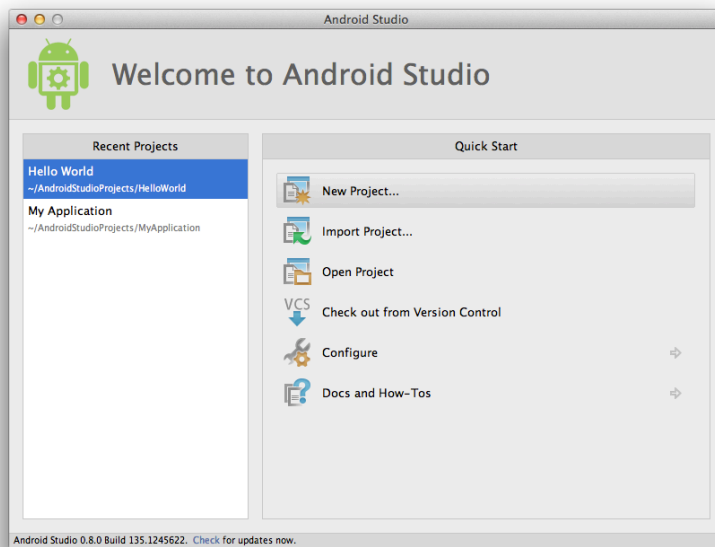


Laboration 1a

Starta Android Studio och skapa projektet HowTo.



Skapa sedan appen från föreläsningen genom att använda föreläsningsunderlaget.

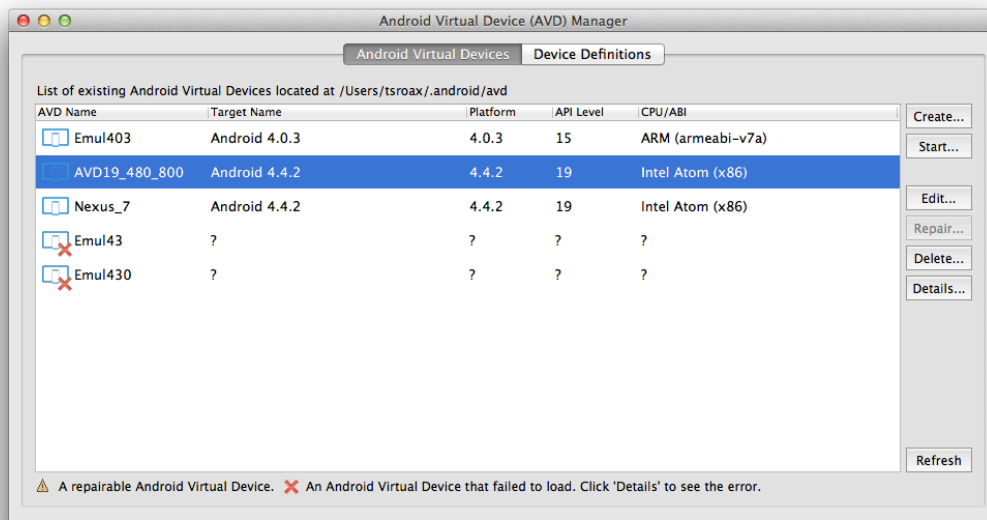
Company Domain bör vara username.mah.se och location bör vara en katalog du skapat för att spara Android-projekt.

The screenshot shows the "New Project" dialog box in Android Studio. It has four input fields. The first field is "Application name:" with the text "HowTo". The second field is "Company Domain:" with the text "tsroax.mah.se". The third field is "Package name:" with the text "se.mah.tsroax.howto" and a blue "Edit" button to its right. The fourth field is "Project location:" with the text "/Users/tsroax/AndroidStudioProjects/HowTo2" and a small square button with three dots to its right.

(Skapa och) starta en emulator om du inte redan startat en.



Klicka på (alternativt välj Tools – Android – AVD Manager). Markera sedan en emulator och klicka på Start...



Klicka sedan på Launch. Nu startas emulatoren, men det tar en god stund.

När Emulatoren startat så kan du stänga AVD Manager. För låset på mobilen till höger för att låsa upp mobilen (peka på låset, tryck ner vänster musknapp, håll knappen nertryckt och dra markören till höger till kant på mobilen och släpp sedan musknappen).

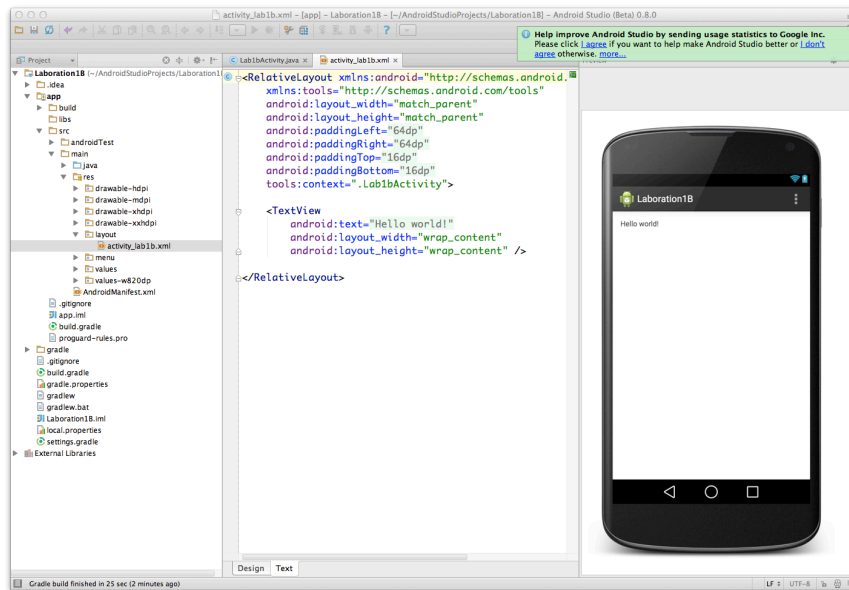
Nu kan du fortsätta jobba med HowTo-appen.

Laboration1b

Stäng det aktiva projektet (File – Close Project) och skapa ett nytt projekt, Laboration1B

Ange Activity name till *Lab1bActivity*.

Nu bygger Android Studio projektet vilket tar en liten stund. Ett fönster liknande det nedan kommer att visa sig.



Nu är det dags att exekvera den enkla app som projektet byggde automatiskt.

Klicka på runknappen och välj sedan emulator i Choose Device-dialogen. Klicka på OK. Strax ska appen visa sig i emulatorn.

Nu ska vi modifiera appen lite:

1. Ändra till ny bild högst upp till vänster i appen
2. Förse appen med både engelsk och svensk text
3. Lägga till ett par komponenter i appen.
4. Reagera på klick på knapp

Ändra till ny bild högst upp till vänster i appen

1. Kopiera l1b.png till katalogen src/main/res/drawable-hdp1
2. Öppna AndroidManifest.xml
3. Ändra android:icon till android:icon="@drawable/l1b"



Förse appen med både engelsk och svensk text

Katalogen ...res/values innehåller filen strings.xml.

```

<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Laboration1B</string>
    <string name="hello_world">Hello world!</string>
    <string name="action_settings">Settings</string>

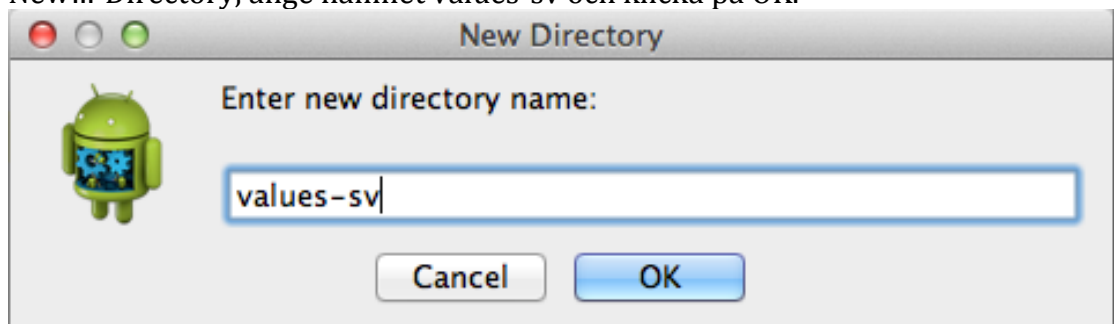
</resources>

```

Det är ur denna fil som strängar hämtas om inget annat anges. I filen är tre strängar definierade: app_name, hello_world och action_settings. Eftersom värdena är på engelska så är engelska det förvalda språket.

Nu ska vi lägga till motsvarande strängar på svenska.

1. Skapa katalogen values-sv i katalogen res genom att högerklicka res, välja New...-Directory, ange namnet values-sv och klicka på OK.



2. Kopiera filen strings.xml till den nya katalogen.
3. Öppna den nya strings.xml och ändra texten till svenska.

```

<string name="app_name">Laboration1B</string>
<string name="hello_world">Hej världen!</string>
<string name="action_settings">Inställningar</string>

```

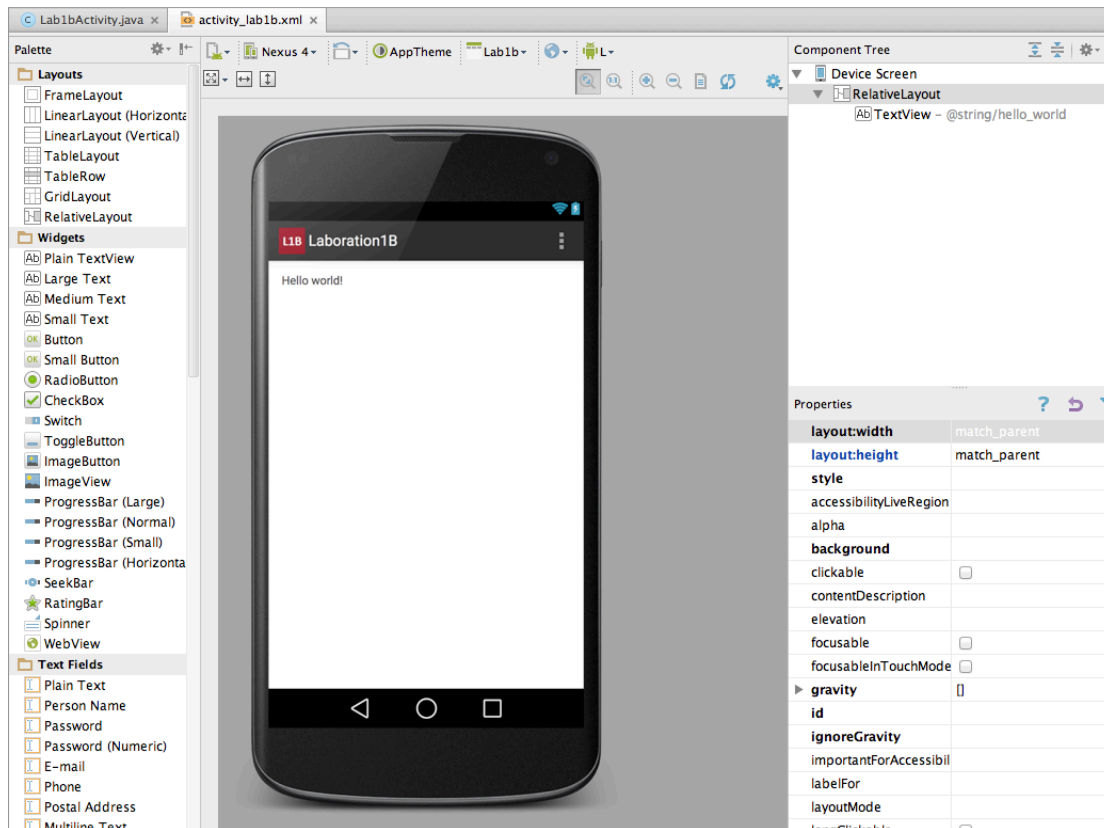
Spara ändringarna.

Om xml-filen för activityn är vald så kan du se de svenska texterna genom att klicka på jordgloben ovanför mobilen och välja Swedish.

När programmet exekveras i en mobil kommer de svenska texterna att visas om språkeställningen i mobilen är Svenska. Annars visas default-texten (strings.xml i value-katalogen).

Lägga till ett par komponenter i appen.

Se till att xml-filen för activityn är vald och välj fliken Design. Nu kommer du till design-läget i vilket du kan lägga till och designa komponenter i appen.



Till vänster är det ett stort antal komponenter du kan använda. I mitten ser du hur det kommer att se ut i mobilen och till höger kan du ändra komponenternas egenskaper.

Just nu har appen endast en komponent, en *TextView* med innehållet "Hello world!". Markera komponenten genom att klicka på den (du kan även markera den övers till höger under *Component Tree*). Till höger ser du olika egenskaper för komponenten. Gå nedåt i listan till egenskapen *text*. Egenskapen *text* anger vad som ska visas i *TextView*-komponenten. I det här fallet är det innehållet i strängen *hello_world* (finns i *strings.xml*) som visas. Det går att skriva in text direkt men om du gör det går det inte att internationalisera appen på ett enkelt sätt.

Gå till *values - strings.xml* och lägg till strängen *name* med innehållet "Name:".

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Laboration1B</string>
    <string name="hello_world">Hello world!</string>
    <string name="name">Name:</string>
    <string name="action_settings">Settings</string>

</resources>
```

Glöm inte att spara. (Om du använder ett flertal språkfiler ska du göra motsvarande tillägg i dem)

Gå tillbaka till Design-läget och ändra värdet efter *text* till *@string/name*.

text	@string/name
-------------	---------------------

Efter att du tryckt på Enter ändras texten i appen till Name:.

Nu kan du leta upp *textColor*-egenskapen och se till att texten blir röd. Hur gör man det undrar du kanske. Testa att klicka på ... till höger om *textColor*. Och använd Google och finn information om resursfil med namnet *colors.xml*.

Lägg till en fil *colors.xml* i *values*-katalogen (Högerklicka *values*-katalogen och välj *New-Values resource file*) och definiera färgerna *red* och *orange*.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <item name="red" type="color">#FF0000</item>
  <item name="orange" type="color">#FF6600</item>
</resources>
```

Ange sedan efter *textColor* att färgen *red* ska användas: *@color/red*. Nu ska textens färg ändras till röd.

En god vana är att ge komponenter ett tydligt id. I vart fall om du ska ändra värde eller egenskaper i dem under exekvering. Leta upp egenskapen *id* för *TextView*-komponenten och skriv in *tvName*.

Nu är det dags att placera några nya komponenter i appen.

EditText för inmatning av text

Markera *Plain Text* under *Text Fields*. Flytta musmarkören till mobilen och klicka till höger om Name:. Om den inte fick position / storlek som du önskade så ändra detta.

Name:

Glöm inte att ge komponenten ett id, t.ex. *etName*.

Lägg till ytterligare tre *TextView*-komponenter och två *EditText*-komponenter så att appen får ett utseende liknande figuren till höger. I den understa *TextView*-komponenten är det några bindestreck och färgen är *orange*. Glöm inte att ge komponenterna vettiga *id*. Och se till att *TextView*-komponenterna visar strängar ur *strings.xml*.

ImageView för att visa en bild

Markera *ImageView* under *Widgets* och placera en komponent i appen. Ge sedan egenskapen *background* ett värde ur mappen *drawable*, t.ex. *ic_launcher*. Passa på och ge *ImageView*-komponenten ett *id*, t.ex. *ivPhoto*.

Button för att reagera på klick

Markera *Button* och placera en komponent i appen. Ändra texten på knappen till "Summary".

Testkör din app innan du fortsätter. Som du märker går det bra att mata in text i *EditText*-komponenterna. Det händer dock inte något när du klickar på knappen.

Händelsehantering

Nu ska vi se till att personens namn, telefon och email visas i den understa *TextView*-komponenten vid klick.

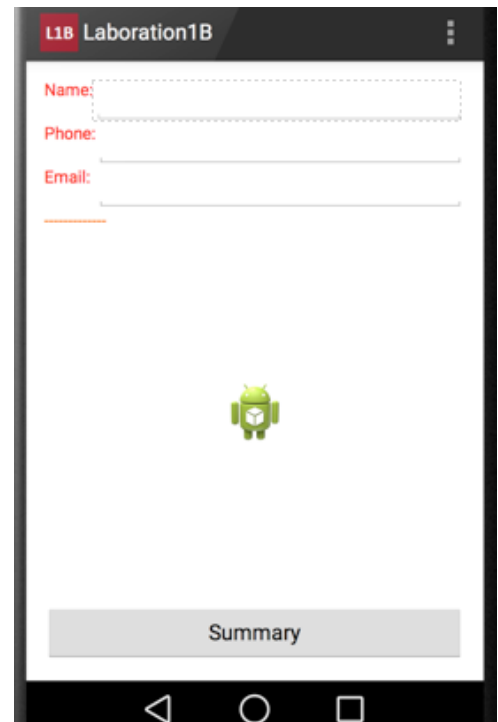
I klassen *Lab1bActivity* ska du

- Se till att du kan använda komponenterna i *Activity*-klassen (*Lab1Activity*)
- Koppla *Button*-komponenten till en lyssnarklass, en klass som innehåller kod som exekveras då användare klickar på knappen.
- Lägga till lyssnarklassen som en inre klass
- Lägga till kod vilken hämtar strängar ur *EditText*-komponenterna och skapar en sträng att visa och ser till att strängen visas i den understa *TextView*-komponenten.

Använda komponenter i Activity-klassen

För att använda komponenter i *Activity*-klassen så krävs det referenser till komponenterna i fråga. Och att lagra referenser som ska användas vid senare exekvering i instansvariabler. Referenser som vi behöver i appen är till *Button*-komponenten, till *EditText*-komponenterna och till den understa *TextView*-komponenten. *Button*-komponenten används endast för att registrera en lyssnare vid klick så den deklarerar lokalt. Övriga referenser deklarerar som instansvariabler.

```
public class Lab1bActivity extends Activity {  
    private EditText etName;  
    private EditText etPhone;  
    private EditText etEmail;  
    private TextView tvSummary;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_lab1b);  
        Button bnSummary;  
    }  
}
```



Med metoden *findViewById(int resource)* får man en referens till en *View*. Referensen måste typkonverteras till korrekt komponent-typ.

```
public class Lab1bActivity extends Activity {
    private EditText etName;
    private EditText etPhone;
    private EditText etEmail;
    private TextView tvSummary;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_lab1b);
        etName = (EditText)findViewById(R.id.etName);
        etPhone = (EditText)findViewById(R.id.etPhone);
        etEmail = (EditText)findViewById(R.id.etEmail);
        tvSummary = (TextView)findViewById(R.id.tvSummary);
        Button bnSummary = (Button)findViewById(R.id.bnSummary);
    }
}
```

Koppla Button-komponenten till en lyssnarklass

Med metoden *setOnClickListener(ref till lyssnarklass)* anger man att kod i en speciell klass ska anropas då användaren klickar på knappen. Lyssnarklassen ska implementera interfacet *View.OnClickListener*. Att klassen implementerar detta interface innebär att metoden *onClick(View)* måste finnas i klassen.

Lägg till satsen:

```
bnSummary.setOnClickListener(new ButtonListener() );
```

på raden under det att du hämtat en referens till knappen.

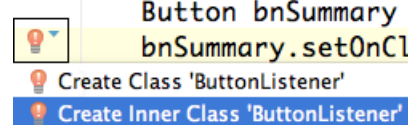
Lägga till lyssnarklass som inre klass

ButtonListener kommer att rödfärgas. Det innebär att något är fel. Felet här är att klassen inte finns. Klicka på texten *ButtonListener*. Nu visar sig en markering till vänster.

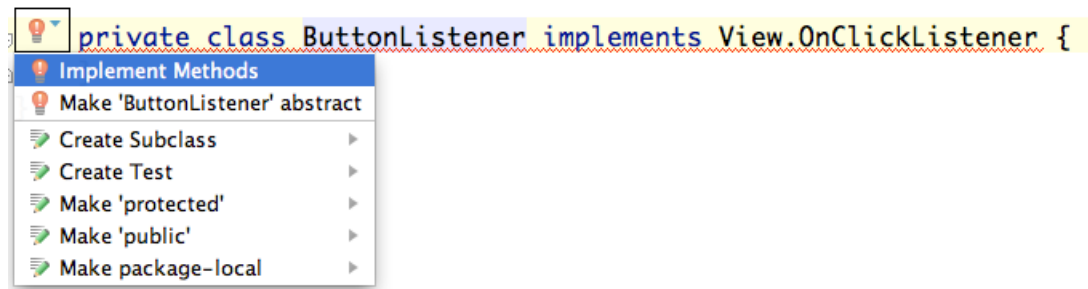
```
Button bnSummary = (Button)findViewById(R.id.bnSummary);
bnSummary.setOnClickListener(new ButtonListener());
```

Klicka på markeringen och välj *Create Inner Class ButtonListener*.

```
Button bnSummary = (Button)findViewById(R.id.bnSummary);
bnSummary.setOnClickListener(new ButtonListener());
```



Nu skapas klassen sist i klassen *Lab1bActivity*. Rödmarkeringen försvinner. Klassen *ButtonListener* är dock inte färdig. Den måste innehålla metoden *onClick*. Klicka någonstans på raden där den nya klassen deklarerats, t.ex. på texten *ButtonListener*. Nu får åter en markering till vänster. Klicka på markeringen och välj *Implement Methods*.



Se till att *onClick* är markerad i nästa dialog och klicka på *OK*. Nu är skalet till klassen *ButtonListener* färdig och det återstår att skriva lite instruktioner.

```
private class ButtonListener implements View.OnClickListener {  
    @Override  
    public void onClick(View v) {  
    }  
}
```

Nedan är kod som placerar det användaren matat in i *EditText*-komponenterna i en *TextView*. Metoden *getText* returnerar en *Editable*-instans vilken man gör om till en sträng med *toString*-metoden.

```
private class ButtonListener implements View.OnClickListener {  
    @Override  
    public void onClick(View v) {  
        String str = summary();  
        tvSummary.setText(str);  
    }  
  
    private String summary() {  
        String res;  
        res = "Name=" + etName.getText().toString() + ", phone=" +  
            etPhone.getText().toString() + ", email=" +  
            etEmail.getText().toString();  
        return res;  
    }  
}
```

Nu är det åter dags att testköra programmet. Mata in ditt namn, telefonnummer och email och klick sedan på Summary-knappen. Körresulttet bör likna det nedan.

Name: Rolf

Phone: 040-6657681

Email: rolf.axelsson@mah.se

Name=Rolf, phone=040-6657681, email=rolf.axelsson@mah.se

Laboration 1c

Nu ska vi lägga till ytterligare en knapp vilken ska ändra färg på "Name:" vid klick. Vartannat klick ska ändra färgen till röd färg och vartannat till orange färg.

Placera en knapp någonstans i fönstret. Ge knappen ett bra id, t.ex. bnColor.

I klassen *Lab1bActivity* ska du lägga till en inre klass vilken ska innehålla kod för att hantera klick på knappen.

```
private class ChangeColor implements View.OnClickListener {
    private boolean redColor = true;
    @Override
    public void onClick(View v) {
        if (redColor) {
            tvName.setTextColor(0xFFFF6600);
        } else {
            tvName.setTextColor(Color.RED);
        }
        redColor = !redColor;
    }
}
```

När användaren klickar på knappen så är det meningen att metoden `onClick(View v)` ska anropas. Klassen `ChangeColor` innehåller en instansvariabel, `redColor`, av typen `boolean`. När användaren klickar på knappen så avläses värdet på `redColor`. Om `true` (texten är röd) så ändras textfärgen till orange

```
tvName.setTextColor(0xFFFF6600);
```

och annars (texten är orange) så ändras textfärgen till röd

```
tvName.setTextColor(Color.RED);
```

I det första fallet anges färgvärdet hexadecimalt. Första två FF är alfavärdet (transparens), följande FF är rödvärdet, 66 är grönvärdet och 00 är blåvärdet. I det andra fallet används färgkonstanten `RED` i klassen `Color`. Klassen `Color` innehåller konstanterna `BLACK`, `BLUE`, `CYAN`, `DKGRAY`, `GRAY`, `GREEN`, `LTGRAY`, `MAGENTA`, `RED`, `WHITE` och `YELLOW`.

Det finns alternativ till ovanstående skrivsätt. Att byta färgen till orange kan t.ex. göras så här:

```
tvName.setTextColor(Color.rgb(255,102,0));
tvName.setTextColor(getResources().getColor(R.color.orange));
```

Den sista varianten använder ett färgvärde i filen *colors.xml*.

Se till att koppla lyssnaren till färg-knappen på samma sätt som en instans av `ButtonListener` kopplades till `bnSummary`.

Laboration 1d

Modifera metoden onCreate i klassen Lab1bActivity så den ser ut så här.

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_lab1b);  
    initializeComponents();  
    registerListeners();  
}
```

Då du kör appen ska körresultatet vara samma som tidigare.

Laboration 1e

Skapa projektet *Laboration1E*. Kalla Activity-klassen för *Lab1eActivity*.

Nu ska du bygga samma app som i Laboration 1b men med hjälp av *LinearLayout*.

Samtidigt kommer du få göra ändringar direkt i activity_lab1e.xml.

Ändra RelativeLayout i xml-filen till LinearLayout.

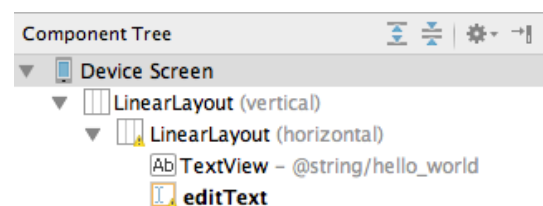
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/  
    xmlns:tools="http://schemas.android.com/tools"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:paddingLeft="@dimen/activity_horizontal_margin"  
    android:paddingRight="@dimen/activity_horizontal_margin"  
    android:paddingTop="@dimen/activity_vertical_margin"  
    android:paddingBottom="@dimen/activity_vertical_margin"  
    android:orientation="vertical"  
    tools:context=".Lab1eActivity">  
  
    <TextView  
        android:text="Hello world!"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content" />  
  
</LinearLayout>
```

Ändra till Design-läge och placera in i layouten ovan:

- En EditText-komponent
- En LinearLayout (horizontal).

Flytta sedan komponenterna i Component Tree så trädet ser ut som figuren till höger

Om du tittar på mobilen så är EditText-komponenten snyggt placerad till höger om Textview-komponenten.



Ändra till Text-läge, markera den inre LinearLayouten tillsammans med komponenterna.

Kopiera den markerade texten och klistra in två gånger direkt under den inre LinearLayouten.

På det viset kommer det bli tre rader med text + inmatningsruta.

Eftersom editText-komponenterna har samma id får du ett par rödmarkeringar. Det är dags att ge samtliga komponenter vettiga id!

Figuren nedan visar hur mobilen och Component Tree ser ut efter kopieringarna.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:orientation="vertical"
    tools:context=".MainActivity">

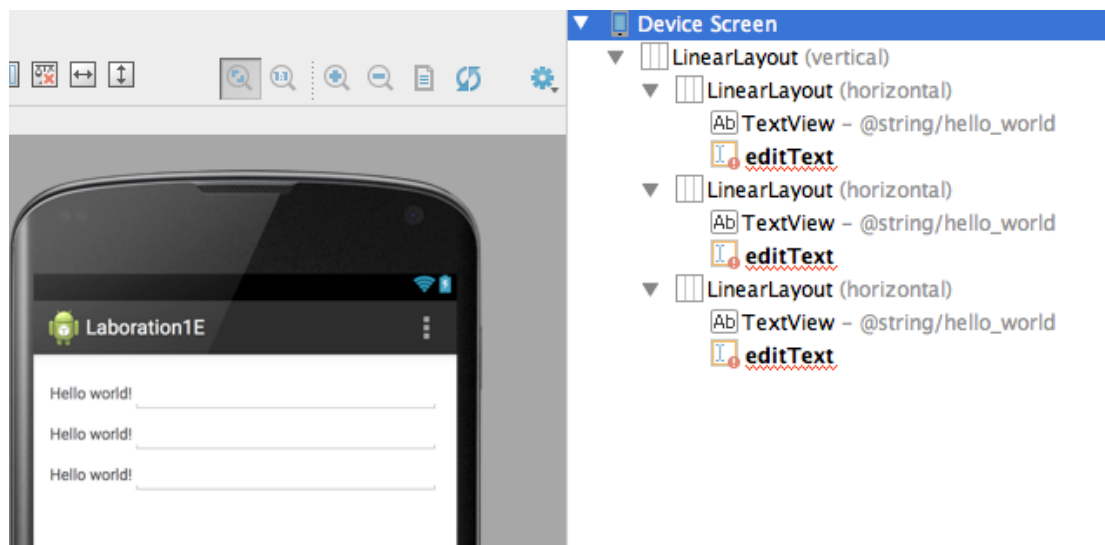
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView
            android:text="@string/hello_world"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />

        <EditText
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@id/editText" />

    </LinearLayout>

    <LinearLayout>
</LinearLayout>
```



Lägg märke till att TextView-texten och EditText-komponenten bredvid har samma höjd.

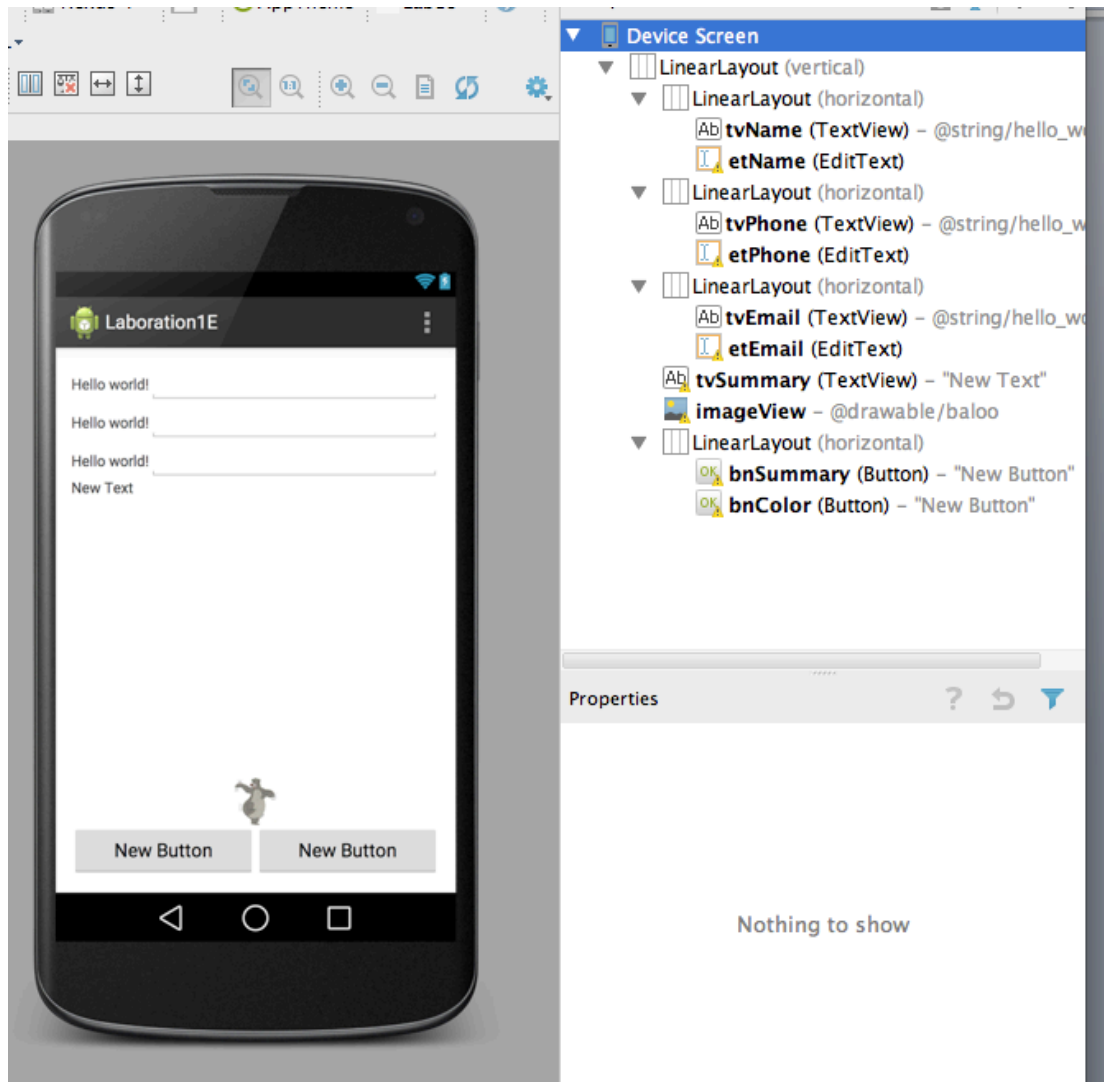
Det ska vara några komponenter till i appen. Lägg till:

- En TextView, ändra id till tvSummary. Låt layout:width ha värdet *match_parent*
- En ImageView, kopiera in en bild (t.ex. baloo.png) och ge src värdet till bilden
- En LinearLayout (horizontal) och placera två knappar i layouten. Ge knapparna vettiga id, layout:weight värdet 1 och layout:width värdet 0dp

Om man vill att bilden och knapparna ska vara längst ner i mobilen kan man sätta Layout:weight till 1 i tvSummary och samtidigt ändra layout:height till 0dp.

Efter ovanstående beskrivning kan mobilen och dess Component Tree se ut som figuren på nästa sida.

Nu återstår det att skapa strings.xml, colors.xml, se till att textviews och buttons får vettigt innehåll och att det händer kul saker när användaren klickar på knappar.



Laboration 1f

Undersök på egen hand hur GridLayout fungerar. Försök att använda GridLayout vid utplacering av de tre översta raderna med komponenter. Om du lyckas så kommer samtliga TextView-komponenter bli lika breda och dessutom kommer TextView-komponent och EditText-komponent vara på samma höjd (som i laboration 1e)