Jake Taggart
Week 2 Day 4 Research

Week 2 Day 4 Research

1. Come up with some use cases in which a content provider would be helpful.

   Some use cases for having a content provider contribute to an application(s) would be for something like a media player, an application(s) that allows one to browse cars for rental or purchase, an application(s) that allows one to browse through sports statistics, an application(s) that allows the user to browse through a catalog of a store, an application(s) that allows the user to look through a list of houses for rent, etc. Simply put, content providers can be incredibly useful for any application that needs to use a database (i.e. the vast majority of applications these days) and be able to access and manipulate the representation of that information in the ways they want. Also, of course, applications that need to be able to add to a database or remove items from a database surely would want to use content providers. Another great use of content providers is the ability to share databases with multiple applications. Of course, whichever entity actually controls the content provider would most likely be the main controller of adding/deleting/modifying data in the database(s). Content providers allow developers to not have to hard code massive libraries of information and allows them to be more flexible with information.

2. How do file streams work in JAVA?

   File streams are a flow of data from or to a destination in Java. For example, a file input stream would be a way to get files directly from the local computer memory or from other networks. An output stream would be a way to store data to the local computer or to send it across networks. Anything being passed through streams is not strictly defined. Java file streams can range anywhere from primitive data (booleans, longs, etc.) to advanced data such as objects. When the developer initiates a file stream, it is extremely important to close said file stream at the end of the process. The risk here is that data leaks will occur, and using the previous question as a stepping stone - when working with content providers, this can result in majorly unwanted consequences especially when working with proprietary data or sensitive information.

3. Explain the process of implementing a content provider, and to get the info from a content provider.

To implement a content provider, one needs to create a class that extends ContentProvider, then create a contract class, a URI matcher definition (for responding to potentially multiple URI's), implement the onCreate() method, implement the getType() method, implement the create, read, update, and delete functions (CRUD), and finally add the content provider to the manifest.

4. What is a vector drawable and how do we implement and use them in android?

Vector drawables are sets of points, lines, curves, and associated color information of those attributes taken together as a whole. Things like simple shapes, lines, and any visual that can be mapped easily with just those four attributes. Drawable vectors are defined in xml with groups and paths. Groups are the holders for any other groups and/or paths that need to be together for whatever reason (perhaps a drawable vector of a basic analog clock would group everything with the hands and center points being grouped with their individual paths going different directions while another group just holds the curves that make up the circle). Each path has its own set of attributes and each group can also have its own set of attributes too. These range from both dimensions' pivots (directional dimension for groups), scale (for groups), rotation (groups), stroke width and height (paths), fill type (paths), fill color (paths), and more. We can implement and use them in android by implementing the Drawable class and importing the vector.Drawable resource library.

5. Define the following:

a) Content Resolver

The content resolver is often confused with the content provider and indeed the two work in close tandem with each other with a thin line of difference between the two. But the difference is very important even if it is a little tricky to demonstrate with source code. The content resolver is a singular, global instance of the content class which provides access to the application's content provider or other applications' content providers assuming they're made public or shared with the application at hand. The content resolver gets queries from the application's code and sends them to the content provider and then resolves the resulting information sent back so that the application's code can make use of the result. Thus, the content resolver is different from the provider in that it is the layer that provides abstraction between the provider and the application's source code. Not everything needs to be directly accessed and modified, so this layer of abstraction is key in making content providers allow public use or shared use of a database without having to worry about the database being directly modified. Note that

the main difference is that the content provider is an entity that actually has direct use and access to the database as the content provider has control over its database(s). The resolver does not.

b) Primary Key (Sql)

A primary key is SQL is a field in a table of a database which can be called as a unique identifier for a row of information. The primary key needs to be unique so that no unwanted information is passed along with the target query information. Most database tables should be kept simple so that the intended column can only have unique values. The more complex the database table gets with multiple fields that match in what would usually be a primary key, often times multiple fields are used as a primary key to ensure uniqueness. A good example of this would be a large music database table in which hundreds of songs are listed. Generally, one would expect a song title to be the primary key, but consider how many songs are titled such cliches as “I Love You”. This would not be fit for a primary key unto itself, but would need to be in tandem with, say, the artist as the second field for the primary key so the correctly sought after song will be returned. Primary key values cannot be null in the database table being pulled from.

c) Foreign Key (Sql)

A foreign key is similar to a primary key in a way, but also very different. A foreign key is used to connect information from two different tables. For example, let’s say the same music table had the artist and song columns, but didn’t have information on any movies those songs were featured in, if any. One would use the foreign key as a way to see if the primary key of the song title matched the song title in another table holding movie soundtrack information in order to return the specific movie(s) the primary key’s song was featured in.

d) Relational Database

A relational database is a structured set of tables in which information is organized with the intention of allowing the retrieval of and restructuring of such information without directly modifying that information in the actual database. Databases are useful in that they allow access, manipulation, and reorganization of data based on queries (requests for information by those who have access which have explicit directions for how to return such information). Whoever the content provider is has the ability to modify, update, delete entities, or add new entities to its own database. Any others usually don’t have permission to modify the database tables, but potentially have the ability to retrieve

information based on how they see fit depending on whether or not they have permission for access.

e) Dangerous Permissions

Dangerous permissions are any permission which the user himself/herself must give consent to for the application to work a certain way. One example of a dangerous permission would be when someone opens up a social media application such as Snapchat and is asked to give consent for the application to gain access to the phone's camera, audio recorder, and contacts to. This would be the system dialogue. Another (optional) part of the dangerous permission dialogue would be the rationale. The rationale is a way for the developer to explain why this system dialogue is asking for access to these things any way. So an example of that would be Snapchat saying "Snapchat would like to access your contacts to make it easier for you to find friends" and the system dialogue would say something like "Allow permission to access Contacts?" with a buttons for "yes" and "no".

6. What is an ORM?

An object-relational mapping as an idea is being able to query a database without having to write the queries in SQL. Instead, object-relational mapping aims to allow the developer to make queries based on the language paradigm they are already coding in. Since we deal with objects in java, the syntax for querying will be much like telling an object to perform a method with the query being what is usually the argument passed. Thus, an object-relational mapper itself is a library which implements this specific mapping technique. A good way to think of an ORM is a library that when implemented, maps all of necessary the object-oriented language conventions to the equivalent SQL conventions. Instead of having to go in and type SQL queries, we now can continue writing in the object-oriented language we are in. Obviously, this can be highly useful when creating a complex content resolver to retrieve information from a content provider. There are tradeoffs, though. Some downsides include potentially not being able to be as specific and flexible with queries as someone with advanced SQL skills using that method would, having to configure the ORM which can be irritating, there is a learning curve to using an ORM, and if not carefully considered can be used as a crutch instead of a helper. However, as aforementioned, there are plenty of pros to using an ORM especially if the developer knows how it works and is willing to configure it properly. For one, we do not have to write in SQL. While SQL is an awesomely flexible and useful language, mastering all the ins and outs takes a lot of time and will distract from other things that hold a higher priority - namely, making the code functional.

Another thing it's great for is providing a layer of abstraction between the language used for a database and our code, so if one were using a database being managed and accessed with a variety of SQL that we are unfamiliar with (there are other database management language variants other than MySQL for SQL), we don't have to worry about having to learn that variant in any depth. Because of this layer of abstraction, another pro is that advanced SQL features are already available to us immediately. Finally, ORM's are great we don't have to worry about whether the syntax of a query or any action using a database is perfect in that language. This allows for far less troubleshooting and headaches in general.

7. Explain how you would upgrade a Table in your database with a new column while preserving the data already in said table.

First of all, you would need to actually update the table in your own content provider's database (the actual database, not the content resolver's side of the operation). To do this, you would need to use the ALTER TABLE and ADD COLUMN keywords in SQL to update the real database. However, the way onUpgrade() works is by taking in the database, the old version's identifier (usually an integer), and the new version's identifier of the database as arguments. So the developer needs to use conditionals to check if there is an upgrade and if there is, how many upgrades were made. Thus, based on how many upgrades are made, the onUpgrade() method should tell the content resolver to alter its own representation of the newly modified structure based on those upgrades.