

EXERCISES QUE TRABAJAREMOS EN EL CUE:

- EXERCISE 1: PROP DENTRO DE COMPONENTE SEGÚN RUTA.
- EXERCISE 2: PROPS COMO OBJETO DESDE RUTA.
- EXERCISE 3: PROPS COMO FUNCIÓN DE PARÁMETROS DE RUTA.

EXERCISE 1: PROP DENTRO DE COMPONENTE SEGÚN RUTA

Para seguir aprendiendo características de vue-router, crearemos un proyecto con vue/cli, al cual debemos indicar que usaremos router, tal como se hizo en el CUE anterior.

En algunas ocasiones, cuando utilizamos rutas, necesitamos que éstas sean dinámicas, para poder obtener datos que nos sirvan al construir lógica, o hacer una llamada hacia una API. Es aquí, donde podemos presentar cierta rigidez en nuestro componente al utilizar `$route.params`, pues esto **nos hace dependientes de una URL específica**, lo que provocará que éste solo se pueda utilizar en esa ruta.

En el siguiente ejercicio, revisaremos como a través de props, podemos solucionar este inconveniente.

Luego de tener instalado el proyecto, vamos a editar la vista "`Home.vue`", que está ubicada en la carpeta "views". En esta, eliminaremos todo el contenido, y crearemos nuestra estructura.

Home.vue debe quedar de la siguiente manera:

```
1 <template>
2
3 </template>
4
5 <script>
6 export default {
7
8 }
9 </script>
10 <style>
11 </style>
```

Aquí podemos eliminar el componente "`HelloWorld.vue`" de la carpeta en que se encuentra, ya que acabamos de borrar su referencia en Home.vue, y no lo utilizaremos.

Editando Home.vue

En la sección de **<template>**, vamos a crear un formulario para desarrollar un buscador. En este ejemplo, ocuparemos PokemonApi, para desplegar algunos datos.

```
1 <template>
2   <form>
3     <h1>Buscador</h1>
4     <label for="">Ingrese Id Pokemon (1-200)</label>
5     <input type="text" v-model="pokemon_id">
6     <button @click.prevent="search">Buscar</button>
7   </form>
8 </template>
```

Una vez creado el **<template>**, pasaremos a la parte de script. Ésta tendrá un dato y un método.

```
1 export default {
2   name: "Home-Component",
3   data: function() {
4     return {
5       pokemon_id: "",
6     }
7   },
8
9   methods: {
10    search() {
11      if(this.pokemon_id === "") return
12      this.$router.push(`/pokemon/${this.pokemon_id}`)
13    }
14  }
15 }
```

El método será el encargado de dirigirnos hacia la ruta dinámica */pokemon/:id*.

Ahora, en la sección style, escribiremos unos estilos para el formulario.

```
1 <style scoped>
2   form{
3     border:2px solid #42b983;
4     padding-bottom: 20px;
5     width:60%;
6     margin:0 auto;
7   }
8 </style>
```

Y Home.vue quedará de esta forma:

```
1 <template>
2   <form>
3     <h1>Buscador</h1>
4     <label for="">Ingrese Id Pokemon (1-200)</label>
5     <input type="text" v-model="pokemon_id">
6     <button @click.prevent="search">Buscar</button>
7   </form>
8 </template>
9 <script>
10 export default {
11   name:"Home-Component",
12   data:function(){
13     return {
14       pokemon_id:"",
15     }
16   },
17   methods:{
18     search(){
19       if(this.pokemon_id === "") return
20       this.$router.push(`/pokemon/${this.pokemon_id}`)
21     }
22   }
23 }
24 </script>
25
26 <style scoped>
27   form{
28     border:2px solid #42b983;
29     padding-bottom: 20px;
30     width:60%;
31     margin:0 auto;
32   }
33 </style>
```

Si guardamos y levantamos nuestra aplicación, se verá así:

Buscador

Ingrese Id Pokemon (1-200)

Instalando axios:

Como en el ejercicio haremos una petición a la API de Pokemon, necesitamos instalar Axios para desarrollarla. Para ello, abriremos el terminal de Visual Studio Code.

Y escribiremos: “npm install axios”.

```
alejandros-MacBook-Pro:cue9 bonilla$ npm install axios
+ axios@0.21.1
added 1 package from 1 contributor and audited 1304 packages in 15.039s

68 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

Vista Pokemon.vue:

En la carpeta “views”, crearemos el archivo Pokemon.vue y su estructura.

```
1 <template>
2
3 </template>
4
5 <script>
6 export default {
7
8 }
9 </script>
10 <style>
11
12 </style>
```

En la sección de **<template>** vamos a crear un div, el cual contendrá el nombre del pokemon, seguido de su imagen.

```

1 <template>
2   <div>
3     <h2>Pokemon: {{name}}</h2>
4     
5   </div>
6 </template>

```

Ahora, en la sección de script, vamos a import la librería axios, y luego en data, nos dirigiremos a referenciar los datos que enlazamos en el template, crearemos una prop para recibir la id, y finalmente desarrollaremos los métodos para poder hacer la petición a la API.

```

1 <script>
2 import axios from 'axios'
3 export default {
4   props: ['id'],
5   data: function() {
6     return {
7       name: "",
8       image: "",
9     }
10  },
11  methods: {
12    fetchPokemon() {
13      axios.get(`https://pokeapi.co/api/v2/pokemon/${this.id}`)
14        .then(resp => {
15          console.log(resp);
16          this.setData(resp.data);
17        })
18        .catch(error => {
19          console.log(error)
20        })
21    },
22    setData(data) {
23      this.name = data.name;
24      this.image = data.sprites.front_default;
25    }
26  },
27  created() {
28    this.fetchPokemon();
29  }
30 }
31 </script>

```

Una vez creado el archivo `Pokemon.vue`, debemos ir a la carpeta `router` y abrir `index.js`, para poder enlazar el componente a una ruta específica.

Pasando props dentro de la ruta

En primer lugar, debemos importar el componente `Pokemon.vue`, que se ubica en la carpeta `views`.

```
1 import Pokemon from '@views/Pokemon.vue'
```

Luego, dentro del Array `routes`, crearemos un nuevo objeto para la ruta de acceso al componente. Adicionalmente, agregaremos el atributo `props:true`, de esta manera estaremos pasando la variable `:id` como prop, al componente `Pokemon`.

```
1 {  
2   path: '/pokemon/:id',  
3   name: "Pokemon",  
4   component: Pokemon,  
5   props: true,  
6  
7 },
```

Finalmente, el archivo `index.js` quedaría:

```
1 import Vue from 'vue'  
2 import VueRouter from 'vue-router'  
3 import Home from '../views/Home.vue'  
4 import Pokemon from '@views/Pokemon.vue'  
5  
6 Vue.use(VueRouter)  
7 const routes = [  
8   {  
9     path: '/',  
10    name: 'Home',  
11    component: Home  
12  },  
13  {  
14    path: '/pokemon/:id',  
15    name: "Pokemon",
```

```
16   component: Pokemon,  
17   props:true,  
18  
19 },  
20 {  
21   path: '/about',  
22   name: 'About',  
23   // route level code-splitting  
24   // this generates a separate chunk (about.[hash].js) for  
25 this route  
26   // which is lazy-loaded when the route is visited.  
27   component: () => import(/* webpackChunkName: "about" */  
28 '../views/About.vue')  
29 }  
30 ]  
31 const router = new VueRouter({  
32   mode: 'history',  
33   base: process.env.BASE_URL,  
34   routes  
35 })  
36  
37 export default router
```

Si levantamos la aplicación, abriendo la consola de vsCode y agregando el comando: “npm run serve”, se vería así:

Buscador

Ingrese Id Pokemon (1-200)

← → ↺ ⓘ localhost:8080/pokemon/2

Pokemon: ivysaur



EXERCISE 2: PROPS COMO OBJETO DESDE RUTA

Para realizar el siguiente ejercicio, seguiremos trabajando en el proyecto que creamos anteriormente.

Ya aprendimos como podemos pasar props cuando tenemos una ruta dinámica, ahora, en este ejemplo, revisaremos qué hacer cuando tenemos una ruta estática, y necesitamos pasarlos a nuestra vista.

Vamos a crear una vista nueva dentro de la carpeta “views”, llamada “History.vue”. En primer lugar, crearemos la estructura básica:

```
1 <template>
2
3 </template>
4
5 <script>
6 export default {
7
8 }
9 </script>
10 <style></style>
```

Ahora, partiremos por la sección de style, donde agregaremos el atributo scoped, para que las clases creadas solo puedan ser aplicadas al componente History.vue. En ella, incluiremos dos clases.

```
1 <style scoped>
2   .normal{
3     border: 2px solid red;
4     width: 60%;
5     margin:0 auto;
6   }
7   .backGround{
8     border: 2px solid green;
9     width: 60%;
10    margin:0 auto;
11    background:grey;
12    color:blue;
13  }
14 </style>
```


Ya teniendo las clases CSS, procederemos a la sección de script de nuestro componente, y allí crearemos una props y dos datos.

```
1 <script>
2 export default {
3   name: 'History',
4   props: {
5     backStyle: {
6       type: Boolean,
7       default: false,
8     }
9   },
10  data: function() {
11    return {
12      backGround: 'backGround',
13      normal: 'normal',
14    }
15  }
16 </script>
```

La props backStyle nos servirá para poder asignar la clase backGround, o la clase normal según su valor.

Por último, editaremos la sección de template.

```
1 <template>
2   <div :class="[backStyle ? backGround : normal]">
3     <h1>Nuestra Historia</h1>
4     <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit.
5 Error amet cupiditate placeat blanditiis sit, inventore
6 repellendus laborum molestiae eum nihil exercitationem officiis,
7 cumque harum animi quas officia mollitia? Beatae, deserunt.
8 Culpa laborum accusantium ipsum nisi perspiciatis neque, saepe
9 impedit maiores numquam quidem provident repudiandae eum aperiam
10 rerum rem officiis at totam deserunt asperiores ut nihil?
11 Excepturi atque amet ullam laudantium!
12 Eligendi incidunt quis officiis vitae ut enim nemo consequatur
13 fugit laboriosam assumenda neque ducimus aliquam, consectetur eius
14 rem provident debitis commodi inventore placeat, omnis sit illum.
15 Consequatur nemo soluta quia?
16 Dicta, dolorem laborum voluptas distinctio officia hic eos
17 minus tempora iste voluptatum, magni consequatur maiores quis
18 excepturi obcaecati commodi ex. Explicabo debitis non odit nulla
19 iure, tenetur provident doloribus error!
20 Aliquid perspiciatis error facilis placeat, ratione nobis est
21 esse amet quod rem reprehenderit. Ipsam eaque assumenda vel, a
```

```

22 maxime natus deserunt obcaecati. Obcaecati magnam fugit veniam,
23 atque ipsam libero eveniet.
24     Eos commodi minus magni at unde libero expedita mollitia id
25 quia aliquam. Nemo ut, dolore repellat pariatum eligendi totam
26 mollitia possimus praesentium qui unde debitis, harum sunt, vero
27 molestiae id.
28     Tenetur perspiciatis similique harum suscipit nobis incidunt
29 non distinctio magnam praesentium mollitia velit vero inventore
30 cum quaerat, cupiditate sapiente. Nostrum tenetur dolores
31 voluptates. Magni doloribus suscipit aliquid laudantium!
32 Distinctio, aut.
33     Amet laboriosam obcaecati cumque deserunt quia, pariatum vero
34 impedit perferendis aliquid, veritatis quaerat fuga eveniet beatae
35 reprehenderit unde quos eum perspiciatis, qui ullam culpa soluta
36 voluptatibus modi rem nulla! Blanditiis.
37     Totam soluta neque maiores ipsam commodi fuga non magni
38 laborum sapiente expedita aliquam, tenetur voluptas praesentium
39 nemo vitae dolor assumenda dolore dolorum deleniti beatae quaerat,
40 quo perspiciatis architecto iste. At?
50     Obcaecati nemo aliquam porro reiciendis nihil. Vitae quisquam
51 iste distinctio blanditiis alias in nulla illum dignissimos
52 numquam velit, reiciendis magni, impedit ullam maiores dolores
53 voluptates cumque. Quo exercitationem dicta natus.</p>
54 </div>
55 </template>
  
```

Como podemos observar, el div principal está utilizando v-bind, para poder crear una condición de verdad. }En este caso, si backStyle es true, se aplicará background; en caso contrario, aplicaremos normal.

El componente History.vue debe quedar de la siguiente manera:

```

1 <template>
2   <div :class="[backStyle ? backGround : normal ]">
3     <h1>Nuestra Historia</h1>
4     <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit.
5 Error amet cupiditate placeat blanditiis sit, inventore
6 repellendus laborum molestiae eum nihil exercitationem officiis,
7 cumque harum animi quas officia mollitia? Beatae, deserunt.
8     Culpa laborum accusantium ipsum nisi perspiciatis neque, saepe
9 impedit maiores numquam quidem provident repudiandae eum aperiam
10 rerum rem officiis at totam deserunt asperiores ut nihil?
11 Excepturi atque amet ullam laudantium!
12     Eligendi incidunt quis officiis vitae ut enim nemo consequatur
13 fugit laboriosam assumenda neque ducimus aliquam, consectetur eius
14 rem provident debitis commodi inventore placeat, omnis sit illum.
15 Consequatur nemo soluta quia?
  
```



```
16 Dicta, dolorem laborum voluptas distinctio officia hic eos
17 minus tempora iste voluptatum, magni consequatur maiores quis
18 excepturi obcaecati commodi ex. Explicabo debitis non odit nulla
19 iure, tenetur provident doloribus error!
20 Aliquid perspiciatis error facilis placeat, ratione nobis est
21 esse amet quod rem reprehenderit. Ipsam eaque assumenda vel, a
22 maxime natus deserunt obcaecati. Obcaecati magnam fugit veniam,
23 atque ipsam libero eveniet.
24 Eos commodi minus magni at unde libero expedita mollitia id
25 quia aliquam. Nemo ut, dolore repellat pariatum eligendi totam
26 mollitia possimus praesentium qui unde debitis, harum sunt, vero
27 molestiae id.
28 Tenetur perspiciatis similique harum suscipit nobis incidunt
29 non distinctio magnam praesentium mollitia velit vero inventore
30 cum quaerat, cupiditate sapiente. Nostrum tenetur dolores
31 voluptates. Magni doloribus suscipit aliquid laudantium!
32 Distinctio, aut.
33 Amet laboriosam obcaecati cumque deserunt quia, pariatum vero
34 impedit perferendis aliquid, veritatis quaerat fuga eveniet beatae
35 reprehenderit unde quos eum perspiciatis, qui ullam culpa soluta
36 voluptatibus modi rem nulla! Blanditiis.
37 Totam soluta neque maiores ipsam commodi fuga non magni
38 laborum sapiente expedita aliquam, tenetur voluptas praesentium
39 nemo vitae dolor assumenda dolore dolorum deleniti beatae quaerat,
40 quo perspiciatis architecto iste. At?
41 Obcaecati nemo aliquam porro reiciendis nihil. Vitae quisquam
42 iste distinctio blanditiis alias in nulla illum dignissimos
43 numquam velit, reiciendis magni, impedit ullam maiores dolores
44 voluptates cumque. Quo exercitationem dicta natus.</p>
45 </div>
46 </template>
47
48 <script>
49 export default {
50   name: 'History',
51   props: {
52     backStyle: {
53       type: Boolean,
54       default: false,
55     }
56   },
57   data: function() {
58     return {
59       backGround: 'backGround',
60       normal: 'normal',
61     }
62   }
63 }
64 </script>
65
66 <style scoped>
67 .normal{
```

```
68     border: 2px solid red;
69     width: 60%;
70     margin: 0 auto;
71   }
72   .backGround{
73     border: 2px solid green;
74     width: 60%;
75     margin: 0 auto;
76     background: grey;
77     color: blue;
78   }
79 </style>
```

Editando Rutas:

Ya teniendo el componente creado, nos falta poder asignarle una ruta dentro de nuestra aplicación, y para ello, iremos a la carpeta router y abriremos el archivo index.js.

En primer lugar, importaremos el componente History.vue.

```
1 import History from '@views/History.vue'
```

Luego, dentro del Array routes, agregaremos un nuevo objeto.

```
1 {
2   path: '/history',
3   name: 'History',
4   component: History,
5   props: {
6     backStyle: false,
7   }
8 },
```

En este caso, estamos pasando una prop a través de un objeto. Esto es útil cuando trabajamos con rutas estáticas, que necesitan de alguna prop para funcionar.

El archivo index.js final, quedará así:

```
1 import Vue from 'vue'
2 import VueRouter from 'vue-router'
3 import Home from '../views/Home.vue'
4 import Pokemon from '@views/Pokemon.vue'
5 import History from '@views/History.vue'
6 Vue.use(VueRouter)
7 const routes = [
8   {
9     path: '/',
10    name: 'Home',
11    component: Home
12  },
13  {
14    path: '/pokemon/:id',
15    name: 'Pokemon',
16    component: Pokemon,
17    props: true,
18  },
19  {
20    path: '/history',
21    name: 'History',
22    component: History,
23    props: {
24      backStyle: false,
25    }
26  },
27  {
28    path: '/about',
29    name: 'About',
30    // route level code-splitting
31    // this generates a separate chunk (about.[hash].js) for this
32    route
33    // which is lazy-loaded when the route is visited.
34    component: () => import(/* webpackChunkName: "about" */
35    '../views/About.vue')
36  }
37 ]
38 const router = new VueRouter({
39   mode: 'history',
40   base: process.env.BASE_URL,
41   routes
42 })
43 export default router
```

Por último, editaremos App.vue, para crear un enlace con router-link.

```
1 <template>
2   <div id="app">
3     <div id="nav">
4       <router-link to="/">Home</router-link> |
5       <router-link to="/history">Historia</router-link> |
6       <router-link to="/about">About</router-link>
7     </div>
8     <router-view/>
9   </div>
10 </template>
11
12 <style>
13 #app {
14   font-family: Avenir, Helvetica, Arial, sans-serif;
15   -webkit-font-smoothing: antialiased;
16   -moz-osx-font-smoothing: grayscale;
17   text-align: center;
18   color: #2c3e50;
19 }
20
21 #nav {
22   padding: 30px;
23 }
24
25 #nav a {
26   font-weight: bold;
27   color: #2c3e50;
28 }
29
30 #nav a.router-link-exact-active {
31   color: #42b983;
32 }
33 </style>
```

Si levantamos la aplicación, veremos lo siguiente:

[Home](#) | [Historia](#) | [About](#)

Buscador

Ingresa Id Pokemon

Si hacemos clic en Historia, nos redireccionará a `"/history"`, y veremos:

[Home](#) | [Historia](#) | [About](#)

Nuestra Historia

Lorem ipsum dolor sit, amet consectetur adipiscing elit. Error amet cupiditate placeat blanditiis sit, inventore repellendus laborum molestiae eum nihil exercitationem officiis, cumque harum animi quas officia mollitia? Beatae, deserunt. Culpa laborum accusantium ipsum nisi perspiciatis neque, saepe impedit maiores numquam quidem provident repudiandae eum aperiam rerum rem officiis at totam deserunt asperiores ut nihil? Excepturi atque amet ullam laudantium! Eligendi incidunt quis officiis vitae ut enim nemo consequatur fugit laboriosam assumenda neque ducimus aliquam, consectetur eius rem provident debitis commodi inventore placeat, omnis sit illum. Consequatur nemo soluta quia? Dicta, dolorem laborum voluptas distinctio officia hic eos minus tempora iste voluptatum, magni consequatur maiores quis excepturi obcaecati commodi ex. Explicabo debitis non odit nulla iure, tenetur provident doloribus error! Aliquid perspiciatis error facilis placeat, ratione nobis est esse amet quod rem reprehenderit. Ipsam eaque assumenda vel, a maxime natus deserunt obcaecati. Obcaecati magnam fugit veniam, atque ipsam libero eveniet. Eos commodi minus magni at unde libero expedita mollitia id quia aliquam. Nemo ut, dolore repellat pariatum eligendi totam mollitia possimus praesentium qui unde debitis, harum sunt, vero molestiae id. Tenetur perspiciatis similique harum suscipit nobis incidunt non distinctio magnam praesentium mollitia velit vero inventore cum quaerat, cupiditate sapiente. Nostrum tenetur dolores voluptates. Magni doloribus suscipit aliquid laudantium! Distinctio, aut. Amet laboriosam obcaecati cumque deserunt quia, pariatum vero impedit perferendis aliquid, veritatis quaerat fuga eveniet beatae reprehenderit unde quos eum perspiciatis, qui ullam culpa soluta voluptatibus modi rem nulla! Blanditiis. Totam soluta neque maiores ipsam commodi fuga non magni laborum sapiente expedita aliquam, tenetur voluptas praesentium nemo vitae dolor assumenda dolore dolorum deleniti beatae quaerat, quo perspiciatis architecto iste. At? Obcaecati nemo aliquam porro reiciendis nihil. Vitae quisquam iste distinctio blanditiis alias in nulla illum dignissimos numquam velit, reiciendis magni, impedit ullam maiores dolores voluptates cumque. Quo exercitationem dicta natus.

Si observamos bien, en este caso se está aplicando la clase `"normal"`; si vamos a `index.js` de `router`, y editamos la prop `"backStyle"`, a `true`.

```
1 {  
2   path: '/history',  
3   name: 'History',  
4   component: History,  
5   props: {  
6     backStyle: true,  
7   }  
8 },
```

Si guardamos y vamos a “/history”, veremos que se está aplicando la clase backGround.

[Home](#) | [Historia](#) | [About](#)

Nuestra Historia

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Error amet cupiditate placeat blanditiis sit, inventore repellendus laborum molestiae eum nihil exercitationem officiis, cumque harum animi quas officia mollitia? Beatae, deserunt. Culpa laborum accusantium ipsum nisi perspiciatis neque, saepe impedit maiores numquam quidem provident repudiandae eum aperiam rerum rem officiis at totam deserunt asperiores ut nihil? Excepturi atque amet ullam laudantium! Eligendi incidunt quis officiis vitae ut enim nemo consequatur fugit laboriosam assumenda neque ducimus aliquam, consectetur eius rem provident debitis commodi inventore placeat, omnis sit illum. Consequatur nemo soluta quia? Dicta, dolorem laborum voluptas distinctio officia hic eos minus tempora iste voluptatum, magni consequatur maiores quis excepturi obcaecati commodi ex. Explicabo debitis non odit nulla iure, tenetur provident doloribus error! Aliquid perspiciatis error facilis placeat, ratione nobis est esse amet quod rem reprehenderit. Ipsam eaque assumenda vel, a maxime natus deserunt obcaecati. Obcaecati magnam fugit veniam, atque ipsam libero eveniet. Eos commodi minus magni at unde libero expedita mollitia id quia aliquam. Nemo ut, dolore repellat pariatur eligendi totam mollitia possimus praesentium qui unde debitis, harum sunt, vero molestiae id. Tenetur perspiciatis similique harum suscipit nobis incidunt non distinctio magnam praesentium mollitia velit vero inventore cum quaerat, cupiditate sapiente. Nostrum tenetur dolores voluptates. Magni doloribus suscipit aliquid laudantium! Distinctio, aut. Amet laboriosam obcaecati cumque deserunt quia, pariatur vero impedit preferendis aliquid, veritatis quaerat fuga eveniet beatae reprehenderit unde quos eum perspiciatis, qui ullam culpa soluta voluptatibus modi rem nulla! Blanditiis. Totam soluta neque maiores ipsam commodi fuga non magni laborum sapiente expedita aliquam, tenetur voluptas praesentium nemo vitae dolor assumenda dolore dolorum deleniti beatae quaerat, quo perspiciatis architecto iste. At? Obcaecati nemo aliquam porro reiciendis nihil. Vitae quisquam iste distinctio blanditiis alias in nulla illum dignissimos numquam velit, reiciendis magni, impedit ullam maiores dolores voluptates cumque. Quo exercitationem dicta natus.

EXERCISE 3: PROPS COMO FUNCIÓN DE PARÁMETROS DE RUTA

Para realizar el siguiente ejercicio, continuaremos trabajando con el mismo proyecto que hemos utilizado hasta ahora.

En algunos casos, podemos necesitar modificar el valor dinámico de la URL, para entregárselo a una props. Para realizar esta tarea, implementaremos una función como props, y así lograremos hacer una pequeña modificación al valor dinámico.

Vamos a crear a una vista en la carpeta "views", llamada "**Client.vue**", y el archivo contendrá una prop, que nos servirá para mostrarlo en pantalla.

```
1 <template>
2   <div>
3     <h1>Bienvenido Cliente: {{client}} </h1>
4   </div>
5 </template>
6
7 <script>
8 export default {
9   name: 'Client-component',
10  props: ['client']
11 }
12 </script>
13
14 <style>
15
16 </style>
```

Una vez creado el archivo Client.vue, iremos a la carpeta router y abriremos el archivo index.js, para importar el componente creado.

Rutas:

Lo primero será importar el componente.

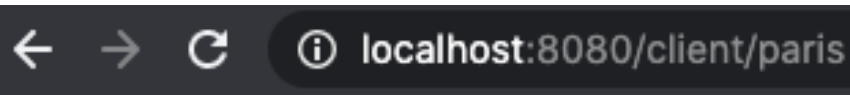
```
1 import Client from '@views/Client.vue'
```

Luego, dentro del Array routes, vamos a crear un nuevo objeto para asignar la ruta.

```
1 {  
2   path: '/client/:client',  
3   name: 'Client',  
4   component: Client,  
5   props: (route) => ({  
6     client: `${route.params.client} s.a`  
7   })  
8 },
```

En este caso, estamos tomando el parámetro “:client”, y lo concatenamos a la palabra s.a”.

De esta forma, si guardamos y vamos a la ruta, se verá así:



← → ↻ ⓘ localhost:8080/client/paris

Bienvenido Cliente: paris s.a