

COMP30510 Mobile Application Development

Storage

Dr. Abraham Campbell
University College Dublin
Abey.campbell@ucd.ie

Four Ways to CRUD* Data

- Preferences
- SQLite
- Plain files
 - Plain files on SD Card
- Network

* - Create Read Update Delete

SharedPreferences Framework

- Provides a mechanism to show, save and manipulate user's preferences
- Supports automatic UI creation
 - Developer declares the type of a user preference
=> a UI for manipulating these preferences is automatically generated

SharedPreferences Framework

- `getPreferences()` - Activity-specific
- `getSharedPreferences()` – Applications specific
- `getDefaultSharedPreferences()` - shared system preferences

SharedPreferences

- `edit()`
- `remove()`
- `Clear()`
- `AND`
- `commit()`

SharedPreferences (committing)

```
SharedPreferences settings =  
getSharedPreferences("myPrefs", 0);
```

```
SharedPreferences.Editor editor = settings.edit();
```

```
editor.putString("username", "Abey");  
editor.commit();
```

SharedPreferences (Restoring)

```
SharedPreferences settings =  
getSharedPreferences("myPrefs", 0);
```

```
String name = settings.getString(("username",  
"Not Stored"));
```

Defining Preferences (Graphical)

```
<PreferenceScreen ...>
```

```
<CheckBoxPreference
```

```
  android:key="checkbox"
```

```
  android:title="Checkbox Preference"
```

```
  android:summary="Check it on, check  
it off" />
```

```
...
```


Adding Preferences

```
public class EditPreferences extends PreferenceActivity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        addPreferencesFromResource(R.xml.preferences);
    }
}
```

Plain Files

- Android has a private storage for each application, which is accessible for both read and write operations
- Also, if a device has external storage (SD Card) and application has appropriate privileges

Plain Files: Available Classes

- General
 - InputStream
 - OutputStream
- Text-Based
 - InputStreamReader
 - OutputStreamWriter
- *Don't forget to close() :-)*
- File paths are NOT accepted!

Working with Plain Files

// Example 1

```
InputStream in = getResources()\n    .openRawResource(R.raw.words);
```

// Example 2

```
String FILENAME = "hello_file";  
String string = "hello world!";  
FileOutputStream fos = openFileOutput(FILENAME, \  
    Context.MODE_PRIVATE);  
fos.write(string.getBytes());  
fos.close();
```

Working with External Storage

- Use
Environment.getExternalStorageDirectory()
for root element
- You CAN use file paths there
- External storage is accessible to all
applications (with appropriate permissions)

SQLite

- Popular embedded database
- Combines SQL interface & small memory footprint
- Open Source and Public Domain
- Built into Android runtime
- SQL Primer is available at http://www.w3schools.com/web/web_sql.asp

SQLite in Android

- Create & open a DB with subclass of `SQLiteOpenHelper()`
- Methods needed:
 - Constructor
 - `onCreate()`
 - `onUpgrade()`
- Use its instance with `getReadableDatabase()` or `getWritableDatabase()`
- Don't forget to `close()` :-)

Android's SQLite Cont'd

- `db.execSQL("CREATE TABLE constants (_id INTEGER PRIMARY KEY AUTOINCREMENT, title TEXT, valueREAL);");`
 - Same for INSERT, UPDATE, DELETE
- Alternatively, use `insert()`, `update()`, `delete()` on `SQLiteDatabase` object:
 - `db.insert("constants", "title", values);`
`constantsCursor.rawQuery();`

Android's SQLite Cont'd

- Other methods available:
 - `rawQuery()` - simply calls your SQL statement
 - `query()` - lets you construct SQL statement from components with `SQLiteQueryBuilder`
- Cursors are also available, i.e.
 - `getCount()`, `moveToFirst()`, `moveToNext()`, `getColumnNames()`, `getColumnIndex()`, `getString()`, `getInt()`, etc.

Android's SQLite Cont'd

- DB inspection is available via adb shell & sqlite3
- SQLite files are available at
`/data/data/your.app.package/databases/your
dbname`
- To get DB in and out, use 'adb pull' and 'adb
push'

Network

- Most Android devices have some sort of network connectivity
 - 3G, EDGE, Wifi, etc
- Wide range of connectivity available:
 - Integrated WebKit browser
 - Various built-in and 3rd party API
 - Raw sockets

Network

- Android does not have built-in SOAP or XMLRPC, but has built-in Apache HTTP Components library (<http://hc.apache.org>)
- HTTP Components basic classes & methods:
 - `HttpClient.execute()`, `HttpRequest` (`HttpGet` & `HttpPost`), `ResponseHandler<>`

Sending Requests

```
HttpGet getMethod=new HttpGet(url);  
try {  
    ResponseHandler<String> responseHandler =  
    new BasicResponseHandler();  
    String responseBody =  
    client.execute(getMethod,  
    ResponseHandler);  
    ...  
} catch (Throwable t) {  
    ...
```

Parsing Responses

- Data will come back to you mainly in HTML, XML, JSON
- Android has 3 main built-in data parsers:
 - DOM (org.w3c.dom), SAX (org.xml.sax), JSON(org.json)
- Nothing stops you from writing your own :-)
- 3rd party parsers are required for RSS/Atom

Parsing Responses Cont'd

...

```
DocumentBuilder builder = DocumentBuilderFactory.  
newInstance().newDocumentBuilder();  
Document doc = builder.parse(new InputSource(new  
StringReader(raw)));  
NodeList elements =  
doc.getElementsByTagName("ourelements");  
for (int i=0;i<elements.getLength();i++) {  
Element elem=(Element)elements.item(i);
```

...

HTTP Components Considerations

- Using HttpClient with SSL is not an easy task due to ambiguity in certificate handling
- HttpClient is single-threaded (so new thread might be needed for each one)
- More info: docs, examples at <http://hc.apache.org>