Androidアプリ塾

実装編

アプリ開発で学ぶ内容

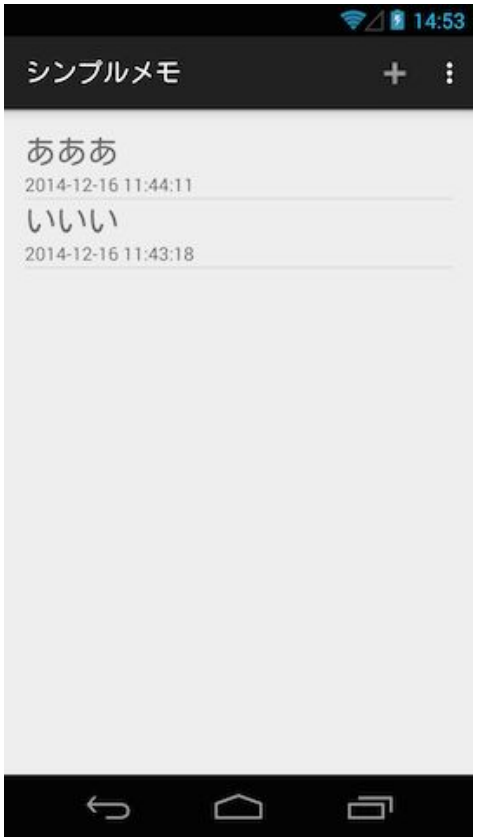
- 画面のレイアウト
- ListViewの使い方
- メニューの使い方
- 画面遷移
- 画面部品のイベント処理
- データベースへの入出力
- Activityのライフサイクル
- アプリの実行・デバッグ

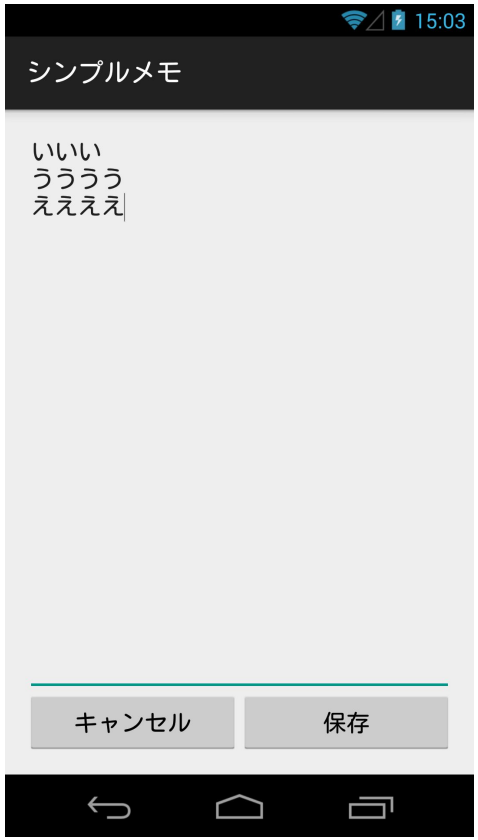
開発のおおまかな流れ

1. レイアウトを作成
2. 必要なリソースを用意
3. 画面要素の表示
4. イベント処理の実装
5. エミュレータまたは実機で実行

シンプルなメモアプリを作成

メモ一覧画面とメモ編集画面の2画面構成





一覧画面の概要

- View
  - 画面レイアウトにListViewを配置
  - 行を表現するViewを定義
- Activity
  - 一覧データ読み込み
  - アダプタを生成
  - アダプタに一覧データを渡す
- アダプタクラス
  - getViewメソッドをオーバーライドし行データを表示

一覧画面のレイアウト定義

- 一覧画面（activity_main.xml）
  - ListViewを配置
- 行のView（row_memo.xml）
  - メモを表示するTextViewを配置
  - 更新日時を表示するTextViewを配置

リソースファイルを準備

- メニュー定義ファイル
  - menu_main.xml
- 文字列リソースファイル
  - string.xml（日本語用はvalues-jaに配置）
- メニューアイコン用画像ファイル

Memoクラスの実装

メモを表現するクラスを作成

- フィールド
  - ID
  - メモ
  - 更新日時

Memo一覧用アダプタクラスの実装

メモ一覧データをListViewに渡すアダプタクラス

- メモ一覧のデータを保持
- 行のレイアウトを保持
- getViewメソッドをオーバーライド
  - メモを表示
  - 更新日時を表示

MainActivityクラスの実装

一覧画面のActivityクラス

- レイアウトからListViewを取得
- アダプタクラスを作成
- メモデータの読み込み
  - この段階ではデータベースを使わずにロジックで仮実装

エミュレータで動作を確認

これまでの実装で仮のメモデータが表示される

編集画面の概要

メモ新規作成と修正の両方を実装

- View
  - 画面レイアウトにEditTextとButtonを配置
- Activity
  - 一覧画面から渡されたモード（新規or修正）を取得
  - ボタンクリック時のイベント処理を実装

画面遷移について

新規メモ編集時と既存メモ修正時に一覧画面から編集画面に遷移する

- 新規メモ
  - ActionBarの新規メモメニュークリック時
- 既存メモ修正
  - メモ一覧で行タップ時

※メモ一覧画面からメモ編集画面を startActivityで起動すると、メモ一覧画面の上にメモ編集画面が覆いかぶさる

編集画面のレイアウト定義

- 編集画面（activity_edit.xml）
  - EditTextを配置
  - キャンセルボタンと保存ボタンを配置
  - 各部品の相対位置を指定してレイアウトを整える
  - 文字列リソースファイルにボタンに表示する文字列を追加

DBアダプタークラスを作成する

- SQLiteの入出力を定義

EditActivityクラスの実装

- EditActivityクラスを新規作成する

  ※Activityを新たに作成した場合はマニュフェストファイルに追加する必要がある
- レイアウトから各部品を取得
- 遷移元から渡されたモードを判定し、適切な初期表示を行う
- キャンセルボタンのイベントリスナーを登録する
  - Activityを終了する
- 保存ボタンのイベントリスナーを登録する
  - モードに応じ、データベースの入出力を行う
  - Activityを終了する

MainActivityクラスの修正

- メモデータをデータベースから読込むよう修正
- 一覧画面から編集画面へ遷移する
  - ActionBarの新規メモクリック時
    - インテントに「新規モード」を設定してstartActivity
  - ListViewの行クリック時
    - インテントに「更新モード」を設定してstartActivity

さらに修正

- コンテキストメニューを作成し、メモを削除できるようにする
- メモを作成・修正した内容を一覧に即座に反映

参考

- イベントリスナーとは
  - 「ボタンがクリックされた」などのイベント発生時に対応する処理を記述するクラス
  - Androidの場合、ボタンなどのコンポーネント毎にインターフェースが用意されているので、それを実装したクラスで処理を記述する
- イベントリスナーの様々な記述方法
  - Activityクラスをイベントリスナーとする
  - イベントリスナーを別クラスにする
  - イベントリスナーを無名クラスする