Visión general

¿qué vamos a ver?

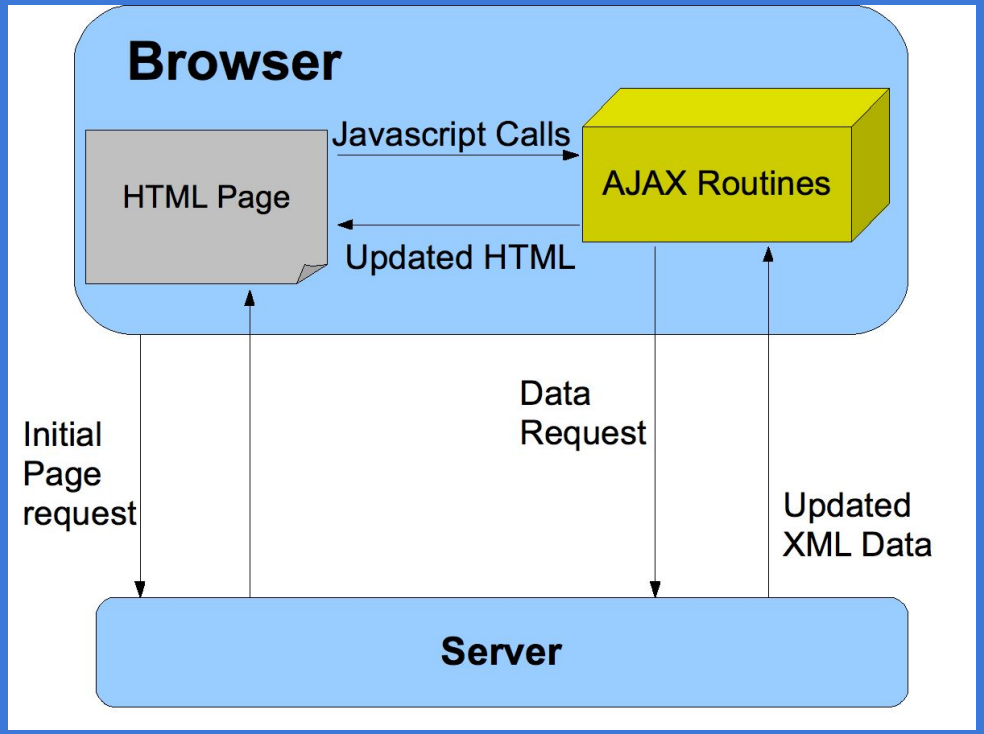
1.- Qué es Ajax

2.- XMLHttpRequest

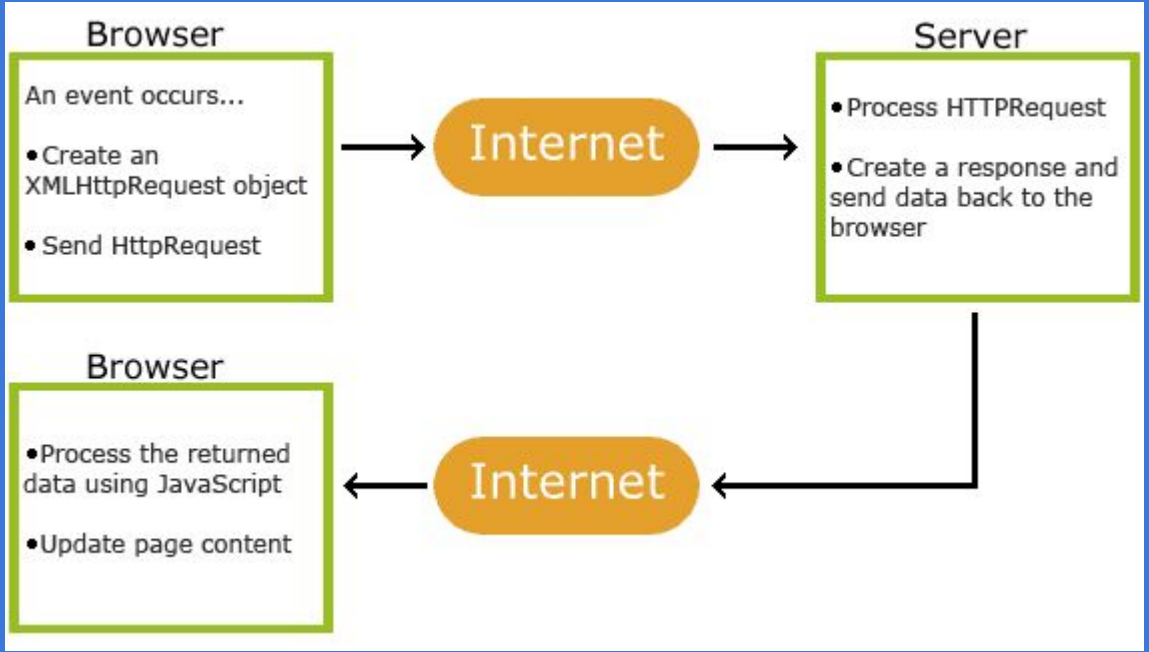
3.- Fetch

@kaixo_w0r1d

¿Qué es AJAX?


Browser
Javascript Calls
HTML Page
AJAX Routines
Updated HTML
Initial
Page
request
Data
Request
Updated
XML Data
Server

¿Qué es ajax?


Browser
An event occurs...
• Create an XMLHttpRequest object
• Send HttpRequest
Internet
Server
• Process HTTPRequest
• Create a response and send data back to the browser
Browser
• Process the returned data using JavaScript
• Update page content
Internet

XMLHTTPREQUEST. MÉTODOS

Method	Description
new XMLHttpRequest()	Creates a new XMLHttpRequest object
abort()	Cancels the current request
getAllResponseHeaders()	Returns header information
getResponseHeader()	Returns specific header information
open(*method,url,async,user,psw*)	Specifies the request *method*: the request type GET or POST *url*: the file location *async*: true (asynchronous) or false (synchronous) *user*: optional user name *psw*: optional password
send()	Sends the request to the server Used for GET requests
send(*string*)	Sends the request to the server. Used for POST requests
setRequestHeader()	Adds a label/value pair to the header to be sent

XMLHTTPREQUEST. ATRIBUTOS

Property	Description
onreadystatechange	Defines a function to be called when the readyState property changes
readyState	Holds the status of the XMLHttpRequest. 0: request not initialized 1: server connection established 2: request received 3: processing request 4: request finished and response is ready
responseText	Returns the response data as a string
responseXML	Returns the response data as XML data
status	Returns the status-number of a request 200: "OK" 403: "Forbidden" 404: "Not Found" For a complete list go to the Http Messages Reference
statusText	Returns the status-text (e.g. "OK" or "Not Found")

¡VAMOS A HACER UNOS EJEMPLOS!

- CON ARCHIVOS TXT
- CON ARCHIVOS XML
- CON ARCHIVOS JSON

FETCH

	fetch	axios	request
Interceptar solicitud y respuesta	✅	✅	✅
Transformar datos de solicitud y respuesta	✅	✅	✅
Cancelar solicitudes	✅	✅	❌
Transformaciones automáticas para datos JSON	✅	✅	✅
Soporte del lado del cliente para proteger contra XSRF	✅	✅	✅
Progreso	✅	✅	✅
Streaming	✅	✅	✅

FETCH

FLUJO BÁSICO

1. Realizas la **solicitud**.
2. Devuelve una **promesa**, que resuelve un objeto Response.
3. El objeto **response** es leído a través de funciones según el tipo de dato (json, blob, text, etc.).

FETCH. SINTAXIS BÁSICA


```
let response = fetch(url);
```

```
fetch(url)
    .then(response => {
        // handle the response
    })
    .catch(error => {
        // handle the error
    });
```

FETCH. SINTAXIS BÁSICA


```
fetch('/readme.txt')
    .then(response => response.text())
    .then(data => console.log(data));
```

FETCH

1. SOLICITUD

Podemos enviar una petición con la función **fetch()** pasándole como argumento la URL del endpoint al cual queremos enviar la petición.

A menos que se especifique lo contrario, de forma predeterminada, fetch enviará lo que se denomina una petición GET a la URL especificada, y responderá con una promesa.

FETCH

2. PROMESA

Una promesa representa el eventual cumplimiento o falla de un evento asincrónico.

FETCH

3. RESPONSE

Respuesta a la petición

@kaixo_w0r1d

¡VAMOS A HACER UNOS EJEMPLOS!