References

Intent - Passing Data

IntentExample

Displays/Edits age

Go button
Calls PersonEditor
Passes data
Name
Age

PersonEditor

Displays/Edits Name and age

Done button
Returns edited data back
Age = 0 cancels edit
public class IntentExample extends Activity implements View.OnClickListener {
    private EditText numberText;
    private static final int INTENT_EXAMPLE_REQUEST = 123;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.intent);
        Button ok = (Button) findViewById(R.id.go);
        ok.setOnClickListener(this);
        numberText = (EditText) this.findViewById(R.id.number);
        numberText.setText("21");
    }

    When we want a reply back from an Intent request we supply a request number. The request number is return with the answer. That way it is possible to know where the request originated from.
public void onClick(View v) {
    Intent go;
    go = new Intent();
    go.setAction("android.intent.action.EDIT");
    go.addCategory("person_editor");
    String newAge = numberText.getText().toString();
    go.putExtra("age", newAge);
    go.putExtra("name", "Roger");
    startActivityForResult(go, INTENT_EXAMPLE_REQUEST);
}

The name was sent just to show we can send multiple items. They can be of any base type or serializable. See the putExtra methods in the Intent class.
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode != INTENT_EXAMPLE_REQUEST) {
        numberText.setText("Not from me");
        return;
    }
    switch (resultCode) {
    case RESULT_OK:
        String editedAge = data.getStringExtra("age");
        numberText.setText(editedAge);
        break;
    case RESULT_CANCELED:
        numberText.setText("Cancelled");
        break;
    }
}
public class PersonEditor extends Activity implements View.OnClickListener {
    private EditText ageText;
    private EditText nameText;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.person_editor);
        Button done = (Button) findViewById(R.id.edit_done);
        done.setOnClickListener(this);
        ageText = (EditText) this.findViewById(R.id.edit_age);
        nameText = (EditText) this.findViewById(R.id.edit_name);
        Bundle personData = getIntent().getExtras();
        String age = personData.getString("age");
        String name = personData.getString("name");
        if ((age != null) && (name != null)) {
            ageText.setText(age);
            nameText.setText(name);
        }
    }
}

Showing how to access the intent that started the activity and extracting the extras from the intent. If the key–value pair was not set in the intent the value will be returned as null.
public void onClick(View v) {
    String newAge = ageText.getText().toString();
    Intent result = getIntent();
    result.putExtra("age", newAge);
    if (newAge.equals("0"))
        setResult(RESULT_CANCELED, result);
    else
        setResult(RESULT_OK, result);
    finish();
}

setResult(int, Intent) returns information to the calling activity. The first parameter is the result code passed back in onActivityResult. The intent is the intent passed back in onActivityResult.
The intent filter for the activity must contain all the categories used by the intent to select the activity. It can contain more categories. The example does not work without the default category.
Data Storage
Data Topics

Preferences
Files
SQLite database
Content Providers
Network
Preferences

Key value pairs for program

Key - string

Value
  boolean
  float
  int
  long
  string

getPreferences(int mode)
  For access in activity only

getSharedPreferences(String name, int mode)
  To share preferences with other activities

mode
  0 = MODE_PRIVATE
  MODE_WORLD_READABLE
  MODE_WORLD_WRITEABLE

Cannot share preferences across applications or threads
public class Calc extends Activity {
    public static final String PREFS_NAME = "MyPrefsFile";
    protected void onCreate(Bundle state){
        super.onCreate(state);

        SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
        boolean silent = settings.getBoolean("silentMode", false);
        setSilent(silent);
    }

    protected void onStop(){
        super.onStop();
        SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
        SharedPreferences.Editor editor = settings.edit();
        editor.putBoolean("silentMode", mSilentMode);
        editor.commit();
    }
}
Files

Application can write/read files on phone

Cannot directly read files written by other application

Write a file
   FileOutputStream openFileOutput(String name, int mode)
      Creates file if it does not exist
      mode
      0 = MODE_PRIVATE
      MODE_APPEND
      MODE_WORLD_READABLE
      MODE_WORLD_WRITEABLE

   FileInputStream openFileInput(String name)
      name can not contain path seperators
Static files

You can package static files with your application

Place file in res/raw/<mydatafile>

Generates resource id in R

Read file using

Resources.openRawResource (R.raw.mydatafile)
File Example

Saves data in local file

Uses local preference to store data
public class FileExamples extends Activity implements View.OnClickListener {

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button update = (Button) findViewById(R.id.update);
        update.setOnClickListener(this);
        restoreData();
    }

    public void onClick(View v) {
        saveData();
    }
}
private void restoreData() {
    String fileContents = readFile();
    EditText fileText = (EditText) this.findViewById(R.id.file);
    fileText.setText(fileContents);

    EditText preferenceText = (EditText) this
        .findViewById(R.id.localPreference);
    SharedPreferences settings = getPreferences(MODE_PRIVATE);
    preferenceText.setText(settings.getString("setting", "No value"));
}
private String readFile() {
    String fileContents;
    try {
        InputStream file = new BufferedInputStream(
            openFileInput("dataFile"));
        byte[] data = new byte[file.available()];
        file.read(data, 0, file.available());
        fileContents = new String(data);
        file.close();
    } catch (Exception noFile) {
        fileContents = "empty";
    }
    return fileContents;
}
private void saveData() {
    EditText fileText = (EditText) this.findViewById(R.id.file);
    String fileContents = fileText.getText().toString();
    writeFile(fileContents);
    EditText preferenceText = (EditText) this
        .findViewById(R.id.localPreference);
    String preferenceContents = preferenceText.getText().toString();
    SharedPreferences settings = getPreferences(MODE_PRIVATE);
    SharedPreferences.Editor editor = settings.edit();
    editor.putString("setting", preferenceContents);
    editor.commit();
}
private void writeToFile(String fileContents) {
    try {
        OutputStream file = new BufferedOutputStream(openFileOutput("dataFile", MODE_PRIVATE));
        file.write(fileContents.getBytes());
        file.close();
    } catch (Exception noFile) {
    }
}


Database
Database

SQLite
- Embedded SQL database engine
- Free
- Source is in public domain
- Transactions
- File format is cross-platform

http://www.sqlite.org/index.html
Key Android Database Classes

android.database.sqlite.SQLiteOpenHelper
  Database creation
  Version management
  Database access

android.database.sqlite.SQLiteDatabase
  Create, delete, execute SQL commands

android.database.Cursor
  Read-write access to the result set
SQLiteOpenHelper

synchronized void close()
   Close any open database object.

synchronized SQLiteDatabase getReadableDatabase()
   Create and/or open a database.

synchronized SQLiteDatabase getWritableDatabase()
   Create and/or open a database that will be used for reading and writing.

abstract void onCreate(SQLiteDatabase db)
   Called when the database is created for the first time.

void onOpen(SQLiteDatabase db)
   Called when the database has been opened.

abstract void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
   Called when the database needs to be upgraded.
Android Databases

Accessible to all classes in an application

Can't access directly databases from other applications

Database errors are logged

Can connect to database on phone from shell

http://code.google.com/android/reference/adb.html#sqlite
Example

Show
- Creating database
- Inserts
- Update
- Delete
- Query
public class DatabaseHelper extends SQLiteOpenHelper {
    private static final String DATABASE_NAME = "name.db";
    private static final int DATABASE_VERSION = 1;

    public DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    public void onCreate(SQLiteDatabase nameDb) {
        nameDb.execSQL("CREATE TABLE " + "NAMES" + " (" + "_ID" + " INTEGER PRIMARY KEY," + "NAME" + " TEXT" + ");");
        nameDb.execSQL("INSERT INTO NAMES ( name) VALUES ('Roger' );");
    }

    public void onUpgrade(SQLiteDatabase arg0, int oldVersion, int newVersion) {
    }
}
public class DatabaseExample extends Activity implements View.OnClickListener {
    private EditText databaseIdText;
    private EditText nameText;
    private DatabaseHelper namesHelper;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        int[] buttonIds = { R.id.delete, R.id.read, R.id.insert, R.id.update };
        for (int id : buttonIds) {
            Button button = (Button) findViewById(id);
            button.setOnClickListener(this);
        }
        databaseIdText = (EditText) this.findViewById(R.id.databaseId);
        nameText = (EditText) this.findViewById(R.id.name);
        namesHelper = (new DatabaseHelper(this));
        displayDatabaseRecord(1);
    }
}
DatabaseExample.java

```java
private void displayDatabaseRecord(int id) {
    displayDatabaseRecord(String.valueOf(id));
}

private void displayDatabaseRecord(String id) {
    SQLiteDatabase nameDb = namesHelper.getWritableDatabase();
    Cursor result = nameDb.rawQuery("select * from NAMES where _ID = ?",
    new String[] { id });
    int rowCount = result.getCount();
    if (rowCount > 0) {
        result.moveToFirst();
        databaseIdText.setText(String.valueOf(result.getInt(0)));
        nameText.setText(result.getString(1));
    }
}
```
private String getName() {
    return nameText.getText().toString();
}

private String getId() {
    return databaseIdText.getText().toString();
}
public void onClick(View clicked) {
    SQLiteDatabase db = namesHelper.getWritableDatabase();
    switch (clicked.getId()) {
    case R.id.read:
        displayDatabaseRecord(getId());
        break;
    case R.id.delete:
        db.delete("NAMES", "_ID = ?", new String[] { getId() });
        break;
    }
onClickListener

  case R.id.insert:
      ContentValues newName = new ContentValues(1);
      newName.put("NAME", getName());
      db.insert("NAMES", null, newName);
      break;
  case R.id.update:
      ContentValues updateName = new ContentValues(1);
      updateName.put("NAME", getName());
      db.update("NAMES", updateName, "_ID = ?", new String[] { getId() });
      break;
  }
  }
}