Android 6.x.x to 7.0 Migration Guide

Upgrade to Alchemer Mobile Android SDK 7.0

Alchemer Mobile Android SDK 7.0 is a modernized update focused on accessibility, stability, and more reliable in-app interactions—plus support for Android 16 full screen mode for digital interactions.

Who this update is for

This update is for teams using the Alchemer Mobile Android SDK (5.x or 6.x) who want the latest improvements and are ready to align with the new minimum Android and compile SDK requirements.

Estimated migration effort: Small (typically under 1 hour)

What's changed in 7.0

Major differences

- **Minimum supported Android version:** Android 7.0+ (**API level 24**)
- **Minimum compileSdkVersion: 35** (per Google requirements)
- **EU region support:** Set via the **region** field during registration (default is **US**)

Improvements and new features

- Support for **Android 16 full screen mode** for digital interactions
- **Major accessibility improvements**
  - Screen reader compatibility
  - Keyboard navigation fixes
  - Improved contrast and labeling
- **Improved SDK stability**
- **Faster, more reliable loading** of in-app interactions
- Important note: **devices using SDK 5.x–6.x may have issues** seeing all content or actions in some interactions

---

Step 1: Check your current SDK version

1. Open `app/build.gradle`.
2. In the `dependencies` section, locate the dependency for: `com.apptentive:apptentive-kit-android`
3. The SDK version is appended to the dependency declaration.

Example patterns you might see:

```
implementation "com.apptentive:apptentive-kit-android:$apptentive_version"
```

Step 2: Update the SDK dependency

In your `build.gradle`, update the dependency to the latest Alchemer Mobile Android SDK version:

```
dependencies {
  implementation "com.apptentive:apptentive-kit-android:APPTENTIVE_VERSION"
}
```

Replace `APPTENTIVE_VERSION` with the most recent version used by your team.

Step 3: Register the SDK in your Application class

Shortly after app launch, register the SDK—usually in your `Application` class—using an `ApptentiveConfiguration` with your **App Key** and **App Signature** from the **API & Development** section of the **Settings** tab in your Alchemer Mobile dashboard.

Kotlin example

```
class MyApplication : Application() {
  override fun onCreate() {
    super.onCreate()
    val configuration = ApptentiveConfiguration(
    apptentiveKey = "< YOUR_APPTENTIVE_KEY>",
    apptentiveSignature = "< YOUR_APPTENTIVE_SIGNATURE>"
    ).apply {
      /**
      * Optional parameters:
      * shouldInheritAppTheme - Default is true
      * logLevel - Default is LogLevel.Info
      * shouldSanitizeLogMessages - Default is true
      * ratingInteractionThrottleLength - Default is TimeUnit.DAYS.toMillis(7)
      * customAppStoreURL - Default is null (Rating Interaction attempts to show Google In-App Review)
      */
    }
    Apptentive.register(this, configuration)
  }
}
```

Java example

```
public class MyApplication extends Application {
  @Override
  public void onCreate() {
    super.onCreate();

    ApptentiveConfiguration configuration = new ApptentiveConfiguration(
      "<YOUR_APPTENTIVE_KEY>",
      "<YOUR_APPTENTIVE_SIGNATURE>"
    );

    Apptentive.register(this, configuration);
  }
}
```

Don't have an Application class yet? Create one and reference it in your app manifest.

Step 4: Register Activity callbacks (required for interactions)

SDK 7.0 needs each Activity to register itself so the SDK can display interactions (surveys, prompts, rating dialogs, Message Center, and more).

What to do

For every Activity:

1. Implement `ApptentiveActivityInfo`
2. Return the Activity in `getApptentiveActivityInfo()`
3. Register the callback in `onResume()`

4. Unregister the callback in `onPause()`

Tip: If you have a `BaseActivity`, implement this once there and have other Activities extend it.

Kotlin example

```
class MainActivity : AppCompatActivity(), ApptentiveActivityInfo {
   override fun onResume() {
      super.onResume()
      Apptentive.registerApptentiveActivityInfoCallback(this)
   }

   override fun getApptentiveActivityInfo(): Activity = this

   override fun onPause() {
      Apptentive.unregisterApptentiveActivityInfoCallback()
      super.onPause()
   }
}
```

Java example

```
public class MainActivity extends AppCompatActivity implements ApptentiveActivityInfo {
   @Override
   protected void onResume() {
      super.onResume();
      Apptentive.registerApptentiveActivityInfoCallback(this);
   }

   @NonNull
   @Override
   public Activity getApptentiveActivityInfo() {
      return this;
   }

   @Override
   protected void onPause() {
      Apptentive.unregisterApptentiveActivityInfoCallback();
      super.onPause();
   }
}
```

Step 5: Update engage events (syntax changes)

Use `Apptentive.engage("event_name")` throughout your app lifecycle to trigger targeting and launch interactions.

Good places to engage events

- When an Activity comes into focus
- Button taps
- Error states or key workflow events

Avoid calling engage events before the SDK has finished registering (for example, too early in the `Application` class).

Kotlin example

```
 Apptentive.engage("my_event")

Apptentive.engage("my_event") { result ->
   when (result) {
      is EngagementResult.InteractionShown -> { /* shown */ }
      is EngagementResult.InteractionNotShown -> { /* not shown */ }
      is EngagementResult.Error -> { /* evaluation error */ }
      is EngagementResult.Exception -> { /* unexpected issue */ }
   }
}
```

Java example

```
 Apptentive.engage("my_event");
```

Step 6: If you use Message Center, update how you open it

In SDK 7.0, call Message Center **without passing an Activity context**:

```
 Apptentive.showMessageCenter()
```

Optional callback pattern is also supported.

If your app still uses minSdkVersion = 21

Apps with `minSdkVersion = 21` **cannot compile** the newest SDK directly because SDK 7.0 requires **minSdkVersion = 24**.

If you still need to support API 21–23 devices *in your app*, you can compile by applying a manifest override and ensuring **all SDK calls are guarded by runtime checks**. This keeps the SDK from running on unsupported devices.

1) Add an override block in AndroidManifest.xml

Insert this inside your app module's manifest:

```
<uses-sdk
  android:minSdkVersion="24"
  tools:overrideLibrary="apptentive.com.android.feedback.enjoyment,
  apptentive.com.android.feedback.survey,
  apptentive.com.android.feedback.notes,
  apptentive.com.android.feedback.ratings,
  apptentive.com.android.feedback.messagecenter,
  apptentive.com.android.feedback.link,
  apptentive.com.android.feedback.inappreview,
  apptentive.com.android.feedback.initiator,
  apptentive.com.android.core"
  />
```

2) Guard all SDK calls with a runtime check

Kotlin example:

```
if (Build.VERSION.SDK_INT >= 24) {
  Apptentive.engage("my_event")
}
```

This prevents crashes on devices running API < 24 while continuing to support users on API 24+.

Notes for Java-based apps

SDK 7.0 supports both Java and Kotlin Android apps, but due to Kotlin/Java interoperability limits, you may see more internal APIs exposed in Java.

Recommendation: Avoid using any fields or methods marked `@InternalUseOnly` (they may change without notice).

Troubleshooting tips

- **Interactions don't appear**
  - Confirm you implemented `ApptentiveActivityInfo` in the current Activity and registered/unregistered callbacks properly.
  - Make sure SDK registration happens before you engage events.
- **Build errors after upgrading**
  - Confirm your `compileSdkVersion` is set to **35**
  - Confirm your minSdkVersion strategy (24+ required, or use the minSdk 21 workaround above)
- **EU data routing**

- Set the `region` field in `ApptentiveConfiguration` during registration (default is US).

FAQ

Do I need to rewrite my app in Kotlin?

No. SDK 7.0 is compatible with both Java- and Kotlin-based Android apps.

Can I upgrade directly from 5.x to 7.0?

If you're currently on 5.x, the recommended path is to upgrade to 6.x first, then move to 7.x (per internal guidance).

Related Articles