NEW MEDIA &
COMMUNICATION
TECHNOLOGY

Mobile App Development

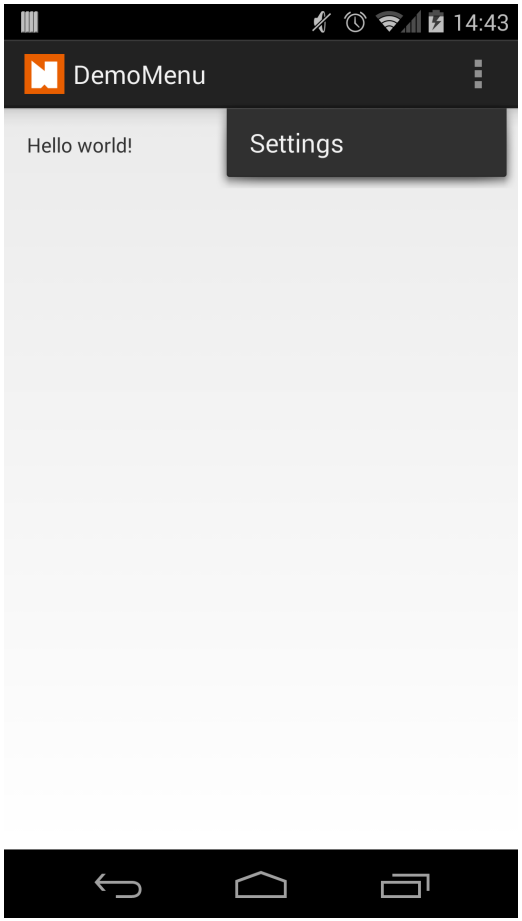
“As far as the customer is concerned,
the interface is the product.”

Menus, Actionbars and ...

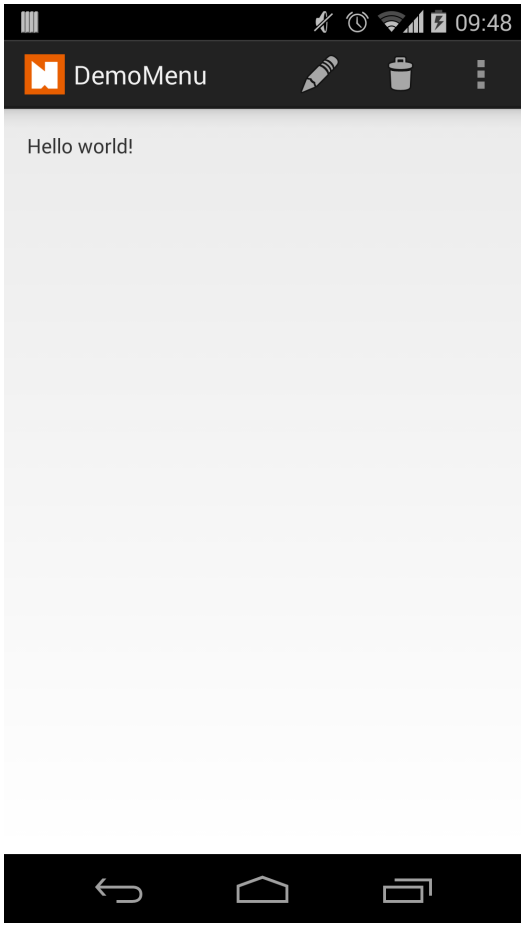
Actions & Menus, Context Menus, Popup Menus, ...

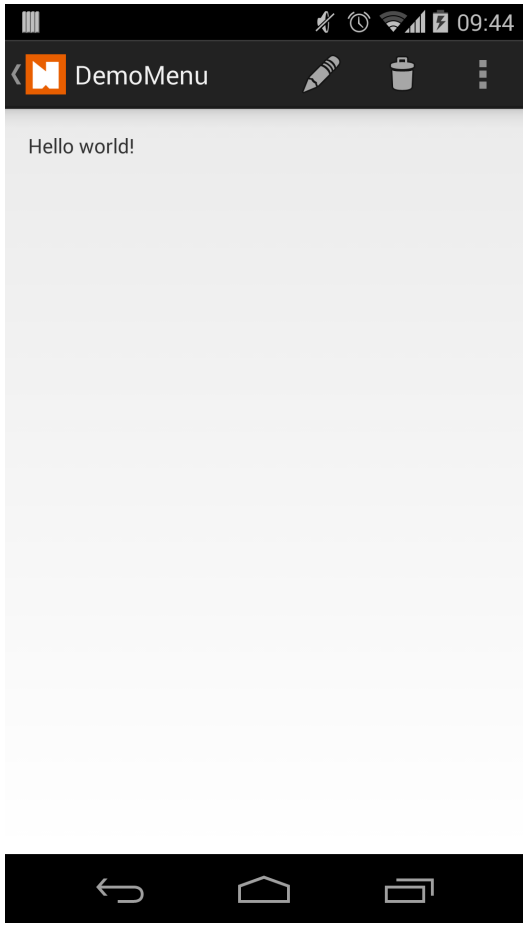
Old and New

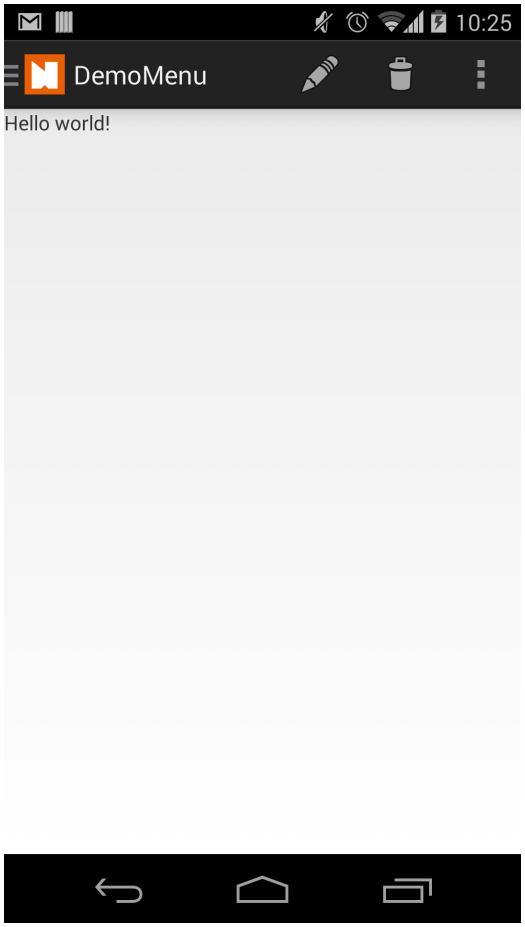

08:31
DemoMenu
Hello world!
Settings


14:43
DemoMenu
Hello world!
Settings

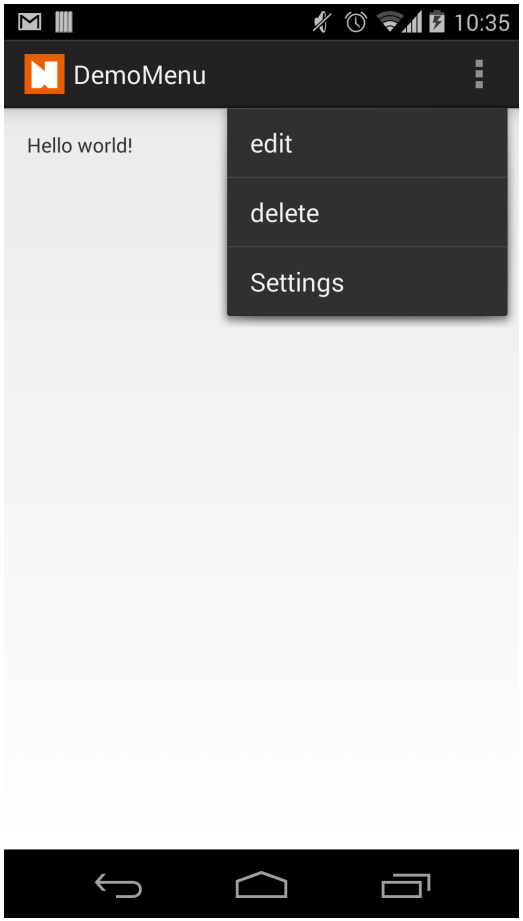
Components of the actionbar







Basic Menu




```
<menu xmlns:android="..." >

    <item android:id="@+id/action_edit"
          android:icon="@drawable/ic_menu_edit"
          android:title="@string/action_edit" />

    <item android:id="@+id/action_delete"
          android:icon="@drawable/ic_menu_delete"
          android:title="@string/action_delete"  />

    <item android:id="@+id/action_settings"
          android:orderInCategory="100"
          android:showAsAction="never" />

</menu>
```

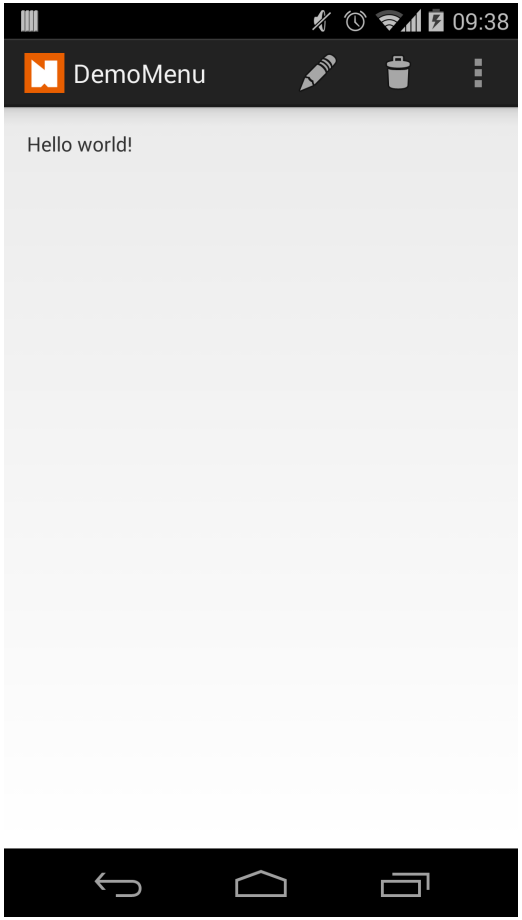
Creating & handling a menu

```
public class MainActivity extends Activity {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

    public boolean onOptionsItemSelected(MenuItem item) {
        switch(item.getItemId()) {
        case R.id.action_edit:
            /* do something */
            return true;
        case R.id.action_delete:
            /* do something */
            return true;
        case R.id.action_settings:
            /* do something */
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

Changing menu items at Runtime

```
public class MainActivity extends Activity {
    private MenuItem editMenuItem;
    private boolean mIsEditable;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        editMenuItem = menu.findItem(R.id.action_edit);
        return true;
    }

    public boolean onPrepareOptionsMenu(Menu menu) {
        super.onPrepareOptionsMenu(menu);
        editMenuItem.setVisible(mIsEditable);
        return true;
    }

    public void onToggleMenuItems(View v) {
        mIsEditable = !mIsEditable;
        invalidateOptionsMenu();
    }
}
```

showAsAction

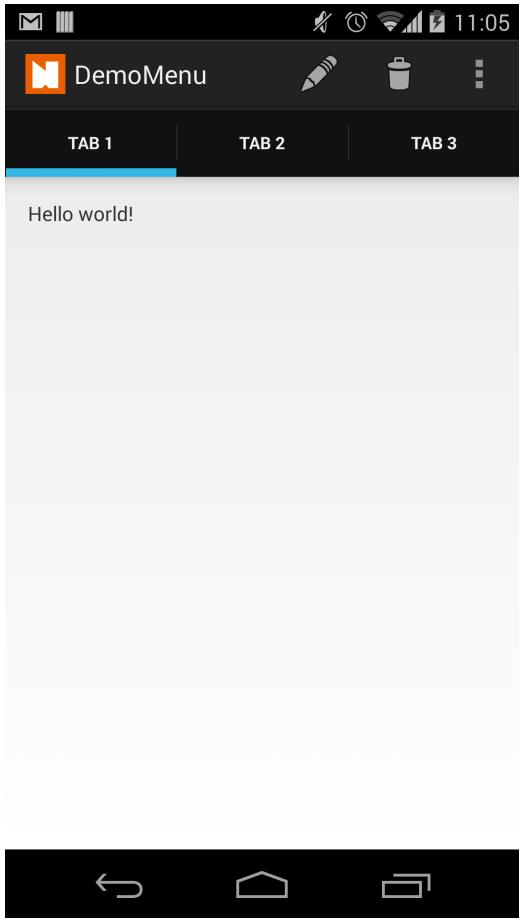



```
<menu xmlns:android="..." >

    <item android:id="@+id/action_edit"
          android:icon="@drawable/ic_menu_edit"
          android:title="@string/action_edit"
          android:showAsAction="ifRoom"/>

    <item android:id="@+id/action_delete"
          android:icon="@drawable/ic_menu_delete"
          android:title="@string/action_delete"
          android:showAsAction="ifRoom"/>

    <item android:id="@+id/action_settings"
          android:orderInCategory="100"
          android:showAsAction="never"
          android:title="@string/action_settings" />

</menu>
```

Icons in Actionbar

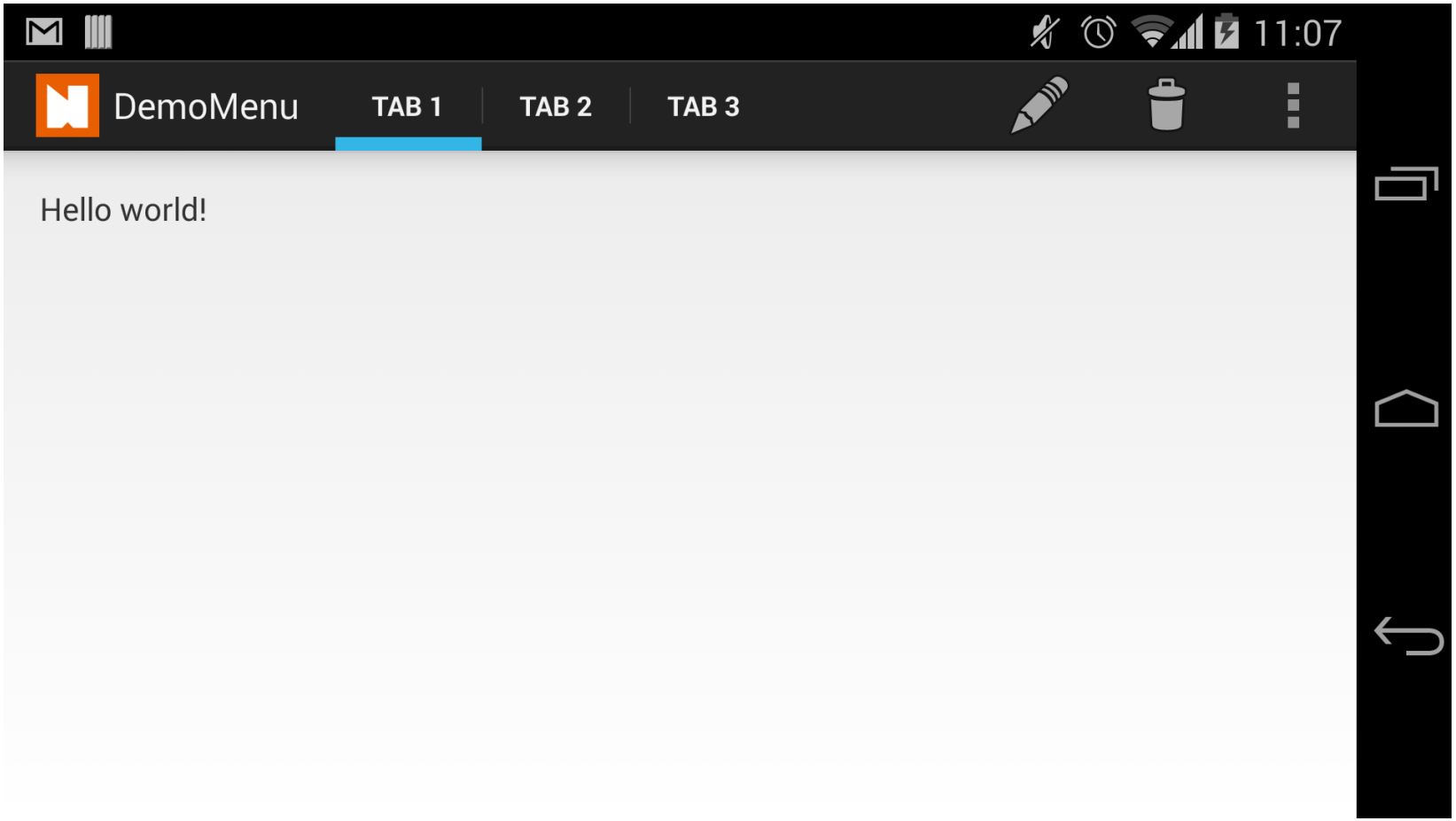
Device	Orientation	Horz. Dp	Icons	Example
Nexus S	Portrait	320	2	oo
Galaxy Nexus	Portrait	360	3	oo=
Nexus S	Landscape	534	4	oooo
7" Tablet	Portrait	600	5	oooo=
Galaxy Nexus	Landscape	640	5	oooo=
10" Tablet	Portrait	800	5	oooo=
7" Tablet	Landscape	1024	5	oooo=
10" Tablet	Landscape	1280	5	oooo=

Adding Tab Navigation

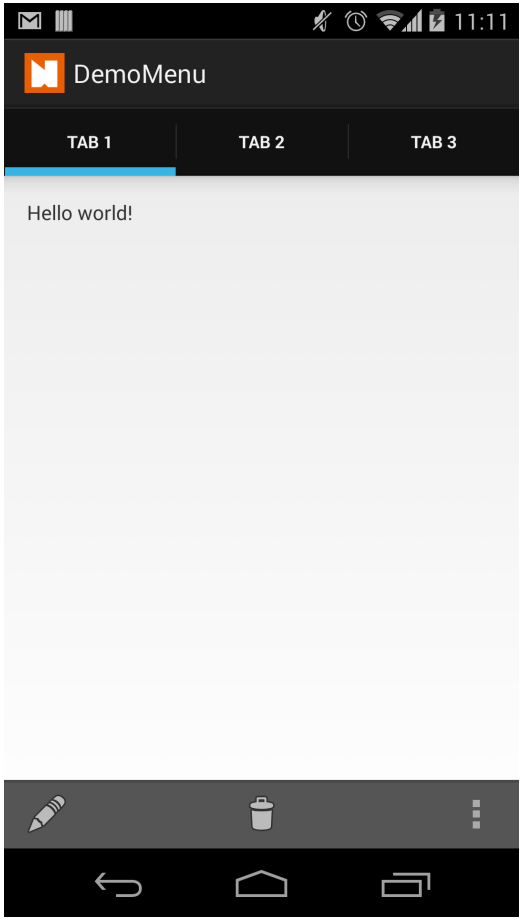



```
public class MainActivity extends Activity {
    protected void onCreate(Bundle sis) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ActionBar actionBar = getActionBar();
        actionBar.setNavigationMode(
                ActionBar.NAVIGATION_MODE_TABS);
        for (int i = 0; i < 3; i++) {
            Tab tab = actionBar
                .newTab()
                .setText("Tab " + (i + 1))
                .setTabListener(mTabListener);
            actionBar.addTab(tab);
        }
    }

    TabListener mTabListener = new TabListener() {
        /* implement methods */
    }
}
```

Tabs in wide mode

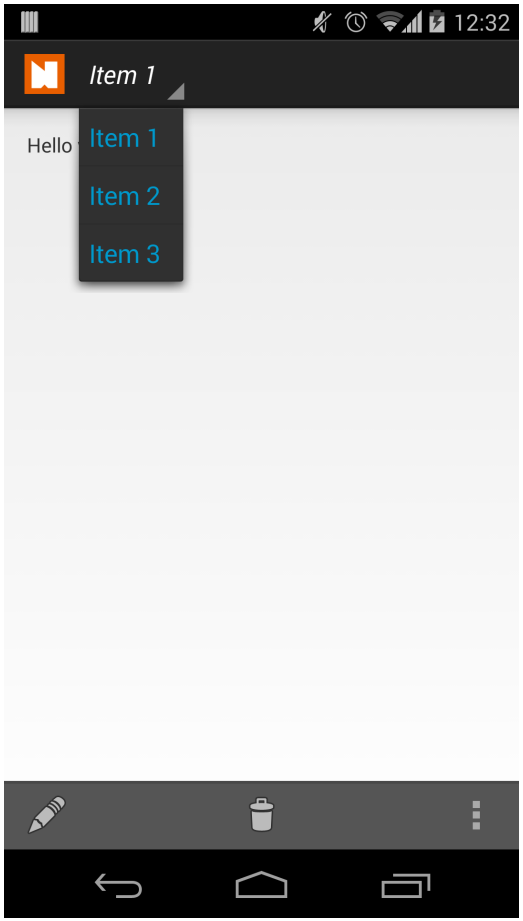


Split Actionbar



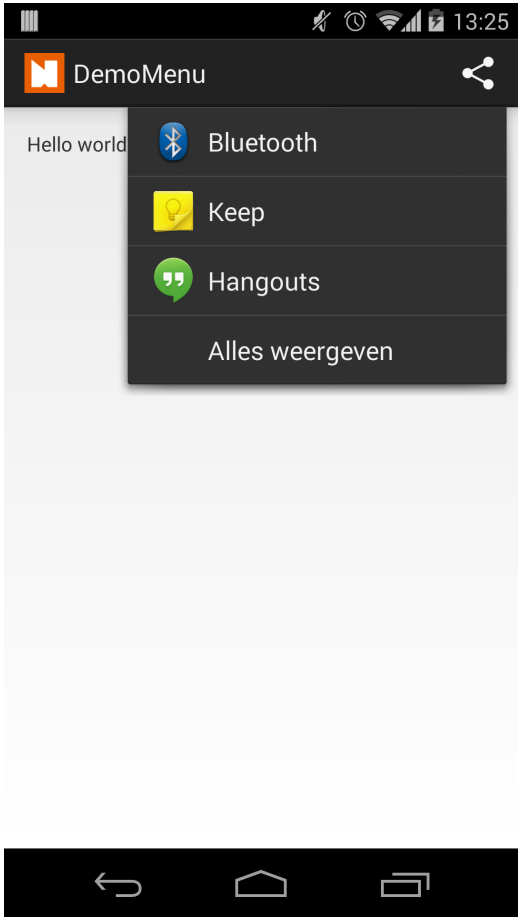

```
<activity
    android:name=".MainActivity"
    android:label="@string/app_name"
    android:uiOptions="splitActionBarWhenNarrow" >
    <!--  intent-filter -->

    <meta-data
        android:name="android.support.UI_OPTIONS"
        android:value="splitActionBarWhenNarrow"  />

</activity>
```

Adding dropdown navigation




```
public class MainActivity extends Activity {
    protected void onCreate(Bundle sis) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ActionBar actionBar = getActionBar();
        actionBar.setNavigationMode(
            ActionBar.NAVIGATION_MODE_LIST);
        actionBar.setDisplayShowTitleEnabled(false);
        ArrayAdapter<String> adapter
            = new ArrayAdapter<String>(
                this, R.layout.actionbar_spinner,
                new String[] {"Item 1", "Item 2",
                    "Item 3"});
        adapter.setDropDownViewResource(
            R.layout.actionbar_spinner_dropdown);
        actionBar.setListNavigationCallbacks(
            adapter, mListener);
    }

}
```

Using the ShareActionProvider




```
<menu xmlns:android="...">
    <item android:id="@+id/action_share"
          android:icon="@drawable/ic_menu_share"
          android:title="@string/action_edit"
          android:actionProviderClass=
               "android.widget.ShareActionProvider"
          android:showAsAction="always" />
</menu>
```

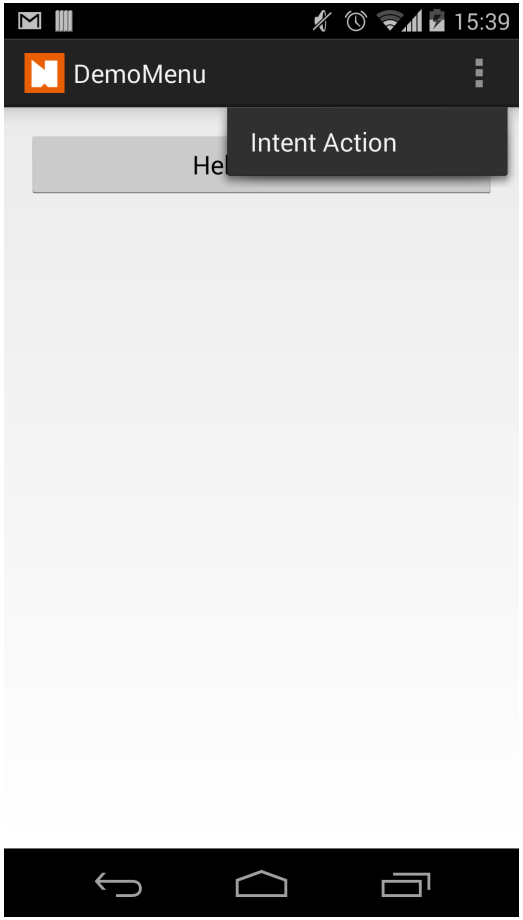
Using the ShareActionProvider

```
public class MainActivity extends Activity {
    private ShareActionProvider mShareProvider;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main, menu);
        MenuItem item = menu.findItem(R.id.action_share);
        mShareProvider = (ShareActionProvider) item.getActionProvider();

        return true;
    }

    public void onItemSelected(View v) {
         Intent shareIntent = new Intent(Intent.ACTION_SEND);
         shareIntent.setType("text/plain");
         shareIntent.putExtra(Intent.EXTRA_TEXT, "Hello world");

         mShareProvider.setShareIntent(shareIntent);
    }

}
```

Custom ActionProvider

```
public class SettingsActionProvider extends ActionProvider {
    private static final Intent sSettingsIntent = new Intent(Settings.ACTION_SETTINGS);
    private final Context mContext;

    public SettingsActionProvider(Context context) {
        super(context);
        mContext = context;
    }

    public View onCreateActionView() {
        LayoutInflater layoutInflater = LayoutInflater.from(mContext);
        View view = layoutInflater.inflate(R.layout.settings_action_provider, null);
        ImageButton button = (ImageButton) view.findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                mContext.startActivity(sSettingsIntent);
            }
        });
        return view;
    }

    public boolean onPerformDefaultAction() {
        mContext.startActivity(sSettingsIntent);
        return true;
    }
}
```

Populating Menus through Intent




```
public boolean onCreateOptionsMenu(Menu menu) {
    super.onCreateOptionsMenu(menu);
    Intent intent = new Intent(
        null, Uri.parse("myscheme://some_url"));
    intent.addCategory(Intent.CATEGORY_ALTERNATIVE);

    menu.addIntentOptions(
        Menu.NONE,
        Menu.NONE,
        Menu.NONE,
        this.getComponentName(),
        null,
        intent,
        0,
        null);

    return true;
}
```

Taking part in intent menus

```
<activity
    android:name="be.howest.nmct.menus.AlternativeActivity"
    android:label="@string/title_activity_alternative"  >

    <intent-filter android:label="Intent Action" >
        <action android:name="be.howest.nmct.actions.TROLOLOLOLOOO"  />

        <category android:name="android.intent.category.DEFAULT"  />
        <category android:name="android.intent.category.ALTERNATIVE"  />
        <category android:name="android.intent.category.SELECTED_ALTERNATIVE"  />

        <data android:scheme="myscheme" />

     </intent-filter>

</activity>
```

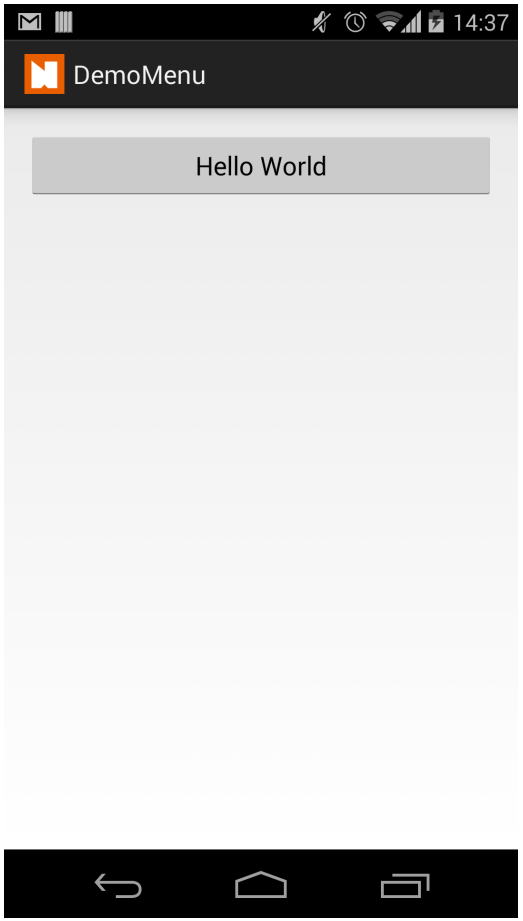
Old style context menus

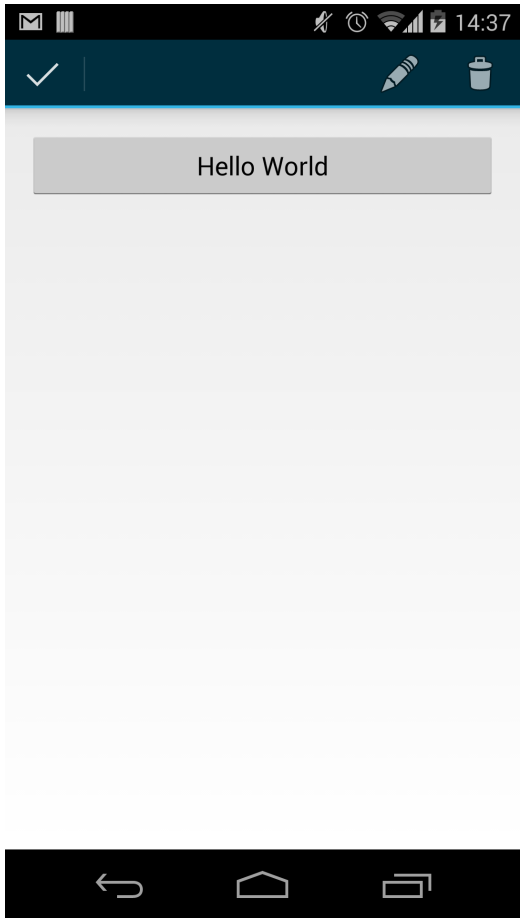

```
<RelativeLayout xmlns:android="..."
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true" >
    </ListView>

</RelativeLayout>
```

Old style context menus

```
public class MainActivity extends Activity {
    private ListView mListView;
    private ArrayAdapter<CharSequence> mAdapter;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mListView = (ListView)  findViewById(R.id.listView);
        mAdapter = ArrayAdapter. createFromResource (
            this, R.array.some_items,android.R.layout.simple_list_item_1);
        mListView. setAdapter(mAdapter);

        registerForContextMenu (mListView);
    }

    public void onCreateContextMenu(ContextMenu m, View v, ContextMenuInfo mi) {
        if(v == mListView) { getMenuInflater().inflate(R.menu.main, m); }
    }

    public boolean onContextItemSelected(MenuItem item) {
        AdapterContextMenuInfo cmi  = (AdapterContextMenuInfo)item. getMenuInfo();
        if(item.getItemId() == R.id.action_delete) { /* delete(cmi.id); */ return true; }
        return super.onContextItemSelected(item);
    }

}
```

contextual action mode





contextual action mode

```
public class MainActivity extends Activity {
    private ListView mListView;
    private ArrayAdapter<CharSequence> mAdapter;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }


    public void onButtonClicked(View v) {
        startActionMode(mActionModeCallback);
    }

}
```

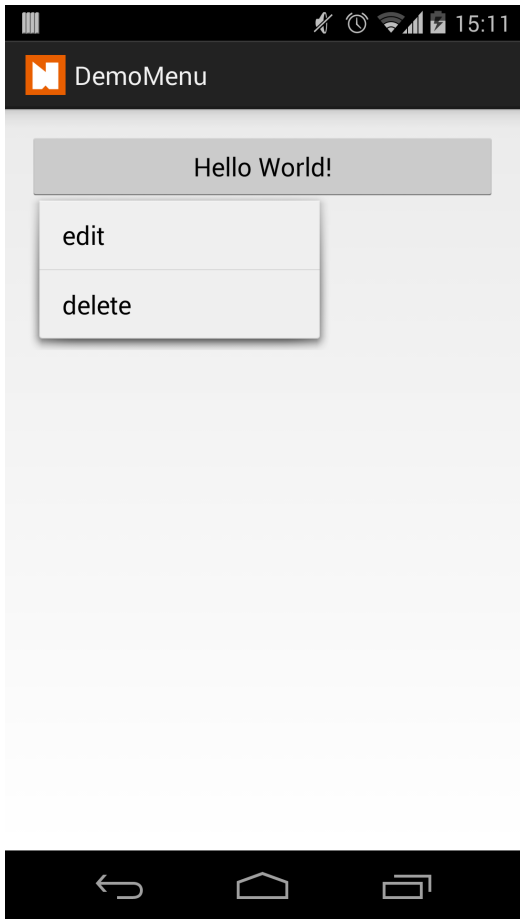
contextual action mode

```
private ActionMode.Callback mActionModeCallback =  new ActionMode.Callback() {

    public boolean onCreateActionMode(ActionMode mode, Menu menu) {
        MenuInflater inflater = mode.getMenuInflater();
        inflater.inflate(R.menu.main, menu);
        return true;
    }

    public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
        return false;
    }

    public boolean onActionItemClicked(ActionMode mode, MenuItem item) {
        switch (item.getItemId()) {
        case R.id.action_delete:
            /* delete() */
            mode.finish();
            return true;
        default:
            return false;
        }
    }

    public void onDestroyActionMode(ActionMode mode) {}

};
```

Lists and ActionMode

```
public class MainActivity extends Activity {
    private ListView mListView;
    private ArrayAdapter<CharSequence> mAdapter;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mListView = (ListView)  findViewById(R.id.list);
        mListView.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);
        mListView.setMultiChoiceModeListener(mListener)
    }

    private MultiChoiceModeListener mListener =  new MultiChoiceModeListener() {
        /* Same methods as ActionModeCallback Listener */

        public void onItemCheckedStateChanged(ActionMode mode,  int position,
                                              long id, boolean checked) {  }

    };
}
```

Popup menus




```
public class MainActivity extends Activity {
    private ListView mListView;
    private ArrayAdapter<CharSequence> mAdapter;

    protected void onCreate(Bundle sis) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void onShowPopup(View v) {
        PopupMenu popup = new PopupMenu(this, v);
        popup.inflate(R.menu.main);
        popup.setOnMenuItemClickListener(mListener);
        popup.show();
    }
}
```

Styles and Themes

Meuh ....

Creating Styles

```
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textColor="#00FF00"
    android:typeface="monospace"
    android:text="@string/hello" />
```

```
<resources>
    <style name="CodeFont" parent="@android:style/TextAppearance.Medium">
        <item name="android:layout_width">fill_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:textColor">#00FF00</item>
        <item name="android:typeface">monospace</item>
    </style>
</resources>
```

```
<TextView style="@style/CodeFont" android:text="@string/hello" />
```

Style Attributes

```
public TextView(Context context, AttributeSet attrs,  int defStyle) {
    super(context, attrs, defStyle);
    Resources res = getResources();
    Resources.Theme theme = context.getTheme();

    TypedArray a = theme.obtainStyledAttributes(
        attrs, com.android.internal.R.styleable.TextViewAppearance, defStyle,  0);

    int ap = a.getResourceId(
        com.android.internal.R.styleable.TextViewAppearance_textAppearance, -1);
    if (ap != -1)
        appearance = theme.obtainStyledAttributes( ap,
            com.android.internal.R.styleable.TextAppearance);

    ColorStateList textColorHint = null;
    if (appearance != null)
        textColorHighlight = appearance.getColor(
            com.android.internal.R.styleable.TextAppearance_textColor, textColorHighlight);


    a = theme.obtainStyledAttributes(
        attrs, com.android.internal.R.styleable.TextView, defStyle,  0);
    textColorHighlight = a.getColorStateList(
        com.android.internal.R.styleable.TextView_textColorHint);
```

Style Attributes

```
<declare-styleable name="TextAppearance">
    <attr name="textColor" />
    <attr name="textSize" />
    <attr name="textStyle" />
    <attr name="typeface" />
    <attr name="fontFamily" />
    <attr name="textColorHighlight" />
    <attr name="textColorHint" />
    <attr name="textColorLink" />
    <attr name="textAllCaps" format="boolean" />
    <attr name="shadowColor" format="color" />
    <attr name="shadowDx" format="float" />
    <attr name="shadowDy" format="float" />
    <attr name="shadowRadius" format="float" />
</declare-styleable>

<declare-styleable name="TextView">
    <attr name="textAppearance" />
    <attr name="textColorHighlight" />
    <!- And many more. -- >
</declare-styleable>
```

Style Attributes

```
<declare-styleable name="TextAppearance">
    <attr name="textColor" />
    <attr name="textSize" />
    <attr name="textStyle" />
    <attr name="typeface" />
    <attr name="fontFamily" />
    <attr name="textColorHighlight" />
    <attr name="textColorHint" />
    <attr name="textColorLink" />
    <attr name="textAllCaps" format="boolean" />
    <attr name="shadowColor" format="color" />
    <attr name="shadowDx" format="float" />
    <attr name="shadowDy" format="float" />
    <attr name="shadowRadius" format="float" />
</declare-styleable>

<declare-styleable name="TextView">
    <attr name="textAppearance" />
    <attr name="textColorHighlight" />
    <!- And many more. -- >
</declare-styleable>
```

Style

```
<style name="TextAppearance">
    <item name="android:textColor">?textColorPrimary</item>
    <item name="android:textColorHighlight">?textColorHighlight</item>
    <item name="android:textColorHint">?textColorHint</item>
    <item name="android:textColorLink">?textColorLink</item>
    <item name="android:textSize">16sp</item>
    <item name="android:textStyle">normal</item>
</style>

<style name="TextAppearance.Small">
    <item name="android:textSize">14sp</item>
    <item name="android:textColor">?textColorSecondary</item>
</style>

<style name="Widget">
    <item name="android:textAppearance">?textAppearance</item>
</style>

<style name="Widget.TextView">
    <item name="android:textAppearance">?android:attr/textAppearanceSmall</item>
    <item name="android:textSelectHandleLeft">?android:attr/textSelectHandleLeft</item>
    <!- And many more. -->
</style>
```

Theme?

```
<style name="Theme">

    <!- Attributes defined in theme -->
    <item name="textColorPrimary" >@android:color/primary_text_dark </item>
    <item name="textColorHighlight" >@android:color/highlighted_text_dark </item>
    <item name="colorForeground" >@android:color/bright_foreground_dark </item>
    <item name="textAppearanceSmall" >@android:style/TextAppearance.Small </item>
    <item name="textColorHint" >@android:color/hint_foreground_dark </item>
    <item name="textColorSecondary" >@android:color/secondary_text_dark </item>



    <!- widget styles defined in theme -->
    <item name="textViewStyle" >@android:style/Widget.TextView </item>
    <item name="editTextStyle" >@android:style/Widget.EditText </item>

    <!-- And many more. -->

</style>
```

Theme Attributes?

```
<declare-styleable name="Theme">

    <attr name="textColorPrimary"  format="reference|color"  />
    <attr name="textColorSecondary"  format="reference|color"  />
    <attr name="textAppearanceSmall"  format="reference"  />
    <attr name="textColorHighlight"  format="reference|color"  />


    <attr name="textViewStyle"  format="reference"  />
    <attr name="editTextStyle"  format="reference"  />
    <!-- And many more. -->

</declare-styleable>
```

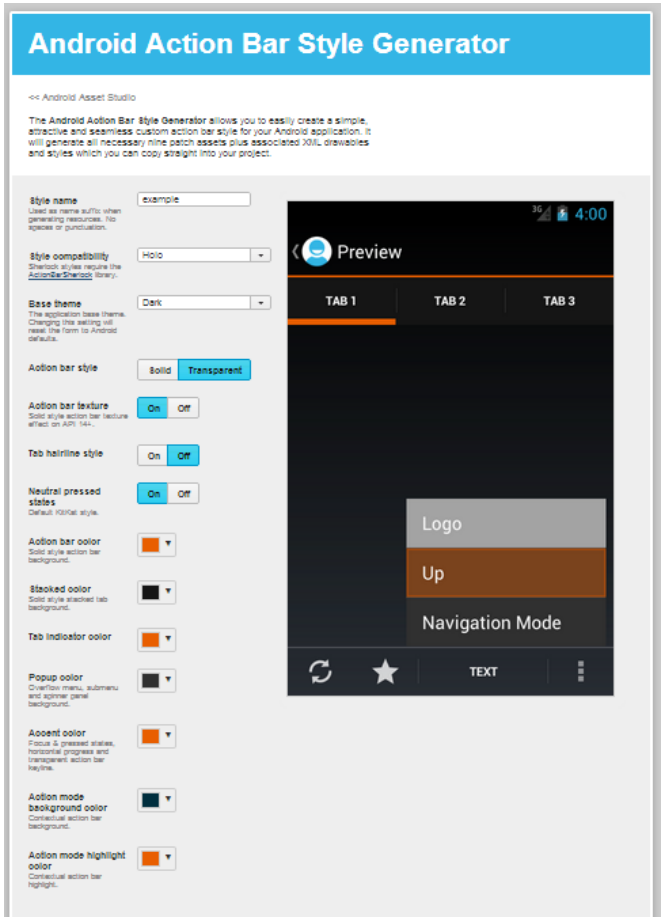
Holo?

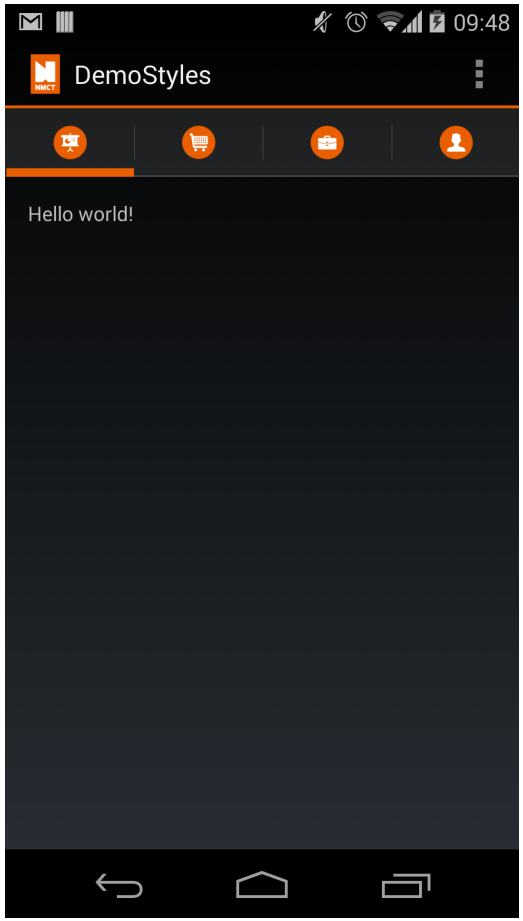
```
<style name="Theme.Holo">

    <item name="colorForeground">@android:color/bright_foreground_holo_dark </item>
    <item name="colorBackground">@android:color/background_holo_dark </item>
    <item name="textAppearance">@android:style/TextAppearance.Holo </item>

    <item name="textViewStyle">@android:style/Widget.Holo.TextView </item>

    <!-- And many more. -->

</style>

<style name="Widget.Holo.TextView"  parent="Widget.TextView">
</style>
```

Applying a theme

```
<application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/Theme.Example" >

        <activity
            android:theme="@style/Theme.Example"
            android:name="be.howest.nmct.ds.MainActivity"
            android:label="@string/app_name" >

        </activity>

</application>
```

Styling the actionbar





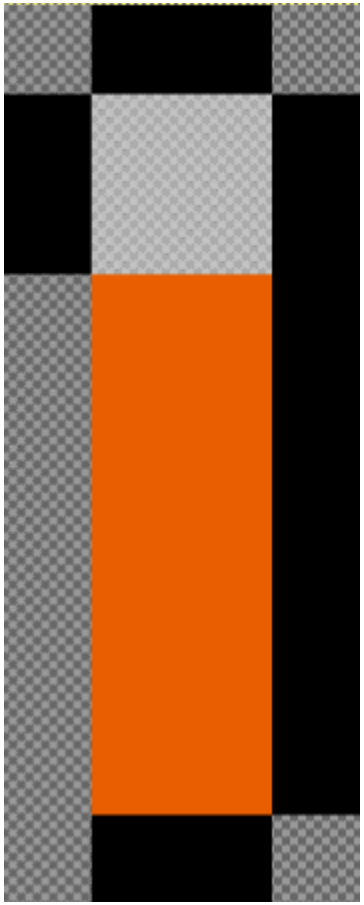
Styling the actionbar

```
<resources>

    <style name="Theme.Example"  parent="@android:style/Theme.Holo" >
        <item name="android:actionBarTabStyle" >@style/ActionBarTabStyle.Example </item>
        <item name="android:actionDropDownStyle" >@style/DropDownNav.Example </item>
        <item name="android:actionBarStyle" >@style/ActionBar.Transparent.Example </item>

        <!-- And many more. -->

    </style>

    <style name="ActionBarTabStyle.Example"
        parent="@android:style/Widget.Holo.ActionBar.TabView" >
        <item name="android:background" >@drawable/tab_indicator_ab_example </item>
    </style>

    <!-- And many more. -->

<resources>
```

Handling States

```
<selector xmlns:android="http://schemas.android.com/apk/res/android" >
    <!-- Non focused states -- >
    <item android:state_focused="false" android:state_selected="false"
        android:state_pressed="false" android:drawable="@android:color/transparent"  />
    <item android:state_focused="false" android:state_selected="true"
        android:state_pressed="false" android:drawable="@drawable/tab_selected"  />

    <!-- Focused states -- >
    <item android:state_focused="true" android:state_selected="false"
        android:state_pressed="false" android:drawable="@drawable/tab_unselected_focused"  />
    <item android:state_focused="true" android:state_selected="true"
        android:state_pressed="false" android:drawable="@drawable/tab_selected_focused"  />

    <!-- Pressed -- >
    <!--    Non focused states -- >
    <item android:state_focused="false" android:state_selected="false"
        android:state_pressed="true" android:drawable="@drawable/tab_unselected_pressed"  />
    <item android:state_focused="false" android:state_selected="true"
        android:state_pressed="true" android:drawable="@drawable/tab_selected_pressed"  />

    <!--    Focused states -- >
    <item android:state_focused="true" android:state_selected="false"
        android:state_pressed="true" android:drawable="@drawable/tab_unselected_pressed"  />
    <item android:state_focused="true" android:state_selected="true"
        android:state_pressed="true" android:drawable="@drawable/tab_selected_pressed"  />

</selector>
```

Handling scaling with 9-patch



Handling scaling with 9-patch


```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentRight="true"
    android:background="@drawable/baloon_bg"
    android:gravity="center_vertical"
    android:text="@string/hello_world" />
```

Styling the other components