

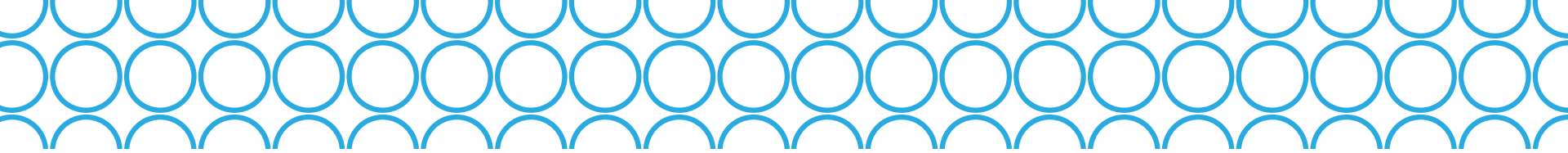


**NEW MEDIA &
COMMUNICATION
TECHNOLOGY**

Mobile App Development

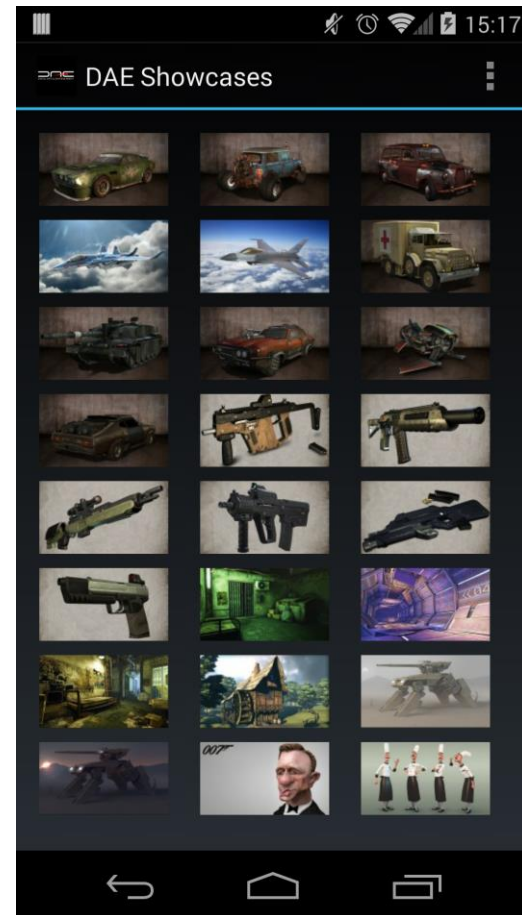
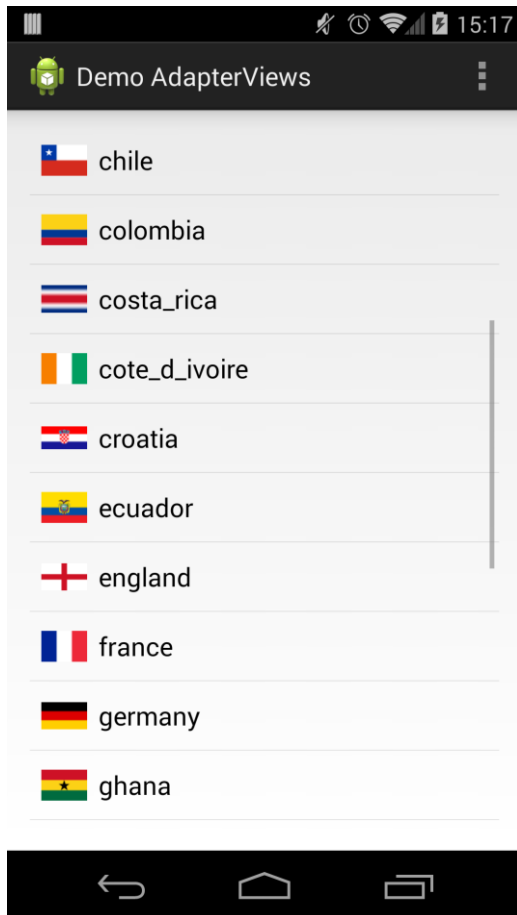
Adapters & AdapterViews

Content driven UI Elements

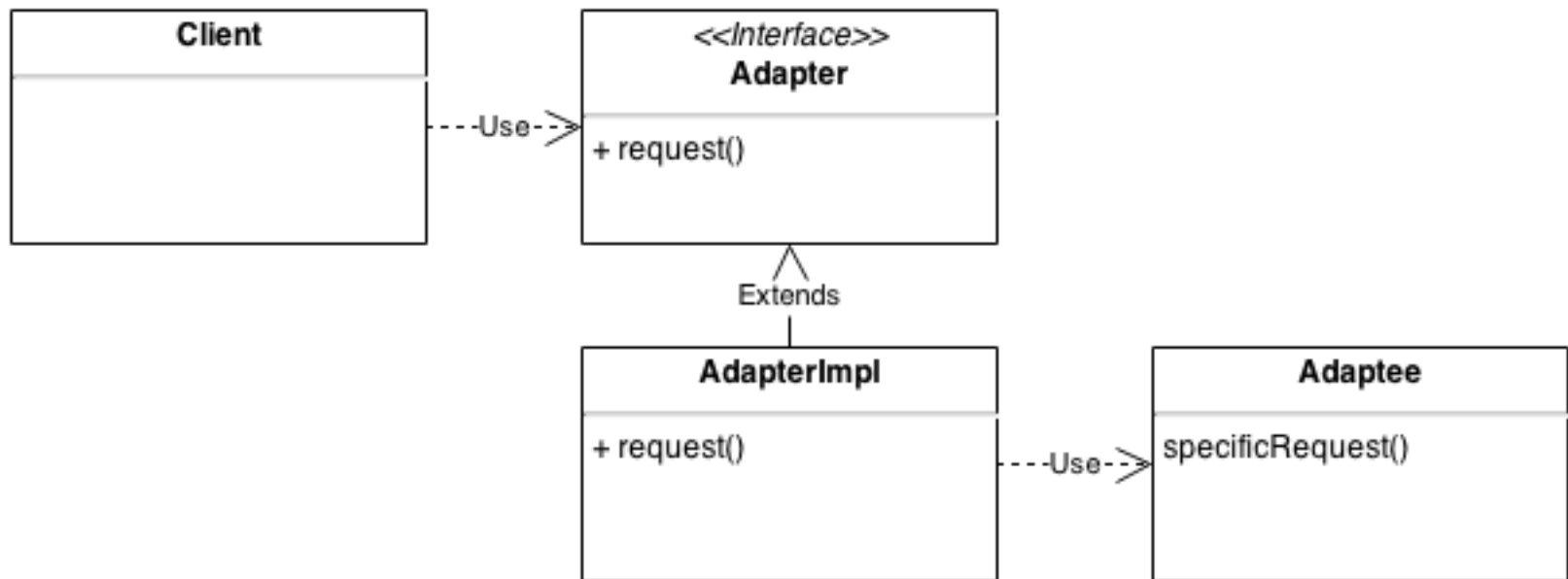


*“a device that is used to connect
two pieces of equipment
that were not designed to be connected,,*

AdapterViews



Adapter Design Pattern

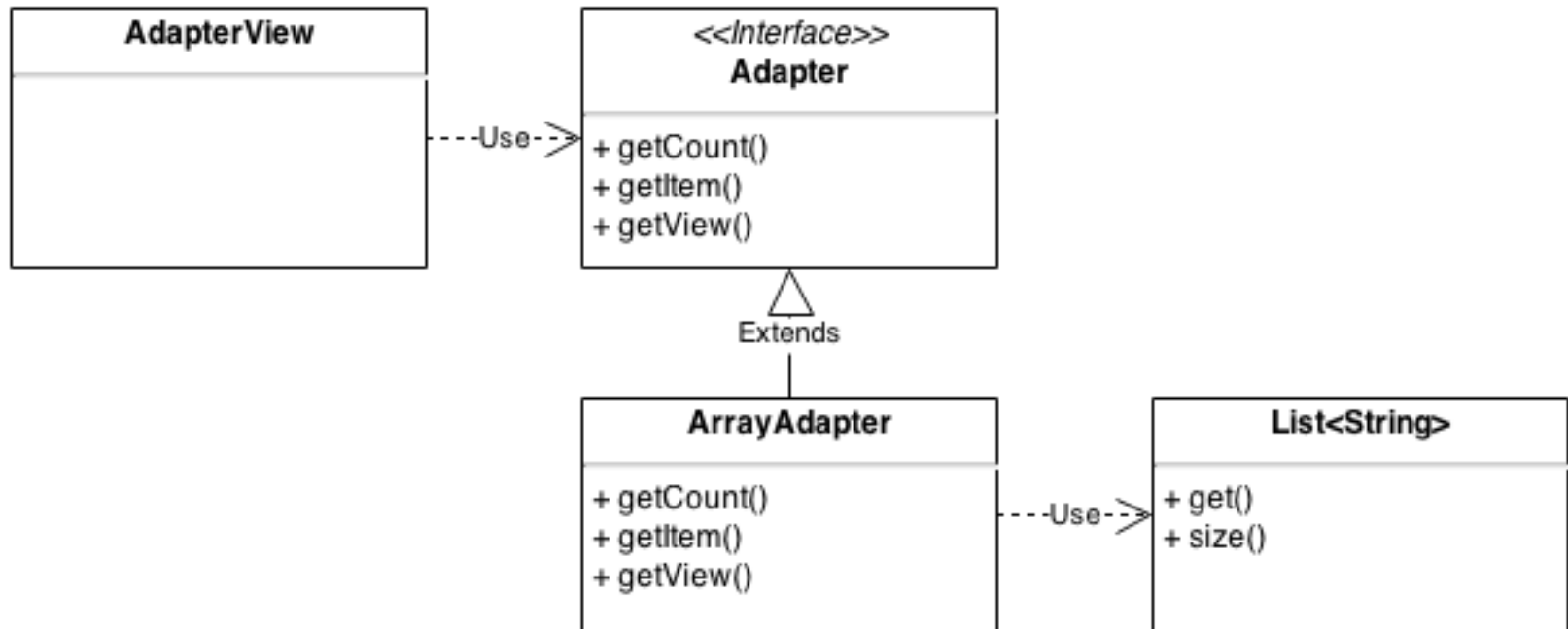




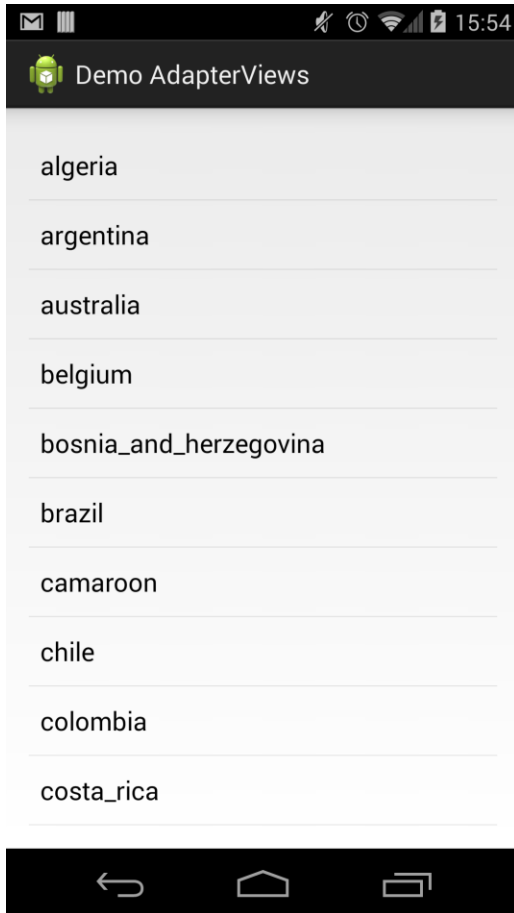
EUR. SCHUKO TO UK 13A PLUG ADAPTER

NEW MEDIA AND COMMUNICATION TECHNOLOGY

Adapter Design Pattern



Listview & ListAdapter



```
public class MainActivity extends Activity {  
    private ListView adapterView;  
    private ListAdapter adapter;  
    private String[] data;  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
        setContentView(R.layout.cv);  
  
        this.data = Data.getStringArray();  
  
        this.adapter = new ArrayAdapter<String>(  
            this,  
            android.R.layout.simple_list_item_1,  
            data);  
  
        this.adapterView =  
            (ListView) findViewById(R.id.av);  
  
        this.adapterView.setAdapter(adapter);  
    }  
}
```


GridView & ListAdapter



```
public class MainActivity extends Activity {  
    private GridView adapterView;  
    private ListAdapter adapter;  
    private String[] data;  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
        setContentView(R.layout.cv);  
  
        this.data = Data.getStringArray();  
  
        this.adapter = new ArrayAdapter<String>(  
            this,  
            android.R.layout.simple_list_item_1,  
            data);  
  
        this.adapterView =  
            (GridView) findViewById(R.id.av);  
  
        this.adapterView.setAdapter(adapter);  
    }  
}
```

Custom Layouts



```
this.adapter = new ArrayAdapter<String>(
    this, R.layout.li_1, R.id.text, data);
```

```
<RelativeLayout android:layout_width="match_parent"
    android:layout_height="48dp" >

    <ImageView
        android:id="@+id/img"
        android:src="@drawable/belgium" />

    <TextView
        android:id="@+id/text"
        android:text="Country Name" />

</RelativeLayout>
```

ListActivity

```
public class MainActivity extends ListActivity {  
  
    private ListView adapterView;  
    private ListAdapter adapter;  
    private String[] data;  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
  
        this.data = Data.getStringArray();  
        this.adapter = new ArrayAdapter<String>( this, R.layout.li_1, R.id.text, data);  
  
        setListAdapter(this.adapter);  
    }  
}
```

ListActivity & Custom View



```
<RelativeLayout>
    <ListView
        android:id="@android:id/list"
        android:layout_below="@+id/map" >
    </ListView>
    <ImageView
        android:id="@+id/map"
        android:src="@drawable/map" />
</RelativeLayout>
```

```
@Override
protected void onCreate(Bundle b) {
    super.onCreate(b);
    setContentView(R.layout.activity_main_list);
    this.data = Data.getStringArray();
    this.adapter = new ArrayAdapter<String>(
        this, R.layout.li_1, R.id.text, data);
    setListAdapter(this.adapter);
}
```

Loader Framework

```
public class MainActivity extends ListActivity implements LoaderCallbacks<List<String>>{
    private static final int LOADER_ID = 1;
    private ArrayAdapter<String> adapter;

    @Override
    protected void onCreate(Bundle b) {
        super.onCreate(b);
        this.adapter = new ArrayAdapter<String>( this, android.R.layout.simple_list_item_1);
        setListAdapter(this.adapter);

        getLoaderManager().initLoader(LOADER_ID, null, this);
    }

    public Loader<List<String>> onCreateLoader(int id, Bundle args) {
        return new StringListLoader(this);
    }

    public void onLoadFinished(Loader<List<String>> loader, List<String> stringList) {
        this.adapter.clear();
        this.adapter.addAll(stringList);
    }

    public void onLoaderReset(Loader<List<String>> arg0) {
        this.adapter.clear();
    }
}
```

Loader Framework

```
public class StringListLoader extends AsyncTaskLoader<List<String>> {
    private List<String> mStringList;

    public StringListLoader(Context context) {
        super(context);
    }

    protected void onStartLoading() {
        if(mStringList == null)
            forceLoad();
    }

    public List<String> loadInBackground() {
        mStringList = new ArrayList<String>();
        for(int i = 0; i < 100; i++)
            mStringList.add("String " + i);

        return mStringList;
    }
}
```

Why Loader Framework?

```
public class MainActivity extends ListActivity {  
  
    private static final String[] PROJECTION = new String[] { "_id", "text_column" };  
    private static final String CONTENT_URI = "content://authority/resources";  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
  
        Cursor cursor = managedQuery( CONTENT_URI, PROJECTION, null, null, null);  
  
        String[] columns = { "text_column" };  
        int[] views = { android.R.id.text1 };  
  
        SimpleCursorAdapter adapter = new SimpleCursorAdapter(this,  
            android.R.layout.simple_list_item_1,  
            cursor,  
            columns,  
            views);  
  
        setListAdapter(adapter);  
    }  
}
```

Custom Loaders: Doing work

```
public class CustomLoader extends AsyncTaskLoader<Object> {
    private Object mData;
    private CustomObserver mObserver;

    public CustomLoader(Context ctx) { super(ctx); }

    public Object loadInBackground() { return new Object(); }

    public void deliverResult(Object data) {
        if (isReset()) { releaseResources(data); return; }

        Object oldData = mData;
        mData = data;

        if (isStarted()) { super.deliverResult(data); }

        if (oldData != null && oldData != data) { releaseResources(oldData); }
    }

    /* ... */
}
```


Custom Loaders: Handling Work

```
public class CustomLoader extends AsyncTaskLoader<Object> {
    /* ... */

    protected void onStartLoading() {
        if (mData != null) { deliverResult(mData); }

        if (mObserver == null) { mObserver = new CustomObserver(); }

        if (takeContentChanged() || mData == null) { forceLoad(); }
    }

    protected void onStopLoading() { cancelLoad(); }

    protected void onReset() {
        onStopLoading();

        if (mData != null) { releaseResources(mData); mData = null; }

        if (mObserver != null) { mObserver = null; }
    }

    public void onCancel(Object data) {
        super.onCanceled(data);
        releaseResources(data);
    }

    private void releaseResources(Object data) { /* ... */ }
}
```

Custom Loaders: Looking at Work

```
public class CustomObserver {  
    private Loader loader;  
    public CustomObserver(Loader loader) {  
        this.loader = loader;  
    }  
    public void onChange(boolean selfChange) {  
        loader.onContentChanged();  
    }  
}
```

Cursors

```
public interface Cursor extends Closeable {
    /* How many rows*/
    int getCount();

    /* How many and what columns */
    int getColumnCount();
    int getColumnIndex(String columnName);

    /* Where am I? */
    int getPosition();
    boolean isFirst();

    /* Take Me Somewhere */
    boolean move(int offset);

    /* Give me the content of a column */
    String getString(int columnIndex);
    short getShort(int columnIndex);

    /* Handling the Cursor Itself */
    void close();

    /* Callback */
    void registerContentObserver(ContentObserver observer);
    void registerDataSetObserver(DataSetObserver observer);
    void setNotificationUri(ContentResolver cr, Uri uri);
}
```

CursorLoader

```
public CursorLoader(Context context, Uri uri, String[] projection, String selection,  
String[] selectionArgs, String sortOrder)
```

| | |
|----------------------|--|
| <i>uri</i> | A content url. E.g. content://be.howest.admin/students |
| <i>projection</i> | List of columns to select from the content provider E.g. new String[] { BaseColumns._ID, "name" } |
| <i>selection</i> | Equivalent of 'SQL WHERE' E.g. (name = ? AND active=true) |
| <i>selectionArgs</i> | Parameters for selection E.g. New String[] { "steven" } |
| <i>sortOrder</i> | Equivalent for 'SQL ORDEBY' E.g. "name ASC" |

Cursors, SQL, Where, WhereArgs

- *"Note: By implementing the BaseColumns interface, your inner class can inherit a primary key field called **_ID** that some Android classes such as cursor adaptors will expect it to have. It's not required, but this can help your database work harmoniously with the Android framework."*
- Use DatabaseUtils, TextUtils, ...

```
String where = "name = ?";  
String[] whereArgs = new String[] {"steven"};  
  
where = DatabaseUtils.concatenateWhere(where, "active = ?");  
whereArgs = DatabaseUtils.appendSelectionArgs(whereArgs, new String[] {"true"});
```

CursorAdapter

```
public class MainActivity extends ListActivity implements LoaderCallbacks<Cursor> {
    static final String[] CONTACTS_SUMMARY_PROJECTION = new String[] {
        Contacts._ID,
        Contacts.DISPLAY_NAME,
        Contacts.CONTACT_STATUS,
        Contacts.CONTACT_PRESENCE,
        Contacts.PHOTO_ID,
        Contacts.LOOKUP_KEY
    };
    /* ... */
    public Loader<Cursor> onCreateLoader(int id, Bundle args) {
        String select = "(" + Contacts.DISPLAY_NAME + " NOTNULL) AND ("
            + Contacts.HAS_PHONE_NUMBER + "=1) AND ("
            + Contacts.DISPLAY_NAME + " != ' ' )";
        return new CursorLoader(this, Contacts.CONTENT_URI,
            CONTACTS_SUMMARY_PROJECTION, select, null,
            Contacts.DISPLAY_NAME + " COLLATE LOCALIZED ASC");
    }
    public void onLoadFinished(Loader<Cursor> loader, Cursor cursor) {
        this.adapter.swapCursor(cursor);
    }
    public void onLoaderReset(Loader<Cursor> arg0) {
        this.adapter.swapCursor(null);
    }
}
```

CursorAdapter

```
public class MainActivity extends ListActivity implements LoaderCallbacks<Cursor> {

    private static final int LOADER_ID = 1;
    private CursorAdapter adapter;

    protected void onCreate(Bundle b) {
        super.onCreate(b);
        this.adapter = new SimpleCursorAdapter(this,
            android.R.layout.simple_list_item_2, null,
            new String[] { Contacts.DISPLAY_NAME, Contacts.CONTACT_STATUS },
            new int[] { android.R.id.text1, android.R.id.text2 }, 0);

        setListAdapter(this.adapter);

        getLoaderManager().initLoader(LOADER_ID, null, this);
    }
    /* ... */
}
```

CursorAdapter

```
<TwoLineListItem xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:minHeight="?android:attr/listPreferredItemHeight"
    android:mode="twoLine"
    android:paddingStart="?android:attr/listPreferredItemPaddingStart"
    android:paddingEnd="?android:attr/listPreferredItemPaddingEnd">

    <TextView android:id="@android:id/text1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dip"
        android:textAppearance="?android:attr/textAppearanceListItem" />

    <TextView android:id="@android:id/text2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@android:id/text1"
        android:layout_alignStart="@android:id/text1"
        android:textAppearance="?android:attr/textAppearanceSmall"/>

</TwoLineListItem>
```


MatrixCursor

```
String[] columns = {
    BaseColumns._ID,
    "name",
    "active"
};

MatrixCursor cursor = new MatrixCursor(columns);

cursor.newRow()
    .add(1)
    .add("steven")
    .add(true);

cursor.addRow(new Object[] { 1, "mark", false });
```

- *Don't Fight the framework, Use the framework!
(there is no spoon)*

SimpleAdapter

```
public class MainActivity extends ListActivity {  
  
    static final List<Map<String, ?>> data = Data.getListOfMaps();  
    private SimpleAdapter adapter;  
  
    @Override  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
        this.adapter = new SimpleAdapter(this, data,  
            android.R.layout.simple_list_item_2,  
            new String[] { "lowercase", "uppercase" },  
            new int[] { android.R.id.text1, android.R.id.text2 });  
  
        setListAdapter(this.adapter);  
    }  
}
```

- *Supports Checkable Views, TextView and ImageView*
- *Allows use of ViewBinder*

SimpleAdapter.ViewBinder

```
public class MainActivity extends ListActivity {  
  
    protected void onCreate(Bundle b) {  
        super.onCreate(b);  
        this.adapter = new SimpleAdapter( /* ... */ );  
        this.adapter.setViewBinder(new ColorViewBinder());  
        setListAdapter(this.adapter);  
    }  
  
    class ColorViewBinder implements SimpleAdapter.ViewBinder {  
        public boolean setViewValue(View view, Object data, String toString) {  
            if(view.getId() == R.id.colorPanel) {  
                view.setBackgroundColor(Color.parseColor((String)data));  
                return true;  
            }  
            return false;  
        }  
    }  
}
```

ResourceCursorAdapter

```
public class CustomAdapter extends android.widget.ResourceCursorAdapter {
    public CustomAdapter(Context context) {
        super(context, R.layout.list_item_custom, null, 0);
    }

    @Override
    public void bindView(View view, Context ctx, Cursor c) {
        ImageView image = (ImageView)view.findViewById(R.id.image);
        image.setImageResource(c.getInt(0));

        TextView text = (TextView)view.findViewById(R.id.text);
        text.setText(c.getString(1));

        TextView description = (TextView)view.findViewById(R.id.description);
        description.setText(c.getString(2));
    }
}
```

Optimizing Adapters

```
public class CustomAdapter extends ResourceCursorAdapter {
    class ViewHolder {
        public ImageView image;
        public TextView text;
        public TextView description;

        public ViewHolder(View v) {
            image = (ImageView)v.findViewById(R.id.image);
            text = (TextView)v.findViewById(R.id.text);
            description = (TextView)v.findViewById(R.id.description);
        }
    }

    public View getView(Context context, Cursor cursor, ViewGroup parent) {
        View v = super.getView(context, cursor, parent);
        v.setTag(new ViewHolder(v));
        return v;
    }

    public void bindView(View view, Context ctx, Cursor c) {
        ViewHolder vh = (ViewHolder) view.getTag();
        vh.image.setImageResource(c.getInt(0));
        vh.text.setText(c.getString(1));
        vh.description.setText(c.getString(2));
    }
    /* ... */
}
```

Adapters from scratch

```
public class CustomAdapter extends BaseAdapter {
    class ViewHolder { /* ... */ }
    private List<String[]> data;

    public CustomAdapter(Context context, List<String[]> data) { this.data = data; }

    public View getView(Context context, ViewGroup parent) {
        View v = LayoutInflater.from(context).inflate(R.layout.list_item_custom, parent);
        v.setTag(new ViewHolder(v));
        return v;
    }

    public void bindView(View view, String[] data) { /* ... */ }

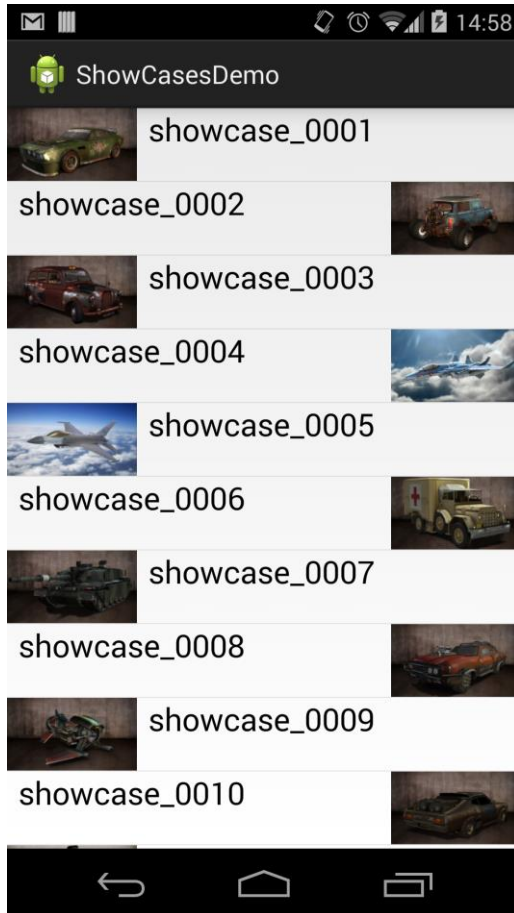
    @Override
    public View getView(int pos, View view, ViewGroup parent) {
        if (view == null) view = getView(getBaseContext(), parent);
        bindView(view, getItem(pos));
        return view;
    }

    @Override public int getCount() { return data.size(); }

    @Override public String[] getItem(int pos) { return data.get(pos); }

    @Override public long getItemId(int pos) { return Long.parseLong( getItem(pos)[3] ); }
}
```

Zebra Listviews



```
<LinearLayout>
    <ImageView
        android:id="@+id/image"
        android:src="@drawable/showcase_no_image" />
    <TextView
        android:id="@+id/text"/>
</LinearLayout>
```

```
<LinearLayout>
    <TextView
        android:id="@+id/text"/>
    <ImageView
        android:id="@+id/image"
        android:src="@drawable/showcase_no_image" />
</LinearLayout>
```

Zebra Listviews

```
public static class ShowcaseAdapter extends BaseAdapter {  
    public View getView(int i, View view, ViewGroup viewGroup) {  
        if(view == null)  
            switch (getItemViewType(i)) {  
                case 0:  
                    view = LayoutInflater.from(mContext)  
                        .inflate(R.layout.list_item_showcase, viewGroup, false);  
                    break;  
                case 1:  
                    view = LayoutInflater.from(mContext)  
                        .inflate(R.layout.list_item_showcase_alternative, viewGroup, false);  
                    break;  
            }  
        /* Use View Holder */  
        return view;  
    }  
  
    public int getItemViewType(int position) { return position % 2; }  
  
    public int getViewTypeCount() { return 2; }  
  
}
```


Item Clicks in AdapterViews

```
public class MainActivity extends Activity {  
  
    private ListView mShowcaseGrid;  
    private OnItemClickListener mOnItemClickListener = new OnItemClickListener() {  
        @Override  
        public void onItemClick(AdapterView<?> list, View item, int pos, long id) {  
            /* Do Something */  
        }  
    };  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main_list);  
        mShowcaseGrid = (ListView) findViewById(R.id.listView);  
        mShowcaseGrid.setAdapter(new ShowcaseAdapter(this));  
        mShowcaseGrid.setOnItemClickListener(mOnItemClickListener );  
    }  
}
```

Item Clicks in ListActivity

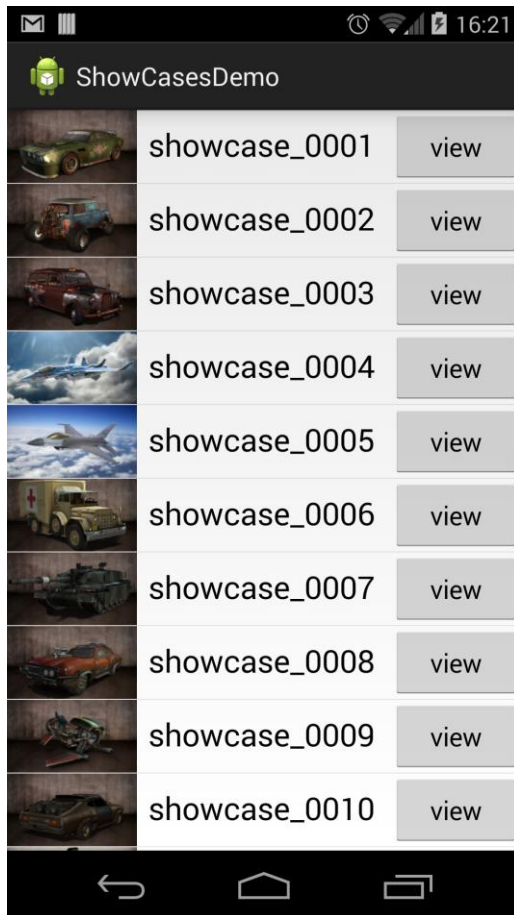
```
public class MainActivity extends ListActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setListAdapter(new ShowcaseAdapter(this));
    }

    @Override
    protected void onListItemClick(ListView l, View v, int position, long id) {
        /* Do Something */
    }

}
```

Lists with Clickable Elements



```
<LinearLayout>
    <ImageView
        android:id="@+id/image"
        android:src="@drawable/showcase_no_image" />
    <TextView
        android:id="@+id/text" />
    <Button
        android:id="@+id/button"
        android:text="view"
        android:focusable="false"/>
</LinearLayout>
```

Lists with Clickable Elements

```
class ViewHolder {
    public ImageView image;
    public TextView text;
    public long id;

    public OnClickListener mClickListener = new OnClickListener() {
        public void onClick(View v) { /* do something with id */ }
    };

    public ViewHolder(View view) {
        image = ((ImageView)view.findViewById(R.id.image));
        text = ((TextView)view.findViewById(R.id.text));
        ((Button) view.findViewById(R.id.button)).setOnClickListener(mClickListener);
    }
}
```

```
public View getView(int i, View view, ViewGroup viewGroup) {
    if(view == null) {
        view = LayoutInflater.from(mContext)
            .inflate(R.layout.list_item_showcase, viewGroup, false);
        view.setTag(new ViewHolder(view));
    }
    ((ViewHolder)view.getTag()).id = getItemId(i);
    /* Do other things to ViewHolder */
    return view;
}
```

Lists with Single ChoiceMode

```
<ListView  
    android:id="@+id/listView"  
    android:choiceMode="singleChoice" />
```

```
public void showSelection(View v) {  
    Toast.makeText(this,  
        "you selected: " + mShowcaseGrid.getCheckedItemPosition(),  
        Toast.LENGTH_SHORT).show();  
}
```

Lists with Multi ChoiceMode

```
<ListView  
    android:id="@+id/listView"  
    android:choiceMode="multipleChoice" />
```

```
public void showSelection(View v) {  
    Toast.makeText(this,  
        "you selected " + mShowcaseGrid.getCheckedItemCount() + " items",  
        Toast.LENGTH_SHORT).show();  
}
```

