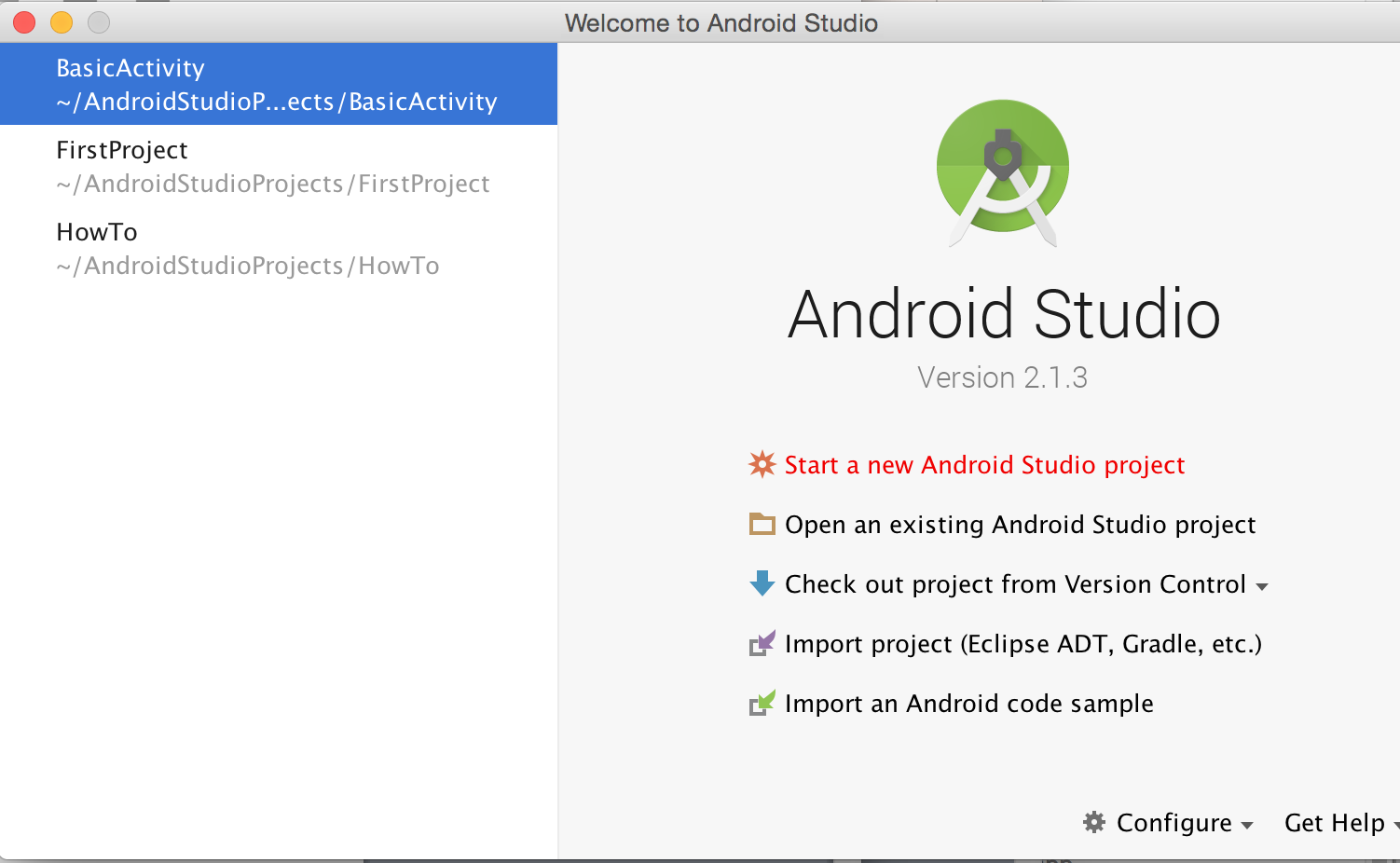
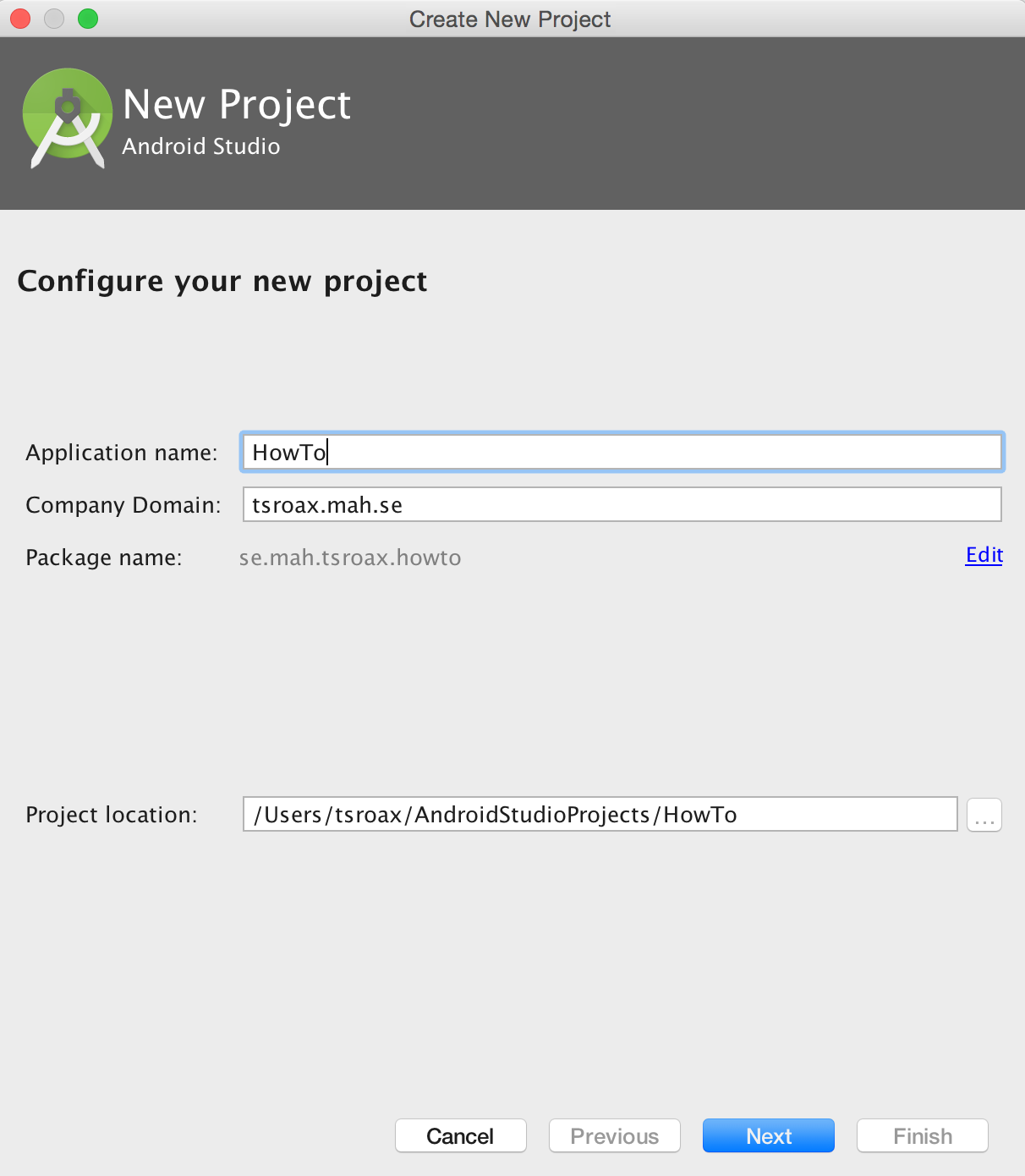
**Laboration 1a**

Starta Android Studio och skapa projektet HowTo.



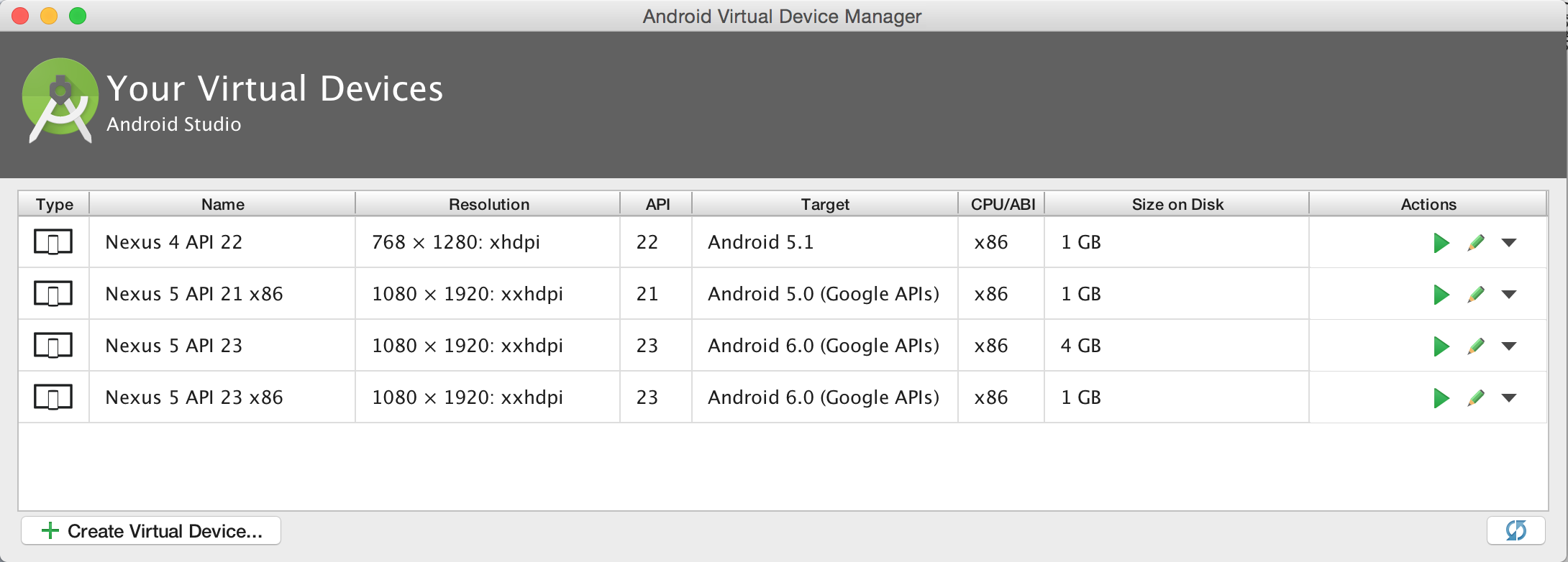
Skapa sedan appen från föreläsningen genom att använda föreläsningsunderlaget.

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



(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 emulatorn, men det tar en god stund.

När Emulatorn 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).

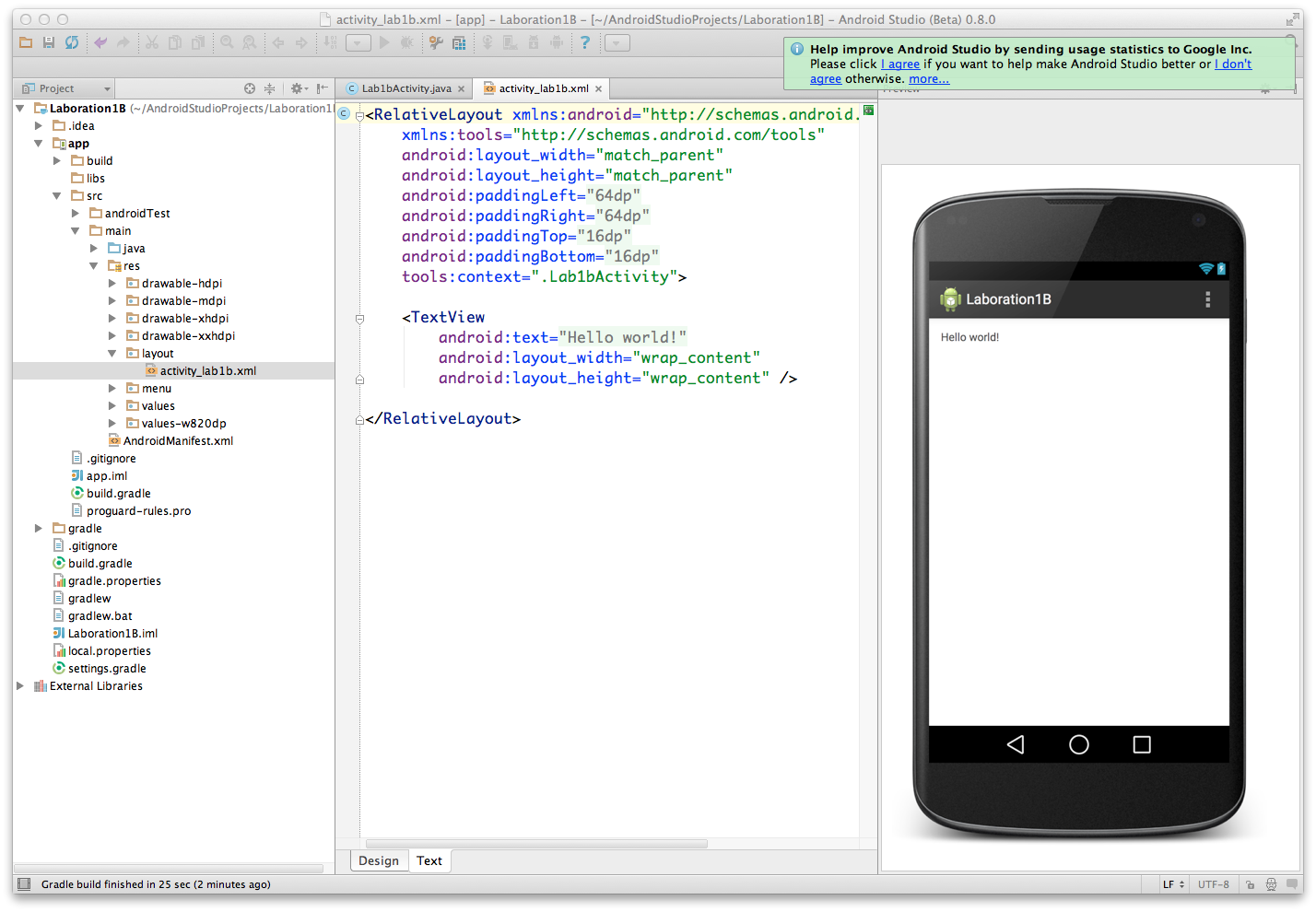
Nu kan du fortsätta jobba med HowTo-appen.

**Laboration1b**

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

Ange Activty name till *Lab1bActivity*.

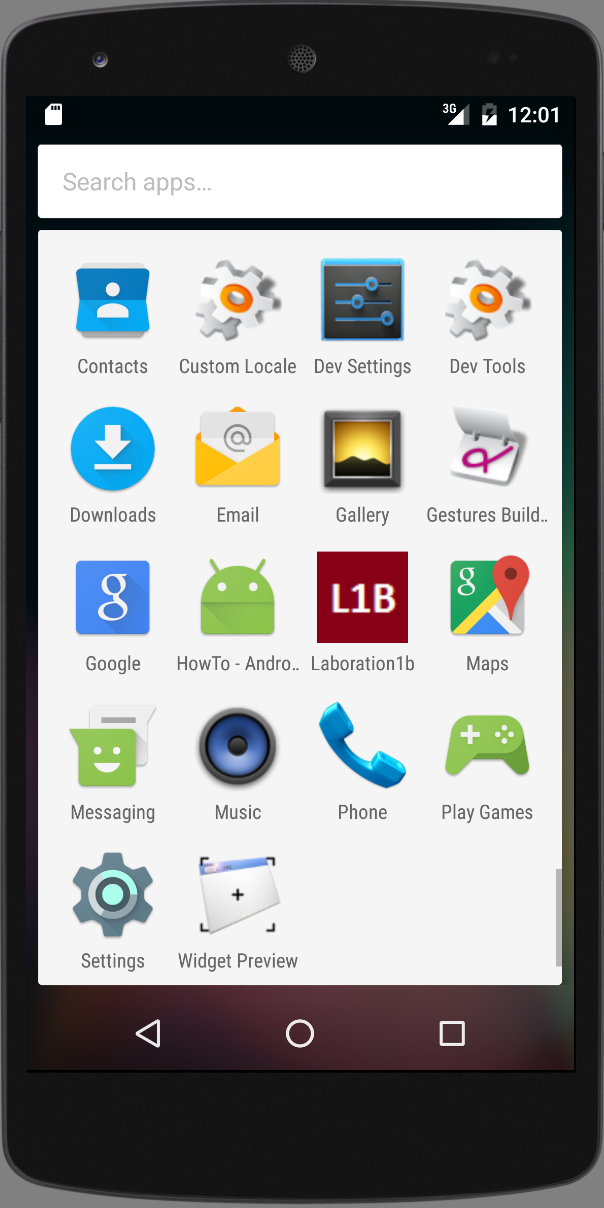
Nu bygger Android Studio projektet vilket tar en liten stund. Ett fönster liknande det nedan kommer att visa sig.



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 icon för appen
2. Förse appen med både engelsk och svensk text
3. Lägga till ett par komponenter i appen.
4. Reagera på klick på knapp

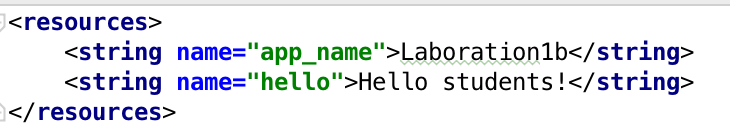
**Ändra till ny icon för appen**

1. Kopiera l1b.png till katalogen src/main/res/mipmap
2. Öppna AndroidManifest.xml
3. Ändra android:icon till android:icon=”@mipmap/l1b”

Denna bild bör egentligen finnas för olika upplösningar.

**Förse appen med både engelsk och svensk text**

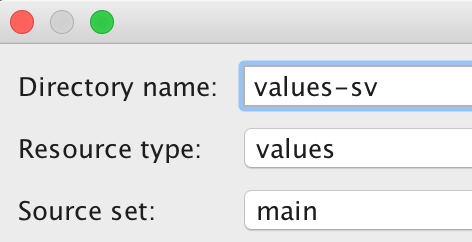
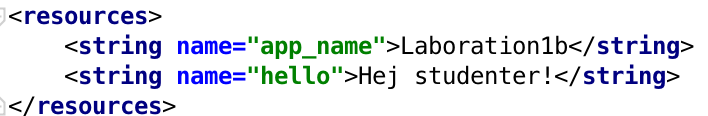
Katalogen …res/values innehåller filen strings.xml. Lägg till en sträng i filen:



Det är ur denna fil som strängar hämtas om inget annat anges. I filen är två strängar definierade: app\_nameoch hello. Eftersom värdena är på engelska (eg hello) så är engelska det förvalda språket.

Se till att strängen hello visas i TextView-komponenten, dvs ändra värde i Text-attributet.

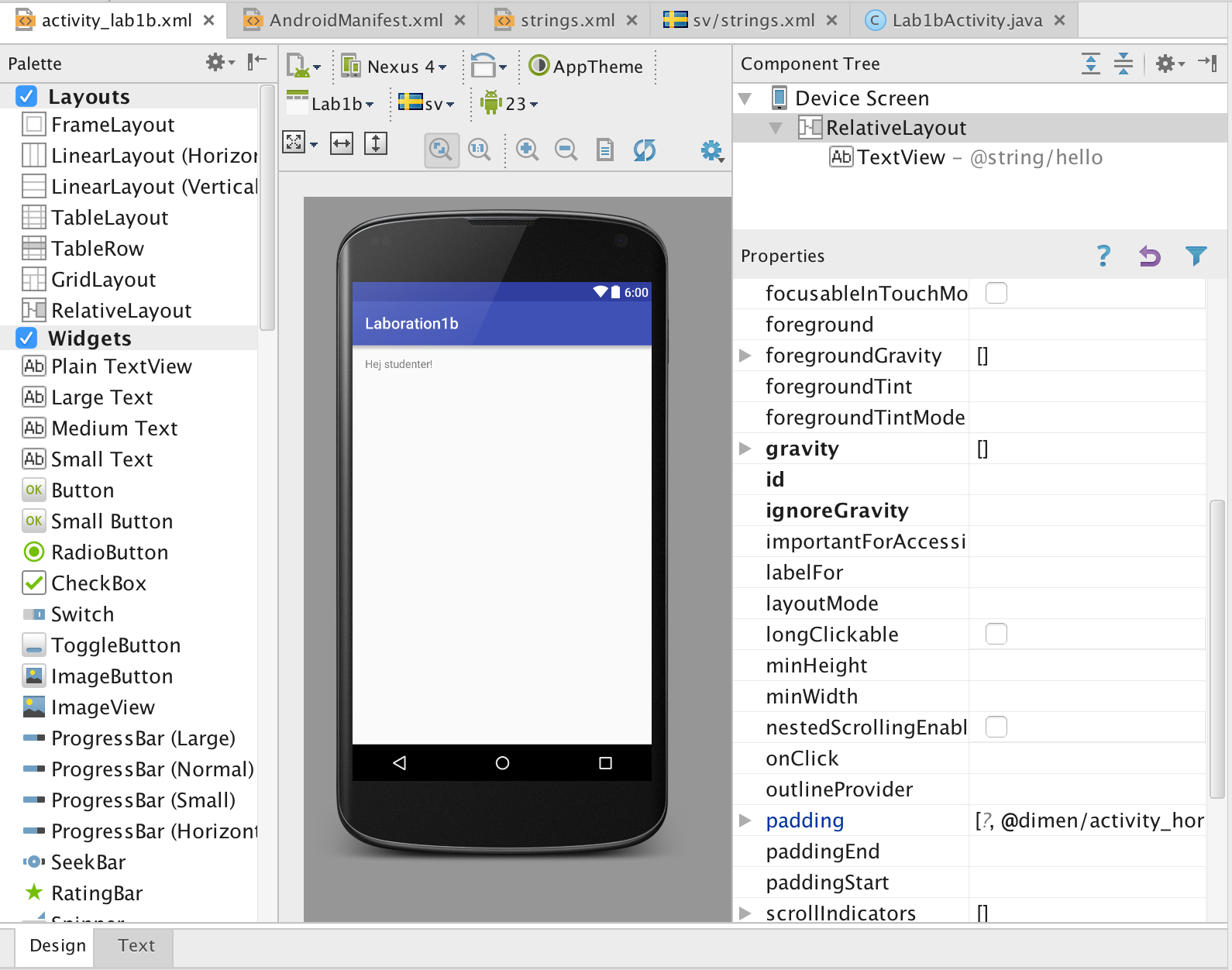
Nu ska vi lägga till motsvarande strängar på svenska.

1. Skapa katalogen values-sv i katalogen res genom att högerklicka res, välja New-Android Resource Directory, ange namnet values-sv och klicka på OK.
2. Markera strings.xml, kopiera, högerklicka res och klistra in. Klicka på boxen med tre prickar och välj values-sv.
3. Öppna den nya strings.xml och ändra texten till svenska. Spara ändringarna.

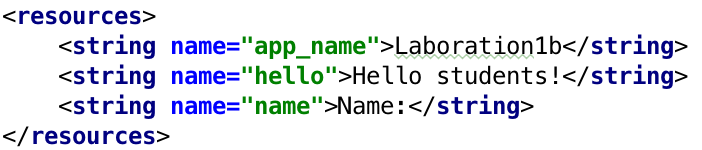
Om xml-filen för activityn är vald så kan du se de svenska texterna genom att klicka på jordgloben ovanför mobilen och välja Swedish.   
När programmet exekveras i en mobil kommer de svenska texterna att visas om språkinställningen i mobilen är Svenska. Annars visas default-texten (strings.xml i value-katalogen).

**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 students!”. Eftersom egenskapen text refererar till en sträng definierad i strings.xml så går det att internationalisera appen på ett enkelt sätt.

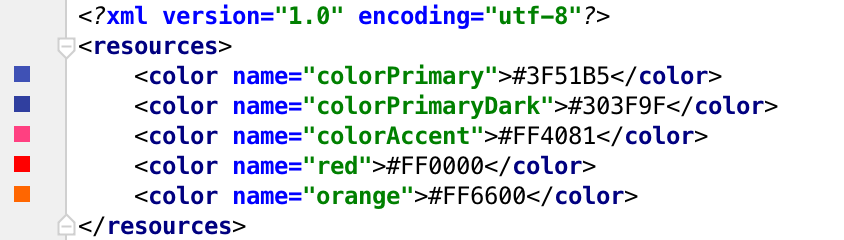
Gå till values - strings.xml och lägg till strängen *name* med innehållet ”Name:”. 

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.

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

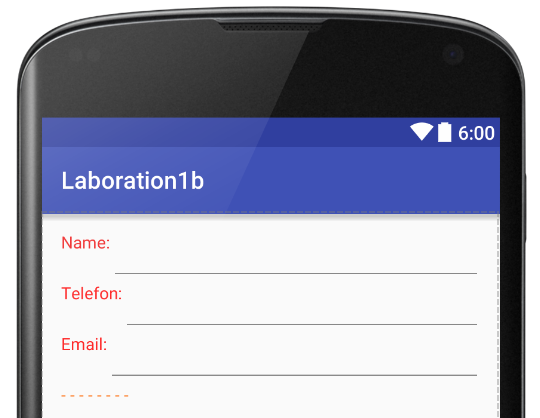
Nu kan du leta upp *textColor*-egenskapen och se till att texten blir blå. Hur gör man det undrar du kanske. Testa att klicka på … till höger om *textColor*. Och välj en färg. Färgen skrivs direkt in i egenskapen.

Öppna filen *colors.xml* i values-mappen och definiera färgerna *red* och *orange*.   
  
  
Ange sedan efter *textColor* att färgen *red* ska användas: *@color/red*. Nu ska textens färg ändras till röd.

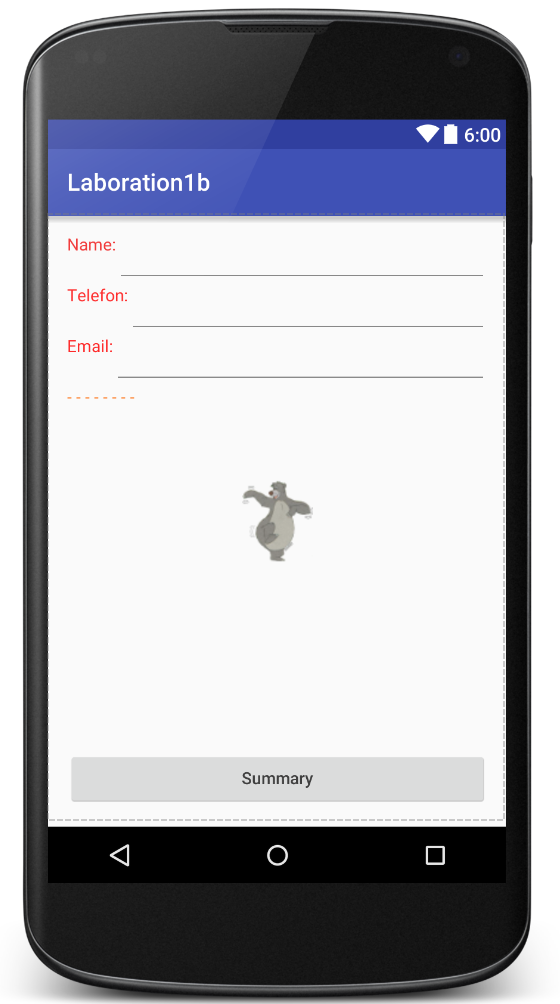
En god vana är att ge komponenter ett tydligt id. I vart fall om du ska ändra värde eller egenskaper i dem under exekvering. Leta upp egenskapen *id* för *TextView*-komponenten och skriv in *tvName*.

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.  
  
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**

Kopiera *baloo.png* till drawable-katalogen. Markera *ImageView* under *Widgets* och dra en komponent till appen. Ge sedan egenskapen *background* ett värde ur mappen *drawable* – t.ex. *baloo*. 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”. Googla för att ta reda på hur en knapp kan visa små bokstäver.

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**

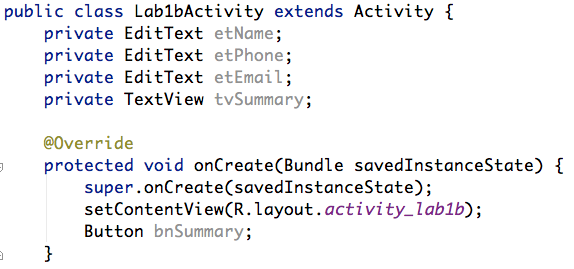
Nu ska vi se till att personens namn, telefon och email visas i den understa TextView-komponenten vid klick.

I klassen *Lab1bActivity* ska du

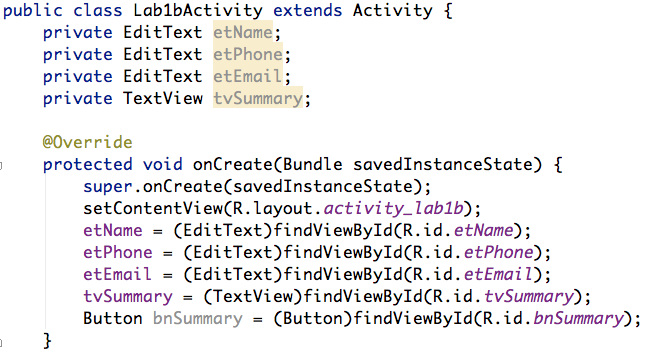
* Se till att du kan använda komponenterna i *Activity*-klassen (*Lab1Activity*)
* Koppla Button-komponenten till en lyssnarklass, en klass som innehåller kod som exekveras då användare klickar på knappen.
* Lägga till lyssnarklassen som en inre klass
* Lägga till kod vilken hämtar strängar ur EditText-komponenterna 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 deklareras lokalt. Övriga referenser deklareras som instansvariabler.



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



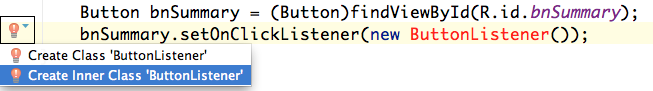
**Koppla Button-komponenten till en lyssnarklass**

Med metoden se *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.

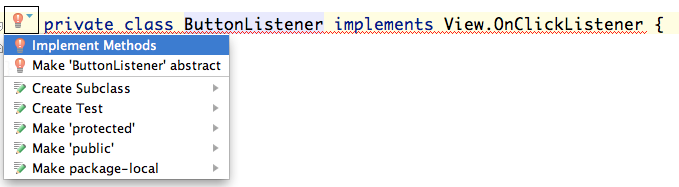
Lägg till satsen:  
 bnSummary.setOnClickListener(new ButtonListener() );  
på raden under det att du hämtat en referens till knappen.

**Lägga till lyssnarklass som inre klass**

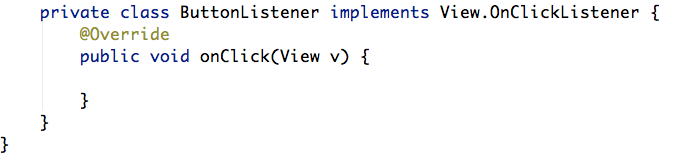
ButtonListener kommer att rödfärgas. Det innebär att något är fel. Felet här är att klassen inte finns. Klicka på texten ButtonListener. Nu visar sig en markering till vänster.  


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


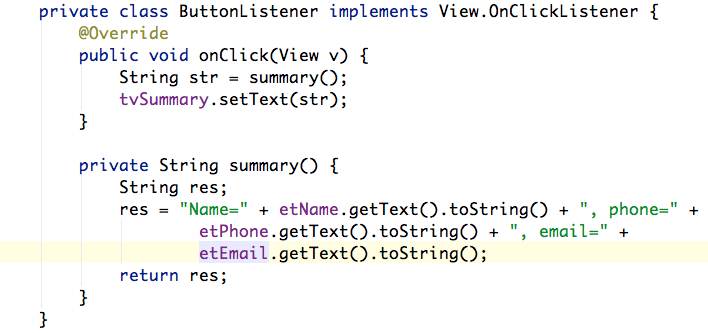
Nu skapas klassen sist i klassen Lab1bActivity. Rödmarkeringen försvinner. Klassen ButtonListener är dock inte färdig. Den måste innehålla metoden *onClick*. Klicka någonstans på raden där den nya klassen deklareras, t.ex. på texten ButtonListener. Nu får åter en markering till vänster. Klicka på markeringen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



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.



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.



**Laboration 1c**

Nu ska du 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.   
  
  
  
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);

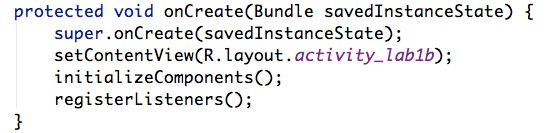
I det första fallet anges färgvärdet hexadecimalt. Första två FF är alfavärdet (transparens), följande FF är rödvärdet, 66 är grönvärdet och 00 är blåvärdet. I det andra fallet används färgkonstanten RED i klassen Color. Klassen Color innehåller konstanterna BLACK, BLUE, CYAN, DKGRAY, GRAY, GREEN, LTGRAY,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

Modifiera metoden onCreate i klassen Lab1bAtivity så den ser ut så här.



Då du kör appen ska körresultatet vara samma som tidigare.

**Laboration 1e**

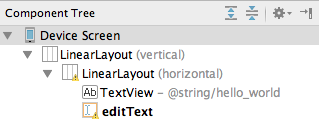
Skapa projektet *Laboration1E*. Kalla *Activity*-klassen för *Lab1eActivity*.

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.



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.

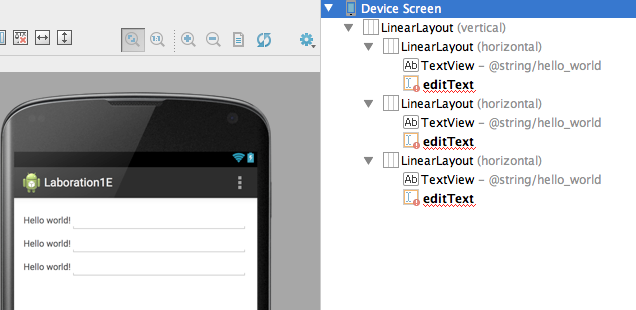
Ändra till Text-läge, markera den inre LinearLayouten tillsammans med komponenterna.

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.



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

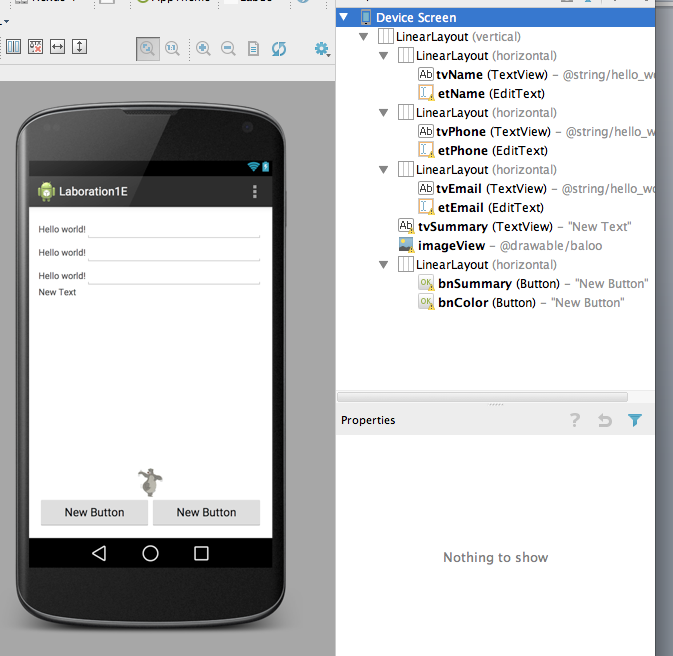
Det ska vara några komponenter till i appen. Lägg till:

* 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:weigth 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 använaren 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)