Allow Always BEST

I've tried to choose the option "While using app" and wait to the next prompt in order to choose "Allow always" but it doesn't work. Choosing "While using app" and going to settings for changing the option to "Allow always" doesn't work. (Both cases are the "official" answer of Apple in the documentation) This is very annoying from a MDM-perspective, since a location-app that must be used now constantly asks the user if it should still be allowed to use "Always allow". The admins need this to be on "Always allow" so we can track the iPads in case they get lost/stolen. Is there a way to always allow Google Chrome to use my microphone and camera for a certain site? I like using the new Google Voice Search but whenever I click the mic button it asks if Chrome is allowed to use my mic, how can I set it to always allow? Yes, click the "third" button. I don't remember the wording, but it was something like "don't ask again". I clicked it, wondering if this means "always deny" or "always allow", and it turned out to be "always allow". At least I think so. I can use Link and can access files on the Quest. I've seen a "don't ask again" checkbox a few times, it's not consistent. Personally, I'm not too bothered by it asking if I want to enable Link. Sometimes I want Link on right away, other times I was just plugging in the headset to charge it while watching Pluto TV or chatting with friends in Alt Space. What bugs me more is it asking if it's okay to allow my PC access to my Quest 2 Every Single Time I plug it in. By mistake, I clicked on "always allow" on a file in my keychain on the Mac. How do I undo this and restrict access? I tried to correct it by clicking on Access Control. There I selected "confirm before allowing access" but it didn't work. I can still see the password for that file without entering a password. As you can only choose between "Allow Apps from the App Store" and "Allow Apps from the App Store and identified developers" in the Security Panel of macOS Sierra, I wonder if there is a way to always allow installing from apps from unidentified developers too, other than rightclicking the Installer and hit "Open". Yes thanks, I think I understood that. My question was really because I don't recall having to make that choice in 1P7, so what was 1P7 doing.....did it effectively have "always allow for every site". I'll be interested to hear what others have to offer here. My take is that, in order to be effective and "just work", it makes sense to allow on all sites. This is because 1) 1Password in your browser only reads the web data in order to fill passwords for you (we don't make our money from advertising, we don't collect sketchy information on you, it's only used for the purposes of doing what you bought 1Password for in the first place, and so on) and 2) the 1Password browser extension is designed with security in mind. To be more accurate, this is not a default of 1Password 8 that we've decided to make, but instead a choice Apple has made with Safari Web Extensions. Other browsers merely offer a choice between "don't install the extension" or "allow the extension to be installed, and access all sites". Safari instead requires "Always allow on Every Website" to be a choice to actively make after installing an extension that would like to do this. As I mentioned above, enabling "Always Allow on Every Website" would give you the smoothest and least interrupted experience. I've always used requestAlwaysAuthorization() for this at the time where the user signals their desire to set up a geofence (the apps functionality is they get a notification when they enter this geofence with some up to date data about it - there is a settings screen where they can choose to start receiving this notification, if they choose to do this, at this point I request Always permission so I can set the geofence) I notice from 1:11:36 in the 2019 Platforms State of the Union - and from testing in the Xcode 11 Beta that calling request**Always**Authorization() now no longer gives the user the user the option to allow always. It only gives them the choice of While Using App, or allow once. The user apparently now has to grant the "While Using App" permission first, then according to the state of the union the app must ask for "Always" location later from the background. They didnt elaborate on how you will do this from the background or when you should. I'm just wondering how this is going to work for geofences? I want to be able to set up the geofences within the app as this is the best experience for the user, rather than them having to leave the app to give background permission, then re-enter the app to set up which geofences they want. It also seems a poor experience for them to have to grant the permission twice - once for in use, then again for always, when I just want Always permission. I'm in a similar boat. I'm not sure how you get woken up in the background when entering a monitored region, to ask for 'always' location services to get woken up in the background in the first place. Seems a bit chicken and egg. Hopefully this is a beta bug, and maybe anyone can confirm that the intention that for step 3, both the **didChangeAuthorization delegate** method, and **CLLocationManager.authorizationStatus()** will come back with .always while the app is in provisional Always state? Yeah, I'm seeing that if you call requestWhenInUseAuthorization() and then subsequently call requestAlwaysAuthorization(), nothing happens on the subsequent call. Even though its always been the case that calling requestAlwaysAuthorization() after the initial call for whenInUse should ask the user to upgrade. I dont think they mentioned this in the talk though? And the Apple documentation pages for these methods don't seem to have changed at all according to the diffs - so I'm not sure what the expected flow is. Given that the documentation hasnt changed though, and that Apple have always made it clear that you should only request what you need - it doesnt sound right to me that the subsequent call to requestAlwaysAuthorization() does nothing. Seems like either the behaviour or the documentation is wrong. Yes that's what I didn't understand when they said that in the talk as when I tried it on Tuesday I was getting back whenInUse authorization even though I asked for always and the guy said you should get back temporary always. I am very concerned the user experience for apps which do require always authorization is a bit pants now to say the least. If you app says it will do something for the user in the background and it doesn't because iOS hasn't gotten around to asking them for always permission and the delegate doesn't get the first few notifications through because of the delay this isn't a great first impression for the user of your app. I don't like Apple second guessing developers. If there is a reason we want to ask for always authorization then let us, and if we get user push back then fine. But its worse not being able to do something because of Apple imposed limitations on what they believe is right. Why have two methods askForAlways and askForWhenInUse when they do the same thing now? "I am very concerned the user experience for apps which do require always authorization is a bit pants now to say the least. If you app says it will do something for the user in the background and it doesn't because iOS hasn't gotten around to asking them for always permission and the delegate doesn't get the first few notifications through because of the delay this isn't a great first impression for the user of your app." Annoying if you go to the Settings screen for you app 'provisional always' shows as 'Always', its only when the user drills in to change it do they see that they really have 'While Using the App' permission granted. Can't even easily ask users to go in and change it manually as they will think they have granted Always looking at that. In iOS 13 we are under the impression that, in addition to the new "provisional" always authorization, it should still be possible to directly prompt users for always authorization, assuming that they had granted when in use authorization first. "If your app has never requested always authorization and its current authorization status is CLAuthorizationStatus.authorizedWhenInUse, you may call this method one time to try and change your app's authorization status to CLAuthorizationStatus.authorizedAlways." More importantly, in keeping with the spirit of other changes in iOS 13, we also think that it gives end users more transparency than the "provisional" always flow, as it lets developers provide additional context in-app before the prompt is displayed (e.g., a screen in an onboarding with illustrations, and not just the plist string displayed in the prompt). The element of the element specifies a unique URL that request filtering will always allow. The element contains a collection URLs that request filtering will allow, which override the values in the collection. The following sample illustrates a combination of a element and an element that will deny any URLs if they contain either of two specific character sequences, but will always allow a specific URL that contains both of those two specific character sequences in a particular order. @joshnystrom Hello! Thanks for sending your feedback! I can see we have it on our backlog to update the AlwaysOpenPdfExternally policy to allow an exception list but there is no ETA for when this will be available. For situations like this where you have no worries of downloading something dangerous, you can eliminate that pop-up message asking your permission. This tutorial shows you how to always allow downloads in Safari on your Mac. By default, Unattended Access is disabled on the AnyDesk client and will not allow unattended connections to the device. In this case, connection requests **need to be manually accepted or rejected** using the Accept Window of the client being connected to.

Allow Always

21f597057a