16시간 ( 8 + 8 )

쉬는시간 : 50분 ~ 정각

점심시간 : 11:50 ~ 13:00

QR : 입실(9시) / 확인(13시) / 퇴실(17:51 이후)

환경설정

1. NodeJS ( ver 18 이상 )

$ node -v

$ npm -v

1. VS Code

교재 pdf는 엣지, 크롬 브라우저에서 열어 주세요

c:\vuejs3\_app 폴더를 생성

node\_modules 폴더가 생성된 후에 js 다운로드가 되어짐

npm i -g @vue/cli

vue --version

<script src=”xxx.jquery.min.js”></script>

VueJS : Vue-CLI <https://cli.vuejs.org/>

ReactJS : Create React App(CRA) https://create-react-app.dev/

Vite : https://vitejs.dev/

1. Babel https://babeljs.io/

: javascript trans compiler

ECMAScript6 버전

Old 브라우저에서 호환성 문제가 발생할 수 있음

ES6 -> ES5

2. Webpack https://webpack.js.org/

: Bundle 번들링 도구

: Dev Server (개발서버) for Client ( HMR hot module replacement )

3. ESLint <https://eslint.org/>

: 코드 정적 분석 도구

CDN(content delivery network)

<script src=”xxxjquery.min.js” >

npm(node package manager) 사용

$ npm i jquery

node\_modules 폴더 아래에 js 다운로드

개발자 작성한 js(vue컴포넌트)와 라이브러리 js 번들링 해서 배포 해야 한다.

2023 stateof js survey 결과

<https://tsh.io/blog/javascript-trends/>

Vue2 Options API 와 Vue3 Composition API 비교 글

<https://vueschool.io/articles/vuejs-tutorials/options-api-vs-composition-api/>

[ ECMAScript6 ]

function sayHello(msg) {

return 'Hello ' + msg;

}

console.log(sayHello('js'));

let sayHello2 = msg => 'Hello ' + msg;

console.log(sayHello2('javascript'));

sayHello2 = msg => {return 'Hello ' + msg;}

console.log(sayHello2('javascript2'));

sayHello2 = msg => ('Hello ' + msg);

console.log(sayHello2('javascript3'));

let my\_arr = [10,20,30,40];

let result = my\_arr.map(val => (val \* 10));

console.log(result);

let result2 = my\_arr.filter(val => val > 20);

console.log(result2);

let result3 = my\_arr.reduce((prev,curr) => prev + curr);

console.log(result3);

my\_arr.forEach(val => console.log(val));

function BlackDog() {

this.name = '흰둥이';

return {

name:'검둥이',

bark: function() {

console.log(this.name + ' 멍멍!');

},

bark2: () => {

console.log(this.name + ' 멍멍!');

}

}

}

const blackDog = new BlackDog();

blackDog.bark(); // 검둥이 멍멍!

blackDog.bark2();

Computed property name 에 대한 설명글

<https://attacomsian.com/blog/javascript-computed-property-names>

let name = '스크립트';

let prop\_key = 'value'

let myObj = {

//name:name

name,

//value:50

[prop\_key]:50

};

console.log(myObj.name);

console.log(myObj.value);

let my\_arr = [10,20,30,40];

//for-in

for(let idx in my\_arr){

console.log(idx + " " + my\_arr[idx]);

}

//for-of

for(let val of my\_arr){

console.log(val)

}

my\_arr.forEach(val => console.log(val));

**//default parameter value**

let volume = (l, w = 3, h = 4 ) => (l \* w \* h);

console.log(volume(2));

//default 값을 지정하지 않은 변수가 항상 첫번째 인자로 선언되어야 한다.

let volume2 = (w = 3, l, h = 4 ) => (l \* w \* h);

console.log(volume2(2)); //NaN

**//Template Literals**

let host = 'aa.com';

let port = 8090

let database = 'mydb';

const db = 'http://' + host + ':' + port + '/' + database;

console.log(db);

// ``(back tick)

const db2 = `http://${host}:${port+10}/

${database}`;

console.log(db2);

**//Array Destructuring Assignment**

let a, b, rest;

[a, b] = [1, 2];

[a, b, ...rest] = [10, 20, 3, 4, 5];

console.log(a);

console.log(b);

console.log(rest);

let foo = ["one", "two", "three"];

let [foo1, foo2, foo3] = foo;

console.log(foo1);

console.log(foo2);

console.log(foo3);

let a1 = 1;

let b1 = 3;

[a1, b1] = [b1, a1];

console.log(a1);

console.log(b1);

**//Object Destructuring Assignment**

let obj1 = {p: 42, q: true};

let val1 = obj1.p;

let val2 = obj1.q;

let {p,q} = obj1

console.log(p);

console.log(q);

const obj = {name:'둘리',

age:10,

phone: {

home: '1234',

office: '5678'

}

};

console.log(obj);

const {name,age,phone:{home,office}} = obj;

console.log(name);

console.log(age);

console.log(home);

console.log(office);

**Spread Operator (펼침연산자) - Array**

// joining arrays

const odd = [1, 3, 5];

const nums = [2 ,4 , 6].concat(odd);

console.log(nums)

const nums2 = [2, ...odd, 4, 6];

console.log(nums2)

// cloning arrays

const arr = [1, 2, 3, 4];

const arr2 = arr.slice();

console.log(arr2);

const arr3 = [...arr];

console.log(arr3);

//주소(ref) 복사 되므로 원본 배열도 수정이 된다. (bad)

const refcopy = arr;

refcopy.push(10);

console.log(arr)

console.log(refcopy);

//값(value) 복사 되므로 원본 배열이 수정되지 않는다(good)

const valcopy = [...arr];

valcopy.push(20);

console.log(arr)

console.log(valcopy);

const { a, b, ...z } = { a: 1, b: 2, c: 3, d: 4 };

console.log(a) // 1

console.log(b) // 2

console.log(z) // { c: 3, d: 4 }

const {c, d} = z;

console.log(c);

console.log(d);

**Spread Operator (펼침연산자) - Object**

const name = '객체이름';

const addr = {

home:'sss',

office:'ooo'

};

const customer = {

name,

home:'tttt',

...addr,

};

console.log(customer);

==========Vue Directive 예제 시작

Template

<body>

<script src="https://cdn.jsdelivr.net/npm/vue@2.7.14/dist/vue.js"></script>

<div id="app">

<h1>Hello, {{ name }}</h1>

<input type="text" v-model="name" />

<h1>Hello, <span v-text="name">정적인Span</span></h1>

<h1>Hello, <span v-show="isFlag" v-html="htmlName"></span></h1>

<h1 v-if="value > 5">value가 5보다 큽니다</h1>

<input type="checkbox" v-model="isFlag"/>Logo 이미지

<img :src="isFlag? vuelogo:anglogo" width="100"/>

<ul>

<li v-for="(todo,idx) in todos" :key="idx">

{{idx}} / {{ todo.text }}

</li>

</ul>

<h1>카운터: {{ value }}</h1>

<button @click="increment">증가</button>

<button v-on:click="decrement">감소</button>

</div>

</body>

Script

var app = new Vue({

el: '#app', // 어떤 엘리먼트에 적용을 할 지 정합니다

// data 는 해당 뷰에서 사용할 정보를 지닙니다

data: {

name: 'Vue',

htmlName: '<i>Vue</i>',

isFlag: true,

value: 0,

vuelogo: 'https://vuejs.org/images/logo.png',

anglogo: 'https://angular.io/assets/images/logos/angular/angular.svg',

todos: [

{ text: 'Vue.js 튜토리얼 작성하기' },

{ text: 'Webpack2 알아보기' },

{ text: '사이드 프로젝트 진행하기' }

],

},//data

methods: {

increment() {

// 인스턴스 내부의 데이터모델에 접근 할 땐, this 를 사용한다

this.value++;

},

decrement() {

this.value--;

}

},//method

});

==========Vue Directive 예제 끝

오류 발생시 조치사항들 정리

아래와 같은 오류가 발생하면 기존의 Vetur 플러그인 설치되어 있다면 uninstall 하면 됩니다.

[vue/no-multiple-template-root]

The template root requires exactly one element.eslint-plugin-vue

아래와 같은 오류가 발생하면 "requireConfigFile": false 추가

Either disable config file checking with requireConfigFile: false, or configure Babel

package.json 파일에

"eslintConfig": {

....

"parserOptions": {

"parser": "@babel/eslint-parser",

"requireConfigFile": false

},

}

아래와 같은 오류가 발생하면 ( console 상에 )

Feature flag \_\_VUE\_PROD\_HYDRATION\_MISMATCH\_DETAILS\_\_ is not explicitly defined. You are running the esm-bundler build of Vue,

vue.config.js 수정 ( 밑줄 )

const { defineConfig } = require('@vue/cli-service')

const webpack = require('webpack')

module.exports = defineConfig({

transpileDependencies: true,

configureWebpack: {

plugins: [

new webpack.DefinePlugin({

// Vue CLI is in maintenance mode, and probably won't merge my PR to fix this in their tooling

// https://github.com/vuejs/vue-cli/pull/7443

\_\_VUE\_PROD\_HYDRATION\_MISMATCH\_DETAILS\_\_: 'false',

})

],

},

})

VUE-CLI

$ vue create vue3-todo

package.json 파일이 생성됨

"scripts": {

"serve": "vue-cli-service serve",

"build": "vue-cli-service build",

"lint": "vue-cli-service lint"

},

Development (개발)환경

$ npm run serve

=> compile & dev server start

Production (운영)환경 빌드

$ npm run build

=> compile & bundling

[http://localhost:8080/](http://localhost:8081/)

src/main.js

index.html =====> src/App.vue ===> src/components/HelloWorld.vue

public/

index.html

<div id="app"></div>

src

main.js

//Version 2.0

new Vue({

render: h => h(App),

}).$mount('#app')

//Version 3.0

createApp(App).mount('#app')

App.vue

<HelloWorld msg="Welcome to Your Vue.js App"/>

Vite

package.json

"scripts": {

"dev": "vite",

"build": "vite build",

"preview": "vite preview"

},

Development (개발)환경

$ npm run dev

Production (운영)환경 빌드

$ npm run build

index.html

<div id="app"></div>

<script type="module" src="/src/main.js"></script>

src/main.js

createApp(App).mount('#app')

src/App.vue

<HelloWorld msg="Vite + Vue" />

Vue2 methods와 computed 비교 설명

https://v2.vuejs.org/v2/guide/computed.html

Express 기반 서버와 ajax 통신

store.js에 actions property 추가

module 화 하기

store.js 모듈화

환경설정 파일

.env.development

: dev mode

.env.production

: prod mode

환경변수의 이름은 VUE\_APP 로 시작해야 합니다.

$npm run build

빌드한 파일들 서버에 deploy 하기

Todo 목록

GET <http://localhost:4500/api/todos>

Todo 1개

GET <http://localhost:4500/api/todos>/1

Todo 등록

POST <http://localhost:4500/api/todos>

header

content-type:application/json

body raw 선택

{

"item": "Todo4",

"completed": true

}

Todo 수정

PUT

<http://localhost:4500/api/todos>/1

header

content-type:application/json

body raw 선택

{

"item": "Todo44",

"completed": false

}

npm install vue-router

main.js (router, store)

index.html =====> App.vue ===>

/ ===> views/HomeView.vue ===> HelloWorld.vue ===> TodoList.vue

/about ===> views/AboutView.vue (lazy loading)

router/index.js

VueRouter 객체사용

path와 views/컴포넌트를 매핑

GET <http://localhost:4500/api/posts>

GET http://localhost:4500/api/posts/1

POST <http://localhost:4500/api/posts>

Pinia

https://pinia.vuejs.org/

<https://www.youtube.com/watch?v=Elv97p2jXK8>

Pinia 예제

<https://blog.deepgram.com/build-a-todo-list-with-pinia-and-vue-3/>

https://github.com/deepgram-devs/todo-pinia/tree/main

Vite 기반 프로젝트의 env 파일 작성 규칙

.env.development

VITE\_TITLE=Dev

VITE\_APIURL=http://localhost:4500/api/todos

설정된 환경변수를 사용하기

const api\_url = import.meta.env.VITE\_APIURL