



1





Android Developers

▸ Android Development Patterns

2015年9月18日

Understanding what Doze mode means for your app

Pro-tip by +Joanna Smith

Marshmallow introduced a new way for devices to preserve battery life by going idle. As an app developer, this means that there is now a difference between your app falling out of the foreground and the device itself being idle.

Before you start to panic about how to wake the device from Doze mode (<https://goo.gl/eiAJbo>), I recommend taking a moment to figure out if you even need to bust through that idleness.

Doze mode is different in that the *entire device* is taking a nap, not just the least-recently used apps (such is the case with App Standby). **Doze mode is activated when the device is not plugged in, when the screen is locked, and when no motion has been detected for some time.** These assumptions imply that the user has put down the device and walked away. So, when they come back and pick their phone up off the counter, it should not have dropped to 30% battery. It should, essentially, be in the exact state it was in when they placed it on the counter. Right?

So, the exceptions to Doze mode should be truly exceptional. There are two situations that I believe would qualify. First, if you have new information that is worth causing the user to walk over and look at their phone. Doze mode honors **high-priority messages through Google Cloud Messaging** (<https://goo.gl/v8l1RH>), so that you *can* respond to important moments, such as that new picture Mom just sent. Just don't use it as an excuse to download a huge amount of data: try instead including the information needed to display a notification in the push message itself, and then fetch everything else when the user returns (and Doze mode ends).

And secondly, you may have work to do that is time-sensitive. But even AlarmManager (<http://goo.gl/HDqEwd>) is not exempt from Doze mode. However, just because the alarm will not fire during Doze mode does not mean it is lost. For example, an alarm set with **setExact()** will actually not trigger while Dozing, but will fire as soon as Doze mode exits. And for most use cases, this is actually fine, because if the user isn't even holding their device, you probably don't need to be syncing right now. But if you do have a task that is worth draining the user's battery when they aren't there, you can rely on **setExactAndAllowWhileIdle()** to wake your app even during Doze mode. Check out our previous pro-tip (<https://goo.gl/IENTID>) on working with alarms and background work for more details.

Ideally, Doze won't disrupt your app flow too much. But spend some time thinking carefully about when your app may need special consideration. And after making any necessary changes, you can spend some more time **testing your app**. We've created several super helpful adb commands (<https://goo.gl/8PGom5>) to make it easy to see how your app recovers when coming out of Doze mode.

So happy Dozing, developers. And continue to [#BuildBetterApps](#)

翻译

Power-Saving Optimizations

goo.gl

+1

261

187

90

公开分享 · 查看动态

查看 181 条以前的评论