**Steps to Integrate HockeyApp SDK with Android Application.**

**CREDENTIALS** :- UserName :- [hd00483723@techmahindra.com](mailto:hd00483723@techmahindra.com)

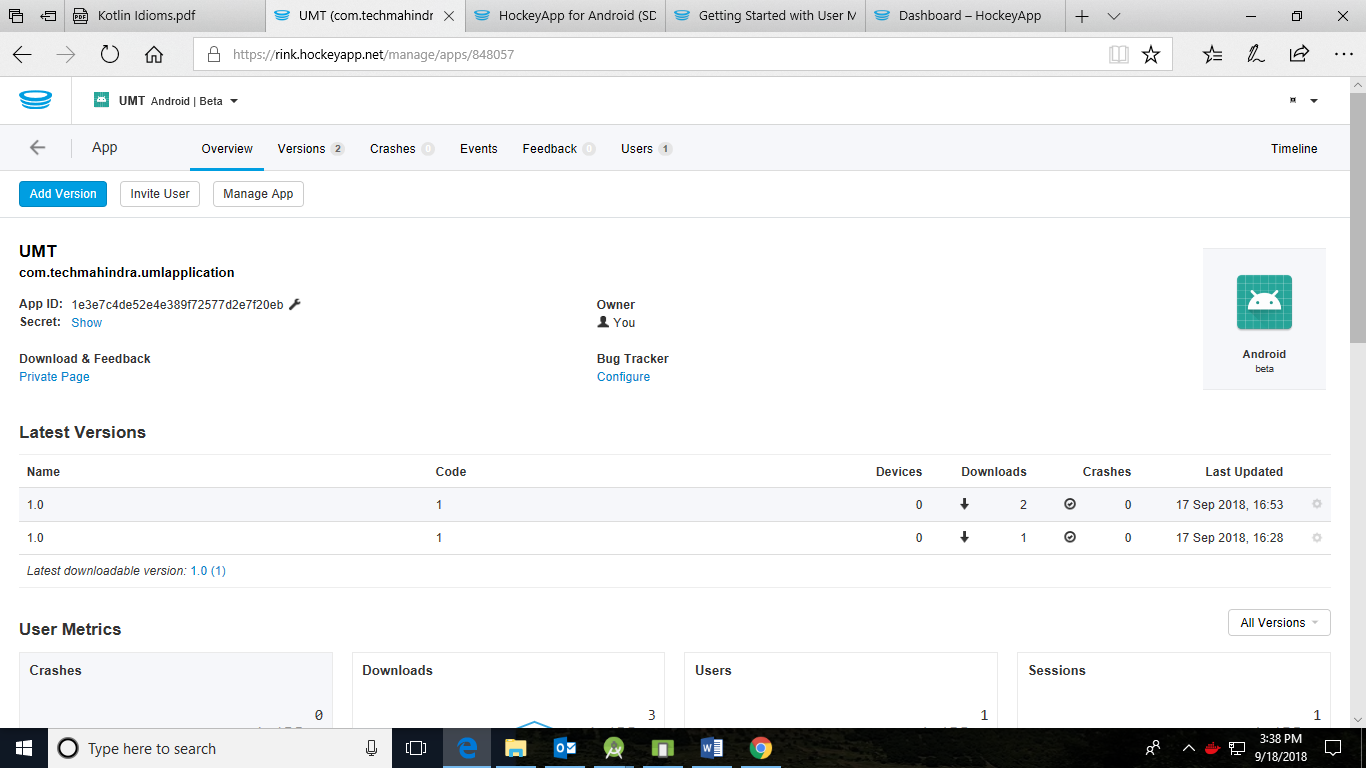
Password :- Techm\_2018

**REQUIREMENTS :-**

1. The SDK runs on devices with API Level 15 or later (Android 4.0.3 and up).
2. Android project in Android Studio or another Android IDE.
3. **SETUP :-**

* **Obtain an app identifier**

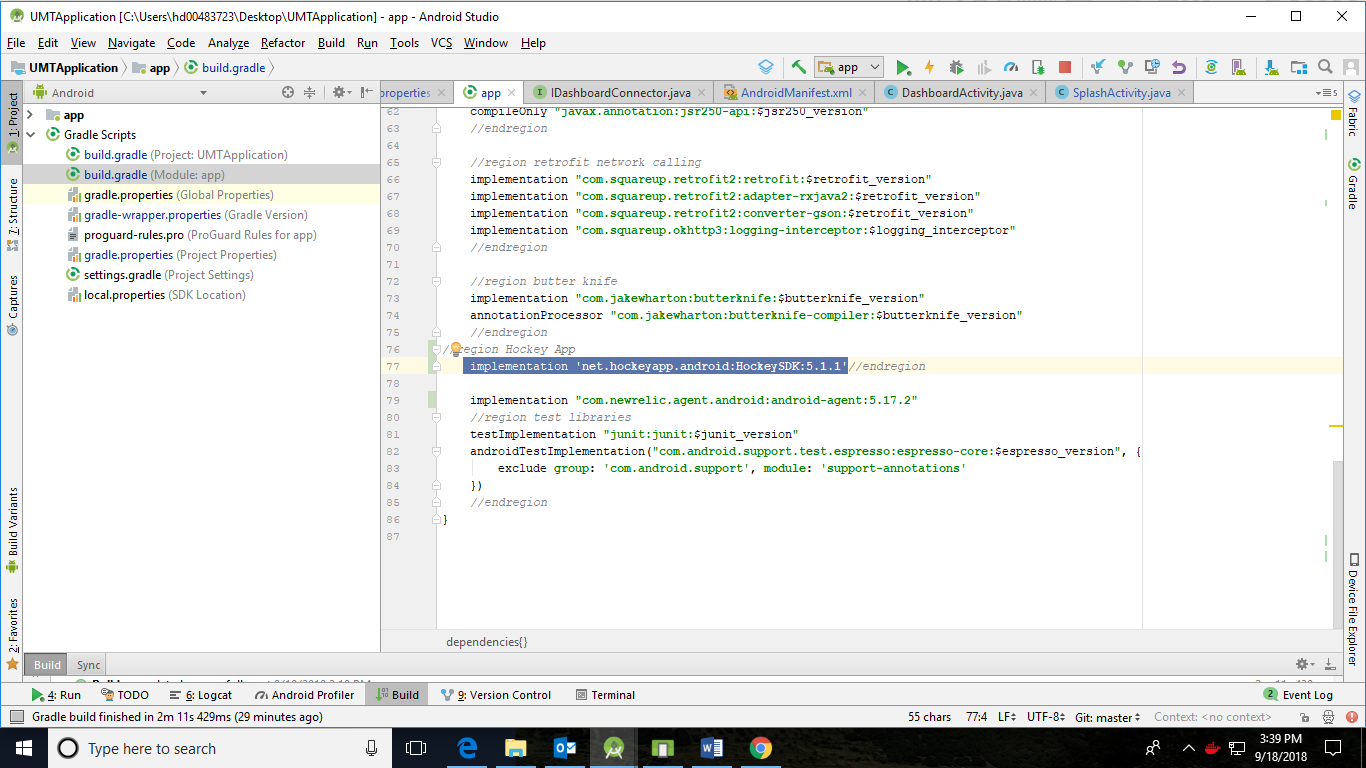
1. Login in hockey app with above mentioned Credentials.
2. Add App & App Details in Hockey App Website.
3. This will provide you with an HockeyApp-specific App Identifier to be used to initialize the SDK.



* **Get the SDK**

1. Add the SDK to your app module's dependencies in Android Studio by adding the following line to your dependencies { ... } configuration:

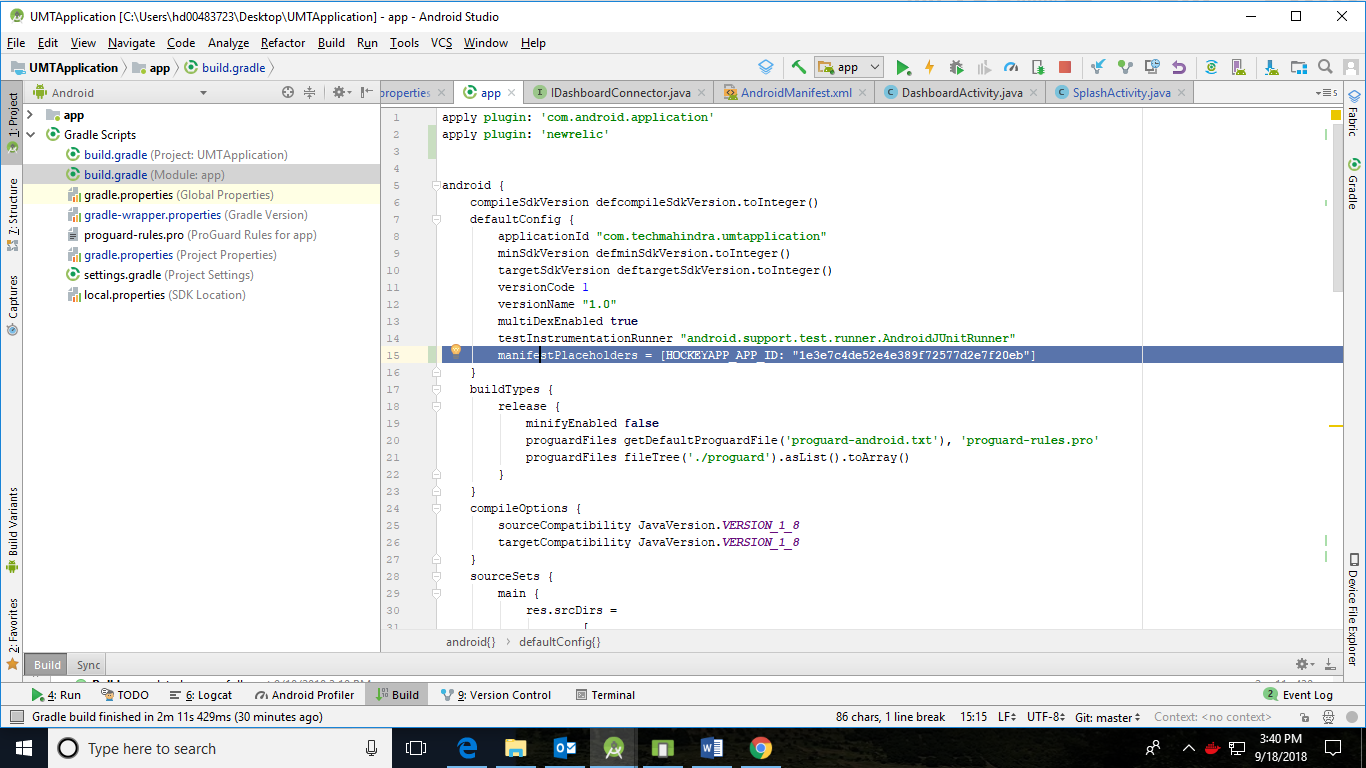
compile 'net.hockeyapp.android:HockeySDK:5.1.1'



* **Integrate Hockey SDK**

1. Open your module's build.gradle file.
2. Add the following manifest placeholder to your configuration (typically the defaultConfig):

manifestPlaceholders = [HOCKEYAPP\_APP\_ID: "$APP\_ID"]



1. The param $APP\_ID must be replaced by your HockeyApp App Identifier. The app identifier can be found on the app's page in the "Overview" section of the HockeyApp backend.
2. Save your build.gradle file and make sure to trigger a Gradle sync.
3. Open your AndroidManifest.xml file and add a meta-data-tag for the HockeySDK.

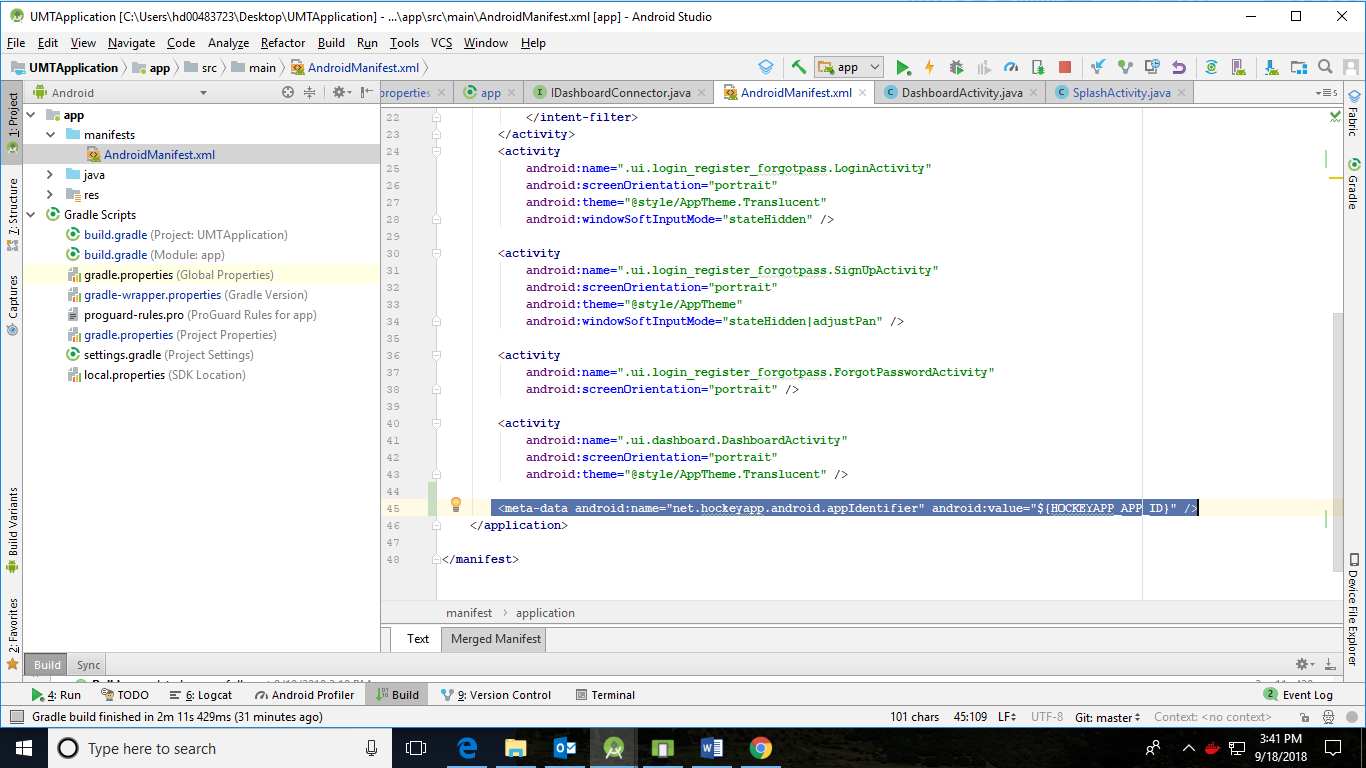
<application>

//your activity declarations and other stuff

<meta-data android:name="net.hockeyapp.android.appIdentifier" android:value="${HOCKEYAPP\_APP\_ID}" />

</application>

Save your AndroidManifest.xml file.



### Add Crash Reporting

This will add Crash Reporting capabilities to app.

1. Open your main activity.
2. Add the following lines:

Java:

import net.hockeyapp.android.CrashManager;

public class YourActivity extends Activity {

@Override

public void onResume() {

super.onResume();

// ... your own onResume implementation

checkForCrashes();

}

private void checkForCrashes() {

CrashManager.register(this);

}

}

### Add User Metrics

To integrate User Metrics perform the following steps:

1. Open your app's main activity and add the import statement and one line of code to the activity's onCreate-callback. Add the trackEvent()-call wherever you want to track a Custom Event.

Java:

//add this import

import net.hockeyapp.android.metrics.MetricsManager;

// add this to your main activity's onCreate()-callback

MetricsManager.register(getApplication());

// add this wherever you want to track a custom event

MetricsManager.trackEvent("YOUR\_EVENT\_NAME");

// add this wherever you want to track a custom event and attach properties or measurements to it

HashMap<String, String> properties = new HashMap<>();

properties.put("Property1", "Value1");

HashMap<String, Double> measurements = new HashMap<>();

measurements.put("Measurement1", 1.0);

MetricsManager.trackEvent("YOUR\_EVENT\_NAME", properties, measurements);

Make sure to replace "YOUR\_EVENT\_NAME" with a name for your custom event, e.g. "Login Button Pressed".

**Limits**

* Accepted characters for tracking events are: [a-zA-Z0-9\_. -]. If you use other than the accepted characters, your events will not show up in the HockeyApp web portal.
* There is currently a limit of 300 unique event names per app per week.
* There is NO limit on the number of times an event can happen.
* Properties and measurements added to Custom Events are available in Application Insights Analytics as a preview. Please have a look at the [public announcement](https://www.hockeyapp.net/blog/2016/08/30/custom-events-public-preview.html) to find out more.

### Add update distribution

1. Open the activity where you want to inform the user about updates. We'll assume you want to do this on startup of your main activity for our example.
2. Add the following lines and make sure to always balance register(...) calls to SDK managers with unregister() calls in the corresponding lifecycle callbacks:

Java:

import net.hockeyapp.android.UpdateManager;

public class YourActivity extends Activity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

// Your own code to create the view

// ...

checkForUpdates();

}

private void checkForUpdates() {

// Remove this for store builds!

UpdateManager.register(this);

}

private void unregisterManagers() {

UpdateManager.unregister();

}

@Override

public void onDestroy() {

super.onDestroy();

unregisterManagers();

}

}

1. We can check Crash Reports,Downloads etc. in Hockey APP Dashboard.

