

Services, Part 2

Prof. Dr.-Ing. V.Brovkov

Service

- Service ist eine Anwendung, die im Hintergrund ohne UI läuft.
- Services können Hintergrundaufgaben übernehmen, die keine Interaktion mit dem Benutzer erfordern und auch weiterlaufen sollen, wenn andere Anwendungen im Vordergrund laufen.
 - Musik soll z.B. auch weiterlaufen, wenn wir gerade eine E-Mail verfassen oder auf Webseiten surfen.
- Hintergrunddienste können periodisch irgendetwas überwachen, z.B. die aktuelle Position, oder Statusmeldungen aus dem Netz empfangen und ggf, eine Benachrichtigung auslösen.
 - Das Herunterladen von Updates und auch die Synchronisierung von Diensten (Kontakte Bilder etc.) finden auch mittels Services statt.
- Grundsätzlich ist die Verwendung von Services mit Bedacht zu wählen, denn je nach Priorität und Ressourcenverbrauch erhöhen Hintergrunddienste den Stromverbrauch
 - Das ist ein häufiger Kritikpunkt am sehr mächtigen Multitasking-Konzept von Android

Beispiel

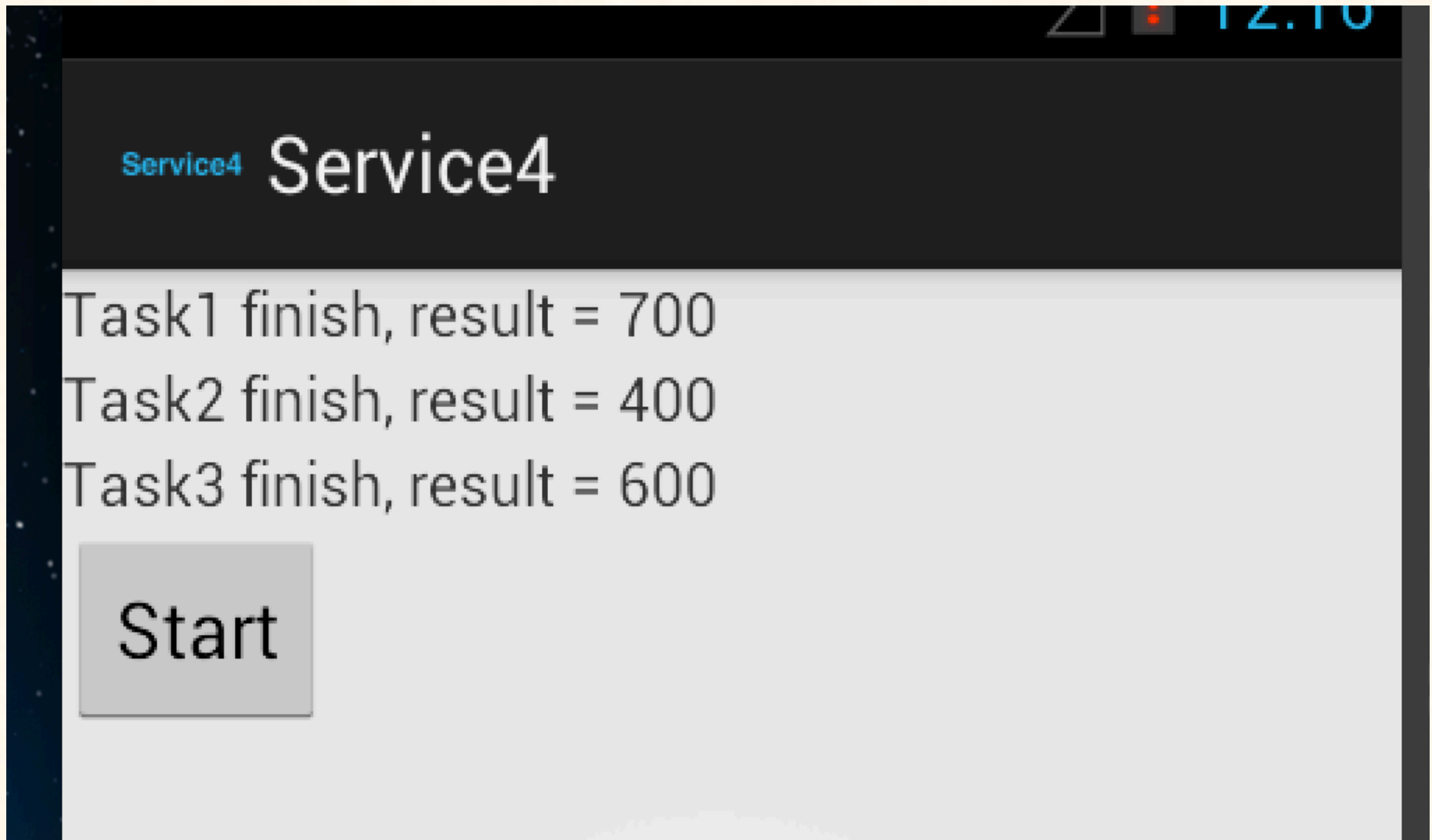
P0951_ServiceBackPendingIntent

<http://startandroid.ru/ru/uroki/vse-uroki-spiskom/160-urok-95-service-obratnaja-svjaz-s-pomoschju-pendingintent.html>

- Untersuchen wir das Datenaustausch zwischen der Activity und dem Service mit Hilfe von **PendingIntent**

P0951_ServiceBackPendingIntent 1

Activity User Interface:



P0951_ServiceBackPendingIntent 2

Ein Zusammenspiel der Activity mit einem Service:

- In der **Activity** erstellen wir ein **PendingIntent Objekt** mit Hilfe von **createPendingResult()** Method
- Das **PendingIntent** wird in einen einfachen **Intent**, eingepackt, womit ein Service mittels **startService()** Method gestartet wird
- Im Service extrahieren wir im Method **onStartCommand()** das **PendingIntent** aus den Object **Intent**
- Wenn wir die Daten aus den **Service** in die **Activity** senden möchten, benutzen wir das **send()** Method für den **PendingIntent** Objekt
- **PendingIntent** enthält eine Verbindung mit der **Activity**, die den erzeugt hat. Das **send()** Method sendet den **PendingIntent** zur Activity, die den hergestellt hat.
- In der **Activity** im Method **onActivityResult()** extrahieren wir die Daten, die das Service geleistet hat.

P0951_ServiceBackPendingIntent 3

```
11 public class MainActivity extends Activity {
12     final String LOG_TAG = "myLogs";
13     final int TASK1_CODE = 1;
14     final int TASK2_CODE = 2;
15     final int TASK3_CODE = 3;
16     public final static int STATUS_START = 100;
17     public final static int STATUS_FINISH = 200;
18     public final static String PARAM_TIME = "time";
19     public final static String PARAM_PINTENT = "pendingIntent";
20     public final static String PARAM_RESULT = "result";
21     TextView tvTask1;
22     TextView tvTask2;
23     TextView tvTask3;
24     /** Called when the activity is first created. */
25     @Override
26     public void onCreate(Bundle savedInstanceState) {
27         super.onCreate(savedInstanceState);
28         setContentView(R.layout.activity_main);
29         tvTask1 = (TextView) findViewById(R.id.tvTask1);
30         tvTask1.setText("Task1");
31         tvTask2 = (TextView) findViewById(R.id.tvTask2);
32         tvTask2.setText("Task2");
33         tvTask3 = (TextView) findViewById(R.id.tvTask3);
34         tvTask3.setText("Task3");
35     }
```

Tasks

Task
Status

P0951_ServiceBackPendingIntent 4

```
36 public void onClickStart(View v) {
37     PendingIntent pi;
38     Intent intent;
39     // PendingIntent for the Task1
40     // Create an Intent for a service call, add a time parameter and the PendingIntent
41     intent = new Intent(this, MyService.class).putExtra(PARAM_TIME, 7);
42     pi = createPendingResult(TASK1_CODE, null, 0);
43     //pi = createPendingResult(TASK1_CODE, intent, 0);
44     intent.putExtra(PARAM_PINTENT, pi);
45     // Start a service
46     startService(intent);
47
48     intent = new Intent(this, MyService.class).putExtra(PARAM_TIME, 4);
49     pi = createPendingResult(TASK2_CODE, intent, 0);
50     intent.putExtra(PARAM_PINTENT, pi);
51     startService(intent);
52
53     intent = new Intent(this, MyService.class).putExtra(PARAM_TIME, 6);
54     pi = createPendingResult(TASK3_CODE, intent, 0);
55     intent.putExtra(PARAM_PINTENT, pi);
56     startService(intent);
57 }
```

Attention! An error here!

An Intent with Time Parameter

A PendingIntent with Task Code, Intent and Flags

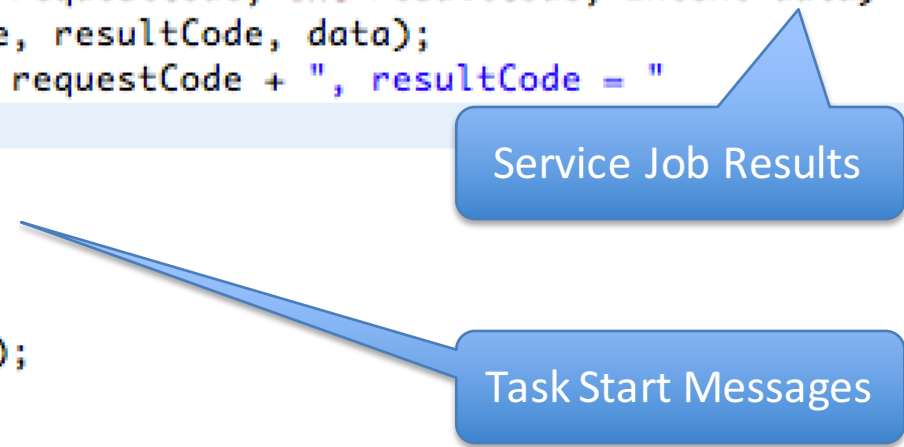
Service Start

The PendingIntent is connected with the Intent as parameter Extra

Correct!

P0951_ServiceBackPendingIntent 5

```
59 @Override
60 protected void onActivityResult(int requestCode, int resultCode, Intent data) {
61     super.onActivityResult(requestCode, resultCode, data);
62     Log.d(LOG_TAG, "requestCode = " + requestCode + ", resultCode = "
63         + resultCode);
64     // Ловим сообщения о старте задач
65     if (resultCode == STATUS_START) {
66         switch (requestCode) {
67             case TASK1_CODE:
68                 tvTask1.setText("Task1 start");
69                 break;
70             case TASK2_CODE:
71                 tvTask2.setText("Task2 start");
72                 break;
73             case TASK3_CODE:
74                 tvTask3.setText("Task3 start");
75                 break;
76         }
77     }
78 }
```



The diagram consists of two blue callout boxes. The first box, labeled 'Service Job Results', has a pointer directed at line 62 of the code, which contains a Log.d statement. The second box, labeled 'Task Start Messages', has a pointer directed at line 65 of the code, which contains an if statement checking for STATUS_START.

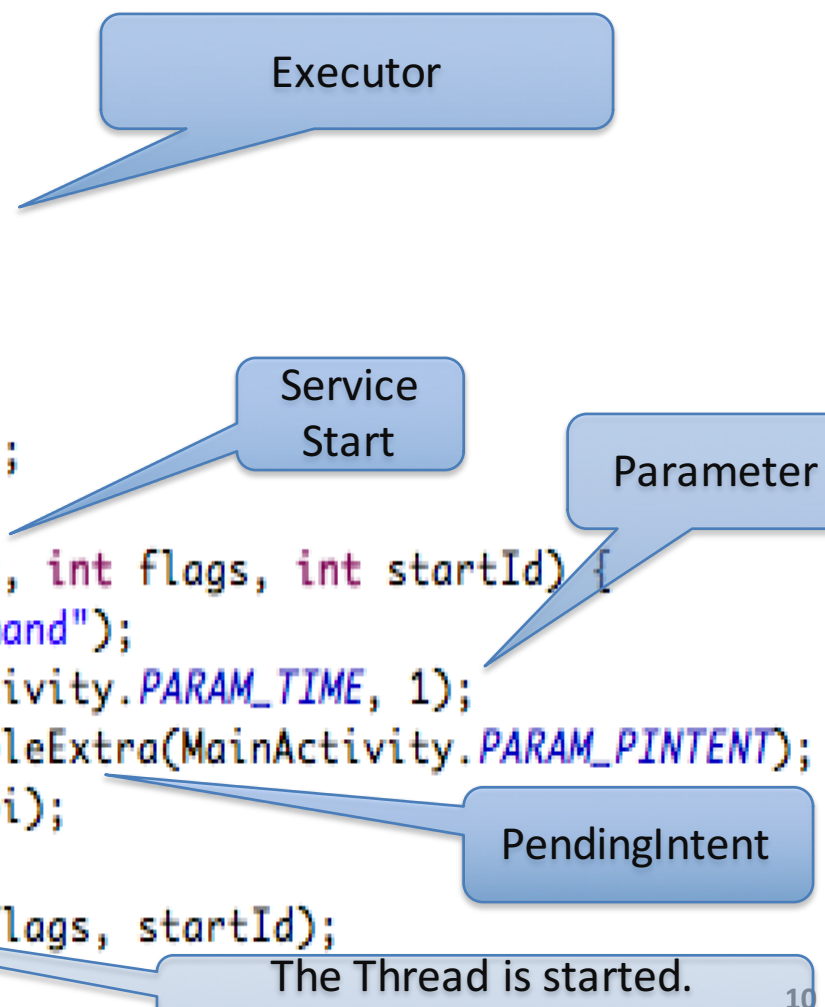
P0951_ServiceBackPendingIntent 6

Task Stop Messages

```
78
79 // Ловим сообщения об окончании задач
80 if (resultCode == STATUS_FINISH) {
81     int result = data.getIntExtra(PARAM_RESULT, 0);
82     switch (requestCode) {
83     case TASK1_CODE:
84         tvTask1.setText("Task1 finish, result = " + result);
85         break;
86     case TASK2_CODE:
87         tvTask2.setText("Task2 finish, result = " + result);
88         break;
89     case TASK3_CODE:
90         tvTask3.setText("Task3 finish, result = " + result);
91         break;
92     }
93 }
94 }
95 }
```

P0951_ServiceBackPendingIntent 10

```
--
13 public class MyService extends Service {
14     final String LOG_TAG = "myLogs";
15     ExecutorService es;
16     public void onCreate() {
17         super.onCreate();
18         Log.d(LOG_TAG, "MyService onCreate");
19         es = Executors.newFixedThreadPool(2);
20     }
21     public void onDestroy() {
22         super.onDestroy();
23         Log.d(LOG_TAG, "MyService onDestroy");
24     }
25     public int onStartCommand(Intent intent, int flags, int startId) {
26         Log.d(LOG_TAG, "MyService onStartCommand");
27         int time = intent.getIntExtra(MainActivity.PARAM_TIME, 1);
28         PendingIntent pi = intent.getParcelableExtra(MainActivity.PARAM_PINTENT);
29         MyRun mr = new MyRun(time, startId, pi);
30         es.execute(mr);
31         return super.onStartCommand(intent, flags, startId);
32     }
}
```



Executor

Service Start

Parameter

PendingIntent

The Thread is started.
PendingIntent is a Parameter

P0951_ServiceBackPendingIntent 11

```
33 public IBinder onBind(Intent arg0) {  
34     return null;  
35 }  
36 class MyRun implements Runnable {  
37     int time;  
38     int startId;  
39     PendingIntent pi;  
40     public MyRun(int time, int startId, PendingIntent pi) {  
41         this.time = time;  
42         this.startId = startId;  
43         this.pi = pi;  
44         Log.d(LOG_TAG, "MyRun#" + startId + " create");  
45     }
```

The Class Constructor.
Time, startID and
PendingIntent are the
Parameters

P0951_ServiceBackPendingIntent 12

```
46 public void run() {  
47     Log.d(LOG_TAG, "MyRun#" + startId + " start, time = " + time);  
48     try {  
49         // сообщаем об старте задачи  
50         pi.send(MainActivity.STATUS_START);  
51         // начинаем выполнение задачи  
52         TimeUnit.SECONDS.sleep(time);  
53         // сообщаем об окончании задачи  
54         Intent intent = new Intent().putExtra(MainActivity.PARAM_RESULT,  
55             time * 100);  
56         pi.send(MyService.this, MainActivity.STATUS_FINISH, intent);  
57     } catch (InterruptedException e) {  
58         e.printStackTrace();  
59     } catch (CanceledException e) {  
60         e.printStackTrace();  
61     }  
62     stop();  
63 }  
64 void stop() {  
65     Log.d(LOG_TAG, "MyRun#" + startId + " end, stopSelfResult(" +  
66         startId + ") = " + stopSelfResult(startId));  
67 }  
68 }  
69 }
```

Send the first Message to the Activity

Service Task Stop

Send the Second Message to the Activity with an Intent

The Service will destroy if the last task was stopped

P0951_ServiceBackPendingIntent 13

Time	Text
03-27 23:37:30.418	requestCode = 1, resultCode = 200
03-27 00:08:01.250	MyService onCreate
03-27 00:08:01.250	MyService onStartCommand
03-27 00:08:01.250	MyRun#1 create
03-27 00:08:01.258	MyService onStartCommand
03-27 00:08:01.258	MyRun#2 create
03-27 00:08:01.258	MyRun#1 start, time = 7
03-27 00:08:01.268	MyService onStartCommand
03-27 00:08:01.268	MyRun#3 create
03-27 00:08:01.268	MyRun#2 start, time = 4
03-27 00:08:01.288	requestCode = 1, resultCode = 100
03-27 00:08:01.288	requestCode = 2, resultCode = 100
03-27 00:08:05.280	MyRun#2 end, stopSelfResult(2) = false
03-27 00:08:05.280	MyRun#3 start, time = 6
03-27 00:08:05.280	requestCode = 2, resultCode = 200
03-27 00:08:05.288	requestCode = 3, resultCode = 100
03-27 00:08:08.270	MyRun#1 end, stopSelfResult(1) = false
03-27 00:08:08.270	requestCode = 1, resultCode = 200
03-27 00:08:11.288	MyRun#3 end, stopSelfResult(3) = true
03-27 00:08:11.288	requestCode = 3, resultCode = 200
03-27 00:08:11.288	MyService onDestroy

Beispiel

P0961_ServiceBackBroadcast

<http://startandroid.ru/ru/uroki/vse-uroki-spiskom/161-urok-96-service-obratnaja-svjaz-s-pomoschju-broadcastreceiver.html>

- Untersuchen wir das Datenaustausch zwischen der Activity und dem Service mit Hilfe von **BroadcastReceiver**

P0961_ServiceBackBroadcast 1



P0961_ServiceBackBroadcast

Task1 finish, result = 700

Task2 finish, result = 400

Task3 finish, result = 600

Start

P0961_ServiceBackBroadcast – MyService 1

```
12 public class MyService extends Service {
13     final String LOG_TAG = "myLogs";
14     ExecutorService es;
15     public void onCreate() {
16         super.onCreate();
17         Log.d(LOG_TAG, "MyService onCreate");
18         es = Executors.newFixedThreadPool(2);
19     }
20     public void onDestroy() {
21         super.onDestroy();
22         Log.d(LOG_TAG, "MyService onDestroy");
23     }
24     public int onStartCommand(Intent intent, int flags, int startId) {
25         Log.d(LOG_TAG, "MyService onStartCommand");
26         int time = intent.getIntExtra(MainActivity.PARAM_TIME, 1);
27         int task = intent.getIntExtra(MainActivity.PARAM_TASK, 0);
28         MyRun mr = new MyRun(startId, time, task);
29         es.execute(mr);
30         return super.onStartCommand(intent, flags, startId);
31     }
32     public IBinder onBind(Intent arg0) {
33         return null;
34     }
35     class MyRun implements Runnable {
70 }
```

P0961_ServiceBackBroadcast – MyService 2

```
35 class MyRun implements Runnable {
36     int time;
37     int startId;
38     int task;
39 public MyRun(int startId, int time, int task) {
40     this.time = time;
41     this.startId = startId;
42     this.task = task;
43     Log.d(LOG_TAG, "MyRun#" + startId + " create");
44 }
45 public void run() {
46     //Intent intent = new Intent(MainActivity.BROADCAST_ACTION);
47     Intent intent = new Intent("ua.opu.brovkov.p0961_servicebackbroadcast");
48     intent.putExtra(MainActivity.PARAM_TASK, task);
49     Log.d(LOG_TAG, "MyRun#" + startId + " start, time = " + time);
50     try {
51         // Start Task Messaging
52         intent.putExtra(MainActivity.PARAM_STATUS, MainActivity.STATUS_START);
53         sendBroadcast(intent);
54         // Task Executing
55         TimeUnit.SECONDS.sleep(time);
56         // Stop Task Messaging
57         intent.putExtra(MainActivity.PARAM_STATUS, MainActivity.STATUS_FINISH);
58         intent.putExtra(MainActivity.PARAM_RESULT, time * 100);
59         sendBroadcast(intent);
60     } catch (InterruptedException e) {
61         e.printStackTrace();
62     }
63     stop();
64 }
65 void stop() {
66     Log.d(LOG_TAG, "MyRun#" + startId + " end, stopSelfResult("
67         + startId + ") = " + stopSelfResult(startId));
68 }
69 }
70 }
```

P0961_ServiceBackBroadcast – MainActivity 1

```
13 public class MainActivity extends Activity {
14     final String LOG_TAG = "myLogs";
15     final int TASK1_CODE = 1;
16     final int TASK2_CODE = 2;
17     final int TASK3_CODE = 3;
18     public final static int STATUS_START = 100;
19     public final static int STATUS_FINISH = 200;
20     public final static String PARAM_TIME = "time";
21     public final static String PARAM_TASK = "task";
22     public final static String PARAM_RESULT = "result";
23     public final static String PARAM_STATUS = "status";
24+ public final static String BROADCAST_ACTION = ..
25
26     TextView tvTask1;
27     TextView tvTask2;
28     TextView tvTask3;
29     BroadcastReceiver br;
30     /** Called when the activity is first created. */
32+ public void onCreate(Bundle savedInstanceState) {..
79+ protected void onDestroy() {..
83+ public void onClickStart(View v) {..
96 }
```

P0961_ServiceBackBroadcast – MainActivity 2

```
31 @Override
32 public void onCreate(Bundle savedInstanceState) {
33     super.onCreate(savedInstanceState);
34     setContentView(R.layout.main);
35     tvTask1 = (TextView) findViewById(R.id.tvTask1);
36     tvTask1.setText("Task1");
37     tvTask2 = (TextView) findViewById(R.id.tvTask2);
38     tvTask2.setText("Task2");
39     tvTask3 = (TextView) findViewById(R.id.tvTask3);
40     tvTask3.setText("Task3");
41     // create a BroadcastReceiver
42     br = new BroadcastReceiver() {
43         // BroadcastReceiver Filter
44         IntentFilter intFilt = new IntentFilter(BROADCAST_ACTION);
45         // BroadcastReceiver registering
46         registerReceiver(br, intFilt);
47     }
48 }
49 @Override
50 protected void onDestroy() {
51     super.onDestroy();
52     unregisterReceiver(br); // BroadcastReceiver unregistering
53 }
```


P0961_ServiceBackBroadcast – MainActivity 3

```
42 br = new BroadcastReceiver() {  
43     // message processing  
44     public void onReceive(Context context, Intent intent) {  
45         int task = intent.getIntExtra(PARAM_TASK, 0);  
46         int status = intent.getIntExtra(PARAM_STATUS, 0);  
47         Log.d(LOG_TAG, "onReceive: task = " + task + ", status = " + status);  
48         // Start message  
49         if (status == STATUS_START) {  
50             switch (task) {  
51                 case TASK1_CODE:  
52                     tvTask1.setText("Task1 start"); break;  
53                 case TASK2_CODE:  
54                     tvTask2.setText("Task2 start"); break;  
55                 case TASK3_CODE:  
56                     tvTask3.setText("Task3 start"); break;  
57             }  
58         }  
59         // Stop message  
60         if (status == STATUS_FINISH) {  
61             int result = intent.getIntExtra(PARAM_RESULT, 0);  
62             switch (task) {  
63                 case TASK1_CODE:  
64                     tvTask1.setText("Task1 finish, result = " + result); break;  
65                 case TASK2_CODE:  
66                     tvTask2.setText("Task2 finish, result = " + result); break;  
67                 case TASK3_CODE:  
68                     tvTask3.setText("Task3 finish, result = " + result); break;  
69             }  
70         }  
71     }  
72 };
```


P0961_ServiceBackBroadcast – MainActivity 4

```
83 public void onClickStart(View v) {  
84     Intent intent;  
85     // Intent for service start with a time and TaskId  
86     intent = new Intent(this, MyService.class).putExtra(PARAM_TIME, 7)  
87         .putExtra(PARAM_TASK, TASK1_CODE);  
88     startService(intent);  
89     intent = new Intent(this, MyService.class).putExtra(PARAM_TIME, 4)  
90         .putExtra(PARAM_TASK, TASK2_CODE);  
91     startService(intent);  
92     intent = new Intent(this, MyService.class).putExtra(PARAM_TIME, 6)  
93         .putExtra(PARAM_TASK, TASK3_CODE);  
94     startService(intent);  
95 }
```

P0961_ServiceBackBroadcast myLogs

Tag	Text
myLogs	MyService onCreate
myLogs	MyService onStartCommand
myLogs	MyRun#1 create
myLogs	MyService onStartCommand
myLogs	MyRun#2 create
myLogs	MyRun#1 start, time = 7
myLogs	MyService onStartCommand
myLogs	MyRun#3 create
myLogs	onReceive: task = 1, status = 100
myLogs	MyRun#2 start, time = 4
myLogs	onReceive: task = 2, status = 100
myLogs	onReceive: task = 2, status = 200
myLogs	MyRun#2 end, stopSelfResult(2) = false
myLogs	MyRun#3 start, time = 6
myLogs	onReceive: task = 3, status = 100
myLogs	MyRun#1 end, stopSelfResult(1) = false
myLogs	onReceive: task = 1, status = 200
myLogs	MyRun#3 end, stopSelfResult(3) = true
myLogs	MyService onDestroy
myLogs	onReceive: task = 3, status = 200

Beispiel

P0971_ServiceBindClient, P0972_ServiceBindServer

<http://startandroid.ru/ru/uroki/vse-uroki-spiskom/161-urok-96-service-obratnaja-svjaz-s-pomoschju-broadcastreceiver.html>

- Untersuchen wir das Datenaustausch zwischen der Activity und dem Service mit Hilfe der Klasse **ServiceConnection**

P0971_ServiceBindClient

```
12 public class MainActivity extends Activity {
13     final String LOG_TAG = "myLogs";
14     boolean bound = false;
15     ServiceConnection sConn;
16     Intent intent;
17     /** Called when the activity is first created. */
18     public void onCreate(Bundle savedInstanceState) {
19         super.onCreate(savedInstanceState);
20         setContentView(R.layout.main);
21         intent = new Intent(
22             "ua.opu.brovkov.p0972_servicebindserver.MyService");
23         sConn = new ServiceConnection() {
24             public void onServiceConnected(ComponentName name, IBinder binder) {
25                 Log.d(LOG_TAG, "MainActivity onServiceConnected");
26                 bound = true;
27             }
28             public void onServiceDisconnected(ComponentName name) {
29                 Log.d(LOG_TAG, "MainActivity onServiceDisconnected");
30                 bound = false;
31             }
32         };
33     }
```

P0971_ServiceBindClient

```
36 public void onClickStart(View v) { startService(intent); }
39
40 public void onClickStop(View v) { stopService(intent); }
43
44 public void onClickBind(View v) { bindService(intent, sConn, BIND_AUTO_CREATE); }
47
48 public void onClickUnBind(View v) {
49     if (!bound)
50         return;
51     unbindService(sConn);
52     bound = false;
53 }
54
55 protected void onDestroy() {
56     super.onDestroy();
57     onClickUnBind(null);
58 }
59 }
```

P0971_ServiceBindService

```
<service android:name="MyService">
  <intent-filter>
    <action android:name="ua.opu.brovkov.p0972_servicebindserver.MyService"/>
  </intent-filter>
</service>
```

```
9  public class MyService extends Service {
10
11      final String LOG_TAG = "myLogs";
12
13      public void onCreate() {
14          super.onCreate();
15          Log.d(LOG_TAG, "MyService onCreate");
16      }
17
18      public IBinder onBind(Intent intent) {
19          Log.d(LOG_TAG, "MyService onBind");
20          return new Binder();
21      }
22
23      public void onRebind(Intent intent) {
24          super.onRebind(intent);
25          Log.d(LOG_TAG, "MyService onRebind");
26      }
27
28      public boolean onUnbind(Intent intent) {
29          Log.d(LOG_TAG, "MyService onUnbind");
30          return super.onUnbind(intent);
31      }
32
33      public void onDestroy() {
34          super.onDestroy();
35          Log.d(LOG_TAG, "MyService onDestroy");
36      }
37  }
```


P0971_ServiceBindService

- Bind:

MyService onCreate
MyService onBind
MainActivity onServiceConnected

- Unbind:

MyService onUnbind
MyService onDestroy

- Bind & kill:

MyService onCreate
MyService onBind
MainActivity onServiceConnected

In wenigen Sekunden:

MyService onCreate
MyService onBind
MainActivity onServiceConnected

Services

Fragen?