Introducción a Firebase
DATABASE

DOCUMENTACIÓN

https://firebase.google.com/docs/database/

La Base de Datos de Firebase almacena información en una base de datos NoSQL, básicamente archivos con formato JSON; y permite que al crear aplicaciones cross-platform (disponibles en distintas plataformas), todos los clientes compartan una misma Base de Datos.

Tiempo Real

Cada vez que hay un cambio en la información, los clientes conectados reciben una actualización y se sincronizan en tiempo real.

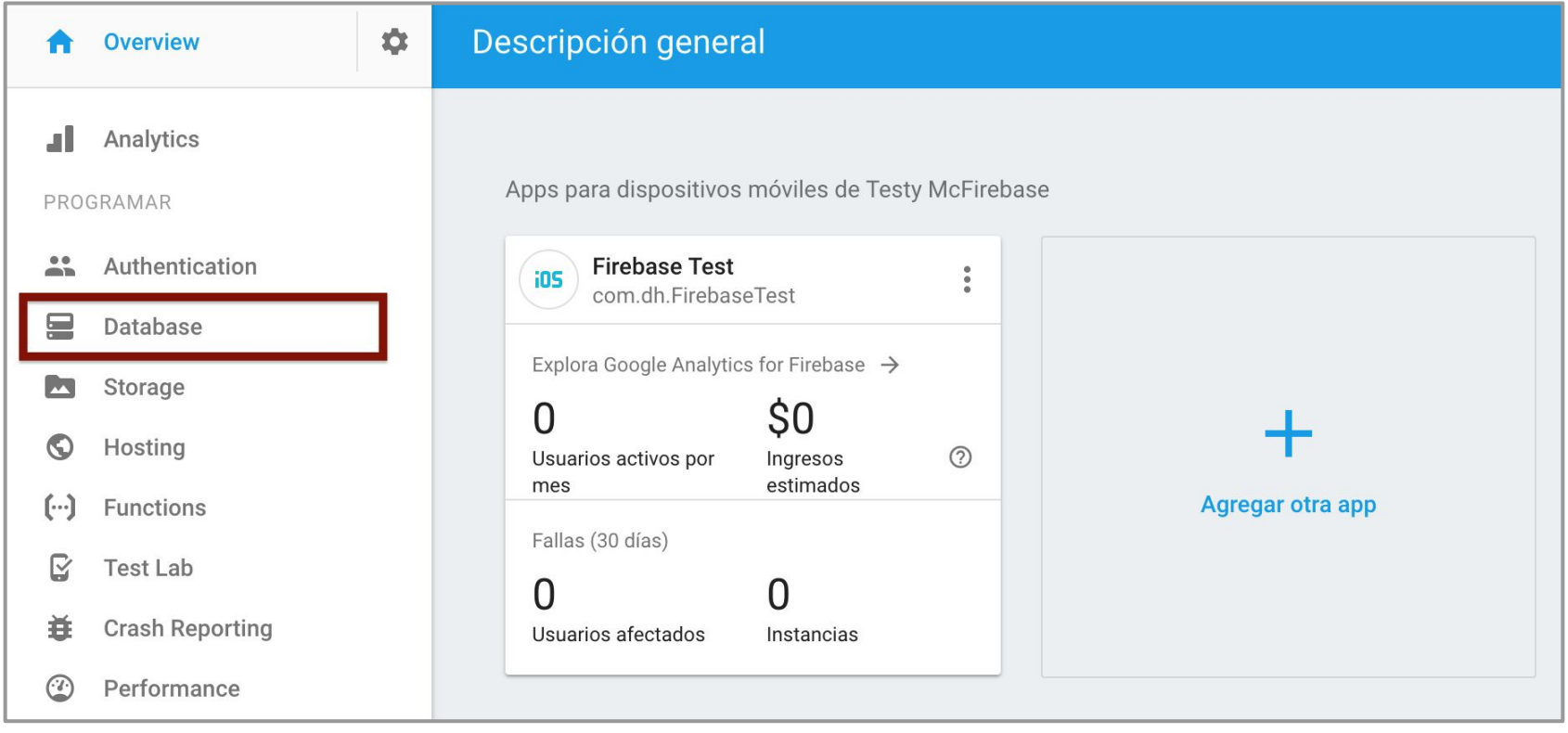
Offline

Esa información está siempre disponible ya que se encuentra almacenada remotamente; y se configurar para que la almacene localmente para acceder aún sin conexión a internet.

Accesible Desde Dispositivos Móviles

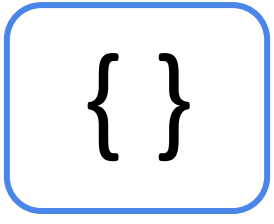
No hay necesidad de una aplicación en servidor, se puede acceder simplemente con un navegador de internet.

En Firebase Console

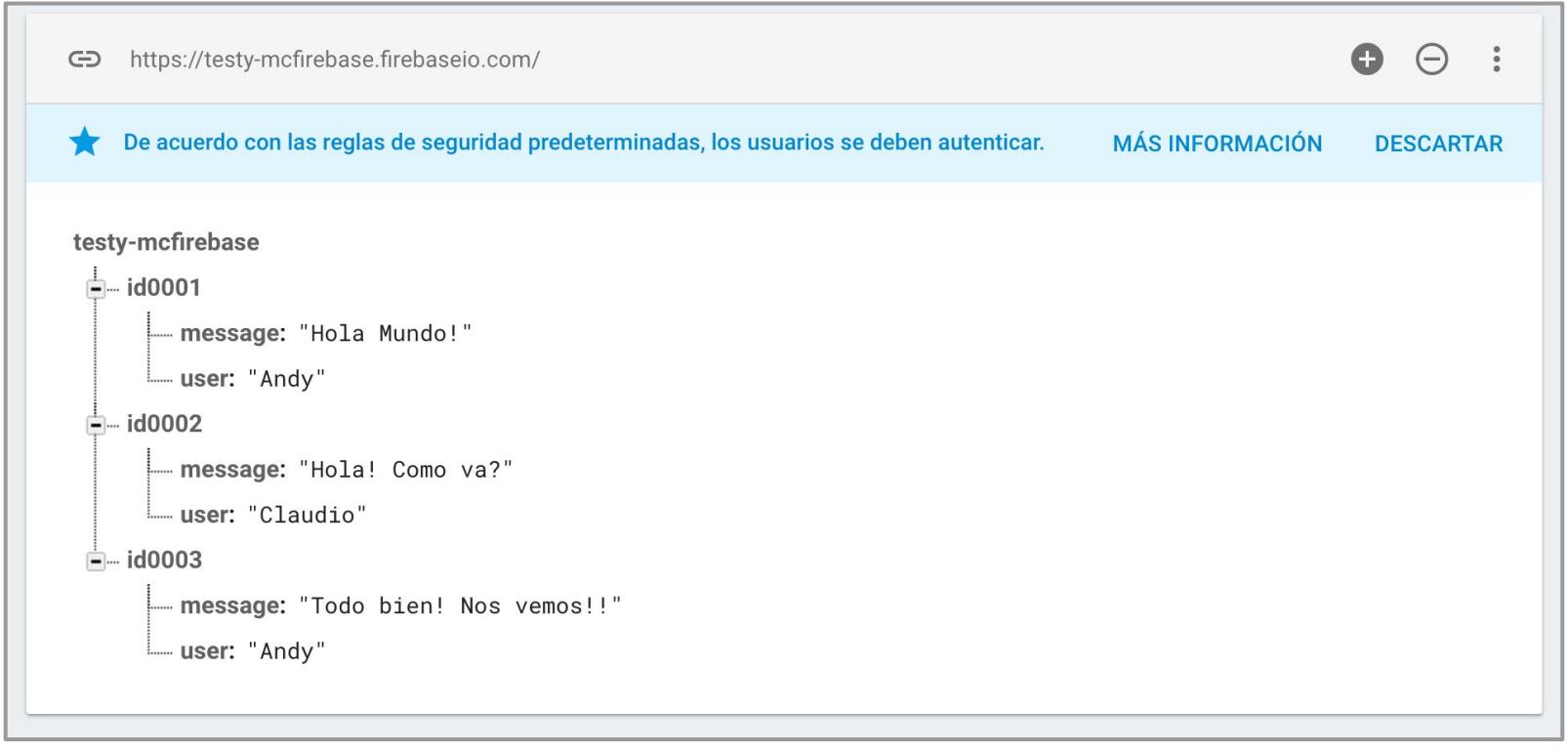


En Firebase Console

Base de Datos Inicial de Firebase



En Firebase Console



En Firebase Console

Base de Datos de Ejemplo

```
{
    "id0001": {
        "message": "Hola Mundo!",
        "user": "Andy"
    },
    "id0002": {
        "message": "Hola! Como va?",
        "user": "Claudio"
    },
    "id0003": {
        "message": "Todo bien!",
        "user": "Andy"
    }
}
```

Alternativa

```
{
    "messages": [
        "id0001": {
            "message": "Hola Mundo!",
            "user": "Andy"
        },
        "id0002": {
            "message": "Hola! Como va?",
            "user": "Claudio"
        },
        "id0003": {
            "message": "Todo bien!",
            "user": "Andy"
        }
    ]
}
```

INSTALACIÓN

```
pod 'Firebase/Database'
```

Firebase - Lectura de Base de Datos

LEER INFORMACIÓN

observeSingleEvent:ofType: es el método a usar cuando queremos obtener información una única vez; por ejemplo, cuando inicializamos una aplicación y queremos obtener el estado actual de la Base de Datos.

```
var ref = Database.database().reference()

ref.child("artists").observeSingleEvent(of: DataEventType.value, with: { (snapshot) in
    // Obtener todo el nodo 'artists' implica que vamos a recibir un array
    // y cada elemento del array va a ser un diccionario representando un 'Artist'
    if let dictionaryArray = snapshot.value as? [ [String: AnyObject] ] {
        var artistsArray: [Artist] = []
        for dict in dictionaryArray {
            if let theArtist = Artist(fromDictionary: dict) {
                artistsArray.append(theArtist)
            }
        }
        callback(artistsArray)
    }
}
```

LEER INFORMACIÓN

observe:ofType: es el método a usar cuando queremos obtener información por cada evento indicado en múltiples ocasiones. Si indicamos el evento 'childAdded', al especificar un callback continuo, la ejecución se va a repetir por cada nuevo elemento agregado al nodo indicado.

```
var ref = Database.database().reference()

ref.child("artists").observe(DataEventType.childAdded, with: { (snapshot) in
    // Obtener un evento 'childAdded' implica que el observer se va a ejecutar
    // con cada nuevo 'Artist' agregado a la Base de Datos; por ello sabemos que
    // el elemento que recibimos va a ser un diccionario representando un 'Artist'
    if let artistDict = snapshot.value as? [String : AnyObject] {
        let theArtist = Artist(fromDictionary: dict)
        callback(theArtist)
    }
})
```

Firebase - Escritura en Base de Datos

ESCRIBIR INFORMACIÓN

Para operaciones básicas de escritura se utiliza **`setValue(...)`** para guardar información en una referencia específica. Con este método se permite enviar tipos de datos que se pueden referenciar en un JSON: `NSString, NSNumber, NSDictionary, NSArray` .

Este método va a sobreescribir la referencia si ya existe.

```
var ref = Database.database().reference()
let artistDictionary = theArtist.toDictionary()

// Opción #1
ref.child("artists").childByAutoId().setValue(artistDictionary)
// El método childByAutoId() automáticamente genera un nuevo elemento en la referencia
// indicada ("artists"), con un ID generado internamente.
// Luego, se setean los valores asociados usando el diccionario 'artistDictionary'.

// Opción #2
ref.child("artists").child(theArtist.id).setValue(artistDictionary)
// El método child(id) automáticamente genera un nuevo elemento en la referencia indicada
// ("artists"), pero utilizando el ID propio del elemento.
```

ACTUALIZAR INFORMACIÓN

Para actualizar el valor del atributo de un elemento, no es necesario sobreescribir todo el elemento. Podemos simplemente actualizar un atributo, indicando la referencia correcta y usando **`setValue(...)`**.

```
var ref = Database.database().reference()
let artistDictionary = theArtist.toDictionary()

ref.child("artists/(theArtist.id)/username").setValue(theArtist.username)
// Se indica la referencia al atributo del elemento en cuestión, y luego en el método
// setValue sólo se indica el valor correspondiente (en lugar de todo un diccionario
// representando al elemento artista).
```

ELIMINAR INFORMACIÓN

Para eliminar un elemento de la base de datos, simplemente se utiliza el método **`removeValue()`** sobre la referencia del elemento correspondiente.

```
var ref = Database.database().reference()
let artistDictionary = theArtist.toDictionary()

// Opción #1
ref.child("artists/(theArtist.id)").removeValue()
// Se indica la referencia del elemento a eliminar, y luego se hace el remove.

// Opción #2
ref.child("artists/(theArtist.id)").setValue(nil)
// También se puede eliminar especificando 'nil' como el valor de una escritura.
```