Lesson 15

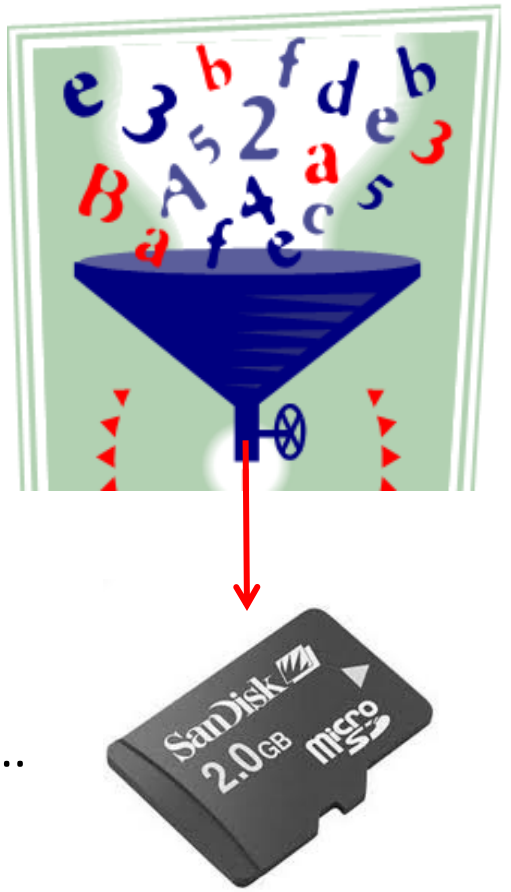
Android Persistency: Files

Victor Matos
Cleveland State University

Notes are based on:

Android Developers
http://developer.android.com/index.html

Android Files



Android's file management is similar to typical Java IO operations.

Files can be stored *internally* in the device's (small) main memory or *externally* in the much larger SD card.

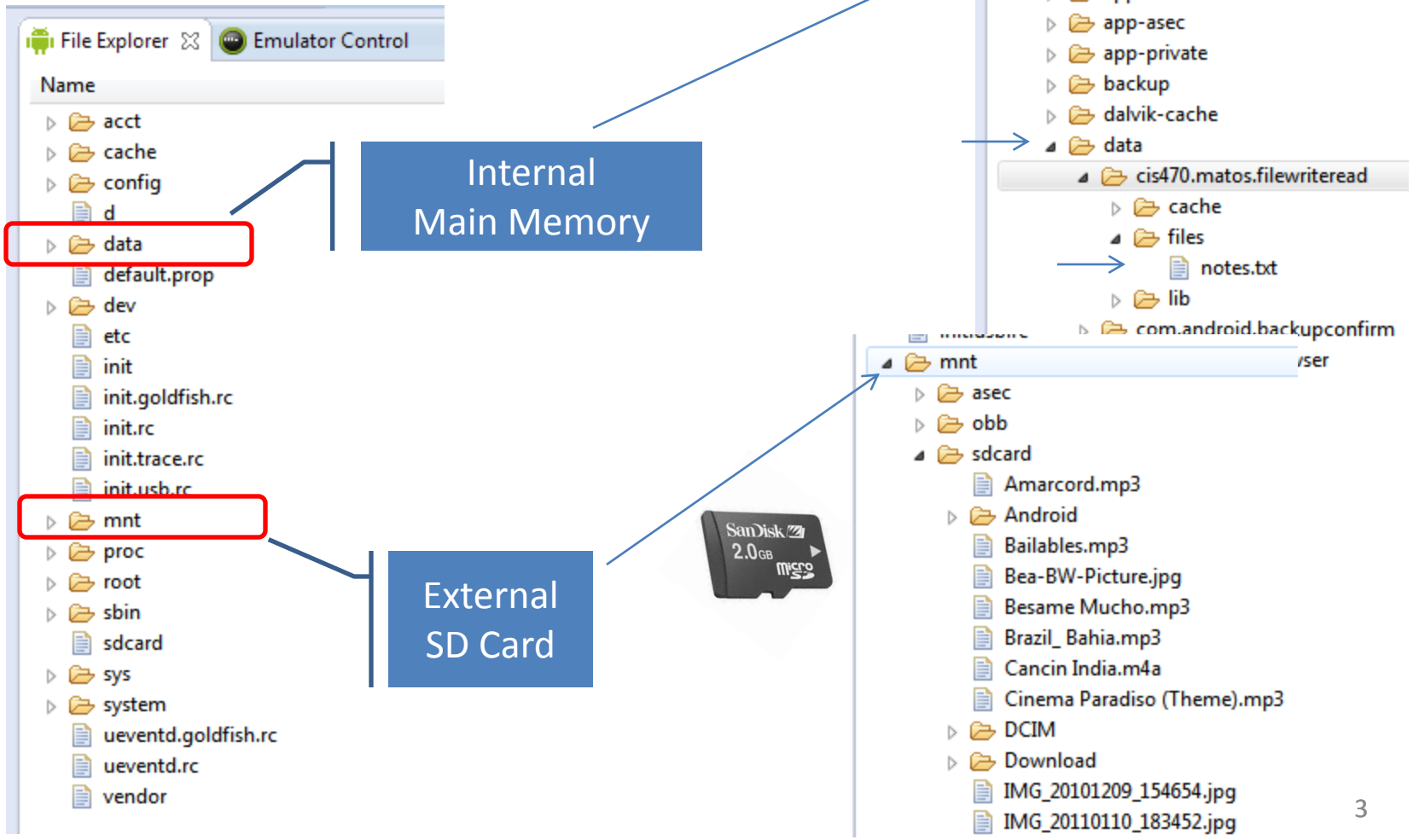
Files stored in the device's memory, share space with other application's resources such as code, icons, pictures, music, ….

Internal files are called: **Resource Files.**

External files to be attached to the compiled **.apk** could be stored in the folder **res/raw** (*create it if needed!*)

Android Files

Use the emulator's **File Explorer** to see and manage your device's storage structure.

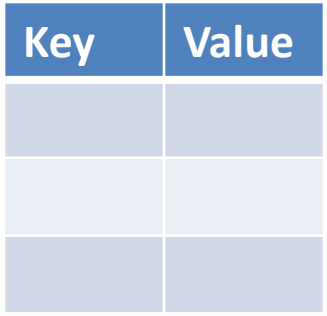


Android Files

Your data storage options are usually driven by parameters such as:
size (small/large), location (internal/external), accessibility (private/public).

1. **Shared Preferences** Store private primitive data in key-value pairs.
2. **Internal Storage** Store private data on the device's memory.
3. **External Storage** Store public data on the shared external storage.
4. **SQLite Databases** Store structured data in a private/public database.
5. **Network Connection** Store data on the web with your own network server.

Android Files



Shared Preferences. Good for a few items saved as <**KeyName, Value**>

```
private void usingPreferences(){
  // Save data in a SharedPreferences container
  // We need an Editor object to make preference changes.

  SharedPreferences settings = getSharedPreferences("my_preferred_Choices",
                                                    Context.MODE_PRIVATE);
  SharedPreferences.Editor editor = settings.edit();
          editor.putString("favorite_color", "#ff0000ff");
          editor.putInt("favorite_number", 101);
  editor.commit();

  // retrieving data from SharedPreferences container
  String favColor = settings.getString("favorite_color", "default black");
  int favNumber = settings.getInt("favorite_number", 0);

  Toast.makeText(this, favColor + " " + favNumber, 1).show();

}
```

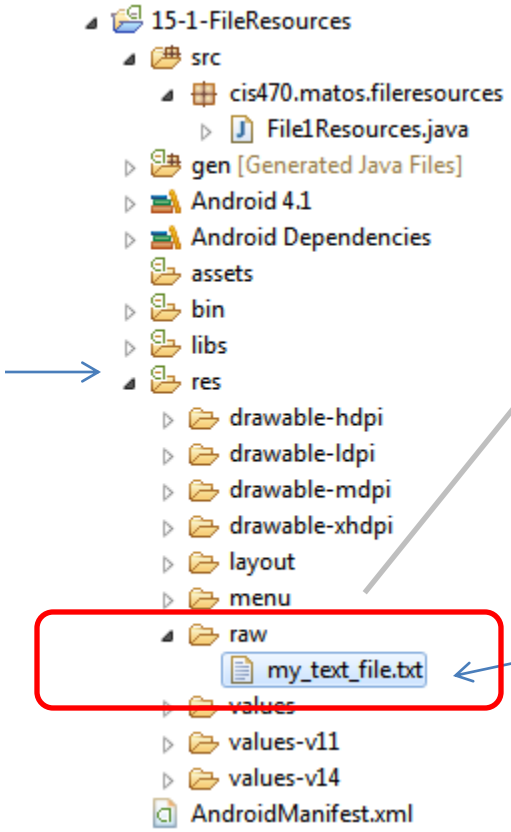
Android Files

Internal Storage. Using Android Resource Files

An Android application may include a number of resources such as those in: **res/drawable** , **res/raw, res/menu, res/style,** etc.

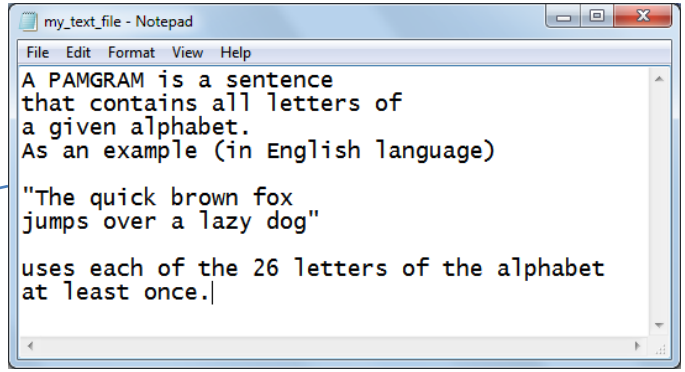
Resources could be accessed through the **.getResources()** method. For example:

```
InputStream is = this.getResources()
                      .openRawResource(R.drawable.my_text_file);
```



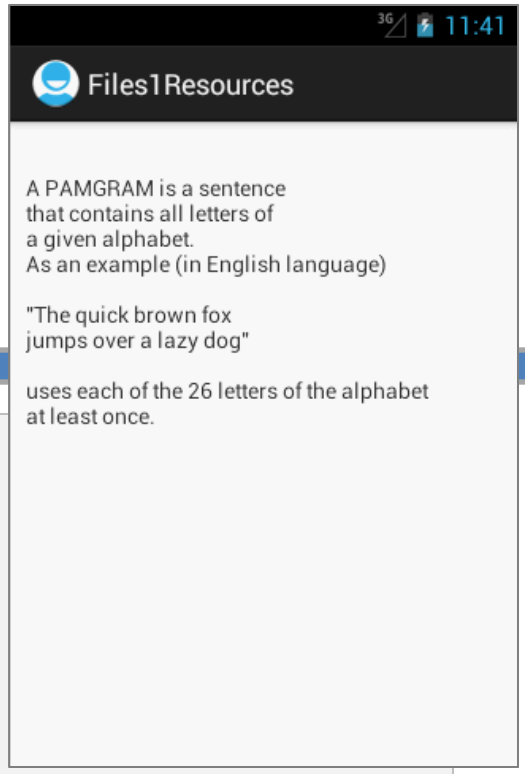
If needed create the **res/raw** folder.

Use drag/drop to place the file **my_text_file.txt** in **res** folder. It will be stored in the device's memory as part of the .apk



Example of a Spanish Pamgram
La cigüeña tocaba cada vez mejor el saxofón y el búho pedía kiwi y queso.

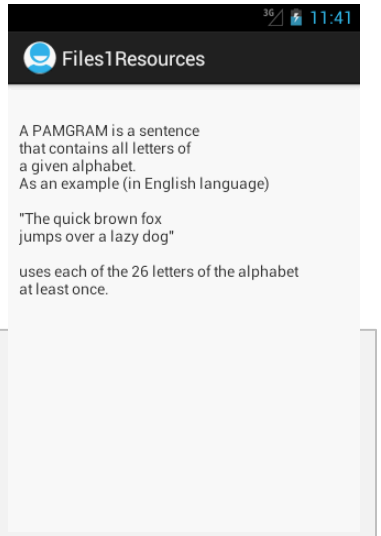
Android Files



Example 0: Reading a Resource File (see previous figure)

```
//reading an embedded RAW data file
public class File1Resources extends Activity {
   TextView txtMsg;
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);

      txtMsg = (TextView) findViewById(R.id.textView1);
      try {
         PlayWithRawFiles();

      } catch (IOException e) {
         txtMsg.setText( "Problems: " + e.getMessage() );
      }
   }// onCreate
```

Android Files



Example 1: Reading a Resource File (see previous figure)

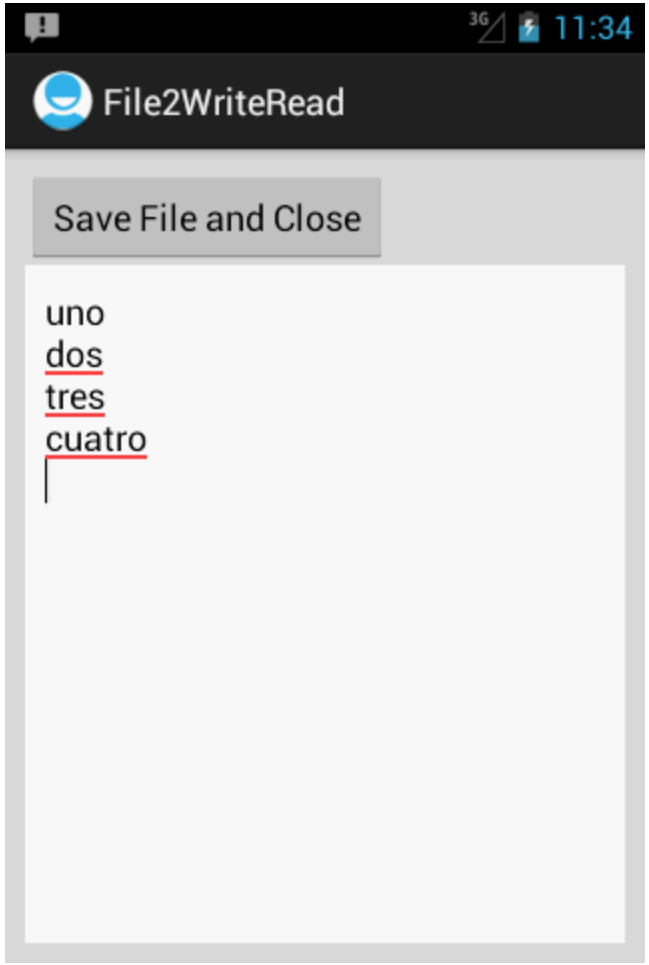
```
   public void PlayWithRawFiles() throws IOException {
      String str="";
      StringBuffer buf = new StringBuffer();

      int fileResourceId = R.raw.my_text_file;
      InputStream is = this.getResources().openRawResource(fileResourceId);
      BufferedReader reader = new BufferedReader(new InputStreamReader(is));

      if (is!=null) {
         while ((str = reader.readLine()) != null) {
              buf.append(str + "\n" );
              }
      }
      is.close();
      txtMsg.setText( buf.toString() );

   }// PlayWithRawFiles

} // File1Resources
```

Android Files

Example 2: **(Internal Storage)** Read/Write an Internal File.

In this example an application collects data from the UI and saves it to a persistent data file into the (limited) internal Android System space area.

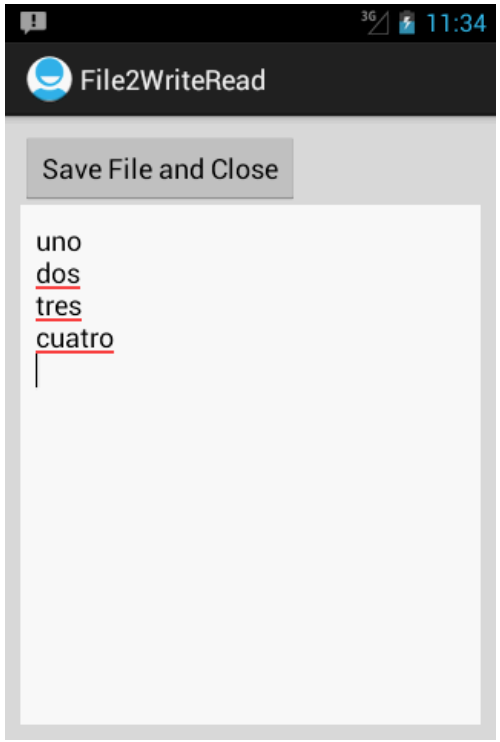
Next time the application is executed the *Resource File* will be read and its data shown on the UI

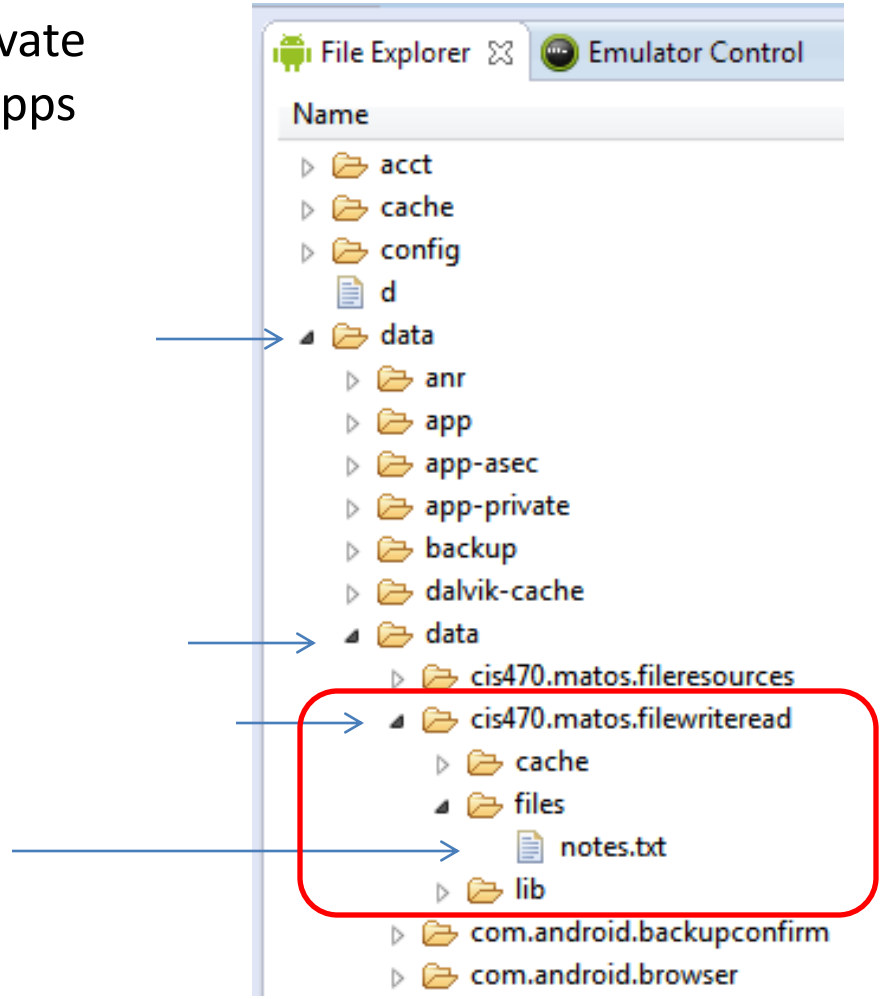


Android Files

Example 2: **(Internal Storage)** Read/Write an Internal File.

The *internal resource file* is private and cannot be seen by other apps residing in main memory.

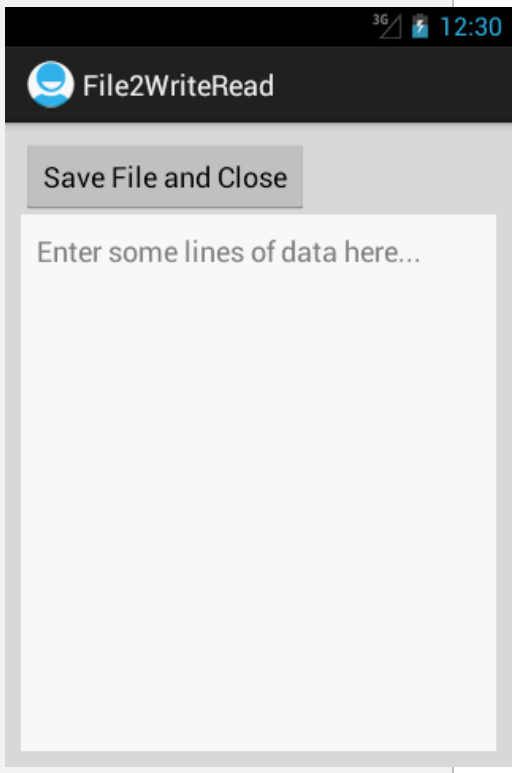




Android Files

Example2: Grab data from screen, save to file, retrieve from file.

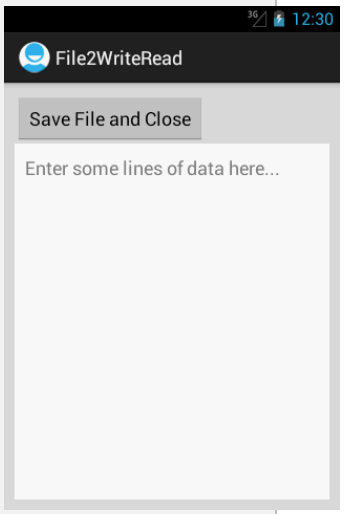
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffdddddd"
    android:padding="10dp"
    android:orientation="vertical" >

    <Button android:id="@+id/btnFinish"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
         android:padding="10dp"
        android:text=" Save File and Close " />

    <EditText
        android:id="@+id/txtMsg"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:padding="10dp"
        android:background="#ffffffff"
        android:gravity="top"
        android:hint="Enter some lines of data here..."  />

</LinearLayout>
```



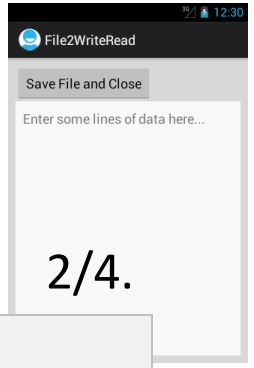
Android Files

Example 2: Grab data from screen, save to file, retrieve from file 1/4.

```
public class File2WriteRead extends Activity {

   private final static String FILE_NAME = "notes.txt";
   private EditText txtMsg;

   @Override
   public void onCreate(Bundle icicle) {
      super.onCreate(icicle);
      setContentView(R.layout.main);
      txtMsg = (EditText) findViewById(R.id.txtMsg);

      // deleteFile();  //keep for debugging

      Button btnFinish = (Button) findViewById(R.id.btnFinish);
      btnFinish.setOnClickListener(new Button.OnClickListener() {
         public void onClick(View v) {
            finish();
         }
      });
   }// onCreate
```



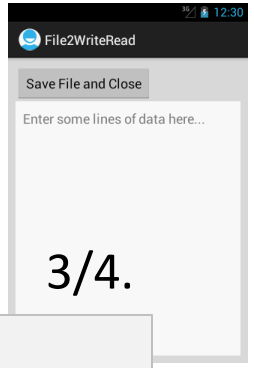
Android Files



Example 2: Grab data from screen, save to file, retrieve from file


```
public void onStart() {
   super.onStart();
   try {
      InputStream inputStream = openFileInput(FILE_NAME);
      if (inputStream != null) {
         InputStreamReader inputStreamReader = new
                                                InputStreamReader(inputStream);
         BufferedReader reader = new BufferedReader(inputStreamReader);
         String str = "READING FROM EXISTING DISK\n";
         StringBuffer stringBuffer = new StringBuffer();

         while ((str = reader.readLine()) != null) {
            stringBuffer.append(str + "\n");
         }

         inputStream.close();
         txtMsg.setText(stringBuffer.toString());
      }
   } catch (java.io.FileNotFoundException e) {

   } catch (Throwable t) {
      Toast.makeText(this, "Exception: " + t.toString(), 1).show();
   }
}// onStart
```

Android Files



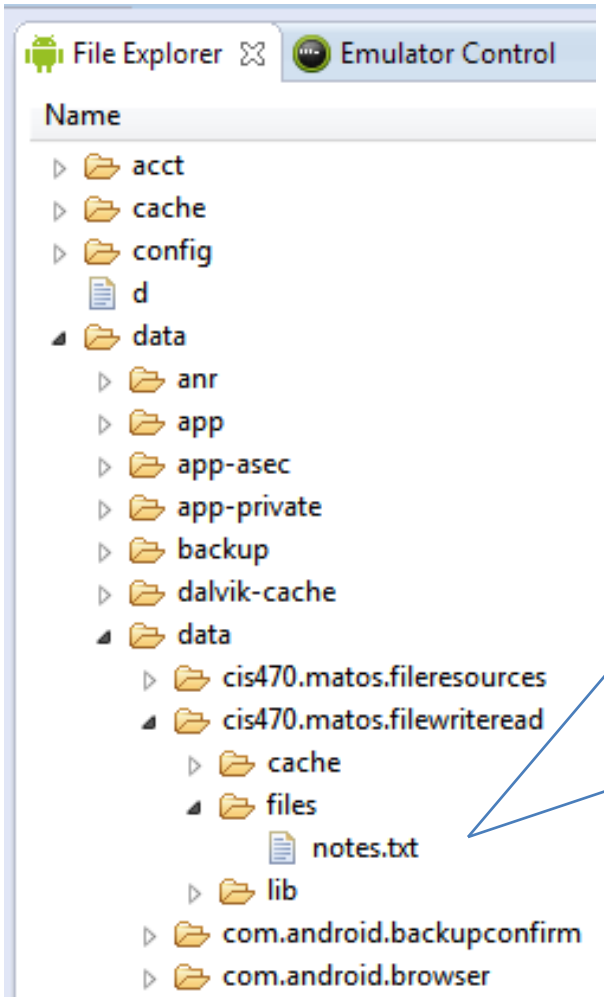
Example 2: Grab data from screen, save to file, retrieve from file


```
public void onPause() {
   super.onPause();
   try {
      OutputStreamWriter out = new OutputStreamWriter(
                                    openFileOutput(FILE_NAME, 0));
      out.write(txtMsg.getText().toString());
      out.close();
   } catch (Throwable t) {
      txtMsg.setText( t.getMessage() );
   }
}// onPause
```

```
private void deleteFile() {
   String path = "/data/data/cis470.matos.filewriteread/files/" + FILE_NAME;
   File f1 = new File(path);
   Toast.makeText(getApplicationContext(), "Exists " + f1.exists() , 1).show();
   boolean success = f1.delete();
   if (!success){
     Toast.makeText(getApplicationContext(), "Deletion failed.", 1).show();
   }else{
      Toast.makeText(getApplicationContext(), "OK. File deleted.", 1).show();
   }
}
```

Android Files

In our example the **notes.txt** file is stored in the phone's internal memory under the name: **/data/data/cis470.matos.fileresources/files/notes.txt**



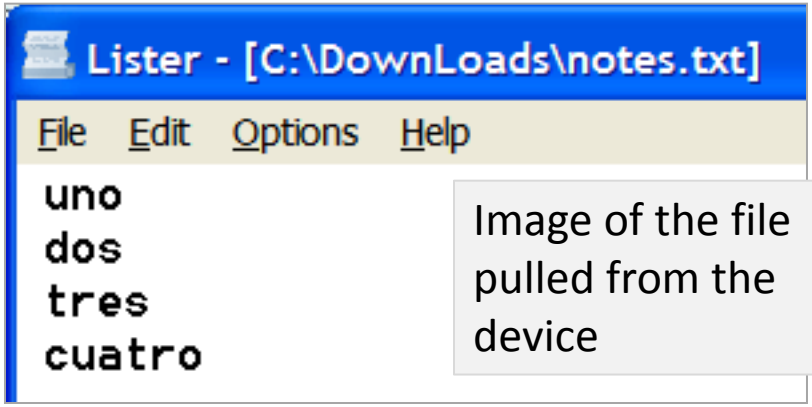


Image of the file pulled from the device

Android Files

Example 3: **(External Storage)**

Reading/Writing to the External Device's **SD card**.

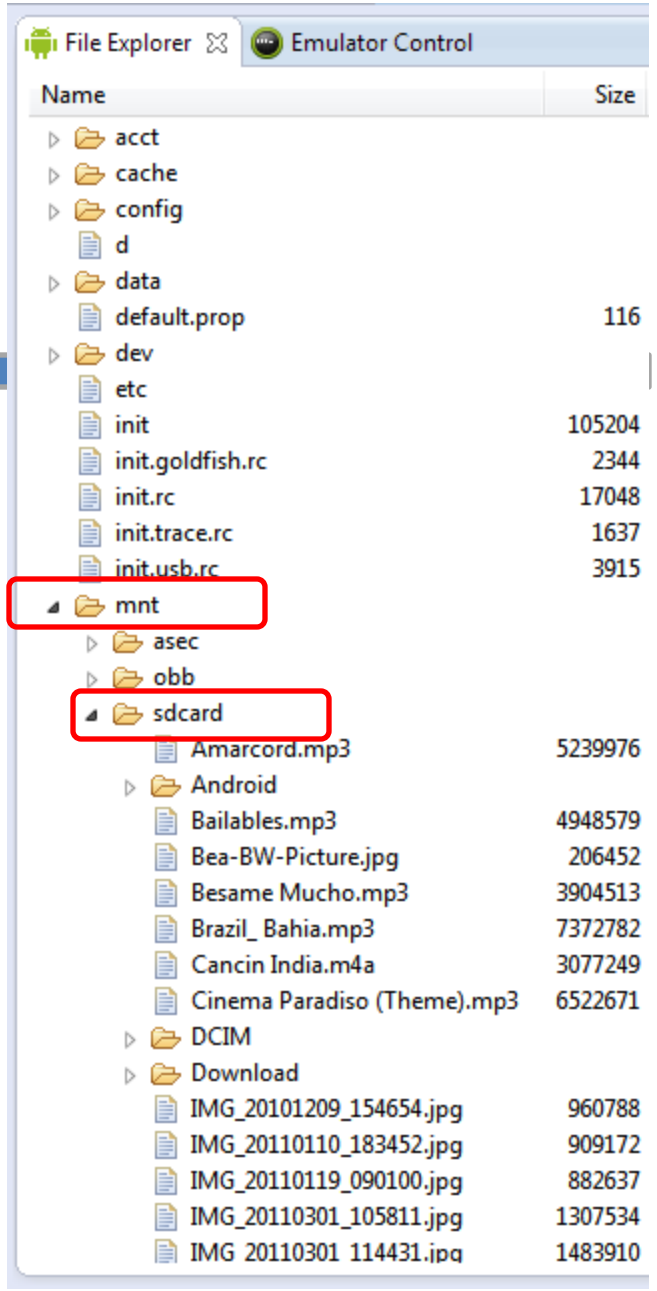
SD cards offer the advantage of a much larger capacity as well as portability (usually cards can be easily removed from one device and reused in another)

Android Files

Example 3: (External Storage)

Reading/Writing to the External Device's **SD card**.

Use **File Explorer** tool to locate files in your device (or emulator)



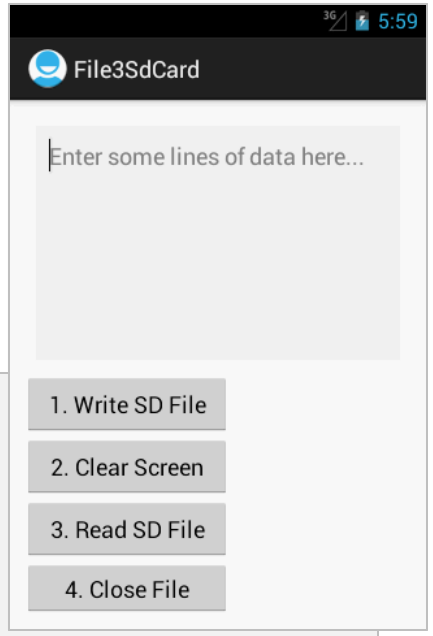
Android Files

WARNING: Reading/Writing to the Device's **SD card**

When you deal with external files you need to request permission to read and write to the SD card. Add the following clauses to your AndroidManifest.xml

```
<uses-permission
        android:name="android.permission.READ_EXTERNAL_STORAGE"/>

<uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

Android Files

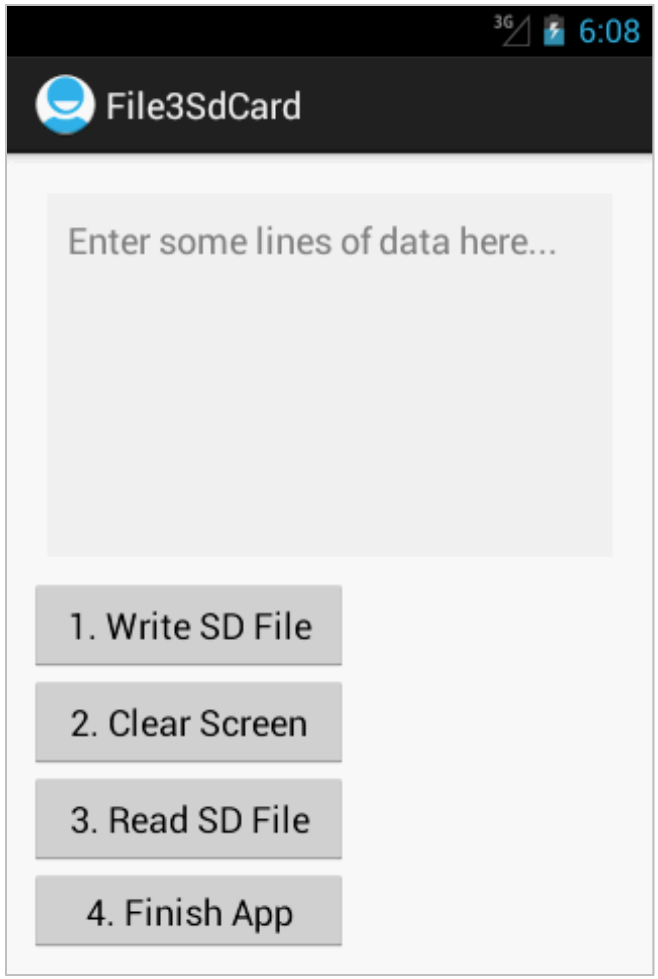


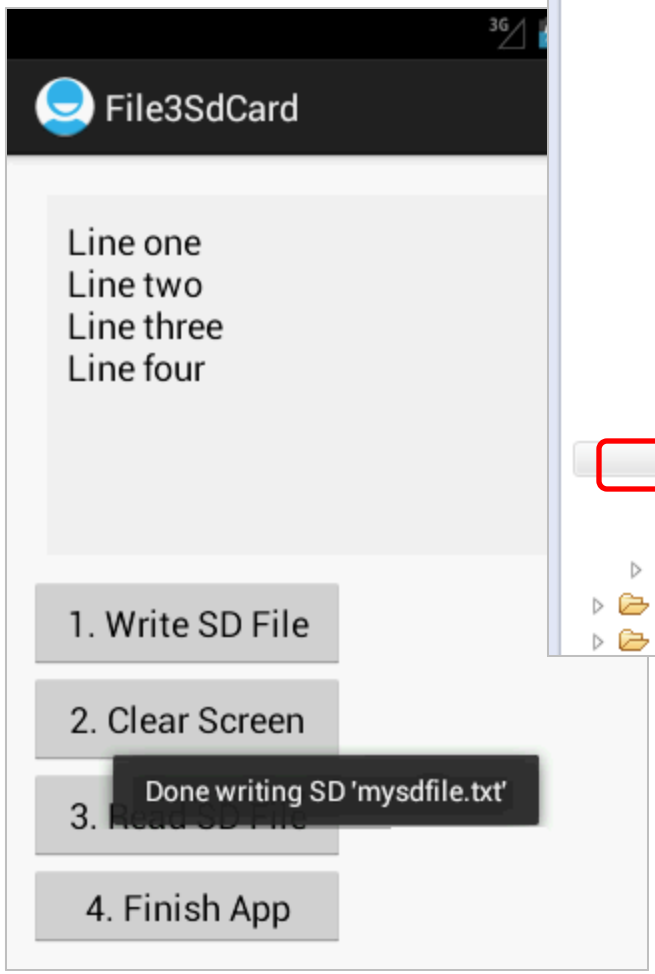
Example 3: Reading/Writing to the Device's SD card.

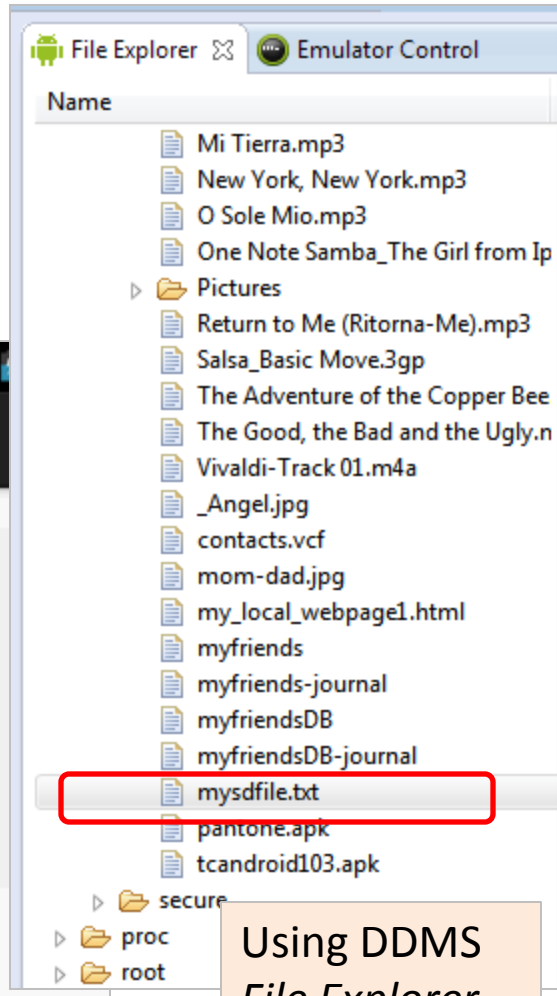
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/widget28"
    android:padding="10dp"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <EditText
        android:id="@+id/txtData"
        android:layout_width="match_parent"
        android:layout_height="180dp"
        android:layout_margin="10dp"
        android:background="#55dddddd"
        android:padding="10dp"
        android:gravity="top"
        android:hint=
       "Enter some lines of data here..."
        android:textSize="18sp" />
    <Button
        android:id="@+id/btnWriteSDFile"
        android:layout_width="160dp"
        android:layout_height="wrap_content"
        android:text="1. Write SD File" />
    <Button
        android:id="@+id/btnClearScreen"
        android:layout_width="160dp"
        android:layout_height="wrap_content"
        android:text="2. Clear Screen" />
    <Button
        android:id="@+id/btnReadSDFile"
        android:layout_width="160dp"
        android:layout_height="wrap_content"
        android:text="3. Read SD File" />
    <Button
        android:id="@+id/btnFinish"
        android:layout_width="160dp"
        android:layout_height="wrap_content"
        android:text="4. Finish App" />
</LinearLayout>
```

Android Files

Example 3: Reading/Writing to the Device's SD card.







Using DDMS *File Explorer* panel to inspect the SD card.

Android Files

Example 3: Reading/Writing to the Device's SD card.

```
public class File3SdCard extends Activity {
// GUI controls
EditText txtData;
Button btnWriteSDFile;
Button btnReadSDFile;
Button btnClearScreen;
Button btnFinish;

@Override
public void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.main);

     // bind GUI elements with local controls
     txtData = (EditText) findViewById(R.id.txtData);
     txtData.setHint("Enter some lines of data here...");
```

Android Files

Example 3: Reading/Writing to the Device's SD card.

```
btnWriteSDFile = (Button) findViewById(R.id.btnWriteSDFile);

btnWriteSDFile.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
    // write on SD card file data from the text box
    try {
        File myFile = new File("mnt/sdcard/mysdfile.txt");
        myFile.createNewFile();
        FileOutputStream fOut = new FileOutputStream(myFile);
        OutputStreamWriter myOutWriter = new OutputStreamWriter(fOut);

        myOutWriter.append(txtData.getText());
        myOutWriter.close();
        fOut.close();
        Toast.makeText(getApplicationContext(),
                    "Done writing SD 'mysdfile.txt'",
                    Toast.LENGTH_SHORT).show();
    } catch (Exception e) {
        } Toast.makeText(getApplicationContext(),
                    e.getMessage(), Toast.LENGTH_SHORT).show();

    }// onClick
}); // btnWriteSDFile
```

Android Files

Example 3: Reading/Writing to the Device's SD card.

```
btnReadSDFile = (Button) findViewById(R.id.btnReadSDFile);
btnReadSDFile.setOnClickListener(new OnClickListener() {
 @Override
 public void onClick(View v) {
 // write on SD card file data from the text box
 try {
       File myFile = new File("mnt/sdcard/mysdfile.txt");
       FileInputStream fIn = new FileInputStream(myFile);
       BufferedReader myReader = new BufferedReader(new InputStreamReader(fIn));
       String aDataRow = "";
       String aBuffer = "";
       while ((aDataRow = myReader.readLine()) != null) {
          aBuffer += aDataRow + "\n";
       }
       txtData.setText(aBuffer);
       myReader.close();
       Toast.makeText(getBaseContext(),
                     "Done reading SD 'mysdfile.txt'", 1).show();
       } catch (Exception e) {
           Toast.makeText(getBaseContext(), e.getMessage(), 1).show();
       }
 }// onClick
}); // btnReadSDFile
```

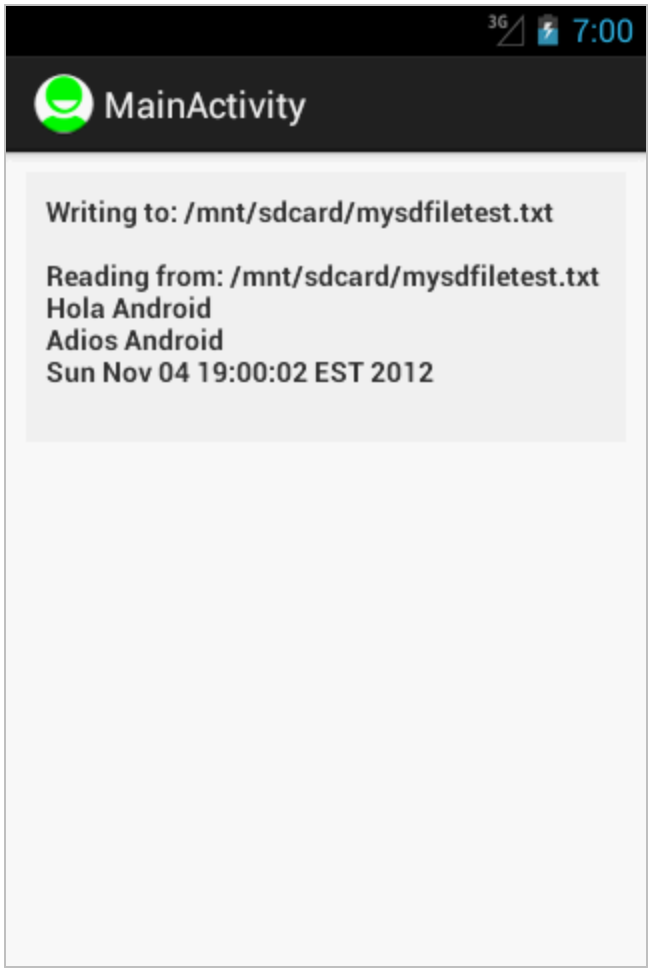
Android Files

Example 3: Reading/Writing to the Device's SD card.

```
 btnClearScreen = (Button) findViewById(R.id.btnClearScreen);
 btnClearScreen.setOnClickListener(new OnClickListener() {
     @Override
     public void onClick(View v) {
          // clear text box
          txtData.setText("");
     }
 }); // btnClearScreen

 btnFinish = (Button) findViewById(R.id.btnFinish);
 btnFinish.setOnClickListener(new OnClickListener() {
     @Override
     public void onClick(View v) {
          finish();
     }
 }); // btnFinish

 }// onCreate

}// class
```

Android Files

Example 4: Reading/Writing to the Device's SD card through the Scanner and PrintWriter classes. 1/4

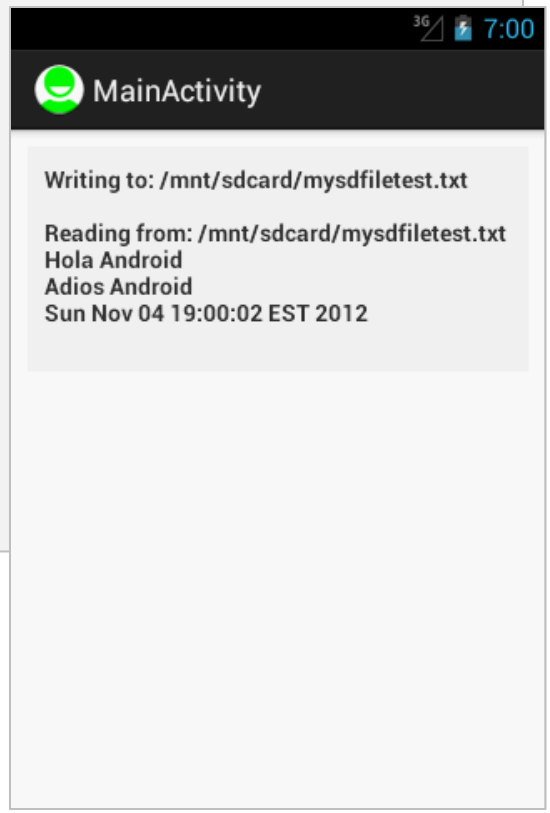



```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_margin="10dp"
    >

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:padding="10dp"
    android:id="@+id/txtMsg"
    android:textStyle="bold"
    android:background="#77eeeeee"
    />
</LinearLayout>
```

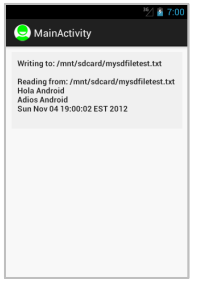
Android Files

Example 4: Reading/Writing to the Device's SD card through the Scanner and PrintWriter classes. 2/4

```
public class File4Scanner extends Activity  {

TextView txtMsg;

  @Override
  public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);
      txtMsg = (TextView) findViewById(R.id.txtMsg);

      testScannerFiles();

  }//onCreate
```



Android Files



Example 4: Reading/Writing to the Device's SD card through the Scanner and PrintWriter classes. 3/4

```
private void testScannerFiles(){
// Add to manifest the following permission request
// <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
   try {
      String SDcardPath = Environment.getExternalStorageDirectory().getPath();
      String mySDFileName = SDcardPath + "/" + "mysdfiletest.txt";

      txtMsg.setText("Writing to: " + mySDFileName);

      PrintWriter outfile= new PrintWriter( new FileWriter(mySDFileName) );

         outfile.println("Hola Android");
         outfile.println("Adios Android");
         outfile.println(new Date().toString());

         outfile.close();
```

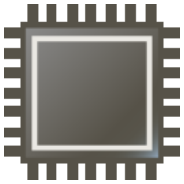
Android Files

Example 4: Reading/Writing to the Device's SD card through the Scanner and PrintWriter classes. 4/4

```
   // read SD-file,show records.
   // <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />

         Scanner infile= new Scanner(new FileReader(mySDFileName));
         String inString= "\n\nReading from: " + mySDFileName + "\n";

         while(infile.hasNextLine()) {
            inString += infile.nextLine() + "\n";
         }

         txtMsg.append(inString);
         infile.close();

      } catch (FileNotFoundException e) {
         txtMsg.setText( "Error: " + e.getMessage());
      } catch (IOException e) {
         txtMsg.setText( "Error: " + e.getMessage());
      }

   }//testScannerFiles

}//class
```

Files

Questions ?



Icon obtained from: http://www.iconseeker.com

Files

Appendix A.

Accessing a file in the SD card

```
String stringPath = Environment
                        .getExternalStorageDirectory()
                        .getAbsolutePath()
                        + "/myFileNameGoesHere.txt";
```