Enfoques de Desarrollo de
Aplicaciones Móviles Multiplataforma

Bases de Datos

Contenido

Persistencia

Implementaciones de los sistemas operativos

Persistencia

- En los diferentes enfoques vistos hasta ahora se trabajó en general con datos que se mantenían en memoria
- Cuando la aplicación se cierra, la información en memoria se pierde

Persistencia

- Los sistemas operativos proveen distintas formas de almacenar datos a las aplicaciones, cada una con diferentes características
- Se debe analizar qué método es el más adecuado para utilizar según el problema que se desea resolver
- Tanto Android como iOS utilizan SQLite, por lo que es posible el uso del DBMS por parte de las aplicaciones

Android

Android

- El sistema proporciona varias opciones para almacenar datos de aplicación:
  - Almacenamiento específico de la app
  - Almacenamiento compartido
  - Preferencias
  - Bases de datos

Android - Almacenamiento específico de la app

- Se pueden crear archivos y directorios en un volúmen de almacenamiento interno o en directorios dedicados en el almacenamiento externo
- Gestión de archivos de Java: es posible manipular archivos de la misma forma que Java
    - Posibilidad de leer/escribir archivos de objetos

Android - Almacenamiento específico de la app

Características:

- No se necesitan permisos especiales
- Se puede controlar si otras aplicaciones pueden acceder a los archivos alojados en el almacenamiento externo
- Los archivos se eliminan cuando se desinstala la app
- Almacenamiento limitado

Android - Almacenamiento compartido

- Archivos a los que pueden acceder otras aplicaciones
- Los archivos no se eliminan cuando se desinstala la app
- Distinción entre archivos multimedia y otros tipos de archivos
    - Diferentes características en su gestión

Android - Almacenamiento compartido

Archivos multimedia:

- Imágenes, audios y videos
- Utilización de API MediaStore
- Requiere permisos especiales para acceder a archivos de otras aplicaciones
    - READ_EXTERNAL_STORAGE
    - WRITE_EXTERNAL_STORAGE

Android - Almacenamiento compartido

Otros archivos:

- Utilización de Intents para el acceso
- En Android 9 o anterior se requiere el permiso READ_EXTERNAL_STORAGE, al igual que con archivos multimedia
- A partir de Android 10 no se requieren permisos extras

Android - Preferencias de la aplicación

- Destinado para almacenar información de preferencias de aplicación, pero puede utilizarse para guardar cualquier dato
- Posibilidad de tener un archivo por cada *actividad*, o uno o múltiples archivos de preferencias compartidos
- Los datos se almacenan como pares clave–valor
    - Claves de tipo String (recomendable definir constantes de clase o en R.string)
    - Valores de tipo Strings o Integer (posibilidad de almacenar objetos serializados)

Android - Preferencias de la aplicación

- No se requieren permisos extras
- Utilización de Android Jetpack (Clase *Preference*)
- Simplicidad de uso, ya que no es necesario manipular archivos

Android - Bases de datos

- SQLite embebido en Android
- No se requieren permisos extras
- Los datos se eliminan al desinstalar la aplicación
- Utilización de biblioteca Room
    - Capa de abstracción de SQLite
    - Requiere que se agregue como dependencia

Android - Bases de datos

- Utilización de biblioteca Room
  - Uso similar a ORMs populares:
    - Definición de objetos Entidades
    - Definición de DAOs para acceder a las entidades
    - Uso de anotaciones para definir información específica del DBMS (columnas, claves, consultas, entre otras)

iOS

iOS

- El sistema proporciona varias opciones para almacenar datos de aplicación:
  - UserDefaults
  - Property Lists
  - Keychain
  - Archivos en almacenamiento interno
  - SQLite
  - CoreData

iOS - UserDefaults

- Se trata de una clase especial que sirve para almacenar información en formato de pares clave–valor
- Pensada para almacenar preferencias de aplicación (análogo a *Preference* de Android)
- Sólo se pueden almacenar tipos primitivos
- Simplicidad de uso
- Objeto *singleton* —uno solo por app

iOS - Property Lists

- Archivos que permiten almacenar información en formato de pares clave–valor
- Posibilidad de definir múltiples archivos para cada app
- Sólo se puede almacenar una cantidad limitada de tipos de datos:
  - Primitivos (Strings, Numbers, Bools, Dates, NSData)
  - Array
  - Dictionary

iOS - Keychain

- Es una base de datos encriptada que provee una API para su uso
- Pensada para almacenar datos sensibles (por ej., contraseñas)
- Conviene utilizar librerías que facilitan su uso
- Acceso lento

iOS - Archivos en almacenamiento interno

- Las aplicaciones en iOS se ejecutan dentro de un sandbox dedicado; iOS provee un espacio de almacenamiento dedicado exclusivo de la app
- iOS provee una estructura definida del sistema de archivos del sandbox que permite a la aplicación separar archivos que el usuario puede (o no) manipular
- La clase FileManager provee una API simple para la gestión de archivos
- FileManager se puede utilizar con iCloud

iOS - SQLite

- SQLite embebido en iOS
- Debe incluirse la librería en el código de la aplicación
- Interacción mediante API de SQLite

iOS - CoreData

- Es una capa de abstracción que puede utilizar SQLite (también puede definirse para utilizar almacenamiento binario, y en memoria —permitiendo que sea utilizado como una caché)
- Permite almacenar objetos; similar a base de datos de grafos (objetos = nodos; relaciones = aristas)
- Similar a ORM en cuanto a funcionamiento

Enfoques multiplataforma

Enfoques multiplataforma

- Existen diferentes métodos de persistencia de datos según el enfoque
- Cada enfoque tiene métodos propios, que se sustentan sobre los que proveen los sistemas operativos
- En cada enfoque es posible utilizar bases de datos para aplicaciones móviles
    - Se consideran bases de datos para aplicaciones móviles a las bases de datos que se instalan y almacenan los datos en el dispositivo, sin la necesidad de un servidor central

Web/PWA

Enfoque Web/PWA

Web/PWA

Storage

- API de almacenamiento del navegador
- Almacenamiento en forma de pares clave–valor de Strings
- Acceso sincrónico
- Tipos:
  - LocalStorage: datos no expiran
  - SessionStorage: datos se eliminan al cerrar la sesión de navegación

IndexedDB

- API de bajo nivel que ofrece almacenamiento en el cliente de cantidades significativas de datos estructurados
- Permite almacenar archivos y blobs
- Almacenamiento en forma de pares clave–valor
- Acceso asincrónico

Ionic

Enfoque Híbrido

Ionic

Storage

Al igual que en el enfoque Web/PWA, también pueden utilizarse LocalStorage y SessionStorage

Ionic Offline Storage

- Base de datos NoSQL que permite almacenar datos en forma de pares clave–valor u objetos JSON
- Capa de abstracción de SQLite, IndexedDB, WebSQL o LocalStorage
- Requiere instalación de plugin cordova (cordova-sqlite-storage) + paquete de ionic (@ionic/storage)

React Native

Enfoque interpretado

React Native

AsyncStorage

- Almacenamiento en forma de pares clave–valor de tipo String
- En iOS almacena valores pequeños en diccionarios serializados y valores grandes en archivos separados
- En Android almacena los datos en RocksDB (si está disponible) o en SQLite
- Acceso asincrónico

.NET MAUI

Enfoque de compilación cruzada

.NET MAUI

- Utiliza diferentes interfaces y clases con una implementación por defecto contenidas en el espacio de nombre *Microsoft.Maui.Storage*
- Cada interfaz/clase representa una abstracción de mecanismos nativos del sistema operativo
- Tres posibilidades:
  - File System
  - Preferences
  - Secure Storage

.NET MAUI

File System

- Interfaz *IFileSystem*, implementación por defecto *FileSystem.Current*
- Posee diferentes métodos *helpers*
- Permite acceder a:
  - directorio de caché
  - directorio de datos
  - archivos empaquetados en la aplicación

.NET MAUI

Preferences

- Interfaz *IPreferences*, implementación por defecto *Preferences.Default*
- Almacena datos en forma de clave-valor
- Claves de tipo *string*
- Valores pueden ser: *Boolean*, *Double*, *Int32*, *Single*, *Int64*, *String*, *DateTime*
- Acceso a datos sincrónico
- Por defecto, sólo visibles por la app, pero pueden compartirse con otras
- En Android se utiliza *Shared Preferences*
- En iOS se utiliza *User Defaults*

.NET MAUI

Secure Storage

- Interfaz *ISecureStorage*, implementación por defecto *SecureStorage.Default*
- Almacena datos en forma de clave-valor
- Claves y valores de tipo *String*
- Acceso a datos asincrónico
- Requiere configuración específica para cada plataforma
- En Android se utiliza *Shared Preferences*, encripta los datos utilizando la clase *EncryptedSharedPreferences*
- En iOS se utiliza *Keychain*

Bases de Datos para Aplicaciones Móviles

Bases de datos móviles

- Son bases de datos que se instalan en dispositivos móviles
- Permiten gestionar información sin conectividad
- Rendimiento estable y predecible, independiente de la conectividad
- Posibilidad de sincronizar información con una base de datos central

Algunas bases de datos móviles

SQLite

#9 general
#6 relacional

- Librería codificada en C que implementa RDBMS autocontenido y de alta disponibilidad
- DBMS más utilizado en el mundo:
  - cada dispositivo Android
  - cada dispositivo iOS y Mac
  - cada dispositivo Windows 10
  - cada navegador Firefox, Safari, Chrome
  - cada instancia de Skype, iTunes, Dropbox

SQLite

#9 general
#6 relacional

- Dominio público
- Primera versión: 2000
- Versión actual: noviembre 2023 (v3.44.0)
- Ranking DB Engines
    - 9 general
    - 6 relacional

Couchbase Lite

#32 general
#5 documental

- Base de datos NoSQL documental embebida
- Almacena datos en documentos JSON
- Posibilidad de utilizarla de modo autónomo o con Couchbase Server y Sync Gateway
- Licencia Apache
- Primera versión: 2011 (Couchbase Server)
- Versión actual: abril 2023 (v3.1)

Firebase

#38 general
#6 documental

- Base de datos NoSQL documental alojada en cloud
- Almacena datos en un único documento JSON
- Sincronización inmediata y automática con el servidor
    - Almacenamiento local sin conexión
- Imposibilidad de uso autónomo
- Licencia comercial
- Primera versión: 2012

Realm

#53 general
#9 documental

- Base de datos NoSQL documental
- Trabaja con datos en formato JSON, EJSON y BSON
- Posibilidad de utilizarla en modo autónomo, o con MongoDB y Realm Sync
- Licencia Apache
- Primera versión: 2014

Oracle Database Lite

#1 general

- Base de datos relacional
- Múltiples versiones. Entre ellas:
  - Enterprise Edition (EE)
  - Lite Edition (LE) (Portable)
- Licencia comercial
- Primera versión: 1980
- Versión actual: enero 2019 (19c)

Oracle Berkeley DB

#102 general
#18 clave-valor

- Base de datos NoSQL clave-valor
- Múltiples versiones: Core, Java Edition, y XML DB
- Licencia dual
- Primera versión: 1994
- Versión actual: junio 2018 (18.1)

IBM Db2 Everyplace

#8 general
#5 relacional

- Base de datos relacional
- Múltiples versiones. Entre ellas: Database, Warehouse, Everyplace
- Licencia comercial
- Primera versión: 1983
- Versión actual: octubre 2016 (12.1)

InterBase

#67 general
#37 relacional

- Base de datos relacional
- Licencia comercial
- Primera versión: 1984
- Versión actual: diciembre 2019 (InterBase 2020)

SAP SQL Anywhere

#76 general
#42 relacional

- Base de datos relacional
- Licencia comercial
- Primera versión: 1992
- Versión actual: julio 2015 (17)

PouchDB

#114 general
#22 documental

- Base de datos NoSQL documental
- Implementada y para utilizar javascript
- Posibilidad de sincronizar con CouchDB
- Primera versión: 2012
- Versión actual: febrero 2023 (v8.0.1)

LiteDB

#170 general
#30 documental

- Base de datos NoSQL documental
- Inspirada en MongoDB
- Documentos BSON
- Sólo plataforma .NET
- Licencia MIT
- Primera versión: 2014
- Versión actual: enero 2020 (v5)

ObjectBox

#179 general
#6 orientada a objetos

- Base de datos NoSQL orientada a objetos
- Embebida y rápida, pensada para IoT y Mobile
- Licencia Apache
- Primera versión: 2017
- Versión actual: noviembre 2023 (v3.7.1)

Sparksee

#343 general
#34 grafos

*Sparksee

- Base de datos NoSQL de grafos
- Ofrece versión mobile
- Licencia comercial
- Primera versión: 2008
- Versión actual: 2021 (v6.0)

Gracias

Referencias

1. https://developer.android.com/training/data-storage
2. https://developer.apple.com/documentation/coredata
3. https://www.iosapptemplates.com/blog/ios-development/data-persistence-ios-swift
4. https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Articles/AboutInformationPropertyListFiles.html
5. https://developer.apple.com/library/archive/documentation/FileManagement/Conceptual/FileSystemProgrammingGuide/FileSystemOverview/FileSystemOverview.html
6. https://medium.com/@imranjutt/data-persistence-in-ios-2804d04bde62
7. https://medium.com/macoclock/ios-persistence-and-core-data-c3c23e7152f3
8. https://expressflow.com/blog/posts/best-storage-for-web-apps
9. https://developer.mozilla.org/es/docs/Web/API/Window/localStorage

Referencias

10. https://ionicframework.com/docs/angular/storage
11. https://reactnative.dev/docs/asyncstorage
12. https://learn.microsoft.com/en-us/dotnet/maui/platform-integration/storage/file-system-helpers?view=net-maui-8.0&tabs=android
13. https://learn.microsoft.com/en-us/dotnet/maui/platform-integration/storage/preferences?view=net-maui-8.0&tabs=android
14. https://learn.microsoft.com/en-us/dotnet/maui/platform-integration/storage/secure-storage?view=net-maui-8.0&tabs=android
15. https://db-engines.com/en/ranking
16. https://greenrobot.org/news/mobile-databases-sqlite-alternatives-and-nosql-for-android-and-ios/
17. https://en.wikipedia.org/wiki/Mobile_database

Referencias

18. MARRERO, Luciano, et al. Un estudio comparativo de bases de datos relacionales y bases de datos NoSQL. En *XXV Congreso Argentino de Ciencias de la Computación (CACIC)(Universidad Nacional de Río Cuarto, Córdoba, 14 al 18 de octubre de 2019)*. 2019.
19. https://www.sqlite.org/index.html
20. https://www.couchbase.com/
21. https://en.wikipedia.org/wiki/Couchbase_Server
22. https://firebase.google.com/docs/database
23. https://realm.io/
24. https://en.wikipedia.org/wiki/Realm_(database)
25. https://www.oracle.com/database/technologies/related/berkeleydb.html
26. https://en.wikipedia.org/wiki/Oracle_Database

Referencias

27. https://es.wikipedia.org/wiki/Oracle_Database
28. https://en.wikipedia.org/wiki/IBM_Db2_Family#IBM_DB2_Everyplace_.28DB2e.29
29. ftp://public.dhe.ibm.com/software/data/db2/everyplace/doc/v82/esp_iug.pdf
30. https://www.embarcadero.com/products/interbase
31. https://pouchdb.com/
32. http://www.litedb.org/
33. https://objectbox.io/
34. http://sparsity-technologies.com/#sparksee