Курс:
Программирование под Android с нуля

Тренер:
Пономарев Алексей
oleksii_android@web-academy.com.ua

WEB ACADEMY
PROGRAMMING COURSES

Лекция 12
Services, AsyncTasks

Service

Сервис – это некая задача, которая работает в фоне и не использует UI. Запускать и останавливать сервис можно из приложений и других сервисов. Также можно подключиться к уже работающему сервису и взаимодействовать с ним. Сервис нужен, чтобы ваша задача продолжала работать, даже когда приложение закрыто.

Service

```
public class MyService extends Service {
    @Override
    public int onStartCommand(Intent intent, int flags,
            int startId) {

       //Сделать работу

        return super.onStartCommand(intent, flags, startId);
    }

    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }
}
```

```
<application
    …
    <service
        android:name=".MyService"
        android:enabled="true"
        android:exported="true" >
    </service>
</application>
```

Restart service

```
public class MyService extends Service {
    @Override
    public int onStartCommand(Intent intent, int flags,
            int startId) {

       //Сделать работу

        return Service.START_STICKY;
    }
}
```

return	Поведение
START_STICKY	Стандартное поведение, сервис запускается при повторном вызове
START_NOT_STICKY	Перезапускается автоматически
START_REDELIVER_INTENT	Перезапускается автоматически, и выполняет **onStartCommand**

Start and stop

```
//Запустить сервис
Intent intent = new Intent(this, MyService.class);
startService(intent);
```

```
//Остановить сервис
Intent intent2 = new Intent(this, MyService.class);
stopService(intent2);
```

```
//Запустить сервис и передать параметры
Intent intent = new Intent(this, MyService.class);

intent.setAction("Action1");  //Действие
intent.putExtra("Test", "Test"); //Параметры

context.startService(intent);
```

Service result (async), Activity

```
Intent intent4 = new Intent(this, MyService3.class);

PendingIntent pendingIntent =
      createPendingResult(1, intent4, 0);
intent4.putExtra("PendingIntent", pendingIntent);

startService(intent4);

@Override
protected void onActivityResult(int requestCode,
      int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    if (resultCode == RESULT_OK) {
        if (requestCode == 1) {

        }
    }
}
```

Service result (async), Service

```
public class MyService3 extends Service {
    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        PendingIntent pendingIntent =
                intent.getParcelableExtra("PendingIntent");

        Intent resultIntent = new Intent();
        resultIntent.putExtra("Result", "Result data");

        try {
            pendingIntent.send(this, Activity.RESULT_OK, resultIntent);
        } catch (PendingIntent.CanceledException e) {
            e.printStackTrace();
        }

        return super.onStartCommand(intent, flags, startId);
    }
}
```

Service binder

```
public class MyService extends Service {

    public void test() {
        Toast.makeText(this, "Test message", Toast.LENGTH_LONG).show();
    }

    @Override
    public IBinder onBind(Intent intent) {
        return new MyBinder();
    }

    //Класс, через который работаем с сервисом
    class MyBinder extends Binder {
        MyService getService() {
            return MyService.this;
        }
    }
}
```

Service sync call

```
ServiceConnection connection = new ServiceConnection() {
    @Override
    public void onServiceConnected(ComponentName componentName,
        IBinder iBinder) {

        //Получаем экземпляр сервиса
        MyService myService = ((MyService.MyBinder) iBinder).getService();
        myService.test();
    }

    @Override
    public void onServiceDisconnected(ComponentName componentName) {

    }
};

Intent intent = new Intent(this, MyService.class);
bindService(intent, connection, BIND_AUTO_CREATE);
```

IntentService

```
public class MyIntentService extends IntentService {

    public MyIntentService() {
        super("MyIntentService");
    }

    @Override
    protected void onHandleIntent(Intent intent) {
        //Сделать работу
    }
}
```

IntentService, best practicies

```
public class MyIntentService extends IntentService {
    private static final String ACTION = ".action.CREATE_STUDENT";
    private static final String FIRST_NAME = ".extra.FIRST_NAME";

    public static void createStudent(Context context, String firstName) {
        Intent intent = new Intent(context, MyIntentService.class);
        intent.setAction(ACTION);
        intent.putExtra(FIRST_NAME, firstName);
        context.startService(intent);
    }

    public MyIntentService() {
        super("MyIntentService");
    }

    @Override
    protected void onHandleIntent(Intent intent) {
        if (intent != null) {
            final String action = intent.getAction();
            if (ACTION.equals(action)) {
                //Сделать работу
            }
        }
    }
}
```

AsyncTask

```
//тип входящих параметров Void
//тип параметров прогресса Void,
//тип результата Void
class MyTask extends AsyncTask<Void, Void, Void> {
  @Override
  protected void onPreExecute() {
    super.onPreExecute();  //Перед запуском, основной поток
  }

  @Override
  protected Void doInBackground(Void... params) {
      //Выполнить работу в фоновом потоке
    return null;
  }

  @Override    //После завершения, основной поток
  protected void onPostExecute(Void result) {
    super.onPostExecute(result);
  }
}
```

AsyncTask

```
class MyTask extends AsyncTask<Void, Void, Void> {
  @Override
  protected void onPreExecute() {
    super.onPreExecute();
  }

  @Override
  protected Void doInBackground(Void... params) {
    publishProgress(); //Сообщить об прогрессе
    return null;
  }

  @Override      //Изменить прогресс, основной процесс
  protected void onProgressUpdate(Void... values) {
    super.onProgressUpdate(values);
  }

  @Override
  protected void onPostExecute(Void result) {
    super.onPostExecute(result);
  }
}
```

Остановка AsyncTask

```
//Запустить AsyncTask
MyTask task = new MyTask();
task.execute();
```

```
//Остановить AsyncTask
task.cancel(true);
```

```
@Override
protected Void doInBackground(Void... params) {
    if(!isCancelled()) {
        //Делать, если не отменен
    }

    return null;
}
```

Спасибо за внимание

Тренер:
Пономарев Алексей
oleksii_android@web-academy.com.ua

WEB ACADEMY
PROGRAMMING COURSES

Лекция 12
Services, AsyncTasks