JESUÏTES educació

DAWM06UF4
COMUNICACIÓ ASÍNCRONA CLIENT-SERVIDOR

UF4.2.3 FETCH API

CFGS Desenvolupament d'Aplicacions Web
M06. Desenvolupament web en entorn client
Fundació Jesuïtes Educació - Escola del Clot
Sergi Grau sergi.grau@fje.edu

OBJECTIUS

- Aprendre les característiques de les comunicacions asíncrones client/servidor

- Comprendre com funciona l'API FETCH

QUÈ ÉS FETCH ?

- L'API Fetch proporciona una interfície per accedir i manipular parts de canal HTTP, com ara peticions i respostes. També proveeix un mètode global fetch () que proporciona una forma fàcil i lògica d'obtenir recursos de manera asíncrona per la xarxa.

- Aquest tipus de funcionalitat s'aconseguia prèviament fent ús de XMLHttpRequest. Fetch proporciona una alternativa millor que pot ser emprada fàcilment per altres tecnologies com Service Workers. Fetch també aporta un únic lloc lògic en el qual definir altres conceptes relacionats amb HTTP com CORS i extensions per a HTTP.

QUÈ ÉS FETCH ?

- L'objecte Promise retornat des de fetch () no serà rebutjat amb un estat d'error HTTP fins i tot si la resposta és un error HTTP 404 o 500. En canvi, aquest es resoldrà normalment (amb un estat ok configurat a false), i aquest només serà rebutjat davant una fallada de xarxa o si alguna cosa va impedir completar la sol·licitud.

- Per defecte, fetch no enviarà ni rebrà galetes de servidor, resultant en peticions no autenticades si el lloc permet mantentener una sessió d'usuari (per enviar galetes, credentials de l'opció init hauran de ser configurades)

- El mètode fetch () pot acceptar opcionalment un segon paràmetre, un objecte init que permet controlar un nombre de diferents ajustos.

```
fetch('http://example.com/movies.json')
  .then(response => response.json())
  .then(data => console.log(data));
```

REALITZANT PETICIONS

- Una petició promise fetch () serà rebutjada amb TypeError quan es trobi un error de xarxa, encara que això normalment significa problemes de permisos o similars - per exemple, un 404 no constitueix un error de xarxa.

- Una forma precisa de comprovar que la petició fetch () és satisfactòria passa per comprovar si la promesa ha estat resolta, a més de comprovar que la propietat Response.ok té el valor true que indica que l'estat de la sol·licitud, també OK (codi 200 -299)

GESTIÓ DE RESPOSTES


```
fetch('flors.jpg').then(function(response) {
  if(response.ok) {
    response.blob().then(function(miBlob) {
      var objectURL = URL.createObjectURL(miBlob);
      miImagen.src = objectURL;
    });
  } else {
    console.log(‘xarxa OK pero resposta HTTP no OK');
  }
})
.catch(function(error) {
  console.log('problema amb la petició Fetch:' + error.message);
});
```

CREACIÓ DE LA PETICIÓ


```
var myHeaders = new Headers();

var myInit = { method: 'GET',
               headers: myHeaders,
               mode: 'cors',
               cache: 'default' };

var myRequest = new Request('flowers.jpg', myInit);

fetch(myRequest)
.then(function(response) {
  return response.blob();
})
.then(function(myBlob) {
  var objectURL = URL.createObjectURL(myBlob);
  myImage.src = objectURL;
});
```

ENVIAR UN ARXIU


```
var formData = new FormData();
var fileField = document.querySelector("input[type='file']");

formData.append('username', 'abc123');
formData.append('avatar', fileField.files[0]);

fetch('https://example.com/profile/avatar', {
  method: 'PUT',
  body: formData
})
.then(response => response.json())
.catch(error => console.error('Error:', error))
.then(response => console.log('Success:', response));
```

```
var content = "Hello World";
var capçalera = new Headers();
capçalera.append("Content-Type", "text/plain");
capçalera.append("Content-Length", content.length.toString());
capçalera.append("X-Custom-Header", "ProcessThisImmediately");
```

```
capçalera = new Headers({
  "Content-Type": "text/plain",
  "Content-Length": content.length.toString(),
  "X-Custom-Header": "ProcessThisImmediately",
});
```

BIBLIOGRAFIA

+ https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch