Weather Wizard SDK for Android

Access Guide

DAPSDK_Weather-release-1.0.2

Contents

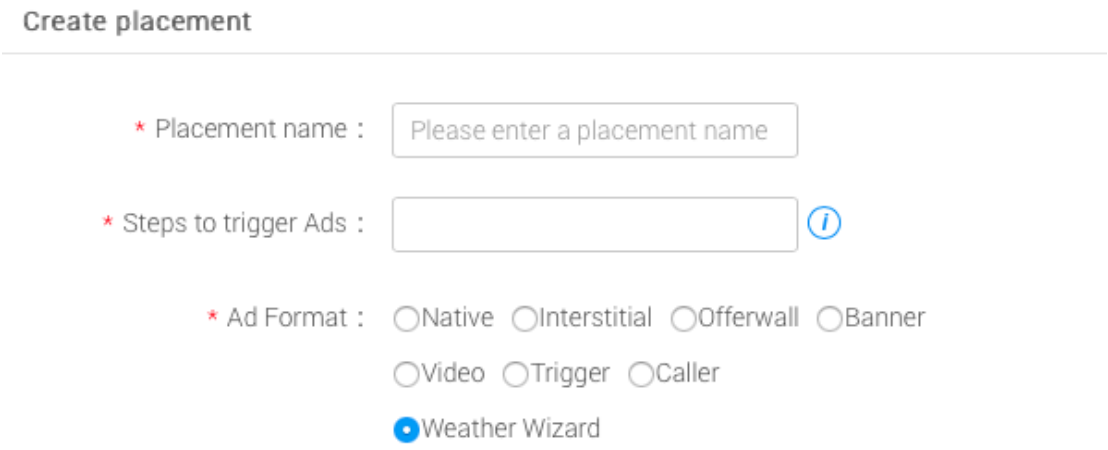
Precondition:

Currently DU Weather Wizard SDK must rely on DU Ad Platform_SDK CW1.0.9.7 or HW1.0.9.8 (included) plus SDK version.

1. Obtain Identity

Please refer to the chapter 3 in HW or CW version of DUADplatform SDK Access Guide to obtain necessary identities.

When applying for the DAP Placement ID , please make sure the app format you choose is 【Weather Wizard】.



2. Load SDK and Configuration

2.1 Load DU Ad Platform SDK

Please refer to the chapter 4.1 in HW or CW version of DUADplatform SDK Access Guide to load DU SDK.

For accessing Weather Wizard, the below operations are needed.

1) Copy the DAPSDK_Weather-release-xxx.aar to your Android Project, under the *libs* directory in root directory
2) Then configure build.gradle：

```
repositories {
    flatDir {
        dirs 'libs'
    }
}
dependencies {
    compile fileTree(include: ['*.jar'], dir: 'libs')
    compile(name: 'DuappsAd-xW-xxx-release', ext: 'aar')      // Weather Wizard SDK must rely on DU AD SDK
    compile(name: 'DAPSDK_Weather-release-xxx', ext:'aar')
}
```

2.2 Configure AndroidManifest.xml

Please refer to the chapter 4.1 in HW or CW version of DUADplatform SDK Access Guide to configure AndroidManifest.xml

A. Add a user-permission element to the manifest. For accessing Weather Wizard, except the basic permissions:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

Additional needed permissions is as below：

```
<!--Location-->
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<!--Floating window-->
<uses-permission android:name="android.permission.GET_TASKS"/>
```

B. Add the below part in your AndroidManifest.xml.

```
    <!--dapweather begin-->
    <activity
        android:name="com.daps.weather.DapWeatherActivity"
        android:screenOrientation="portrait"
        />
    <receiver
android:name="com.daps.weather.reciver.DapWeatherBroadcastReceiver">
        <intent-filter android:priority="90000">
            <action android:name="android.intent.action.USER_PRESENT"/>
        </intent-filter>
    </receiver>
    <service
android:name="com.daps.weather.service.DapWeatherMsgService"></service>
    <service
android:name="com.daps.weather.location.DapWeatherLocationsService">
    </service>
    <!--dapweather end-->
```

2.3 Obfuscate Code

Please refer to the chapter 4.3 in HW or CW version of DUADplatform SDK Access Guide to Obfuscate Code.

```
-keep public class com.daps.weather.notification.DapWeatherNotification {
  public <methods>;
}
-keep public class com.daps.weather.weathercard.DapWeatherView {
  public <methods>;
}
-keep public class com.daps.weather.DapWeatherActivity {
  public <methods>;
}
-keep public class com.daps.weather.service.DapWeatherMsgService {
  *;
}
-keep public class com.daps.weather.location.DapWeatherLocationsService {
  *;
}
-keep public class com.daps.weather.DapWeather {
  public <methods>;
}
-keep public class com.daps.weather.weathercard.DapWeatherEnterImageView
{
  public <methods>;
}
-keep public class com.daps.weather.floatdisplay.FloatDisplayController {
  *;
}
-keep public class com.daps.weather.base.SharedPrefsUtils {
  public static boolean isSuspensionOn(android.content.Context);
  public static void setSuspensionOn(android.content.Context,boolean);
}
-keep class com.daps.weather.bean.**{*;}
```

3. Initialization

Please refer to the chapter 5 in HW or CW version of DUADplatform SDK Access Guide to finish SDK initialization.

For accessing Weather Wizard ads, please add a “weather” tag in Json, and input your corresponding placement ID.

```
{
    "weather": [
      {
      "pid": "YOUR_DAP_PLACEMENT_ID"
       }
     ]
}
```

- **Method：**

Add a call to DapWeather.init() from onCreate in your Application class. Please make sure you also call DuAdNetwork.init() for DUAd SDK as well. (see chapter 5 in HW or CW version of DUADplatform SDK)

- **Interface Instruction：**

public static void init(Context context , String pidsJson);

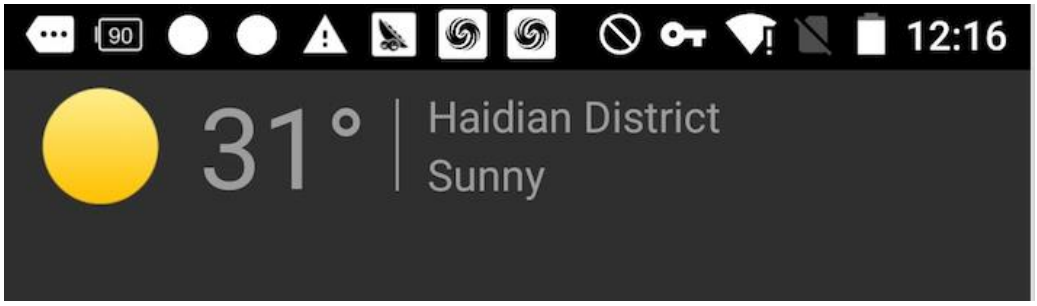
Parameters	Description
Context context	Activity Context
String pidsJson	DAP Placement ID

- **Code Sample：**

```
/**
  * Initialization Weather Wizard SDK
  * @param context Application Context
  * @param pidsJson configure json
  */
DapWeather.init((Application) getApplicationContext(),pidsJson);
```

4. Weather Card

Figure 1: Weather Card sample



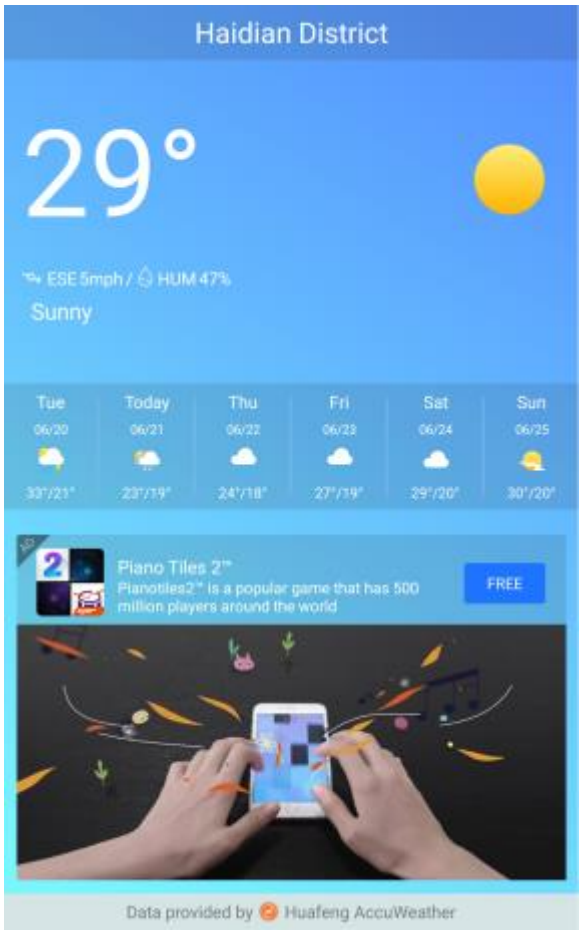
Add the **DapWeatherView** to your layout and use **load()** to load weather data.

- **Code Sample：**

```
//Add weather card View

 RelativeLayout rl = (RelativeLayout)

findViewById(R.id.demo_weather_view_rl);

mDuWeatherView = new DapWeatherView(this);

rl.addView(mDuWeatherView);



//load weather data, SDK will auto-fill DapWeatherView card if data is

loaded.

 mDuWeatherView.load();
```

After user click the weather card, it will jump to weather landing page automatically. See Figure 2.

Figure 2: Weather landing page



5. Weather floating window

Figure 3: weather floating window sample

FloatDisplayController.setFloatSerachWindowIsShow（Context context, Boolean shown）

Parameters	Description
Context context	Activity Context
Boolean shown	“true” for enable，”false” for disable

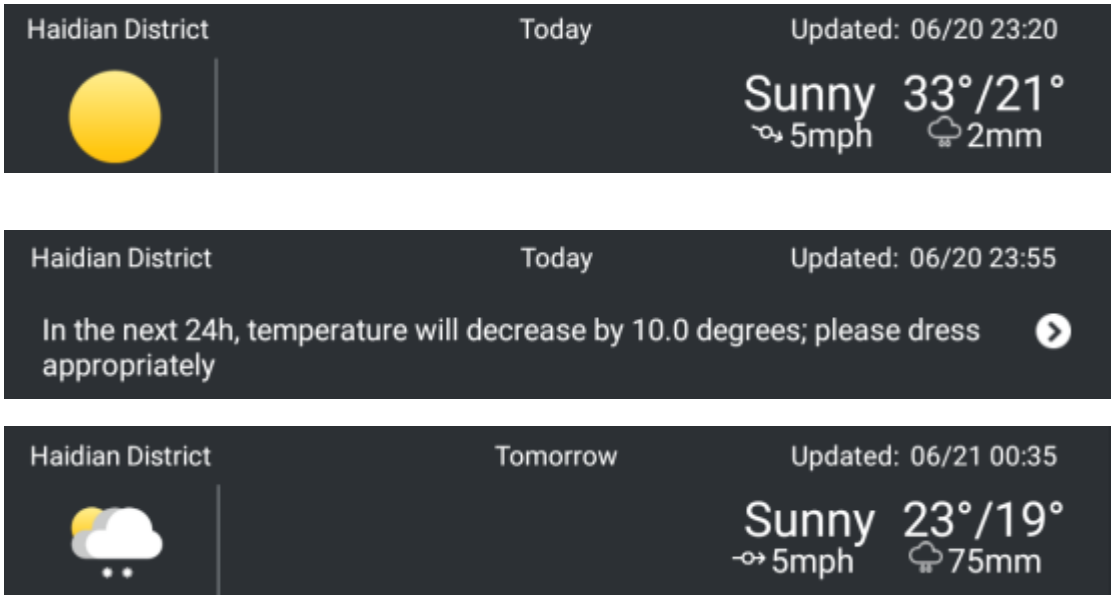
- **Code Sample：**

```
//Floating window entrance，input “true” to enable，input ”false” to disable.

FloatDisplayController.setFloatSerachWindowIsShow(getApplicationContext(),

true);



FloatDisplayController.setFloatSerachWindowIsShow(getApplicationContext(),

false);
```

After user click the weather floating window, it will jump to weather landing page automatically. See Figure 2.

6. Weather Notification bar

Figure 4: weather notification bar sample



The weather notification bar is controlled by DU weather wizard SDK, which has a timer to push notification, followed by the bellow rules.

7:30-12:00: push today's weather.

13:00-20.30: push temperature rise or drop.

20.30-23:00: push tomorrow's weather；

The notification will be pushed only for one time within each time period.

```
// Weather Notification， Du SDK has a timer to execute it automatically by default



mNotification=

DapWeatherNotification.getInstance(getApplicationContext());//Initialize
```

```
notification



mNotification.setOngoing(true);//”true” for Resident notification bar. “False” for

temporary notification bar

            mNotification.setPushElevatingTemperature(true);//Enable or disable

notification for temperature rise or drop，default value is true



mNotification.setPushTodayTomorrowWeather(true);// Enable or disable

notification for today and tomorrow's weather or not，default value is true
```

After user click the weather notification bar, it will jump to weather landing page automatically. See Figure 2.

7. Use weather landing page

You can also use the weather landing page directly:

startActivity(new Intent(MainActivity.this, DemoWeatherViewActivity.class));