Database connectivity using SQLite

Chapter-4

Prepared By : Dharmendra Ambani   (Harivandana College – Rajkot)

Using Android Data and Storage APIs

- 1) Working with Preferences

- - Creating Preferences
- - Reading Preferences
- - Creating program of preferences

- Working with Files and Directories
- 1) Creating and Writing Files
- For Example, the following code create and opens a file called “filename.txt”.
- We write a single line of text to the file and then close the file.

- import java.io.FileOutputStream;
- ...
- FileOutputStream fos;
- String strFileContents = “Some text to write to the file.”;
- fos = openFileOutput(“Filename.txt”, MODE_PRIVATE);
- fos.write(strFileContents.getBytes());
- fos.close();

Reading files

```
FileInputStream fis = openFileInput(filename);
StringBuffer sBuffer = new StringBuffer();
DataInputStream dataIO = new DataInputStream(fis);
String strLine = null;
while ((strLine = dataIO.readLine()) != null) {
sBuffer.append(strLine + “\n”);
}
dataIO.close();
fis.close();
```

- Working with directories
- File fileDir = getFilesDir();
- String strNewFileName = “myFile.dat”;
- String strFileContents = “Some data for our file”;
- File newFile = new File(fileDir, strNewFileName);
- newFile.createNewFile();
- FileOutputStream fo =
- new FileOutputStream(newFile.getAbsolutePath());
- fo.write(strFileContents.getBytes());
- fo.close();

2) Managing Data using SQLite

- 1) Creating a SQLite Database
- 2) Creating a Table
- 3) Creating, Updating, and Deleting Database Records
- 4) Working with cursor
- 5) Executing simple queries.

- 6) Deleting Tables
- Db.execSQL(“DROP TABLE TBLNAME”);
- 7) Closing a SQLite Database
- Db.close();

3) Sharing Data Between Applications with Content Providers

- Application can access data within other applications on the Android system through content provider interfaces and expose internal application data to other applications by becoming a content provider.

- Provider
- -Media Store
  - Audio visual data on the phone and external storage.
- -Call Log
- sent and receive calls

- Browser
  - Browser history and bookmarks.
- Contacts
  - Phone contact database or phone book.

- Settings
  - System wide device settings and preferences.
- UserDictionary
  - A dictionary of user defined words for use with predictive text input.

- For Example read contacts

```
Cursor oneContact = managedQuery( People.CONTENT_URI, null,
null, null,
“name desc LIMIT 1”);

Log.d(debugTag, “Count: “ + oneContact.getCount());
```

- String nameIdx = oneContact.getColumnIndex(Contacts.People.NAME);

- String emailIDIdx = oneContact
- .getColumnIndex(Contacts.People.PRIMARY_EMAIL_ID);

- int phoneIDIdx = oneContact
- .getColumnIndex(Contacts.People.PRIMARY_PHONE_ID);

- END