**1st Scene 🡪**

It looks like you're trying to create a BroadcastReceiver that listens for changes to the airplane mode and shows a toast when the airplane mode is toggled. However, you are not seeing the expected message after changing the airplane mode.

Here are a few things to check and potentially fix in your code:

**1. Typo in the IntentFilter Action Name**

The action name in the IntentFilter is incorrect in your MainActivity. You have a typo:

IntentFilter intentFilter = new IntentFilter("andrid.intent.action.AIRPLANE\_MODE");

The correct action is:

IntentFilter intentFilter = new IntentFilter(Intent.ACTION\_AIRPLANE\_MODE\_CHANGED);

So, it should be updated to:

IntentFilter intentFilter = new IntentFilter(Intent.ACTION\_AIRPLANE\_MODE\_CHANGED);

**2. Manifest Registration: Double-check the <receiver> Tag**

In your AndroidManifest.xml, the receiver is registered to listen for the action android.intent.action.AIRPLANE\_MODE. However, the intent-filter inside the manifest should also use the correct action Intent.ACTION\_AIRPLANE\_MODE\_CHANGED.

<receiver

android:name=".AirplaneModeReceiver"

android:exported="true">

<intent-filter>

<action android:name="android.intent.action.AIRPLANE\_MODE\_CHANGED"/>

</intent-filter>

</receiver>

Ensure that the action is properly registered as android.intent.action.AIRPLANE\_MODE\_CHANGED instead of android.intent.action.AIRPLANE\_MODE.

**3. Correct Use of Intent Extras**

In your AirplaneModeReceiver, you are retrieving the state of the airplane mode with:

boolean isAirplaneModeOn = intent.getBooleanExtra("State: ", false);

However, the correct extra key for the state is "state", not "State: ". It should be:

boolean isAirplaneModeOn = intent.getBooleanExtra("state", false);

So, change that line to:

boolean isAirplaneModeOn = intent.getBooleanExtra("state", false);

**4. Ensure Dynamic Registration**

In your MainActivity, you're dynamically registering the broadcast receiver:

AirplaneModeReceiver broadcastReceiver = new AirplaneModeReceiver();

registerReceiver(broadcastReceiver, intentFilter);

This approach is correct for handling broadcasts at runtime. However, if you also want to handle broadcasts while the app is in the background, make sure to unregister the receiver when appropriate (e.g., in onPause or onDestroy), like so:

@Override

protected void onDestroy() {

super.onDestroy();

unregisterReceiver(broadcastReceiver); // Unregister the receiver to avoid memory leaks

}

**5. Airplane Mode Action and Permissions**

Ensure that your app has the necessary permissions to receive the AIRPLANE\_MODE\_CHANGED broadcast. Starting from Android 11 (API level 30), the ACCESS\_NETWORK\_STATE permission might be needed to properly access airplane mode status.

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

This permission is typically used to read the network state, but it can also help your app receive certain broadcasts.

**Final Code Summary:**

Here is a final version of your code after making the above corrections:

**MainActivity.java**

package com.example.broadcast;

import android.content.IntentFilter;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

// Correct action name for AIRPLANE\_MODE\_CHANGED

IntentFilter intentFilter = new IntentFilter(Intent.ACTION\_AIRPLANE\_MODE\_CHANGED);

// Register the receiver dynamically

AirplaneModeReceiver broadcastReceiver = new AirplaneModeReceiver();

registerReceiver(broadcastReceiver, intentFilter);

}

@Override

protected void onDestroy() {

super.onDestroy();

// Unregister the receiver to avoid memory leaks

unregisterReceiver(broadcastReceiver);

}

}

**AirplaneModeReceiver.java**

package com.example.broadcast;

import android.content.BroadcastReceiver;

import android.content.Context;

import android.content.Intent;

import android.widget.Toast;

public class AirplaneModeReceiver extends BroadcastReceiver {

@Override

public void onReceive(Context context, Intent intent) {

if (intent.getAction() != null &&

intent.getAction().equals(Intent.ACTION\_AIRPLANE\_MODE\_CHANGED)) {

// Correct extra key for airplane mode state

boolean isAirplaneModeOn = intent.getBooleanExtra("state", false);

String message = isAirplaneModeOn ? "Airplane Mode is ON" : "Airplane Mode is OFF";

// Show a toast message

Toast.makeText(context, message, Toast.LENGTH\_LONG).show();

}

}

}

**AndroidManifest.xml**

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

package="com.example.broadcast">

<application

android:allowBackup="true"

android:icon="@mipmap/ic\_launcher"

android:label="@string/app\_name"

android:roundIcon="@mipmap/ic\_launcher\_round"

android:supportsRtl="true"

android:theme="@style/Theme.Broadcast"

tools:targetApi="31">

<activity android:name=".MainActivity" android:exported="true">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<!-- Receiver to listen for AIRPLANE\_MODE\_CHANGED -->

<receiver

android:name=".AirplaneModeReceiver"

android:exported="true">

<intent-filter>

<action android:name="android.intent.action.AIRPLANE\_MODE\_CHANGED" />

</intent-filter>

</receiver>

</application>

</manifest>

**Debugging Tips:**

* **Check Logcat for Errors**: If it still doesn't work, check Logcat for any errors or warnings related to your BroadcastReceiver or manifest registration.
* **Use Debugging**: Set breakpoints or add log statements to see whether the receiver is being triggered properly.

With these changes, your app should display a toast when the airplane mode is toggled.

**2nd Scene 🡪**

**IntentFilter intentFilter = new IntentFilter(Intent.ACTION\_AIRPLANE\_MODE\_CHANGED);**

**getting result but when i turn on from off then also it dispalys off.. message not shows messhae for on..**

The issue you're describing suggests that your BroadcastReceiver is not properly handling the change from **OFF** to **ON** for Airplane Mode. It always shows the "Airplane Mode is OFF" message, even when you expect it to show "ON" when Airplane Mode is turned on.

**Likely Cause:**

In your AirplaneModeReceiver, you are checking the airplane mode state using:

boolean isAirplaneModeOn = intent.getBooleanExtra("state", false);

The key "state" is not always set for the Intent.ACTION\_AIRPLANE\_MODE\_CHANGED broadcast. The correct key to check the airplane mode state can vary between different Android versions.

**Solution:**

To fix this, you should modify how you're extracting the airplane mode state. You don't need to rely on intent.getBooleanExtra("state", false) in all cases. Instead, you can directly check the system's current airplane mode state using Settings.System (to handle all cases robustly).

Here's how to update your code:

**Updated AirplaneModeReceiver.java:**

package com.example.broadcast;

import android.content.BroadcastReceiver;

import android.content.Context;

import android.content.Intent;

import android.provider.Settings;

import android.widget.Toast;

public class AirplaneModeReceiver extends BroadcastReceiver {

@Override

public void onReceive(Context context, Intent intent) {

// Check if the broadcast is for AIRPLANE\_MODE\_CHANGED

if (intent.getAction() != null && intent.getAction().equals(Intent.ACTION\_AIRPLANE\_MODE\_CHANGED)) {

// Retrieve the current airplane mode state

boolean isAirplaneModeOn = Settings.System.getInt(context.getContentResolver(),

Settings.System.AIRPLANE\_MODE\_ON, 0) != 0;

// Display the appropriate message based on the state

String message = isAirplaneModeOn ? "Airplane Mode is ON" : "Airplane Mode is OFF";

// Show the message via Toast

Toast.makeText(context, message, Toast.LENGTH\_LONG).show();

}

}

}

**Explanation of Changes:**

1. **Checking Airplane Mode State**:
   * Instead of relying on the Intent extras (which may not be reliable or consistent across Android versions), I directly query the system settings using Settings.System.AIRPLANE\_MODE\_ON to determine whether airplane mode is enabled or disabled.
   * Settings.System.getInt(context.getContentResolver(), Settings.System.AIRPLANE\_MODE\_ON, 0) returns 1 if airplane mode is on, and 0 if it's off.
2. **Displaying the Toast Message**:
   * The Toast.makeText() method is used to show the appropriate message based on the current state of airplane mode.

**No Need for intent.getBooleanExtra():**

Since the Intent.ACTION\_AIRPLANE\_MODE\_CHANGED broadcast itself does **not** provide a reliable boolean extra (the key "state" doesn't always work correctly), querying the system directly is a more consistent approach.

**Double-check the Manifest:**

Just to ensure everything works correctly, verify that your AndroidManifest.xml is still properly set up:

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

package="com.example.broadcast">

<application

android:allowBackup="true"

android:icon="@mipmap/ic\_launcher"

android:label="@string/app\_name"

android:roundIcon="@mipmap/ic\_launcher\_round"

android:supportsRtl="true"

android:theme="@style/Theme.Broadcast"

tools:targetApi="31">

<activity android:name=".MainActivity" android:exported="true">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<!-- Receiver to listen for AIRPLANE\_MODE\_CHANGED -->

<receiver

android:name=".AirplaneModeReceiver"

android:exported="true">

<intent-filter>

<action android:name="android.intent.action.AIRPLANE\_MODE\_CHANGED" />

</intent-filter>

</receiver>

</application>

</manifest>

**Testing:**

* **Test turning airplane mode ON and OFF**:
  + Toggle airplane mode on and off and observe the Toast messages.
  + If it's still not working, you can add logging (Log.d()) or further debugging to verify that the receiver is being triggered properly.

Now, with this fix, your app should correctly display "Airplane Mode is ON" when airplane mode is turned on and "Airplane Mode is OFF" when it's turned off.