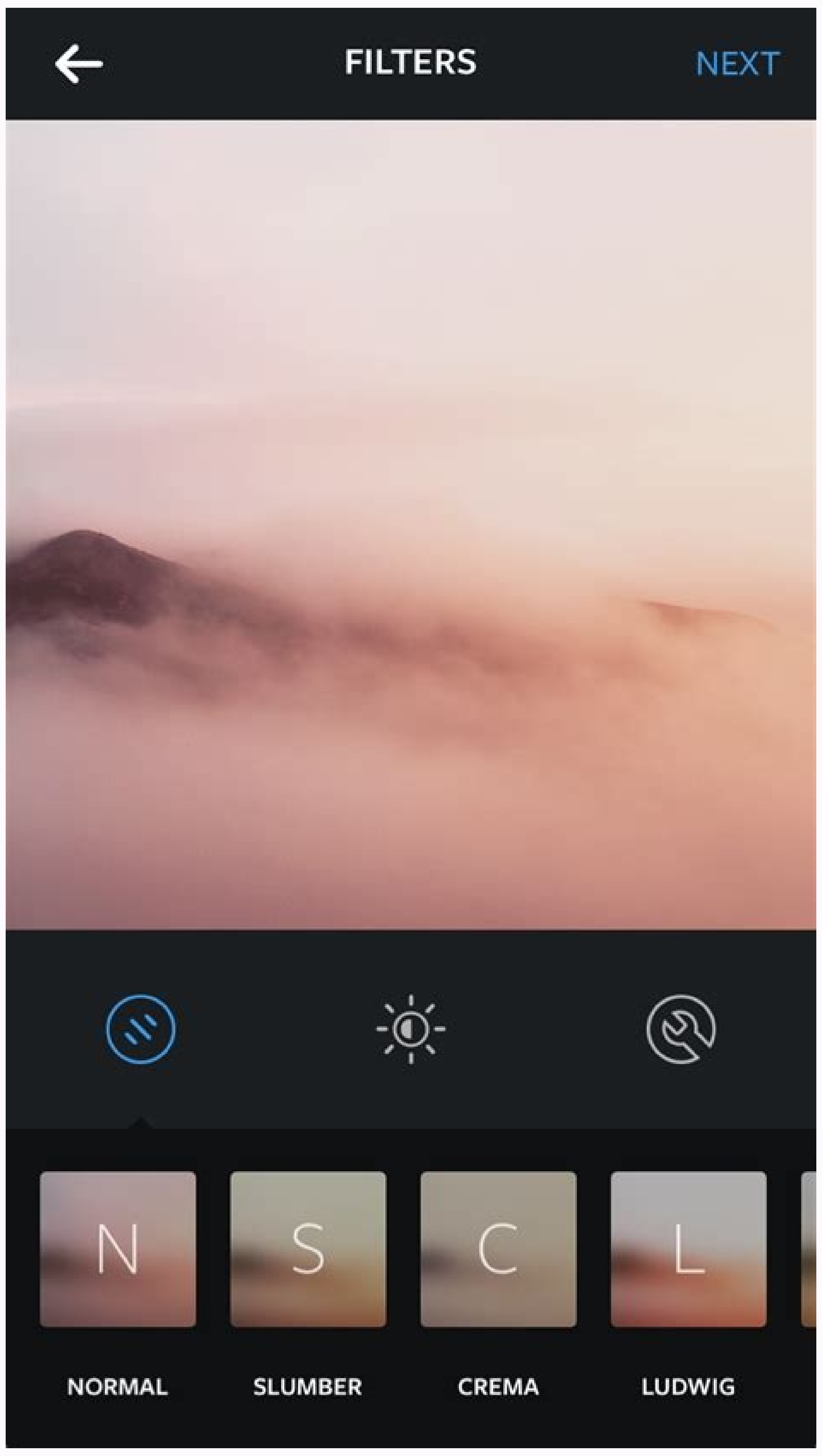
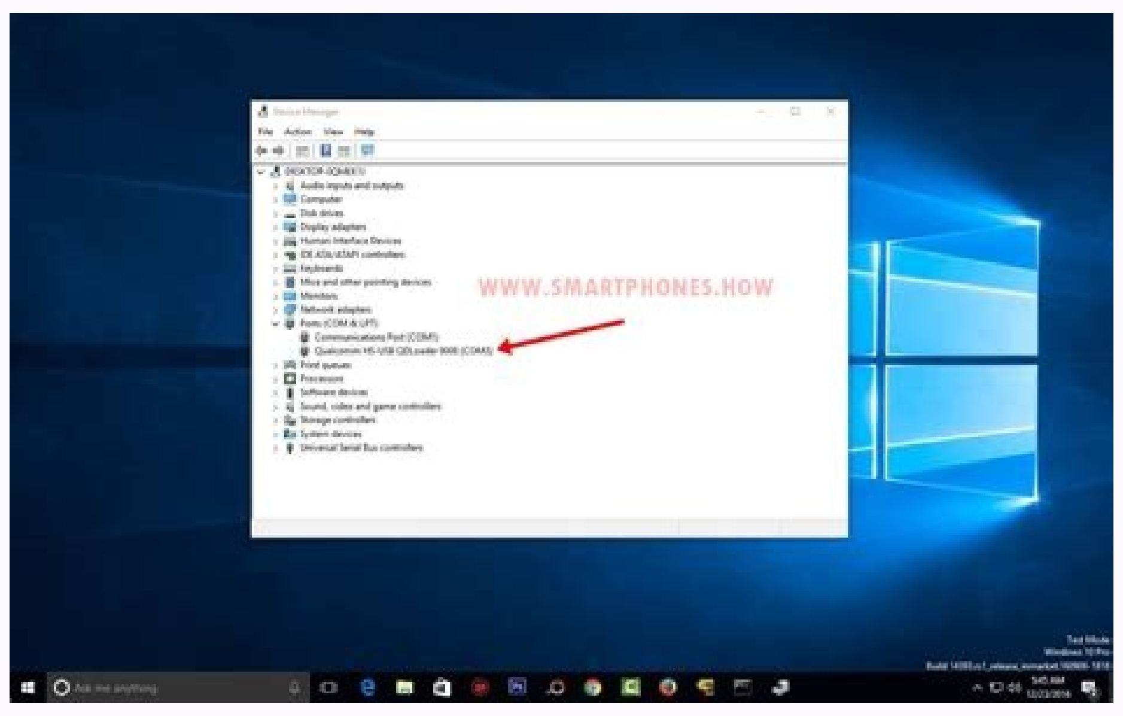
Xamarin android device name

I'm not robot

reCAPTCHA

Continue





Android device name. Xamarin android get device name. Difference between xamarin forms and xamarin android. Android display name.

The problem has recently arisen. Studio 2019, 2022 The Xamarin Android project is launched every twice. Android devices, both the emulator and the physical device, are not displayed. I tried to delete the trash and the OBJ files. However, it helped to completely reinstall the investigation. When the project is closed, the situation recurs. The same project in two different environments. The project is completely pure xamarin. ADB Recharge Menu Disabled. Another answer works well. But I added more details. The program name is selected by the file androidmanifest.xml, for example, strings.xml file. Do not cord the program name here. Good practice is to distinguish all strings.xml file used in the lines. Strings.xml Program Name The label feature disregards and determines the program name, the name of the APK to the value of the label determined in the MaynACTIVity questionnaires. It is recommended to eliminate the value of the label from the ASSEMBLY MAINACIVITIES.CS ABOUT [activity (because it ignores this implicit value when debuting) changes the meaning of strings.xml to be universal and consistent. With Xamarin and Windows, you have access to the same device information: GeteAppid: Used to generate unique ID of your program. ID: This is the device model device ID: Get the device model version: You will get an OS version if you need the device name, it is not on this list. So we can create our own features. First of all we have to create an interface, I call it idevice. using the system; Names area MyApp. Interfaces {public interface Idevice {String Device (); }} Method GetDevicame () returns the device name. This method will need to be implemented by iOS and Android. So in this project I added a new addiction, for example: to use the system; using myApp.ios; Using myApp. interfaces; [ASSEMBLY: Xamarin.forms.Dependency (Typeof (Device_ios))]] Names of MyApp }} To get the device name in the iOS system, use the UIDEVICE Class for Uikit. In the case of Android, I added the following code to the project: using a system; Using Android.bluetooth; Using myApp.droid; Using myApp. interfaces; Using Xamarin.forms [Build: Deendency (Typeof (Device_droid))]] Names MyApp.droid.Dependenties {Public Class Device_droid: Idevice {// // // Get DeviceName /// /// Title. Generally accessible line Devicename () {Bluetooth adapter MyDevice = Bluetooth adapter.defaultadapter; returns mydevice.name; }}} To get the name of the device on Android is a little difficult, because there is no public API for this. You can get the BluetoothAdapter class; However, to use the BluetoothAdapter class, you must add Bluetooth resolution to your Android project. Then I need to add my main project and cause addiction. VAR Platform = DependenCyLERVICE.GET (); Line name = platform.getdevicename (); Successful coding! It is described in detail here. But you really do not need to do this and try to get the identifier of each device. The creation and preservation of the leadership on your device also works. Xamarin prefers to maintain the value of the device in the future. You can create Guid and save it in prefectures, as I did below: Var Dickeid = Preferences.get ("My_deviceid", String.empty); if (string.isnullorwhitspace (bone identifier)) {Deviceid = System.guid.newguid (). To string(); Settings.set ("My_deviceid", "Dicealid"); } The advantage of this approach is that you will always have the same identifier when transferring the application to another device; But if you delete and reinstall, you will receive a new identifier. In other cases, when you receive an identifier from one device, it changes when transferring the application to another device. In other cases when you want to get the device identifier: iOS: IdentifirefirDevice Public String ID => Uidevice.currentVice.ididtifierForvertor.asstring (); Android: Serial Line Identifier, Getserial and Androidid = String.empty; Public identifier of the line {Get {if (! String.isnullorwhitespace (ID)) Return ID; ID = Android.os.Build.serial; if (string.isnullorwhitespace (id) || ID == constructions ID = Secure.getString (CONTEXT.ContentResolver, Secure.androidid); } Catch (Exception EX) {Android.util.log.warn ("Deviceinfo", "failed to get the id:" + ex.tostring ()); }} Returned identifier; }} Uwp: getpackageSpecifikenken or GetsyStymidforpublisher string ID = NULL; A public line identifier {get {if (id! = null) return; Try {if (apiinformation.istypepresent ("Windows.system.profile.systemidification")) {Var Systemid = SystemidificationIfixation.getSySteforpublisher (); // Make sure that this device can generate IDS if (systemid.Source! = SystemidtificationSource.none) {// ID property has a buffer with a unique identifier VarpingWareid = Systemid.id; Data reader VAR =var byte = nový bajt [hardwareID.Length]; dataReader.ReadBytes(byte); id = Convert.ToBase64String(bytes); } } if (id == null && ApiInformation.IsTypePresent("Windows.System.Profile.HardwareIdentification")) { var token = HardwareIdentification.GetPackageSpecificToken(null); var id hardwaru = token.Id; var dataReader = Windows.Storage.Streams.DataReader.FromBuffer(hardwareId); var byte = nový bajt [hardwareID.Length]; dataReader.ReadBytes(byte); id = Convert.ToBase64String(bytes); } if (id == null) { id = "nepodporováno"; } } catch (Výjimka) { id = "nepodporováno"; } return id; } } }