

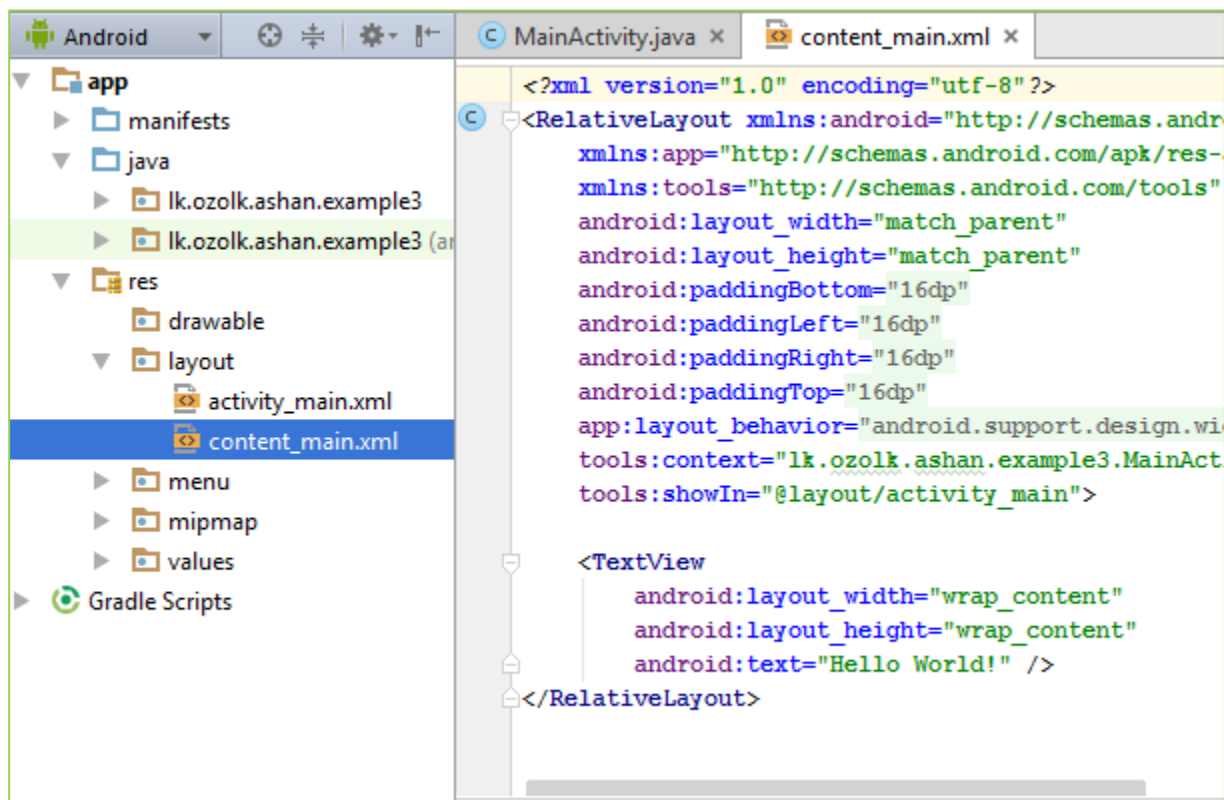
5. What is Android XML layout file ?

කලින් අපි කීවා android වල සෑම single screen එකක් සඳහාම files දෙකක් අවශ්‍ය වන බව. එනම් එක් java file එකක් සහ එක් xml file එකක්. java file එකට අපි activity කියලා කියනවා. xml file එකට layout කියලා කියනවා. දැන් අපි මෙම layout xml file එක අදානය කරමු.

Creates a new blank activity with an app bar.

Activity Name:	MainActivity
Layout Name:	activity_main
Title:	MainActivity
Menu Resource Name:	menu_main
<input type="checkbox"/> Use a Fragment	

මෙහිදී අපිට ජේනවා අලුත් project එක හදන් යද්දී එන මෙම window එකේදී activity එක සඳහා නමක් ඉල්ලනවා. එය fill කරද්දී අනිත් ඒවා ඉබේම fill වෙනවා. එනම් එම activity එක හා සම්බන්ද අනිත් files ටිකට නම් ඉබේම fill වෙනවා. උදාහරණයක් ලෙස එම activity එකට අදාළ layout එකේ නම එහෙම fill වෙනවා ඉබේම. මොකද සෑම screen එකක් සඳහාම java file එකක් සහ layout එකක් එනම් xml file එකක්ද තිබිය යුතුයි.



android studio වලදී, අපි activity එක සඳහා නම දෙද්දී, layout එක සඳහා ඉබේම fill වුන නම වුයේ activity_main යන්න වේ. අපිට ඉහත පින්තූරය දිහා බලපුහාම ජේනවා layout folder එකේ එම නමින් xml file එකක් සෑදී තිබෙනවා. නමුත් ඊට පහලින් තව xml file එකක් තියෙනවා content_main කියලා. අපිට layout එක වෙනස් කිරීම සහ එක එක views වල (එනම් buttons, edit text, වැනි ui components වල) ලක්ෂණ වෙනස් කිරීමට අදාළ codes තියෙන්නේ මෙම content_main xml file එක තුලයි. එය ගැනයි අපි දැන් මේ කතා කරන්නට යන්නේ. එහි ඇති codes වලින් කීපයක් තමා ඉහත රූපයේත් තියෙන්නේ.

XML ගැන බිඳක්

xml යනු extensible Markup Language යන්න වේ. එලසම අපි දන්නවා HTML යනු Hyper Text Markup Language යන්න බව. So markup language එකක් යනු කුමක්ද? Markup language එකක් යනු අවශ්‍ය දත්ත tags අතර පිහිටුවන, එහෙම නැත්නම් අවශ්‍ය දත්ත ඒවා අර්ථ දැක්වෙන tags වලින් වට කරන language එකකට markup language එකක් යැයි කියනු ලැබේ.

HTML භාවිතා කරන්නේ web පිටු නිර්මාණය කිරීම සඳහා වේ. web server එකෙන් අවශ්‍ය දත්ත tags අතර පිහිටුවා එවනවා. එම tags, එහෙම නැත්නම් එම elements predefined ඒවා වේ. එනම් ඒ ඒ tag එක අතර නියම දත්ත කෙසේ display කළ යුතුද කියලා browser එක දන්නවා. So HTML වල භාවිතා කරන elements (tag එකක් යනු වැනි දෙයකටයි. මෙහි b යනු එම tag එකේ element එක වේ) predefined ඒවා වේ. නමුත් xml වල tags predefined නැ. අපිට කැමති ලෙස tags අර්ථ දක්වා ගන්න පුළුවන්.

XML වල ඇති වැදගත්ම වාසිය වන්නේ එය interoperable format එකක් වන නිසයි. xml interoperable යනු, platform independent වේ, technology independent වේ, language independent වේ. එනම් අපිට xml format එක භාවිත කර එක් technology එකක දත්ත ඊට වෙනස් technology එකක් වෙතට transfer කළ හැක. මෙහිසා විවිද technologies, languages වලින් නිපදවා ඇති systems අතර දත්ත හුවමාරුව සඳහා xml භාවිතා කරනු ලැබේ.

XML භාවිතා කර අපිට දත්ත යම් කිසි ආකෘතියකට අනුව store කළ හැක. එම නිසා xml යන්න අපිට database එකක් ලෙසත් භාවිතා කළ හැක. xml වලට භාවිතා කළ යුතු යැයි කියා predefined tags නැ. අපිට කමති විදියට tags අර්ථ දක්වා ගත හැක. උදාහරණයක් ලෙස මට පරිගණකයේ තොරතුරු පහත පරිදි xml file එකක save කරලා තියන්න පුළුවන්.

```
<computer>
  <manufacturer> Asus </manufacturer>
  <model> i7 </model>
  <components>
    <processor> 2.4 GHz intel core i7 </processor>
    <ram> 4GB </ram>
    <price> 90000 </price>
  </components>
</computer>
```

xml file එකක් ලිවීමේදී පිළිපැදිය යුතු නීති - ->

- සෑම xml file එකකටම root element එකක් තිබිය යුතුයි. ඉහත උදාහරණයේනම් root element එක වන්නේ computer යන්නයි. (so xml file එකක් ලියන විට tree structure එක සිහියේ තබාගෙන ලිව්වා නම් හරි)
- සෑම element එකක්ම නියමාකාරව ගොනු කර තිබිය යුතුයි. එනම් සෑම tag එකක්ම නියමිත ස්ථානයේදී close කර තිබිය යුතුයි.
- element එකක් නම් කිරීමේදී spaces තියෙන්න බෑ නමට. first name වැනි දෙයක් නම් <first_name> ලෙස හෝ <firstName> වැනි ක්‍රම භාවිතා කළ යුතුයි.

ඉහතින් සඳහන් කර ඇත්තේ xml file එකක් ලිවීමේදී භාවිතා කළ යුතු මූලික හා පොදු නීති කිහිපයක් වේ. නමුත් අපි හිතමු අපිට අපේම කියලා නීති ටිකක් define කළ යුතුයි කියලා. එනම් උදාහරණයක් ලෙස ඉහත xml code එකේ manufacture යන tag එකේ innerHTML ලෙස අකුරු 4කට වඩා වෙනසක් අනිවාර්යෙන් type කළ යුතුයි වැනි නීතියක්, එහෙම නැත්නම් price tag එකේ innerHTML වශයෙන් type කළ හැක්කේ ඉලක්කම් පමණයි වැනි

නීතියක් දැමීමට අවශ්‍යයැයි විනමු. මෙලස අපේම restriction එකක් භාවිතා කල යුතු වීම DTD (Document Type Definition) එකක් හෝ XSD (xml Scheema Definition) එකක් භාවිතා කල හැක. මෙයින් දැන් බහුලවම භාවිතා කරන්නේ XSD වේ.

android වලදී සෑම xml file එකක්ම XSD එකක් follow කරනු ලබයි. එනම් android වල xml file එකක් code කරද්දී මූලික පොදු නීතිවලට අමතරව, android වලට ආවේනික වූ නීති සම්ප්‍රදායක්ද අපි follow කල යුතුයි xml file එකක් code කරද්දී.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

මෙහිදී මෙම layout xml code එකේ මූලිකම ඇත්තේ එම XSD එක පිහිටි location එකේ address එකයි. එම address එකේ අපිට පේනවා schemas.android ලෙස නියෝගවා. එනම් මේ හිටින්නේ xml Schema Definition එක නියෝග place එකේ address එක වේ

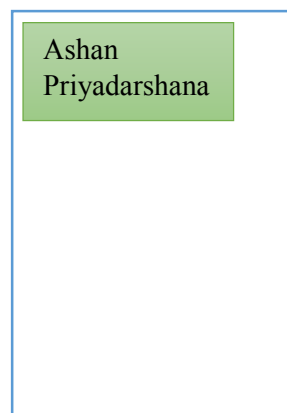
ඉහත code එකේ xmlns : android ලෙස දීලා එයටයි සමාන කර ඇත්තේ XSD එක නියෝග place එකේ url එක. මෙම xmlns යන්නෙහි අරුත නම් xml NameSpace යන්න වේ. ඉන් ඉදිරියි android ලෙස දී ඇත්තේ නිකන් නමකි. එනම් XSD එකේ location එක refere කිරීම සඳහා නමකි. අපිට එය සඳහා කමහිටි නමක් දිය හැක. ඉන් පසුව code එකේ android වෙනුව එම නම භාවිතා කල හැක. එනම් පහත ආකාරයටයි.

```
<RelativeLayout
    xmlns:ashan="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    ashan:layout_width="match_parent"
    ashan:layout_height="match_parent"
    ashan:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
```

දැන් මෙම code එකේ පෙන්වා, කලින් android ලෙස තිබුන දෙය වෙනස් කර ashan ලෙස යොදා ඇති බව. සහ පහල attribute නම් කරද්දී පටන් අරන් නියෝගන්නේ ashan : ලෙසය. android : ලෙස පටන් අරගන්නොත් රතු පාටින් errors ලෙසයි දැන් පෙන්වන්නේ.

layout xml file එකේ දමන tags හා ඒවාහි attributes

අපි හිතමු අපිට අපේ app එකේ පහත පරිදි එක screen එකක් ඕනා නිකන් වචන ටිකක් පමණක් තිබෙන. මේ සඳහා android වල අකුරු display කල හැකි ui components (views) නියෝගවා (textView යනු එලස එක view එකකි). එම component එකක් screen එකට දාල තමයි එහි අකුරු type කල හැක්කේ.



අපගේ අවශ්‍යතාව එය වුනත්, එලස අපිට නිකන්ම views(ui components) එහෙම screen එකේ place කල නොහැක. ඊට ප්‍රථම අපි හඳුන්වා සිටිය යුතුයි එම components වලට screen එකේ පෙළගැසෙන්න ක්‍රමයක්. එනම් එක component එකකට පහලින් අලුත් components place විය යුතුද, නත්තම් ඊට එහා පැත්තෙන් එක පෙළකට සිටින අයුරින් අලුත් components place විය යුතුද, එහම් නැත්තම් කැමති කැමති ස්ථානවලින් එම components place කිරීමට ඉඩ සලසනවාද ලෙස යම් කිසි ක්‍රමයක් හඳුන්වා, ඉන් පසුවයි අපිට ui components screen එකේ place කල හැක්කේ. සැබවින්ම මෙම ක්‍රමයට තමා අපි layout කියලා කියන්නේ. එම නිසා මුලින් layout එකක් හඳුන්වා screen එකට, එම layout එක තුලයි ui components අපි place කල යුත්තේ.

android වල මෙලස layouts කීපයක්ම පවතිනවා අපිට භාවිතා කල හැකි,

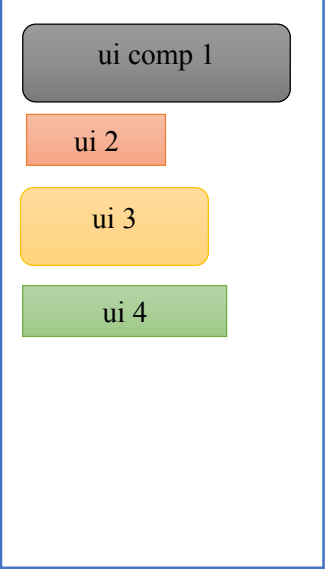
- LinearLayout - එකා පසු පස එකා මෙන් (vertically හෝ horizontally යන දෙයාකාරයටම)
- RelativeLayout - කැමති ස්ථානවලින්. එක් component එකකට සාපේක්ෂව ස්ථානය කොතනද
- TableLayout – table එකක ලෙස. rows හා coloums වල (cell තුල)
-

android වල screen එක මත place කල හැකි ui components විශාල ප්‍රමාණයක් පවතී. පහත ඇත්තේ ඉන් කීපයක් පමණි.

- TextView - අකුරු පෙන්විය හැක.
- EditText – user inputs ලබා ගැනීමට
- Spinner – dropdown list එකක් වගේ.
- Button

LinearLayout

LinearLayout යනු ui group එකක් වේ. මෙම layout එකේදී අපිට ui components සිරස් අතට හෝ තිරස් අතට place කිරීම කල හැක. සිරස් ක්‍රමයද, එහෙම නැත්තම් තිරස් වැඩි පෙළගැස්විය යුත්තේ කියලා අපි LinearLayout tag එක තුල attribute එකක් ලෙස සඳහන් කල යුතුයි.



```

<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"

android:layout_width="match_parent"

android:layout_height="match_parent"

android:orientation="vertical" >

  <ui_comp_1

    android:layout_width="match_parent"

    android:layout_hight="wrap_content"

    android:background="#00ffb0" />

  <ui_2 ..... />

  <ui_3 ..... />

  <ui_4 ..... />

</LinearLayout>

```

අපිට ඉහත xml code එක දිහා බලපුහම ජේනවා ඉස්සෙල්ලාම හදුන්වා ඇත්තේ යම් යම් attributes කීපයක් සමග LinearLayout tag එකකි.එය හා එම tag එකේ closing tag එක වන </LinearLayout> tag එක අතරයි අපිට අවශ්‍ය ui componets place කරලා තියෙන්නේ. So දැන් මේ අනුව අපිට ජේනවා සෑම ui component එකක්ම xml tag එකක් වශයෙන් තමයි අපි නිරූපනය කරන්නේ android වලදී.

ඉහත LinearLayout tag එක තුල යම් යම් attributes කීපයක් ඇත.ඉන් මුල එකනම් අපි දන්නවා.ඒ තමයි මෙම xml file එක පිළිපැදිය යුතු android වලට ආවේනික නීති ටික තියෙන තැන location එකයි එහම මුලින්ම root element එක විදියට අරගෙන තියෙන්නේ.

ඉන් පසුව තියෙන්නේ android:layout_width= “match_parent” හා android:layout_height= “match_parent” ලෙස attributes දෙකකි.මෙයින් කියන්නේ, මෙම component එකේ උස සහ පළල කොපමණ විය යුතුද කියලයි.මේ layout එක නිසා එය screen එක පුරාම විහිදෙන්න කියලා match_parent ලෙස උසට සහ පළල යන දෙකටම දීලා තියෙනවා.මොකද layout එකට parent වන්නේ phone එකේ screen එක වේ.so screen එක පුරාම layout එක විහිදෙන්න කියලයි එතන කියලා තියෙන්නේ.

අපිට ජේනවා ඊළගට තියෙන ui component එකෙන් මේ attribute දෙකම තියෙනවා.නමුත් ඒ සඳහා දීලා තියෙන අගයන් මඳක් වෙනස්.width එක සඳහා match_parent යන්න දී තියෙනවා.එනම් එම ui component එකේ parent ගේ width එකම match කරගෙන ඔයාගේ width එකත් හදා ගන්න කියලයි.මෙහිදී එම ui component එකේ parent වන්නේ layout එකයි.නමුත් එම ui component එකේ උස සඳහා දීලා තියෙන අගය වන්නේ wrap_content යන්න වේ.එහි අදහස එම component එකේ place කරන යම් content එකක් වේද,එම content එක වටේට එතිය හැකි පමණින් උසක් ලබා ගන්න යන්නයි.එනම් සරලව එම content එකේ උසම ගන්න කියලයි.මොකද උස සඳහා match_parent වැනි දෙයක් දුන්නොත් මෙම component එකේ උසද layout එකේ උසම වෙනවා.

අපිට component වල layout_width හා layout_height සඳහා දිය හැකි values පහත පරිදි වේ.

- wrap_content
- match_parent
- pixels වලින්
- dp වලින්

මෙහිදී pixels වලින් දීම අනුමත කරන්නේ නෑ android වලදී.මොකද එවිට එක එක size screen වල අපේ components එක එක size වලින් පෙනිය හැකි නිසයි.එම නිසා e වෙනුවට dp වලින් අගයන් දීමයි අනුමත කරන්නේ android වල.

LinearLayout tag එකේ අවසාන attribute එක වන්නේ orientation යන්න වේ.එය එම tag එකටම ආවේනික වූ attribute එකක් වේ.එම attribute එක සඳහා අපට දිය හැකි අගයන් වන්නේ vertical හා horizontal යන්න වේ.එනම් ui components පෙළගැසිය යුත්තේ එකා පසු පස එකා කුමන ආකාරයටද යන්නයි.

තවත් අපි දැනගත යුතු ලක්ෂණයක් වන්නේ ui component tags වලට closing tags නොමැත.ඒවා වල තියෙන්නේ එම component එක විස්තර කිරීම සඳහා attributes පමණි.එම නිසා එම tags අවසානයේ අපි නිකනම් > ලකුණක් දමනවා වෙනුව /> ලෙසයි type කරන්නේ. So <LinearLayout>, <RelativeLayout>, <TableLayout> වැනි ඒවාට පමණයි closing tags තියෙන්නේ.

තවත් විශේෂ දෙයක් වන්නේ එක xml file එකක තිබිය හැක්කේ එක මේවයින් <LinearLayout>, <RelativeLayout>, <TableLayout> එක tag වර්ගයයි කියා දෙයක් නොමැත.අපිට අවශ්‍ය නම් එක layout tag එක තුල තව layout වර්ගයක tag එකක් අන්තර්ගත කර, ui components අපිට කැමති ආකාරයට screen එකේ place කර ගත හැක.

දැන් අපි බලමු TextView නම් view එකක් (ui component) එකක් ආශ්‍රයෙන් වචන කීපයක් display වන විදියට xml code එකේ ලියන්නේ කොහොමද කියල.

මෙහිදී ප්‍රථමයෙන් අපි සුදුසු ui group එකක් (එනම් ui elements පෙළගැසිය යුතු layout එකක්) තෝරාගෙන එහි attributes ටිකට සුදුසු පරිදි values දිය යුතුයි. ඉන් පසුව TextView element එකක් ගෙන පහත පරිදි වචන ටිකක් screen එකේ display වන පරිදි සැකසිය හැකියි.

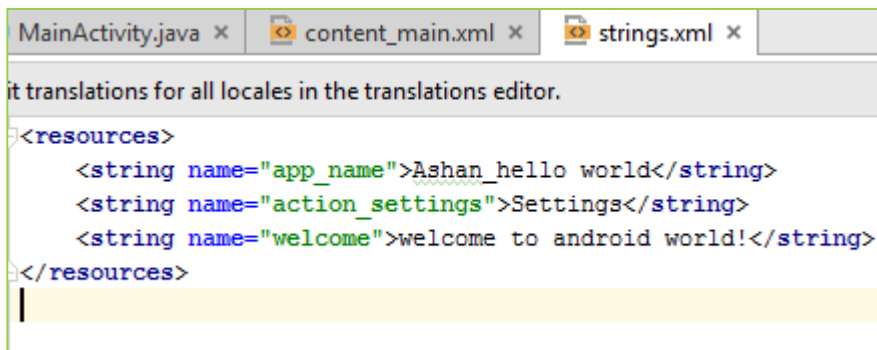
```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#00ff00"
        android:text="welcome to android world!" />

</LinearLayout>
```

අපි මෙලස xml code එකේදී TextView element එක add කර ගන්නවා වෙනුවට දැන් සියලුම, IDE වලින් පහසුකම් සපයනවා නිකන්ම design view එකේදී විවිද ui components ඇදලා දැමීමට layout එක උඩට. එවිට ඇත්තටම අපි xml code එකට ගිහින් බලපුහාම ජේනවා මේ ආකාරයට ඒ ඇදලා දාපු element එකට අදාලව codes ඉබේම type වෙලා තියෙනවා කියල.

තවද මෙසේ අපි code එකෙන් type කරලා element එකක් සාදලා එයට යම් කිසි text එකක් දුන් විට එය error එකක් හෝ warning එකක් පෙන්වන්නවා. එයට හේතුව නම් android වලදී අපි යම් කිසි string එකක් භාවිතා කරනවා නම්, එය resource එකක් ලෙස සලකා res folder එක තුළ ඇති values folder එකේ තියෙන strings.xml file එක තුළට දැමිය යුතුයි එම string එක. එනම් පහත පරිදියි.



```
it translations for all locales in the translations editor.
<resources>
    <string name="app_name">Ashan_hello world</string>
    <string name="action_settings">Settings</string>
    <string name="welcome">welcome to android world!</string>
</resources>
```

අපිට මෙහි ජේනවා welcome to android world! යන string එක string tag එකක් ඇතුළේ දාල තියෙනවා. එම string tag එකේ attribute එකක් තියෙනවා name කියල. එය සඳහා අපිට කැමති නමක් දුන්න හැකියි. එම නම භාවිතා කරලයි අපි මෙම welcome to android world! යන නම display වෙන්න කියලා කියන්නේ layout xml

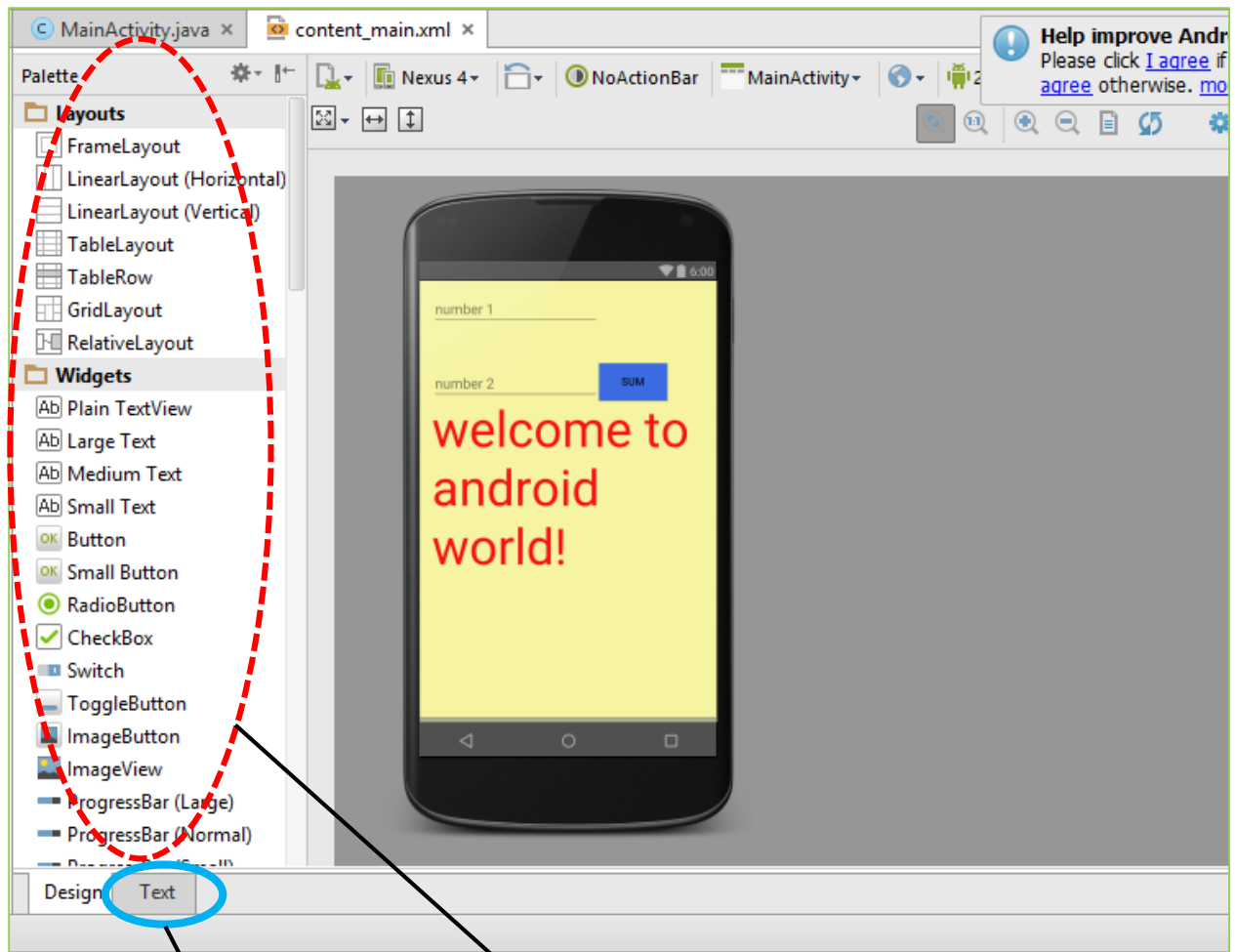
එකේදී තවද මෙම xml file එකේ root element එක resources වේ. so එයින් පැහැදිලියි මේ අපි කරන්නේ අපිට අවශ්‍ය වන string එකක් resource එකක් බවට පත් කිරීම බව. මෙවිට ඇති වාසිය නමා අපිට මෙම string එක තව එක එක ස්ථාන වලිදී අවශ්‍ය වුනොත් එම ස්ථානව වල එය හදන්නේ නැතුව මෙතනින් call කරලා ගත හැකි වීමයි. දැන් අපි බලමු ඉහත TextView එක තුලදී මෙම string resource එක call කරලා ගන්නේ කෙසේද කියලා.

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#00ff00"
        android:text="@string/welcome" />

</LinearLayout>
```

So දැන් ජේනවා මෙහිදී අපි @string/welcome ලෙසයි type කරන්නේ welcome to android world! යන්න display කිරීම සඳහා. මෙහි @string යන කොටසෙන් කියන්නේ string resources තියෙන xml file එකේ කියන්නයි. ඉන් පසුව welcome යනු අපිට අවශ්‍ය string resource එකට දීලා තියෙන නමයි. so මේ ආකාරයෙන් අපිට string එක resource එකක් බවට හරවා එය call කිරීමයි android වල preferred ක්‍රමය යම් texts display කිරීමේදී.



මෙම text යන button එක ඔබා අපිට xml codes වෙත යන්න පුලවන්. එනම් layout එකට අදාළ xml file එකටයි. එහෙම ගියහම අපිට බලාගන්න පුලවන් ඒ ඒ ui components වලට අදාළ xml codes ඉහතදී විස්තර කළ කාරණා අනුව සකස් වී ඇති බව

මේ තියෙන්නේ එක එක ui elements වේ. අපිට මේවා නිකන්ම ඇදලා දමන්න පුළුවන් layout එක උඩට. එවිට ඉබේම ඒවාට අදාළ xml tags type වෙනවා layout xml file එකේ.