How can I tell whether the user has disabled toast notifications for my app?

devblogs.microsoft.com/oldnewthing/20200810-00

August 10, 2020

Raymond Chen

A customer wanted to know how to get the complete list of programs from the *Settings*, *System*, *Notifications & Actions*, *Get notifications from these senders* list, as well as whether the user has enabled notifications for each app.

Upon closer questioning, it turns out that this isn't really what they wanted. They didn't need the complete list of apps and the user's preference for each one. When asked what they were going to do with the information, they said that they were going to search through the list looking for their app, to see whether notifications are enabled. If notifications are disabled for their app, then they are going to warn the user, "You need to enable notifications for Contoso Sports to receive game alerts."

In a sense, this was a case of the "for/if" antipattern: Instead of asking for the thing that they actually want, they ask for *everything* and then try to filter to the thing that they want.

The way to find out whether notifications are enabled for your app is to check the `ToastNotifier.Setting` property. Here's some sample code:

```
// C#
var notifier = ToastNotificationManager.CreateToastNotifier();
var setting = notifier.Setting;

// C++/WinRT
auto notifier = ToastNotificationManager::CreateToastNotifier();
auto setting = notifier::Setting();

// C++/CX
auto notifier = ToastNotificationManager::CreateToastNotifier();
auto setting = notifier->Setting;

// JS
var notifier = ToastNotificationManager.createToastNotifier();
var setting = notifier.setting;

// VB
Dim notifier = ToastNotificationManager.CreateToastNotifier()
Dim setting = notifier.Setting
```

The resulting value in `setting` tells you whether toasts are enabled, or if not, why not. You can use this to display specific instructions to the user.

```
switch (setting) {
case NotificationSetting.Enabled:
    // everything is great.
    break;

case NotificationSetting.DisabledForApplication:
    Warn("Please go to Settings, System, Notifications & Actions " +
         "and enable this application in the "
         "'Get notifications from these senders' list.");
    break;

case NotificationSetting.DisabledForUser:
    Warn("Please go to Settings, System, Notifications & Actions " +
         "and set "
         "'Get notifications from apps and other senders' to On.");
    break;

case NotificationSetting.DisabledByGroupPolicy:
    Warn("Your system administrator has prevented us from " +
         "showing notifications.");
    break;

case NotificationSetting.DisabledByManifest:
    Warn("Oops. We forgot to ask the operating system for permission " +
         "to display toast notifications. Please file a bug.");
    break;

// Catch-all case for reasons that are defined
// in future versions of Windows.
default:
    Warn("It doesn't look like toast notifications are enabled, " +
         "but we don't know why.");
    break;
}
```

Note that this is even better than what the customer requested, because it also detects other cases, such as where the user left the notification enabled for your app, but then went and disabled all notifications globally.

Bonus chatter: You could go the extra mile and launch the Settings app directly to the *Notifications & Actions* page.

```
await Launcher.LaunchUriAsync("ms-settings:notifications");
```

Raymond Chen

Follow