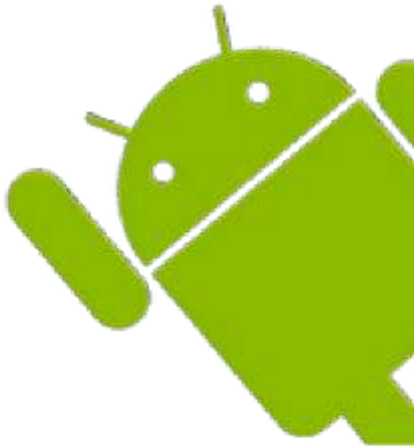
Mobile app development – week 5



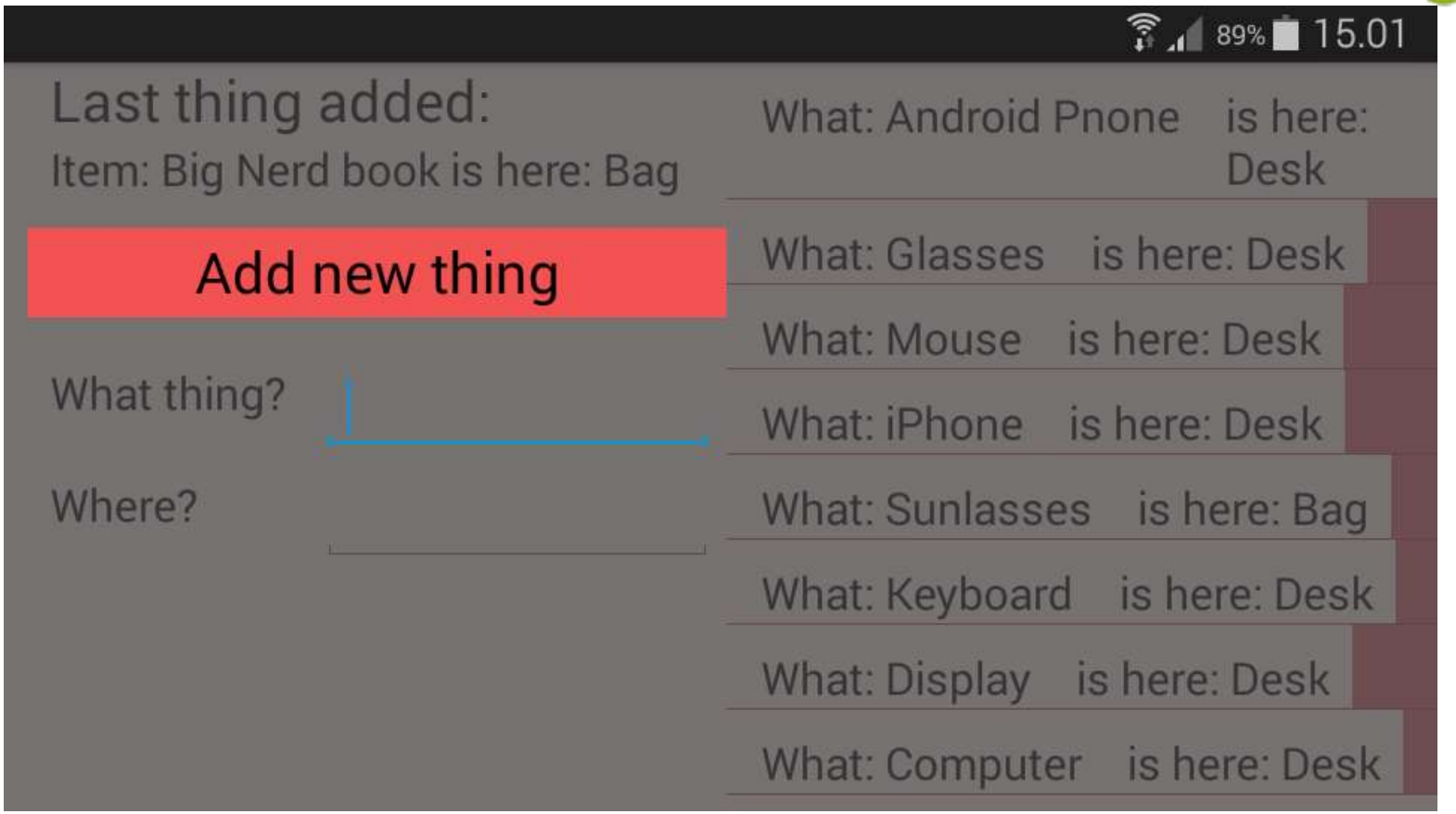
Questions related to WorkplanW5 (Mandatory assignment)

RecyclerView (chapters 9, 10 and 11)

Threading (continued)

Exercises

Updating a list



Tingle structure

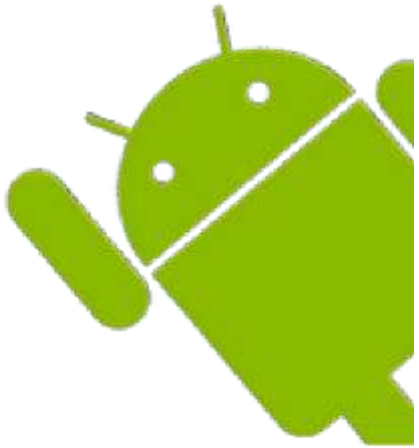
TingleActivity

UIFragment

AddThing

ListFragment

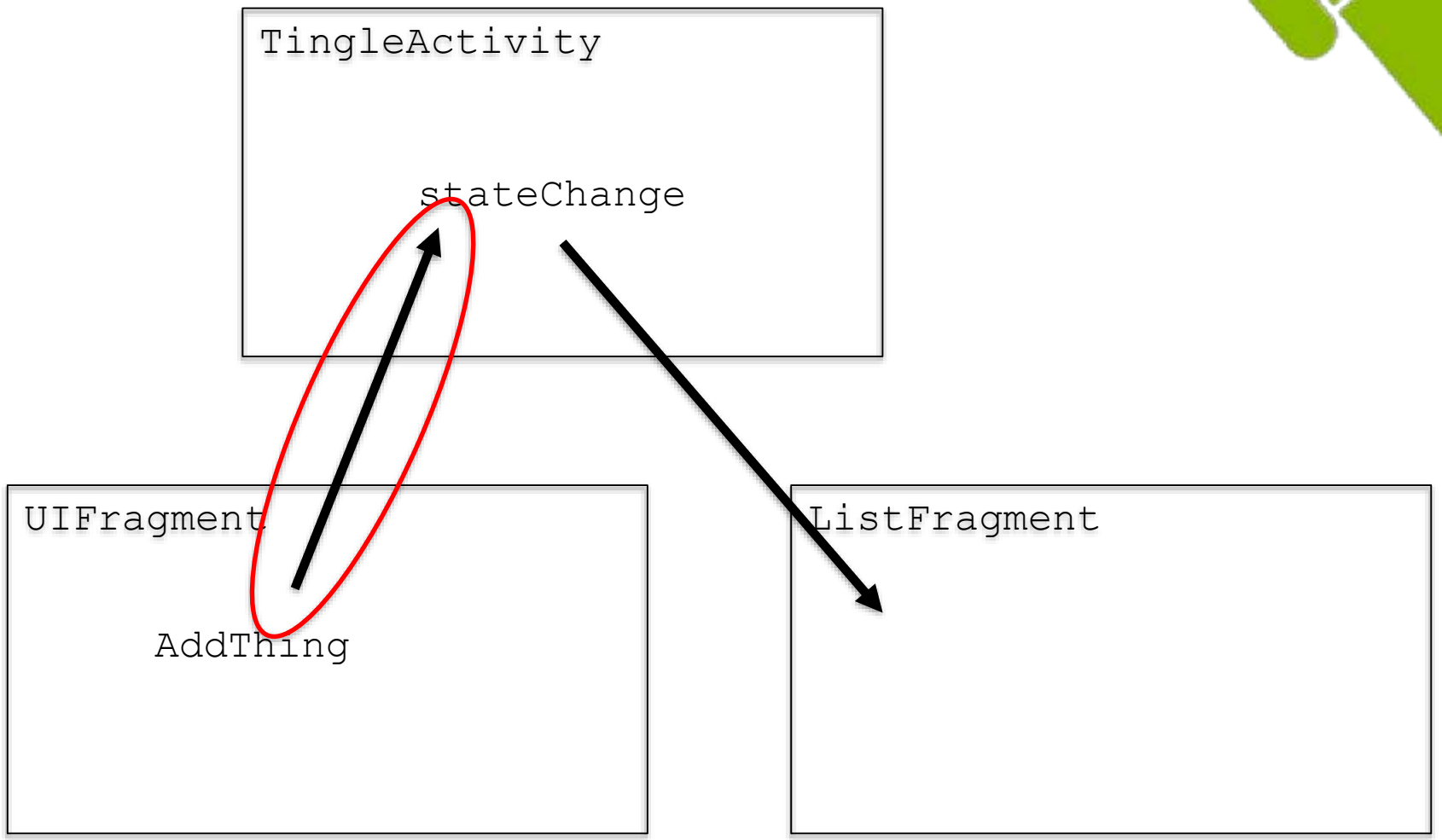
Reusable fragments



BigNerd book (p.199 just below listing 10.7)

Hosting activities should know the specifics of how to host their fragments, but fragments should not have to know specifics about their activities. At least, not if you want to maintain the flexibility of independent fragments.

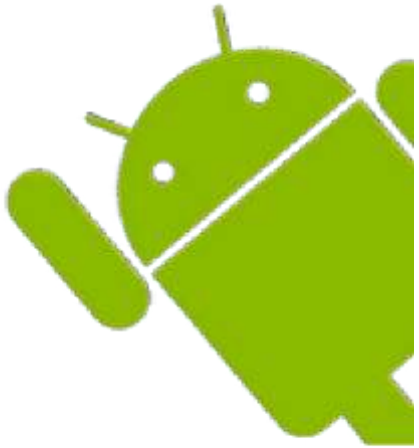
Communicating between fragments and activities



Communicating with an interface (1)

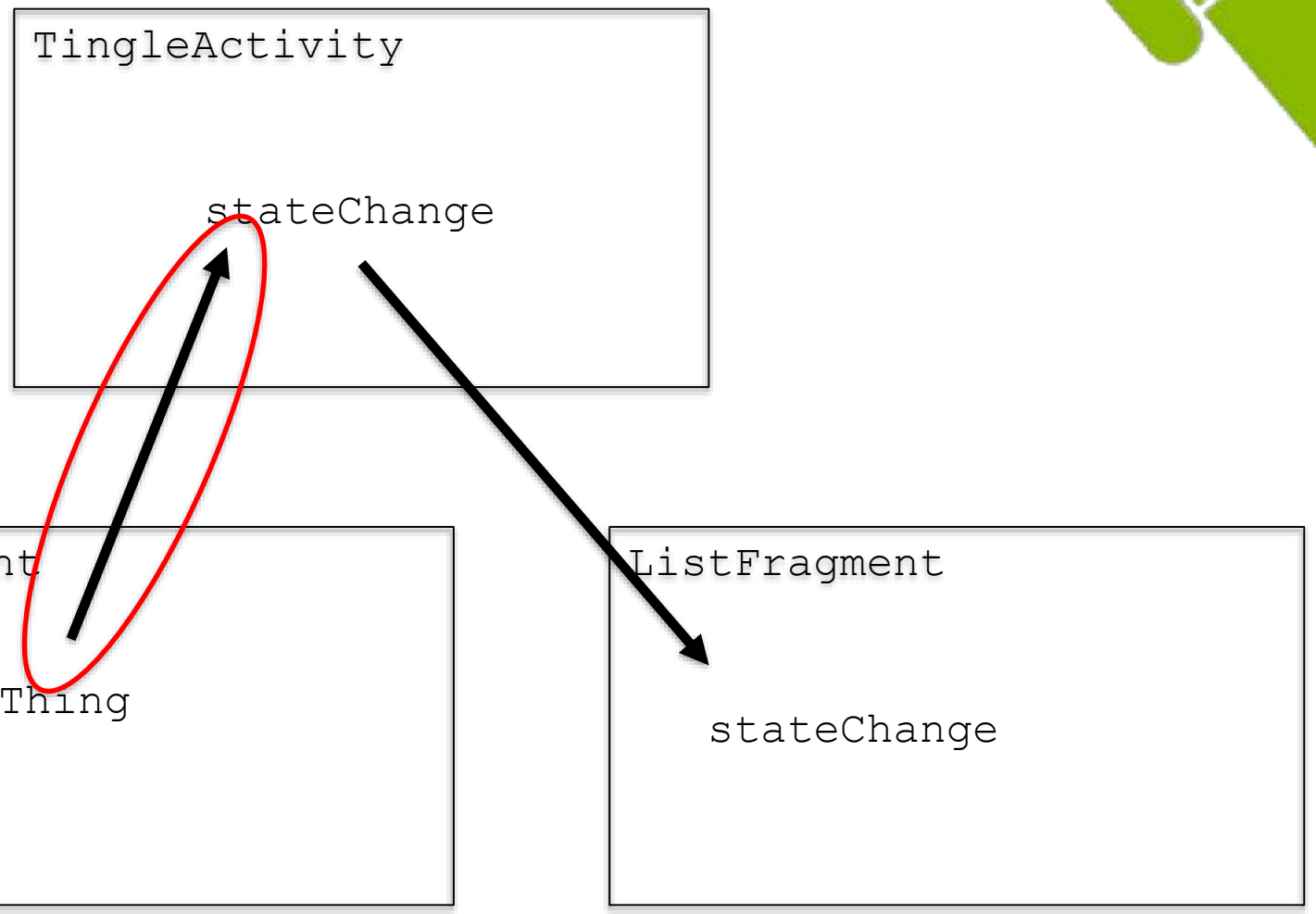
```
public class UIFragment extends Fragment {
  // GUI variables
  private Button addThing;
   …
  public interface toActivity { public void stateChange();  }

  public void onCreate(Bundle savedInstanceState) {
   …  }

  public View …onCreateView( …

      addThing.setOnClickListener(new View.OnClickListener() {
      @Override
        public void onClick(View view) {
       …
      ((toActivity) getActivity()).stateChange();
    }
   }
  });

  …
```

Communicating with an interface (2)

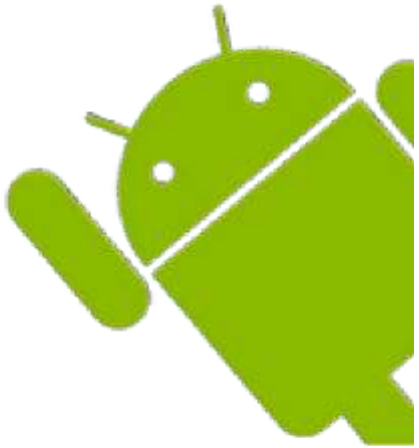


```
public class TingleActivity extends FragmentActivity implements UIFragment.toActivity
{
  private  Fragment fragmentListLand;

  @Override
  public void stateChange() {
        ((ThingListFragment) fragmentListLand).stateChange();
  }

  protected void onCreate(Bundle savedInstanceState) {

     fragmentListLand= fm.findFragmentById(R.id.fragment_container_list);



  …
```

Communicating between fragments and activities

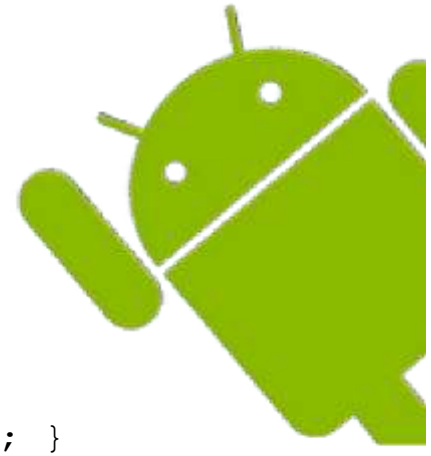


Communicating with an interface (2)

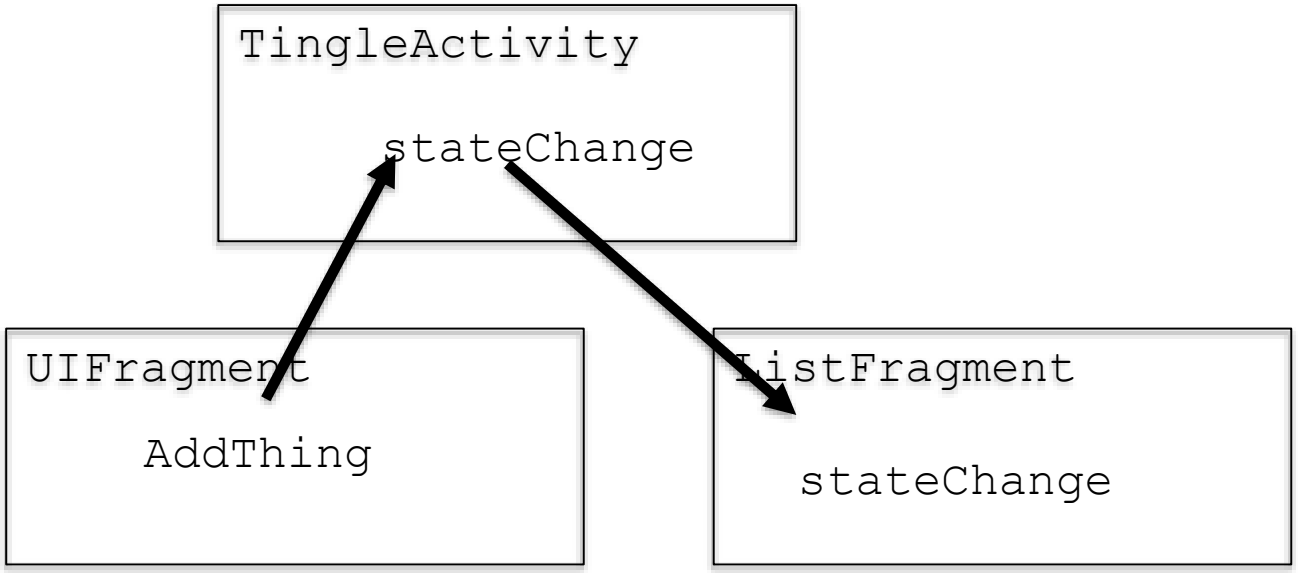


```
public class TingleActivity extends FragmentActivity implements UIFragment.toActivity
{
  private  Fragment fragmentListLand;

  @Override
  public void stateChange() {
        remove and add Fragment  }

  protected void onCreate(Bundle savedInstanceState) {

     fragmentListLand= fm.findFragmentById(R.id.fragment_container_list);



  …
```

Communicating with an interface (3)

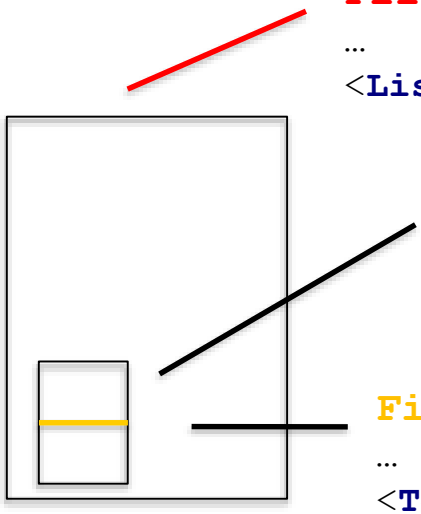


```
public class ThingListFragment extends Fragment {

  static private ThingArrayAdapter listAdapter;


  public void stateChange() { listAdapter.notifyDataSetChanged(); }

  …
```



http://developer.android.com/training/basics/fragments/communicating.html

Scrollable lists



File: tingle_activity.xml

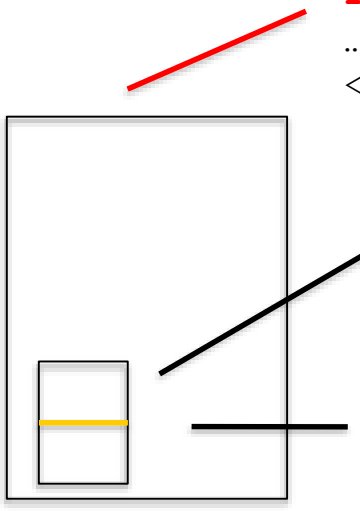
```
…
<ListView
    …
    android:id="@+id/thing_list_view"
    …    />
```

File: list_item.xml

```
…
<TextView
    …
    …    />
```

```
List<Thing>

------
-------
--
---

---
```

Scrollable lists with RecyclerView



File: tingle_activity.xml

```
…
<ListView
    …
    android:id="@+id/thing_list_view"
    …    />
```

File: list_item.xml

```
…
<RecyclerView
    …
    …    />
```

```
List<Thing>

------
-------
--
---

---
```

RecyclerView adapter

Listing 9.19 CrimeAdapter filled out (CrimeListFragment.java)

```
private class CrimeAdapter extends RecyclerView.Adapter<CrimeHolder> {
...
@Override
public CrimeHolder onCreateViewHolder(ViewGroup parent, int viewType) {




}

@Override
public void onBindViewHolder(CrimeHolder holder, int position) {


}

@Override
public int getItemCount() {                    }
}
```

RecyclerView adapter

Listing 9.19 CrimeAdapter filled out (CrimeListFragment.java)

```
private class CrimeAdapter extends RecyclerView.Adapter<CrimeHolder> {
...
@Override
public CrimeHolder onCreateViewHolder(ViewGroup parent, int viewType) {
   LayoutInflater layoutInflater = LayoutInflater.from(getActivity());
   View view = layoutInflater.inflate(android.R.layout.simple_list_item_1, parent, false);
   return new CrimeHolder(view);
}

@Override
public void onBindViewHolder(CrimeHolder holder, int position) {



}

@Override
public int getItemCount() {                    }
}
```

Recyclerview adapter

Listing 9.19 CrimeAdapter filled out (CrimeListFragment.java)

```
private class CrimeAdapter extends RecyclerView.Adapter<CrimeHolder> {
...
@Override
public CrimeHolder onCreateViewHolder(ViewGroup parent, int viewType) {
   LayoutInflater layoutInflater = LayoutInflater.from(getActivity());
   View view = layoutInflater.inflate(android.R.layout.simple_list_item_1, parent, false);
   return new CrimeHolder(view);
}

@Override
public void onBindViewHolder(CrimeHolder holder, int position) {
  Crime crime = mCrimes.get(position);
  holder.mTitleTextView.setText(crime.getTitle());
}

@Override
public int getItemCount() { return mCrimes.size();}
}
```

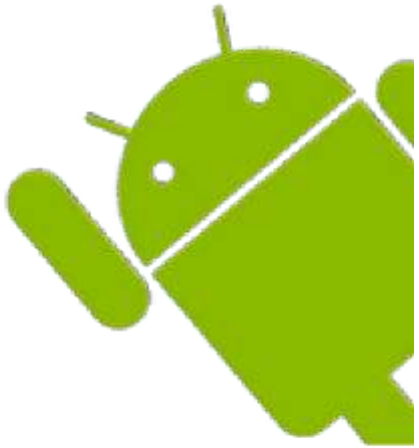
Future of Tingle

Option A:
Stay with ListView
but go through chapter 9, 10 and 11

Option C(hallenge):
Rewrite ListFragment using RecyclerView

Multistream programming

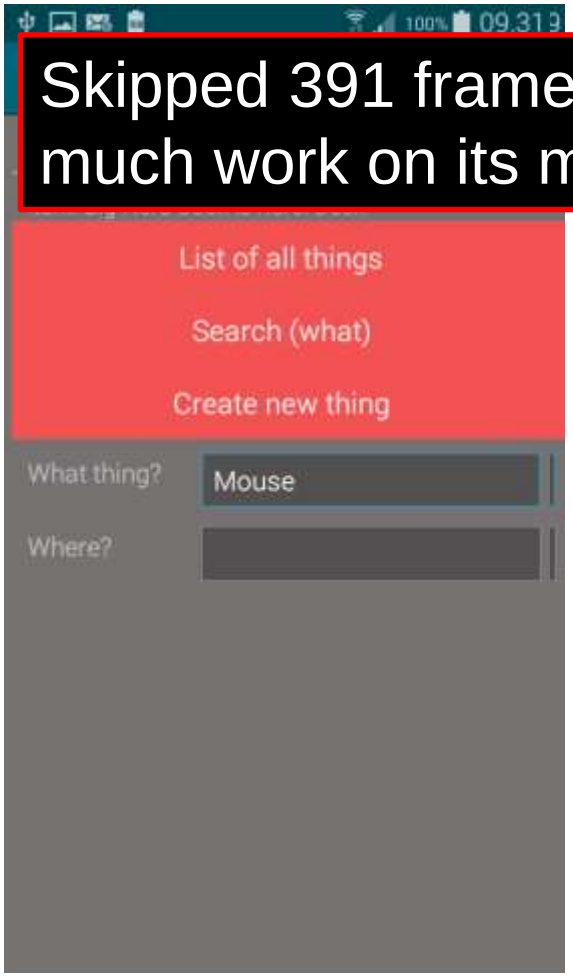
Transistormaskinen GIER, 1961



Single stream

Multistream

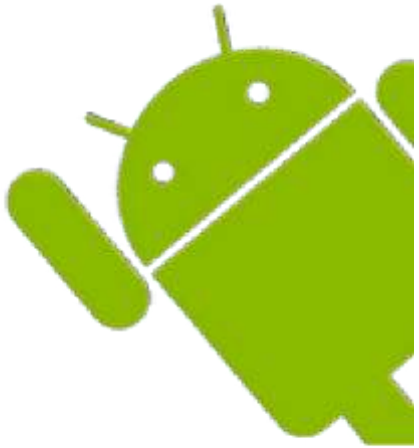
Adding a search button



Skipped 391 frames! The application may be doing too much work on its main thread.

```
private String searchThing(String what) {
    int i = 0;        Boolean found = false;
    int s = thingsDB.size();

    while ((!found) && (i < s)) {
        found= thingsDB.get(i).getWhat().equals(what);
        i = i + 1;
     }
     return
       (found) ?
           thingsDB.get(i - 1).getWhere() : //found
           "????";                                  // not found
}
```

Searching in a separate thread



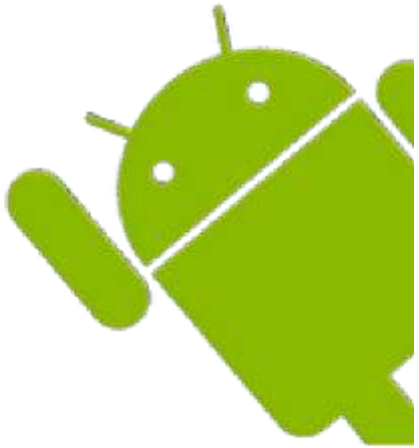
```
private void searchThing() {
        (new Thread(new ClientThread())).start();
}
class ClientThread implements Runnable {
        @Override  public void run() {

          String what= newWhat.getText().toString();
          int i= 0;          Boolean found= false;
```

FATAL EXCEPTION: Thread-67732
Process: dk.staunstrups.things, PID: 18505
android.view.ViewRootImpl$CalledFromWrongThreadException:
Only the original thread that created a view hierarchy can touch its views.

```
          }
          newWhere.setText(
              (found) ? thingsDB.get(i-1).getWhere()
                   : "????");
          return;
}     }
```

https://docs.oracle.com/javase/tutorial/essential/concurrency/index.html

Android: AsyncTask



```
private class someName extends AsyncTask<Params, Progress, Result> {

    protected abstract Result doInBackground(Params... param) {    ….   }

    protected void onProgressUpdate(Progress  … prog  {   …  }

    protected void onPostExecute(Result  ….  result) {  …  }

}
```

Using AsyncTask to search for Things

```
private class searchClass extends AsyncTask<String, Void, String> {
      String response = "????";
      Boolean found= false;       String mWhat;
      protected String doInBackground(String... param) {
        int i = 0;
        Boolean found = false;
        mWhat= param[0];
        int s = thingsDB.size();
        while ((!found) && (i < s)) {
           found = thingsDB.get(i).getWhat().equals(mWhat);
           i = i + 1;
        }
        return (found) ? thingsDB.get(i - 1).getWhere() : "????";
      }
     @Override
      protected void onPostExecute(String result) {  newWhere.setText(result);  }
    }
```