

When working on my dc Redial Android App, I had to learn how to work with the Android Call Log Content Provider. The information I used was spread over several different sources, so I figured I would post an article detailing how to work with the Android Call Log Content Provider.

Working with the Android Call Logs is really pretty simple. The primary weapon in your arsenal for dealing with the Android Call Logs is what Android calls a Content Provider. Basically, a content provider is a special kind of database that Android uses so that programs can expose their data to each other. To use a content provider, you just need to do is the call the "getContentResolver()" method. Once you have a content content resolver object, you can query it using SQL or any of the built in query functions. For example, in my dc Redial Android App, I use a piece of code that looks like this:

```
String[] strFields = {      android.provider.CallLog.Calls.NUMBER,
android.provider.CallLog.Calls.TYPE,      android.provider.CallLog.Calls.CACHED_NAME,
      android.provider.CallLog.Calls.CACHED_NUMBER_TYPE      }; String strOrder =
android.provider.CallLog.Calls.DATE + " DESC";      Cursor mCallCursor =
getContentResolver().query(      android.provider.CallLog.Calls.CONTENT_URI,
strFields,      null,      null,      strOrder      ); //
```

This little piece of code does a few things, so we'll take them each in turn. First, it sets up a String Array to hold the names of the fields that we are going to query for. Then it creates a String to hold an SQL "ORDER-BY" command, but because the query() method has the ability to order things built in, we leave out the actual "ORDER-BY" keyword. Finally, we query the Android Call Logs Content Provider. That query takes five arguments: one, the table we are going to query; two, a String[] Array of the fields we want; three, a selection String (sort of like an SQL "WHERE" clause, but without the "WHERE"); four, a selection arguments String[] Array, this allows you to use a ? character in your selection string (in the previous argument) that will be replaced by the values supplied in this argument; and five, an order String so you can sort your results. Let's take a look at all this in a little more detail

Call Logs Content Provider Fields

As you can see above, this query will get the number, type of call, name of caller, and number type (e.g. mobile, home, etc). But this is clearly not an exhaustive list. Here are a few of the highlights:

- CACHED_NAME: the name of the caller
- CACHED_NUMBER_TYPE: type of number, e.g. mobile, work, home, etc.
- DATE: date of the call in milliseconds since the "Epoch" (January 1, 1970)
- DURATION: length of the call in seconds

Android Tutorial: Call Logs

Written by dcPages

- NUMBER: phone number
- TYPE: type of the call, e.g. incoming, outgoing, etc.

For a complete list of all the members of the CallLog.Calls provider, you can visit the Android Developer's Website at

<http://developer.android.com/reference/android/provider/CallLog.Calls.html> . On that website you can find all sorts of great information. Like, for example, the values associated with the different call types, i.e. incoming = 1, outgoing = 2 and missed = 3. Or the values associated with the different number types (hint: that one comes from android.provider.Contacts.Contacts.PhonesColumns).

Once you have executed your query, Java will return a Cursor that you can use to loop through your results. Working with cursors is beyond the scope of this tutorial, but I will mention that I typically like to loop through cursors using a Do...While loop. That code looks something like this:

```
// get start of cursor if(mCallCursor.moveToFirst()){ // loop through cursor do{ ...  
some code goes here ... } while (mCallCursor.moveToNext()); }
```

That little bit of code will check to make sure that you have at least one record so that your app doesn't crash. Then it will execute the code in the Do loop as long as the moveToNext() method keeps spitting out records.

Well, that about covers the basics of working with the Android Call Logs, if you have any questions or comments, you can email me at info@dcpagesapps.com or just submit a comment below. As always, if you've gotten anything out of this tutorial, consider giving something back using the donate button on the left.

Thanks,

dcPages