SmsSender

SmsSender_Service_SendSms

Create New Project

Create Android Project

Application name

SmsSender

Company domain

htl.at

Project location

/htl/skripten/themen/android.kotlin/presentations.1718/06_service/SmsSender_Service_SendSms/SmsSender ...

Package name

at.htl.smssender Edit

☐ **Include C++ support**

☑ **Include Kotlin support**

Cancel Previous Next Finish

Create New Project

Target Android Devices

Select the form factors and minimum SDK

Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.

- [x] **Phone and Tablet**

  API 23: Android 6.0 (Marshmallow)

  By targeting **API 23 and later**, your app will run on approximately **39.3%** of devices. Help me choose

  - [ ] Include Android Instant App support

- [ ] **Wear OS**

  API 23: Android 6.0 (Marshmallow)

- [ ] **TV**

  API 21: Android 5.0 (Lollipop)

- [ ] **Android Auto**

- [ ] **Android Things**

  API 24: Android 7.0 (Nougat)

Cancel | Previous | Next | Finish

Create New Project

Add an Activity to Mobile

Add No Activity

Activity & Fragment+ViewModel

Basic Activity

Bottom Navigation Activity

Empty Activity

Fullscreen Activity

Google AdMob Ads Activity

Google Maps Activity

Cancel | Previous | Next | Finish

Create New Project

Con ig re Activity

Creates a new empty activity

Activity Name: MainActivity

☑ Generate Layout File

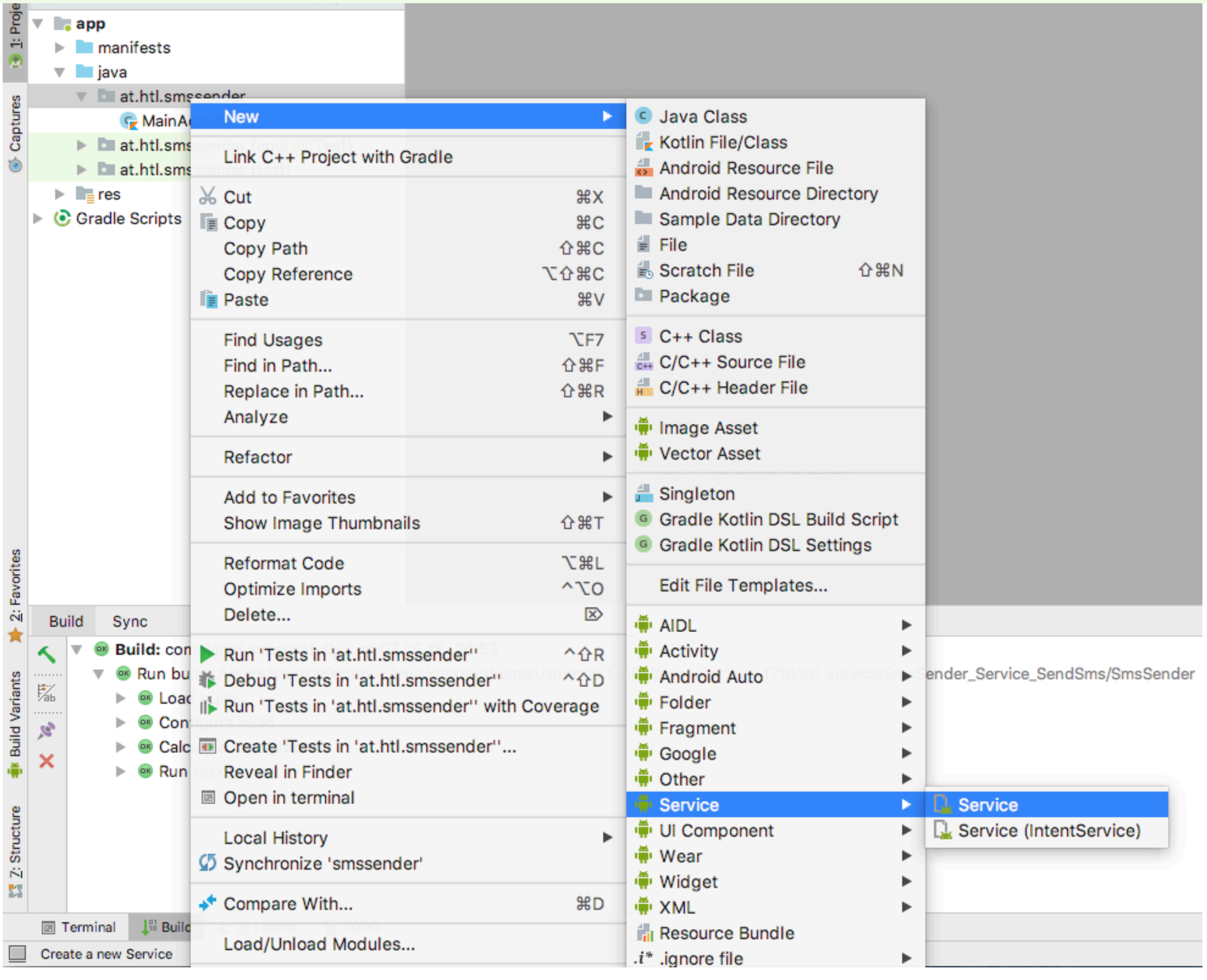
Layout Name: activity_main

☑ Backwards Compatibility (AppCompat)

The name of the activity class to create

Cancel | Previous | Next | Finish

SmsSenderService



SmsSenderService

New Android Component

Configur Compon nt

Android Studio

Creates a new service component and adds it to your Android manifest.

Class Name: SmsSenderService

Exported

Enabled

Source Language: Kotlin

Whether or not components of other applications can invoke the service or interact with it

Cancel Previous Next Finish

SmsSenderService

wird als innere Klasse am Ende implementiert

```
class SmsSenderService : Service() {

    private val LOG_TAG = SmsSenderService::class.java.simpleName
    private val smsSenderServiceBinder = SmsSenderServiceBinder()

    // Systemmeldungen über den Versand der SMS per BroadcastReceiver empfangen
    internal var smsSentReceiver: BroadcastReceiver? = null
    internal var smsDeliveredReceiver: BroadcastReceiver? = null
    private var activityNotificationHandler: Handler? = null
    fun setNotificationHandler(handler : Handler){
        activityNotificationHandler = handler
    }

    override fun onUnbind(intent: Intent): Boolean {...}


    override fun onBind(intent: Intent):IBinder?{...}

    override fun onCreate() {...}
    override fun onDestroy() {...}

    /**
     * BroadcastReceiver für die PendingIntents des Empfangs
     * der Versende- und Zustellungsmeldung
     */
    private fun createBroadcastReceivers() {...}

    /**
     * Sms per SmsManager verschicken. Für die Actions SMS_SENT und
     * SMS_DELIVERED werden Broadcastreceiver registriert.
     * @param phoneNo Telefonnummer
     * @param msg     Nachricht
     */
    fun sendSMS(phoneNo: String, msg: String) {...}

    /**
     * Statusmeldungen über das Versenden der SMS an die Activity
     * zurückmelden
     *
     * @param notificationText Text
     */
    fun sendNotificationToActivity(notificationText: String) {...}

    /**
     * Binder implementiert die Schnittstellenmethoden des
     * Service (quasi Contract).
     * Nutzmethoden im Service bilden Wrapper für Servicemethoden.
     * Oft wird z.B. der Context benötigt, der im Binder nicht verfügbar ist
     */
    inner class SmsSenderServiceBinder : Binder() {...}
```

SmsSenderService

```
override fun onUnbind(intent: Intent): Boolean {
    Toast.makeText(this, "SmsSenderService onUnbind()", Toast.LENGTH_SHORT).show()
    Log.d(LOG_TAG, "SmsSenderService onUnbind()")
    activityNotificationHandler = null
    return super.onUnbind(intent)
}

override fun onBind(intent: Intent): IBinder {
    Toast.makeText(this, "SmsSenderService onBind()", Toast.LENGTH_SHORT).show()
    Log.d(LOG_TAG, "onBind()")
    return smsSenderServiceBinder
}
```

Die Methode onBind(…) gibt das Binder-Objekt zurück

```
override fun onCreate() {
    Toast.makeText(this, "SmsSenderService onCreate()", Toast.LENGTH_SHORT).show()
    Log.d(LOG_TAG, "onCreate()")
    super.onCreate()
    createBroadcastReceivers()
    // Beide Broadcastreceiver für die pending intents (sent, delivered) registrieren
    registerReceiver(smsSentReceiver, IntentFilter("SMS_SENT"))
    registerReceiver(smsDeliveredReceiver, IntentFilter("SMS_DELIVERED"))
}

override fun onDestroy() {
    Log.d(LOG_TAG, "onDestroy()")
    Toast.makeText(this, "SmsSenderService onDestroy()", Toast.LENGTH_SHORT).show()
    unregisterReceiver(smsSentReceiver)
    unregisterReceiver(smsDeliveredReceiver)
    super.onDestroy()
}
```

SmsSenderService

```
/**
 * BroadcastReceiver für die PendingIntents des Empfangs
 * der Versende- und Zustellungsmeldung
 */
private fun createBroadcastReceivers() {
    // Direkte Instanzierung zweier BroadcastReceiver
    smsSentReceiver = object : BroadcastReceiver() {
        override fun onReceive(context: Context, intent: Intent) {
            var logText =
                    when (resultCode) {
                        Activity.RESULT_OK -> "SMS has been sent"
                        SmsManager.RESULT_ERROR_GENERIC_FAILURE -> "Generic Failure"
                        SmsManager.RESULT_ERROR_NO_SERVICE -> "No Service"
                        SmsManager.RESULT_ERROR_NULL_PDU -> "Null PDU"
                        SmsManager.RESULT_ERROR_RADIO_OFF -> "Radio Off"
                        else -> "Unknown error"
                    }
            Toast.makeText(baseContext, logText, Toast.LENGTH_SHORT).show()
            sendNotificationToActivity("SENT $logText")
            Log.d(LOG_TAG, "Broadcast smsSentReceiver")
        }
    }
    // Direkte Instanzierung ohne Ableitung
    smsDeliveredReceiver = object : BroadcastReceiver() {
        override fun onReceive(context: Context, intent: Intent) {
            var logText = "Illegal"
            when (resultCode) {
                Activity.RESULT_OK -> logText = "SMS delivered"
                Activity.RESULT_CANCELED -> logText = "SMS not delivered"
            }
            Toast.makeText(baseContext, logText, Toast.LENGTH_SHORT).show()
            sendNotificationToActivity("DELIVERED $logText")
            Log.d(LOG_TAG, "Broadcast smsDeliveredReceiver")
        }
    }
}
```

SmsSenderService

```
/**
 * Sms per SmsManager verschicken. Für die Actions SMS_SENT und
 * SMS_DELIVERED werden Broadcastreceiver registriert.
 * @param phoneNo Telefonnummer
 * @param msg     Nachricht
 */
fun sendSMS(phoneNo: String, msg: String) {
    Log.d(LOG_TAG, "_sendSMS($phoneNo,$msg)")
    val smsManager = SmsManager.getDefault()
    val pendingIntentSent = PendingIntent.getBroadcast(applicationContext, 0, Intent("SMS_SENT"), 0)
    val pendingIntentDelivered = PendingIntent.getBroadcast(applicationContext, 0, Intent("SMS_DELIVERED"), 0)
    smsManager.sendTextMessage(phoneNo, null, msg, pendingIntentSent, pendingIntentDelivered)
}

/**
 * Statusmeldungen über das Versenden der SMS an die Activity
 * zurückmelden
 *
 * @param notificationText Text
 */
fun sendNotificationToActivity(notificationText: String) {
    val message = Message()
    val bundle = Bundle()
    bundle.putString("LogString", notificationText)
    message.data = bundle
    activityNotificationHandler?.sendMessage(message)
}

/**
 * Binder implementiert die Schnittstellenmethoden des
 * Service (quasi Contract).
 * Nutzmethoden im Service bilden Wrapper für Servicemethoden.
 * Oft wird z.B. der Context benötigt, der im Binder nicht verfügbar ist
 */
inner class SmsSenderServiceBinder : Binder() {

    fun getService() : SmsSenderService{
        return this@SmsSenderService
    }

}
```

Binder implementiert IBinder

Hier wird eine Referenz auf den Service (this) an den Client zurückgegeben, damit der Client mit dem Service kommunizieren kann,

Ask

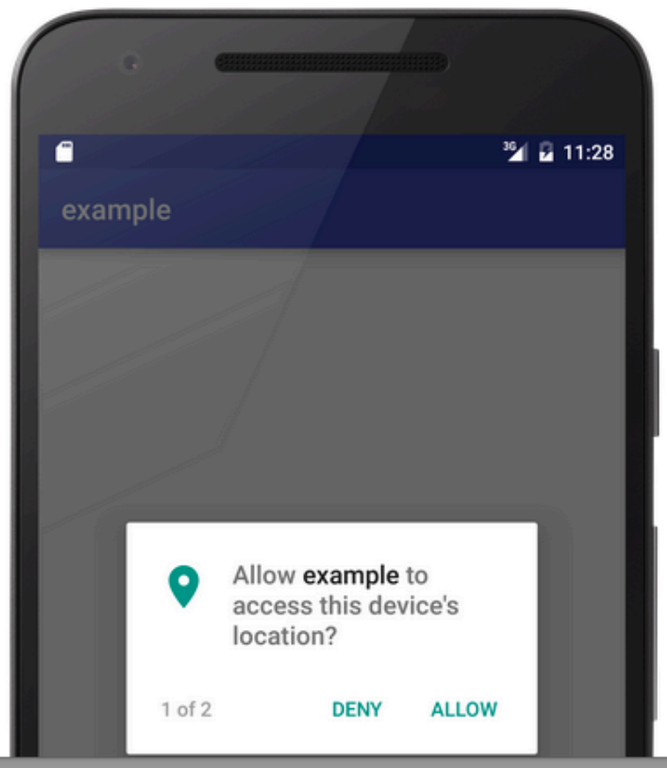
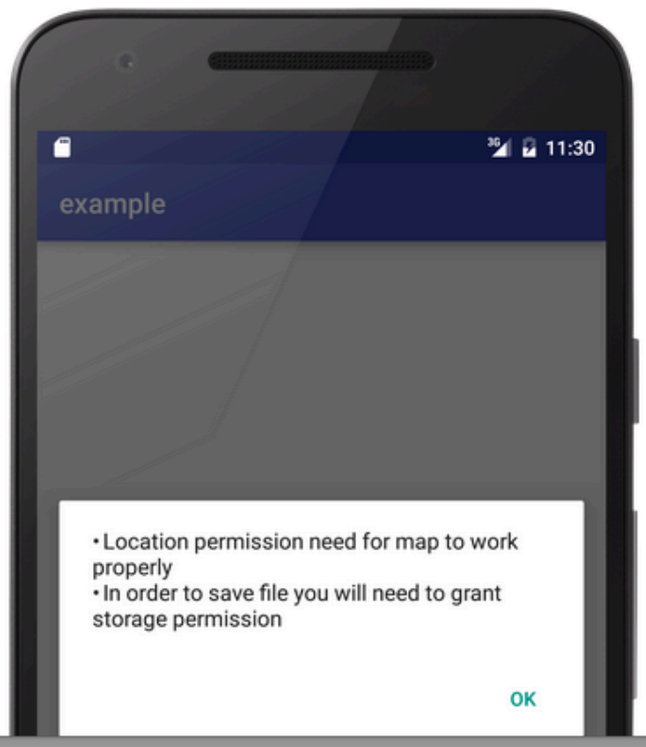
Android runtime permissions make easy

Why?

In marshmallow, android has introduced runtime permission check, that means when application runs, it asks user provide permission to run specific functionality. for example if you using Camera or saving files on external storage or location.

The android basic code to request permission is to complex and tedious to understand (if you don't trust me, check here 😃). **Ask** is a library make asking for the particular permission easy for developer. This is very simple and light weight library with just few lines of code and you good to go.

Demo(How it looks!)

Ask for permission	Show rationale
	

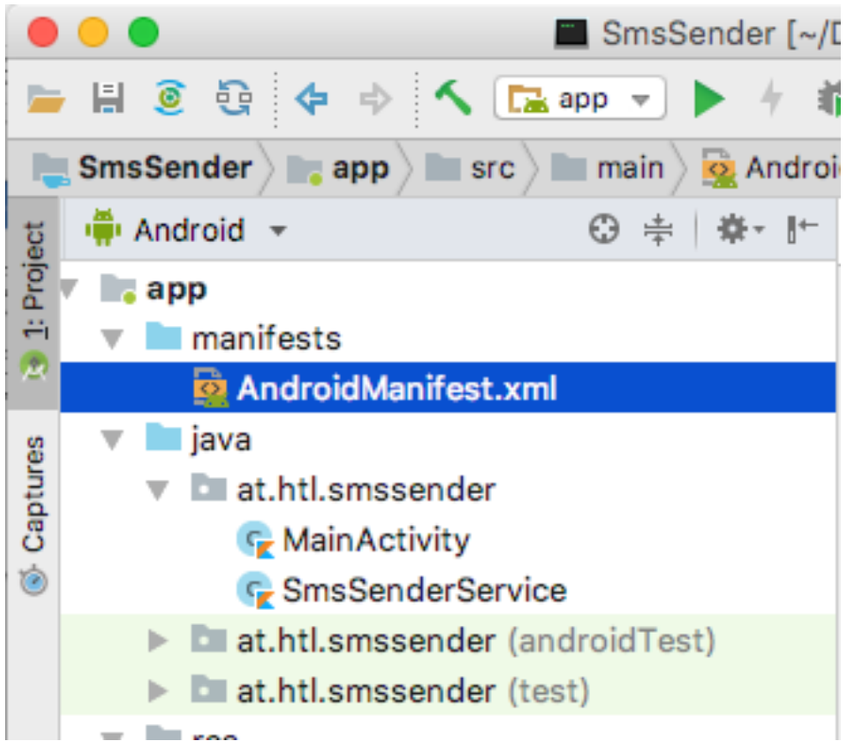
https://github.com/00ec454/Ask

build.gradle (Module: app)

app

```
apply plugin: 'com.android.application'

apply plugin: 'kotlin-android'

apply plugin: 'kotlin-android-extensions'

android {
    compileSdkVersion 27
    defaultConfig {
        applicationId "at.htl.smssender"
        minSdkVersion 23
        targetSdkVersion 27
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation"org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation 'com.android.support:appcompat-v7:27.1.1'
    implementation 'com.android.support.constraint:constraint-layout:1.1.1'
    implementation 'com.android.support:design:27.1.1'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.2'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
    implementation 'com.vistrav:ask:2.5'
}
```

AndroidManifest.xml



AndroidManifest.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="at.htl.smssender">
4
5      <uses-permission android:name="android.permission.SEND_SMS"/>
6
7      <application
8          android:allowBackup="true"
9          android:icon="@mipmap/ic_launcher"
10         android:label="SmsSender"
11         android:roundIcon="@mipmap/ic_launcher_round"
12         android:supportsRtl="true"
13         android:theme="@style/AppTheme">
14         <activity
15             android:name=".MainActivity"
16             android:label="@string/app_name"
17             android:theme="@style/AppTheme.NoActionBar">
18             <intent-filter>
19                 <action android:name="android.intent.action.MAIN" />
20
21                 <category android:name="android.intent.category.LAUNCHER" />
22             </intent-filter>
23         </activity>
24
25         <service
26             android:name=".SmsSenderService"
27             android:enabled="true"
28             android:exported="false">
29         </service>
30     </application>
```

```
2018-06-07 23:47:41.063 7139-7139/at.htl.smssender E/AndroidRuntime: FATAL
 EXCEPTION: main
    Process: at.htl.smssender, PID: 7139
    java.lang.RuntimeException: Unable to start activity ComponentInfo{at.htl
 .smssender/at.htl.smssender.MainActivity}: java.lang.IllegalStateException:
 This Activity already has an action bar supplied by the window decor. Do not
 request Window.FEATURE_SUPPORT_ACTION_BAR and set windowActionBar to false in
 your theme to use a Toolbar instead.
```

MainActivity

```
class MainActivity : AppCompatActivity() {

    private val LOG_TAG = MainActivity::class.java.simpleName

    internal var logLines: MutableList<String> = mutableListOf()

    /**
     * Handler wird vom Service über Versand und Zustellung der SMS
     * verständigt. Wird über Looper in Protokoll-TextView ausgegeben
     */
    private val logFromServicesNotificationHandler = object : Handler() {...}

    internal var smsSenderService: SmsSenderService? = null
    // ServiceConnection repräsentiert die Verbindung zum Service
    private val smsSenderServiceConnection = object : ServiceConnection {...}

    override fun onCreate(savedInstanceState: Bundle?) {...}

    /**
     * Service starten
     */
    override public fun onResume() {...}
    override public fun onPause() {
        Log.d(LOG_TAG, "onPause() SmsService beendet")
        unbindService(smsSenderServiceConnection)
        super.onPause()
    }

    override fun onCreateOptionsMenu(menu: Menu): Boolean {...}

    override fun onOptionsItemSelected(item: MenuItem): Boolean {...}

    public override fun onDestroy() {...}

    //optional
    @AskGranted(Manifest.permission.SEND_SMS)
    fun smsSendGranted() {...}

    //optional
    @AskDenied(Manifest.permission.WRITE_EXTERNAL_STORAGE)
    fun smsSendDenied() {...}

    /**
     * Neue LogMessage oben in Protokollview hinzufügen
     * Maximal 8 Einträge anzeigen
     *
     * @param logMessage Auszugebende Zeile
     */
    private fun addToLogView(logMessage: String) {...}
}
```

MainActivity

```
package at.htl.smssender

import android.Manifest
import android.content.ComponentName
import android.content.Context
import android.content.Intent
import android.content.ServiceConnection
import android.os.Bundle
import android.os.Handler
import android.os.IBinder
import android.os.Message
import android.support.design.widget.Snackbar
import android.support.v7.app.AppCompatActivity
import android.support.v7.widget.Toolbar
import android.util.Log
import android.view.Menu
import android.view.MenuItem
import android.view.View
import android.widget.Toast
import com.vistrav.ask.Ask
import com.vistrav.ask.annotations.AskDenied
import com.vistrav.ask.annotations.AskGranted
import kotlinx.android.synthetic.main.activity_main.*
import kotlinx.android.synthetic.main.content_main.*
import java.text.SimpleDateFormat
import java.util.*
```

MainActivity

```
class MainActivity : AppCompatActivity() {

    private val LOG_TAG = MainActivity::class.java.simpleName

    internal var logLines: MutableList<String> = mutableListOf()

    /**
     * Handler wird vom Service über Versand und Zustellung der SMS
     * verständigt. Wird über Looper in Protokoll-TextView ausgegeben
     */
    private val logFromServicesNotificationHandler = object : Handler() {
        override fun handleMessage(msg: Message) {
            Log.d(LOG_TAG, "handleMessage()")
            val bundle = msg.data
            val logString = bundle.getString("LogString")
            addToLogView(logString)
            Toast.makeText(applicationContext, "Handler, handleMessage: " + logString!!,
                    Toast.LENGTH_SHORT).show()
            super.handleMessage(msg)
        }
    }

    internal var smsSenderService: SmsSenderService? = null
    // ServiceConnection repräsentiert die Verbindung zum Service
    private val smsSenderServiceConnection = object : ServiceConnection {
        override fun onServiceConnected(name: ComponentName, binder: IBinder) {
            Toast.makeText(applicationContext, "ServiceConnection, onServiceConnected",
                    Toast.LENGTH_SHORT).show()
            val smsSenderServiceBinder = binder
                    as SmsSenderService.SmsSenderServiceBinder
            smsSenderServiceBinder?.getService()
                    .setNotificationHandler(logFromServicesNotificationHandler)
            smsSenderService = smsSenderServiceBinder?.getService()
        }
        override fun onServiceDisconnected(name: ComponentName) {

            Toast.makeText(applicationContext, "ServiceConnection, onServiceDisconnected", Toast.LENGTH_SHORT).show()
            smsSenderService = null
        }
    }
```

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    val toolbar = findViewById(R.id.toolbar)     as Toolbar
    setSupportActionBar(toolbar)
    fab.setOnClickListener(View.OnClickListener { view ->
        val phoneNo = et_phone_number.text.toString()
        val text = et_text.text.toString()
        val isPhoneNumberOk = if (phoneNo.length == 0) {
            til_phoneNumber.error = "Phonenumber cannot be blank"
            false
        } else {
            til_phoneNumber.isErrorEnabled = false
            true
        }
        val isTextOk = if (text.length == 0) {
            til_text.error = "Sms-Text cannot be blank"
            false
        } else {
            til_text.isErrorEnabled = false
            true
        }
        if (isPhoneNumberOk && isTextOk){
            smsSenderService?.sendSMS(phoneNo, text)
            Snackbar.make(view, "SMS sent", Snackbar.LENGTH_LONG).show()
        }
    })
    Ask.on(this)
            .forPermissions(Manifest.permission.SEND_SMS)
            .withRationales("SEND-SMS permission need")
            .go()
}
```

```
/**
 * Service starten
 */
override public fun onResume() {
    super.onResume()
    Log.d(LOG_TAG, "onResume() SmsService gebunden")
    val smsServiceIntent = Intent(this, SmsSenderService::class.java)
    bindService(smsServiceIntent, smsSenderServiceConnection, Context.BIND_AUTO_CREATE)
}
override public fun onPause() {
    Log.d(LOG_TAG, "onPause() SmsService beendet")
    unbindService(smsSenderServiceConnection)
    super.onPause()
}

override fun onCreateOptionsMenu(menu: Menu): Boolean {
    // Inflate the menu; this adds items to the action bar if it is present.
    menuInflater.inflate(R.menu.menu_main, menu)
    return true
}

override fun onOptionsItemSelected(item: MenuItem): Boolean {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    val id = item.itemId
    return if (id == R.id.action_settings) {
        true
    } else super.onOptionsItemSelected(item)

}

public override fun onDestroy() {
    Log.d(LOG_TAG, "onDestroy()")
    super.onDestroy()
}
```

```
    //optional
    @AskGranted(Manifest.permission.SEND_SMS)
    fun smsSendGranted() {
        Log.d(LOG_TAG, "SEND_SMS  GRANTED")
    }

    //optional
    @AskDenied(Manifest.permission.WRITE_EXTERNAL_STORAGE)
    fun smsSendDenied() {
        Log.d(LOG_TAG, "SEND_SMS  DENIED")
    }

    /**
     * Neue LogMessage oben in Protokollview hinzufügen
     * Maximal 8 Einträge anzeigen
     *
     * @param logMessage Auszugebende Zeile
     */
    private fun addToLogView(logMessage: String) {
        Log.d(LOG_TAG, "addToLogView()")
        val sdf = SimpleDateFormat("HH:mm:ss")
        val line = sdf.format(Date()) + ": " + logMessage
        logLines.add(0, line)
        if (logLines.size > 8) {
            logLines.removeAt(7)
        }
        val stringBuilder = StringBuilder()
        for (i in logLines.indices) {
            stringBuilder.append(logLines.get(i))
            if (i + 1 < logLines.size) {
                stringBuilder.append("\n")
            }
        }
        tv_log_lines.setText(stringBuilder.toString())
    }
}
```

strings.xml

strings.xml

Edit translations for all locales in the translations editor.

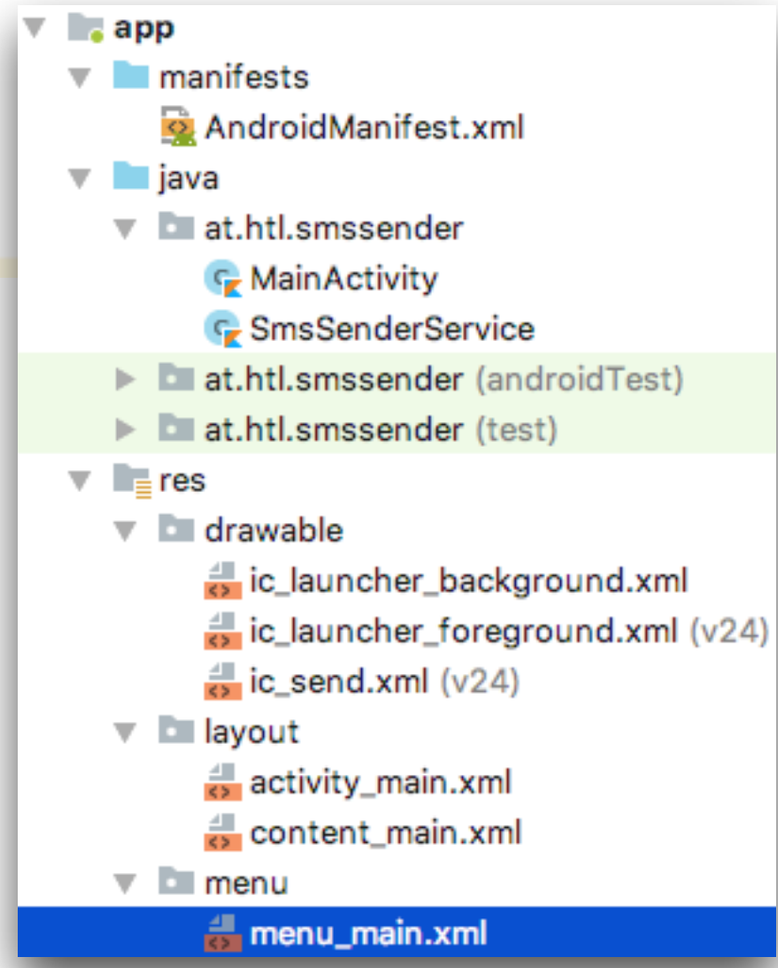
```
1   <resources>
2       <string name="app_name">SmsSender</string>
3       <string name="action_settings">Settings</string>
4       <string name="number_validation_msg">Phonenumber cannot be blank</string>
5       <string name="text_validation_msg">Sms-Text cannot be blank</string>
6       <string name="sent_success">Successfully Logged In !</string>
7       <string name="number_hint">Phonenumber</string>
8       <string name="text_hint">Sms-Text</string>
9       <string name="txt_log">Protokoll</string>
10  </resources>
11
```

menu_main.xml

menu_main.xml

```
1   <menu xmlns:android="http://schemas.android.com/apk/res/android"
2       xmlns:app="http://schemas.android.com/apk/res-auto"
3       xmlns:tools="http://schemas.android.com/tools"
4       tools:context="at.htl.smssender.MainActivity">
5       <item
6           android:id="@+id/action_settings"
7           android:orderInCategory="100"
8           android:title="Settings"
9           app:showAsAction="never" />
10  </menu>
```

Verwende nach Möglichkeit den Assistenten



activity_main.xml

Palette

Common
Text
Buttons
Widgets
Layouts
Containers
Google
Legacy

Ab TextView
Button
ImageView
RecyclerView
<fragment>
ScrollView
Switch

Component Tree

CoordinatorLayout
AppBarLayout
toolbar
<include>- @layout/con...
fab

Nexus 4 | 27 | NoActionBar | Default (en-us) | 38%

8:00

Phonenumber

Sms-Text

Protokoll

activity_main.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

    </android.support.design.widget.AppBarLayout>

    <include layout="@layout/content_main" />

    <android.support.design.widget.FloatingActionButton
        android:id="@+id/fab"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom|end"
        android:layout_margin="16dp"
        android:src="@drawable/ic_send" />

</android.support.design.widget.CoordinatorLayout>
```

content_main.xml

Palette

Common
Text
Buttons
Widgets
Layouts
Containers
Google
Legacy

Ab TextView
Button
ImageView
RecyclerView
<fragment>
ScrollView
Switch

Component Tree

ConstraintLayout
Ab til_phoneNumber
et_phone_number
Ab til_text
et_text
Ab tv_label_log - "@string/tx...
Ab tv_log_lines

Nexus 4 | 27 | NoActionBar | Default (en-us) | 38%

8dp

8:00

Phonenumber

Sms-Text

Protokoll

Protokoll

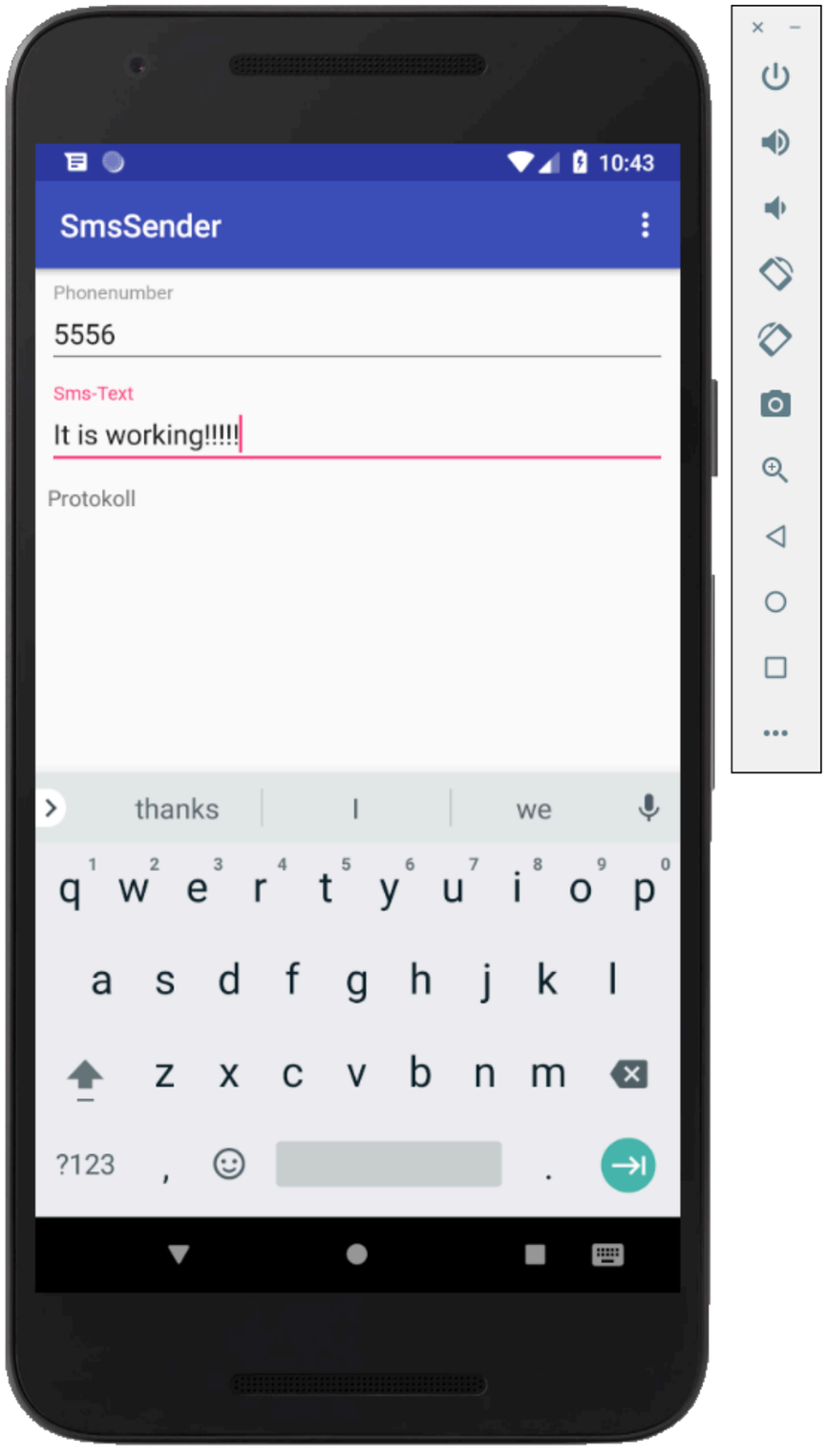
content_main.xml

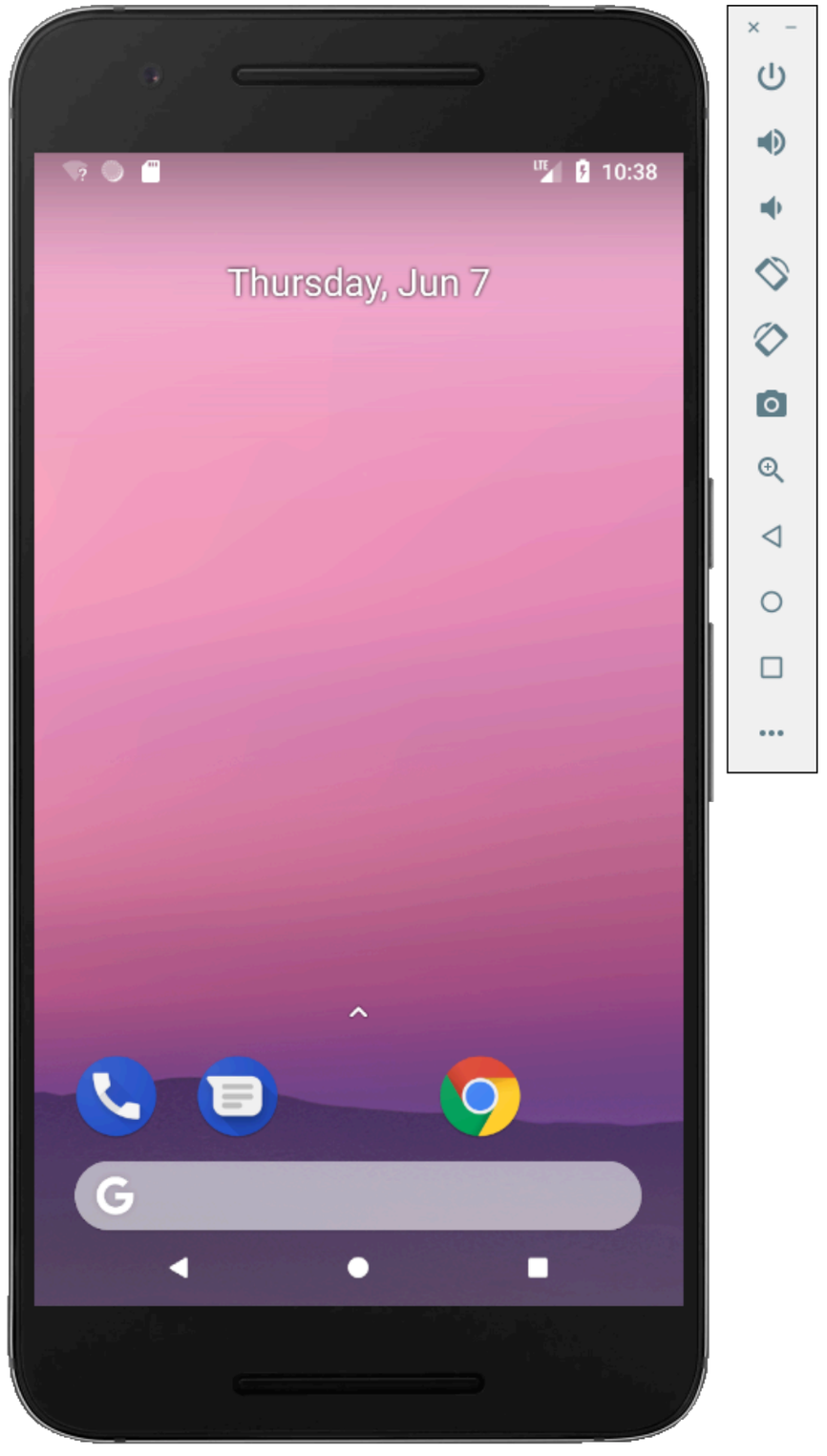
```xml
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context=".MainActivity"
    tools:showIn="@layout/activity_main">
    <android.support.design.widget.TextInputLayout
        android:id="@+id/til_phoneNumber"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">
        <android.support.design.widget.TextInputEditText
            android:id="@+id/et_phone_number"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="@string/number_hint"
            android:inputType="phone" />
    </android.support.design.widget.TextInputLayout>
    <android.support.design.widget.TextInputLayout
        android:id="@+id/til_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/til_phoneNumber"

        <android.support.design.widget.TextInputEditText
            android:id="@+id/et_text"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Sms-Text"
            android:inputType="text" />
    </android.support.design.widget.TextInputLayout>

    <TextView
        android:id="@+id/tv_label_log"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:text="Protokoll"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/til_text" />

    <TextView
        android:id="@+id/tv_log_lines"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:lines="8"
        android:linksClickable="false"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/tv_label_log" />

</android.support.constraint.ConstraintLayout>
```

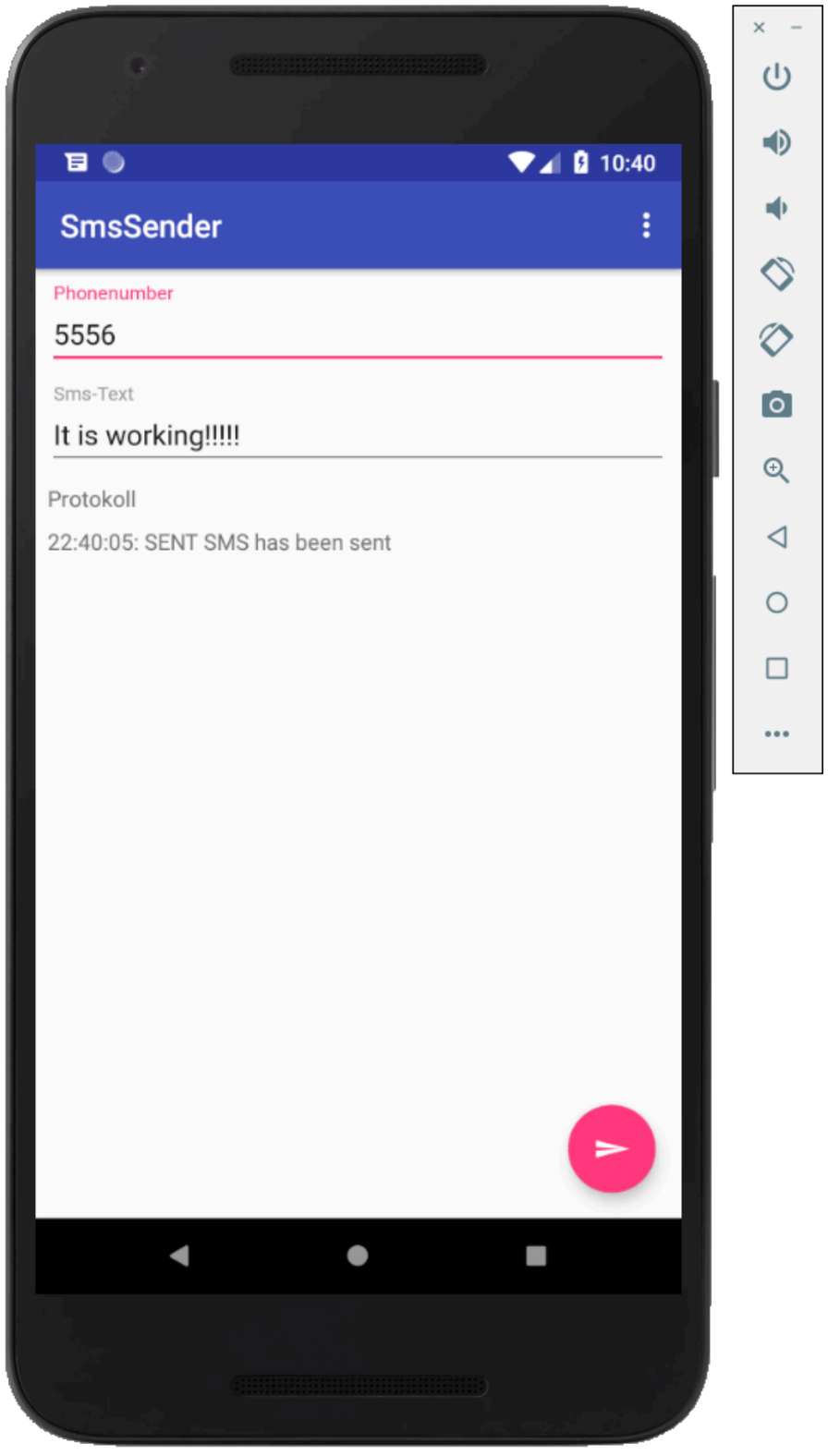
Ausprobieren unseres SMS Servers

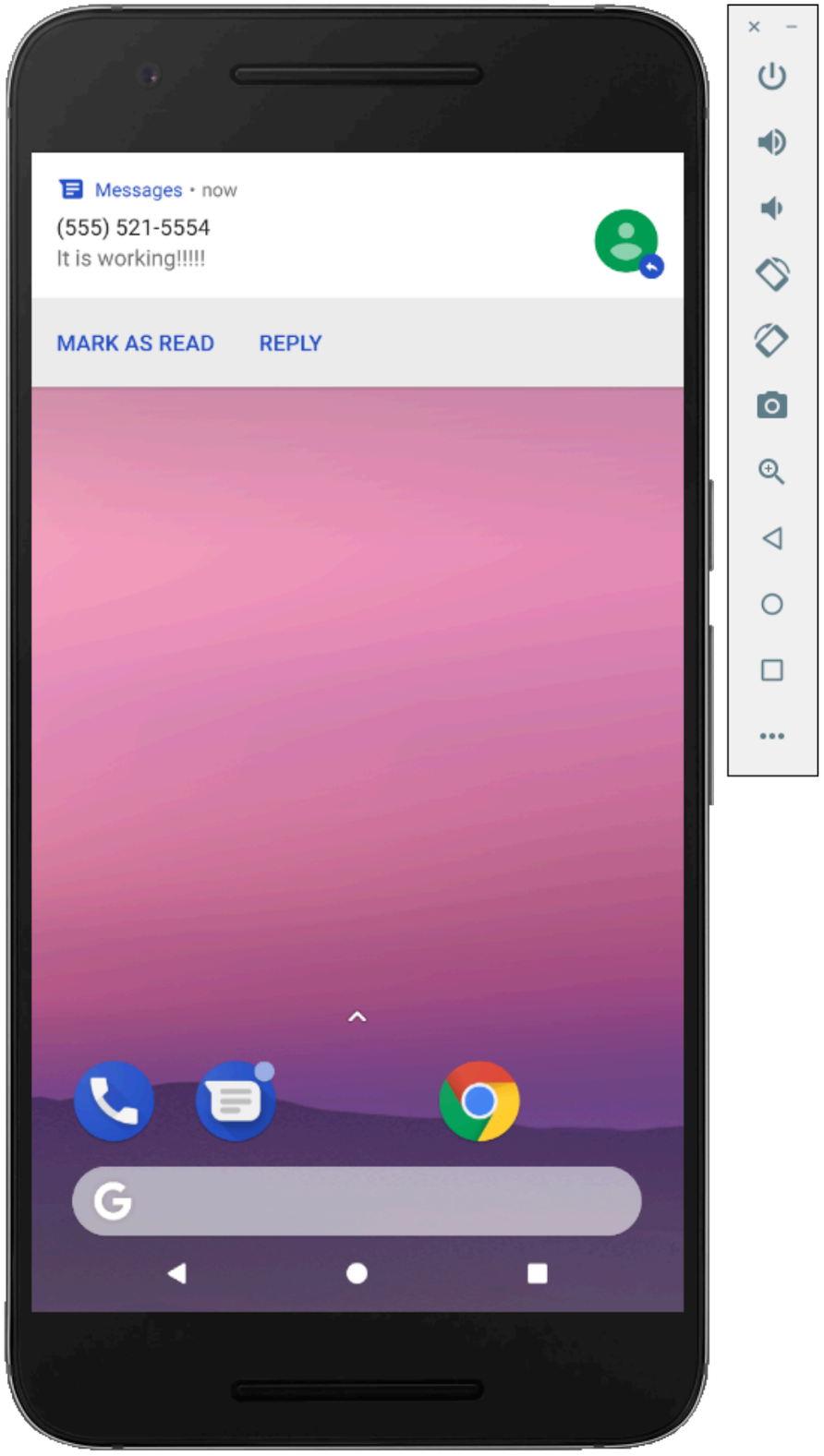
Feststellen der Port-Nrn

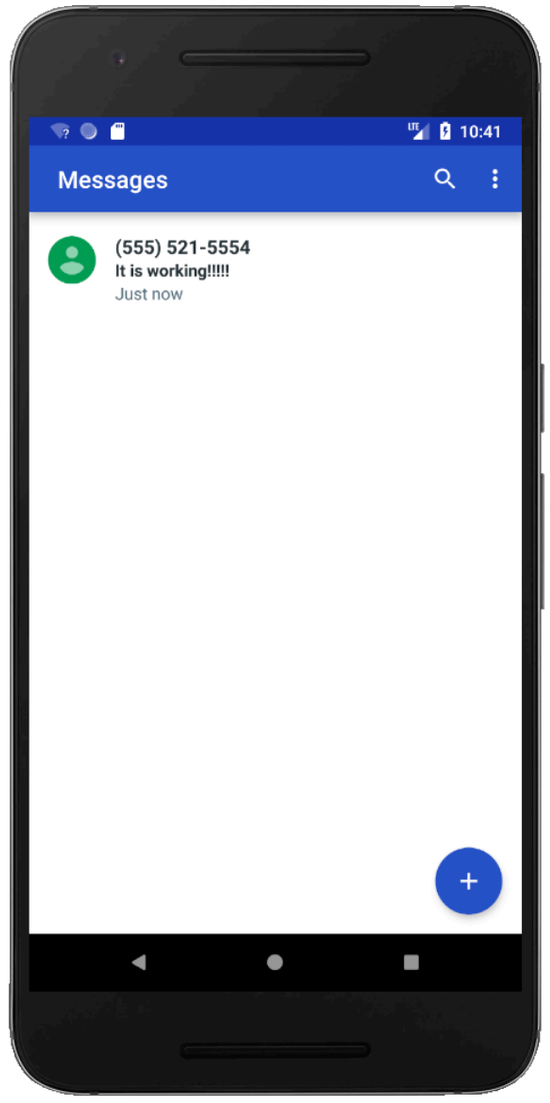
2 Emulatoren müssen vorab geöffnet werden


10:43
SmsSender
Phonenumber
5556
Sms-Text
It is working!!!!!
Protokoll
thanks
I
we
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123
,
.


10:38
Thursday, Jun 7


10:40
SmsSender
Phonenumber
5556
Sms-Text
It is working!!!!!
Protokoll
22:40:05: SENT SMS has been sent


Messages • now
(555) 521-5554
It is working!!!!!
MARK AS READ
REPLY


10:41
Messages
(555) 521-5554
It is working!!!!!
Just now