sum = (a, b) => a + b

*//Closure: What does the last line return?*

const addTo = x => y => x + y

var addToTen = addTo(10)

addToTen(3) *// returns 13*

*//Currying: What does the last line return?*

const sum = (a, b) => a + b

const curriedSum = (a) => (b) => a + b

curriedSum(30)(1) *// 31*

*//Currying: What does the last line return?*

const sum = (a, b) => a + b

const curriedSum = (a) => (b) => a + b

const add5 = curriedSum(5)

add5(12) *// 17*

*//Composing: What does the last line return?*

const compose = (f, g) => (a) => f(g(a));

const add1 = (num) => num + 1;

const add5 = (num) => num + 5;

compose(add1, add5)(10) *// 16*

*//What are the two elements of a pure function?*

1. Deterministic *-->* always produces the same results given the same inputs

2. No Side Effects *-->* It does not depend on any state, or data, change during a program’s execution. It must only depend on its input elements.

## Command line

It's time to get comfortable with the terminal! Open up the terminal or the command prompt on your computer and give these commands a try:  
  
**FOR MAC OR LINUX:**

ls

Pwd

Cd

Cd ..

Clear

Cd / —> root director

Cd ~

Cd folder/folder/test

Mkdir name

Open folder

Touch index.html

Open index.html

Open -a “Sublime Text”

Open .

Mv index.html about.html

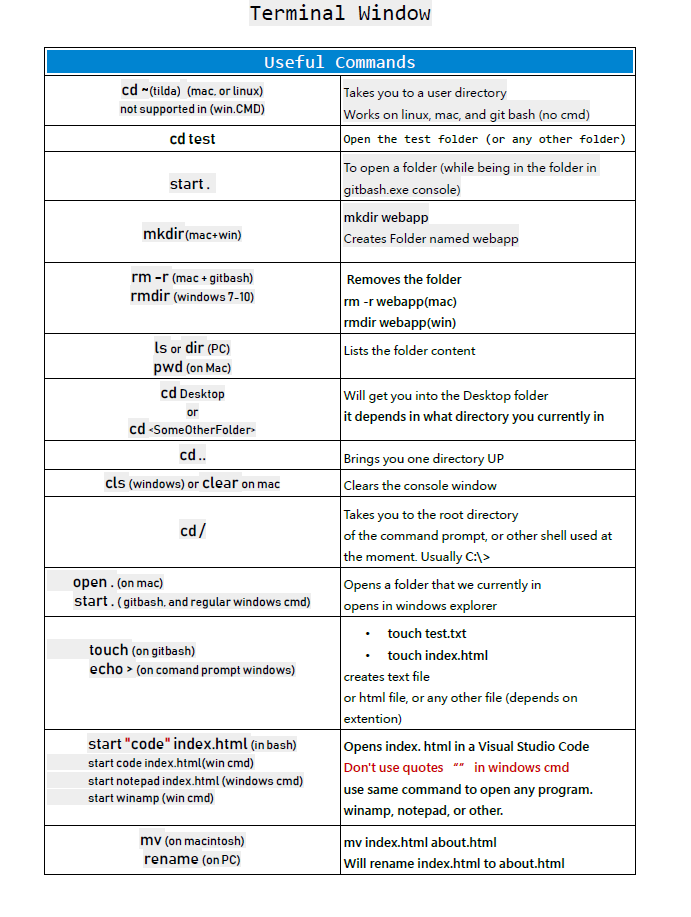
Up and down arrow.

Rm file

Rm -r folder

Say hello (only on Mac).

**FOR WINDOWS:**



dir - list files

cd {directory name} - change directory

cd / - go to root (top) directory

cd .. - go up a level

mkdir {file name} - make a directory

echo > {filename} - create an empty file

del {filename} - remove a file

rmdir {directory name} - remove a directory and all files within

rename {filename} {new filename} - rename a file or folder

start {filename} - open file in default program

start . - open current directory

cls - clear the terminal screen

## IDE

Sublime text

Here are my recommended resources to get your Sublime Text personalized:  
  
1.  [Material Theme](https://github.com/equinusocio/material-theme)  
2.  [Oceanic Next Colour Scheme](https://github.com/voronianski/oceanic-next-color-scheme)  
3.  [Package Control](https://packagecontrol.io/)  
4.  [A great article going over sublime text 3 setup](https://scotch.io/bar-talk/best-of-sublime-text-3-features-plugins-and-settings)  
  
My recommended packages to install (command + shift +p):

Babel  
SidebarEnhancements  
A File Icon  
GitGutter  
BracketHighlighter  
  
Other options other than Sublime Text 3 (Both are free):  
  
[Atom](https://atom.io/)  
[Visual Studio Code](https://code.visualstudio.com/)

Vs code

Install:

Quokka – gợi ý

Identicator - cách vào

Git

Tải về:

Git clone <http://....git>

Git status

Git add .

Git commit -a -m “msg”

Git push origin master

Git pull

Clear

Git branch // show all branch

Git branch new\_branch // create

Git checkout new\_branch // chuyển nhánh

Git merge master

A fellow student **@antonykidis** created this great diagram and cheat sheet for the community to help you with all of the steps we followed in the previous videos. Have a look:   
  
[Part 1](https://github.com/antonykidis/GitHub-guide/blob/master/Git%20and%20GitHub.pdf)  
[Part 2](https://github.com/antonykidis/GitHub-guide/blob/master/Git%20and%20GitHub2.pdf)  
[Part 3](https://github.com/antonykidis/GitHub-guide/blob/master/GitHubGuidePart3.pdf)

Install node

npm -v

node -v

npm init

***IMPORTANT: In this course, we need to be using Node version 8.9.1 or higher, and NPM*** ***version 5.5.1 or higher!!!!***  
  
If you are having issues installing NPM and Node.js use the below resources to guide you:

[Installing NPM instructions](https://www.npmjs.com/get-npm)  
  
[Node.js install website](https://nodejs.org/en/download/)

To check if you have Node.js installed, run this command in your terminal:

node -v

To confirm that you have npm installed you can run this command in your terminal:

npm -v

If you get the versions number back, everything is working!

While using npm or node commands, if you ever get permission issue, you may need to run the commands with "sudo" in front of each command. For more details, [review this](https://docs.npmjs.com/getting-started/fixing-npm-permissions). And for windows: you may have to [use another command](https://stackoverflow.com/questions/9652720/how-to-run-sudo-command-in-windows)

**Optional:**You can use a tool like nvm to be able to install multiple versions of node and change between them for each project. This is advanced so delve into it at your own risk: [NVM Tutorial](https://www.sitepoint.com/quick-tip-multiple-versions-node-nvm/)

Install lodash

Npm install lodash

Install live-server

Tự động cập nhật nếu có thay đổi trên server

Npm install -g live-server

Sudo npm install -g browerify // use require syntax

Browerify a.js > bundle.js // update and understand npm

Tại file package.json cập nhật scripts để gõ lệnh nhanh

## React & redux

Divide into small components => reuse

One way data flow: chỉ có con mới biết sự thay đổi, parent sẽ không quan tâm đến vì hướng dữ liệu là downstream

Virtual DOM: đưa obj cho react nó sẽ tự động thay đổi DOM và tạo nên website

Material design

<https://mdbootstrap.com/docs/react/>

Git app robo

<https://github.com/aneagoie/robofriends>

Create react app

<https://github.com/facebook/create-react-app>

Đầu tiên install create-react-app

Npm install -g create-react-app

Nếu có lỗi thêm sudo ở đầu

Clear

create-react-app robofriends

cd robofriends

npm start

Sau đó tìm hiểu cấu trúc source

Nếu có phiên bản mới thì cập nhật bằng cách vào file package.json và npm install

<https://github.com/nerdjfpb/robofriends>

Tạo ra các component ở file .js

import React from 'react';

import logo from './logo.svg';

import './App.css';

function App() {

return (

<div className="App">

<header className="App-header">

<img src={logo} className="App-logo" alt="logo" />

<p>

Edit <code>src/App.js</code> and save to reload.

</p>

<a

className="App-link"

href="https://reactjs.org"

target="\_blank"

rel="noopener noreferrer"

>

Learn React

</a>

</header>

</div>

);

}

export default App;

Install tachyons

Thư viện hỗ trợ design web thay vì làm css

Có thể định nghĩa các className bằng những gì được hỗ trợ(giống như bs4)

Npm install tachyons

<https://tachyons.io/docs/>

Khi tạo ứng dụng đầu tiên ta có thể tạo file Hello.js có 2 cách tạo:

import React, { Component } from "react";

import "./App.css";

import "tachyons"; // import vào để xài

*// class Hello extends Component {*

*// render() {*

*// return (*

*// <div className="tc i f1">*

*// <p>*

*// Hello msg <code>{this.props.gretting}</code>*

*// </p>*

*// </div>*

*// );*

*// }*

*// }*

const Hello = props => {

return (

<div className="tc i f1">

<p>

Hello msg <code>{props.gretting}</code>

</p>

</div>

);

};

// có hàm thì bỏ từ khóa this đi vì nó k phải là 1 obj

Tại file index.js sửa thành:

ReactDOM.render(

<Hello greeting={"hello" + "my friends"} />,

document.getElementById("root")

);

Hình

<https://robohash.org/>

<https://robohash.org/>test?200x200

chỉ cần thêm đuôi bất kì sẽ có được hình khác nhau

Tạo file card.js rồi chỉnh sửa lại index .js

Card.js

import React from "react";

import "./Card.css";

import "tachyons";

const Card = () => {

return (

<div className="bg-light-green dib br3 pa3 ma2 grow bw2 shadow-5">

<img alt="robot" src="https://robohash.org/test?200x200" />

<div>

<h2>Jane Doe</h2>

<p>Janedoe@gmail.com</p>

</div>

</div>

);

};

export default Card;

index.js

ReactDOM.render(

<div>

<Card />

<Card />

<Card />

</div>,

document.getElementById("root")

);

Tạo file robot.js để import data vào

import {robots} from "./robots";

Vì nếu có export nhiều biến khác ta có thể thêm dấu phẩy và import thêm vào

Cập nhật lại index.js

ReactDOM.render(

<div>

<Card id={robots[0].id} name={robots[0].name} email={robots[0].email} />

<Card id={robots[1].id} name={robots[1].name} email={robots[1].email} />

<Card id={robots[2].id} name={robots[2].name} email={robots[2].email} />

</div>,

document.getElementById("root")

);

Card.js

const Card = props => {

const {name, email} = props;

return (

<div className="tc bg-light-green dib br3 pa3 ma2 grow bw2 shadow-5">

<img alt="robot" src={`https://robohash.org/${props.id}?200x200`} />

<div>

<h2>{name}</h2>

<p>{email}</p>

</div>

</div>

);

};

// sửa lại gọn hơn

const Card = ({name, email}) => {

tiếp tục chỉnh sửa tạo file CardList.js

import React from "react";

import Card from "./Card";

import "tachyons";

const CardList = ({ robots }) => {

*// const cardComponent = robots.map((user, i) => {*

*// return (*

*// <Card*

*// key={i}*

*// id={robots[i].id}*

*// name={robots[i].name}*

*// email={robots[i].email}*

*// />*

*// );*

*// });*

return (

<div>

{robots.map((user, i) => {

return (

<Card

key={i}

id={robots[i].id}

name={robots[i].name}

email={robots[i].email}

/>

);

})}

</div>

);

};

export default CardList;

ReactDOM.render(<CardList robots={robots} />, document.getElementById("root"));

Tại phần 3 chỉnh sửa file App.js để chứa CardList

Thêm SearchBox

Tìm hiểu về state mô tả app những gì thay đổi trong app, nó chỉ là 1 obj , props never change

import React, { Component } from "react";

import "./App.css";

import "tachyons";

import CardList from "./CardList";

import { robots } from "./robots";

import SearchBox from "./SearchBox";

class App extends Component {

constructor() {

super();

this.state = {

robots: robots,

searchfield: ""

};

}

onSearchChange = event => {

*// console.log(event);*

*// console.log(event.target.value);*

this.setState({ searchfield: event.target.value });

};

render() {

const filtersRobot = this.state.robots.filter(robot => {

return robot.name

.toLowerCase()

.includes(this.state.searchfield.toLowerCase());

});

console.log(filtersRobot);

return (

<div className="tc">

<h1>Robot friends</h1>

<SearchBox searchChange={this.onSearchChange} />

<CardList robots={filtersRobot} />

{

*// sau khi cập nhật state*

*// <CardList robots={this.state.robots} />*

}

</div>

);

}

}

export default App;

<https://www.w3schools.com/jsref/jsref_includes.asp>

<https://www.w3schools.com/jsref/jsref_tolowercase.asp>

<https://reactjs.org/docs/handling-events.html>

search sega gg font css <https://www.cufonfonts.com/font/sega-logo-font>

sau đó tải file zip về có css để copy vào

<https://css-tricks.com/snippets/css/using-font-face/>

Phần 4 Tạo data giả để app thật hơn:

<https://jsonplaceholder.typicode.com/users>

Life cycle:

<https://reactjs.org/docs/react-component.html>

lấy data từ API về

<https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API>

Phần 5: khi scroll thì mất ô search vì vậy cần tạo tag <Scroll>

Phẩn 6: tổ chức source

Npm run build // để build app

ErrorBoundary

Fix lỗi ErrorBoundary in react 16: tạo file ErrorBoundary trong components

Bọc lại CardList nếu có lỗi nó sẽ bắt lại và hiển thị ra app

Sate management

Khi app càng lớn thì có càng nhiều stage mỗi khi thay đổi => affect => redux keep it in store

3 nguyên tắc đối với redux:

* Single source of truth(1 obj lớn)
* State is read only
* Changes using pure func

Action -> reducer -> store -> make changes

**Flux pattern**

Action -> dispatcher -> store -> view

Redux replace state in react

Redux

npm install redux

npm install react-redux

<https://github.com/aneagoie/robofriends-redux>

20. Redux Actions and Reducers: Tạo file actions.js và constants.js, reducers.js

21. Redux Store and Provider: vào file index.js để import

import { createStore } from "redux";

import { Provider } from "react-redux";

import { searchRobots } from "./reducers";

const store = createStore(searchRobots);

ReactDOM.render(<App store={store}/>, document.getElementById("root"));

// vào file App.js ghi log ở componentDidMount để xem

Update:

const store = createStore(searchRobots);

ReactDOM.render(

<Provider store={store}>

<App />

</Provider>,

document.getElementById("root")

);

*// step 6 end*

22. Redux connect()

Vào file app.js thêm :

// parameter state comes from index.js provider store state(rootReducers)

const mapStateToProps = (state) => {

return {

searchField: state.searchRobots.searchField

}

}

// dispatch the DOM changes to call an action. note mapStateToProps returns object, mapDispatchToProps returns function

// the function returns an object then uses connect to change the data from redecers.

const mapDispatchToProps = (dispatch) => {

return {

onSearchChange: (event) => dispatch(setSearchField(event.target.value))

}

}

// action done from mapDispatchToProps will channge state from mapStateToProps

export default connect(

mapStateToProps,

mapDispatchToProps

)(App);

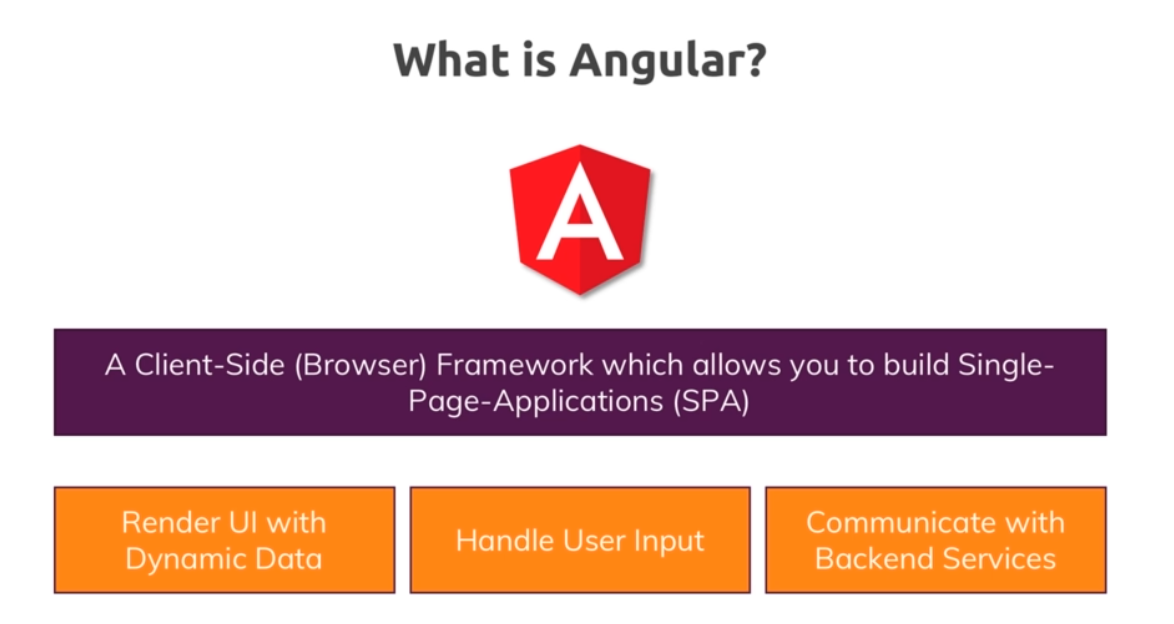
Sau đó sửa trong hàm render

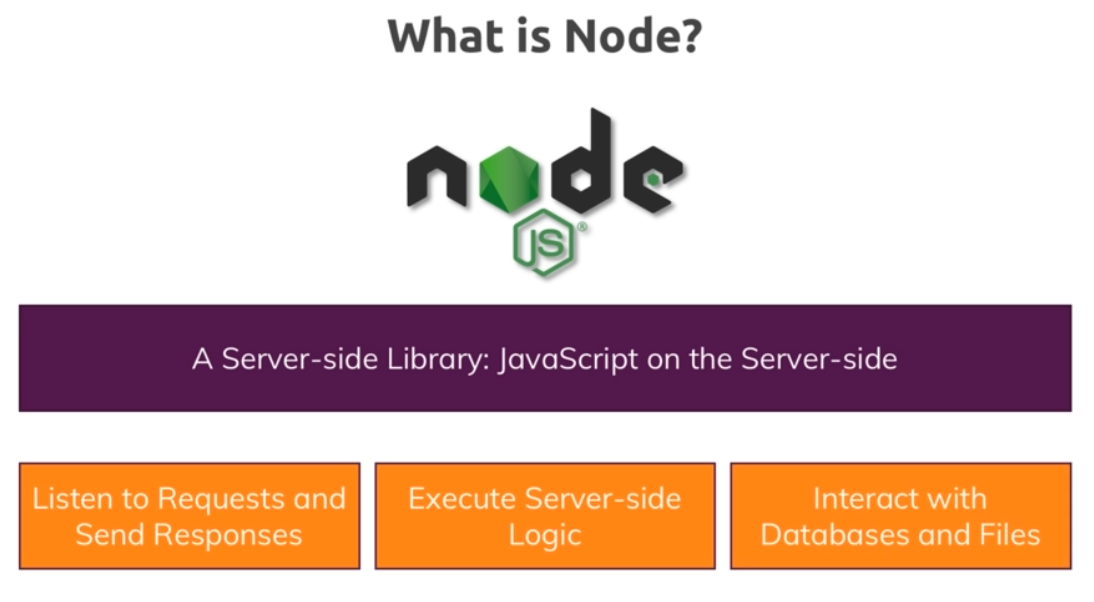
Middleware

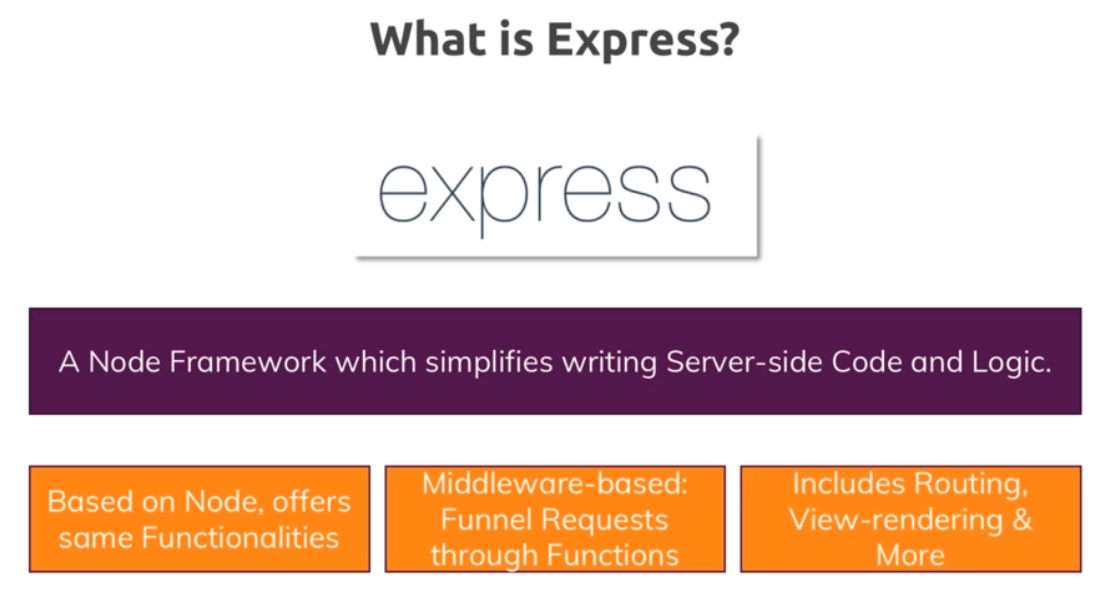
Npm install redux-logger

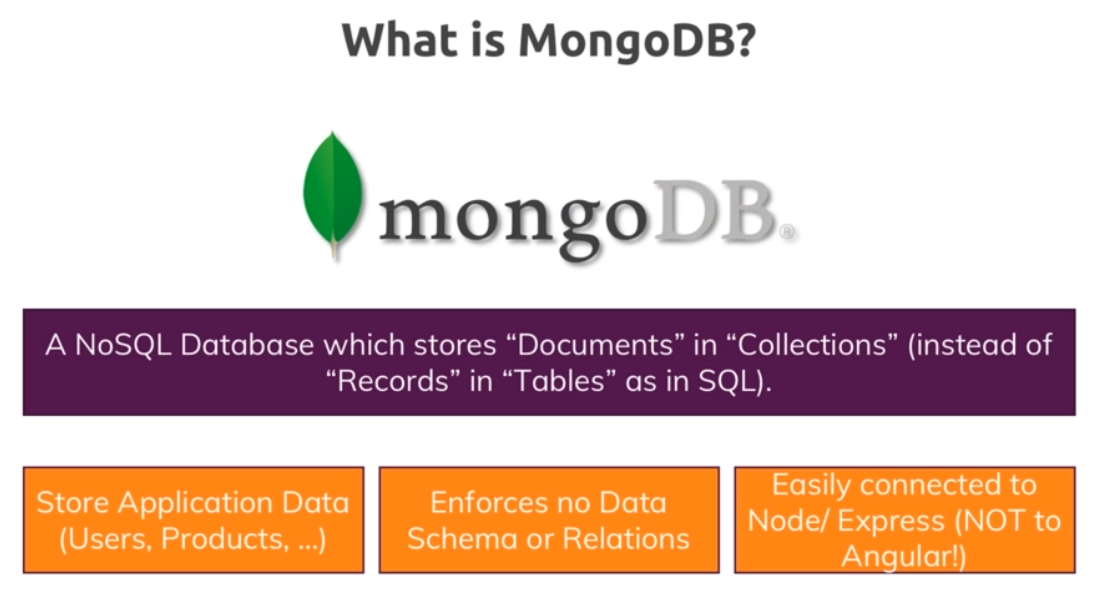
# The angular & node js – MEAN Stack guide

MEAN là Mongodb, express, angular, node js









Single page application: no reload page, only remove and add new some element

Angular tutorial

<https://www.javatpoint.com/angularjs-tutorial> <https://www.tutorialspoint.com/angular6/index.htm>

Install nodejs 10+

Install angular cli

npm install -g @angular/cli

ng new mean-course

cd mean-course

ng serve

clear

ng -v

## Extensions

Cài đặt:

* Angular essentials
* Material icon theme

Cấu trúc source: CTRL U

<https://freetuts.net/thiet-lap-project-voi-angular-6-1718.html>

File index.html

Chạy file main.ts trước nó sẽ import app.module.ts, trong này NgModule khai báo component

Add first component

Tạo post-create component

import {Component} from '@angular/core';

// đánh dấu là component

@Component({

selector: 'app-post-create',

templateUrl: './post-create.component.html'

})

export class PostCreateComponent {

}

File module khai báo:

@NgModule({

declarations: [

AppComponent,

PostCreateComponent

],

imports: [

BrowserModule,

AppRoutingModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

Sau khi khai báo và file **app.component.html** thêm thẻ vừa mới định nghĩa vào: <app-post-create></…>

Link

* Learn everything about Angular: <https://academind.com/learn/angular>
* Angular Material Tutorial: <https://academind.com/learn/angular/angular-material-a-thorough-guide/>
* Angular Material Docs: <https://material.angular.io/>
* Reference vs Primitive Types in JS: <https://academind.com/learn/javascript/reference-vs-primitive-values/>
* RxJS Tutorial: <https://academind.com/learn/javascript/understanding-rxjs/>

## The Angular Frontend - Understanding the Basics

Event:

File html: bắt event bằng sự kiện trong dấu (), dấu [] để thuộc tính

<h2 style="background: yellow">Post create component</h2>

<textarea rows="6"></textarea>

<hr />

<button (click)="onAddPost()">save post</button>

Event: ở file angular:

Khai báo hàm onAddPost ở trong class ở file angular:

// step 2 start

onAddPost(postInput: HTMLTextAreaElement) {

alert("event button click");

// this.newPost = "the user post";

}

// step 2 end

Output content: SD {{ }} để in biến ra ngoài(nhớ khai báo biến trong class trước, ở trong file ts truy cập thông qua từ khóa this và ở file html thì không cần)

Binding prop:

SD [value]=’”test”’ // gán là chuỗi test

[value]=’newPost’: nếu muốn tham chiếu đến tt của class

Get user input: thêm postInput vào đây và truyền vào hàm

<h2 style="background: yellow">Post create component</h2>

<textarea rows="6" [value]="newPost" #postInput></textarea>

<hr />

<button (click)="onAddPost(postInput)">save post</button>

<p>{{ newPost }}</p>

Tại file .ts:

onAddPost(postInput: HTMLTextAreaElement) {

*// xem nội dung user input*

*// console.dir(postInput);*

*// this.newPost = postInput.value;*

}

Cách 2 get user input: two way binding

Vào file app.module.ts:

import { FormsModule } from "@angular/forms";

rồi cập nhật lại:

imports: [BrowserModule, AppRoutingModule, FormsModule],

chỉnh lại file html:

*<!-- step 3 start -->*

<textarea rows="6" [(ngModel)]="enteredContent"></textarea>

<hr />

<button (click)="onAddPost()">save post</button>

<p>{{ newPost }}</p>

*<!-- step 3 end -->*

File angular:

enteredContent = "";

newPost = "no content";

*// step 3 start*

onAddPost() {

*// xem nội dung user input cách 2*

this.newPost = this.enteredContent;

}

*// step 3 end*

Installing Angular Material

npm install –save @angular/material // cũ k xài

từ angular cli 6+ nó sẽ tự động thêm 1 số config

ng add @angular/material

nó sẽ tự động có:

import { BrowserAnimationsModule } from "@angular/platform-browser/animations";

* Angular Material Docs: <https://material.angular.io/>

File angular.json có chứa style chỉ định theme mình đang xài

Import thêm:

import { BrowserModule } from "@angular/platform-browser";

import { NgModule } from "@angular/core";

import { FormsModule } from "@angular/forms";

import {

MatInputModule,

MatCardModule,

MatButtonModule,

MatToolbarModule,

MatExpansionModule

} from "@angular/material";

import { AppRoutingModule } from "./app-routing.module";

import { AppComponent } from "./app.component";

import { PostCreateComponent } from "./posts/post-create/post-create.component";

import { BrowserAnimationsModule } from "@angular/platform-browser/animations";

@NgModule({

declarations: [AppComponent, PostCreateComponent],

imports: [

BrowserModule,

AppRoutingModule,

FormsModule,

BrowserAnimationsModule,

MatInputModule,

MatCardModule,

MatButtonModule,

MatToolbarModule,

MatExpansionModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule {}

// để file html có thể sửa dụng

Tạo file post-create.css

mat-form-field,

textarea {

width: 100%;

}

Rồi cập nhật lại ở file .ts

@Component({

selector: "app-post-create",

templateUrl: "./post-create.component.html",

styleUrls: ["./post-create.component.css"]

})

*<!-- step 4 start -->*

<mat-card>

<mat-form-field>

<textarea matInput rows="6" [(ngModel)]="enteredContent"></textarea>

</mat-form-field>

<button mat-raised-button color="accent" (click)="onAddPost()">

Save Post

</button>

<p>{{ newPost }}</p>

</mat-card>

*<!-- step 4 end -->*

Add tool bar(source 1.zip)

Tạo folder header

Outputing post

Tạo folder post-list và khai báo

Sử dụng Extension Panel

Import thêm MatExpansionModule

Khai báo modules:

@NgModule({

declarations: [

AppComponent,

PostCreateComponent,

HeaderComponent,

PostListComponent

],

Có thể chỉnh thêm css cho đẹp trong app.component.css

ở file css trong post-list thêm:

:host {

display: block;

margin-top: 1rem;

}

// thẻ đặc biệt đại diện là chính nó

.info-text {

text-align: center;

}

// class

Tại class html để đổ data post:

<mat-accordion multi="true" \*ngIf="posts.length > 0">

<mat-expansion-panel \*ngFor="let post of posts">

<mat-expansion-panel-header>

{{ post.title }}

</mat-expansion-panel-header>

<p>{{ post.content }}</p>

</mat-expansion-panel>

</mat-accordion>

<p class="info-text mat-body-1" \*ngIf="posts.length <= 0">No posts added yet!</p>

Nếu multi=”true” thì có thể expand nhiều cái 1 lúc

Post-list.component.ts thêm:

import { Component, Input } from "@angular/core";

@Component({

selector: "app-post-list",

templateUrl: "./post-list.component.html",

styleUrls: ["./post-list.component.css"]

})

export class PostListComponent {

posts = [

{ title: "First Post", content: "This is the first post's content" },

{ title: "Second Post", content: "This is the second post's content" },

{ title: "Third Post", content: "This is the third post's content" }

];

*// @Input() posts = [];*

}

11. Diving Into Structural Directives: đổ data ra

Vào guide trong material chọn using angular typography để thêm các class cần thiết nhanh vào thẻ

<p class="info-text mat-body-1" \*ngIf="posts.length <= 0">No posts added yet!</p>

Create post with properties and event binding

Chap 12

Sửa file post-create.component.html và post-create.component.ts:

Import EventEmitter để emit các event

import { Component, EventEmitter, Output } from "@angular/core";

@Component({

selector: "app-post-create",

templateUrl: "./post-create.component.html",

styleUrls: ["./post-create.component.css"]

})

export class PostCreateComponent {

enteredTitle = "";

enteredContent = "";

@Output() postCreated = new EventEmitter(); // khởi tạo postCreated is an event

onAddPost() {

const post = {

title: this.enteredTitle,

content: this.enteredContent

};

this.postCreated.emit(post);

}

}

<mat-card>

<mat-form-field>

<input matInput type="text" [(ngModel)]="enteredTitle">

</mat-form-field>

<mat-form-field>

<textarea matInput rows="6" [(ngModel)]="enteredContent"></textarea>

</mat-form-field>

<button

mat-raised-button

color="accent"

(click)="onAddPost()">Save Post</button>

</mat-card>

Sau đó vào file app.component.html, app.component.ts thêm để get và store lại post data

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

storedPosts = [];

onPostAdded(post) {

this.storedPosts.push(post);

}

}

Listen to event

<app-header></app-header>

<main>

<app-post-create (postCreated)="onPostAdded($event)"></app-post-create>

<app-post-list [posts]="storedPosts"></app-post-list>

</main>

Tại file post-list.component.ts sửa : @Input() posts = []; // binding

Create a post model

Chap 13

Tạo file posts/post-model.ts:

export interface Post {

title: string;

content: string;

}

Tại file app.component.ts

export class AppComponent {

storedPosts: Post[] = [];

onPostAdded(post) {

this.storedPosts.push(post);

}

}

Ở các file sử dụng post như post-list hay post-create.ts:

@Output() postCreated = new EventEmitter<Post>();

...

const post: Post = {

title: form.value.title,

content: form.value.content

};

@Input() posts: Post[] = [];

File app.ts

postCreated: Post[] = [];

Adding form

Thêm thẻ form vào file post-create.html, xóa binding thuộc tính chỉ để lại ngModel

Khi xài thẻ form nhớ khai báo thêm FormsModule

<mat-card>

<form (submit)="onAddPost(postForm)" #postForm="ngForm">

<mat-form-field>

<input

matInput

type="text"

name="title"

ngModel

required

minlength="3"

#title="ngModel">

<mat-error \*ngIf="title.invalid">Please enter a post title.</mat-error>

</mat-form-field>

<mat-form-field>

<textarea

matInput

rows="4"

name="content"

ngModel

required

#content="ngModel"></textarea>

<mat-error \*ngIf="content.invalid">Please enter a post title.</mat-error>

</mat-form-field>

<button

mat-raised-button

color="accent"

type="submit">Save Post</button>

</form>

</mat-card>

postForm cho phép chúng ta access tới angular form object

**Xử lý lỗi:** Thêm required, minlength để validate

Có thể ghi *<!-- <mat-error \*ngIf="postForm.getControl('title').invalid">Please enter a post title.</mat-error> --> thay vì* \*ngIf="title.invalid", #title="ngModel"

#postForm="ngForm" giúp ta access obj and manage

File angular

import { Component, EventEmitter, Output } from "@angular/core";

import { NgForm } from "@angular/forms";

import { Post } from '../post.model';

@Component({

selector: "app-post-create",

templateUrl: "./post-create.component.html",

styleUrls: ["./post-create.component.css"]

})

export class PostCreateComponent {

enteredTitle = "";

enteredContent = "";

@Output() postCreated = new EventEmitter<Post>();

onAddPost(form: NgForm) {

if (form.invalid) {

return;

}

const post: Post = {

title: form.value.title,

content: form.value.content

};

this.postCreated.emit(post);

}

}

Lấy dữ liệu ra bằng cách gọi form.value.title theo name đã đặt ở file html

File angular-02-added-post-form đã đủ

15. Getting Posts from Post-Create to Post-List

Khi ứng dụng ngày càng lớn không thể pas data như vậy => service: 1 cách để truy cập data từ trong component khác mà k cần property and event binding

Chúng ta chỉ cần get service này vào component và nó sẽ được thực hiện bởi tính năng gọi là dependence injection, bạn chỉ cần đi đến component sử dụng chúng và thêm constructor

Tạo file posts.service.ts

* Có class, có hàm addPost, getPosts, biến post

Tại file post-list.component.ts

constructor(public postsService: PostsService) {}

// nghĩa là nó sẽ tự động tạo tt mới trong component này và store giá trị đầu vào trong thuộc tính này

Cách 1 : Vào file service posts.service.ts thêm Injectable tức là nó sẽ tạo ra 1 instance trong suốt app:

import { Injectable } from '@angular/core';

import { Subject } from 'rxjs';

import { Post } from './post.model';

@Injectable({providedIn: 'root'})

export class PostsService {

private posts: Post[] = [];

private postsUpdated = new Subject<Post[]>();

getPosts() {

return [...this.posts]; //copy chứ k cần ref tới obj thật sự

tạo array mới và add them into new array

}

getPostUpdateListener() {

return this.postsUpdated.asObservable();

}

addPost(title: string, content: string) {

const post: Post = {title: title, content: content};

this.posts.push(post);

this.postsUpdated.next([...this.posts]);

}

}

Cách 2 : có thể khai báo vào file module trong đó để nó nhận biết service thông qua »  providers: [],

16. Calling GET Post

Tại file post-list.component.ts sử dụng life cycle trong angular implement OnInit

export class PostListComponent implements OnInit, OnDestroy {

posts: Post[] = [];

constructor(public postsService: PostsService) {}

ngOnInit() {

this.posts = this.postsService.getPosts();

}

ngOnDestroy() {

}

}

Tại file post-create.component.ts thêm constructor và sửa lại hàm:

export class PostCreateComponent {

enteredTitle = "";

enteredContent = "";

constructor(public postsService: PostsService) {}

onAddPost(form: NgForm) {

if (form.invalid) {

return;

}

this.postsService.addPost(form.value.title, form.value.content);

form.resetForm();

}

Sau đó vào file app.component.html sửa:

<app-header></app-header>

<main>

<app-post-create></app-post-create>

<app-post-list></app-post-list>

</main>

File app.component.ts:

import { Component } from '@angular/core';

@Component({

selector: 'app-root',

templateUrl: './app.component.html',

styleUrls: ['./app.component.css']

})

export class AppComponent {

}

Tiếp đó sử dụng thư viện rxjs ở file post.service.ts để emit event khi add a post: this is a copy of my posts after I updated them

import { Subject } from 'rxjs';

thêm thuộc tính

private postsUpdated = new Subject<Post[]>();

addPost(title: string, content: string) {

const post: Post = {title: title, content: content};

this.posts.push(post);

this.postsUpdated.next([...this.posts]); // add new

}

// hàm này để lắng nghe sự kiện

getPostUpdateListener() {

return this.postsUpdated.asObservable();

}

Tại file post-list.component.ts sử dụng Subscription to remove Subcription and prevent memory leak:

import { Component, OnInit, OnDestroy } from "@angular/core";

import { Subscription } from "rxjs";

import { Post } from "../post.model";

import { PostsService } from "../posts.service";

@Component({

selector: "app-post-list",

templateUrl: "./post-list.component.html",

styleUrls: ["./post-list.component.css"]

})

export class PostListComponent implements OnInit, OnDestroy {

posts: Post[] = [];

private postsSub: Subscription;

constructor(public postsService: PostsService) {}

ngOnInit() {

this.posts = this.postsService.getPosts();

this.postsSub = this.postsService

.getPostUpdateListener()

.subscribe((posts: Post[]) => { // post.service.ts có emit …post

this.posts = posts;

});

}

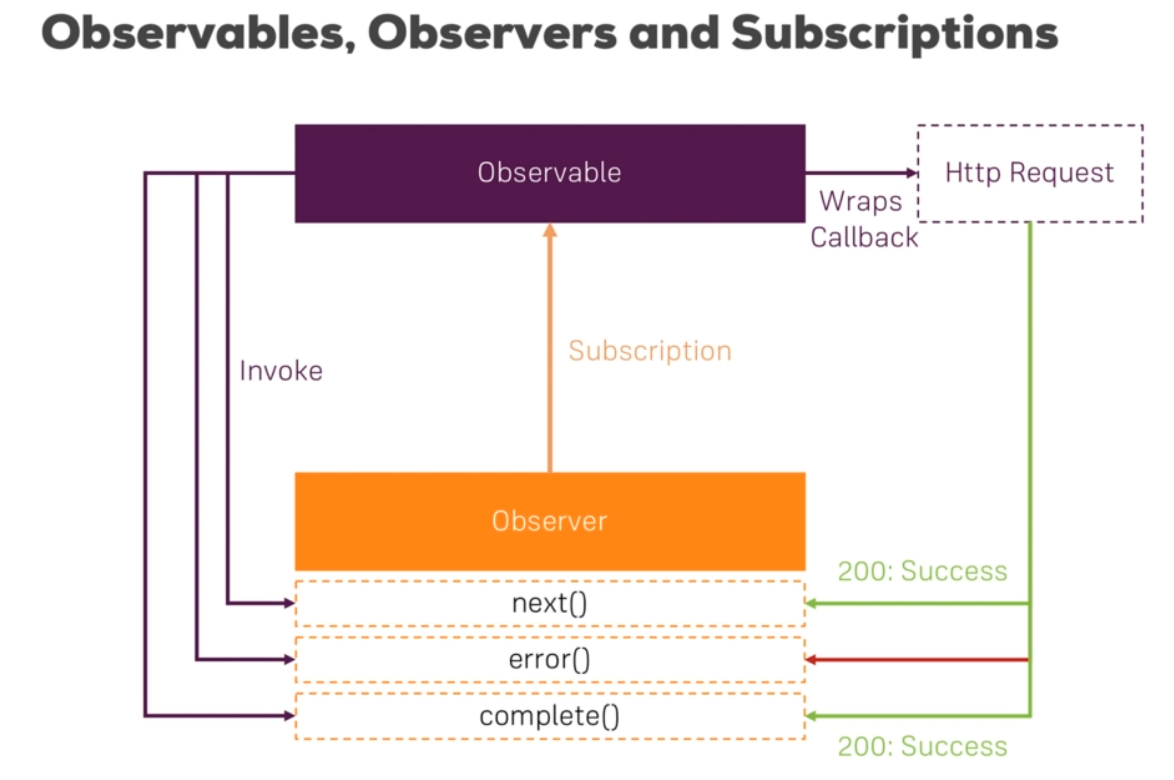
ngOnDestroy() {

this.postsSub.unsubscribe();

}

}

17. More About Observables



ở cuối có hướng dẫn

18. Working on our Form

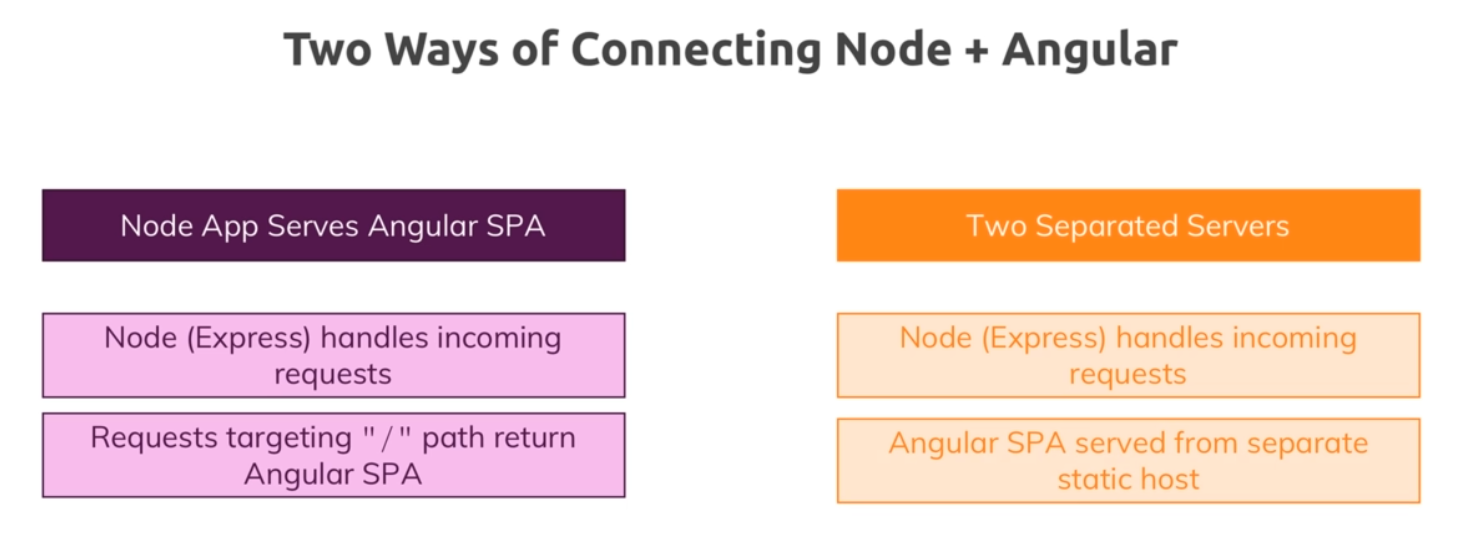
Reset form

Cải tiến thêm 2 file html

## Adding NodeJS to our Project

* Learn Node + Express from Scratch (for free!): <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs>
* Creating a REST API with Node + Express (+ MongoDB): <https://academind.com/learn/node-js/building-a-restful-api-with/>

2. Connect Node to angular



Rest full API là 1 stateless backend, không quan tâm client connect to it

Tạo file server.js và folder backend

const http = require("http");

const server = http.createServer((req, res) => {

res.end("new server");

});

server.listen(process.env.PORT || 3000);

3. Add express framework

npm install --save express

Tạo file app.js

const express = require("express");

const app = express();

app.use((req, res, next) => {

console.log("First middleware");

next();

});

// response

app.use((req, res, next) => {

res.send("Hello from express!");

});

module.exports = app;

4. Improve server.js

Tại file server.js

const app = require("./backend/app");

const debug = require("debug")("node-angular");

const http = require("http");

const normalizePort = val => {

var port = parseInt(val, 10);

if (isNaN(port)) {

*// named pipe*

return val;

}

if (port >= 0) {

*// port number*

return port;

}

return false;

};

const onError = error => {

if (error.syscall !== "listen") {

throw error;

}

const bind = typeof port === "string" ? "pipe " + port : "port " + port;

switch (error.code) {

case "EACCES":

console.error(bind + " requires elevated privileges");

process.exit(1);

break;

case "EADDRINUSE":

console.error(bind + " is already in use");

process.exit(1);

break;

default:

throw error;

}

};

const onListening = () => {

const addr = server.address();

const bind = typeof port === "string" ? "pipe " + port : "port " + port;

debug("Listening on " + bind);

};

const port = normalizePort(process.env.PORT || "3000");

app.set("port", port);

const server = http.createServer(app);

server.on("error", onError);

server.on("listening", onListening);

server.listen(port);

5. Chạy lệnh để tự động restart server node khi có thay đổi

npm install --save-dev nodemon

Sau đó vào file package.json

"scripts": {

"ng": "ng",

"start": "ng serve",

"build": "ng build",

"test": "ng test",

"lint": "ng lint",

"e2e": "ng e2e",

"start:server": "nodemon server.js"

},

Chạy lệnh :

npm run start:server

Sẽ có lỗi ta vài file server.js import

const debug = require("debug")("node-angular");

7. Fetching Initial Posts

file app.js

app.use("/api/posts", (req, res, next) => {

const posts = [{ id: 12343, title: "First", content: "Hello" }];

*//return data json format*

res.status(200).json({

message: "This is a message",

posts: posts

});

});

1. Using the Angular HTTP Client

Thêm constructor và dữ liệu trả về chỉnh model có thêm id và phải vào module import

posts.service.ts

import { Injectable } from "@angular/core";

import { HttpClient } from "@angular/common/http";

import { Subject } from "rxjs";

import { Post } from "./post.model";

import { stringify } from "@angular/compiler/src/util";

@Injectable({ providedIn: "root" })

export class PostsService {

private posts: Post[] = [];

private postsUpdated = new Subject<Post[]>();

constructor(private http: HttpClient) {}

getPosts() {

*// return [...this.posts];*

this.http

.get<{ message: string; posts: Post[] }>(

"http://localhost:3000/api/posts"

)

.subscribe(postData => {

this.posts = postData.posts;

this.postsUpdated.next([...this.posts]);

});

}

getPostUpdateListener() {

return this.postsUpdated.asObservable();

}

addPost(title: string, content: string) {

const post: Post = { id: null, title: title, content: content };

this.posts.push(post);

this.postsUpdated.next([...this.posts]);

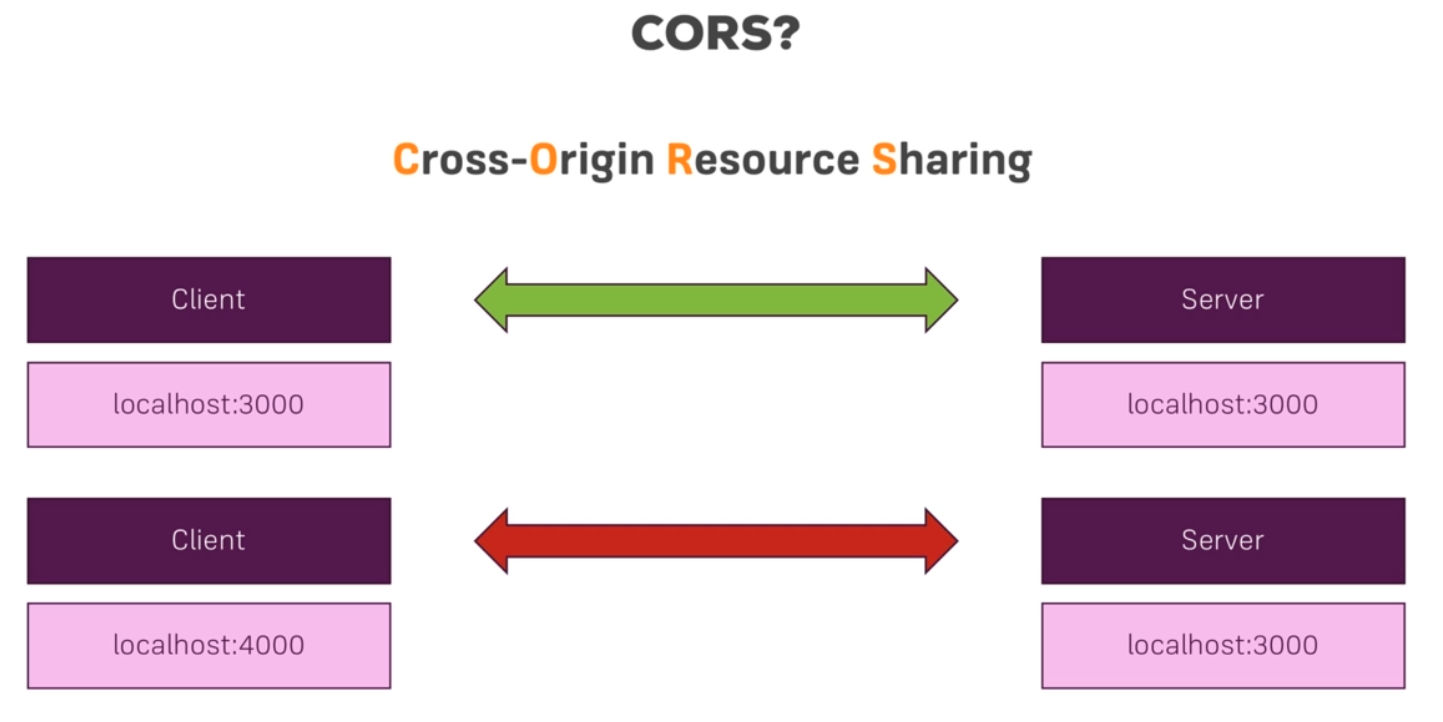
}

}

Vào file post-list.component.ts xóa giá trị trả về của hàm getPosts để hết lỗi

Node 02 add

9. Understanding CORS



Vào file app.js thêm

app.use((req, res, next) => {

res.setHeader("Access-Control-Allow-Origin", "\*");

res.setHeader(

"Access-Control-Allow-Headers",

"Origin, X-Requested-With, Content-Type, Accept"

);

res.setHeader(

"Access-Control-Allow-Methods",

"GET, POST, PATCH, DELETE, OPTIONS"

);

next();

});

10. Adding the POST Backend Point

App.js

app.post("/api/posts", (req, res, next) => {

const post = req.body;

console.log(post);

res.status(201).json({

message: "Post added successfully"

});

});

11. Add body parser

Để parse incoming req body => convert obj

Npm install –save body-parser

const bodyParser = require("body-parser");

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({ extended: false })); // chỉ những tính năng default url encoding

// use

req.body

Adding angular:

Vào file service

addPost(title: string, content: string) {

const post: Post = { id: null, title: title, content: content };

this.http

.post<{ message: string }>("http://localhost:3000/api/posts", post)

.subscribe(responseData => {

console.log(responseData.message);

this.posts.push(post);

this.postsUpdated.next([...this.posts]);

});

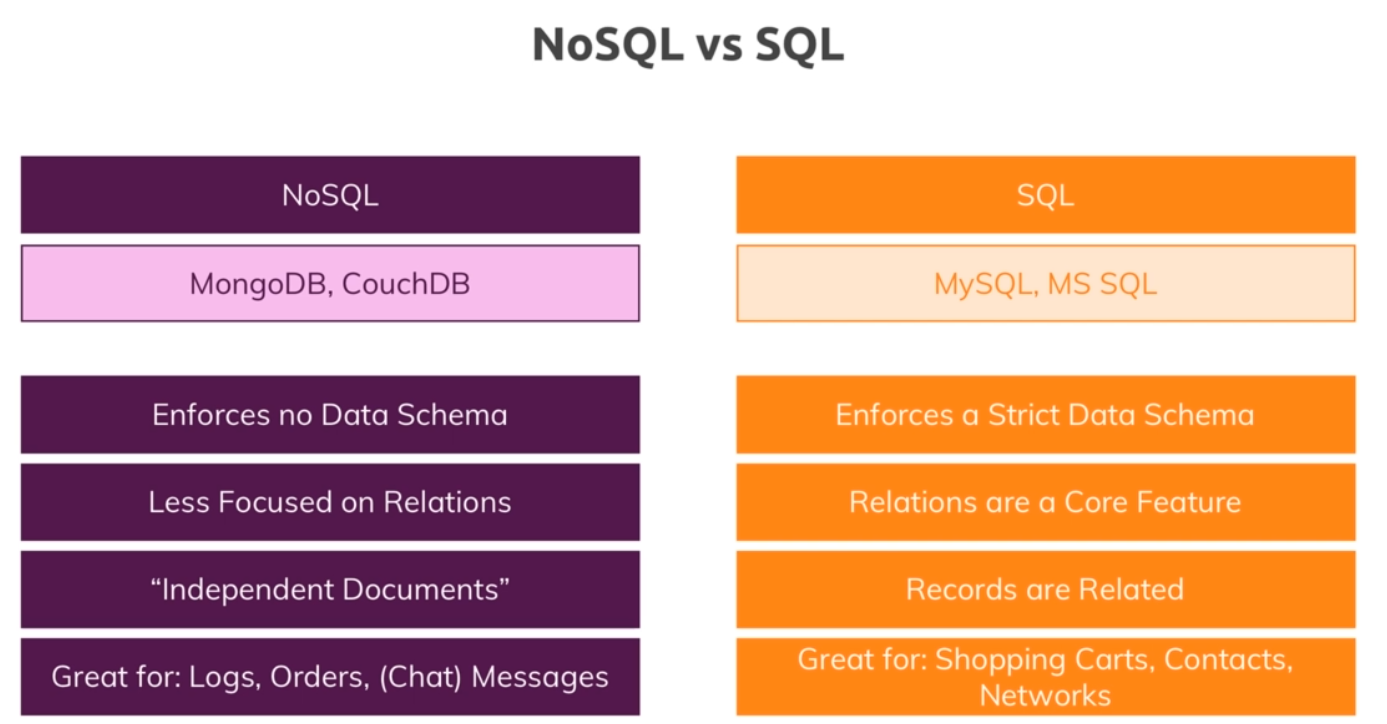
}

## 4. Working with MongoDB

link

* Mongoose Docs: <http://mongoosejs.com/docs/guide.html>
* MongoDB Docs: <https://www.mongodb.com/>
* MongoDB Atlas Docs: <https://www.mongodb.com/cloud/atlas>

So sánh NoSQL và SQL



4. Connecting Angular to a Database

5. Setting Up MongoDB

Download mongodb server <https://www.mongodb.com/download-center/community>

Tải file .tgz

Cloud: <https://www.mongodb.com/download-center/cloud>

By selecting the "Free Sandbox" in the previous video, we're using **MongoDB Atlas** (<https://www.mongodb.com/cloud/atlas>) in this course - a hosted MongoDB cluster.

There, we added our local IP to the "IP Whitelist". **One important note on that**: If you're away from the project for a couple of days (or maybe even after one day), you might've received a new local IP by your internet provider. Hence you **should update that "IP Whitelist" if you're facing any connection issues!**

Vào phần Get started free đăng kí pass duyphuong1020 mail dp20100: name- phuong

ấn vào chữ cluster trên cùng, build new cluster rồi kéo xuống dưới cùng chọn create cluster

Sau đó vào tab security/ chọn add new user/ database access/ read & write…/ nhập name là phuong và pass là 12345

Vào network access/ add/ add current IP address/ confirm

Sau vài ngày mà k vào dk thì cập nhật lại IP

7. Adding Mongoose

Install mongoose

Npm install –save mongoose

Tạo ra file model/post.js để tạo schema: <https://mongoosejs.com/docs/guide.html>

const mongoose = require('mongoose');

const postSchema = mongoose.Schema({

title: { type: String, required: true },

content: { type: String, required: true }

});

module.exports = mongoose.model('Post', postSchema);

9. Creating a POST Instance

Vào file app.js thêm

app.post("/api/posts", (req, res, next) => {

*// const post = req.body;*

const post = new Post({

title: req.body.title,

content: req.body.content

});

console.log(post);

res.status(201).json({

message: "Post added successfully"

});

});

10. Connecting our Node Express App to MongoDB

Vào clusters/ connect your application/ copy link

ở source phần 2:

const mongoose = require("mongoose");

const Post = require("./models/post");

const app = express();

mongoose

.connect(

*// "mongodb+srv://max:QuBqs0T45GDKPlIG@cluster0-ntrwp.mongodb.net/node-angular?retryWrites=true"*

"mongodb+srv://phuong:12345@cluster0-eyssh.mongodb.net/test?retryWrites=true&w=majority",

{ useNewUrlParser: true }

)

.then(() => {

console.log("Connected to database!");

})

.catch(() => {

console.log("Connection failed!");

});

Thêm post.save();

Có thể chỉnh sửa lại tên database thay cho chữ test ở url

Vào clusters/ …shell/ sau đó tại file về run

Vào thư mục bin

Cd bin

mongo "mongodb+srv://cluster0-eyssh.mongodb.net/test" --username **phuong**

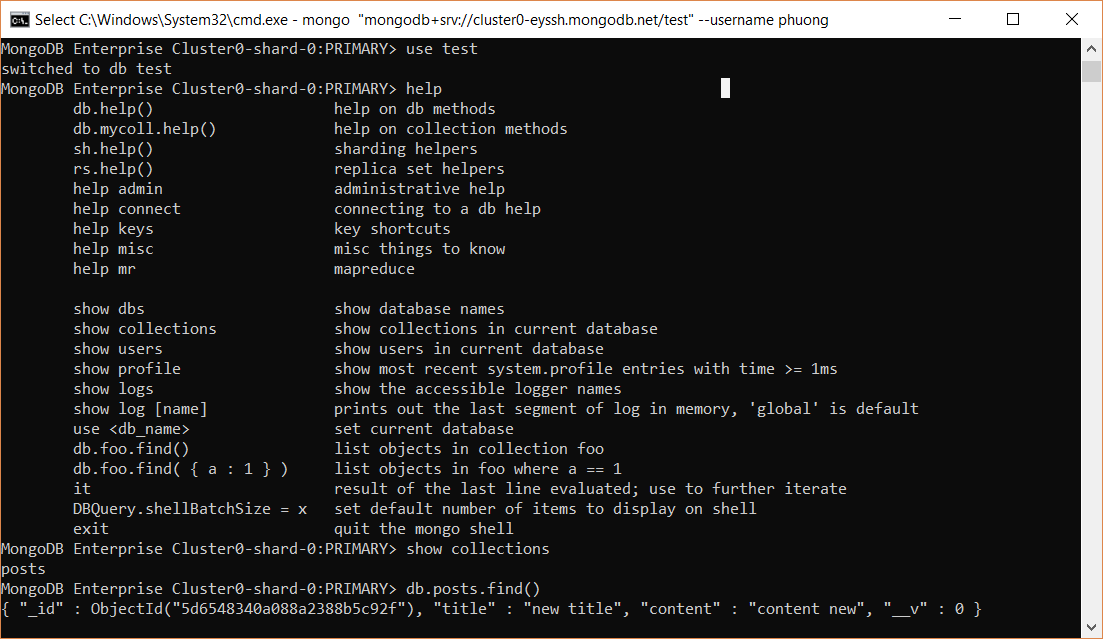
**12345**

Use test

Help

Show collections

Db.test.find()



12. Fetching Data From a Database

Chỉnh lại hàm get

app.get("/api/posts", (req, res, next) => {

Post.find().then(documents => {

res.status(200).json({

message: "Posts fetched successfully!",

posts: documents

});

});

});

13. Transforming Response Data

Khi db trả về \_id khác vs model, vào file post.service.ts thêm hàm pipe

getPosts() {

this.http

.get<{ message: string; posts: any }>("http://localhost:3000/api/posts")

.pipe(

map(postData => {

return postData.posts.map(post => {

return {

title: post.title,

content: post.content,

id: post.\_id

};

});

})

)

.subscribe(transformedPosts => {

this.posts = transformedPosts;

this.postsUpdated.next([...this.posts]);

});

}

14. Deleting Documents

Vào file app.js thêm Api delete

app.delete("/api/posts/:id", (req, res, next) => {

Post.deleteOne({ \_id: req.params.id }).then(result => {

console.log(result);

res.status(200).json({ message: "Post deleted!" });

});

});

Vào file post-list cập nhật hàm delete và thêm event vào file html

onDelete(postId: string) {

this.postsService.deletePost(postId);

}

Vào file service

deletePost(postId: string) {

this.http.delete("http://localhost:3000/api/posts/" + postId)

.subscribe(() => {

console.log('Deleted!');

});

}

15. Updating the Frontend after Deleting Posts

Source phần 3

deletePost(postId: string) {

this.http

.delete("http://localhost:3000/api/posts/" + postId)

.subscribe(() => {

const updatedPosts = this.posts.filter(post => post.id !== postId);

this.posts = updatedPosts;

this.postsUpdated.next([...this.posts]);

console.log("Delete!");

});

}

16. sửa lỗi add post with ID

Sửa hàm

app.post("/api/posts", (req, res, next) => {

const post = new Post({

title: req.body.title,

content: req.body.content

});

post.save().then(createdPost => {

res.status(201).json({

message: "Post added successfully",

postId: createdPost.\_id

});

});

});

Vào file service sửa hàm addPost

const id = responseData.postId;

post.id = id;

## Enhance the app

2. Adding Routing

Tạo file app-routing.modules.ts

import { NgModule } from "@angular/core";

import { RouterModule, Routes } from "@angular/router";

import { PostListComponent } from "./posts/post-list/post-list.component";

import { PostCreateComponent } from "./posts/post-create/post-create.component";

const routes: Routes = [

{ path: '', component: PostListComponent },

{ path: 'create', component: PostCreateComponent }

];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]

})

export class AppRoutingModule {}

Vào module import **AppRoutingModule**

Vào file html thêm:

<app-header></app-header>

<main>

*<!-- <app-post-create></app-post-create>*

*<app-post-list></app-post-list> -->*

<router-outlet></router-outlet>

</main>

Vào header sửa

<mat-toolbar color="primary">

<span>

<a routerLink="/">MyMessages</a>

</span>

<span class="spacer"></span>

<ul>

<li>

<a mat-button routerLink="/create" routerLinkActive="mat-accent">New Post</a>

</li>

</ul>

</mat-toolbar>

Enhancing 01 source

6. Creating the edit Form

Vào file app-routing thêm edit/

const routes: Routes = [

{ path: '', component: PostListComponent },

{ path: 'create', component: PostCreateComponent },

{ path: 'edit/:postId', component: PostCreateComponent },

];

Sau đó vào file post-create

constructor(

public postsService: PostsService,

public route: ActivatedRoute

) {}

ngOnInit() {

this.route.paramMap.subscribe((paramMap: ParamMap) => {

if (paramMap.has("postId")) {

this.mode = "edit";

this.postId = paramMap.get("postId");

this.post = this.postsService.getPost(this.postId);

} else {

this.mode = "create";

this.postId = null;

}

});

File service

getPost(id: string) {

return { ...this.posts.find(p => p.id === id) };

}

File post-list

<a mat-button color="primary" [routerLink]="['/edit', post.id]">EDIT</a>

File post-create sửa:

[ngModel]="post.title"

7. Finishing the Edit Feature

App.js

app.put("/api/posts/:id", (req, res, next) => {

const post = new Post({

\_id: req.body.id,

title: req.body.title,

content: req.body.content

});

Post.updateOne({ \_id: req.params.id }, post).then(result => {

console.log(result);

res.status(200).json({ message: "Update successful!" });

});

});

Service

updatePost(id: string, title: string, content: string) {

const post: Post = { id: id, title: title, content: content };

this.http

.put("http://localhost:3000/api/posts/" + id, post)

.subscribe(response => console.log(response));

}

Post-create sửa hàm onAddPost

onSavePost(form: NgForm) {

if (form.invalid) {

return;

}

if (this.mode === "create") {

this.postsService.addPost(form.value.title, form.value.content);

} else {

this.postsService.updatePost(

this.postId,

form.value.title,

form.value.content

);

}

form.resetForm();

}

Source phần 2

Important: You might be getting an error ("can't find property title") when you reload the Edit page after/ during the next lecture.

This is totally normal and will be fixed in this lecture: <https://www.udemy.com/angular-2-and-nodejs-the-practical-guide/learn/v4/t/lecture/10523172?start=460>

Max

8. Updating Posts on the Server

Service cập nhật lại hàm update post

updatePost(id: string, title: string, content: string) {

const post: Post = { id: id, title: title, content: content };

this.http

.put("http://localhost:3000/api/posts/" + id, post)

.subscribe(response => {

const updatedPosts = [...this.posts];

const oldPostIndex = updatedPosts.findIndex(p => p.id === post.id);

updatedPosts[oldPostIndex] = post;

this.posts = updatedPosts;

this.postsUpdated.next([...this.posts]);

this.router.navigate(["/"]);

});

}

constructor(private http: HttpClient, private router: Router) {}

getPost(id: string) {

return this.http.get<{ \_id: string; title: string; content: string }>(

"http://localhost:3000/api/posts/" + id

);

}

Vào post-create.ts sửa hàm ngOnInit:

this.postsService.getPost(this.postId).subscribe(postData => {

//this.isLoading = false;

this.post = {id: postData.\_id, title: postData.title, content: postData.content};

});

Sửa html

[ngModel]="post?.title"

9. Re-Organizing Backend Routes

Tạo folder routre/post.js

File app.js

app.use("/api/posts", postsRoutes);

10. Adding Loading Spinners

Service

constructor(private http: HttpClient, private router: Router) {}

………

this.router.navigate(["/"]);

import module MatProgressSpinnerModule, vào PostCreateComponent chỉnh sửa isLoading và html

Source phần 3

## Adding Image Uploads to our App

* Multer Docs: <https://github.com/expressjs/multer>
* Tutorial on Angular Image Upload: <https://academind.com/learn/angular/snippets/angular-image-upload-made-easy>

2. Adding the File Input Button

Vào file post-create chỉnh sửa file html

<button mat-stroked-button type="button" (click)="filePicker.click()">Pick Image</button>

<input type="file" #filePicker>

Có thể nhập <input type=”file”> để tạo button chọn file

Chỉnh lại css để nút add image ẩn:

input[type="file"] {

visibility: hidden;

}

3. Converting the Form from a Template Driven to a Reactive Approach

Vào file module chỉnh lại thành :

import { ReactiveFormsModule } from "@angular/forms";

Vào file html xóa

[ngModel]="post?.title"

#postForm="ngForm"

#title="ngModel"

required

minlength="3"

// sửa lại

<form (submit)="onSavePost()" \*ngIf="!isLoading">

Vào file angular post-create import

import { FormGroup, FormControl } from "@angular/forms";

// Khai báo thêm thuộc tính cho class

form: FormGroup;

// hàm init

this.form = new FormGroup({

title: new FormControl(null, {

validators: [Validators.required, Validators.minLength(3)]

}),

content: new FormControl(null, { validators: [Validators.required] }),

image: new FormControl(null, {

validators: [Validators.required]

})

});

…

this.form.setValue({

title: this.post.title,

content: this.post.content

});

Chỉnh lại hàm

onSavePost() {

if (this.form.invalid) {

return;

}

this.isLoading = true;

if (this.mode === "create") {

this.postsService.addPost(this.form.value.title, this.form.value.content);

} else {

this.postsService.updatePost(

this.postId,

this.form.value.title,

this.form.value.content

);

}

this.form.reset();

}

Sửa lại form

<mat-card>

<mat-spinner \*ngIf="isLoading"></mat-spinner>

// [formGroup]="form" tham chiếu qua biến đã khai báo

<form [formGroup]="form" (submit)="onSavePost()" \*ngIf="!isLoading">

<mat-form-field>

<input matInput type="text" formControlName="title" placeholder="Post Title">

<mat-error \*ngIf="form.get('title').invalid">Please enter a post title.</mat-error>

</mat-form-field>

<div>

<button mat-stroked-button type="button" (click)="filePicker.click()">Pick Image</button>

<input type="file" #filePicker>

</div>

<mat-form-field>

<textarea matInput rows="4" formControlName="content" placeholder="Post Content"></textarea>

<mat-error \*ngIf="form.get('content').invalid">Please enter a post title.</mat-error>

</mat-form-field>

<button mat-raised-button color="accent" type="submit">Save Post</button>

</form>

</mat-card>