

Creating A List With Multiple Select Delete Using Eclipse

ECE 480 FALL 2014

Ben Lauzon



Abstract

Creating lists with deletion capabilities is an important feature to many android applications. This application note provides the reader with an understanding of how the Eclipse IDE works by demonstrating how to set setup your system to begin programming with Eclipse and how to create a list with multiple select delete capabilities.

Keywords

Eclipse, Listview, activity, XML, action bar, SDK ListViewAdapter, TextViews, class, package, application, AndroidManifest, and strings.

How to Setup Programming Environment

The first step will be to download/install Eclipse:

<http://www.eclipse.org/home/index.php>

Next, download/install the Android SDK:

<http://developer.android.com/sdk/index.html>

Next, download/install the ADT plugin for Eclipse:

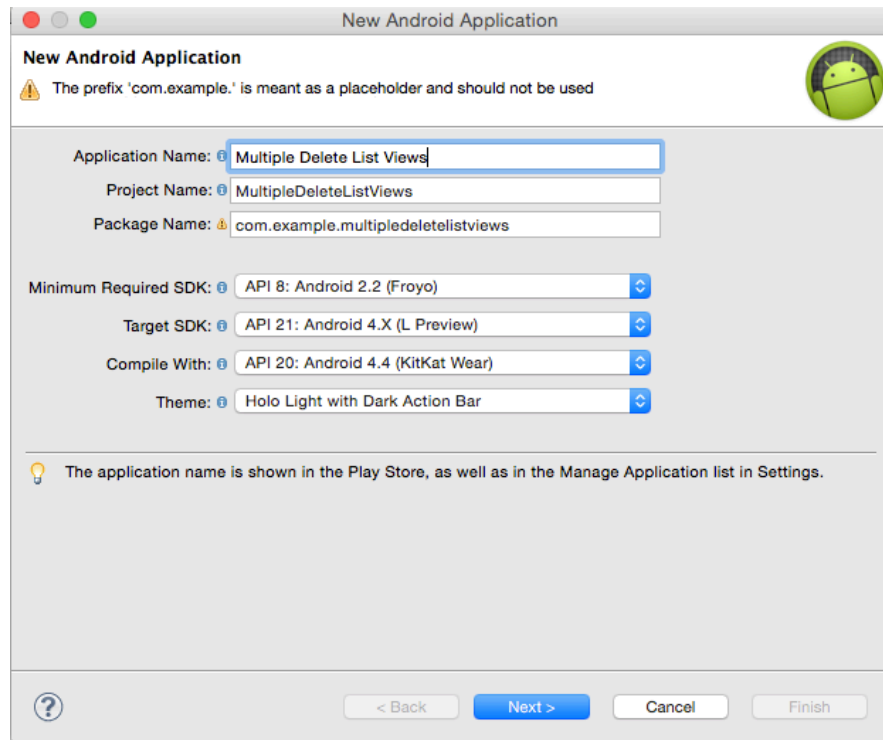
<http://developer.android.com/tools/sdk/eclipse-adt.html>

Once these tasks have been completed, it is now time to open Eclipse and begin building your first app.

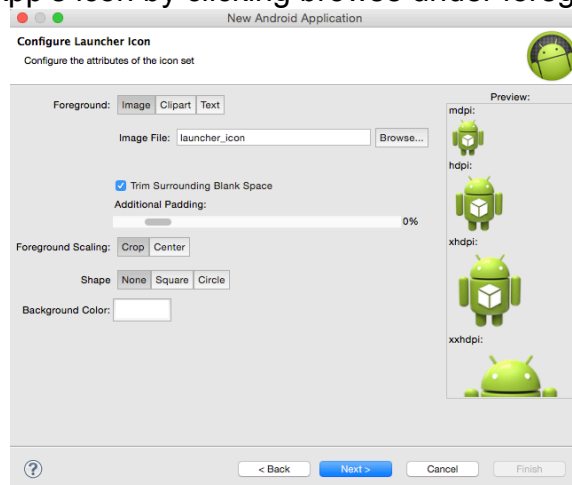
Creating An App

In this tutorial we will create an app called MultipleDeleteListView that will create a shopping list with items and a prices.

1. Select File > New > Android Application
The following window will appear:



2. Type "Multiple Delete List View" In the Application Name text box
3. Click Next
4. Click Next
5. Select your App's icon by clicking browse under foreground.



6. Click Next


```

47
48 // Binds the Adapter to the ListView
49 list.setAdapter(listviewadapter);
50 list.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE_MODAL);
51 // Capture ListView item click
52 list.setMultiChoiceModelListener(new MultiChoiceModelListener() {
53
54     @Override
55     public void onItemCheckedStateChanged(ActionMode mode,
56         int position, long id, boolean checked) {
57         // Capture total checked items
58         final int checkedCount = list.getCheckedItemCount();
59         // Set the CAB title according to total checked items
60         mode.setTitle(checkedCount + " Selected");
61         // Calls toggleSelection method from ListViewAdapter Class
62         listviewadapter.toggleSelection(position);
63     }
64
65     @Override
66     public boolean onActionItemClicked(ActionMode mode, MenuItem item) {
67         switch (item.getItemId()) {
68             case R.id.delete:
69                 // Calls getSelectedIds method from ListViewAdapter Class
70                 SparseBooleanArray selected = listviewadapter
71                     .getSelectedIds();
72                 // Captures all selected ids with a loop
73                 for (int i = (selected.size() - 1); i >= 0; i--) {
74                     if (selected.valueAt(i)) {
75                         Shopping selecteditem = listviewadapter
76                             .getItem(selected.keyAt(i));
77                         // Remove selected items following the ids
78                         listviewadapter.remove(selecteditem);
79                     }
80                 }
81                 // Close CAB
82                 mode.finish();
83                 return true;
84             default:
85                 return false;
86         }
87     }
88     @Override
89     public boolean onCreateActionMode(ActionMode mode, Menu menu) {
90         mode.getMenuInflater().inflate(R.menu.activity_main, menu);
91         return true;
92     }
93     @Override
94     public void onDestroyActionMode(ActionMode mode) {
95         // TODO Auto-generated method stub
96         listviewadapter.removeSelection();
97     }
98     @Override
99     public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
100         // TODO Auto-generated method stub
101         return false;
102     }
103 }
104 }
105 }

```

This activity creates string arrays with sample data and passes it into the ListViewAdapter class. The sample data can be easily changed to display different items and prices. On listview allows the user to press and hold an item, which puts them in select mode. While in select mode multiple items can be selected at once. This will allow multiple items to be deleted at the same time.

listview_main.xml

1. Create an XML graphical layout for your MainActivity
2. Expand res
3. Expand layout
4. Right Click on layout and select New > Android XML File
5. Name the XML file listview_main.xml
6. Paste the following code.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:layout_width="fill_parent"
4     android:layout_height="fill_parent" >
5
6     <ListView
7         android:id="@+id/listview"
8         android:layout_width="fill_parent"
9         android:layout_height="fill_parent" />
10
11 </RelativeLayout>
```

activity_main.xml

1. Create an XML graphical layout for your contextual action bar
2. Expand res
3. Right Click on menu
4. Select New > Android XML File
5. Name your XML file activity_main.xml
6. Paste the following code.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <menu xmlns:android="http://schemas.android.com/apk/res/android" >
3
4     <item
5         android:id="@+id/delete"
6         android:title="@string/delete"/>
7
```

Shopping.java

1. Create an array class
2. Select File > New > Class
3. Name it Shopping.java
4. Select the package com.androidbegin.multipledeteletelstview
5. Click Finish
6. Open Shopping.java
7. Paste the following code

```
1 package com.androidbegin.multipledeteletelstview;
2
3 public class Shopping {
4     private String Item;
5     private String Price;
6
7     public Shopping(String Item, String Price) {
8         this.Item = Item;
9         this.Price = Price;
10    }
11
12    public String getItem() {
13        return Item;
14    }
15
16    public void setItem(String Item) {
17        this.Item = Item;
18    }
19
20    public String getPrice() {
21        return Price;
22    }
23
24    public void setPrice(String Price) {
25        this.Price = Price;
26    }
27 }
```

ListViewAdapter.java

1. Create a custom listview adapter
2. Select File > New > Class
3. Name it ListViewAdapter.java
4. Select the package com.androidbegin.multipledeteletelstview
5. Click Finish.
6. Open ListViewAdapter.java

7. Paste the following code.

```
1 package com.androidbegin.multipledeteletelistview;
2
3 import android.util.SparseBooleanArray;
4 import android.view.LayoutInflater;
5 import android.view.View;
6 import android.view.ViewGroup;
7 import android.widget.TextView;
8 import java.util.List;
9
10 import android.content.Context;
11 import android.widget.AdapterView;
12 import android.widget.ImageView;
13
14 public class ListViewAdapter extends ArrayAdapter<Shopping> {
15     // Declare Variables
16     Context context;
17     LayoutInflater inflater;
18     List<Shopping> shoppinglist;
19     private SparseBooleanArray mSelectedItemsIds;
20
21     public ListViewAdapter(Context context, int resourceId,
22         List<Shopping> shoppinglist) {
23         super(context, resourceId, shoppinglist);
24         mSelectedItemsIds = new SparseBooleanArray();
25         this.context = context;
26         this.shoppinglist = shoppinglist;
27         inflater = LayoutInflater.from(context);
28     }
29
30     private class ViewHolder {
31         TextView Item;
32         TextView Price;
33     }
34
35     public View getView(int position, View view, ViewGroup parent) {
36         final ViewHolder holder;
37         if (view == null) {
38             holder = new ViewHolder();
39             view = inflater.inflate(R.layout.listview_item, null);
40             // Locate the TextViews in listview_item.xml
41             holder.Item = (TextView) view.findViewById(R.id.Item);
42             holder.Price = (TextView) view.findViewById(R.id.Price);
43             view.setTag(holder);
44         } else {
45             holder = (ViewHolder) view.getTag();
46         }
47         // Capture position and set to the TextViews
48         holder.Item.setText(shoppinglist.get(position).getItem());
49         holder.Price.setText(shoppinglist.get(position).getPrice());
50         return view;
51     }
52     @Override
53     public void remove(Shopping object) {
54         shoppinglist.remove(object);
55         notifyDataSetChanged();
56     }
57     public List<Shopping> getShopping() {
58         return shoppinglist;
59     }
59 }
```

```

59     }
60
61     public void toggleSelection(int position) {
62         selectView(position, !mSelectedItemsIds.get(position));
63     }
64
65     public void removeSelection() {
66         mSelectedItemsIds = new SparseBooleanArray();
67         notifyDataSetChanged();
68     }
69
70     public void selectView(int position, boolean value) {
71         if (value)
72             mSelectedItemsIds.put(position, value);
73         else
74             mSelectedItemsIds.delete(position);
75         notifyDataSetChanged();
76     }
77
78     public int getSelectedCount() {
79         return mSelectedItemsIds.size();
80     }
81
82     public SparseBooleanArray getSelectedIds() {
83         return mSelectedItemsIds;
84     }
85 }

```

In this listview adapter class, string arrays are passed into the ListViewAdapter and set into the TextViews followed by the positions.

listview_item.xml

1. Create an XML graphical layout for your listview item
2. Expand res
3. Right Click on layout
4. Select New > Android XML File
5. Name the XML file listview_item.xml
6. Paste the following code.

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      android:layout_width="fill_parent"
4      android:layout_height="fill_parent"
5      android:background="?android:attr/activatedBackgroundIndicator" >
6
7      <TextView
8          android:id="@+id/Itemlabel"
9          android:layout_width="wrap_content"
10         android:layout_height="wrap_content"
11         android:text="@string/Itemlabel" />
12
13     <TextView
14         android:id="@+id/Item"

```

```

15         android:layout_width="wrap_content"
16         android:layout_height="wrap_content"
17         android:layout_toRightOf="@+id/Itemlabel" />
18
19     <TextView
20         android:id="@+id/Pricelabel"
21         android:layout_width="wrap_content"
22         android:layout_height="wrap_content"
23         android:layout_below="@+id/Itemlabel"
24         android:text="@string/Pricelabel" />
25
26     <TextView
27         android:id="@+id/Price"
28         android:layout_width="wrap_content"
29         android:layout_height="wrap_content"
30         android:layout_below="@+id/Item"
31         android:layout_toRightOf="@+id/Pricelabel" />
32
33 </RelativeLayout>

```

strings.xml

1. Change the application name and texts
2. Open your strings.xml by expanding res > values folder
3. Paste the following code.

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3
4     <string name="app_name">Multiple Selection ListView</string>
5     <string name="hello_world">Hello world!</string>
6     <string name="Itemlabel">""</string>
7     <string name="Pricelabel">""</string>
8     <string name="delete">Delete</string>
9
10 </resources>

```

AndroidManifest.xml

1. Set the minimum SDK version to 11
2. Open AndroidManifest.xml
3. Paste the following code.

```

1 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
2     package="com.androidbegin.multipledeletelistview"
3     android:versionCode="1"
4     android:versionName="1.0" >
5
6     <uses-sdk
7         android:minSdkVersion="11"
8         android:targetSdkVersion="15" />
9
10    <application
11        android:icon="@drawable/ic_launcher"

```

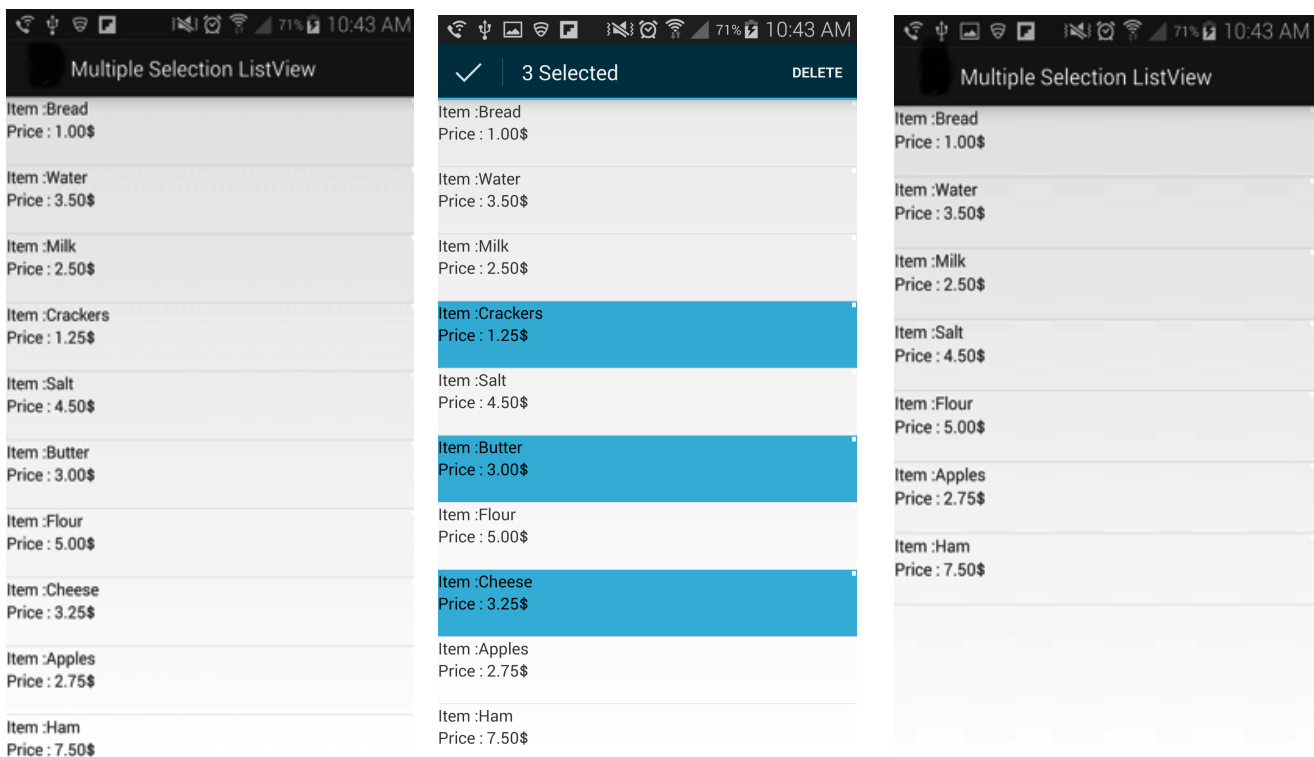
```

12         android:label="@string/app_name"
13         android:theme="@style/AppTheme" >
14         <activity android:name=".MainActivity" >
15             <intent-filter>
16                 <action android:name="android.intent.action.MAIN" />
17
18                 <category android:name="android.intent.category.LAUNCHER" />
19             </intent-filter>
20         </activity>
21     </application>
22
23 </manifest>

```

Congratulations you have just created a list application with multiple delete capabilities!

Demonstration



Conclusion

As seen in the demonstration, the Multiple Delete List View application is able to display items and prices specified within MainActivity.java. By pressing and holding an item the app goes into select mode. The user then selects the items that they wish to delete. Once the delete button is pressed, the items are

removed from the list. Lists are often used in android application and items need to be frequently added and removed. Having the knowledge of creating lists with deletion is essential to anyone looking to develop an android application that will require lists.

References

- [1] "Creating an Android Project." *Android Developers*. Android, n.d. Web. 11 Nov. 2014.
- [2] "Android ListView Tutorial." *AndroidBegin*. N.p., 13 Oct. 2013. Web. 11 Nov. 2014.
- [3] Scocco, Daniel. "Essential Tips to Writing the Ultimate Tutorial." *Daily Blog Tips*. N.p., n.d. Web. 11 Nov. 2014.
- [4] "Android App Development Tutorial Beta." *Android Tutorial*. N.p., n.d. Web. 11 Nov. 2014.