

# Progressive Web App PWA

**Intervenant : Jean-Frédéric VINCENT**





Module 4

# Indexed DB

Indexed DB

# Indexed DB



IndexedDB est une API qui permet le stockage côté client de quantités importantes de données structurées, incluant des fichiers.

Cette API utilise des index afin de permettre des recherches performantes sur ces données.

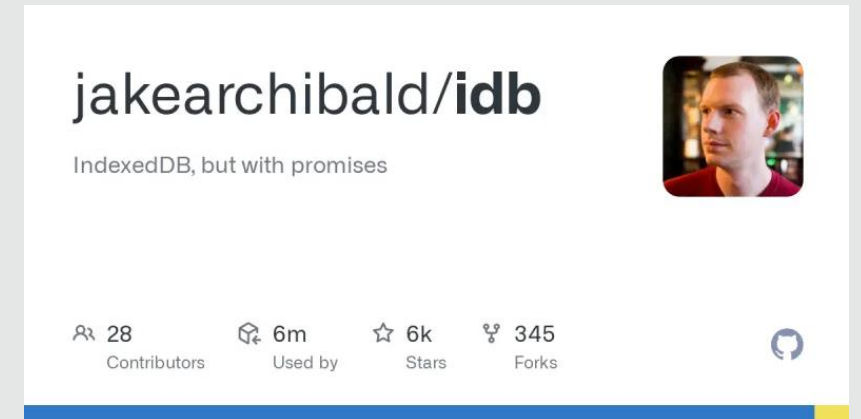
[https://developer.mozilla.org/fr/docs/Web/API/IndexedDB\\_API](https://developer.mozilla.org/fr/docs/Web/API/IndexedDB_API)

Indexed DB

# Prise en main d'une librairie simplifiée

Utilisation de idb de Jake Archibald

*En effet l'utilisation des fonctions natives est considérée comme « clunky »*  
et complexe.



<https://github.com/jakearchibald/idb>

Indexed DB

# Mise en place

Plutôt que d'utiliser le cdn, nous allons télécharger le fichier umd.js

<https://cdn.jsdelivr.net/npm/idb@8/build/umd.js>

# Création de la data base

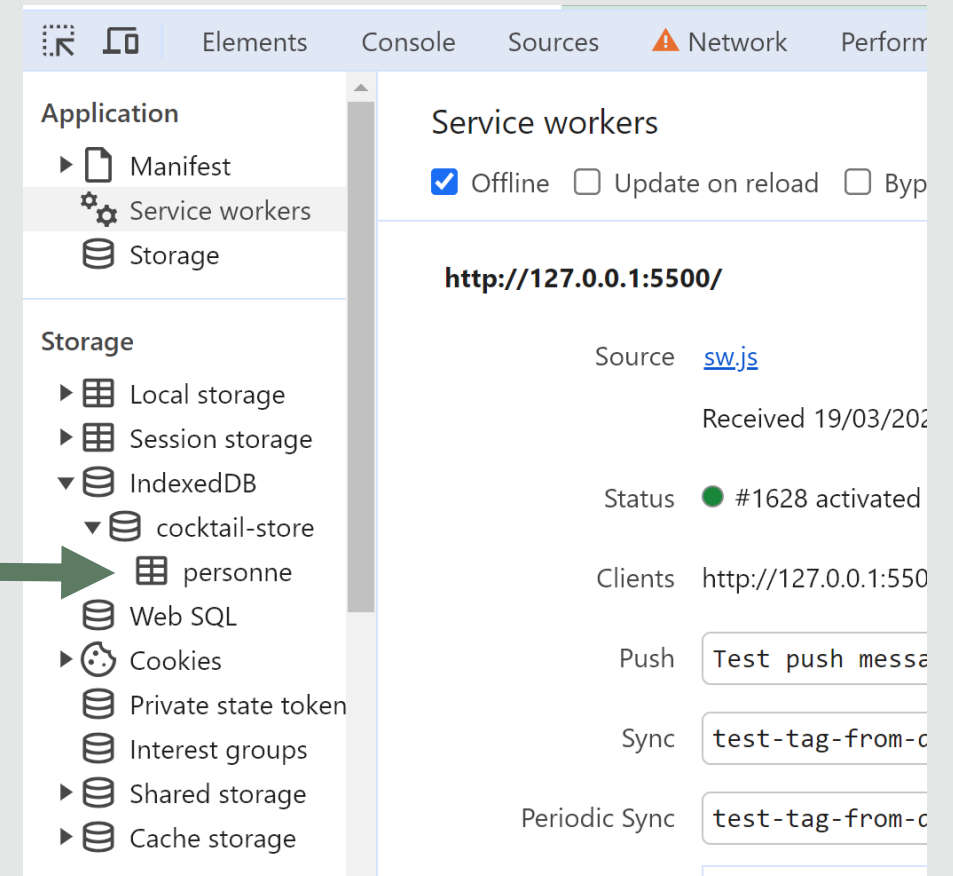
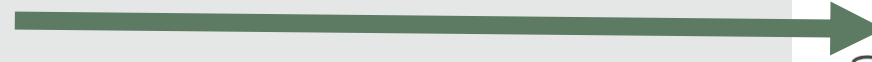
## Définition

- du nom de la base
- de la version
- de la table

```
// creation de la base donnée sur indexed DB
const dbPromise = await idb.openDB("cocktail-store", 1, {
  upgrade(db) {
    // Création de la table personne
    //const store = db.createObjectStore('personne', {keyPath: 'id', autoIncrement: true });
    const store = db.createObjectStore("personne", { keyPath: "id" });
    console.log("creation de la base");
  },
});
```

# Indexed DB Vérification

Dans dev tools > applications



# Insertion des données

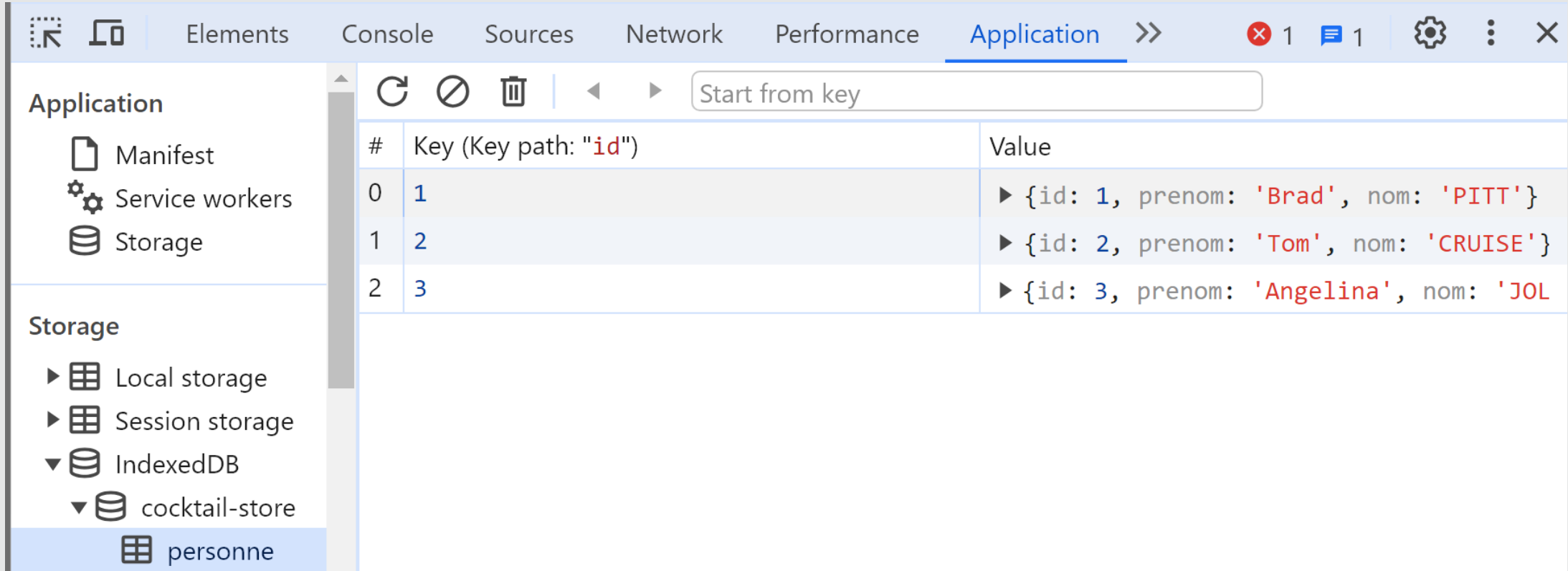
## Insertion de donnée en dure

```
const personnes = [  
  { id: 1, prenom: "Brad", nom: "PITT" },  
  { id: 2, prenom: "Tom", nom: "CRUISE" },  
  { id: 3, prenom: "Angelina", nom: "JOLIE" },  
];  
const tx = dbPromise.transaction("personne", "readwrite"); // readonly  
for (let p of personnes) {  
  await tx.store.put(p);  
}  
await tx.done;
```



# Insertion des données

## Vérification

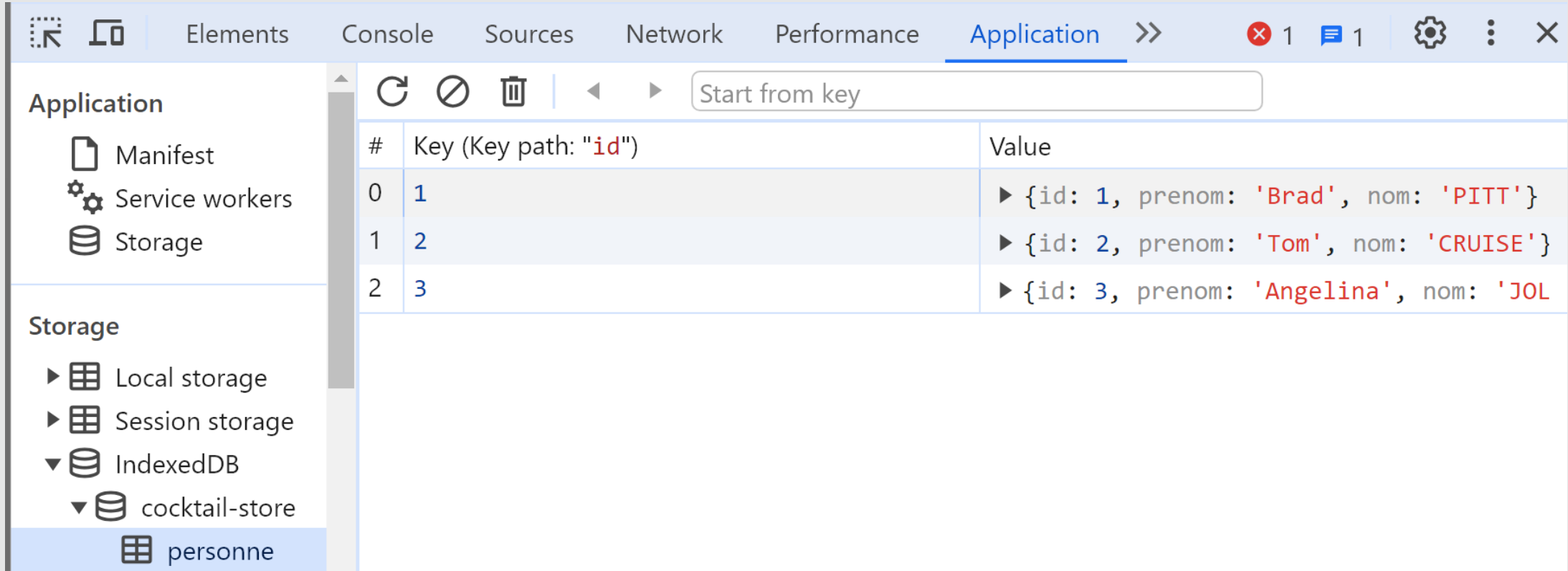


The screenshot shows the Chrome DevTools Application tab. The left sidebar has a tree view with 'Application' and 'Storage' sections. Under 'Storage', 'IndexedDB' is expanded, showing a database named 'cocktail-store'. Inside 'cocktail-store', a table named 'personne' is selected. The main pane displays the contents of the 'personne' table as a table with three columns: '#', 'Key (Key path: "id")', and 'Value'.

#	Key (Key path: "id")	Value
0	1	▶ {id: 1, prenom: 'Brad', nom: 'PITT'}
1	2	▶ {id: 2, prenom: 'Tom', nom: 'CRUISE'}
2	3	▶ {id: 3, prenom: 'Angelina', nom: 'JOL'}

# Insertion des données

## Vérification



The screenshot shows the Chrome DevTools Application tab. The left sidebar has a tree view with 'Application' and 'Storage' sections. Under 'Storage', 'IndexedDB' is expanded, showing a database named 'cocktail-store'. Inside 'cocktail-store', a table named 'personne' is selected. The main pane displays the contents of the 'personne' table as a table with three columns: '#', 'Key (Key path: "id")', and 'Value'.

#	Key (Key path: "id")	Value
0	1	▶ {id: 1, prenom: 'Brad', nom: 'PITT'}
1	2	▶ {id: 2, prenom: 'Tom', nom: 'CRUISE'}
2	3	▶ {id: 3, prenom: 'Angelina', nom: 'JOL'

Indexed DB

# Extractions des données

Equivalent de SELECT en SQL

```
const store = tx.objectStore("personne");  
const response = await store.getAll();  
for (let p of response) {  
  console.log(p);  
}
```

Indexed DB

# Vide la table

Equivalent de SELECT en SQL

```
const tx = dbPromise.transaction('personne', "readwrite");  
await tx.store.clear();  
return tx.complete;
```