3rd National Conference on Green Technology and Science for Sustainable Development
© 2020 IJSRST | Volume 5 | Issue 6 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
International Journal of Scientific Research in Science and Technology

Database Connectivity with Different API in APK Generation

Dr. A. S. Kapse*, Harshada D. Raut[2], Ankita Amilkanthwar[3]
[*1]Assistant Professor (Sr. Scale), Department of Computer Science and Engineering, AEC, Chikhali, Maharashtra, India
[2,3] B.E Scholar, Department of Computer Science and Engineering, AEC, Chikhali, Maharashtra, India

ABSTRACT

Programming is knowledge intensive. While it's well understood that programmers pay various time searching for information, with less exceptions, there's a major lack on what information they look for. When programmers need answer, they move towards Stack Overflow. The main aim of database connectivity is to store and retrieve data. This provides a novel opportunity. There are vast numbers of applications for Android devices, which can be readily many traces of interactions, and analyse on Stack Overflow. Here, we can examine data from Android applications and Stack Overflow together, to find out what the programmers want to know and why. In this distributed, or crowd source, education system, people ask questions and others answer them, usually for no remuneration. As APIs see more usage, more questions are asked about them. As APIs and libraries extend in complexity there has been an increased focus on lever-aging, combining, and understanding multiple sources of developer documentation. This paper presents a method for creating a centralized database which will be used by both Web as well as Android application. The android application will be having its own local database which will be used for offline storage of data.

Keywords : Android, Android Database, Centralized Database, Common Database, Web Services Stack, Database connectivity, API.

I. INTRODUCTION

Ever growing android market has forced a lot of web applications to have a companion mobile application. Since the web application is already built and having database which is already used online and we need to use that same database for both android as well as for web application. This will reduce the data redundancy.

The android application have its own database (SQLite Database) which is used by android application to store data offline. When the android applications goes online this data should be updated to the online database which will be common for both the application i.e. android and web. Both the application will be able to connect to the database using web services. There will be a mechanism to synchronize the local database on android application to online database and vice-versa. The local database will keep track of whether the data is synchronized or not. The web services will get local data from the android application and check the similar data on the online database if it is not similar data then the local data will be added to the online database otherwise the local data will be replace by online data. Similarly reverse process will take place i.e. synchronize the online data in local database [1].

In this paper, the web application and web services will be developed in C#, ASP.net and the centralized

database by using SQL server because it is by default compatible with .NET. The main aim of database connectivity is to store and retrieve data.

Database for Android Application

Android allows to save data persistently. For example:- If a user wants that login-id and password for an application should be save persistently so that whenever he or she wants to login credentials should not be needed again. Android provides the various data storage options such as shared preferences, internal and external storages, SQLite database and network connection (web).

SQLite database is store in structure format in android. The data of android application can also be stored on the web through network server [2].

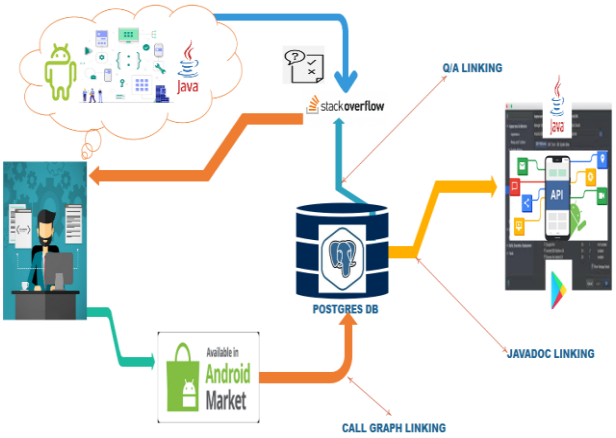

Fig -1: Database connectivity involves various stack flow parameters

II. LITERATURE SURVEY

We have performed a lot of searching on 'Using databases with Android and Web Application' (in particular ASP .Net). We always found 2 different results for the same. The first would show us how to connect the web application to a database and the other would show us how to use database in android. The expected result was to find a method which can be used to connect both the application to a common database. So now we changed the search - How to connect android to online database (Since the web application is already online we dint use that to search)

This search gave us the techniques to connect android to the network and use different API's to fetch online data. There was very less information on creating a custom web service that will provide the connection between the online database and android.

III. CHARACTERISTICS OF UBIQUITOUS MOBILE DBMS

The data access and management requirements of mobile applications are significantly different from that of traditional database. The new application must be able to run on multiple platforms, from devices servers web and various existing database mechanism [3] [4].

Embeddable in application: This applications carries the DBMS functionality with it and installed without any configuration and administration with support for multiple transaction and application.

Small footprint: The size of the DBMS affects the performance of application so it is important to minimize DBMS.

Run on mobile devices: The Database management system should run on variety of mobile devices with the limitation of memory, processor, and disk space and on specialized operating system that can be synchronized with back-end systems.

Self-managed DBMS: The embedded DBMS is invisible to the application user. There can be no database administration to manage the database and operation like backups, recovery, indexing tuning etc.
In-memory DBMS: The database is too small to be contained in main memory so by using in memory DBMS techniques for optimizing query processing and

indexing, these special purpose database application improve performance on the database .

Synchronize with back-end data sources: It must be possible to synchronize the data with back-end data sources.

IV. ADVANTAGES

A. Lightweight

SQLite is a very light weighted database so, it is easy to use it as an embedded software with devices like televisions, Mobile phones, cameras, home electronic devices, etc.

B. Better Performance

Reading and writing operations are very fast for SQLite database. It is almost 35% faster than File system.

It only loads the data which is needed, rather than reading the entire file and hold it in memory.

If you edit small parts, it only overwrite the parts of the file which was changed.

C. No Installation Needed

- SQLite is very easy to learn. You don't need to install and configure it. Just download SQLite libraries in your computer and it is ready for creating the database.

D. Reliable

- It updates your content continuously so, little or no work is lost in a case of power failure or crash.
- SQLite is less bugs prone rather than custom written file I/O codes.
- SQLite queries are smaller than equivalent procedural codes so, chances of bugs are minimal.

E. Portable

- SQLite is portable across all 32-bit and 64-bit operating systems and big- and little-endian architectures.
- Multiple processes can be attached with same application file and can read and write without interfering each other.
- It can be used with all programming languages without any compatibility issue.

F. Accessible

- SQLite database is accessible through a wide variety of third-party tools.
- SQLite database's content is more likely to be recoverable if it has been lost. Data lives longer than code.

G. Reduce Cost and Complexity

- It reduces application cost because content can be accessed and updated using concise SQL queries instead of lengthy and error-prone procedural queries.
- SQLite can be easily extended in in future releases just by adding new tables and/or columns. It also preserve the backwards compatibility. [5]

V. DISADVANTAGES

- SQLite is used to handle low to medium traffic HTTP requests.
- Database size is restricted to 2GB in most cases. [6]

VI. CONCLUSION

This paper is divided into two sections, the first describes about the database in Android and the second describes about the database in Web Application along with Web Services. The first section will help to create a local database for the Android application. The second section describes about the online database that can be created in Microsoft SQL Server. This section provides details about creating Web service which will be used for connecting to the database. By combining the knowledge from both these section one can create a centralized database for Web and Android Application.

database for Web and Android Application.

VII. REFERENCES

[1]. http://dkavaler.github.io/docs/usingandasking.pdf

[2]. Pradeep Kothari and Kogent Learning Solutions Inc., "Android Application Development (with Kitkat Support) Black Book", Dreamtech Press, ISBN: 978-93-5119-409-5, 2014

[3]. https://en.wikipedia.org/wiki/Microsoft_SQL_Server#SQL_Server_2008

[4]. https://msdn.microsoft.com/en-us/library/ms176061.asp

[5]. http://www.sqlite.org/

[6]. http://programmerguru.com/android-tutorial/what-is-sqlite/

[7]. http://developer.android.com/guide/topics/data/data-storage.html#db.

[8]. http://www.codeproject.com/Articles/119293/Using-SQLite-Database-with-Android

[9]. http://developer.android.com/training/volley/index.html

[10]. http://www.androidhive.info/2014/05/android-working-with-volley-library-1