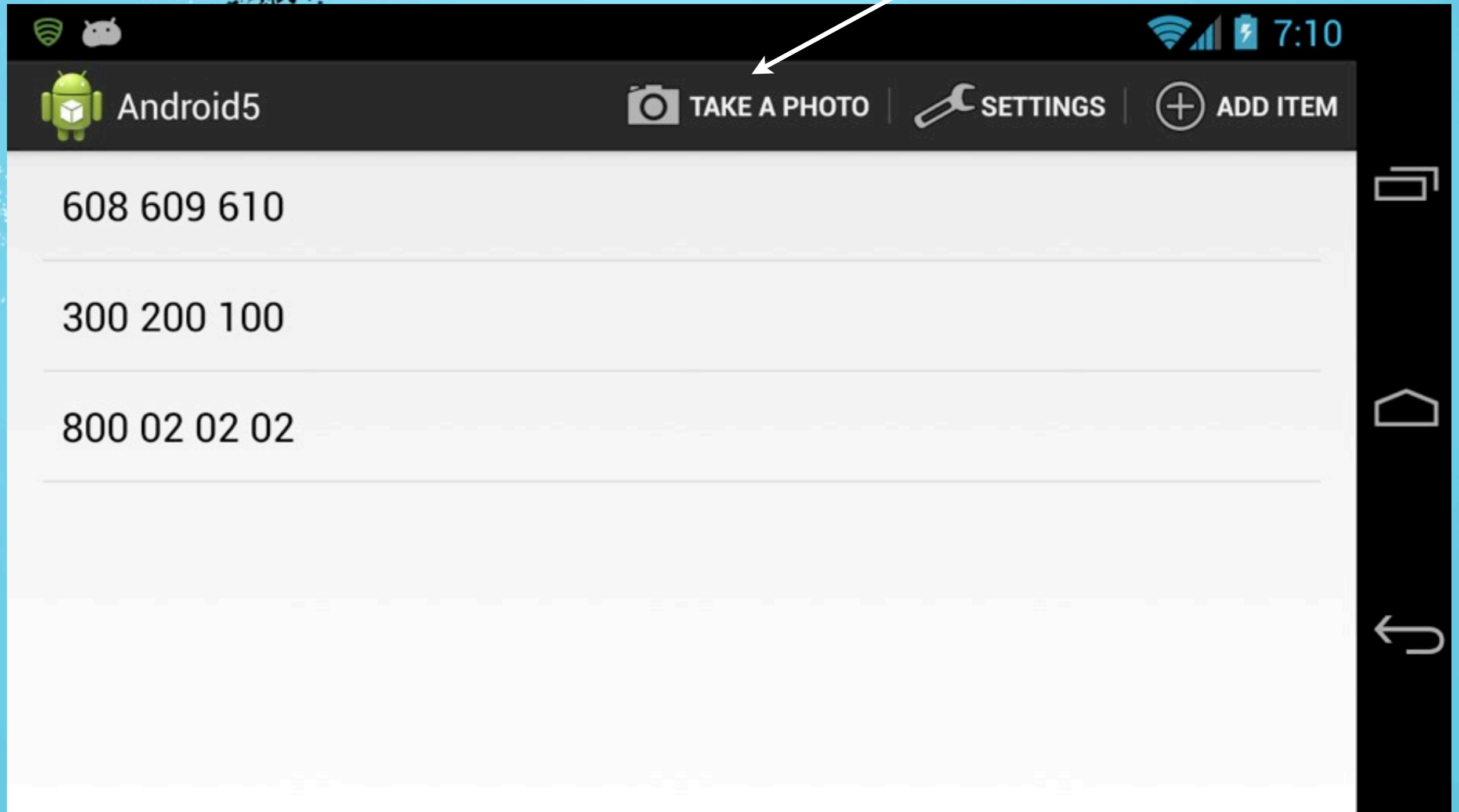


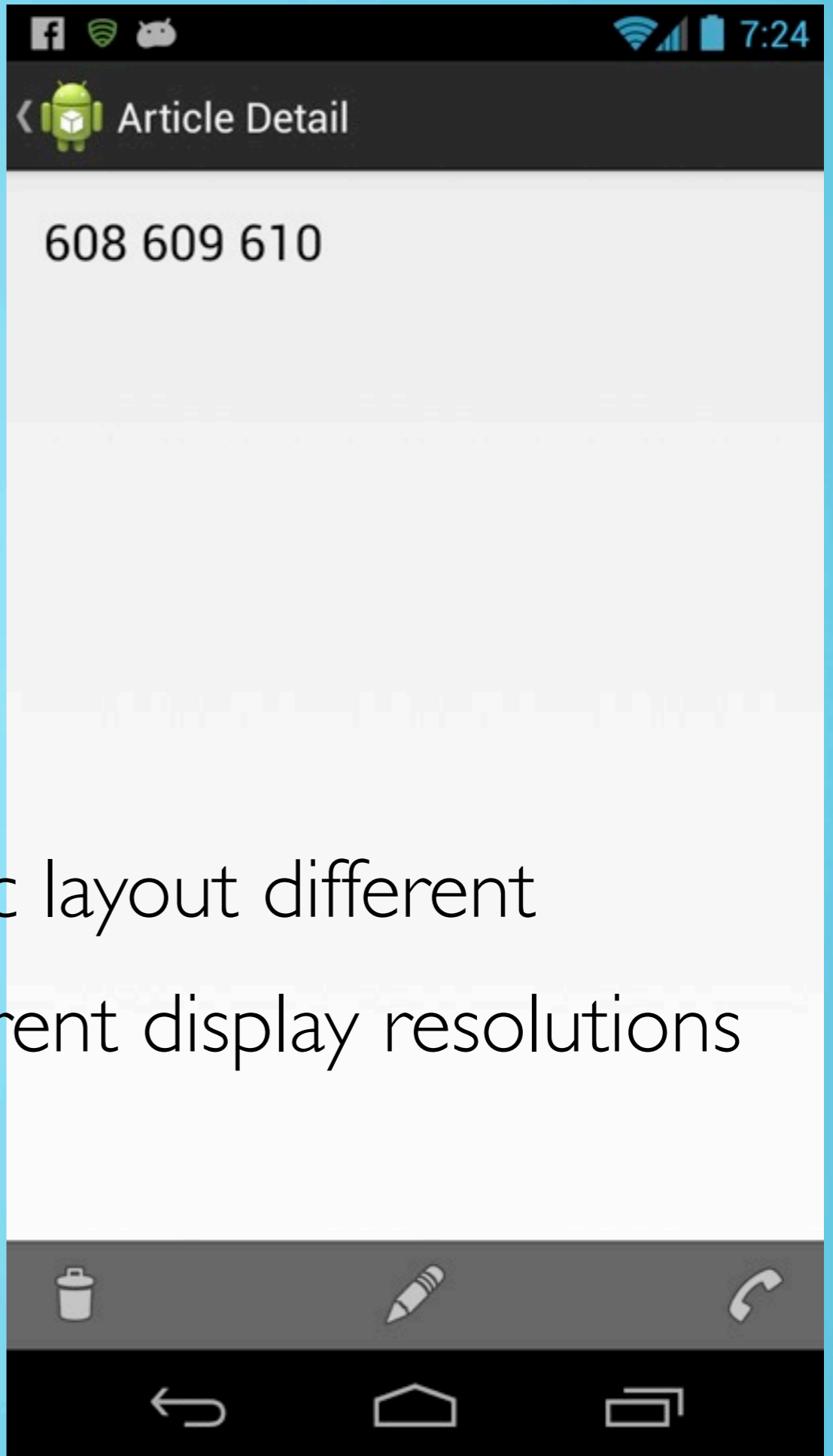


ANDROID DEVELOPMENT #5

@brmlab

ACTIONBAR





FRAGMENTS

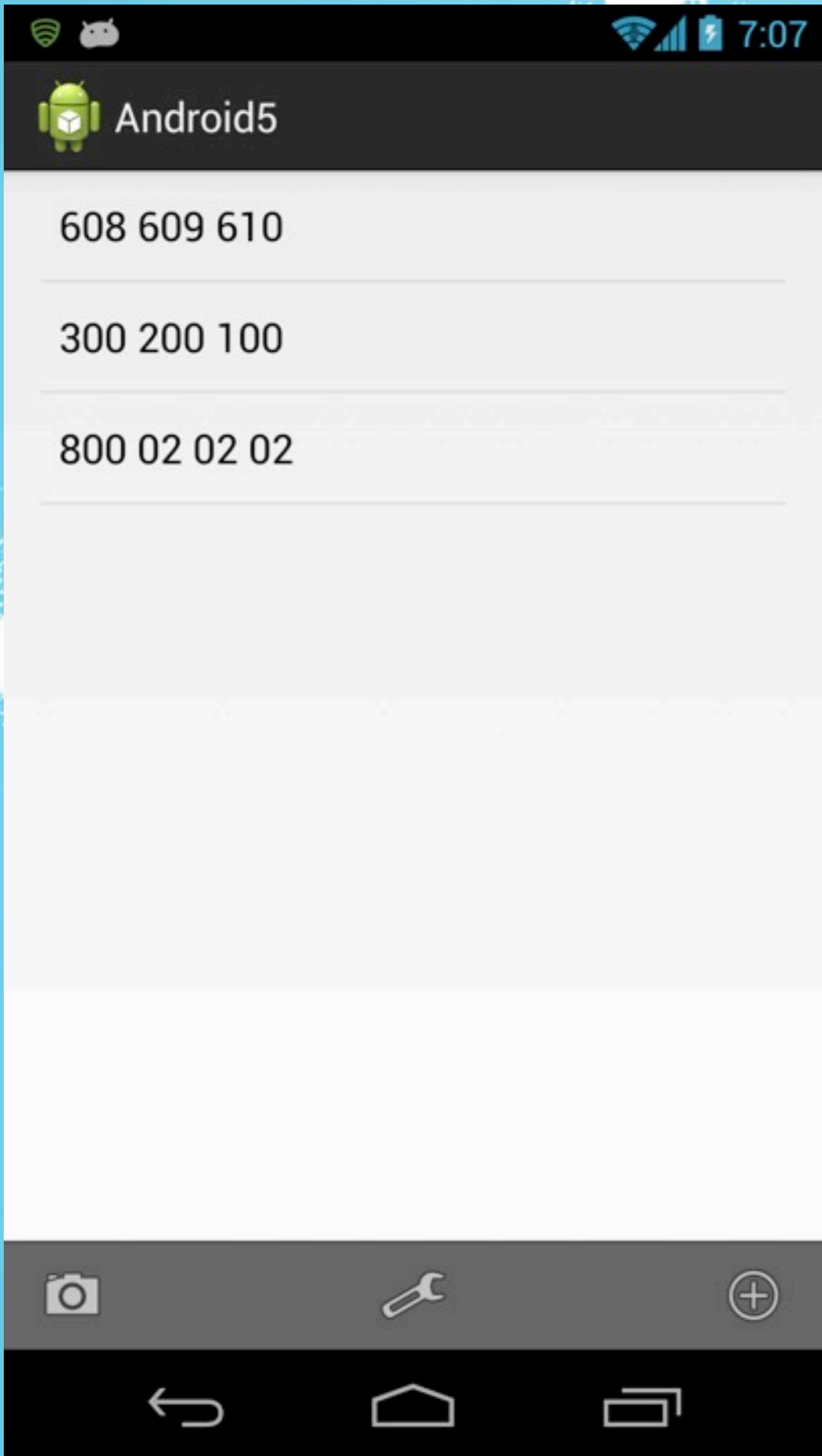
Dynamic layout different
for different display resolutions

MASTER DETAIL FLOW

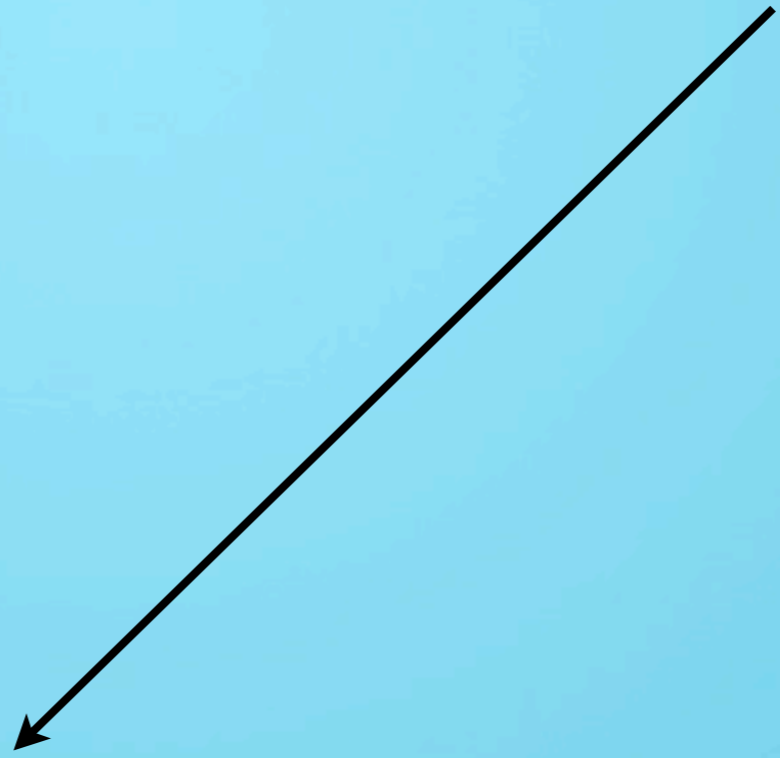
The screenshot displays an Android application interface. At the top, the status bar shows the time as 7:16 and signal strength. Below the status bar, the app's name "Android5" is visible next to the Android logo. A navigation bar contains four icons: a camera for "TAKE A PHOTO", a wrench for "SETTINGS", a plus sign for "ADD ITEM", and a trash can for "REMOVE ITEM".

The main content area shows a list of items. The first item, "608 609 610", is selected and highlighted in blue. To its right, a larger detail view displays the same text "608 609 610". Below the selected item, two other items are visible: "300 200 100" and "800 02 02 02".

At the bottom of the screen, the Android navigation bar is visible with icons for back, home, and recent apps.



ACTIONBAR SPLIT



ACTIONBAR SPLIT

- Split is system auto-magic

- AndroidManifest -> Activity

- `android:uiOptions="splitActionBarWhenNarrow"`

- Showing on small display resolution

- Display without text labels

- Long-press causes Toast notification with item label

ACTIONBAR SPLIT

On Split

• Top

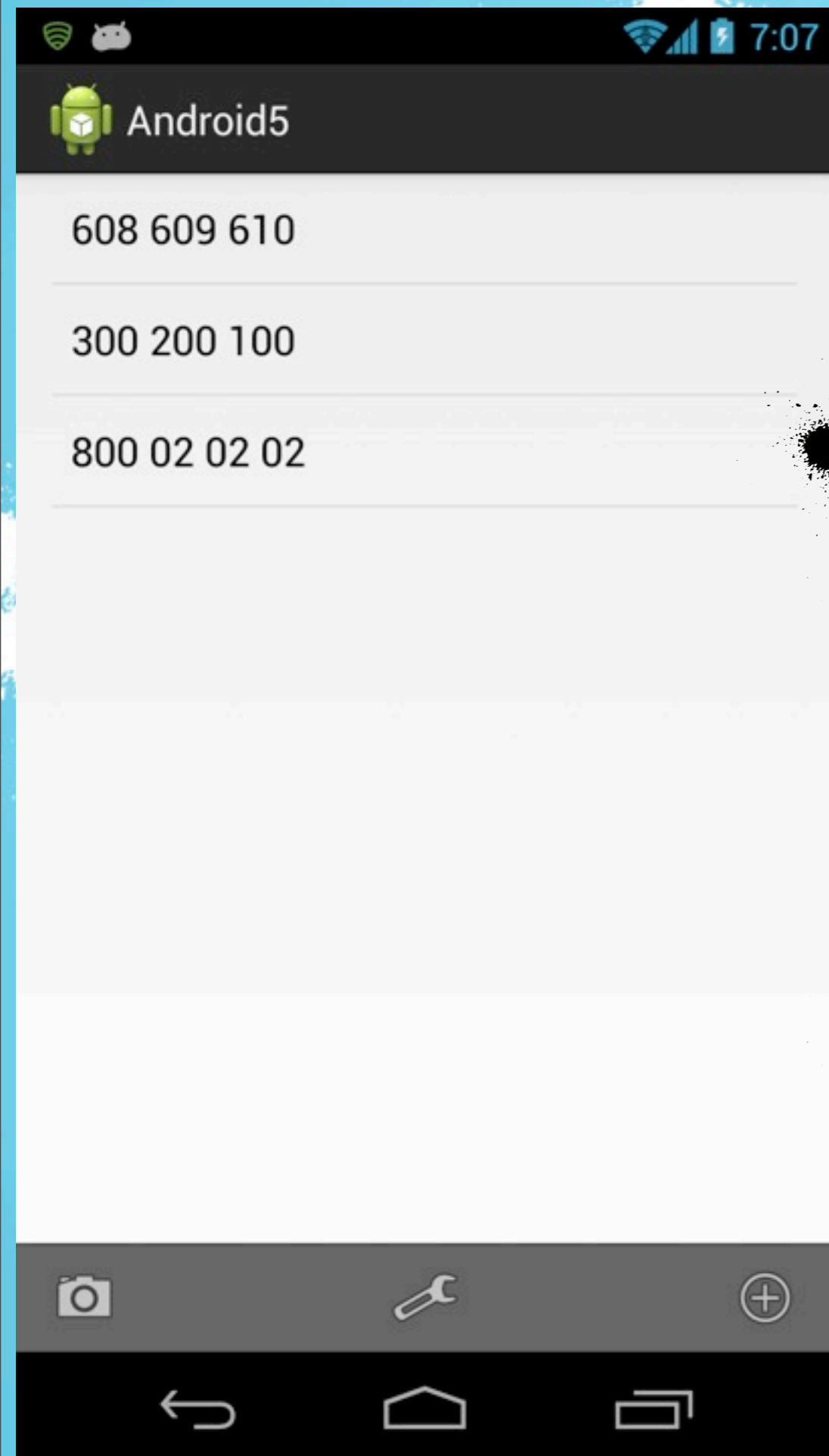
• NavigationMode

• Activity Title

• Home Button

• Bottom

• Menu items + Menu overflow



OPTIONS MENU

The screenshot displays an Android 5.0 interface. At the top, there is a status bar with the text "Android5" on the left and "3G 7:15" on the right. Below the status bar is a dark navigation bar containing three icons: a camera icon labeled "TAKE A PHOTO", a wrench icon labeled "SETTINGS", and a plus sign icon labeled "ADD ITEM". The main content area is a light gray background with a vertical line on the left side, creating a list of three items. Each item consists of a number followed by a space and then another number. The numbers are: "608 609 610", "300 200 100", and "800 02 02 02". At the bottom of the screen is a black navigation bar with three white icons: a back arrow, a home house, and a recent apps task switcher.

608 609 610
300 200 100
800 02 02 02

OPTIONS MENU

Activity

- onCreateOptionsMenu(Menu)
- onOptionsItemSelected(MenuItem)

Fragment

- onCreateOptionsMenu(Menu, MenuInflater)
- onOptionsItemSelected(MenuItem)
- setHasOptionsMenu(boolean)

OPTIONSMENU CREATE

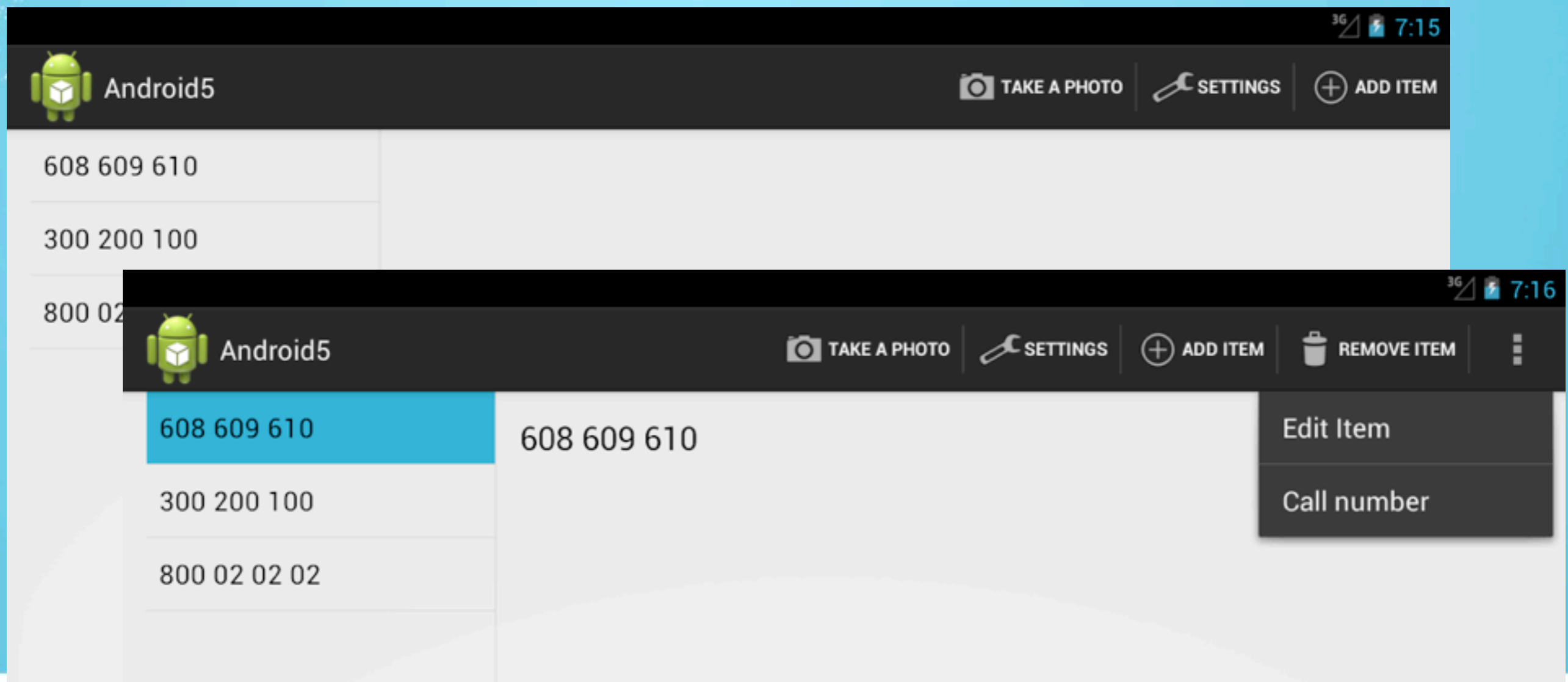
```
@Override  
public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {  
    menu.add(Menu.NONE, MenuConstants.MENU_ADD, Menu.NONE, "Add Item")  
        .setIcon(android.R.drawable.ic_menu_add)  
        .setShowAsActionFlags(  
            MenuItem.SHOW_AS_ACTION_IF_ROOM  
            | MenuItem.SHOW_AS_ACTION_WITH_TEXT);  
}
```

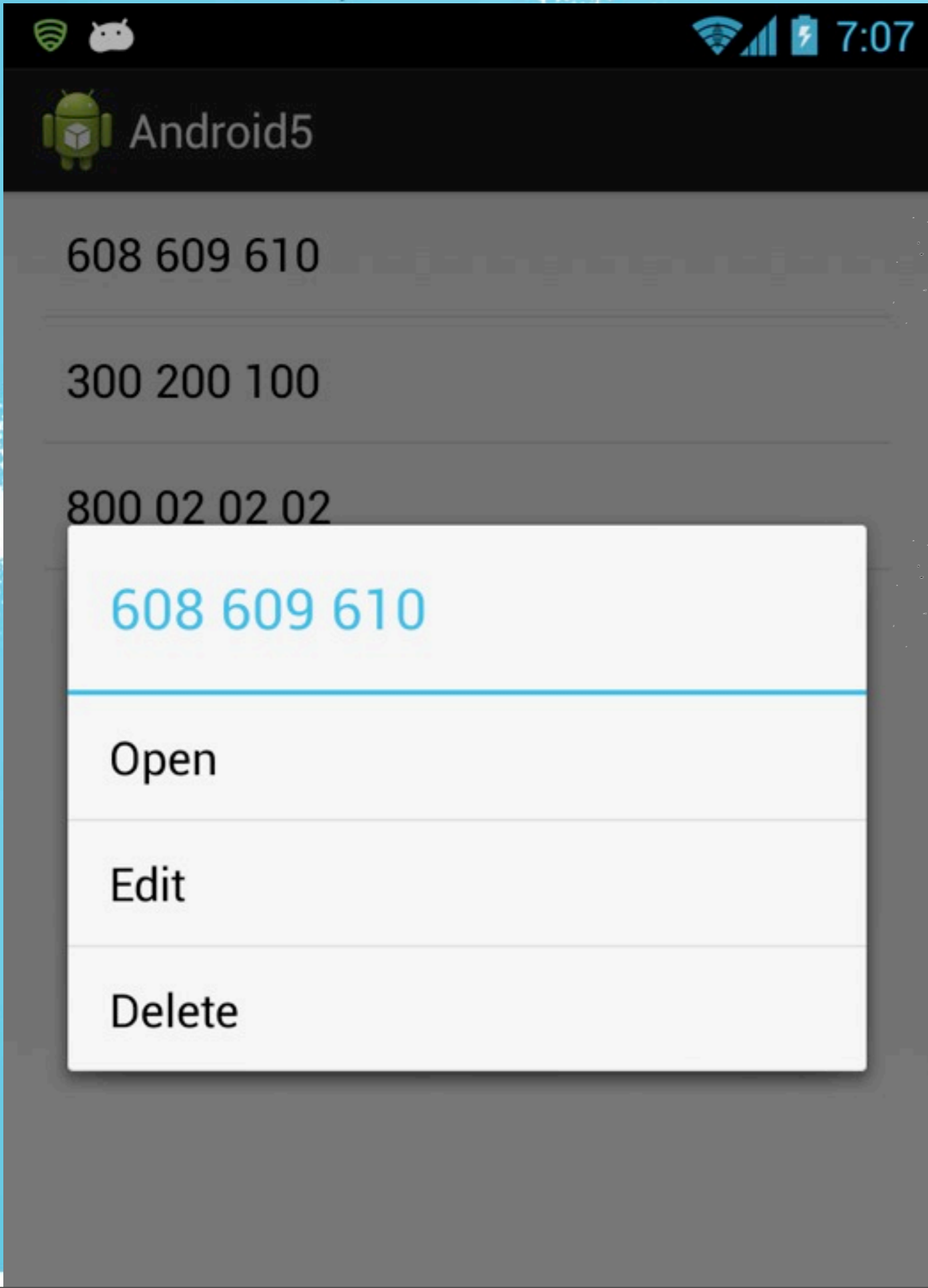
OPTIONSMENU SELECT

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case MenuConstants.MENU_ADD:
            Toast.makeText(getActivity(), "Přidat novou položku",
                Toast.LENGTH_LONG).show();
            return true;
    }
    return false;
}
```

MENU CHAINING

Menu from all Fragments is passed to
FragmentActivity when Fragment visible





CONTEXT MENU

design pattern for long-press event

registerForContextMenu (ListView)

ListActivity

ListFragment

Activity

Fragment

CONTEXT MENU

onCreateContextMenu
(ContextMenu, View, ContextMenuItem)

onContextItemSelected(MenuItem)

CONTEXTMENU CREATE

- Very similar to creating standard Options Menu
- Can alter / setTitle for pop-up dialog
- Retrieving item using ContextMenuInfo

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
    ContextMenuInfo menuInfo) {
    AdapterView.AdapterContextMenuInfo info = (AdapterView.AdapterContextMenuInfo) menuInfo;
    DummyItem item = (DummyItem) getListAdapter().getItem(info.position);
    if (item != null) {
        menu.setHeaderTitle(item.content);
        menu.add(Menu.NONE, 2, Menu.NONE, "Open");
        menu.add(Menu.NONE, 1, Menu.NONE, "Edit");
        menu.add(Menu.NONE, 0, Menu.NONE, "Delete");
    }
}
```

INTENTS

- Used to run action by System
- Starting new Activities
- Requesting data from external Apps
- Starting system activities
- Camera, Call, Calendar, New Email,...

INTENT EXTRAS

- Intents hold primitive extras
- `Intent.putExtra(String key, Class<?> data)`
- `Intent.get<?>Extra(String key)`

- Intents accessible via `Activity getIntent()`
- Intent may not be null in activities

INTENT EXTRAS

Intent Extras are used as return value holders

eg. `onActivityResult(int req, int res, Intent data)`

IMPORTANT INTENTS

SHARE

Intent.ACTION_SEND

CAMERA

android.media.action.IMAGE_CAPTURE

DIAL

Intent.ACTION_DIAL

CHOOSE

Intent.ACTION_CHOOSER

....

CALLING INTENT

- startActivity(Intent)
- onActivityResult(int, int, Intent)

```
Intent intent = new Intent(Intent.ACTION_DIAL);  
intent.setData(Uri.parse("tel:" + mItem.content));  
startActivity(intent);
```

```
@Override  
public void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (resultCode == Activity.RESULT_OK && requestCode == 0) {  
        String result = data.toURI();  
        // ...  
    }  
}
```