Modul 5 – Service

Anda sudah paham service secara garis besar berikut pemanfaatannya. Sekarang saatnya kita menerapkannya.

1. Baik, buat proyek baru dengan nama MyService. **Pilih Empty Activity** dengan pilihan *default* pada *set up* proyek. Setelah proyek tercipta, lengkapi activity_main.xml dengan contoh seperti ini:

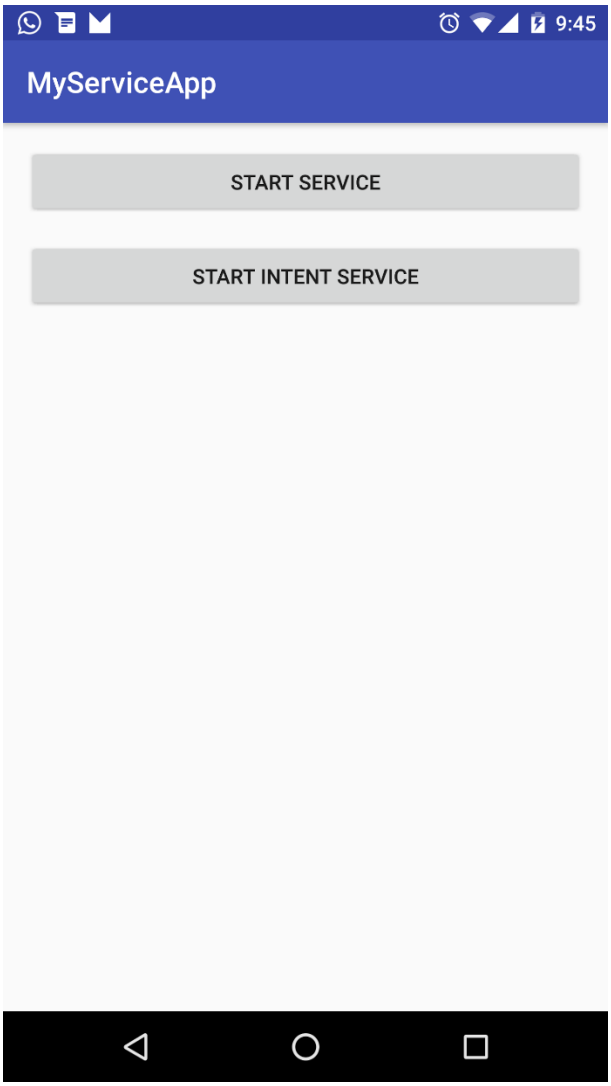
```
1. <?xml version="1.0" encoding="utf-8"?>
2. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android
   "
3.     xmlns:tools="http://schemas.android.com/tools"
4.     android:layout_width="match_parent"
5.     android:layout_height="match_parent"
6.     android:paddingBottom="@dimen/activity_vertical_margin"
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"
11.    tools:context="com.dicoding.myserviceapp.MainActivity">
12.    <Button
13.        android:id="@+id/btn_start_service"
14.        android:layout_width="match_parent"
15.        android:layout_height="wrap_content"
16.        android:text="Start Service"
17.        android:layout_marginBottom="@dimen/activity_vertical_margin"/>
18.    <Button
19.        android:id="@+id/btn_start_intent_service"
20.        android:layout_width="match_parent"
21.        android:layout_height="wrap_content"
22.        android:text="Start Intent Service"/>
23. </LinearLayout>
```

2. Pada MainActivity.java silakan lengkapi kode-nya menjadi sebagai berikut:

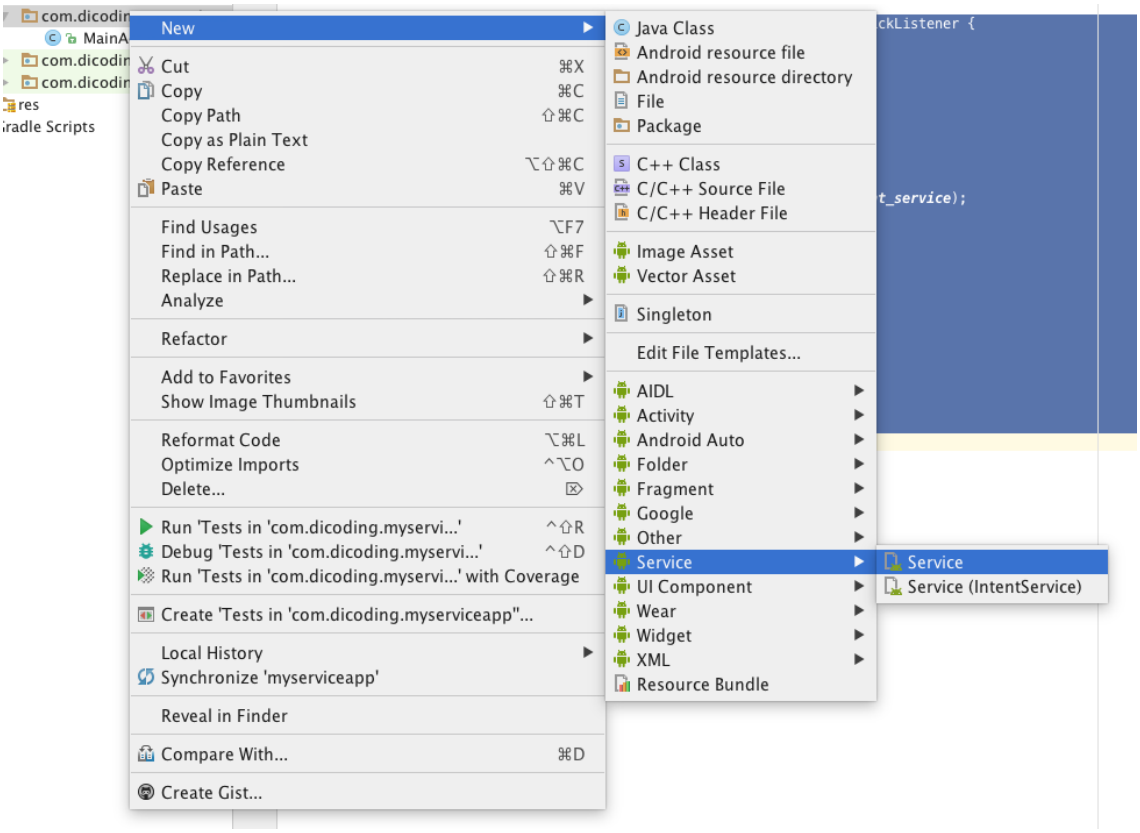
```
1. public class MainActivity extends AppCompatActivity implements View.OnC
   lickListener {
2.     private Button btnStartService;
3.     private Button btnStartIntentService;
4.
5.     @Override
6.     protected void onCreate(Bundle savedInstanceState) {
7.         super.onCreate(savedInstanceState);
8.         setContentView(R.layout.activity_main);
9.
10.        btnStartService = (Button)findViewById(R.id.btn_start_service);
11.        btnStartService.setOnClickListener(this);
```

```
12.         btnStartIntentService = (Button)findViewById(R.id.btn_start_int
    ent_service);
13.         btnStartIntentService.setOnClickListener(this);
14.     }
15.
16.     @Override
17.     public void onClick(View v) {
18.         switch (v.getId()){
19.             case R.id.btn_start_service:
20.                 break;
21.             case R.id.btn_start_intent_service:
22.                 break;
23.         }
24.     }
25. }
```

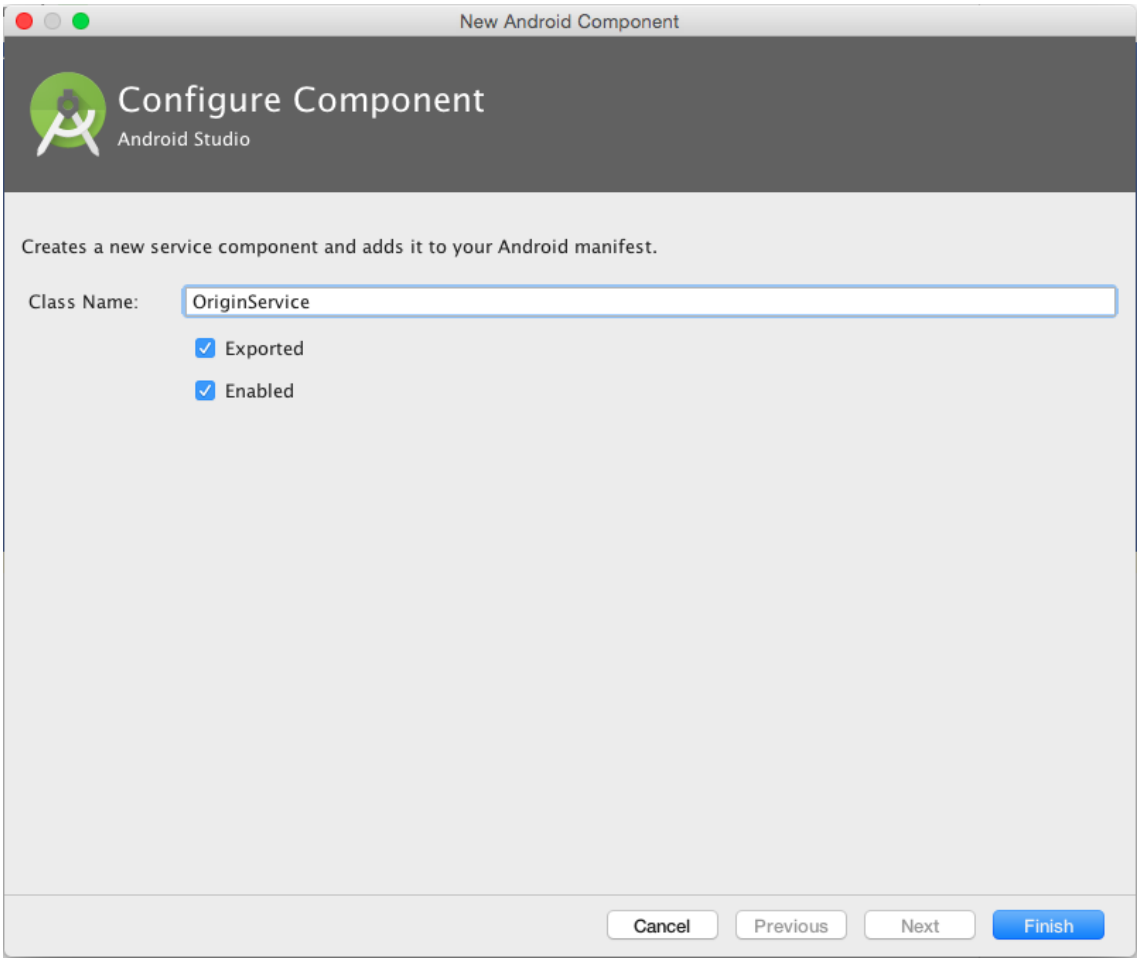
3. Tampilan yang seharusnya ada pada MainActivity, adalah seperti ini.



4. Lanjut, buat kelas service bernama OriginService dengan cara **klik kanan** pada **package project** → **New** → **Service** → **Service**. OriginService akan *inherit* (*extends*) kepada kelas Service.



5. Selanjutnya pada dialog yang tampil, isikan nama kelas service yang diinginkan. Di sini kita menamainya sebagai OriginService dan biarkan *exported* dan *enabled* tercentang. Klik **Finish** untuk menyelesaikan proses.



6. Selanjutnya, buka berkas AndroidManifest.xml pada package manifest dan perhatikan isi berkas tersebut. Service yang baru saja kita buat sudah ada didalam tag <application> :

```
1.  <?xml version="1.0" encoding="utf-8"?>
2.  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3.      package="com.dicoding.myserviceapp">
4.      <application
5.          android:allowBackup="true"
6.          android:icon="@mipmap/ic_launcher"
7.          android:label="@string/app_name"
8.          android:supportsRtl="true"
9.          android:theme="@style/AppTheme">
10.         <activity android:name=".MainActivity">
11.             <intent-filter>
12.                 <action android:name="android.intent.action.MAIN" />
13.                 <category android:name="android.intent.category.LAUNCHE
    R" />
14.             </intent-filter>
15.         </activity>
16.         <service
17.             android:name=".OriginService"
18.             android:enabled="true"
19.             android:exported="true">
```

```
        </service>
    </application>
</manifest>
```

7. Berkas AndroidManifest sudah dibuat secara otomatis. Dengan demikian kita sudah bisa menjalankan kelas service tersebut. Namun, sebelum menjalankan aplikasi, lengkapi kode pada OriginService menjadi seperti berikut :

```
public class OriginService extends Service {
    public static final String ORIGIN_SERVICE = "OriginService";

    public OriginService() {

    }
    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Return the communication channel to the service.
        throw new UnsupportedOperationException("Not yet implemented");
    }
    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        Log.d(ORIGIN_SERVICE, "OriginService dijalankan");
        return START_STICKY;
    }
}
```

8. Selanjutnya pada MainActivity.java di metode onClick() pada case R.id.btn_start_service tambahkan baris berikut :

```
Intent mStartServiceIntent = new Intent(MainActivity.this, OriginServic
e.class);
startService(mStartServiceIntent);
```

9. Sehingga kode pada metode onClick() menjadi seperti ini :

```
@Override
public void onClick(View v) {
    switch (v.getId()){
        case R.id.btn_start_service:
            Intent mStartServiceIntent = new Intent(MainActivity.this,
OriginService.class);
            startService(mStartServiceIntent);
            break;
        case R.id.btn_start_intent_service:
            break;
    }
}
```

10. Sekarang jalankan aplikasi. Klik tombol 'start service' dan perhatikan pada *log*-nya. OriginService telah dijalankan dan tidak akan pernah mati sampai dimatikan oleh sistem atau metode stopSelf() atau stopService() dijalankan.

11. Baik, sekarang kita akan menambahkan sebuah inner class AsyncTask. Ia seakan-akan menjalankan sebuah proses secara *asynchronous* dan mematikan/menghentikan dirinya sendiri dengan memanggil metode stopSelf(). Lengkapi kodenya menjadi sebagai berikut:

```
1.  public class OriginService extends Service {
2.
3.      public static final String ORIGIN_SERVICE = "OriginService";
4.
5.      public OriginService() {
6.
7.      }
8.
9.      @Override
10.     public IBinder onBind(Intent intent) {
11.         // TODO: Return the communication channel to the service.
12.         throw new UnsupportedOperationException("Not yet implemented");
13.     }
14.
15.     @Override
16.     public int onStartCommand(Intent intent, int flags, int startId) {
17.         Log.d(ORIGIN_SERVICE, "OriginService dijalankan");
18.         ProcessAsync mProcessAsync = new ProcessAsync();
19.         mProcessAsync.execute();
20.         return START_STICKY;
21.     }
22.
23.     private class ProcessAsync extends AsyncTask<Void, Void, Void>{
24.         @Override
25.         protected Void doInBackground(Void... params) {
26.             try {
27.                 Thread.sleep(3000);
28.             } catch (InterruptedException e) {
29.                 e.printStackTrace();
30.             }
31.             return null;
32.         }
33.
34.         @Override
35.         protected void onPostExecute(Void aVoid) {
36.             super.onPostExecute(aVoid);
37.             Log.d(ORIGIN_SERVICE, "StopService");
38.             stopSelf();
39.         }
40.     }
41.
42.     @Override
43.     public void onDestroy() {
44.         super.onDestroy();
45.         Log.d(ORIGIN_SERVICE, "onDestroy()");
46.     }
```

```
47. }
```

12. Jalankan aplikasinya. Klik tombol ‘start service’ dan perhatikan *log*-nya. Service dijalankan secara *asynchronous* dan mematikan dirinya sendiri setelah proses selesai.

13. Jika berhasil dijalankan, pada *log* androiod monitor akan seperti ini :
    09-22 09:52:25.028 10209-10209/com.dicoding.myserviceapp D/OriginService: OriginService dijalankan
    09-22 09:52:28.074 10209-10209/com.dicoding.myserviceapp D/OriginService: StopService
    09-22 09:52:28.078 10209-10209/com.dicoding.myserviceapp D/OriginService: onDestroy()

Bedah Kode :

Kita baru saja berkenalan dengan service melalui kelas OriginService. Kita *inherit* kelas dasar dari service seperti yang dijelaskan sebelumnya. Service jenis ini berada pada *thread* yang sama yaitu *ui thread*.

Service di atas dijalankan dengan cara berikut ini :

```
1. Intent mStartServiceIntent = new Intent(MainActivity.this, OriginService.clas
   s);
2. startService(mStartServiceIntent);
```

Ingat! Bukan startActivity() seperti pada contoh sebelumnya. Namun startService() karena kita menginginkan sebuah service yang berjalan. Setelah dijalankan, metode onStartCommand() pada OriginService akan dijalankan.

```
1. @Override
2.     public int onStartCommand(Intent intent, int flags, int startId) {
3.         Log.d(ORIGIN_SERVICE, "OriginService dijalankan");
4.         ProcessAsync mProcessAsync = new ProcessAsync();
5.         mProcessAsync.execute();
6.         return START_STICKY;
7.     }
```

Pada metode tersebut, kita menjalankan sebuah *thread* asynctask untuk melakukan proses yang menyerupai proses yang sulit. Dan ia berjalan secara *asynchronous.*

Kekurangan dari service tipe ini adalah tidak menyediakan *worker thread* diluar *ui thread* secara *default*. Jadi tidak ada cara lain, selain membuat *thread* secara sendiri.

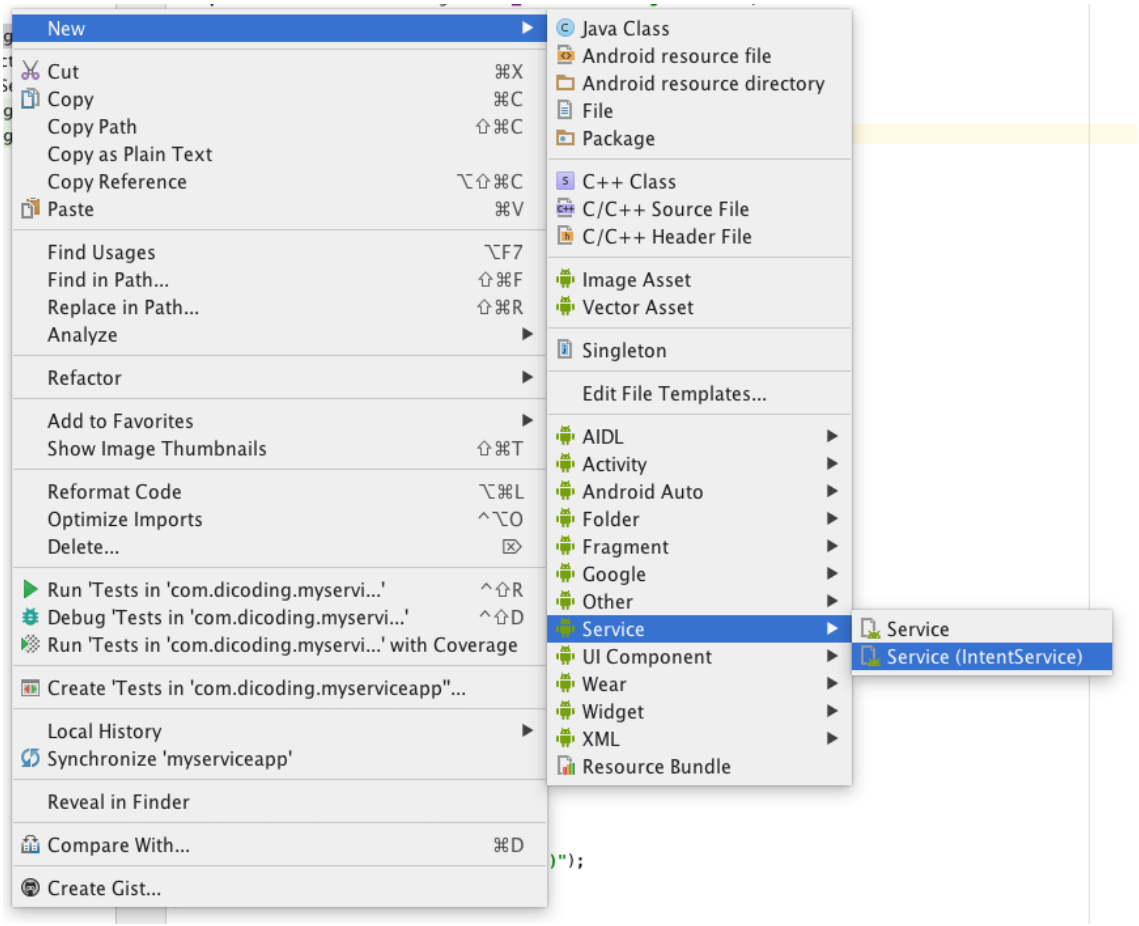
START_STICKY menandakan bahwa bila service tersebut dimatikan oleh sistem Android karena kekurangan memori, maka ia akan diciptakan kembali jika sudah ada memori yang bisa digunakan. Metode onStartCommand() juga akan dijalankan kembali.

```
1. stopSelf();
```

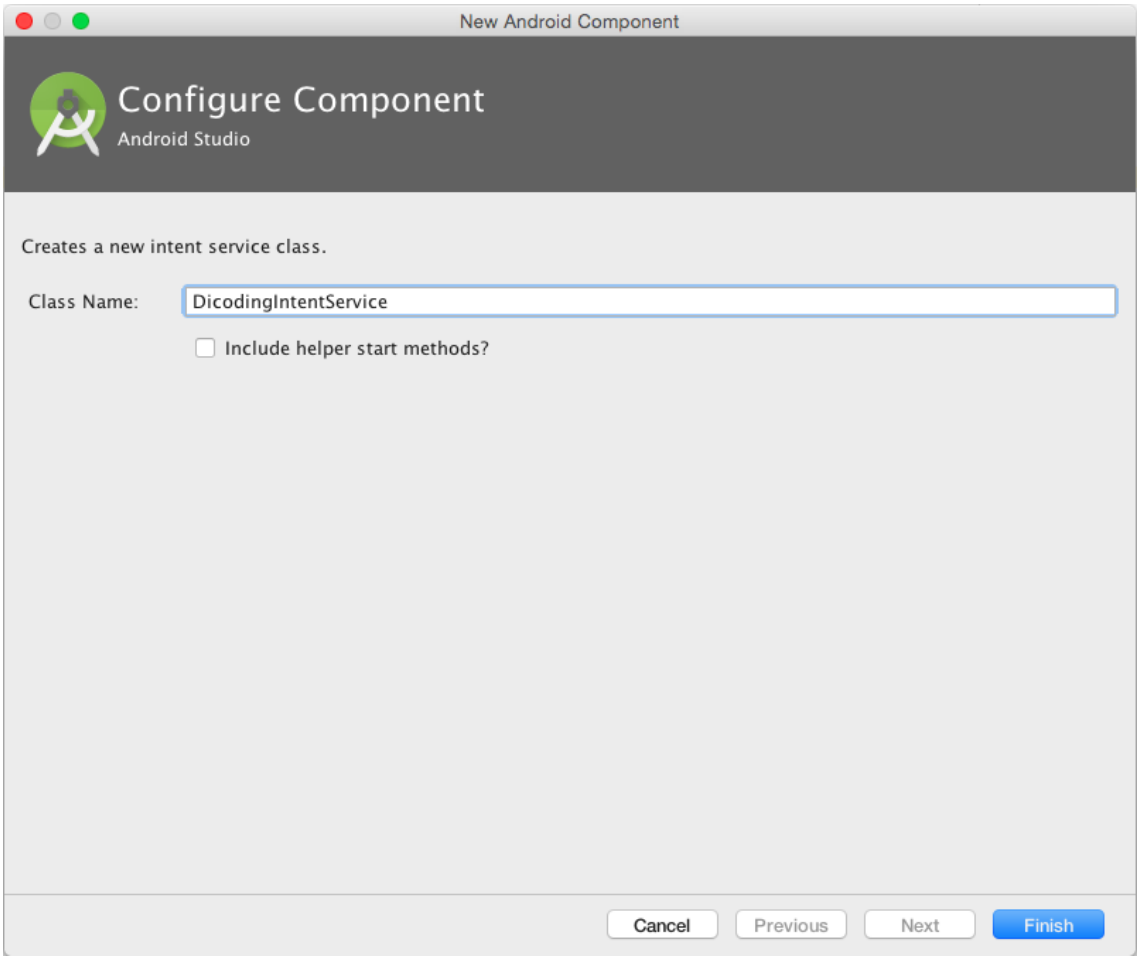
Masih ingat dengan pembahasan asynctask pada materi sebelumnya ? Pada method onPostExecute() akan memanggil stopSelf() yang berarti akan memberhentikan atau mematikan OriginService dari sistem Android.

Pada titik ini, Anda sudah bisa membuat sebuah service yang berjalan di background.

1. Sekarang, buat kembali sebuah service dengan nama DicodingIntentService dengan cara klik kanan pada **package utama project → New → Service → Service (Intent Service)**.



2. Pada dialog yang tampil, masukkan nama nama DicodingIntentService seperti ini :



Klik **Finish** untuk menyelesaikan proses. Pastikan untuk tidak mencentang *include helper*. *Include helper* akan menambahkan metode yang dapat membantu di dalam pembuatan services. Akan tetapi pada modul ini, kita tidak menggunakan metode tersebut.

3. Setelah kelas DicodingIntentService diciptakan, terdapat banyak kode *auto-generated* yang telah disertakan. Kita tidak membutuhkan semua metode tersebut. Sehingga ubah kelas tersebut menjadi seperti contoh di bawah ini :

```
1. public class DicodingIntentService extends IntentService {
2.     public DicodingIntentService() {
3.         super("DicodingIntentService");
4.     }
5.     @Override
6.     protected void onHandleIntent(Intent intent) {
7.         if (intent != null) {
8.         }
9.     }
10. }
```

4. Skenarionya adalah sebagai berikut :
   - Kita menjalankan IntentService tersebut dengan sebuah obyek Intent dari MainActivity dengan membawa data, dalam konteks ini adalah nilai integer yang menentukan berapa lama *background* proses dijalankan.
   - DicodingIntentService dijalankan, dan kemudian melakukan pemrosesan obyek Intent yang dikirimkan untuk menjalankan *background*.
   - Seperti sifatnya, IntentService tidak perlu mematikan dirinya sendiri, secara otomatis ketika proses yang dilakukan selesai, maka, IntentService berhenti dengan sendirinya.

5. Pada DicodingIntentService lengkapi kodenya menjadi seperti berikut :

```
0.  public class DicodingIntentService extends IntentService {
1.      public static String EXTRA_DURATION = "extra_duration";
2.      public static final String TAG = "DicodingIntentService";
3.      public DicodingIntentService() {
4.          super("DicodingIntentService");
5.      }
6.      @Override
7.      protected void onHandleIntent(Intent intent) {
8.                  Log.d(TAG, "onHandleIntent()");
9.          if (intent != null) {
10.             int duration = intent.getIntExtra(EXTRA_DURATION, 0);
11.             try {
12.                 Thread.sleep(duration);
13.             } catch (InterruptedException e) {
14.                 e.printStackTrace();
15.                 Thread.currentThread().interrupt();
16.             }
17.         }
18.     }
19.     @Override
20.     public void onDestroy() {
21.         super.onDestroy();
22.         Log.d(TAG, "onDestroy()");
23.     }
24. }
```

6. Selanjutnya pada MainActivity.java di metode onClick() pada *case* R.id.btn_start_intent_service kita tambahkan 3 baris berikut untuk menjalankan IntentService yang baru saja dibuat.

```
0.  Intent mStartIntentService = new Intent(MainActivity.this, DicodingI
    ntentService.class);
1.  mStartIntentService.putExtra(DicodingIntentService.EXTRA_DURATION, 5
    000);
2.  startService(mStartIntentService);
```

Sehingga kode metode onClick() kita menjadi sebagai berikut.

```
3.  @Override
4.  public void onClick(View v) {
5.     switch (v.getId()){
6.         case R.id.btn_start_service:
7.             Intent mStartServiceIntent = new Intent(MainActivity.this
    , OriginService.class);
8.             startService(mStartServiceIntent);
9.             break;
10.        case R.id.btn_start_intent_service:
11.            Intent mStartIntentService = new Intent(MainActivity.this
    , DicodingIntentService.class);
12.            mStartIntentService.putExtra(DicodingIntentService.EXTRA_
    DURATION, 5000);
13.            startService(mStartIntentService);
14.            break;
15.    }
16. }
```

7. Setelah selesai semua, silakan jalankan kembali aplikasinya dan perhatikan proses yang terekam pada *log* aplikasi di Android Studio.
   09-22 10:32:25.555 10209-28872/com.dicoding.myserviceapp D/DicodingIntentService: onHandleIntent()
   09-22 10:32:30.557 10209-10209/com.dicoding.myserviceapp D/DicodingIntentService: onDestroy()

Bedah Kode

Kita tinjau kembali skenarionya. Kita menjalankan IntentService tersebut dengan sebuah obyek intent yang membawa data dari MainActivity. Kita dapat menentukan berapa lama service ini akan berjalan.

```
1.  Intent mStartIntentService = new Intent(MainActivity.this, DicodingIntentSe
    rvice.class);
2.  mStartIntentService.putExtra(DicodingIntentService.EXTRA_DURATION, 5000);
3.  startService(mStartIntentService);
```

DicodingIntentService dijalankan. Service tersebut akan melakukan pemrosesan obyek Intent yang dikirimkan dan menjalankan suatu proses yang berjalan di background.

```
1.  @Override
2.     protected void onHandleIntent(Intent intent) {
3.                 Log.d(TAG, "onHandleIntent()");
4.         if (intent != null) {
5.             int duration = intent.getIntExtra(EXTRA_DURATION, 0);
6.             try {
7.                 Thread.sleep(duration);
8.             } catch (InterruptedException e) {
9.                 e.printStackTrace();
10.                Thread.currentThread().interrupt();
11.            }
12.        }
13.    }
```

Kode diatas akan dijalankan pada *thread* terpisah secara *asynchronous*. Jadi kita tidak membutuhkan lagi obyek AsyncTask seperti pada service sebelumnya.

Terakhir, IntentService tidak perlu mematikan dirinya sendiri. Service ini akan berhenti dengan sendirinya ketika sudah menjalankan tugasnya.

Hingga saat ini, Anda sudah mempelajari teori dasar dari service. Pada modul berikutnya, Anda akan menerapkan service pada kasus yang lebih nyata.

Lembar Kerja Praktikum

NPM:	Nama Asisten:
Nama:	Nilai:
Kelas:	

Form ini digunakan untuk menilai kerja mahasiswa selama proses praktikum berlangsung. Selama praktikum mahasiswa diminta untuk membuat Aplikasi Service. Semua tombol harus dapat berfungsi sesuai dengan materi yang telah disampaikan.