Cheat Code for testing on Android 9.0

S.No	App has functionality	Impact with 9.0 (Pie)	Test Case	Impacted Devices	Additional Information
1	Limited access to sensors in background	Android 9 limits the ability for background apps to access user input and sensor data.	If your app is running in the background on a device running Android 9, the system applies the following restrictions to your app: 1. Your app cannot access the microphone or camera. 2. Sensors that use the continuous reporting mode, such as accelerometers and gyroscopes, don't receive events. 3. Sensors that use the on-change or one-shot reporting modes don't receive events. 4. If your app needs to detect sensor events on devices running Android 9, use a foreground service.	All devices with android 9.0	A foreground service is a service that the user is actively aware of and isn't a candidate for the system to kill when low on memory. A foreground service must provide a notification for the status bar, which is placed under the Ongoing heading. This means that the notification cannot be dismissed unless the service is either stopped or removed from the foreground. **Note:-** If your app needs to detect sensor events on devices running Android 9, use a foreground service.
2	Restricted access to call logs	The **CALL_LOG** permission group introduced which has **READ_CALL_LOG, WRITE_CALL_LOG** and **PROCESS_OUTGOING_CALLS** . Previously, these permissions were located in the **PHONE** permission group.	1. if you create an app that requests to get permission to access the call logs through the **PHONE** group. 2. In android 9.0 it will **throw SecurityException.** 3. Remove this exception, make sure your app requests permission from **CALL_LOG** group and not the PHONE group.	All devices with android 9.0	This **CALL_LOG** permission group gives users better control and visibility to apps that need access to sensitive information about phone calls, such as reading phone call records and identifying phone numbers.
3	Restricted access to phone numbers	Apps running on Android 9 cannot read phone numbers or phone state without first acquiring the **READ_CALL_LOG** permission	1. if your app needs to read phone numbers from phone state, update app to request necessary permission. 2. To read numbers from the **PHONE_STATE** intent action, you need both the **READ_CALL_LOG** permission and the **READ_PHONE_STATE** permission. 3. To read numbers from **onCallStateChanged()** , you need the **READ_CALL_LOG** permission only.	All devices with android 9.0	Phone numbers associated with incoming and outgoing calls are visible in the phone state broadcast, such as for incoming and outgoing calls and are accessible from the PhoneStateListener class. Without the READ_CALL_LOG permission, however, the phone number field that's provided in PHONE_STATE_CHANGED broadcasts and through PhoneStateListener is empty.
4	Restricted access to Wi-Fi location and connection information	In Android 9, the permission requirements for an app to perform Wi-Fi scans are more strict than in previous versions.	1. create an app that has getConnectionInfo() method 2. it returns a WifiInfo object that describes current wifi connection. 3. to retrieve SSID and BSSID values we need the following permissions: 1. ACCESS_FINE_LOCATION or ACCESS_COARSE_LOCATION 2. ACCESS_WIFI_STATE.	All devices with android 9.0	Retrieving the SSID or BSSID also requires location services to be enabled on the device (under Settings > Location).
5	Information removed from Wi-Fi service methods	In Android 9, the following events and broadcasts don't receive information about the user's location or personally identifiable data: 1. The **getScanResults()** and **getConnectionInfo()** methods from **WifiManager** . 2. The **discoverServices()** and **addServiceRequest()** methods from **WifiP2pManager** . 3. The **NETWORK_STATE_CHANGED_ACTION** broadcast.	1. If your app needs information about the SSID, BSSID or the connection information 2. It is no longer available in **NETWORK_STATE_CHANGED_ACTION** broadcast from wifi. 3. To get this information use **getConnectionInfo()** method instead.	All devices with android 9.0	
6	Telephony information now relies on device location setting	If the user has disabled device location on a device running **Android 9,** the following methods don't provide results: 1. **getAllCellInfo** () 2. **listen**() 3. **getCellLocation** () 4. **getNeighboringCellInfo** ()		All devices with android 9.0	
7	Libraries removed from the framework	Android 9 reorganized JUnit based classes into 3 libraries: android.test.base, android.test.runner and android.test.mock.	if your app needs to run tests against a version of JUnit with your project's dependencies, it can be easily achieved with this change.	All devices with android 9.0	Note:- https://developer.android.com/training/testing/set-up-project
8	Test suite build changes	The addRequirements() method in the TestSuiteBuilder class has been removed, and the TestSuiteBuilder class itself been deprecated.		All devices with android 9.0	The addRequirements() method had required developers to supply arguments whose types are hidden APIs, making the API invalid. **Note:-** https://developer.android.com/reference/android/test/suitebuilder/TestSuiteBuilder
9	Screen rotation changes	Beginning with Android 9, there are significant changes to the portrait rotation mode. In Android 8.0 (API level 26), users could toggle between auto-rotate and portrait rotation modes using a **Quicksettings tile** or **Display settings**. **The portrait mode has been renamed rotation lock** and it is active when auto-rotate is toggled off. There are no changes to auto-rotate mode.		All devices with android 9.0	

Cheat Code for testing on Android 8.0

S.No	App functionality	Impact with 8.0 (Oreo)	Test Case	Impacted Devices	Additional Info
1	Tracks Current Location of the user (Less frequent background location updates)	Android has restricted the no. of location updates when the app uses background service for location update	1:- Open your application check for current location. 2:- Walk or Drive ahead (100 MTRs). Now again request from UI for location 3:- If you tried the step 2 a few times within a minute, you will see, it will give you the older location and not the updated one.Does it impact the location based feature in the app and the user experience?	All	If you have android 8.0 Device and you are using OLA/UBER then also you can find the same scenario. You will not get the Updated Cab/Auto location.
2	Background Execution Limits	Services running in the background can consume device resources, potentially resulting in a worse user experience. To mitigate this problem, the system applies a number of limitations on services	1:- If your app is targeting Android 8.0 and your app's using background services with older way we used in Android 5 & 6 or 7. 2:- Try using the functionality invokes the Service. 3:- startService() method now throws an IllegalStateException		You can check with your Dev team about which functionalities are using background services

3	Multi language Support/Locales and Internationalization	Change of Default category locale API	1:- If your app supports multiple language support, in android O language related changes have been done. ex: Locale.getDefault() 2:- Change the language of your phone ex:- French 3:- Launch your app and if you have used Locale.getDefault() method, that will return you null. 4:- Language will not be change.	All	
			Check time Zone conversion if your app shows date and time		

4	Web form autofill	New Auto fill feature	Open forms/webView in the App and check for Auto fill of fields 1:- If you app is targeting andorid O then you have to enjoy the Web form auto fill feature 2:- Run you app on Android O devcie fill the details on website, App or WebView. 3:- Android O will auto suggest you fill username, card details etc. 4:- Again visit the same page and fill the details, You will see the auto fill suggestions. 4:- If still you are not able to fill the details using Auto fill suggestions, then take a deep dive inside your code and update your code 5:- Build your app and test the code		

5	PIP Mode	PIP is a special type of multi-window mode mostly used for video playback.	1:- Open App and play a video. 2:- Change the mode to PIP. Video should continue playing. 3:- Change back to normal mode.		1:- These days Whatsapp, Google hangout, VLC are all PIP compatible. 2:- Picture-in-picture Support Android 8.0 (API level 26) allows activities to launch in picture-in-picture (PIP) mode. PIP is a special type of multi-window mode mostly used for video playback. It lets the user watch a video in a small window pinned to a corner of the screen while navigating between apps or browsing content on the main screen.

6	HTTPS and Network	HTTPS and Network	Test App functionalities using Proxy to Tunnel the request		
			Test App functionality which rely on third party plugin/browser for login- oAuth(Login through google,FB etc) and Enterprise SSO		
			Stress testing for funtionality using native device features like Camera, geosensors etc.		
			Test your App functionalities while switching off and on wifi.		
7	SSLv3	SSLv3 support removed	Test App functionalities using SSLv3		

8	Input and Navigation	Keyboard as Navigation input. Now you can use arrow- and tab-based navigation on App	1:- Install your app on a device that offers a hardware keyboard. If you don't have a hardware device with a keyboard, connect a Bluetooth keyboard or a USB keyboard (though not all devices support USB accessories). 2:- You can also use the Android emulator: In the AVD Manager, either click New Device or select an existing profile and click Clone. 3:- In the window that appears, ensure that Keyboard and DPad are enabled. 4:- To test your app, use only the Tab key to navigate through your UI, ensuring that each UI control gets focus as expected. 5:- Using Tabs if Its not focusing on controls, You have to update your app on Target API 28	Tabs and Higer form factor devices	Android supports physical keyboards attached to the device
9	Bluetooth	Change in API related to data transfer	Test Functionalities of Your App where bluetooth data sync or data transfer is required	All	

10	Alert windows	Apps that use the **SYSTEM_ALERT_WINDOW** permission can no longer use the following window types to display alert windows above other apps and system windows	1:- If your app is targeting Android O , you have to test the behaviour below Alert Windows. 2:- If an app targets Android 8.0 (API level 26), the app uses the TYPE_APPLICATION_OVERLAY window type to display alert windows. Window type **-TYPE_PHONE** Phone. These are non-application windows providing user interaction with the phone (in particular incoming calls). These windows are normally placed above all applications Window type -**TYPE_PRIORITY_PHONE** Priority phone UI, which needs to be displayed even if the keyguard is active. These windows must not take input focus, or they will interfere with the keyguard. Window type -**TYPE_SYSTEM_ERROR** System window, such as low power alert. These windows are always on top of application windows. In multiuser systems shows only on the owning user's window.	All	