# Què és Redux?

Redux és una llibreria JavaScript de codi obert per al maneig de l’estat de les aplicacions. És comunament utilitzada amb altres llibreries com React o Angular per a la construcció d’interfícies d’usuari.

Al 2015 per Dan Abramov i Andrew Clark es van inspirar en una altra llibreria de Facebook, Flux per crear Redux. Abramov es trobava impressionat per la similitud del patró Flux amb la funció de reduir.

*"Estava pensant sobre Flux com una operació reduir... el teu magatzem, com acumulen un estat en resposta a unes accions. Vaig pensar a anar més enllà. Què si el teu magatzem Flux no fos un magatzem sinó una funció reduir?"*

Abramov va contactar amb Clark per col·laborar junts. Gràcies a Clark hi ha les eines que fan possible l'ecosistema Redux, va ajudar a definir un API coherent i implementar la possibilitat d’extensió mitjançant middleware i store enhancers.

Redux és una petita llibreria amb una API simple i limitada que está dissenyada per ser un contenidor predictible de l’estat de l’aplicació.

El propòsit de Redux és fer predictibles els canvis d'estat, imposant certes restriccions sobre com i quan es poden produir les actualitzacions. Redux aconsegueix que la teva gestió d'estat sigui transparent i determinista, cosa que entre altres coses aporta:

* Millor comprensió de l’evolució de l’estat en un moment donat.
* Facilitat per incorporar noves característiques a l’app.
* Nou ventall d’eines de debugging.
* Capacitat de reproduir un bug.
* Millores en el procés de desenvolupament i poden reiniciar l’execució a partir d’un estat concret.

## Avantatges del store de Redux

* Model de dades més consistent i segur
* Gestió senzilla de múltiples fonts de dades:
* Dades accessibles en temps real:

# El nostre projecte

## APIS Utilitzades

## Lògica de la aplicació

## Emmagatzematge de dades

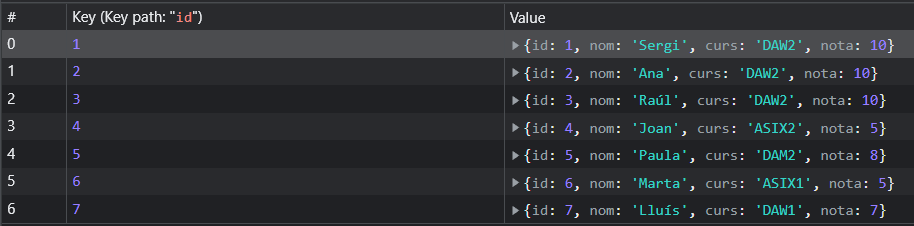
## HTML5 Apis

### [Web Storage](https://developer.mozilla.org/es/docs/Web/API/Web_Storage_API)

Hem utilitzat Web Storage per emmagazemar únicament el ID de l’alumne i el seu nom. Quan carrega la pàgina obté les dades de IndexedDB, cada vegada que afegim un usuari, es guarda el ID i el nom, quan eliminem un usuari, s’elimina també, es guarda mitjançant Local storage i no Session Storage. A més la utilitzem per a mostrar les dades per consola mitjançant una funció dinàmica que es veurà proximament.

### [IndexedDB](https://developer.mozilla.org/es/docs/Web/API/IndexedDB_API)

Hem utilitzat IndexedDB per a guardar els objectes dels alumnes. La key que utilitze es el id d’usuari. Al entrar a la pàgina, IndexedDB envia cada usuari al reducer de la api Redux i d’aquesta forma els carreguem en el estat, que es mostrat en la taula d’alumnes



### [Drag And Drop](https://developer.mozilla.org/es/docs/Web/API/HTML_Drag_and_Drop_API/File_drag_and_drop)

Hem utilitzat la API de drag and drop per a poder arrossegar arxius JSON, amb els quals podem afegir alumnes de forma massiva.

### [FileReader](https://developer.mozilla.org/es/docs/Web/API/FileReader)

Hem utilitzat la API FileReader per a processar les dades del fitxer.

## POO amb JS

Hem creat una classe emmagatzematge que conté dos mètodes estàtics. El mètode desar

Texto

Descripción generada automáticamente

També, hem creat una classe anomenada WebStorageEmmagatzematge que hereta de Emmagatzematge, aquesta compta amb un override dels dos mètodes de la superclasse.

Texto

Descripción generada automáticamente

## Funcions amb JS

## Arrays amb JS

### Métodes d’array utilitzats

map(),

filter()

sort()

splice()

forEach()

reduce() (Relatiu ja que es la funció reducer que uitlitza redux

for in