Gestion des permissions dans Android

Introduction

Les permissions servent à protéger la confidentialité d'un utilisateur dans un appareil Android.

Les applications Android doivent demander l'autorisation d'accéder aux données sensibles de l'utilisateur. Parmi ces données sensibles, on peut citer les contacts.

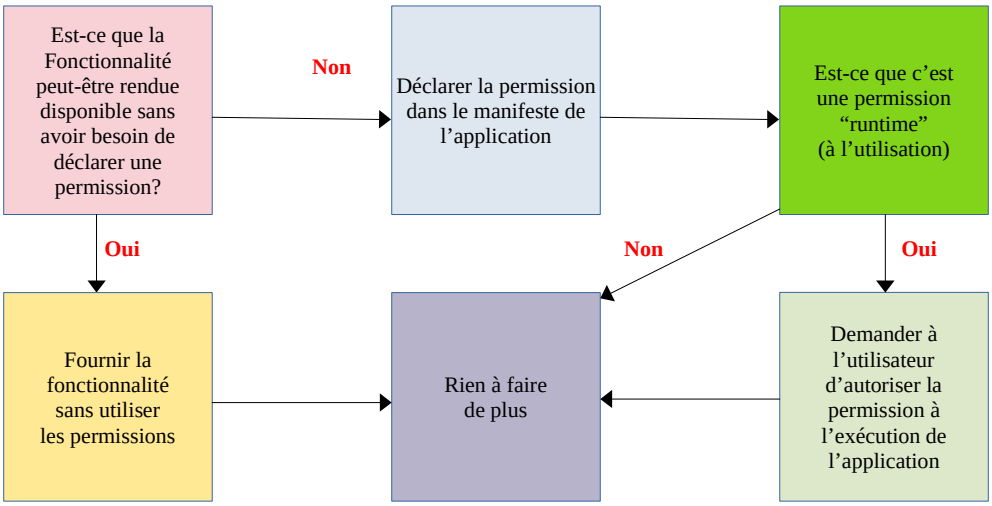
Les applications Android doivent demander aussi l'autorisation d'accéder à certaines fonctionnalités du système comme l'appareil photo, l'internet, etc.

Selon la fonctionnalité, le système peut accorder la permission automatiquement ou peut inviter l'utilisateur à approuver la demande.

Aucune application n'est autorisée à effectuer des opérations qui auraient un impact négatif sur les autres applications, le système d'exploitation ou l'utilisateur.

On peut citer par exemple la lecture ou l'écriture des données privées de l'utilisateur (telles que les contacts ou les courriels), la lecture ou l'écriture des fichiers d'une autre application, la création d'un accès réseau, le maintien de l'appareil en veille, etc.

Flux d’utilisation des permissions



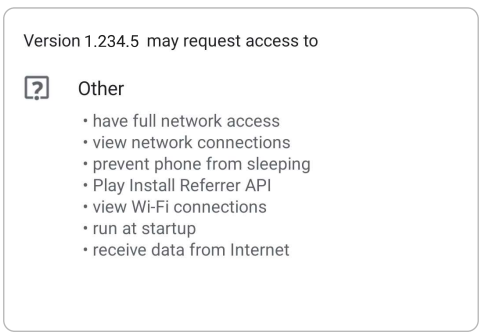
Niveaux de permission

Les permissions sont divisées en plusieurs niveaux de protection. Le niveau détermine la nature de l’accès ainsi que les demandes d'autorisation d'exécution sont requises ou non.

Permission à l’installation

Ces permissions donnent un accès limité et restreint à des données. Elles permettent aussi à l’application de réaliser des tâches sans affecter le système ni les autres applications installées sur l’appareil.

Ces permissions sont mentionnées à l’utilisateur à travers une notice au moment de l’installation de l’application.



Nous distinguons deux sous-catégories :

Permission normale

Une permission est dite normale quand elle représente un risque très faible pour la confidentialité de l'utilisateur ou le fonctionnement d'autres applications.

La permission d'accéder à l'internet est une autorisation normale.

Si une application déclare dans son manifeste qu'elle a besoin d'une permission normale, le système lui attribue automatiquement cette permission au moment de l'installation.

Le système ne demande pas à l'utilisateur d'accorder les permissions normales.

Les utilisateurs ne peuvent pas révoquer ces permissions.

Permission de signature

Le système accorde ces permissions à l'application au moment de l'installation, mais uniquement lorsque l'application qui tente de les utiliser est signée par le même certificat que l'application qui définit ces permissions.

C'est le type le plus "robuste" de signature, car elle nécessite l'utilisation d'une clé cryptographique.

Les applications utilisant ce type de permissions sont toutes contrôlées par le même développeur.

C'est le système qui décide s'il est nécessaire de faire intervenir l'utilisateur.

Ces permissions sont généralement utilisées par les applications du système.

Permission à l’exécution (« runtime »)

Elles sont dites aussi dangereuses.

Elles souhaitent obtenir des données ou des ressources impliquant les informations privées de l'utilisateur ou susceptibles d'affecter les données stockées de l'utilisateur ou le fonctionnement d'autres applications.

Par exemple, la possibilité de lire les contacts de l'utilisateur.

Si une application déclare avoir besoin d'une permission dangereuse, l'utilisateur doit explicitement l'accorder.

Jusqu'à ce que l'utilisateur approuve la permission, votre application ne peut pas fournir de fonctionnalités qui dépendent de cette permission.

Pour utiliser une permission dangereuse, votre application doit inviter l'utilisateur à accorder cette permission au moment de l'exécution.

Approbation de la permission

Une application doit publier les autorisations requises en incluant des bases <uses-permission> dans le manifeste de l' application.

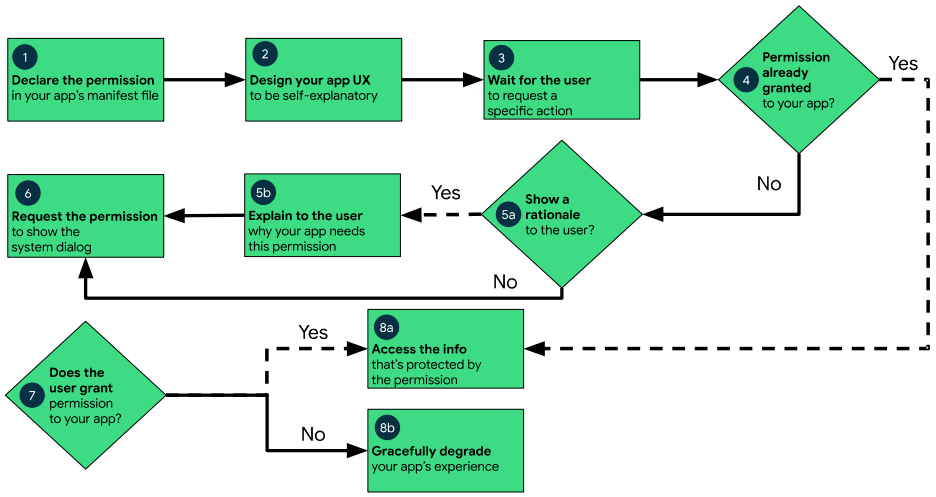
Si une application doit envoyer des messages SMS, elle doit inclure cette permission dans le fichier manifeste :

```
<manifest xmlns: android = "http://schemas.android.com/apk/res/android"
        package = "com.example.snazzyapp" >

   <uses-permission android: name = "android.permission.SEND_SMS" />

   <application ... >
       ...
   </ application>

</ manifest>
```

Si l'autorisation demandée dans le fichier manifeste ne présentant pas un risque élevé pour la confidentialité de l'utilisateur ou le fonctionnement du périphérique, le système accorde automatiquement cette autorisation à votre application.

Si par contre l'autorisation demandée représente un risque, affecte la confidentialité de l'utilisateur ou le fonctionnement normal du périphérique, tel qu'envoyer un SMS par exemple, l'utilisateur doit explicitement accorder cette autorisation.

Flux d'une demande de permission

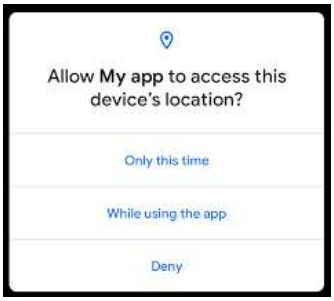


Permissions de groupe

Permission Group	Permissions
CALENDAR	READ_CALENDAR WRITE_CALENDAR
CAMERA	CAMERA
CONTACTS	READ_CONTACTS WRITE_CONTACTS GET_ACCOUNTS
LOCATION	ACCESS_FINE_LOCATION ACCESS_COARSE_LOCATION
MICROPHONE	RECORD_AUDIO
PHONE	READ_PHONE_STATE CALL_PHONE READ_CALL_LOG WRITE_CALL_LOG ADD_VOICEMAIL USE_SIP PROCESS_OUTGOING_CALLS
SENSORS	BODY_SENSORS
SMS	SEND_SMS RECEIVE_SMS READ_SMS RECEIVE_WAP_PUSH RECEIVE_MMS
STORAGE	READ_EXTERNAL_STORAGE WRITE_EXTERNAL_STORAGE

Autorisation unique

Depuis Android 11+, il est possible d'autoriser l'application à utiliser la permission réclamée une seule fois, suite à une première demande, ou bien tant que l'application est utilisée (en mémoire) ou rejeter la demande.



Remise à zéro des permissions autorisées

Depuis Android 11+, si l'application n'est pas utilisée depuis plusieurs mois, le système va automatiquement remettre à zéro les permissions autorisées par l'utilisateur.

Rejet définitif d'une autorisation

Depuis Android 11+, si l'utilisateur a rejeté plusieurs fois la permission demandée par une application donnée, le système n'affichera plus une nouvelle demande d'autorisation de cette application.

Le système considère que l'utilisateur a simulé par son action, ne me pose plus cette question.

Gestion des permissions dangereuses

Si votre appareil cible Android 6.0 ou plus (API >= 23), le système d'autorisation des permissions fonctionne comme suit :

- Si l'application n'a pas déjà la permission demandée du groupe demandé, le système affiche un dialogue à l'utilisateur. Ce dialogue va contenir la permission de groupe dont l'application a demandé l'autorisation d'accès. Il n'affiche pas spécifiquement la permission demandée, mais uniquement le groupe dans lequel cette permission appartient. Exemple, si l'application demande à avoir accès à la permission « READ_CONTACTS », le système va informer que l'application veut accéder à la permission du groupe « CONTACTS ». Si l'utilisateur approuve la permission du groupe, le système va autoriser que la permission demandée.

- Si l'application a déjà la permission de groupe, le système ne va pas informer l'utilisateur de cette demande, il va l'accorder de facto. Exemple, si l'application a demandé la permission « READ_CONTACTS », et l'utilisateur l'accepte. Il va accorder la permission de groupe « CONTACTS ». Si par la suite cette même application va demander la permission « WRITE_CONTACTS », le système va l'accorder de facto sans poser la question à l'utilisateur.

Attention : cette logique dans la façon d'accorder les permissions va probablement changer dans le futur.

Caution: Future versions of the Android SDK might move a particular permission from one group to another. Therefore, don't base your app's logic on the structure of these permission groups.

For example, `READ_CONTACTS` is in the same permission group as `WRITE_CONTACTS` as of Android 8.1 (API level 27). If your app requests the `READ_CONTACTS` permission, and then requests the `WRITE_CONTACTS` permission, don't assume that the system can automatically grant the `WRITE_CONTACTS` permission.

Le précédent passage n'existe plus dans la documentation de Google. Est-ce que le comportement a changé pour autant? La seule façon pour le vérifier est de demander la permission pour lire un contact, l'autoriser, et voir si de facto vous avez la permission d'enregistrer de nouveaux contacts!

Afficher les permissions d'une application

- À travers l'interface

application Paramètres

- Via adb

adb shell dumpsys package net.learn2develop.makecalls

```
   requested permissions:
     android.permission.CALL_PHONE
       User 0: ceDataInode=-4294951552 installed=true hidden=false suspended=false stopped=false notLaunched=false enabled=0 instant=false virtual=false
     runtime permissions:
       android.permission.CALL_PHONE: granted=true
```

Références

Exemple de la démo « Nicolas Hurtubise »
https://github.com/316k/IFT1155-Permissions

https://developer.android.com/guide/topics/permissions/index.html

https://material.io/design/platform-guidance/android-permissions.html

https://android.googlesource.com/platform/frameworks/base/+/master/core/java/android/permission/Permissions.md

https://inthecheesefactory.com/blog/things-you-need-to-know-about-android-m-permission-developer-edition/en