Android SDK Documentation

Version: 1.0 Last Updated: 23-Sep-2019

1. Introduction

Essentially Precise has developed an SDK that you can utilize in your mobile application to integrate SMS services directly by connecting to our platform and without the need to do additional programming or API integrations. This document will cover the details of the SDK on Android platform, how this can be implemented in your code and how to initiate SMS. Sample codes are also provided. For iOS SDK, please check iOS SDK download section on www.tobeprecisesms.com

2. How to Configure your Project

The name of the SDK is TbpSMS(supports Android KitKat and above). Once downloaded and added to your folder, follow the below steps to configure in your project.

1. Add TbpSMS SDK to your project as a new module using ‘Import .JAR/.AAR Package’ option.
   In Android Studio, choose File -> New -> New Module -> Import .JAR/.AAR Package

2. Add TbpSMS SDK as dependency in your app level build.gradle
   ```
   implementation project(':TbpSMS')
   ```

3. Add TbpSMS SDK’s dependencies in your app level build.gradle
   ```
   implementation 'com.squareup.retrofit2:retrofit:2.5.0'
    implementation 'com.squareup.retrofit2:converter-gson:2.5.0'
    implementation 'com.squareup.okhttp3:okhttp:3.12.1'
    implementation 'com.squareup.retrofit:adapter-rxjava:2.0.0-beta1'
   ```

4. Permissions required in AndroidManifest file
   ```
   <uses-permission android:name="android.permission.INTERNET"/>
   <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
   ```

5. Android Studio IDE details
   Android Studio Version : 3.5
   Andriod Gradle Plugin Version : 3.5.0
   Gradle Version : 5.4.1

You are all set to start using the SDK in your project !

2.1 Parameter details

In order to use the SMS service, some of the parameters in the SDK are mandatory. These are listed below. For more details on these parameters, visit www.tobeprecisesms.com and download our API documentation.

- User Name
- Password
- Mobile Number
- Message
- Sender Name

3. Using TbpSMS SDK

Before you can start using the SDK, please make sure you have the user name and password of your SMS account created on https://portal.tobeprecisesms.com platform. If you do not have an account yet, please write to support@tobeprecisesms.com for getting help. Once the account is obtained, follow the below steps.

- Import TbpSMS to start access SMS API’s

  ```
  import com.ep.epsmsmanager.RestApi.SMSCallbacks;
  import com.ep.epsmsmanager.RestApi.SMSManager;
  import com.ep.epsmsmanager.model.param.BulkSMSParam;
  import com.ep.epsmsmanager.model.param.CustomSMSParam;
  import com.ep.epsmsmanager.model.param.MessageDetail;
  import com.ep.epsmsmanager.model.param.SingleSMSParam;
  ```

- After importing set user and password before accessing any SMS API’s

  ```
  SMSManager.user = "<your user name>"
  SMSManager.password = "<your password>"
  ```

Once the initial setups are done, you can use one of the following functions of the SDK.

3.1 Checking Balance

To check the balance in your account before sending any SMS or to know if you have to purchase additional credits, use the method “GetBalance”. Sample is as below.

```
SMSManager.GetBalance(this,new SMSCallbacks() {
        @Override
        public void onSuccess(@NonNull String value) {
            Log.d("","value="+value);
        }

        @Override
        public void onError(@NonNull Throwable throwable) {
            Log.d("","throwable="+throwable.getMessage());
        }


        @Override
        public void onValidateError(String errorMessage){
            Log.d("","errorMessage="+errorMessage);
        }
    });
```

3.2 Sending Single SMS message

This method is used for sending SMS to one single number. This is useful if you are sending a test or OTP messages. Please avoid using this method to send bulk SMS.

Below code snippet will explain how to use the Single message method – “SingleSMS”. The mandatory parameters are pointed out. The remaining parameters are optional, which should be omitted from the method signature when not required. Refer sample project for further details.

```
ArrayList<String> mNumbers = new ArrayList<>();
 mNumbers.add("971551234567");
SingleSMSParam singleSMSParam =new SingleSMSParam();
singleSMSParam.setMobileNumbers(mNumbers); //Mandatory
singleSMSParam.setMessage("This is sample "); //Mandatory
singleSMSParam.setSenderName("XXXYYY "); //Mandatory
singleSMSParam.setReportRequired(true);
singleSMSParam.setCallbackQueryString("paramX=12345&paramY=abc");
singleSMSParam.setReferenceName("sample string 5");
singleSMSParam.setBrightLinks(true);
singleSMSParam.setBrightLinksBaseURL("http://prc.is");
singleSMSParam.setBrightLinksFinalURL("http://www.example.com");
SMSManager.SingleSMS(singleSMSParam,this, new SMSCallbacks() {
    @Override
    public void onSuccess(@NonNull String value) {
        Log.d("","value="+value);
    }

    @Override
    public void onError(@NonNull Throwable throwable) {
        Log.d("","throwable="+throwable.getMessage());
    }

    @Override
    public void onValidateError(String errorMessage){
        Log.d("","errorMessage="+errorMessage);
    }
});
```

3.3 Sending Bulk SMS

This method is similar to SingleSMS, except that here you can send the same message to multiple mobile numbers at the same time. The below sample code snippet highlights on how to use the “BulkSMS” method. An additional feature in Bulk SMS method is where you can schedule the SMS for a later delivery.

Below code snippet will explain how to use. The mandatory parameters are pointed out. The remaining parameters are optional, which should be omitted from the method signature when not required. Refer sample project for further details.

```
 ArrayList<String> mNumbers = new ArrayList<>();
mNumbers.add("971551234567");
```

```
 mNumbers.add("971501234567");
 BulkSMSParam bulkSMSParam=new BulkSMSParam();
 bulkSMSParam.setMobileNumbers(mNumbers); //Mandatory
 bulkSMSParam.setMessage("This is sample");  //Mandatory
 bulkSMSParam.setSenderName("XXXYYY");  //Mandatory
 bulkSMSParam.setScheduledDate("2019-09-29T14:49:59.5746271+04:00");
 bulkSMSParam.setRemoveDuplicates(true);
 bulkSMSParam.setReferenceName("sample string 5");
 bulkSMSParam.setReportRequired(false);
 bulkSMSParam.setCallbackQueryString("paramX=12345&paramY=abc");
 bulkSMSParam.setReturnIndividualResponse(true);
 bulkSMSParam.setAsynchronousSubmission(false);
 bulkSMSParam.setBrightLinks(true);
 bulkSMSParam.setBrightLinksBaseURL("http://prc.is");
 bulkSMSParam.setBrightLinksFinalURL("http://www.example.com");

SMSManager.BulkSMS(bulkSMSParam,this, new SMSCallbacks() {
     @Override
     public void onSuccess(@NonNull String value) {
         Log.d("","value="+value);
     }

     @Override
     public void onError(@NonNull Throwable throwable) {
         Log.d("","throwable="+throwable.getMessage());
     }

     @Override
     public void onValidateError(String errorMessage){
         Log.d("","errorMessage="+errorMessage);
     }
  });
```

Note: ScheduledDate format is **"2019-01-31T14:49:59.5746271+04:00"**, if you pass incorrect format or empty date and time, the message will be immediately scheduled and broadcasted.

Bright Links is used for converting your URL within the message to shorter links and for tracking purpose. For more details on Bright Links or to enable for your account, please contact your account manager.

3.4 Sending Customized Message

Customized message method (“CustomSMS”) allows you to send bulk SMS, but with the possibility to change the message for each number. Below code highlights on how to use this method.

Below code snippet will explain how to use. The mandatory parameters are pointed out. The remaining parameters are optional, which should be omitted from the method signature when not required. Refer sample project for further details.

```
ArrayList<MessageDetail> messageDetails = new ArrayList<>();
MessageDetail messageDetail1 = new MessageDetail();
messageDetail1.setTo("971501234567");
messageDetail1.setMessage("This is sample message ");
messageDetails.add(messageDetail1);

MessageDetail messageDetail2 = new MessageDetail();
messageDetail2.setTo("971551234567");
messageDetail2.setMessage("This is second sample message ");
messageDetails.add(messageDetail2);

CustomSMSParam customSMSParam = new CustomSMSParam();
customSMSParam.setMessageDetails(messageDetails); //Mandatory
customSMSParam.setSenderName("XXXYYY ");  //Mandatory
customSMSParam.setScheduledDate("2019-09-29T14:51:05.6904323+04:00 ");
customSMSParam.setRemoveDuplicates(true);
customSMSParam.setReferenceName("sample string 4");
customSMSParam.setReportRequired(true);
customSMSParam.setCallbackQueryString("paramX=12345&paramY=abc");
customSMSParam.setReturnIndividualResponse(true);
customSMSParam.setAsynchronousSubmission(false);
customSMSParam.setBrightLinks(true);
customSMSParam.setBrightLinksBaseURL("http://prc.is");
customSMSParam.setBrightLinksFinalURL("http://www.example.com");
SMSManager.CustomSMS(customSMSParam, this,new SMSCallbacks() {
   @Override
```

```
    public void onSuccess(@NonNull String value) {
        Log.d("","value="+value);
    }

    @Override
    public void onError(@NonNull Throwable throwable) {
        Log.d("","throwable="+throwable.getMessage());
    }

    @Override
    public void onValidateError(String errorMessage){
        Log.d("","errorMessage="+errorMessage);
    }
});
```

4. Reports and Analytics

All your SMS reports are available on our web portal https://portal.tobeprecisesms.com and can be viewed or downloaded using the same SMS account credentials. You can also generate detailed analytics based on Sender name, Country or Delivery status.

5. Support

For any support related to SDK, SMS delivery or our platform, please feel free to visit our website or email to support@tobeprecisesms.com Make sure to provide as much details as possible and also mention the user name of your account for faster response.