linkalist SDK for Android

Getting Started

30 November 2018



# Introduction

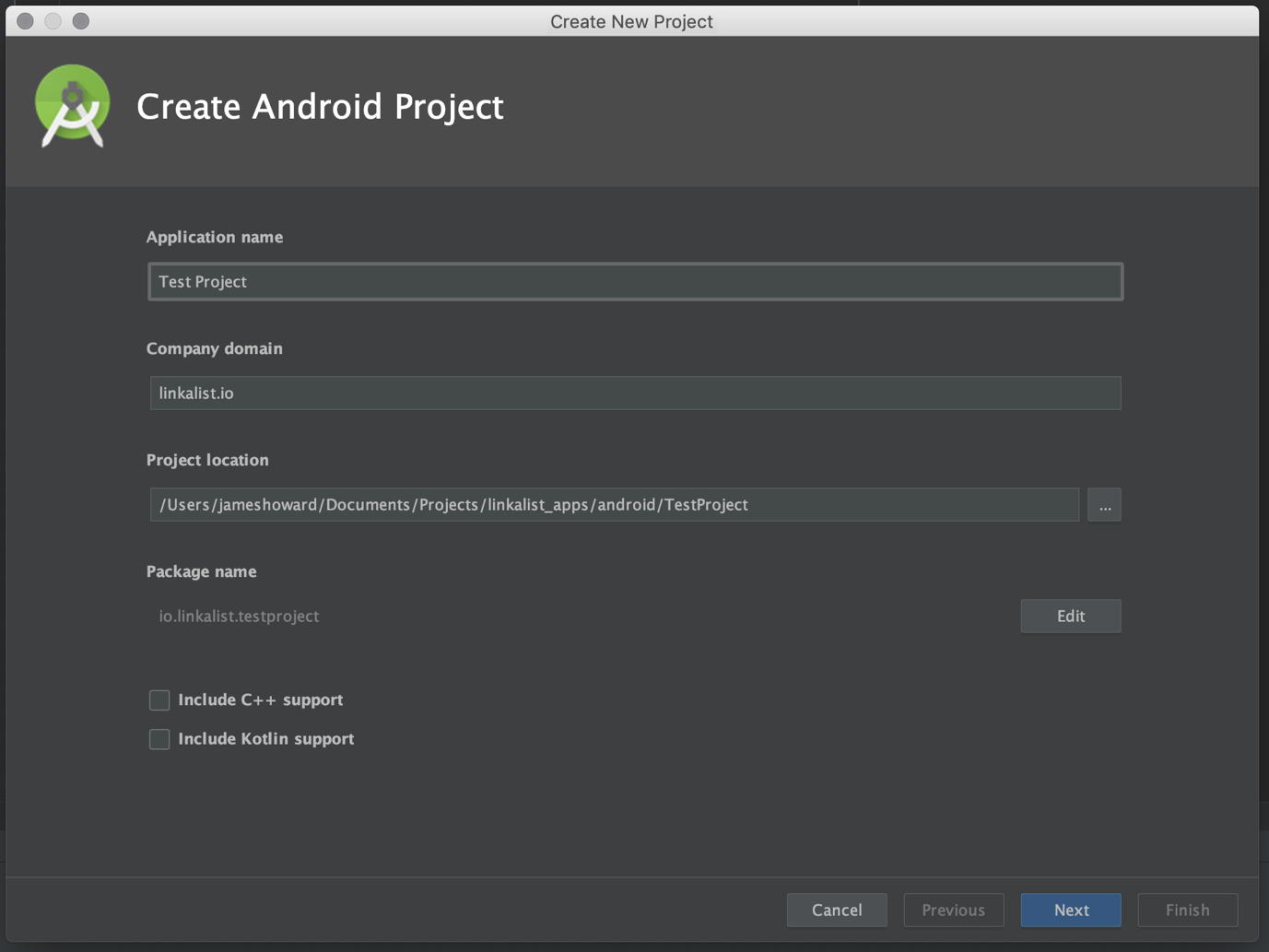
This document is all about getting you started with creating your linkalist application as a native Android application. Our Beta SDK will allow you to manage data, implement Login using various social media platforms and also access some generally useful functions such as the device location. We also plan to implement a camera interface.

# Version Requirements

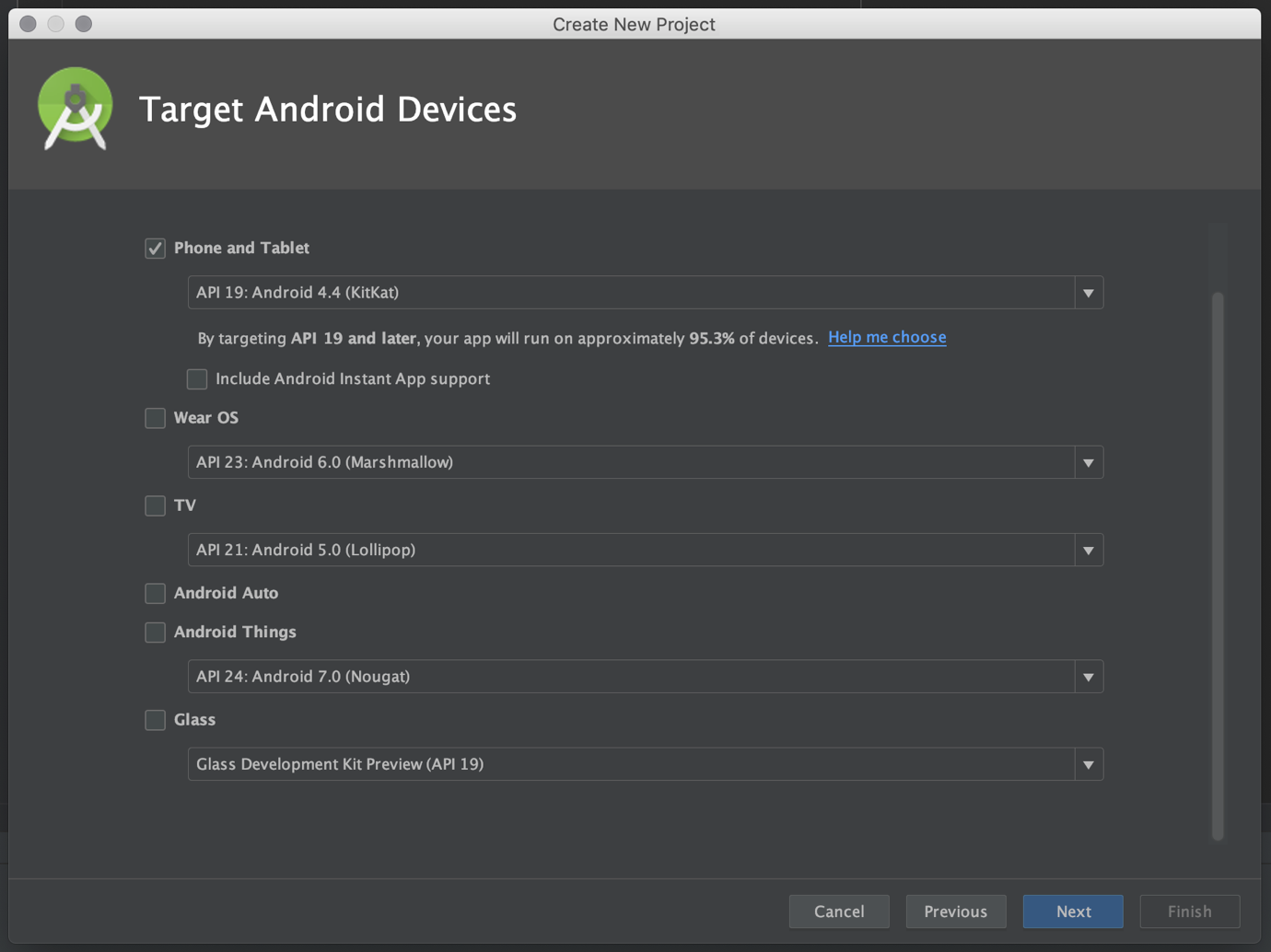
The linkalist SDK supports versions of android as old as Jelly Bean with a minimum SDK version of 16, so your minimum version will need to be at 16 or greater. We have compiled the SDK with version 28 and we recommend you build your own code on this version with the latest version of Android Studio.

# Application Setup

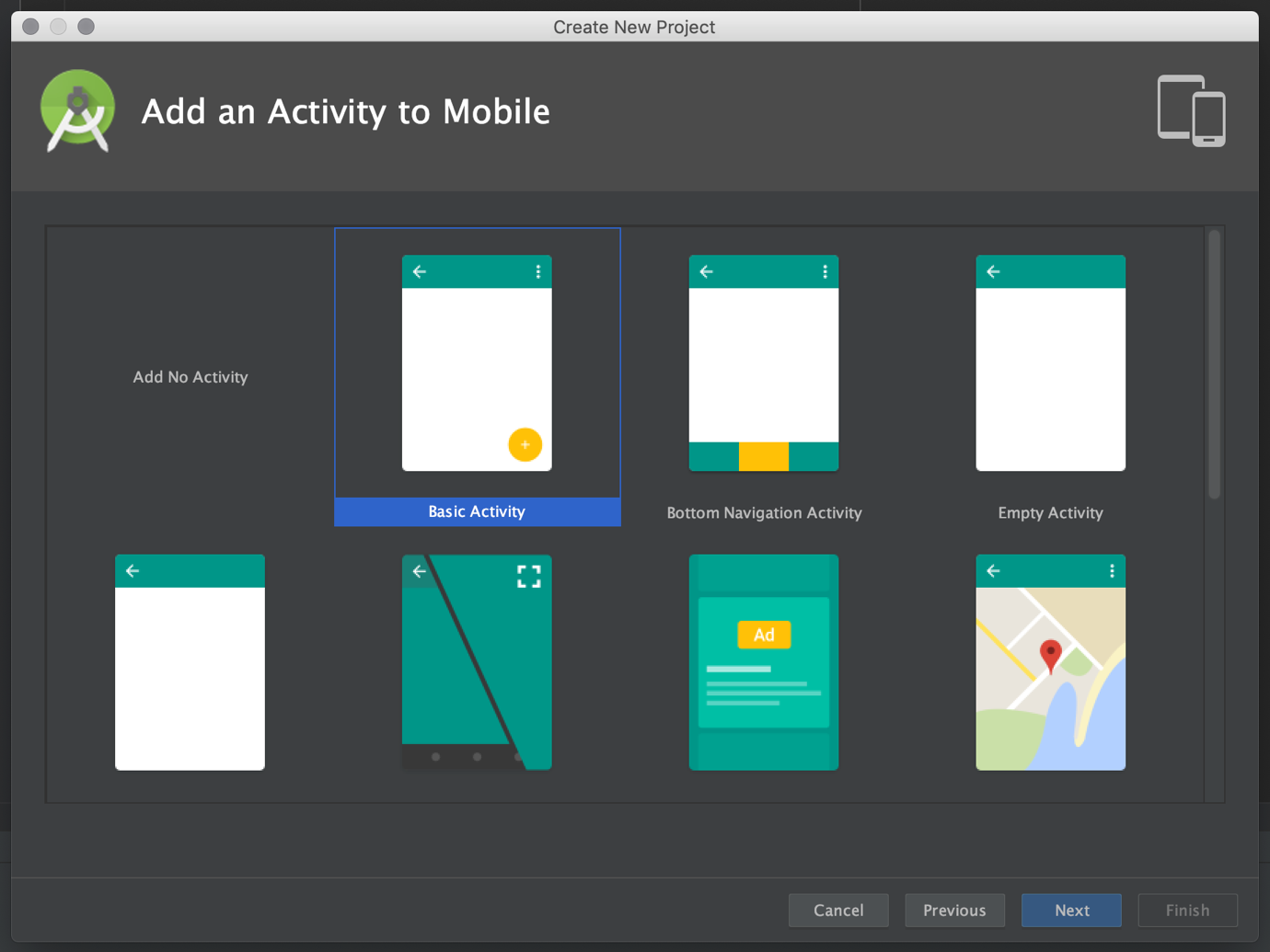
Now, you’re going to need to create a new Android Project in Android Studio and import our library into it so that you can get started properly. Use your own domain as the root of the package domain so that you end up with “com.yourdomain.testproject”.



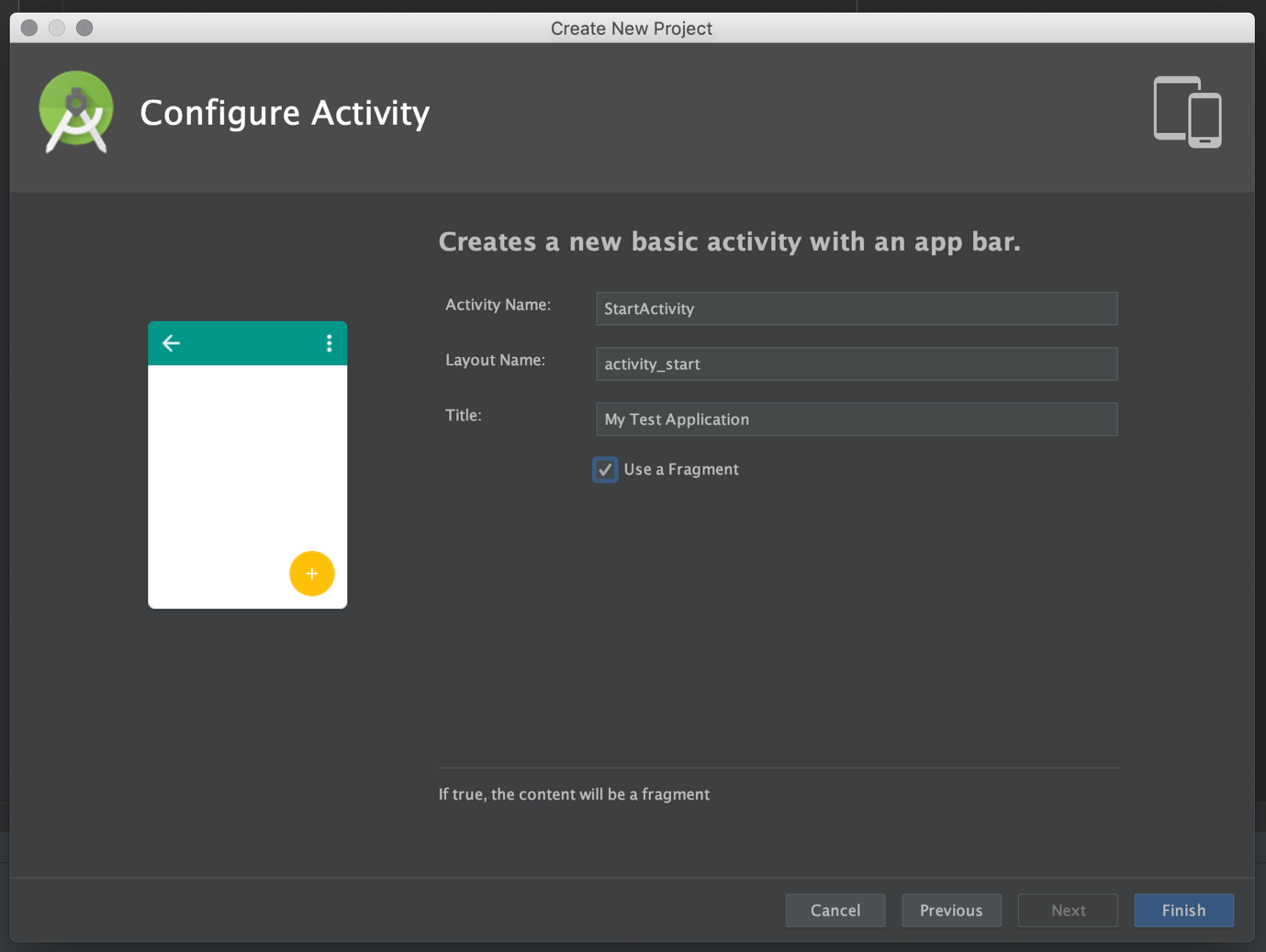
In the next step, you’ll need to choose an Android version. While we go all the way back to 16, unless your app has some feature that requires the support of ancient devices, you’re probably best sticking with KitKat – Version 19 as that is supported on around 95% devices at the time of writing



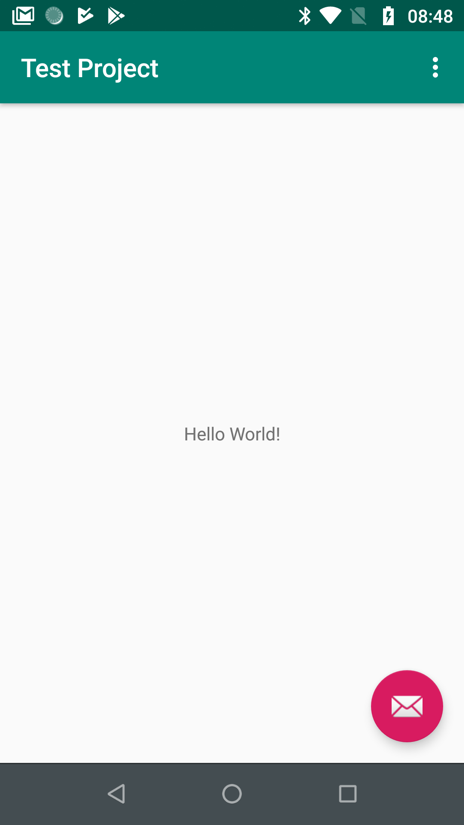
Next, you need to add your basic activity from a choice. You can go with what you want here but if you’re just playing around, I’d suggest using either BasicActivity or Empty Activity. We would **not** recommend choosing Login Activity as your start activity as we provide one of those in the SDK and it makes for bad flow to present a LoginActivity every time your app starts.



The final step is to enter a name and some details for your application. Knock yourself out here – you can use anything you like. But we highly recommend ticking the “Use a Fragment” option as this allows you much more flexibility to move your code around. Also most of the linkalist library user-interface components are fragment-based which will make your life easier when getting started.



At this point, it’s probably a good idea to run a build and check that what you have set up starts on a phone.



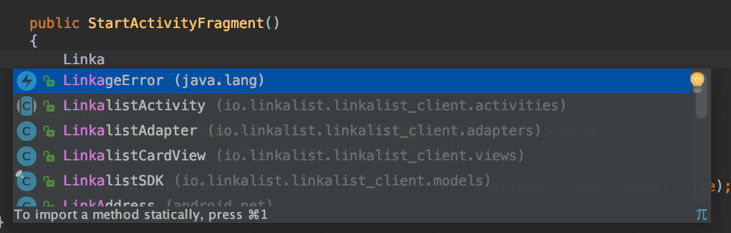
# Importing the Linkalist Library

The linkalist library is provided in aar format. To import this, the first step is to import the AAR file into your project. To do this, go to File – New Module, select “Import .JAR / .AAR Package” and choose the Linkalist AAR file.

Next you need to modify your app’s build.gradle file. This is the one under app directory – be sure not to modify the project’s gradle file. You need to add the following three lines. The first makes sure the linkalist library is included while the second two are dependencies we need for linkalist.

implementation project(path: ':linkalist-client-release’)  
  
api "com.facebook.android:facebook-android-sdk:4.18.0"  
api "com.nostra13.universalimageloader:universal-image-loader:1.9.1"

At this point, after a build, the Linkalist classes should appear when you start typing code if everything has worked properly.



It’s probably best to quickly start your app on your device again.

# Initialising your App

The linkalist SDK now needs to be initialised. This should be done as early as possible in your application’s lifecycle and the best place to do this is by extending the Application class. It’s also a good idea to set the base URL at this point to that used by your project

public class TestProjectApplication extends Application  
{  
 @Override  
 public void onCreate()  
 {  
 super.onCreate();  
  
 new LinkalistSDK().initializeSDK(this, BuildConfig.*BUILD\_TYPE*);  
 LinkalistCore.*getInstance*().setBaseUrl("https://testproject.linkalist.io");  
 }  
}

Then you need to add the new application class to your application manifest as follows

<application  
 android:name=".TestProjectApplication"  
 ……

</application>

# Logging In

We’re going to assume that your app depends on your customers being logged in so the first thing we need to do is allow them to log in. For the moment, we’ll assume that your customers already have created an account and set a password.

We are now going to make the app so that once it starts up, it checks if your customer is logged in and if not launches the system Login Dialog. You can also create your own login dialog but the details for managing this are documented elsewhere. The login dialog code will take care of all of the session managed so that all you need to do is launch the dialog and handle the result.

@Override  
protected void onCreate(Bundle savedInstanceState)  
{  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_dive\_list*);  
  
 if (false == LinkalistCore.*getInstance*().isLoggedIn())  
 {  
 LoginActivity.*launchActivity*(this);  
 }  
}

You can handle the result by overriding onActivityResult in the normal manner. For a start, displaying a nice little Toast will do the job nicely.

@Override  
protected void onActivityResult(int requestCode, int resultCode, Intent data)  
{  
 if (requestCode == LoginActivity.*REQUEST\_LOGIN*)  
 {  
 if (resultCode == Activity.*RESULT\_OK*)  
 {  
 Toast.*makeText*(this, R.string.*login\_success*, Toast.*LENGTH\_LONG*).show();  
 }  
 else if (resultCode == Activity.*RESULT\_CANCELED*)  
 {  
 //Write your code if there's no result  
 }  
 }  
 else  
 {  
 super.onActivityResult(requestCode, resultCode, data);  
 }  
}

So that will about do the trick. You now have an app that starts up and allows your customers to log into your system. The next step is to make the app do something vaguely useful.