



*Белорусский государственный университет
информатики и радиоэлектроники*

Эргономика мобильных приложений

Преподаватель: Меженная Марина Михайловна

К.Т.Н., доцент,

доцент кафедры инженерной психологии и эргономики
а 609-2

mezhennaya@bsuir.by



Кафедра инженерной психологии и эргономики

Лекция 12: Сервис (Service)

План лекции:

1. Зачем нужен сервис?
2. Первый вариант сервиса: `MyService extends Service` с ручной организацией потока для работы в фоновом режиме.
3. Второй вариант сервиса: `MyIntentService extends IntentService` с автоматическим выделением потока для работы в фоновом режиме.

Зачем нужны сервисы?

Сервис нужен, чтобы ваша задача продолжала работать, даже когда приложение закрыто.

Сервис не имеет интерфейса!!!

Сервис должен работать в фоне и не использовать UI.

Service прописывается в манифесте!

Зачем нужны сервисы?

Примеры: почтовое приложение, мобильные банкинги, др.

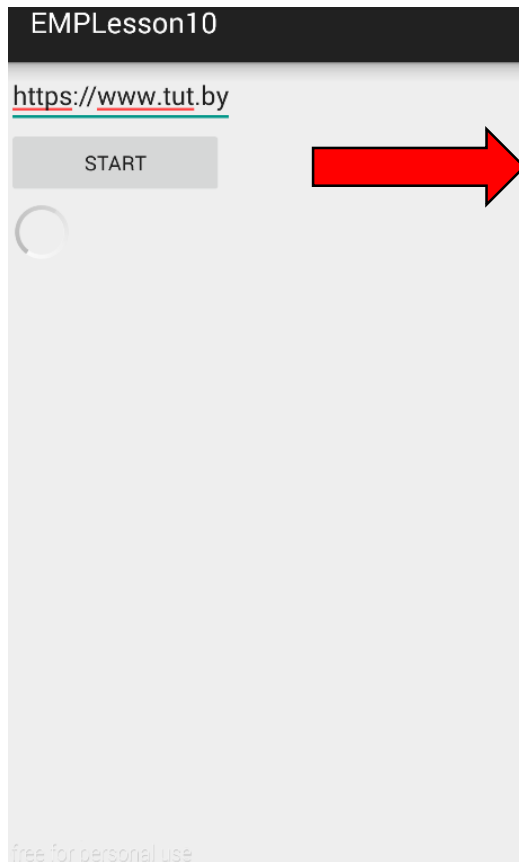
Алгоритм работы почтовой программы: она состоит из приложения и сервиса.

Сервис работает в фоне и периодически проверяет наличие новой почты, скачивает ее и выводит уведомления.

А когда вы запускаете приложение, оно отображает вам эти загруженные сервисом письма.

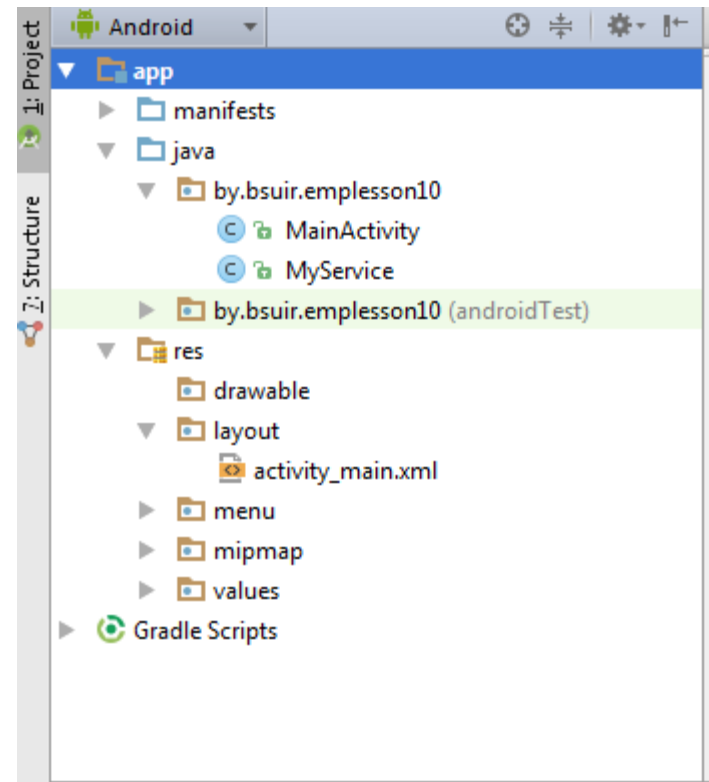
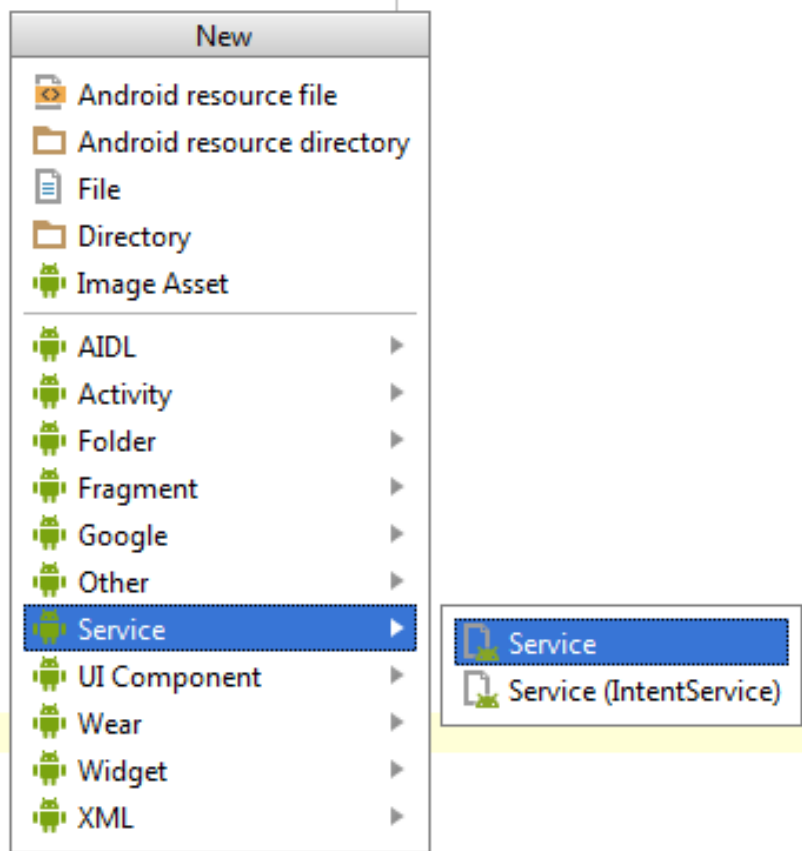
Также приложение может подключиться к сервису и поменять в нем, например, период проверки почты или совсем закрыть сервис, если постоянная проверка почты больше не нужна.

Первый вариант сервиса: MyService extends Service с ручной организацией потока для работы в фоновом режиме



```
04-13 03:05:16.397    2088-2088/by.bsuir.emplesson10 D/myLogs: onCreate
04-13 03:05:16.397    2088-2088/by.bsuir.emplesson10 D/myLogs: onStartCommand
04-13 03:05:16.397    2088-2088/by.bsuir.emplesson10 D/myLogs: https://www.tut.by
04-13 03:05:20.029    2088-2138/by.bsuir.emplesson10 D/myLogs: <!DOCTYPE html>
    <html class="no-js smart" lang="ru"><head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
    <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1"/>
    <title>Белорусский портал TUT.BY</title>
    <meta name="referrer" content="always" />
    <meta name="HandheldFriendly" content="True"/>
    <meta name="MobileOptimized" content="320"/>
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
    <meta name="keywords" content="" />
    <meta name="description" content="Последние новости Беларуси и зарубежья. Быстрый поиск. На
    <meta name="google-site-verification" content="f-6GnRRA7oTzx-aPcIQ546aRHeu2ynaXw64h6PhlgY8'
    <meta name="yandex-verification" content="72b4676d68aa83ff"/>
    <meta property="og:title" content="Белорусский портал TUT.BY"/>
    <meta property="og:url" content="https://smart.tut.by/" />
    <meta property="og:description" content="Последние новости Беларуси и зарубежья. Быстрый п
    <meta property="og:image" content="https://img.tyt.by/titul_by2/logo_ru3.gif"/>
    <meta property="vk:title" content="Белорусский портал TUT.BY"/>
    <meta property="vk:url" content="https://smart.tut.by/" />
    <meta property="vk:image" content="https://img.tyt.by/titul_by2/logo_ru3.gif"/>
    <meta property="vk:description" content="Последние новости Беларуси и зарубежья. Быстрый п
    ...
04-13 03:05:20.189    2088-2088/by.bsuir.emplesson10 D/myLogs: onDestroy
```

Первый вариант сервиса: MyService extends Service с ручной организацией потока для работы в фоновом режиме



AndroidManifest.xml

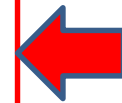
```
<?xml version="1.0" encoding="utf-8"?>
```

```
<manifest
```

```
xmlns:android="http://schemas.android.com/apk/res/android"
```

```
package="by.bsuir.emplesson10" >
```

```
    <uses-permission  
        android:name="android.permission.INTERNET"/>
```

 Для доступа
к интернету

```
    <application
```

```
        android:allowBackup="true"
```

```
        android:icon="@mipmap/ic_launcher"
```

```
        android:label="@string/app_name"
```

```
        android:theme="@style/AppTheme" >
```

AndroidManifest.xml

```
<activity
    android:name=".MainActivity"
    android:label="@string/app_name" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category
            android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <service
        android:name=".MyService"
        android:enabled="true"
        android:exported="true" >
    </service>
</application>

</manifest>
```


MainActivity.java

```
public class MainActivity extends ActionBarActivity implements
View.OnClickListener{

    private Button btnStart, btnStop;
    private TextView tvRequest;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        initView();
    }
}
```

MainActivity.java

```
private void initView() {  
    btnStart = (Button) findViewById(R.id.btnStart);  
    btnStart.setOnClickListener(this);  
    tvRequest = (TextView) findViewById(R.id.tvRequest);  
}  
  
@Override  
public void onClick(View view) {  
    String request = tvRequest.getText().toString();  
    startService(new Intent(this, MyService.class).putExtra("request",  
request));  
}  
  
}
```

MyService.java

```
public class MyService extends Service {
```

```
    final String LOG_TAG = "myLogs";
```

```
    @Override
```

```
    public void onCreate() {
```

```
        super.onCreate();
```

```
        Log.d(LOG_TAG, "onCreate");
```

```
    }
```

```
    @Override
```

```
    public int onStartCommand(Intent intent, int flags, int startId) {
```

```
        Log.d(LOG_TAG, "onStartCommand");
```

```
        String request = intent.getStringExtra("request");
```

```
        Log.d(LOG_TAG, request);
```

```
        someTask();
```

```
        return START_STICKY;
```

```
    }
```

Метод onStartCommand
срабатывает, когда сервис
запущен методом startService

MyService.java

@Override

```
public void onDestroy() {  
    super.onDestroy();  
    Log.d(LOG_TAG, "onDestroy");  
}
```

@Override

```
public IBinder onBind(Intent intent) {  
    // TODO: Return the communication channel to the service.  
    throw new UnsupportedOperationException("Not yet  
implemented");  
}
```

MyService.java

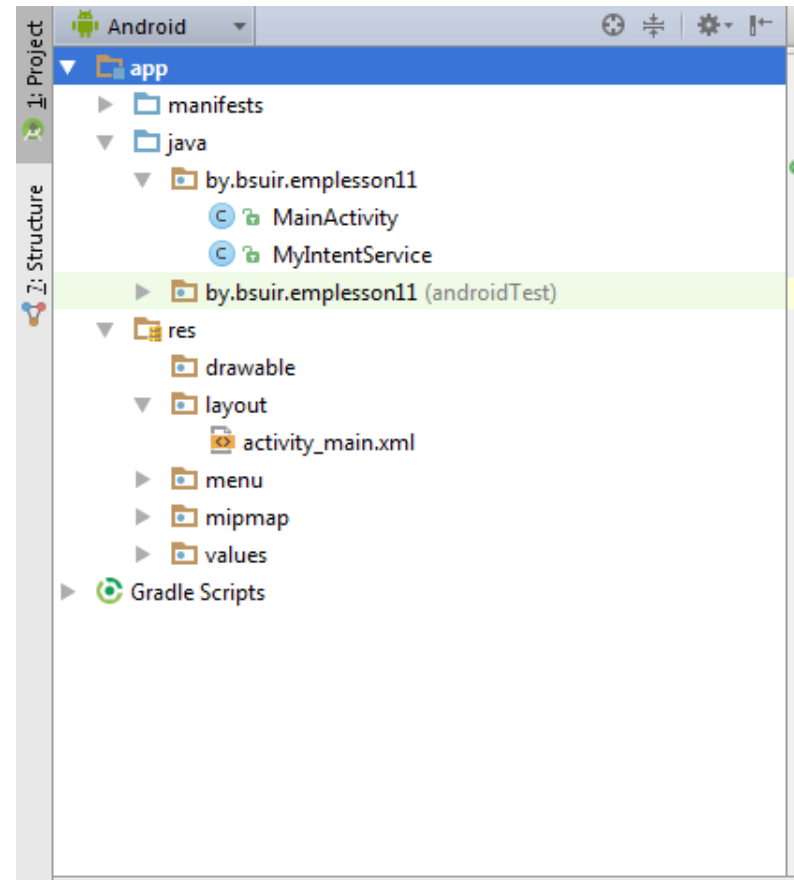
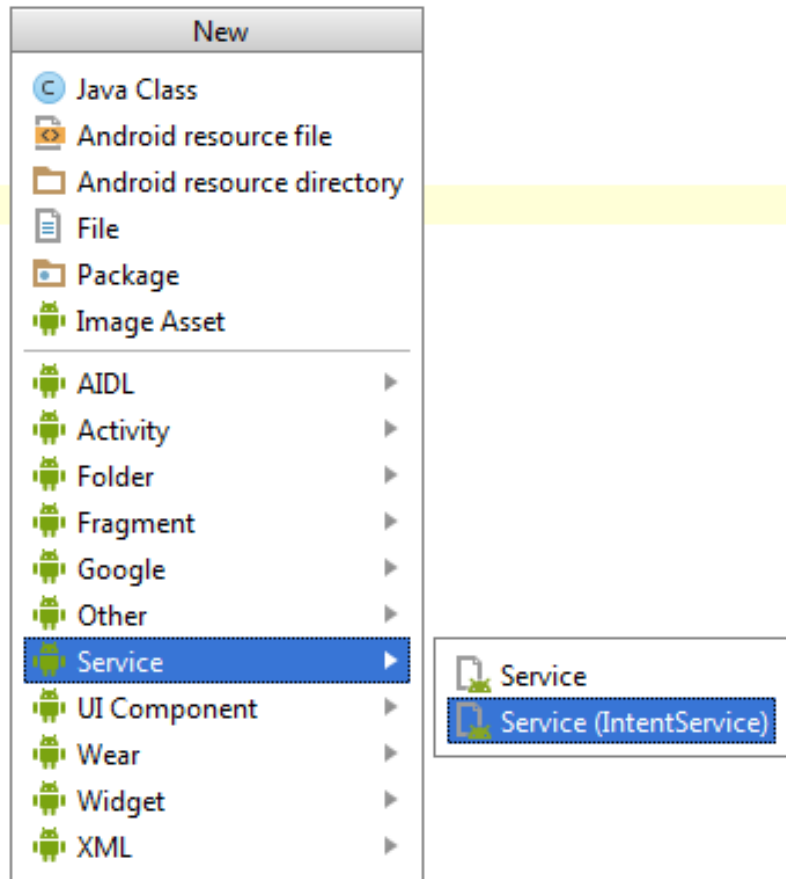
```
void someTask() {  
    new Thread(new Runnable() {  
        public void run() {  
            String path = "http://www.tut.by";  
            BufferedReader reader=null;  
            HttpURLConnection urlConnection=null;  
            try {  
                URL url= null;  
                url = new URL(path);  
                urlConnection= (HttpURLConnection)url.openConnection();  
                reader=new BufferedReader(new InputStreamReader(urlConnection.getInputStream()))  
                StringBuilder buf=new StringBuilder();  
                String line=null;  
                while ((line=reader.readLine()) != null) {  
                    buf.append(line + "\n");  
                }  
                Log.d(LOG_TAG, buf.toString());  
            }catch (Exception e) {  
                e.printStackTrace();  
            }  
        }  
    }  
}
```

MyService.java

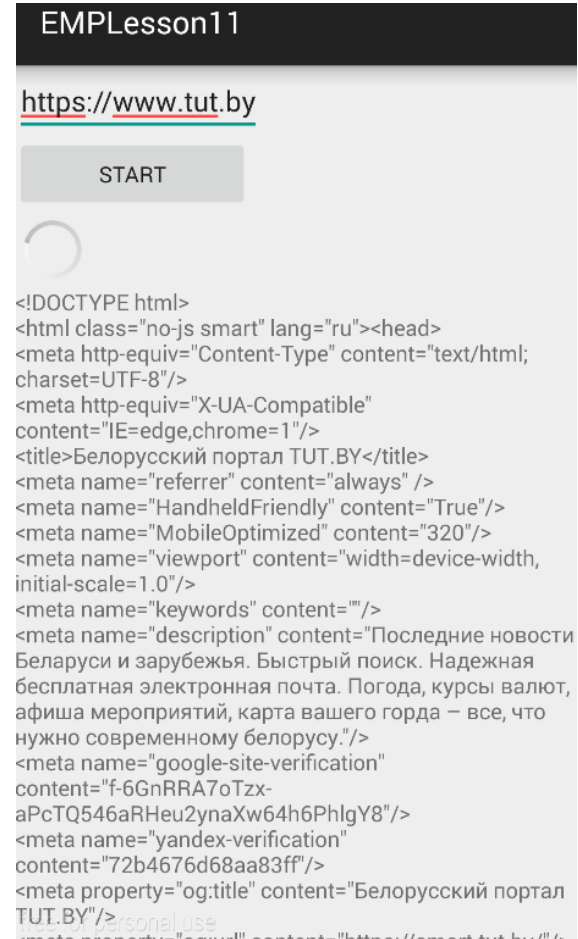
```
finally {  
    urlConnection.disconnect();  
    if (reader != null) {  
        try {  
            reader.close();  
        } catch (IOException e) {  
            e.printStackTrace();  
        }  
    }  
    }  
    stopSelf();  
}  
}).start();  
  
}  
  
}
```

stopSelf – метод, аналогичный методу **stopService**, он останавливает сервис, в котором был вызван

Второй вариант сервиса: MyIntentService extends IntentService с автоматическим выделением потока для работы в фоновом режиме.



Второй вариант сервиса: MyIntentService extends IntentService с автоматическим выделением потока для работы в фоновом режиме.



Второй вариант сервиса: MyIntentService extends IntentService с автоматическим выделением потока для работы в фоновом режиме.

Для получения обратной связи от сервиса используем **BroadcastReceiver**:

1. В Activity создаем BroadcastReceiver, а также создаем IntentFilter, настроенный на определенный Action, и регистрируем (включаем) эту пару.

Теперь BroadcastReceiver будет получать Intent-ы, подходящие под условия IntentFilter.

2. В сервисе, когда нам понадобится передать данные в Activity, мы создаем Intent (с Action из предыдущего пункта), кладем в него данные, которые хотим передать, и посылаем его на поиски BroadcastReceiver.

Второй вариант сервиса: MyIntentService extends IntentService с автоматическим выделением потока для работы в фоновом режиме.

Для получения обратной связи от сервиса используем **BroadcastReceiver**:

3. BroadcastReceiver в Activity ловит этот Intent и извлекает из него данные.

Т.е. тут все аналогично вызовам Activity с использованием Action и IntentFilter. Если Action в Intent (отправленном из сервиса) и в IntentFilter (у BroadcastReceiver в Activity) совпадут, то BroadcastReceiver получит этот Intent и сможет извлечь данные для Activity.

AndroidManifest.xml

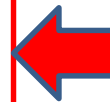
```
<?xml version="1.0" encoding="utf-8"?>
```

```
<manifest
```

```
xmlns:android="http://schemas.android.com/apk/res/android"
```

```
package="by.bsuir.emplesson11" >
```

```
    <uses-permission  
        android:name="android.permission.INTERNET" />
```



Для доступа
к интернету

```
    <application
```

```
        android:allowBackup="true"
```

```
        android:icon="@mipmap/ic_launcher"
```

```
        android:label="@string/app_name"
```

```
        android:theme="@style/AppTheme" >
```

AndroidManifest.xml

```
<activity
    android:name=".MainActivity"
    android:label="@string/app_name" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category
            android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

```
<service
    android:name=".MyIntentService"
    android:exported="false" >
</service>
```

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

MainActivity.java

```
public class MainActivity extends ActionBarActivity implements
View.OnClickListener{

    private Button btnStart, btnStop;
    private TextView tvRequest, tvResult;
    private MyBroadcastReceiver myBroadcastReceiver;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        initView();
    }
    private void initView() {
        btnStart = (Button) findViewById(R.id.btnStart);
        btnStart.setOnClickListener(this);
        tvRequest = (TextView) findViewById(R.id.tvRequest);
        tvResult = (TextView) findViewById(R.id.tvResult);
    }
}
```

MainActivity.java

@Override

```
public void onClick(View view) {  
    String request = tvRequest.getText().toString();  
    startService(new Intent(this, MyIntentService.class).putExtra("request",  
request));
```

```
    myBroadcastReceiver = new MyBroadcastReceiver();  
    // создаем фильтр для BroadcastReceiver  
    IntentFilter intentFilter = new IntentFilter(  
        MyIntentService.ACTION_MYINTENTSERVICE);  
    intentFilter.addCategory(Intent.CATEGORY_DEFAULT);  
    // регистрируем BroadcastReceiver  
    registerReceiver(myBroadcastReceiver, intentFilter);  
}
```

@Override

```
protected void onDestroy() {  
    super.onDestroy();  
    unregisterReceiver(myBroadcastReceiver);  
}
```

MainActivity.java

```
public class MyBroadcastReceiver extends BroadcastReceiver {  
  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        String result = intent.getStringExtra(MyIntentService.RESULT);  
        tvResult.setText(result);  
    }  
}  
  
}
```

MyIntentService.java

```
public class MyIntentService extends IntentService {  
  
    final String LOG_TAG = "myLogs";  
    public static final String ACTION_MYINTENTSERVICE =  
"android.intent.action.intentservice.RESPONSE";  
    public static final String RESULT = "RESULT";  
  
    public MyIntentService() {  
        super("MyIntentService");  
    }  
  
    public void onCreate() {  
        super.onCreate();  
        Log.d(LOG_TAG, "onCreate");  
    }  
}
```


MyIntentService.java

```
protected void onHandleIntent(Intent intent) {

    String request = intent.getStringExtra("request");
    Log.d(LOG_TAG, request);
    String path = "https://www.tut.by";
    BufferedReader reader=null;
    HttpURLConnection urlConnection=null;
    try {
        URL url= null;
        url = new URL(path);
        urlConnection= (HttpURLConnection)url.openConnection();
        reader=      new      BufferedReader(new      InputStreamReader(
urlConnection.getInputStream()));
        StringBuilder buf=new StringBuilder();
        String line=null;
        while ((line=reader.readLine()) != null) {
            buf.append(line + "\n");
        }
        Log.d(LOG_TAG, buf.toString());
    }
```

MyIntentService.java

// возвращаем результат

```
Intent responseIntent = new Intent();
```

```
responseIntent.setAction(ACTION_MYINTENTSERVICE);
```

```
responseIntent.addCategory(Intent.CATEGORY_DEFAULT);
```

```
responseIntent.putExtra(RESULT, buf.toString());
```

```
sendBroadcast(responseIntent);
```

```
} catch (Exception e) {
```

```
    e.printStackTrace();
```

```
}
```

```
finally {
```

```
    urlConnection.disconnect();
```

```
    if (reader != null) {
```

```
        try {
```

```
            reader.close();
```

```
        } catch (IOException e) {
```

```
            e.printStackTrace();
```

```
        }
```

```
    }
```

```
}
```

```
stopSelf();
```

```
}
```

```
}
```