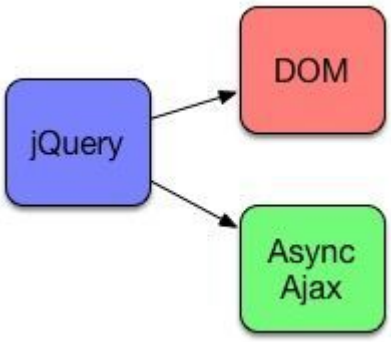
Axios js es una librería de JavaScript construida con el objetivo de gestionar la programación asíncrona con promesas. Mucha gente me preguntará ¿Por qué usar Axios y no usar jQuery que tiene una librería de promesas madura? . La pregunta más bien es ¿Por qué usar jQuery?. jQuery es la librería de Javascript de referencia a nivel de manipulación del arbol DOM . Pero su nivel de abstracción es básico, simplifica el manejo de DOM y gestiona las peticiones Ajax.



Hoy por hoy las opciones basadas en componentes como React.js están avanzando en JavaScript a gran velocidad. React no usa jQuery para gestionar la capa de presentación. Así pues nos encontramos con soluciones que no se apoyan en jQuery para gestionar la capa de presentación y el árbol DOM. Es necesario tener a nuestra disposición alguna librería de JavaScript que maneje las peticiones Ajax y promesas de forma independiente, Axios js es una de las opciones.

Un ejemplo con Axios js

Vamos a construir un ejemplo sencillo con Axiosjs . Supongamos que tenemos una aplicación de Node.js que se encarga de listar varios tipos de teléfonos:

```
var express = require('express');
```

```
var bodyParser = require('body-parser');
var app = express();
app.use(express.static('.'));
app.use(bodyParser.json());

var telefonos = [{
 nombre: "iphone",
 precio: "700"
}, {
 nombre: "android",
 precio: "500"
}];


app.get('/telefonos', function(req, res) {
 res.send(telefonos);
});



app.put('/telefonos/:nombre', function(req, res) {

 var indice = telefonos.findIndex(function(elemento) {

 return elemento.nombre == req.params.nombre;

 });

 telefonos[indice]=req.body;

 res.sendStatus(200);
```

```
});

app.listen(3000, function() {
 console.log('Example app listening on port 3000!');
});
```

En este caso tenemos una url /telefonos que nos devuelve una lista de teléfonos cuando la solicitamos vía GET. En cambio se realizamos una operación de PUT actualizaremos un teléfono. Vamos a usar Axiosjs para generar un par de botones en una página HTML que se encargue de listar y actualizar los datos por consola.

```
<html>

<head>
 <script src="https://unpkg.com/axios/dist/axios.min.js">ç
 </script>
 <script type="text/javascript">
 function peticionGet() {

 axios.get('/telefonos')
 .then(function(response) {
 console.log(response.data);
 })
 .catch(function(error) {
 console.log(error);
 });
 }


 function peticionPut() {
```

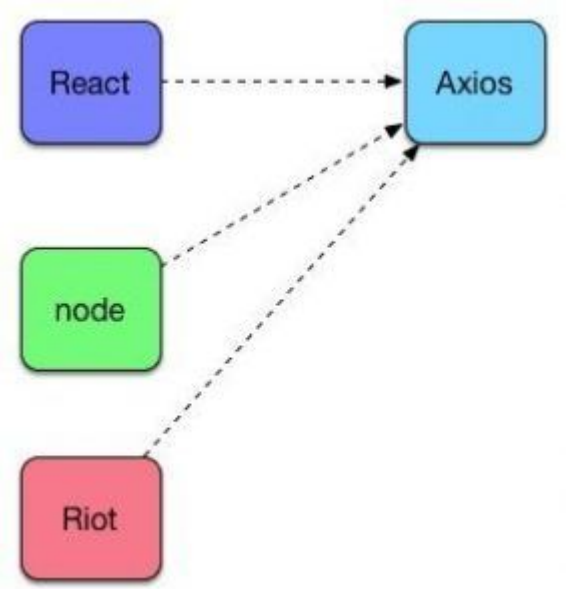
```
 axios.put('/telefonos/iphone', {
 nombre: "iphone",
 precio: "900"
 })
 .then(function(response) {
 console.log(response.data);
 })
 .catch(function(error) {
 console.log(error);
 });
 }
 </script>
</head>

<body>
 <input type="button" value="lista" onclick="peticionGet()" />
 <input type="button" value="actualizar" onclick="peticionPut()" />

</body>

</html>
```

En este caso hemos creado dos botones sencillos que se apoyan en axiosjs para realizar las peticiones Ajax con un API de promesas más sencilla que la de jQuery y sobre todo más independiente.



Cada día es mas habitual en el mundo de librerías de componentes de JavaScript apoyarse en otras librerías muy especializadas Axios es una muy interesante.

PD Muchas gracias al lector que me la recomendó echar un veo

Otros artículos relacionados: JavaScript Streams vs Promises , JavaScript Promise y la programación asíncrona , RxJS y la programación reactiva.