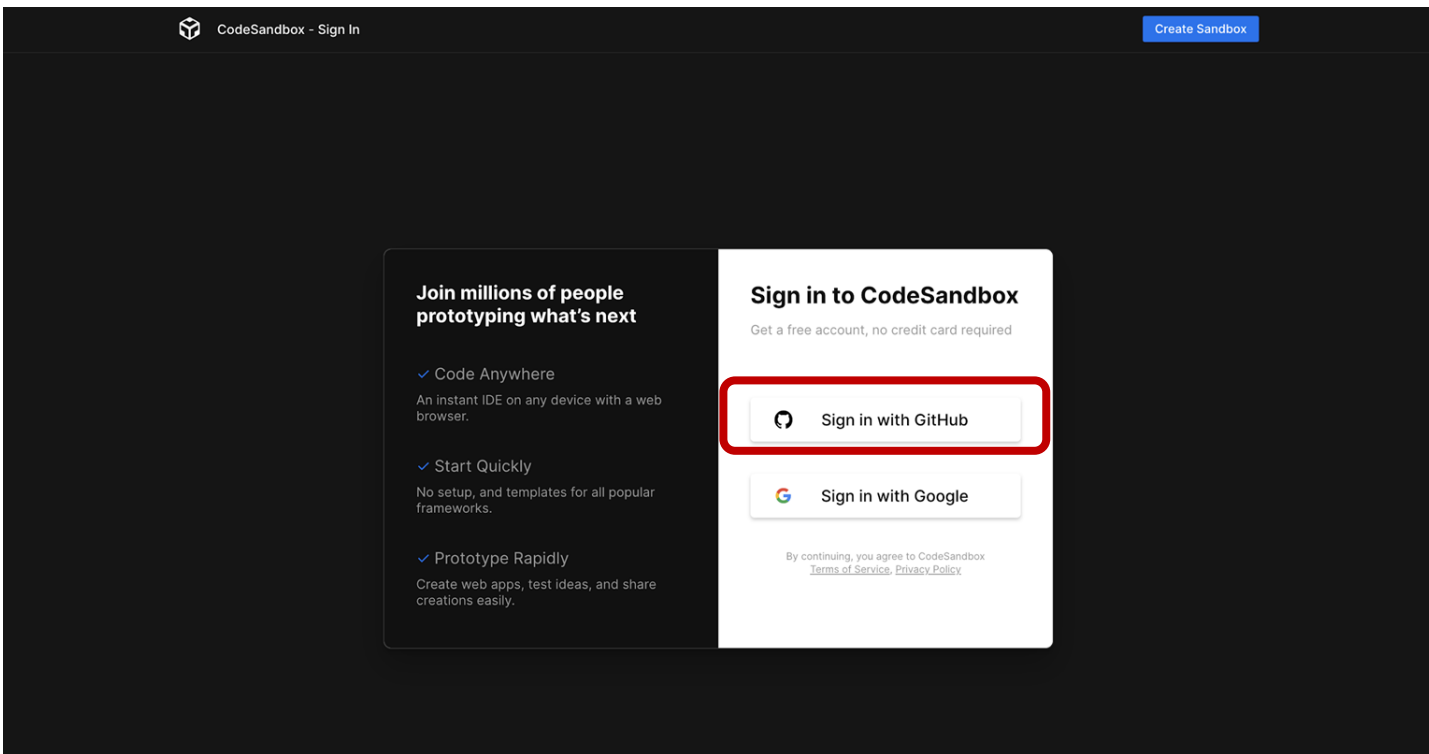


Prerequisite

1. สมัคร account Github : <https://github.com/>
2. เปิด Code Sandbox และทำตามสไลด์ถัดไป : <https://codesandbox.io/signin>

Prerequisite

Sign in Code Sandbox ជាមួយ Github



Prerequisite

Sign in Code Sandbox ត້យប័ណ្ណ Github



Sign in to GitHub
to continue to CodeSandbox

Username or email address

Password

[Forgot password?](#)

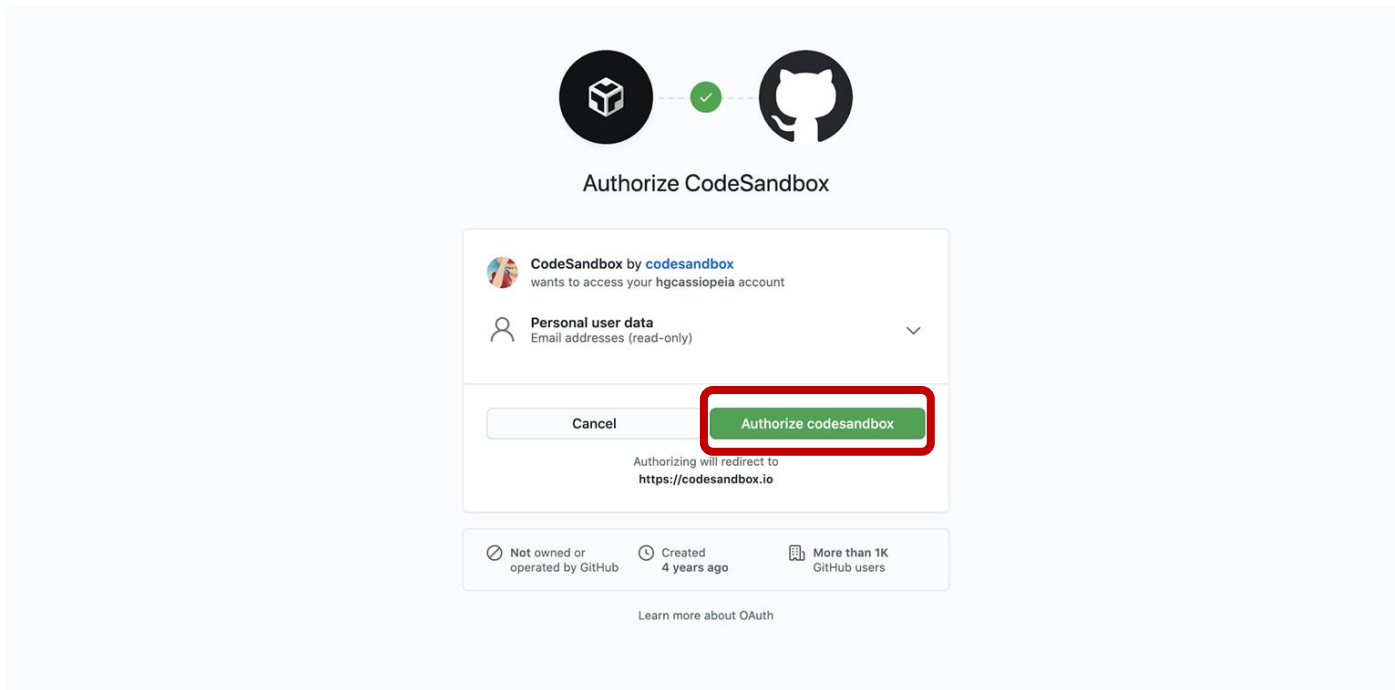
Sign in

New to GitHub? [Create an account.](#)

[Terms](#) [Privacy](#) [Security](#) [Contact GitHub](#)

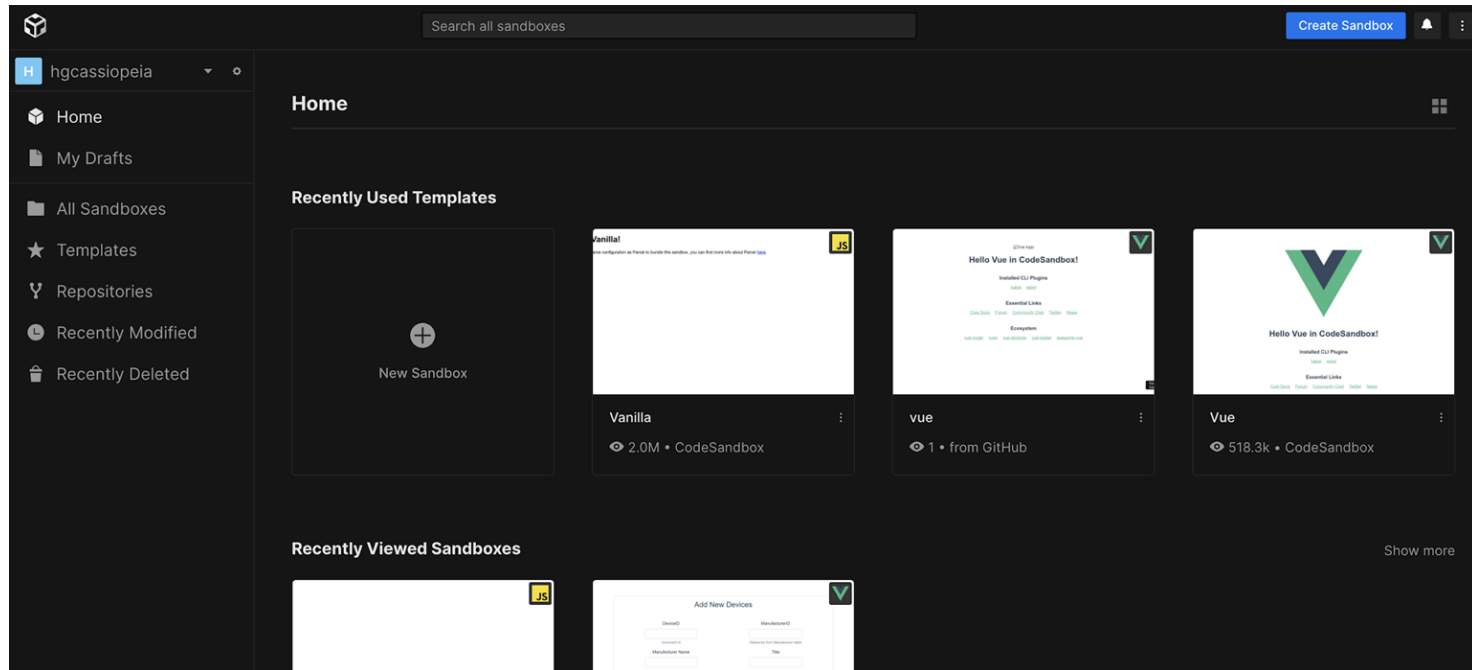
Prerequisite

Sign in Code Sandbox តំបន់ Github

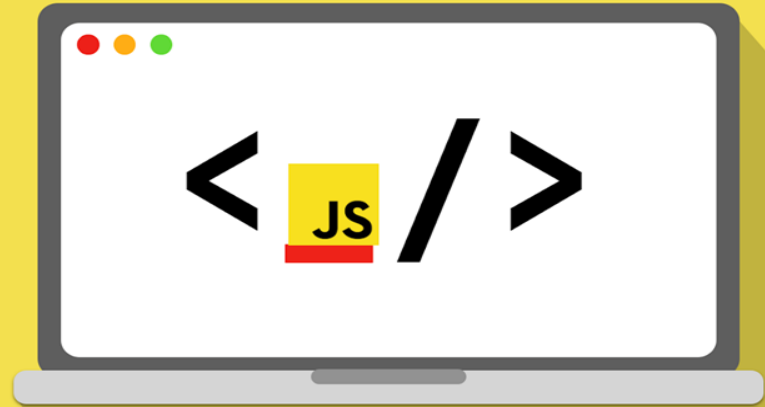


Prerequisite

Sign in Code Sandbox ด้วຍັလູຮັ Github



JAVASCRIPT



What is JavaScript

JavaScript คือ ภาษาที่ใช้งานร่วมกับ HTML เพื่อช่วยให้เว็บไซต์มีลูกเล่น สามารถโต้ตอบกับผู้ใช้งานได้ สามารถใช้เป็นโค้ดฝั่ง Server ให้เชื่อมต่อกับ Database ได้ด้วย Node.js อีกทั้งยังใช้ใน Web Framework ยอดนิยมอย่าง React/Vue/Angular ได้ด้วย

Content

- 1 Syntax พื้นฐาน JavaScript
- 2 การเขียน Ternary Operator
- 3 การเขียน import, export
- 4 รู้จักกับ 'Backtick' และการเขียน Template String
- 5 การเขียน Object Destructuring

Syntax พื้นฐาน JavaScript

คำสั่งแสดงผลทาง Console หรือ Debug

```
1 console.log('Hello')
2 console.warn('Hello')
3 console.error('Hello')
4
5 const users = [
6   { name: 'A', age: 20 },
7   { name: 'B', age: 25 }
8 ]
9 console.table(users)
10
11 const condition = 2 > 3
12 console.assert(condition, "it's false")
```

Web Console เป็นเครื่องมือที่ใช้ในการ Debug หรือหาข้อผิดพลาดของโปรแกรม

โดยคำสั่งที่ใช้งานบ่อย ๆ คือ console.log นอกจากนี้คำสั่ง console ยังมีอีกหลายตัวที่ช่วยให้สามารถ Debug ตัวแปรได้สะดวกขึ้น

| Syntax พื้นฐาน JavaScript

การสร้างตัวแปรด้วย let/const

let และ **const** เป็น keyword ที่ใช้ในการประกาศตัวแปร โดย let จะสามารถเปลี่ยนแปลงค่าของตัวแปรที่ประกาศได้ แต่ const ไม่สามารถเปลี่ยนได้ เนื่องจากเป็นค่าคงที่

ทั้ง let/const เป็นการประกาศตัวแปรที่ทำงานอยู่ภายในขอบเขตที่มันถูกประกาศ

การประกาศตัวแปร

```
1 let message = 'Hello'
2 message = 'Hi' // ทำได้
3
4 const message = 'Hello'
5 message = 'Hi' // TypeError: Assignment to constant variable
6
```

ผลลัพธ์เมื่อ Assign ค่าให้กับตัวแปรที่ประกาศด้วย let

```
1 let message = 'Hello'
2 message = 'Hi' // ทำได้
3
4 const message = 'Hello'
5 message = 'Hi' // TypeError: Assignment to constant variable
6
```

ผลลัพธ์เมื่อ Assign ค่าให้กับตัวแปรที่ประกาศด้วย const

```
1 let message = 'Hello'
2 message = 'Hi' // ทำได้
3
4 const message = 'Hello'
5 message = 'Hi' // TypeError: Assignment to constant variable
6
```

let/const ជាអ្វី? var មិន?



```
1 const items = 3
2 for(var i = 1; i < items; i++){
3   console.log("loop: ",i) // loop: 1, loop: 2
4 }
5 console.log("result: ",i) // result: 3
6
```

Syntax พื้นฐาน JavaScript

การสร้าง Function

function เป็น keyword ที่ใช้ในการกำหนดการทำงานของโปรแกรมหรือแอปพลิเคชัน

โดย function นั้นสามารถที่จะรับ **parameter** หรือก็คือการกำหนดตัวแปรที่ทำหน้าที่รับค่าจากข้างนอก function เข้ามาใช้งานภายใน function ได้

นอกจากนี้ function ยังสามารถใช้คำสั่ง **return** เพื่อส่งค่าจากใน function ออกไปใช้งานข้างนอก function ได้

เราสร้าง function ยังไง?



```
1 /* สร้าง function */  
2 function sayHello(text){  
3   return 'Function say '+text  
4 }  
5  
6 /* เรียกใช้งาน function */  
7 const message = 'Hello'  
8 console.log(sayHello(message)) // Function say Hello
```


กำหนดการทำงานใน function



```
1 /* สร้าง function */  
2 function sayHello(text){  
3   return 'Function say '+text  
4 }  
5  
6 /* เรียกใช้งาน function */  
7 const message = 'Hello'  
8 console.log(sayHello(message)) // Function say Hello
```

ลองประกาศตัวแปรเพื่อเก็บข้อความที่จะส่งไปใน function



```
1 /* สร้าง function */  
2 function sayHello(text){  
3   return 'Function say '+text  
4 }  
5  
6 /* เรียกใช้งาน function */  
7 const message = 'Hello'  
8 console.log(sayHello(message)) // Function say Hello
```

ลองเรียกใช้ function

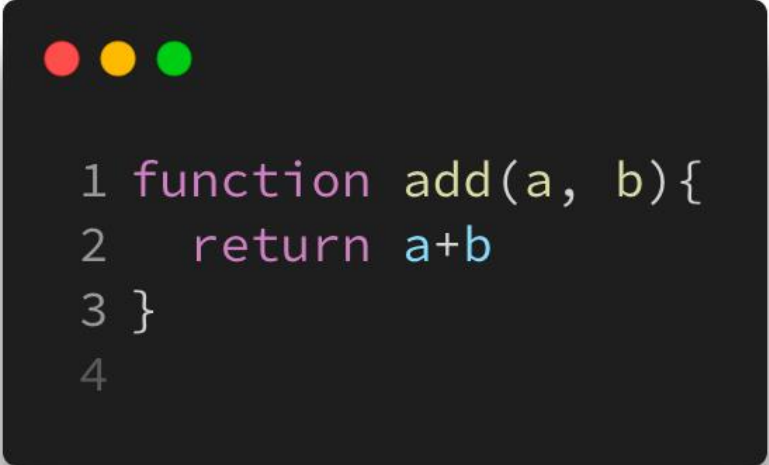


```
1 /* สร้าง function */  
2 function sayHello(text){  
3   return 'Function say '+text  
4 }  
5  
6 /* เรียกใช้งาน function */  
7 const message = 'Hello'  
8 console.log(sayHello(message)) // Function say Hello
```

I การเขียน Ternary Operator

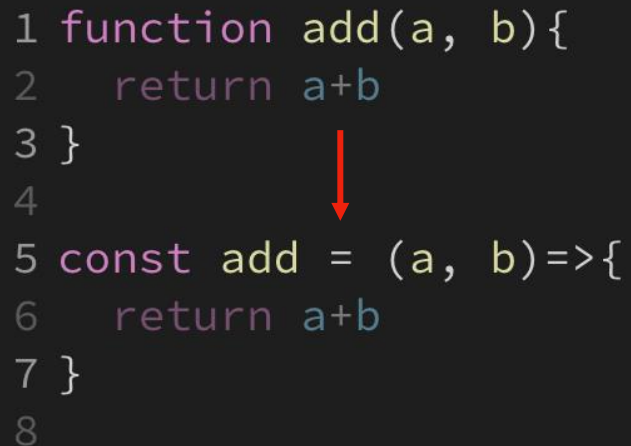
Ternary Operator คือการเขียน code แบบย่อโครงสร้าง เช่น short if, short function (arrow function)

ปกติเราเขียน function ยังไง?



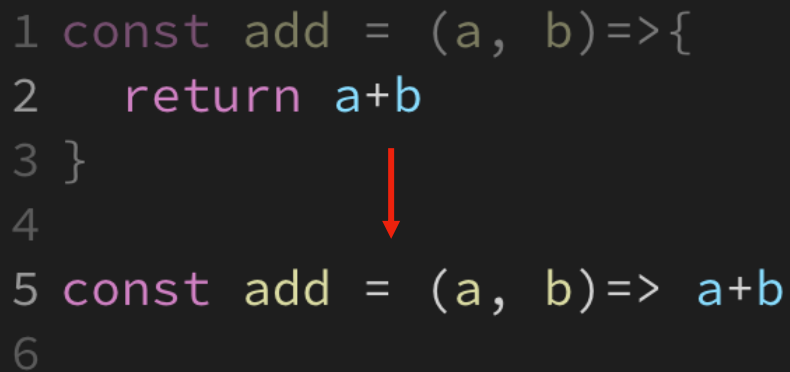
```
1 function add(a, b){  
2   return a+b  
3 }  
4
```

เราจะเขียนแบบ Arrow functions ได้ยังไง?



```
1 function add(a, b){  
2   return a+b  
3 }  
4  
5 const add = (a, b)=>{  
6   return a+b  
7 }  
8
```

เขียนสั้นได้อีกไหม?



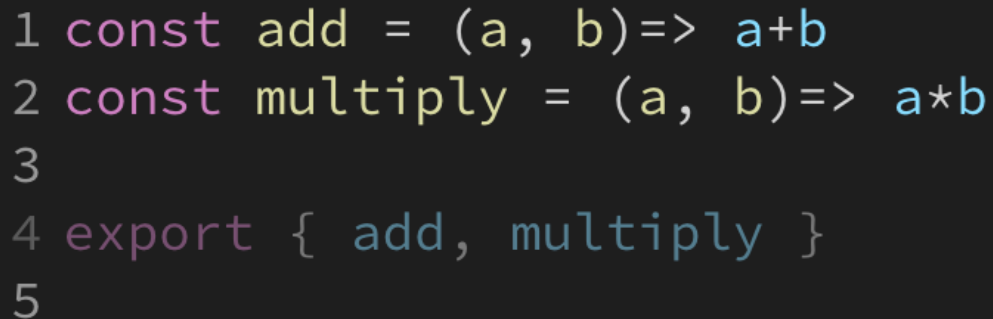
```
1 const add = (a, b) => {  
2   return a+b  
3 }  
4  
5 const add = (a, b) => a+b  
6
```

I การเขียน import, export

Modules (import/export) คือ คำสั่งที่ใช้ในการแยก code js ไฟล์เดียว ให้แยกออกมาเป็นหลายๆชิ้นส่วน หรือ หลายๆ Modules ได้

จะแยก code js ออกเป็น modules ได้ยังไง?

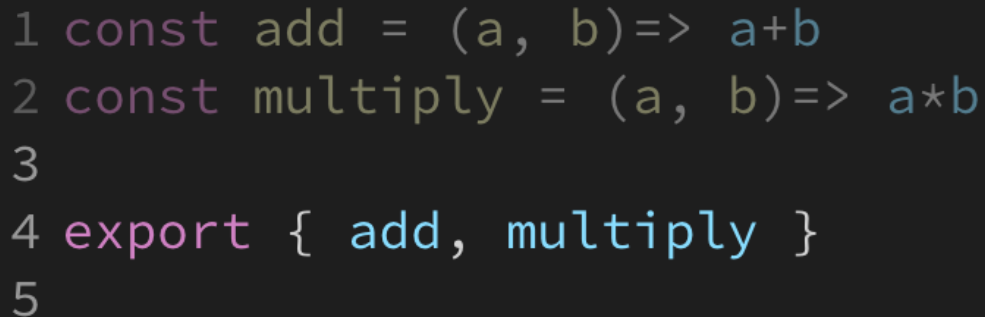
ดู ไฟล์ calculator.js



```
1 const add = (a, b) => a+b  
2 const multiply = (a, b) => a*b  
3  
4 export { add, multiply }  
5
```

จะแยก code js ออกเป็น modules ได้ยังไง?

ดู ไฟล์ calculator.js



```
1 const add = (a, b)=> a+b  
2 const multiply = (a, b)=> a*b  
3  
4 export { add, multiply }  
5
```


จะ import modules ที่ export มาแล้วได้ยังไง?

ดู ไฟล์ main.js

```
1 import { add } from './calculator'
2 console.log(add(2, 3)) // 5
3
```

ຈະ import ຖຸກ modules ຍັງໄວ?

ລູ ໄວ້ main.js



```
1 import * as cal from './calculator'
2 console.log(cal.add(2, 3)) // 5
3 console.log(cal.multiply(2, 3)) // 6
4
```

I การเขียน Template String

Template Strings เป็นวิธีการเขียน string ที่สามารถแทรกคำสั่งหรือตัวแปรใน string ได้ และสามารถเขียนได้หลายบรรทัด

โดยวิธีการเขียน Template strings สามารถใช้ ` (backtick) ในการเขียน string แทน ' (single quote) หรือ " (double quote)

และการแทรกตัวแปรลงใน string สามารถใช้ \${ตัวแปร}

ms concat string แบบปกติ

```
1 const user = 'A-Host'
2 const message1 = 'Hello, ' + user
3 const message2 = `Hello, ${user}`
4 console.log(message1) // Hello, A-Host
5 console.log(message2) // Hello, A-Host
6
```

จะเขียน template strings ได้ยังไง?



```
1 const user = 'A-Host'
2 const message1 = 'Hello, ' + user
3 const message2 = `Hello, ${user}`
4 console.log(message1) // Hello, A-Host
5 console.log(message2) // Hello, A-Host
6
```

มากกว่า 1 บรรทัดหน้าต่างเป็นยังไง?

```
1 const user = 'A-Host'
2 const message = `
3 Hello, ${user}.
4 This is JS class
5 `
6
7 console.log(message)
8 // Hello, A-Host.
9 // This is JS class
10
```


I การเขียน Object Destructuring

การ **Destructuring** คือ การดึงค่าของ Object มาใส่ในตัวแปร เพื่อลดโครงสร้างหรือความลึกของ Object นั้น ๆ

ปกติจะดึงค่าจาก Object ยังไง?

```
1 const studentA = {  
2   name: 'A',  
3   score: 100  
4 }  
5 const name = studentA.name  
6 const score = studentA.score  
7 console.log(`${name}: ${score}`) // A: 100  
8
```

ปกติจะดึงค่าจาก Object ยังไง?

```
1 const studentA = {  
2   name: 'A',  
3   score: 100  
4 }  
5 const name = studentA.name  
6 const score = studentA.score  
7 console.log(`${name}: ${score}`) // A: 100  
8
```

แล้ว Object Destructuring เขียนยังไง?



```
1 const studentA = {  
2   name: 'A',  
3   score: 100  
4 }  
5 const { name, score } = studentA  
6 console.log(`${name}: ${score}`) // A: 100  
7
```

แล้ว Object Destructuring เขียนยังไง?

```
1 const studentA = {  
2   name: 'A',  
3   score: 100  
4 }  
5 const { name, score } = studentA  
6 console.log(`${name}: ${score}`) // A: 100  
7
```

ถ้าอยากตั้งชื่อใหม่ให้ตัวแปรที่ Destructuring แล้วล่ะ?



```
1 const studentA = {  
2   name: 'A',  
3   score: 100  
4 }  
5 const { name, score:myScore } = studentA  
6 console.log(`${name}: ${myScore}`) // A: 100  
7
```

Getting started with **Vue.js**





What is a Vuejs

Vuejs เป็นหนึ่งใน Web framework ถูกสร้างขึ้น
ขึ้นมาในปี 2014 พัฒนาโดย “**Evan You**” ทำ
หน้าที่เป็น View ใน MVC (Model View
Controller) ที่ไว้สำหรับพัฒนา UI (User
Interface)



Content

- 1 รู้จักกับ Component
- 2 รู้จักกับ Vue Properties
- 3 การใช้งาน Directive Vue.js
- 4 รู้จักกับ Vue Router
- 5 รู้จักกับ Slot
- 6 รู้จักกับ Axios



What is a Component

Component คือ ส่วนประกอบย่อยที่ประกอบมาเป็นแอปพลิเคชัน เช่น Component สำหรับแสดงเมนูหลักด้านบน Component สำหรับแสดงช่องค้นหา Component สำหรับแสดงไซด์บาร์ด้านข้าง Component สำหรับแสดงเรื่องราวต่างๆ ของเว็บไซต์ ฯลฯ

<search-bar>

<menu-bar>



Search

Home Following Worst



<card>



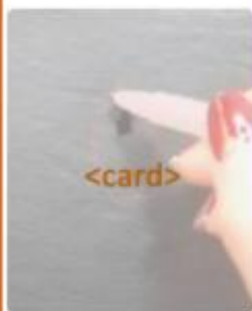
<card>



<card>



gettyimages.com



<card>

▶ 3.25

@choey Chantanaat

Common Opposite Verbs

open	off	close	play	on	work
has	off	into	garden	in	move
down	in	upright	at	in	stand
that	in	all	from	in	catch
down	in	and	hard	in	fast
hold	in	and	for	in	the
into	in				in
put	in	move	the	in	moving
down	in	the	for	in	the
up	in	and	put	in	put
upright	in	and	just	in	the
down	in	the	into	in	the
up	in	the	straight	in	the
up	in	the	into	in	the
up	in	the	up	in	the





Vue Component

Template

Script

Style

```
1 <template>
2   <p>{{ myPropertyName }} Company!</p>
3 </template>
4
5 <script>
6 export default{
7   name: 'MyComponent',
8   data() {
9     return {
10       myPropertyName: 'A-Host'
11     }
12   }
13 }
14 </script>
15
16 <style scoped>
17   p {
18     font-size: 2em;
19     text-align: center;
20   }
21 </style>
```



Vue Properties

`name` : ใช้ในการตั้งชื่อให้กับ Component

`components` : ใช้ในการ Import component อื่นๆ เข้ามาใช้งานร่วมกัน

`data()` : เป็นพื้นที่ในการประกาศตัวแปรของ Component

`methods` : เป็นพื้นที่ในการสร้างการทำงาน หรือ Function สำหรับ Component

`created()` : เป็น Event properties ที่จะทำงานเมื่อ component ถูกสร้าง

`watch` : เป็นพื้นที่ในการเฝ้าดูค่าของตัวแปร เมื่อมีการประมวลผลหรือเปลี่ยนแปลง

`computed` : เป็นพื้นที่ในการสร้างตัวแปรที่มีการคำนวณจากค่าอื่นๆก่อนนำออกมาใช้งาน



Directive Vue.js

- v-model
- v-bind
- v-if
- v-for
- v-on



V-Model

v-model จะทำงานร่วมกับตัวแปรของ Vue โดยเมื่อตัวแปรนั้นมีการเปลี่ยนแปลงค่า ค่าที่เปลี่ยนแปลงนั้นจะ Update ที่ v-model กันที หรือถ้า Element นั้นมีการเปลี่ยนแปลงค่าของ v-model ก็จะมี update ที่ตัวแปรด้วย การทำงานลักษณะแบบสองทิศทางแบบนี้ เราเรียกว่า “Two-Way Data Binding”



Ex. Data Binding

One Way Data Binding

<template></template>



<script></script>

<template></template>



<script></script>

Two Way Data Binding

<template></template>



<script></script>



V-Model

```
1 <!-- vue html -->
2 <template>
3   <input v-model="name" type="text" placeholder="name" />
4   <h1>{{ name }}</h1>
5 </template>
6 <!-- vue js -->
7 <script>
8 export default{
9   data() {
10     return {
11       name: ''
12     }
13   }
14 }
15 </script>
```



Vue Router

Vue Router คือ Library ตัวหนึ่งของ vue ที่ใช้ในการกำหนดเส้นทางการแสดงผลของหน้าเว็บหรือสร้าง link เพื่อไปยังหน้าต่างๆ โดยการทำงานจะเป็นแบบ SPA (Single page Application)

Single Page Application คือ เราไม่ต้องโหลดหน้าขึ้นมาใหม่เมื่อมีการเปลี่ยนหน้า ข้อดีคือ มันจะไม่โหลด style , logo , script ต่างๆใหม่ แต่จะโหลดเฉพาะส่วนที่มีการเปลี่ยนแปลง



Vue Router

Create File : src/router/Index.js

```
import Vue from "vue";
import Router from "vue-router";

Vue.use(Router);

export default new Router({
  mode: "history",
  routes: [
    {
      path: "/",
      component: () => import("../components/HelloWorld.vue")
    },
  ]
});
```



Vue Router

Router : src/main.js

```
import Vue from "vue";
import App from "./App.vue";
import router from "./router";

Vue.config.productionTip = false;

new Vue({
  render: (h) => h(App),
  router
}).$mount("#app");
```



Vue Router

`<router-link>` คือ การสร้างปุ่ม link ที่ไม่ทำให้เกิดการ refresh ที่ browser

`<router-view>` คือ พื้นที่ที่นำ Page หรือ Component มาแสดงผล



V-on

v-on เป็นคำสั่งที่ใช้ร่วมกับ attribute ของ html ที่เป็น Event Listener เช่น click, submit, drag หรือ input เป็นต้น โดย v-on จะทำให้ event เหล่านั้นสามารถเรียกใช้ method หรือ function ของ vue ได้



V-on

```
1 <!-- vue html -->
2 <template>
3   <input
4     v-on:input="convertToUpperCase"
5     type="text"
6     placeholder="input text"
7     :value="message"
8   />
9 </template>
10 <!-- vue js -->
11 <script>
12 export default{
13   data() {
14     return {
15       message: ''
16     }
17   },
18   methods: {
19     convertToUpperCase(event) {
20       this.message = event.target.value.toUpperCase()
21     }
22   }
23 }
24 </script>
```



V-on เขียนย่อ

```
1 <input
2     v-on:input="convertToUpperCase"
3     type="text"
4     placeholder="input text"
5     :value="message"
6 />
```



```
1 <input
2     @input="convertToUpperCase"
3     type="text"
4     placeholder="input text"
5     :value="message"
6 />
```




V-on

input text





V-Bind

v-bind ใช้สำหรับการเปลี่ยนแปลง หรือกำหนดค่าต่างๆ ไม่ว่าจะเป็น attribute, style, src เป็นต้น เช่น เราต้องการที่จะสร้างปุ่มขึ้นมา และเมื่อทำการคลิกให้ทำการเปลี่ยนสีตัวอักษรด้วย



V-Bind

```
1 <!-- vue html -->
2 <template>
3   <h1 v-bind:style="color">Hello World!</h1>
4   <button @click="changeColor">Change Color</button>
5 </template>
6 <!-- vue js -->
7 <script>
8 export default{
9   data() {
10     return {
11       colors: { color: '#000000' }
12     }
13   },
14   methods: {
15     changeColor() {
16       this.colors.color = 'red'
17     }
18   }
19 }
20 </script>
```



V-Bind เปลี่ยนย่อ



```
1 <h1 v-bind:style="color">Hello World!</h1>
```



```
1 <h1 :style="color">Hello World!</h1>
```



V-Bind Result

Hello World!

change color



v-if

v-if เป็นคำสั่งที่ใช้สร้างเงื่อนไขในการแสดงผล HTML โดยถ้าเงื่อนไขเป็นจริง (True) จะแสดง Element นั้นออกมา ถ้าเงื่อนไขเป็นเท็จ (False) จะซ่อน Element นั้นไว้ ไม่แสดงออกมา



V-if



```
1 <!-- vue html -->
2 <template>
3   <p v-if="isMe">I am {{ message }}</p>
4   <p v-else>Who am I?</p>
5 </template>
6 <!-- vue js -->
7 <script>
8 export default{
9   data() {
10     return {
11       message: 'A-Host',
12       isMe: true
13     }
14   }
15 }
16 </script>
```



V-for

v-for เป็นคำสั่งที่ใช้ในการ render Element หรือ template หลายๆ ครั้ง ตามจำนวนของ data ที่เราใช้ในการวน loop



V-for

```
1 <!-- vue html -->
2 <template>
3   <ul>
4     <li v-for="color in colors" :key="color.id">{{ color.name }}</li>
5   </ul>
6 </template>
7 <!-- vue js -->
8 <script>
9 export default{
10   data() {
11     return {
12       colors: [
13         { id: 1, name: 'Red' },
14         { id: 2, name: 'Green' },
15         { id: 3, name: 'Blue' },
16         { id: 4, name: 'Orange' },
17       ]
18     }
19   }
20 }
21 </script>
```



V-for Result

- Red
- Green
- Blue
- Orange



Props

คือ การระบุว่า Component นั้น ๆ สามารถรับค่าอะไรได้บ้าง เป็นวิธีการในการส่งค่าจาก Component แม่ ไปยัง Component ลูก เช่น เราสามารถส่งค่าจาก Component A (แม่) ไป Component B (ลูก) ด้วยการส่งค่าผ่าน Props นั้นเอง โดย Component B ต้องระบุ Props ด้วย



Props

A

```
1 <!-- vue html -->
2 <template>
3   <test message="Hi!"></test>
4 </template>
5 <!-- vue js -->
6 <script>
7 import Test from './components/Test.vue'
8 export default{
9   components:{ Test }
10 }
11 </script>
```

B

```
1 <!-- vue html -->
2 <template>
3   <h1>{{ message }}</h1>
4 </template>
5 <!-- vue js -->
6 <script>
7 export default{
8   name: 'Test',
9   //props: ['message']
10  props: {
11    message: String
12  }
13 }
14 </script>
```



slot

slot คือการส่งค่าที่เป็นลักษณะ content ไปยัง component อื่น แทนการ
เรียกใช้ props

เช่น หากเราต้องการทำ component ที่เป็น alert box ขึ้นมาเราจะไม่สามารถ
ส่งค่าลงไปใน props ได้เพราะบางทีมันก็จะยาวมากเกินไปเนื่องจากมี html
เข้ามา เราจึงต้องไปใช้วิธีการที่เรียกว่า "Slot Component"



Slot

```
<template>
<div class="card">
<slot>Default Card</slot>
</div>
</template>
<script>
export default {
  name: "card",
};
</script>
<style>
.card {
  box-shadow: 0 4px 8px 0 rgba(0, 0, 0,
0.2);
  transition: 0.3s;
  padding: 16px;
  margin-bottom: 20px;
  width: 300px;
}
.card:hover {
  box-shadow: 0 8px 16px 0 rgba(0, 0, 0,
0.2);
}
</style>
```



Axios

Axios คือ Open Source Javascript Library สำหรับเขียน
http request หรือก็คือตัวที่ใช้ในการเชื่อมต่อ API (Application
Programming Interface)



Axios : function

function GET - เป็น method ที่ใช้ select หรือดึงข้อมูล
`axios.get("URI")`

function POST - เป็น method ที่ใช้ insert ข้อมูล
`axios.post("URI", { data })`

function PUT - เป็น method ที่ใช้ update ข้อมูล
`axios.put("URI/:id", { data })`

function DELETE - เป็น method ที่ใช้ delete ข้อมูล
`axios.delete("URI/:id")`

Workshop: Demo

Manufacturer	Devices
	Acer ENDURO Urban N3
	SWIFT 3 AMD
	Swift 7
	Swift 5
	ASPIRE VERO
	Nitro 7
	Nitro 5
	Asus L410
	Asus ZenBook
	Asus Chromebook
	Asus ROG Zephyrus M16
	Asus ROG Zephyrus G14
	Asus ROG Flow X13
	Asus ProArt StudioBook 15
	Inspiron 15
	Dell Vostro 14
	Dell Latitude 7420
	Dell Latitude 9520



Device ID : 1

Acer ENDURO Urban N3
Intel Core i5-1135G7 (2.40 GHz)
Price : \$1799.99

Lab : Workshop 1

- ပေါ် Tailwind Css
 - Copy link CDN ပြောသောပတ်ဝန်းကျင် public ဖိုင် index.html ခေါ်ဝေါ်ရန် <head>

```
<link href "https://unpkg.com/tailwindcss@^2/dist/tailwind.min.css" rel "stylesheet" />
```

- ဖိုင် Component
 - ဖိုင် Manufacturer.vue ပြောသောပတ်ဝန်းကျင် components
 - ဖိုင် Devices.vue ပြောသောပတ်ဝန်းကျင် components

Lab : Workshop 2

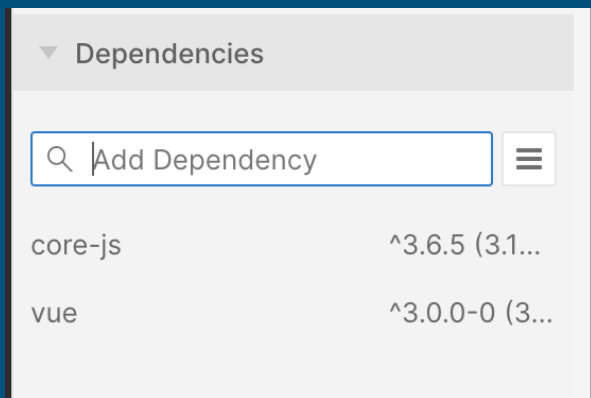
- สร้างข้อมูล Mockup
 - สร้างข้อมูล Mockup สำหรับ Manufacturer
 - สร้างข้อมูล Mockup สำหรับ Devices

<https://gist.github.com/mrbrooklyn/1689dc8f711a7f5e6631f1ee75531e11>

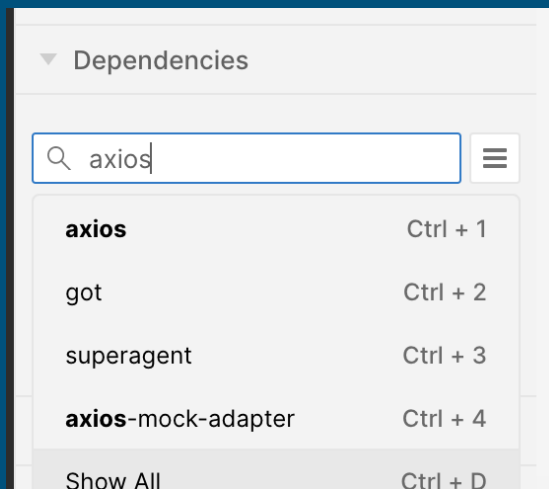
Lab : Workshop 3-1

- ติดตั้ง axios

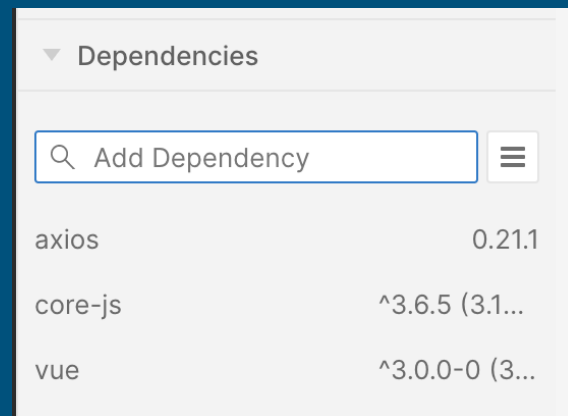
1



2



3



Lab : Workshop 3-2

- ดึงข้อมูล Manufacturer, Devices Data
 - API Manufacturer: [YOUR_API]/manufacturers
 - API Devices: [YOUR_API]/devices

<https://gist.github.com/mrbrooklyn/5a6c62d3d6f2886c8e2554187cf3b6f7>

Lab : Workshop 4

- ดึงข้อมูลรูปใส่ Manufacturer
 - โหลดไฟล์รูปเข้า Folder : assets

< <https://drive.google.com/drive/folders/1I1ff0HtkGlelBPdc8nIWGintzfh09xTb?usp=sharing> >

<https://gist.github.com/mrbrooklyn/35b0ef32a2d64114895f12bbbba819bc>

Lab : Workshop 5

- ปรับแต่งการแสดงผล
 - ปรับแต่ง Manufacturer Component
 - ปรับแต่ง Devices Component

<https://gist.github.com/mrbrooklyn/f5661f4babe53e7e45c5817f140ecfbd>

Lab : Workshop 6

- หาค่า ManufacturerID
 - ประกาศตัวแปรและเขียน function รับค่าของ Manufacturer ที่ถูกคลิกอยู่

<https://gist.github.com/mrbrooklyn/2f9c7a77e99d5e5047b710faa3fa69ac>

Lab : Workshop 7

- กำหนด props ที่ Devices.vue เพื่อรับค่า
 - ประกาศตัวแปร props ที่ Devices.vue
 - ทดสอบส่งค่า props ไปที่ Devices.vue

<https://gist.github.com/mrbrooklyn/04875c61f0afecf46196279edd2b6de6>

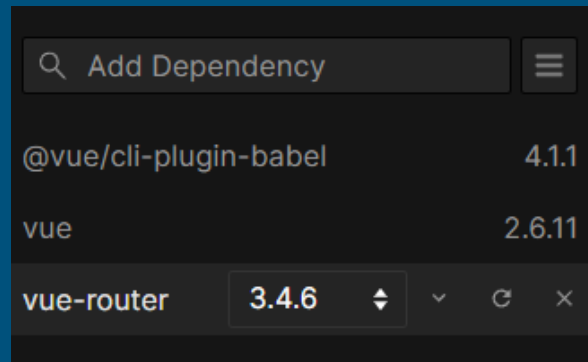
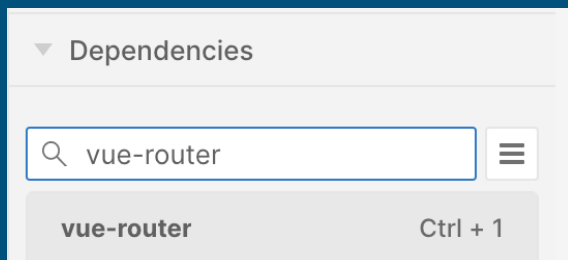
Lab : Workshop 8

- ทำให้ Manufacturer.vue ส่ง ID ให้ Devices.vue
 - สร้าง Slot ที่ Manufacturer.vue และส่ง parameter
 - ทำการ watch ID ที่ส่งเข้ามาใน Devices.vue

<https://gist.github.com/mrbrooklyn/37113dd225eb827bee1fa0cefd567f35>

Lab : Workshop 9

- ติดตั้ง Vue-Router
 - เพิ่ม Dependencies ที่ชื่อว่า vue-router
 - เลือก version เป็น 3.4.6



Lab : Workshop 10

- สร้าง Router
 - สร้างไฟล์เตอร์ pages และไฟล์ main.vue, detail.vue
 - สร้างไฟล์ router.js
 - import router.js ที่ไฟล์ main.js

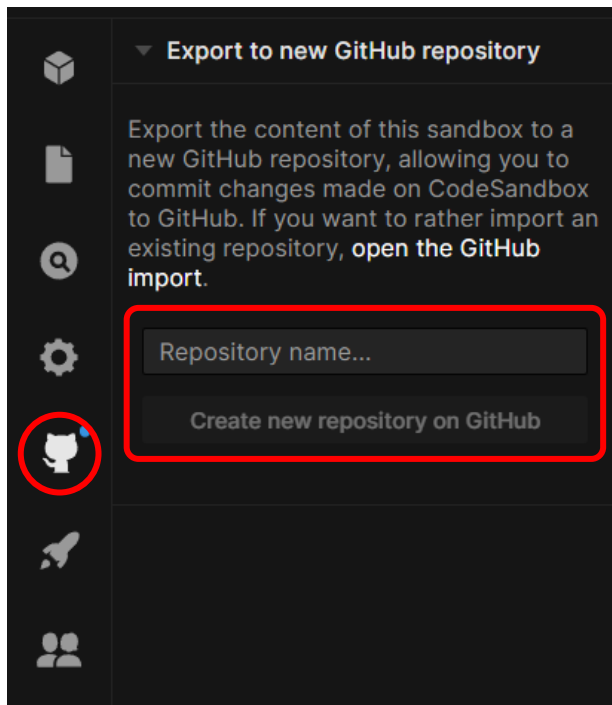
<https://gist.github.com/mrbrooklyn/2df18d0ed65641b251d7d06ac5ce145d>

Lab : Workshop 11

- เพิ่มหน้า Detail แสดง Detail ของ Device
 - เพิ่ม Route ใหม่สำหรับ Detail ที่ router.js
 - ปรับแต่งหน้า Detail
 - เพิ่มปุ่ม <router-link> ให้กดที่ Device แล้วไปหน้า Detail

<https://gist.github.com/mrbrooklyn/e18b37966d0dd4429283740f14a2c6a2>

การเอา Code ขึ้น GitHub

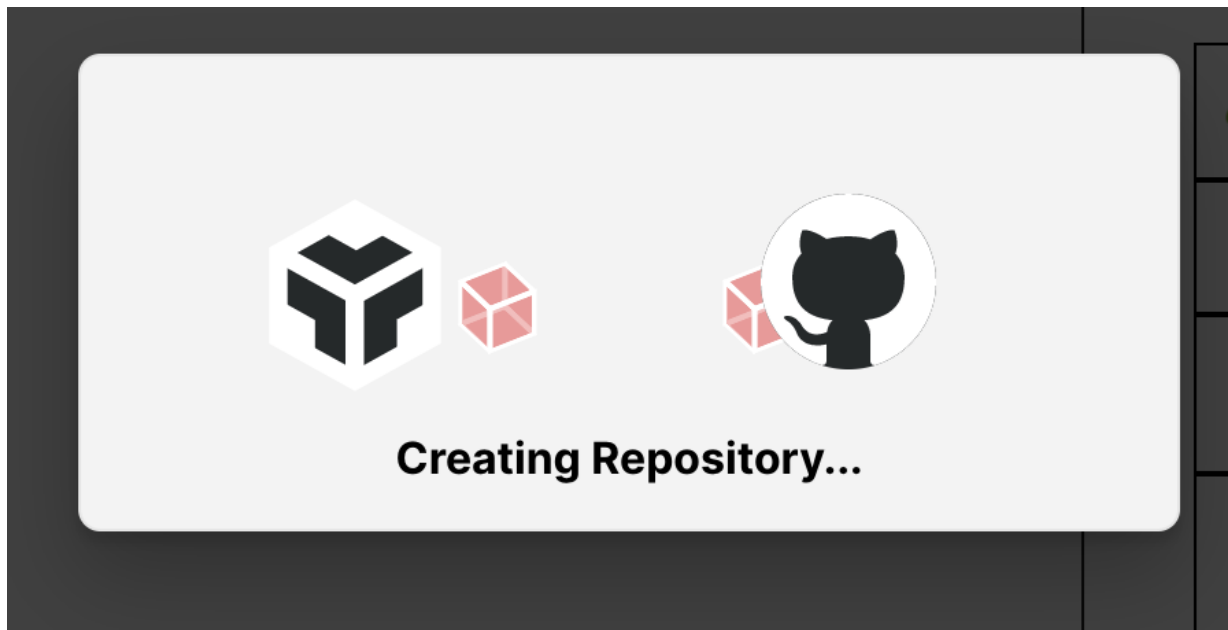


1. คลิกไปที่ icon GitHub (รูปแมว)
2. ตั้งชื่อ Repository โดยตั้งเป็น

(ชื่อตัวเอง)_VueLab

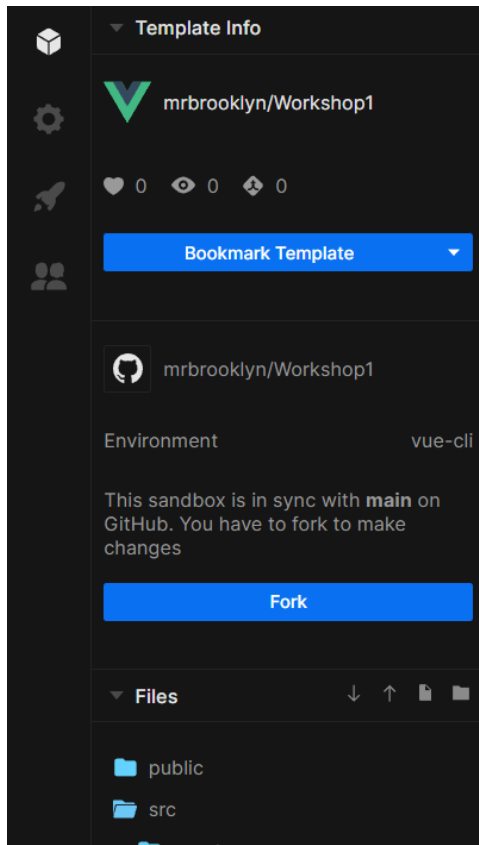
3. คลิกปุ่ม Create new repository on GitHub

การเอา Code ขึ้น GitHub



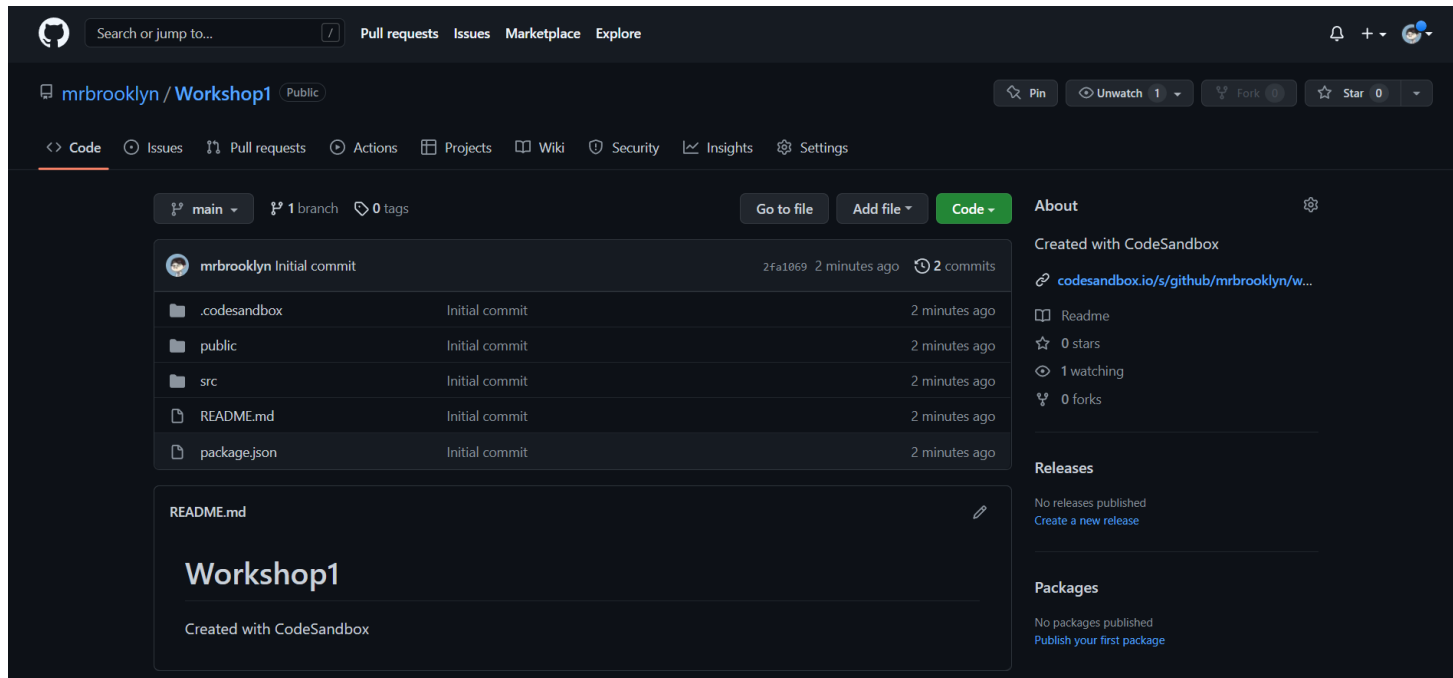
รอสักครู่

การเอา Code ขึ้น GitHub



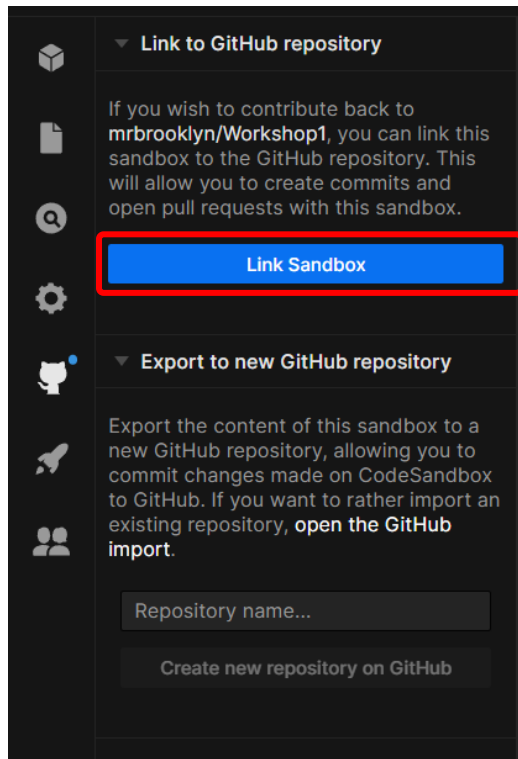
ถ้าขึ้นแบบนี้แปลว่าเสร็จแล้ว สามารถ
ไปตรวจสอบดู Repository ได้ที่
GitHub ของตัวเอง

การเอา Code ขึ้น GitHub



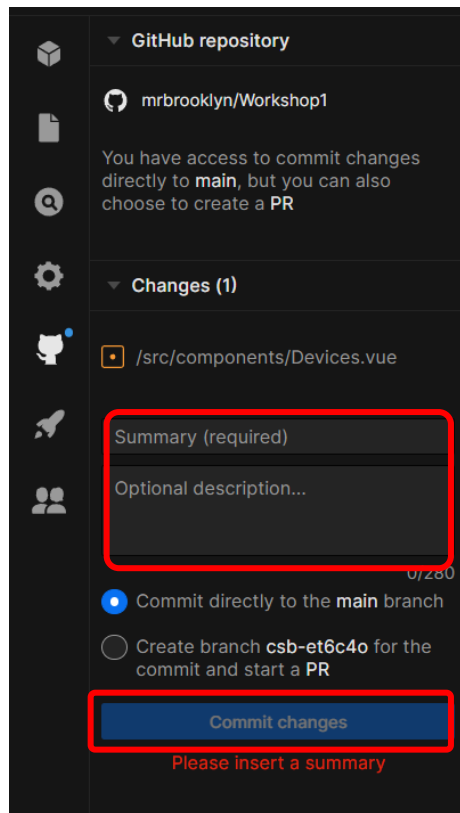
ถ้าได้โปรเจกแบบนี่แล้วให้ส่งงานได้ใน Link Slide ส่ง Lab

การแก้ไข Project ที่เอาขึ้น GitHub ไปแล้ว



คลิกปุ่ม Link Sandbox มันจะทำการ Connect กับ GitHub Repository ของเราที่สร้างไปแล้วให้ จากนั้นก็สามารถแก้ไขไฟล์ที่ต้องการได้เลย

การแก้ไข Project ที่เอาขึ้น GitHub ไปแล้ว



เมื่อแก้ไขไฟล์เสร็จแล้ว เมื่อกลับมาที่แถบ GitHub (รูปแบบ) จะพบว่า มีชื่อไฟล์ที่เราแก้ไขปรากฏขึ้น ให้ทำการใส่ Text Commit ว่าทำการแก้ไขอะไรไปบ้าง เสร็จแล้วค่อยกด Commit changes เท่านั้นเป็นอันเสร็จสิ้นการแก้ไข สามารถดูสิ่งที่แก้ไขได้ที่ GitHub Repository เลย

đv Lab

https://docs.google.com/spreadsheets/d/1MZrX0lwQpgC_gtXnxlOXvr1gpsZKUPUg7pXM1nxo7Vc/edit?usp=sharing