Managing and Accessing Local Databases

SQLite is a very popular embedded database, as it combines a clean SQL interface with a very small memory footprint and decent speed. Moreover, it is public domain, so everyone can use it. Many firms (e.g., Adobe, Apple, Google, Sun, and Symbian) and open source projects (e.g., Mozilla, PHP, and Python) ship products with SQLite.

For Android, SQLite is “baked into” the Android runtime, so every Android application can create SQLite databases. Since SQLite uses a SQL interface, it is fairly straightforward to use for people with experience in other SQL-based databases. However, its native API is written in C, and while Java libraries such as JDBC are available for it, JDBC might be too much overhead for a memory-limited device like a phone, anyway. Hence, Android programmers have a different API to learn. The good news is that it is not very difficult.

This chapter will cover the basics of SQLite use in the context of working on Android. It by no means is a thorough coverage of SQLite as a whole. If you want to learn more about SQLite and how to use it in environments other than Android, a fine book is The Definitive Guide to SQLite, Second Edition, by Grant Allen (your present author) and Michael Owens (Apress, 2010). This also covers other complementary topics such as security of SQLite databases and more.

Much of the sample code shown in this chapter comes from the Database/Constants application. This application presents a list of physical constants, with names and values culled from Android’s SensorManager, as shown in Figure 32–1.
You can pop up a menu to add a new constant, which brings up a dialog box in which to fill in the name and value of the constant, as shown in Figure 32–2.