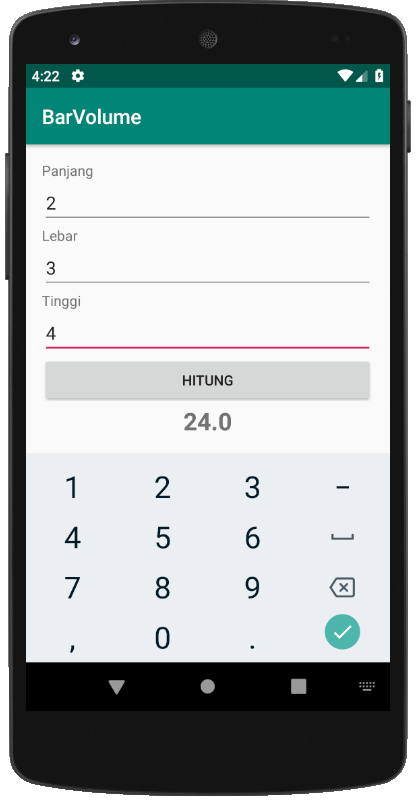
**PRAKTIK ACTIVITY**

**Tujuan**

Praktikum ini bertujuan untuk mengimplementasikan komponen activity pada aplikasi pertama yang Anda bangun. Harapannya aktifitas ini dapat memberi gambaran yang jelas tentang cara kerja activity.

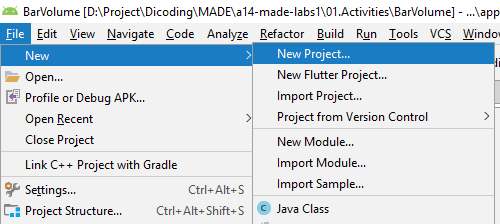
Praktikum pertama adalah dengan membuat aplikasi yang dapat menghitung volume balok. Seperti ini tampilannya.



**Logika Dasar**

Melakukan input ke dalam obyek EditText → melakukan validasi input → melakukan perhitungan volume balok ketika tombol hitung diklik.

**Praktikum Membuat Proyek Baru**

1. Buat proyek baru dengan klik **File → New → New Project**pada Android Studio Anda atau Anda bisa memilih **Start a new Android Studio project** di bagian **dashboard**.
2. Pada bagian ini kita akan memilih tipe activity awal dari *template* yang telah disediakan. Saat ini Android Studio sudah menyediakan berbagai macam *template* activity dari yang paling sederhana hingga yang paling kompleks seperti:

**Add No Activity** : Tidak ada activity yang ditambahkan  
**Basic Activity** : Activity dengan template komponen material design

seperti FloatingActionButton  
**Bottom Navigation Activity** : Activity dengan tampilan side bar menu di bagian

bawah  
**Empty Activity** : Activity dalam bentuk yang paling dasar  
**Fragment + ViewModel** : Activity dengan menerapkan *architecture component*  
**Fullscreen Activity** : Activity *fullscreen*tanpa status bar  
**Google AdMob Ads Activity** : Activity dengan konfigurasi *default*iklan Admob  
**Google Maps Activity** : Activity dengan menyediakan konfigurasi dasar

Google Maps  
**Login Activity** : Activity untuk halaman login  
**Master / Detail Flow** : Activity yang diperuntukan untuk alur aplikasi *master*

*detail* pada peranti tablet  
**Navigation Drawer Activity** : Activity dengan tampilan side bar menu  
**Scrolling Activity** : Activity dengan kemampuan scroll konten

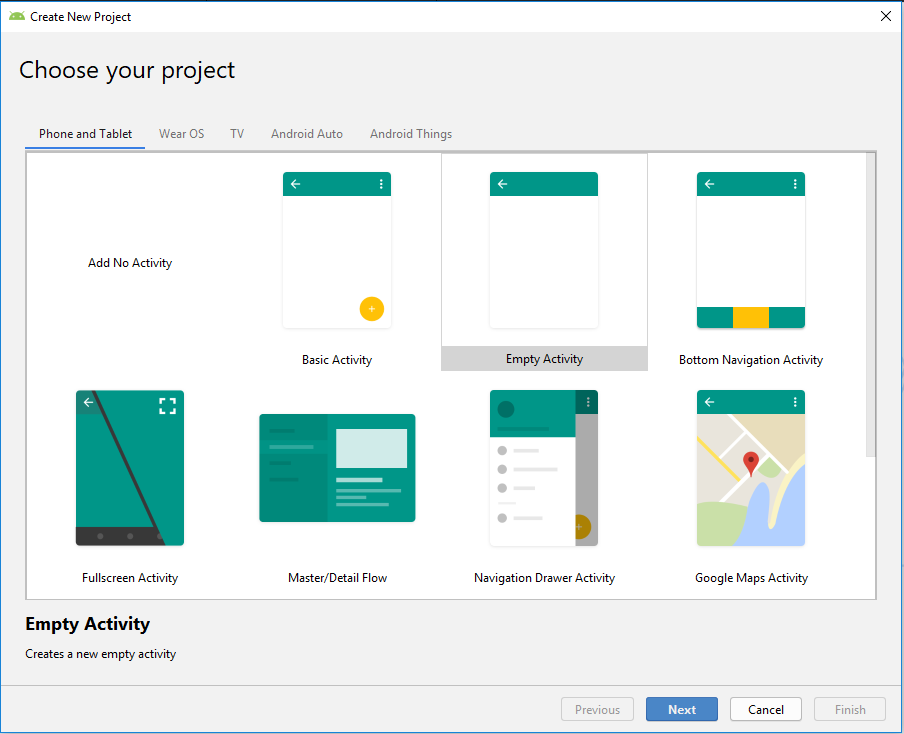
didalamnya secara vertikal  
**Settings Activity** : Activity yang diperuntukkan untuk konfigurasi

aplikasi  
**Tabbed Activity** : Activity yang diperuntukkan untuk menampilkan

lebih dari satu tampilan, dapat digeser ke kanan dan

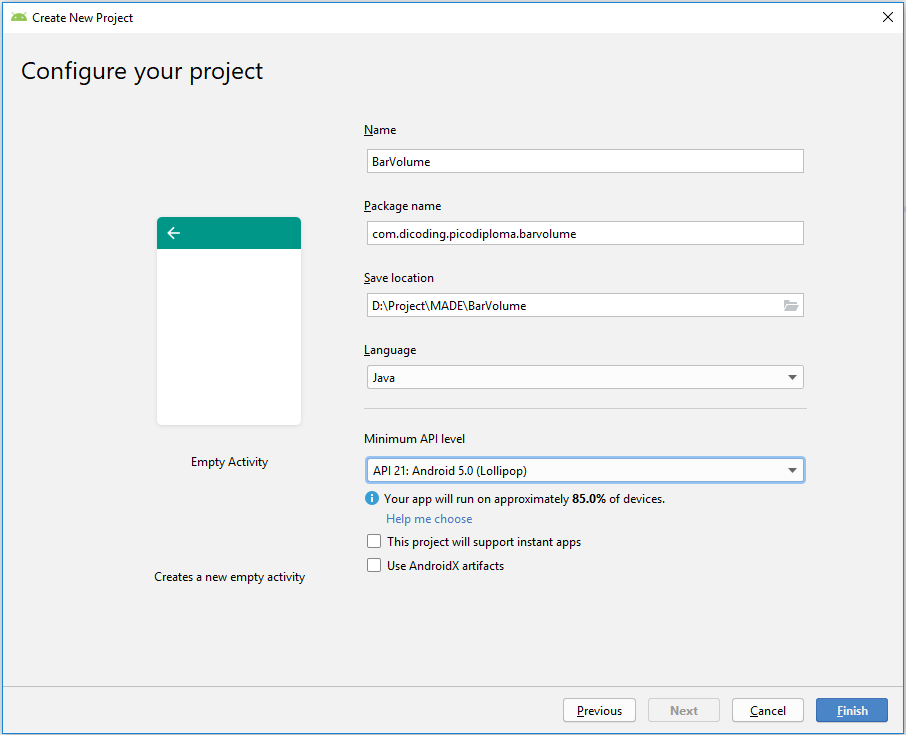
ke kiri (*swipe*) dan dengan menggunakan komponen

ViewPager  
  
Selain itu, Anda juga bisa memilih target device mana yang akan dibuat seperti**Phone and Tablet**, **Wear OS**, **TV**, **Android Auto** atau**Android Things**.

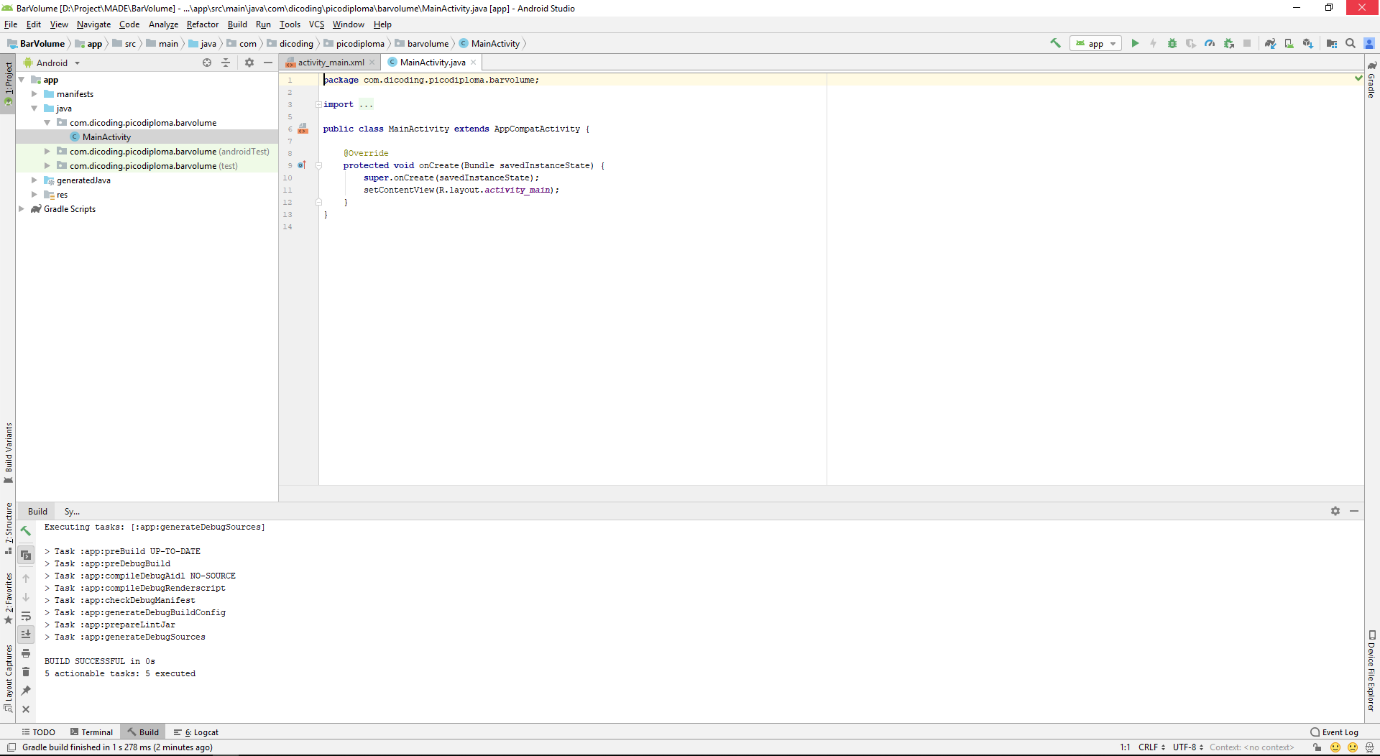


Saat ini kita pilih tipe **Empty Activity**, klik **Next**untuk melanjutkan.

1. Selanjutnya masukkan **nama aplikasi**dan nama *package*aplikasi Anda. Sebaiknya jangan sama dengan apa yang ada di contoh, karena ini berfungsi sebagai iddari aplikasi yang Anda buat. Kemudian Anda bisa menentukan lokasi proyek yang akan Anda buat. Setelah itu pilih tipe gawai/peranti (*device*) untuk aplikasi beserta target minimum SDK yang akan digunakan. Pilihan target Android SDK akan mempengaruhi banyaknya peranti yang dapat menggunakan aplikasi. Di sini kita memilih nilai minimum SDK kita pasang ke **Level 21 (Lollipop)**. Klik **Finish**untuk melanjutkan.



| **Catatan: Praktikum ini sudah mendukung bahasa Kotlin. Untuk menggunakan bahasa Kotlin ubahlah languages dari Java menjadi Kotlin, maka secara otomatis Android Studio akan menyesuikan dengan bahasa Kotlin.** |
| --- |

1. Tampilan layar Anda akan seperti contoh di bawah ini:  
   
2. Di sebelah kanan Anda adalah *workspace*di mana Activity anda berada dan bernama MainActivity dengan layoutnya activity\_main.xml. Di sebelah kiri Anda terdapat struktur proyek, di mana nanti kita akan banyak menambahkan berbagai komponen baru, *asset* dan *library*. Untuk lebih mengenal Android Studio lebih dalam silakan baca materi ini <https://developer.android.com/studio/intro/index.html>

Selanjutnya kita akan mulai melakukan pengkodean aplikasi atau lebih enaknya disebut ngoding.

Berikut flow umumnya.

1. **Coding Layout** untuk user interface aplikasi
2. **Coding Activity** untuk menambahkan logika aplikasi

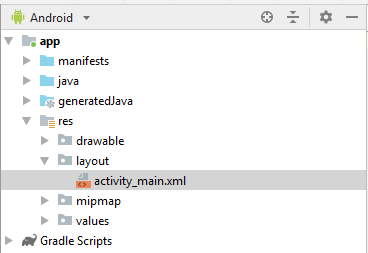
Untuk mengoptimalkan proses pengetikan, Anda dapat memanfaatkan *code completion* dengan menekan ctrl + space.Android Studio juga akan mengimpor *package*dari komponen yang digunakan.

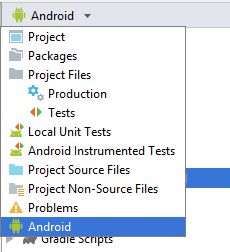
**Dilarang Keras untuk copy - paste!** Ngoding pelan-pelan akan membuat Anda lebih jago di masa depan.

**Selamat ngoding!**

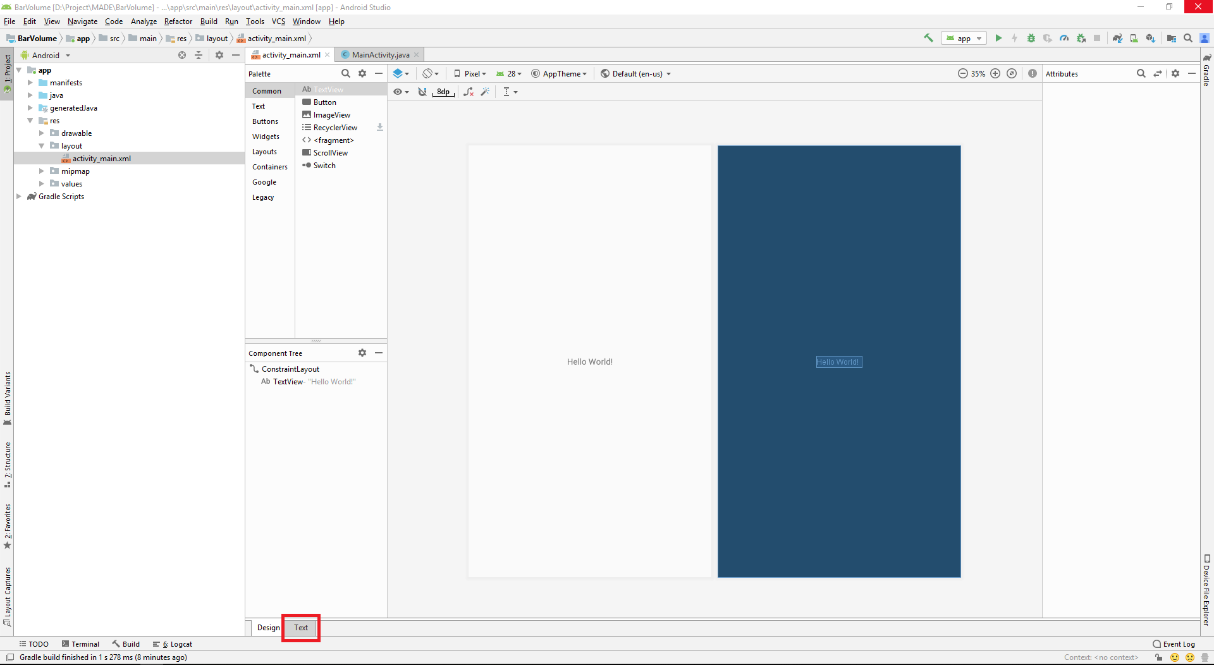
**Code Layouting**

**Menambahkan Code Sederhana pada Layout Activity**

1. Silakan pilih **tab**berkas activity\_main.xml pada *workspace*Anda(res/layout/activity\_main.xml).  
   

Pastikan **project window** pada pilihan **Android**, seperti di bawah ini.  


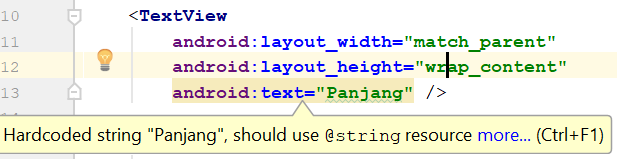
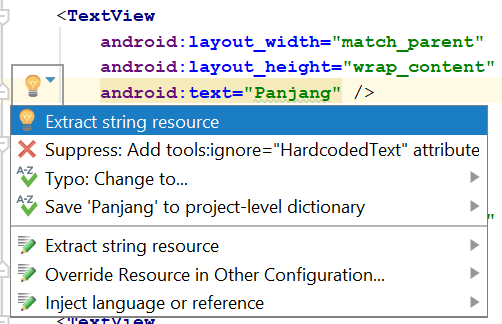
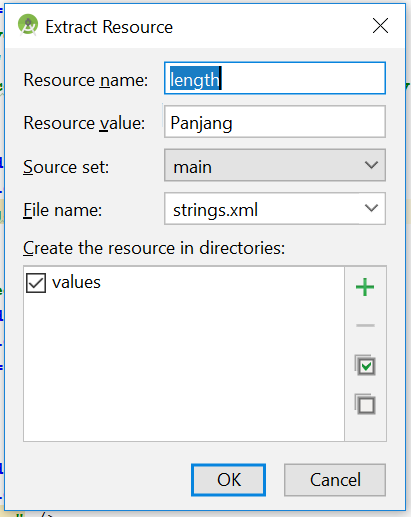
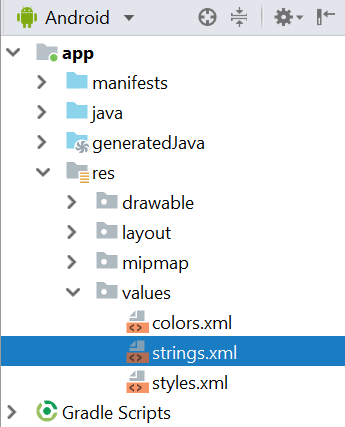
Maka akan ada tampilan seperti ini, kemudian pilih metode text.



Dan tambahkan baris-baris berikut:

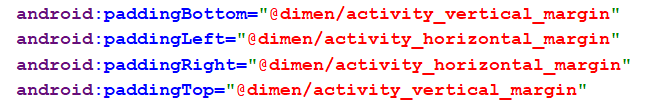
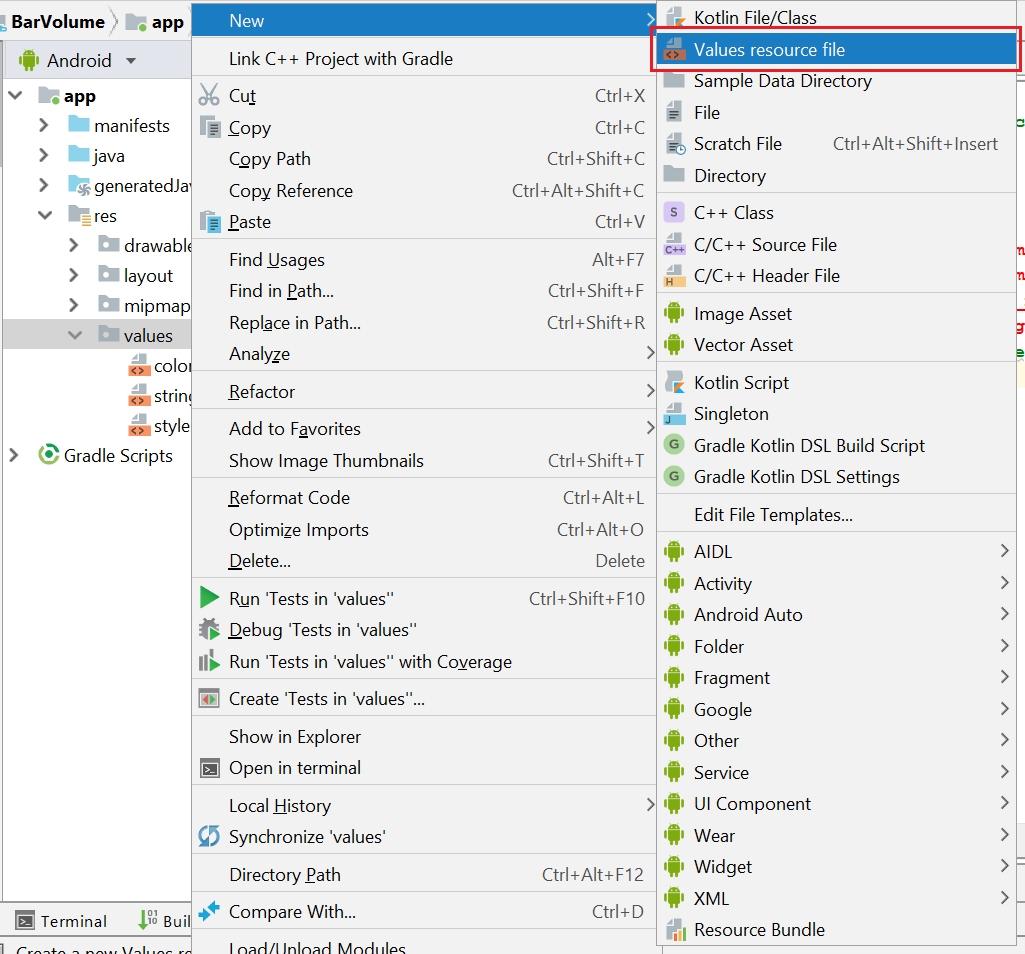
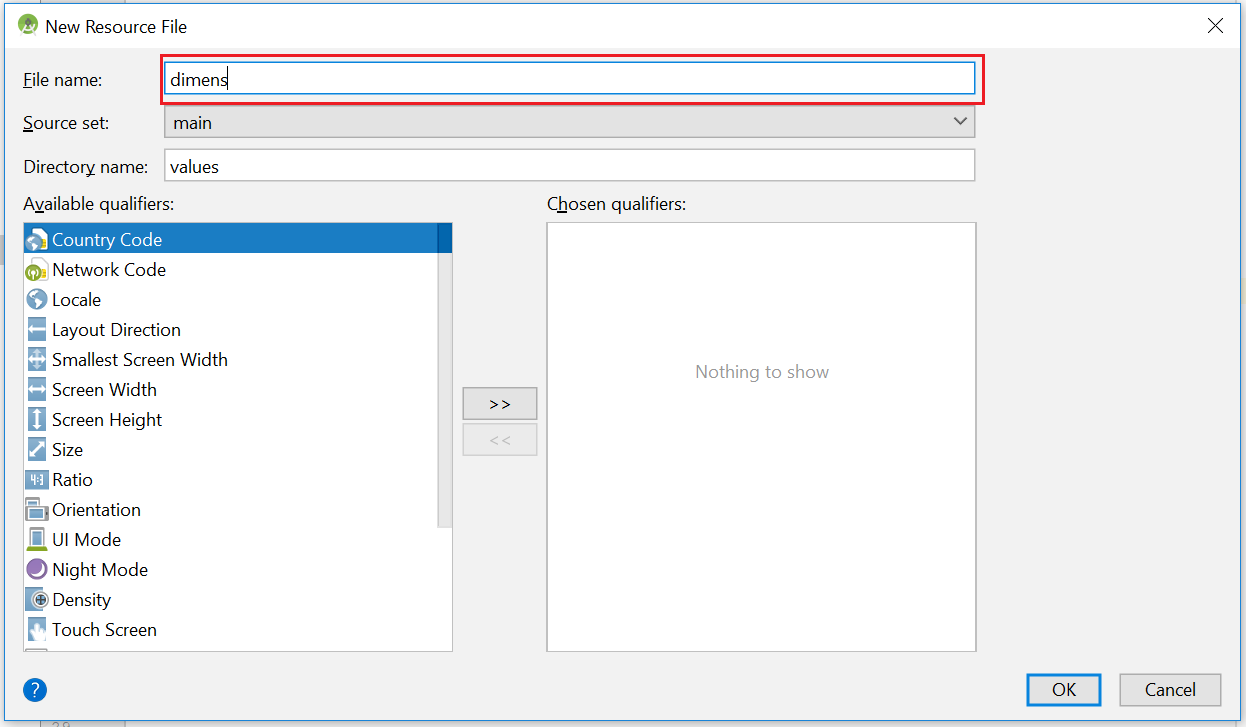
| 1. <?xml version="1.0" encoding="utf-8"?> 2. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android" 3. xmlns:tools="http://schemas.android.com/tools" 4. android:layout\_width="match\_parent" 5. android:layout\_height="match\_parent" 6. **android:paddingBottom="@dimen/activity\_vertical\_margin"** 7. **android:paddingLeft="@dimen/activity\_horizontal\_margin"** 8. **android:paddingRight="@dimen/activity\_horizontal\_margin"** 9. **android:paddingTop="@dimen/activity\_vertical\_margin"** 10. **android:orientation="vertical"**> 12. **<TextView** 13. **android:layout\_width="match\_parent"** 14. **android:layout\_height="wrap\_content"** 15. **android:text="Panjang" />** 16. **<EditText** 17. **android:id="@+id/edt\_length"** 18. **android:layout\_width="match\_parent"** 19. **android:layout\_height="wrap\_content"** 20. **android:inputType="numberDecimal"** 21. **android:lines="1" />** 22. **<TextView** 23. **android:layout\_width="match\_parent"** 24. **android:layout\_height="wrap\_content"** 25. **android:text="Lebar" />** 26. **<EditText** 27. **android:id="@+id/edt\_width"** 28. **android:layout\_width="match\_parent"** 29. **android:layout\_height="wrap\_content"** 30. **android:inputType="numberDecimal"** 31. **android:lines="1" />** 32. **<TextView** 33. **android:layout\_width="match\_parent"** 34. **android:layout\_height="wrap\_content"** 35. **android:text="Tinggi" />** 36. **<EditText** 37. **android:id="@+id/edt\_height"** 38. **android:layout\_width="match\_parent"** 39. **android:layout\_height="wrap\_content"** 40. **android:inputType="numberDecimal"** 41. **android:lines="1" />** 42. **<Button** 43. **android:id="@+id/btn\_calculate"** 44. **android:layout\_width="match\_parent"** 45. **android:layout\_height="wrap\_content"** 46. **android:text="Hitung" />** 47. **<TextView** 48. **android:id="@+id/tv\_result"** 49. **android:layout\_width="match\_parent"** 50. **android:layout\_height="wrap\_content"** 51. **android:gravity="center"** 52. **android:text="Hasil"** 53. **android:textSize="24sp"** 54. **android:textStyle="bold" />** 55. </LinearLayout> |
| --- |

Perlu diperhatikan root layout (**tag layout terluar**) yang dipakai di sini adalah LinearLayout. Jika kita menggunakan Android Studio versi 3 maka secara *default root* yang dipakai adalah ConstraintLayout. Agar sesuai dengan latihan ini, kita tinggal menggantinya menjadi LinearLayout.  
Untuk Layout akan dibahas nanti pada modul yang berbeda.

1. Kemudian akan muncul *warning* pada atribut android:text pada layout tersebut.   
   Ini karena kita melakukan *hardcoding* pada nilai *string*-nya. Mari kita hilangkan *code warning* tersebut dengan menekan **Alt+Enter (option + return**padaMac**)** atau menekan lampu kuning yang muncul pada attribut android:text.  
   Akan muncul dialog seperti ini, pilih **extract string resource***.*  
   
2. Kemudian akan muncul *dialog*seperti di bawah ini. Sesuaikan dengan nama yang ada.  
   
3. Fungsi **extract string resource**akan secara otomatis menambahkan nilai dari android:text ke dalam berkas **res → values → strings.xml**.   
   

Lakukan hal yang sama pada View lainnya hingga tidak ada *warning* lagi. Jika kita buka berkas strings.xml, maka isinya akan menjadi seperti ini:

| 1. <resources> 2. <string name="app\_name">BarVolume</string> 3. **<string name="width">Lebar</string>** 4. **<string name="height">Tinggi</string>** 5. **<string name="calculate">Hitung</string>** 6. **<string name="result">Hasil</string>** 7. **<string name="length">Panjang</string>** 8. </resources> |
| --- |

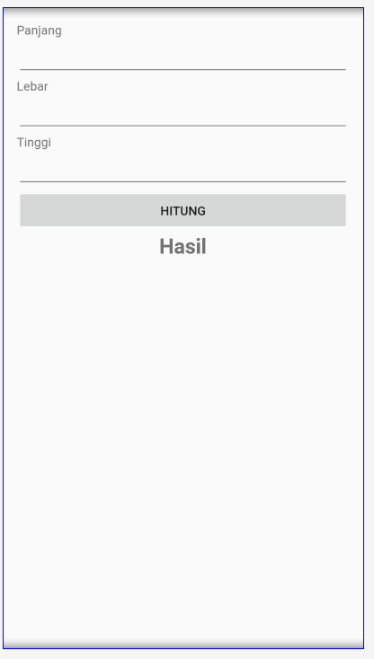
1. Jika terjadi kesalahan pada atribut dimens, maka kita perlu menambahkan berkas dimens.xml di dalam**res → values → dimens.xml**. Kesalahan ini disebabkan karena pada Android Studio 3, berkas dimens.xml sudah tak dibuat secara otomatis.  
   Langsung saja tambahkan dengan cara **klik kanan**pada directory res. Akan muncul dialog seperti ini.
2. Kemudian masukkan dengan nama *file* dimens, sesuai di bawah ini.  
   
3. Jika berkas dimens.xml sudah dibuat, sesuaikan isinya menjadi seperti berikut:

| 1. <?xml version="1.0" encoding="utf-8"?> 2. <resources> 3. **<dimen name="activity\_horizontal\_margin">16dp</dimen>** 4. **<dimen name="activity\_vertical\_margin">8dp</dimen>** 5. </resources> |
| --- |

1. Maka kode di dalam activity\_main.xml akan menjadi seperti ini:

| 1. <?xml version="1.0" encoding="utf-8"?> 2. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android" 3. xmlns:tools="http://schemas.android.com/tools" 4. android:layout\_width="match\_parent" 5. android:layout\_height="match\_parent" 6. android:paddingBottom="@dimen/activity\_vertical\_margin" 7. android:paddingLeft="@dimen/activity\_horizontal\_margin" 8. android:paddingRight="@dimen/activity\_horizontal\_margin" 9. android:paddingTop="@dimen/activity\_vertical\_margin" 10. android:orientation="vertical"> 12. <TextView 13. android:layout\_width="match\_parent" 14. android:layout\_height="wrap\_content" 15. android:text="@string/length" /> 17. <EditText 18. android:id="@+id/edt\_length" 19. android:layout\_width="match\_parent" 20. android:layout\_height="wrap\_content" 21. android:inputType="numberDecimal" 22. android:lines="1" /> 24. <TextView 25. android:layout\_width="match\_parent" 26. android:layout\_height="wrap\_content" 27. android:text="@string/width" /> 29. <EditText 30. android:id="@+id/edt\_width" 31. android:layout\_width="match\_parent" 32. android:layout\_height="wrap\_content" 33. android:inputType="numberDecimal" 34. android:lines="1" /> 36. <TextView 37. android:layout\_width="match\_parent" 38. android:layout\_height="wrap\_content" 39. android:text="@string/height" /> 41. <EditText 42. android:id="@+id/edt\_height" 43. android:layout\_width="match\_parent" 44. android:layout\_height="wrap\_content" 45. android:inputType="numberDecimal" 46. android:lines="1" /> 48. <Button 49. android:id="@+id/btn\_calculate" 50. android:layout\_width="match\_parent" 51. android:layout\_height="wrap\_content" 52. android:text="@string/calculate" /> 54. <TextView 55. android:id="@+id/tv\_result" 56. android:layout\_width="match\_parent" 57. android:layout\_height="wrap\_content" 58. android:gravity="center" 59. android:text="@string/result" 60. android:textSize="24sp" 61. android:textStyle="bold" /> 62. </LinearLayout> |
| --- |

Jika Anda perhatikan, hasil layout sementara akan menjadi seperti ini:



**Coding Logika**

**Menambahkan Kode Logika Sederhana pada MainActivity.**

1. Selanjutnya setelah selesai, lanjutkan dengan membuka berkas MainActivity dan lanjutkan ngoding baris-baris di bawah ini.  
   Tambahkan beberapa variabel yang akan digunakan untuk menampung View.

| Java |
| --- |
| 1. private EditText edtWidth; 2. private EditText edtHeight; 3. private EditText edtLength; 4. private Button btnCalculate; 5. private TextView tvResult; |

| Kotlin |
| --- |
| 1. private lateinit var edtWidth: EditText 2. private lateinit var edtHeight: EditText 3. private lateinit var edtLength: EditText 4. private lateinit var btnCalculate: Button 5. private lateinit var tvResult: TextView |

Kemudian inisiasi variabel yang telah kita buat dengan menambahkan kode berikut di dalam metode onCreate.

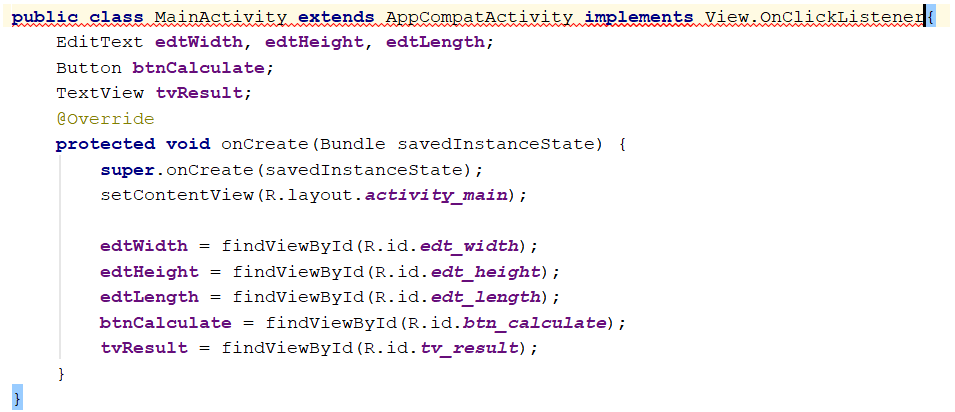
| Java |
| --- |
| 1. @Override 2. protected void onCreate(Bundle savedInstanceState) { 3. super.onCreate(savedInstanceState); 4. setContentView(R.layout.activity\_main); 6. **edtWidth = findViewById(R.id.edt\_width);** 7. **edtHeight = findViewById(R.id.edt\_height);** 8. **edtLength = findViewById(R.id.edt\_length);** 9. **btnCalculate = findViewById(R.id.btn\_calculate);** 10. **tvResult = findViewById(R.id.tv\_result);** 11. } |

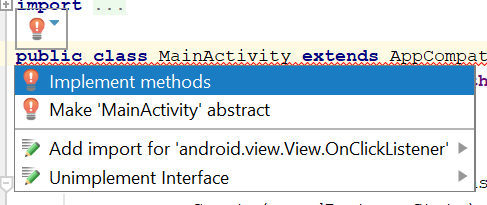
| Kotlin |
| --- |
| 1. override fun onCreate(savedInstanceState: Bundle?) { 2. super.onCreate(savedInstanceState) 3. setContentView(R.layout.activity\_main) 5. **edtWidth = findViewById(R.id.edt\_width)** 6. **edtHeight = findViewById(R.id.edt\_height)** 7. **edtLength = findViewById(R.id.edt\_length)** 8. **btnCalculate = findViewById(R.id.btn\_calculate)** 9. **tvResult = findViewById(R.id.tv\_result)** 10. } |

Kemudian tambahkan implementasi interface onClickListener pada kelas MainActivity.

| Java |
| --- |
| 1. public class MainActivity extends AppCompatActivity **implements View.OnClickListener** { 3. ... 5. } |

| Kotlin |
| --- |
| 1. class MainActivity : AppCompatActivity()**,** **View.OnClickListener** { 3. ... 5. } |

Jika terdapat baris merah seperti ini :  


Jangan khawatir! Silakan klik di atas baris merah tersebut dan klik pada ikonbola lampu**.**Atau dengan tekan tombol**Alt + Enter (option + return pada Mac)**atau menekan lampu merah yang muncullalu pilih **implement methods.**  
Maka secara

otomatis akan ada penambahan metode onClick di kelas MainActivity. Tambahkan kode berikut ini.

| 1. @Override 2. protected void onCreate(Bundle savedInstanceState) { 3. super.onCreate(savedInstanceState); 4. setContentView(R.layout.activity\_main); 6. ... 8. **btnCalculate.setOnClickListener(this);** 9. } 11. **@Override** 12. **public void onClick(View v) {** 13. **if (v.getId() == R.id.btn\_calculate) {** 14. **String inputLength = edtLength.getText().toString().trim();** 15. **String inputWidth = edtWidth.getText().toString().trim();** 16. **String inputHeight = edtHeight.getText().toString().trim();** 17. **boolean isEmptyFields = false;** 18. **boolean isInvalidDouble = false;** 19. **if (TextUtils.isEmpty(inputLength)) {** 20. **isEmptyFields = true;** 21. **edtLength.setError("Field ini tidak boleh kosong");** 22. **}** 23. **if (TextUtils.isEmpty(inputWidth)) {** 24. **isEmptyFields = true;** 25. **edtWidth.setError("Field ini tidak boleh kosong");** 26. **}** 27. **if (TextUtils.isEmpty(inputHeight)) {** 28. **isEmptyFields = true;** 29. **edtHeight.setError("Field ini tidak boleh kosong");** 30. **}** 31. **Double length = toDouble(inputLength);** 32. **Double width = toDouble(inputWidth);** 33. **Double height = toDouble(inputHeight);** 34. **if (length == null) {** 35. **isInvalidDouble = true;** 36. **edtLength.setError("Field ini harus berupa nomer yang valid");** 37. **}** 38. **if (width == null) {** 39. **isInvalidDouble = true;** 40. **edtWidth.setError("Field ini harus berupa nomer yang valid");** 41. **}** 42. **if (height == null) {** 43. **isInvalidDouble = true;** 44. **edtHeight.setError("Field ini harus berupa nomer yang valid");** 45. **}** 46. **if (!isEmptyFields && !isInvalidDouble) {** 47. **double volume = length \* width \* height;** 48. **tvResult.setText(String.valueOf(volume));** 49. **}** 50. **}** 51. **}** |
| --- |

Terakhir tambahkan kode untuk validasi nilai *double*.

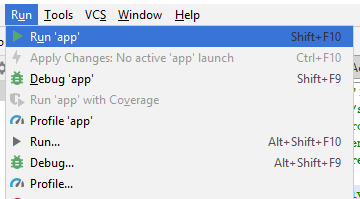
| Java |
| --- |
| 1. private Double toDouble(String str) { 2. try { 3. return Double.valueOf(str); 4. } catch (NumberFormatException e) { 5. return null; 6. } 7. } |

| Kotlin |
| --- |
| 1. private fun toDouble(str: String): Double? { 2. return try { 3. str.toDouble() 4. } catch (e: NumberFormatException) { 5. null 6. } 7. } |

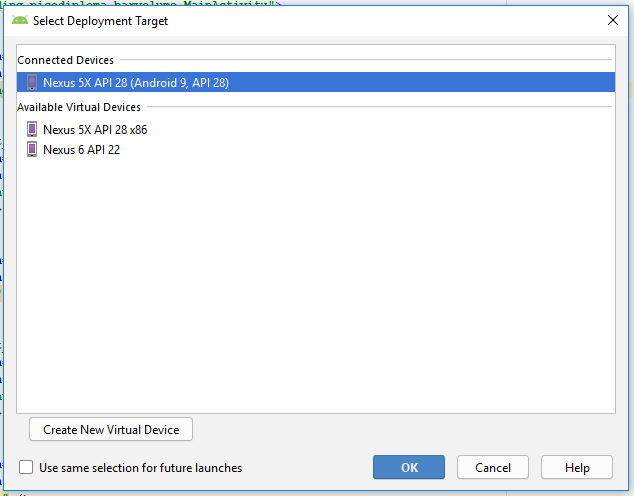
Akhirnya kelas MainActivity akan memiliki kode seperti berikut ini.

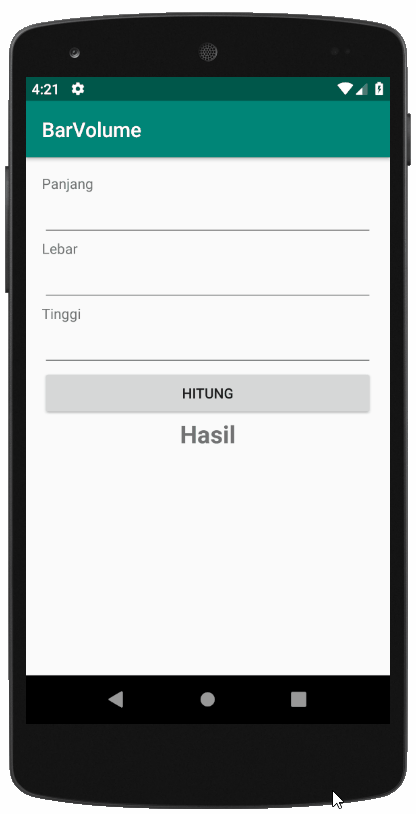
| 1. public class MainActivity extends AppCompatActivity implements View.OnClickListener { 2. private EditText edtWidth, edtHeight, edtLength; 3. private Button btnCalculate; 4. private TextView tvResult; 6. @Override 7. protected void onCreate(Bundle savedInstanceState) { 8. super.onCreate(savedInstanceState); 9. setContentView(R.layout.activity\_main); 11. edtWidth = findViewById(R.id.edt\_width); 12. edtHeight = findViewById(R.id.edt\_height); 13. edtLength = findViewById(R.id.edt\_length); 14. btnCalculate = findViewById(R.id.btn\_calculate); 15. tvResult = findViewById(R.id.tv\_result); 17. btnCalculate.setOnClickListener(this); 18. } 20. @Override 21. public void onClick(View v) { 22. if (v.getId() == R.id.btn\_calculate) { 23. String inputLength = edtLength.getText().toString().trim(); 24. String inputWidth = edtWidth.getText().toString().trim(); 25. String inputHeight = edtHeight.getText().toString().trim(); 27. boolean isEmptyFields = false; 28. boolean isInvalidDouble = false; 30. if (TextUtils.isEmpty(inputLength)) { 31. isEmptyFields = true; 32. edtLength.setError("Field ini tidak boleh kosong"); 33. } 35. if (TextUtils.isEmpty(inputWidth)) { 36. isEmptyFields = true; 37. edtWidth.setError("Field ini tidak boleh kosong"); 38. } 40. if (TextUtils.isEmpty(inputHeight)) { 41. isEmptyFields = true; 42. edtHeight.setError("Field ini tidak boleh kosong"); 43. } 45. Double length = toDouble(inputLength); 46. Double width = toDouble(inputWidth); 47. Double height = toDouble(inputHeight); 49. if (length == null) { 50. isInvalidDouble = true; 51. edtLength.setError("Field ini harus berupa nomer yang valid"); 52. } 54. if (width == null) { 55. isInvalidDouble = true; 56. edtWidth.setError("Field ini harus berupa nomer yang valid"); 57. } 59. if (height == null) { 60. isInvalidDouble = true; 61. edtHeight.setError("Field ini harus berupa nomer yang valid"); 62. } 64. if (!isEmptyFields && !isInvalidDouble) { 65. double volume = length \* width \* height; 66. tvResult.setText(String.valueOf(volume)); 67. } 68. } 69. } 71. private Double toDouble(String str) { 72. try { 73. return Double.valueOf(str); 74. } catch (NumberFormatException e) { 75. return null; 76. } 77. } 78. } |
| --- |

1. Setelah selesai, silakan jalankan aplikasi dengan memilih menu **Run → Run ‘app’** dari menu bar.



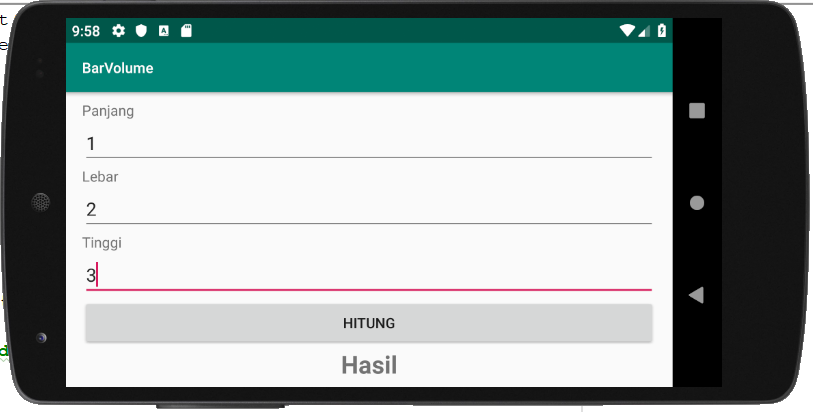
Kemudian ada pilihan seperti ini.



Itu tandanya adb (*Android Debugger*) pada peranti yang Anda punya telah terhubung dengan Android Studio. Jika Anda tidak memiliki peranti, maka Anda dapat menggunakan emulator. Ikuti materinya di modul sebelumnya [https://www.dicoding.com/academies/14/tutorials/1132](https://www.dicoding.com/academies/14/tutorials/1132?) atau di sini <https://developer.android.com/studio/run/managing-avds.html>.  
  
Direkomendasikan menggunakan peranti Android sewaktu mengembangkan aplikasi. Selain karena beban memori pada peranti Anda akan jadi lebih rendah, pendekatan ini juga akan memungkinkan Anda untuk merasakan bagaimana aplikasi berjalan di device sebenarnya.  
  
Pilih **OK**untuk menjalankan dan tunggu hingga proses *building* dan instalasi APK selesai. Jika sudah, seharusnya hasilnya akan seperti ini.  


Silakan masukan nilai panjang, lebar dan tinggi kemudian tekan tombol **Hitung**dan hasilnya akan ditampilkan di obyek textview tvHasil*.*

1. Apakah kita sudah selesai?

Belum! Masih ada yang kurang. Ketika nilai volume sudah dihitung dan kemudian terjadi pergantian orientasi (portrait-landscape) pada peranti, maka hasil perhitungan tadi akan hilang.   


Untuk mengatasinya, tambahkan metode onSaveInstanceState() pada MainActivity dan sesuaikan seperti berikut:

| Java |
| --- |
| 1. **private static final String STATE\_RESULT = "state\_result";** 3. ... 5. @Override 6. protected void onSaveInstanceState(Bundle outState) { 7. super.onSaveInstanceState(outState); 8. **outState.putString(STATE\_RESULT, tvResult.getText().toString());** 9. } |

| Kotlin |
| --- |
| 1. **companion object {** 2. **private const val STATE\_RESULT = "state\_result"** 3. **}** 5. **...** 7. override fun onSaveInstanceState(outState: Bundle) { 8. super.onSaveInstanceState(outState) 9. **outState.putString(STATE\_RESULT, tvResult.text.toString())** 10. } |

Kemudian tambahkan juga beberapa baris berikut pada baris terakhir metode onCreate.

| Java |
| --- |
| 1. @Override 2. protected void onCreate(Bundle savedInstanceState) { 3. super.onCreate(savedInstanceState); 4. setContentView(R.layout.activity\_main); 6. ... 8. **if (savedInstanceState != null) {** 9. **String result = savedInstanceState.getString(STATE\_RESULT);** 10. **tvResult.setText(result);** 11. **}** 12. } |

| Kotlin |
| --- |
| 1. override fun onCreate(savedInstanceState: Bundle?) { 2. super.onCreate(savedInstanceState) 3. setContentView(R.layout.activity\_main) 5. ... 7. **if (savedInstanceState != null) {** 8. **val result = savedInstanceState.getString(STATE\_RESULT) as String** 9. **tvResult.text = result** 10. **}** 11. } |

1. Silakan jalankan kembali aplikasinya. Ulangi proses perhitungan seperti sebelumnya. Kemudian ganti orientasi peranti Anda. Jika sudah benar maka hasil perhitungan tidak akan hilang.

### **Bedah Kode**

#### Pembahasan tentang layout xml

Layout merupakan user interface dari suatu activity. Layout dituliskan dalam format xml (extensible markup language).

| 1. xml version="1.0" encoding="utf-8"?> |
| --- |

Baris ini mengidentifikasi bahwa berkas ini berformat xml.

| 1. xmlns:android="http://schemas.android.com/apk/res/android" |
| --- |

Kode di atas menandakan namespace yang digunakan dalam keseluruhan berkas xml ini.

#### Macam Views

Di sini kita menggunakan beberapa komponen *user interface* yang disebut view. Diantaranya:

**TextView**             : Komponen view untuk menampilkan teks ke layar

**EditText**               : Komponen viewuntuk memberikan input teks

**Button**                  : Komponen viewuntuk melakukan sebuah aksi klik

**LinearLayout**     : Komponen viewbertipe viewgroupyang menjadi *parent* dari semua sub komponen view(sub view) di dalamnya. Komponen ini bersifat sebagai kontainer untuk komponen lain dengan orientasi secara vertikal atau horizontal.

Cara membaca :

| 1. <TextView 2. android:layout\_width="match\_parent" 3. android:layout\_height="wrap\_content" 4. android:text="@string/calculate" 5. android:layout\_marginBottom="@dimen/activity\_vertical\_margin"/> |
| --- |

Komponen di atas adalah sebuah TextView. Perhatikan gambar di bawah ini. Warna ungu menandakan *namespace*yang digunakan; warna biru adalah atribut dari komponen dan warna hijau adalah nilai dari atribut. Penjelasannya seperti di bawah ini:

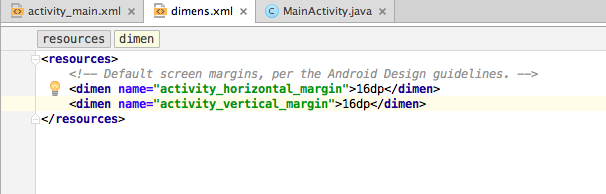




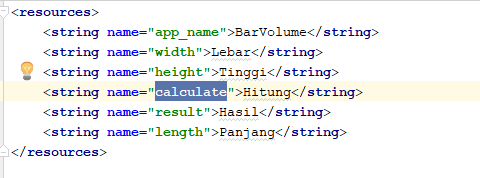
**match\_parent:**Ini berarti bahwa ukuran dimensi sebuah View disesuaikan dengan ukuran layar secara horizontal jika pada layout\_width dan vertikal jika pada layout\_height**.**

**wrap\_content :**Ini berarti bahwa ukuran dimensi sebuah View disesuaikan dengan ukuran konten di dalamnya baik secara horizontal pada layout\_width dan vertikal jika pada layout\_height**.**

**@dimen/activity\_vertical\_margin:**Value activity\_vertical\_margin berasal dari berkasdimens.xml yang bisa Anda lihat dengan cara menekan dan tahan tombol **Ctrl** (atau command) + arahkan kursor ke atasnya dan kemudian klik sekali. .



**@string/calculate:**value calculate berasal dari berkas strings.xml.



Penggunaan *centralize resource value* akan memudahkan Anda sewaktu mengembangkan aplikasi Android. Cara tersebut digunakan agar Anda tidak menulis nilai yang sama berulang-ulang.

#### Apa itu ‘@+id/’ ?

Salah satu contoh penerapan penggunaan **@+id/** sebagai berikut:

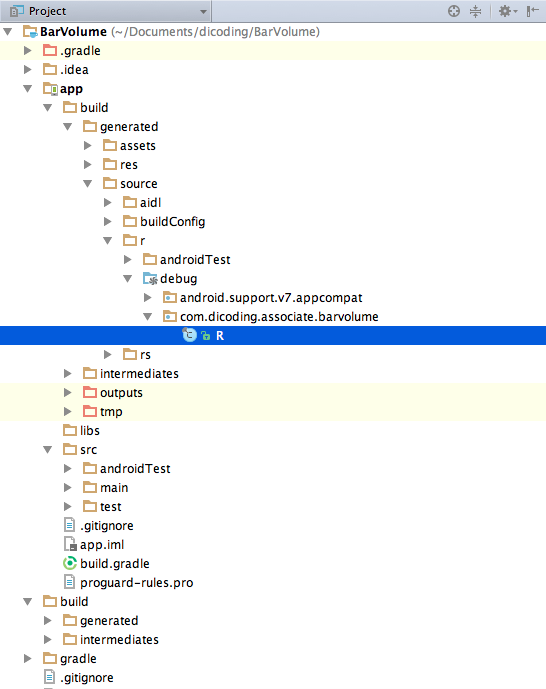
| 1. <Button 2. android:id="@+id/btn\_calculate" 3. android:layout\_width="match\_parent" 4. android:layout\_height="wrap\_content" 5. android:text="@string/calculate"/> |
| --- |

Penjelasannya sebagai berikut:

| * 1. android:id="@+id/btn\_calculate" |
| --- |

Jika kita memberikan id pada sebuah view maka kita telah memberikan *identifier* untuk viewtersebut. Pemberian idini dimaksudkan agar kita bisa melakukan manipulasi/pengendalian pada *level logic* di komponen seperti activity atau fragment.

Iddi atas akan diciptakan di berkas R dan disimpan dalam bentuk *hexa* bertipe data integerpublic static final int btn\_calculate=0x7f0b0057.



Acuan untuk menyusun tampilan pada relativelayout akan dibahas pada modul selanjutnya.

#### Pembahasan tentang Logika Kode

Kode logika dituliskan ke dalam kelas Java atau Koltin. Di sini lah semua aktifitas dari suatu aplikasi berjalan.

**Activity**

| Java |
| --- |
| 1. public class MainActivity extends **AppCompatActivity** |

| Kotlin |
| --- |
| 1. class MainActivity : **AppCompatActivity()** |

Menandakan bahwa kelas Java / Kotlin di atas merupakan sebuah activitykarena *inherit* ke *superclass*bernama AppCompatActivity**.**

**OnClickListener**

| Java |
| --- |
| * 1. implements **View.OnClickListener** |

| Kotlin |
| --- |
| 1. , **View.OnClickListener** |

Ini adalah *listener* yang kita implementasikan untuk memantau kejadian klik pada komponen tombol (button).

**Views**

| Java |
| --- |
| 1. private EditText edtWidth; 2. private EditText edtHeight; 3. private EditText edtLength; 4. private Button btnCalculate; 5. private TextView tvResult; |

| Kotlin |
| --- |
| 1. private lateinit var edtWidth: EditText 2. private lateinit var edtHeight: EditText 3. private lateinit var edtLength: EditText 4. private lateinit var btnCalculate: Button 5. private lateinit var tvResult: TextView |

Kode di atas mendeklarasikan semua komponen view yang akan dimanipulasi. Kita deklarasikan secara global agar bisa dikenal di keseluruhan bagian kelas.

**OnCreate**

| Java |
| --- |
| 1. @Override 2. protected void onCreate(Bundle savedInstanceState) { 3. super.onCreate(savedInstanceState); 4. setContentView(R.layout.activity\_main); 5. edtWidth = findViewById(R.id.edt\_width); 6. edtHeight = findViewById(R.id.edt\_height); 7. edtLength = findViewById(R.id.edt\_length); 8. btnCalculate = findViewById(R.id.btn\_calculate); 9. tvResult = findViewById(R.id.tv\_result); 11. btnCalculate.setOnClickListener(this); 12. } |

| Kotlin |
| --- |
| 1. override fun onCreate(savedInstanceState: Bundle?) { 2. super.onCreate(savedInstanceState) 3. setContentView(R.layout.activity\_main) 5. edtWidth = findViewById(R.id.edt\_width) 6. edtHeight = findViewById(R.id.edt\_height) 7. edtLength = findViewById(R.id.edt\_length) 8. btnCalculate = findViewById(R.id.btn\_calculate) 9. tvResult = findViewById(R.id.tv\_result) 11. btnCalculate.setOnClickListener(this) 12. } |

Metode onCreate() merupakan metode utama pada activity. Di sinilah kita dapat mengatur layout xml. Semua proses inisialisasi komponen yang digunakan akan dijalankan di sini. Pada metode ini kita *casting* semua komponen viewyang kita telah deklarasikan sebelumnya, agar dapat kita manipulasi.

**SetContentView**

| Java |
| --- |
| 1. setContentView(R.layout.**activity\_main**); |

| Kotlin |
| --- |
| 1. setContentView(R.layout.**activity\_main**) |

Maksud baris di atas adalah kelas MainActivity akan menampilkan tampilan yang berasal dari layout activity\_main.xml**.**

**Casting View**

| Java |
| --- |
| 1. edtWidth = findViewById(R.id.edt\_width); |

| Kotlin |
| --- |
| 1. edtWidth = findViewById(R.id.edt\_width) |

Maksud dari baris diatas adalah obyek edittextedtWidthdisesuaikan (cast) dengan komponen edittextber-ID edt\_widthdi layout xml melalui metode findViewById().

**SetOnClickListener**

| Java |
| --- |
| 1. btnCalculate.setOnClickListener(this); |

| Kotlin |
| --- |
| 1. btnCalculate.setOnClickListener(this) |

Kita memasang *event click listener*untuk obyek btnCalculatesehingga sebuah aksi dapat dijalankan ketika obyek tersebut diklik. Keyword thismerujuk pada obyek Activitysaat ini yang telah mengimplementasikan listener OnClickListenersebelumnya.

#### **Pembahasan saveInstanceState**

| Java |
| --- |
| 1. @Override 2. protectedvoidonSaveInstanceState(Bundle outState) { 3. outState.putString(STATE\_RESULT, tvResult.getText().toString()); 4. super.onSaveInstanceState(outState); 5. } |

| Kotlin |
| --- |
| 1. override fun onSaveInstanceState(outState: Bundle) { 2. super.onSaveInstanceState(outState) 3. outState.putString(STATE\_RESULT, tvResult.text.toString()) 4. } |

Perhatikan metode onSaveInstanceState**.**Di dalam metode tersebut, hasil perhitungan yang ditampilkan pada tvResult dimasukkan pada bundlekemudian disimpan isinya. Setelah onSaveInstanceState berhasil dijalankan, maka activity akan melakukan proses onDestroy dan menjalankan kembali onCreate secara otomatis.

| Java |
| --- |
| 1. if (savedInstanceState != null){ 2. String hasil = savedInstanceState.getString(STATE\_RESULT); 3. **tvResult**.setText(hasil); 4. } |

| Kotlin |
| --- |
| 1. if (savedInstanceState != null) { 2. val result = savedInstanceState.getString(STATE\_RESULT) as String 3. **tvResult**.text = result 4. } |

Pada onCreate inilah kita menggunakan nilai bundle yang telah kita simpan sebelumnya pada onSaveInstanceState. Nilai tersebut kita isikan kembali pada tvResult.

Selamat! Anda telah berhasil membuat dan menjalankan latihan Android pertama di peranti atau *emulator*. Silakan lanjut ke codelab berikutnya.