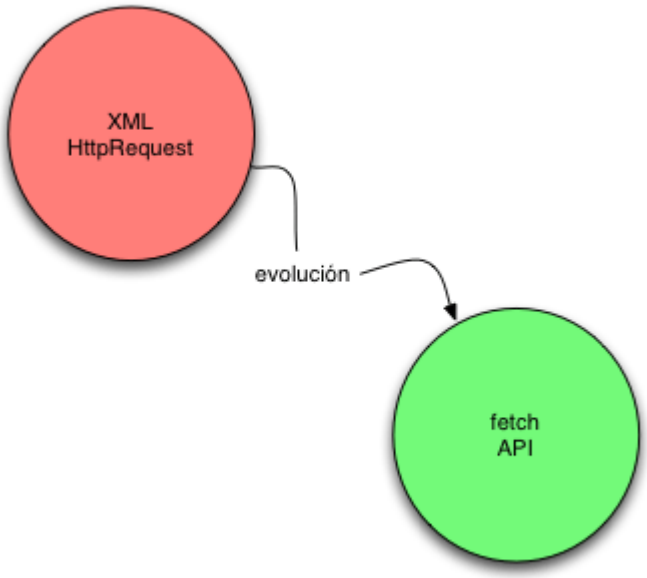
Hay que reconocer que JavaScript ES6 Fetch API es un avance a la hora de realizar peticiones AJAX importante comparado con las peticiones AJAX clásicas que eran insufribles. Hay que recordar que con JavaScript si uno quería hacer una petición AJAX el código era algo de este estilo:

```
function cargar() {
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
if (xhttp.readyState == 4 && xhttp.status == 200) {
.. blabla bla
}
};
xhttp.open("GET", "hola", true);
xhttp.send();
}
```

Todos terminamos usando el API de jQuery que era claramente superior y a través del uso de promesas aportaba una gran flexibilidad. El problema en algunos casos de este enfoque es que a veces nuestro proyecto no necesita obligatoriamente utilizar jQuery y lo acabamos añadiendo simplemente para simplificar peticiones AJAX o similar. ES6 nos ayuda a solventar este tipo de problema

JavaScript ES6 fetch API

A partir de JavaScript ES6 disponemos del API de fetch que permite hacer peticiones asíncronas de forma sencilla apoyándose en un API de promesas.

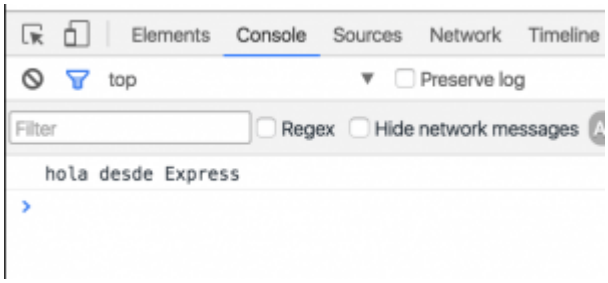


Vamos a verlo en código:

```
<html>
<head>
<script type="text/javascript" >

window.onload=function() {

 var boton=document.getElementById("botonAjax");

 boton.addEventListener("click",function() {


 fetch("/hola").then(function(respuesta) {
```

```
 return respuesta.text();


 }).then(function(texto) {

 console.log(texto);
 });


});
};
</script>

</head>

<body>
<input type="button" id="botonAjax" value="ajax"/ />
</body>

</html>
```

En este caso estamos haciendo una petición AJAX usando JavaScript ES6 Fetch API y su nueva sintaxis. Como podemos ver se trata de una simple instrucción de fetch y un encadenamiento de promesas. El resultado nos saldrá por la consola



Acabamos de configurar nuestra primera petición AJAX con JavaScript ES6 Fetch API , podemos tratar también de una forma natural las peticiones que incluyen información en formato JSON.

```
 fetch("/holaobjeto").then(function(respuesta) {

 return respuesta.json();

 }).then(function(objeto) {

console.log(objeto.nombre)
})
```

El soporte del API de fetch cada día es mayor aunque todavía hay algunos navegadores que tienen problemas con ello revisemos siempre la página de caniuse.

Otros artículos relacionados: JavaScript Promises , JavaScript Prototypes , JavaScript Memorization