UNIVERSITÉ CÔTE D'AZUR

UTILISATION DE CAPTEURS SUR ANDROID – GESTION DES PERMISSIONS

@.fR Frédéric RALLO

FONCTIONNEMENT GÉNÉRAL

- Chaque application Android s'exécute dans un sandBox
- Si une application doit utiliser des ressources ou des informations en dehors de son sandbox, l'application doit demander l'*autorisation* appropriée
- Vous devez déclarez que votre application pourrait besoin d'une autorisation en répertoriant l'autorisation dans le manifest

AUTORISATIONS DANS LE MANIFEST

- **Sur toutes les versions d'Android, vous devez placez un élément dans le manifeste de l'application**
- **Il faut le placer en tant qu'enfant de l'élément de niveau supérieur.**
- **Par exemple, une application qui doit accéder à Internet aurait cette ligne dans le manifeste :**

```
<manifest xmlns : android = "http://schemas.android.com/apk/res/android"
      package = « edu.demo.ihm" >

   <uses-permission android : name = "android.permission.INTERNET" />
   <! - d'autres autorisations vont ici ->

   <application ... >
      ...
   </application>
</manifest>
```

AUTORISATIONS DANS LE MANIFEST

- **Le comportement du système après avoir déclaré une autorisation dépend de la sensibilité de l'autorisation.**
- **Certaines autorisations sont considérées comme «normales», de sorte que le système les accorde immédiatement lors de l'installation.**
- **Les autres autorisations nécessitent que l'utilisateur doit explicitement accorder l'accès à votre application.**
- **Il existe plusieurs niveaux de protection qui affectent les applications tierces :**
  - les autorisations normales ,
  - les autorisations de signature
  - les autorisations dangereuse
  - les autorisations spéciales

CAS DES AUTORISATIONS NORMALES

- Cela s'applique à toute application qui a besoin d'accéder à des données ou des ressources en dehors du sandBox de l'application,
- Lorsqu'il y a très peu de risques pour la confidentialité de l'utilisateur ou le fonctionnement d'autres applications. Par exemple, l'autorisation de définir le fuseau horaire est une autorisation normale.
- Si une application déclare dans son Manifeste qu'elle a besoin d'une autorisation normale, le système accorde systématiquement cette autorisation à l'application au moment de l'installation.
- Le système n'invite pas l'utilisateur à accorder des autorisations normales et les utilisateurs ne peuvent pas révoquer ces autorisations.

CAS DES AUTORISATIONS SIGNÉES

- C'est une technique de cryptographie asymétrique utilisant des clés publiques et privées.
- Le système accorde ces autorisations d'application au moment de l'installation
- Une autorisation signée peut être accordée si et seulement si l'application qui tente d'utiliser cette autorisation est elle-même signée (par le même certificat que l'application qui définit l'autorisation…)
- L'application qui utilise une autorisation et qui voudrait communiquer avec une autre application qui aurait besoin de la même autorisation doit être signé par la même clé (même signature).

CAS DES AUTORISATIONS DANGEREUSES

- **Pour les applications qui souhaitent accéder à des données ou à des ressources qui impliquent les informations privées (de l'utilisateur),** ou qui pourraient potentiellement affecter les données stockées de l'utilisateur ou le fonctionnement d'autres applications.

- **Par exemple, la possibilité de lire les contacts de l'utilisateur est une autorisation dangereuse.**

- **L'utilisateur doit explicitement accorder l'autorisation à l'application.**
  - Tant que l'utilisateur n'approuve pas l'autorisation, l'application ne peut pas fournir de fonctionnalités qui dépendent de cette autorisation.
  - L'application doit inviter l'utilisateur à accorder l'autorisation au moment de l'exécution.
  - L'application doit prévoir le cas où l'utilisateur n'accorde pas l'autorisation.

CAS DES AUTORISATIONS SPÉCIALES

- **Il existe quelques autorisations qui ne se comportent pas comme des autorisations normales et/ou dangereuses.**

- **SYSTEM_ALERT_WINDOW et WRITE_SETTINGS sont particulièrement sensibles, donc la plupart des applications ne devraient pas les utiliser.**
  - permet à une application d'afficher des éléments graphiques en superposition par-dessus les autres applications.
  - Accorde à l'application un accès privilégié et la possibilité d'interagir avec l'interface utilisateur du système.
  - Peut potentiellement être utilisée de manière abusive (afficher des publicités intrusives, des fenêtres contextuelles trompeuses ou voler des informations à l'utilisateur).

- **Si une application a besoin de l'une de ces autorisations, elle doit déclarer l'autorisation dans le Manifeste et envoyer une intention demandant l'autorisation de l'utilisateur.**

- **Le système répond à l'intention en affichant un écran de gestion détaillé à l'utilisateur.**

OBTENIR L'ACCORD DES AUTORISATIONS (DANGEREUSES)

- **La façon dont Android demande à l'utilisateur d'accorder des autorisations dangereuses dépend de la version d'Android exécutée sur l'appareil de l'utilisateur et de la version du système ciblée par votre application.**

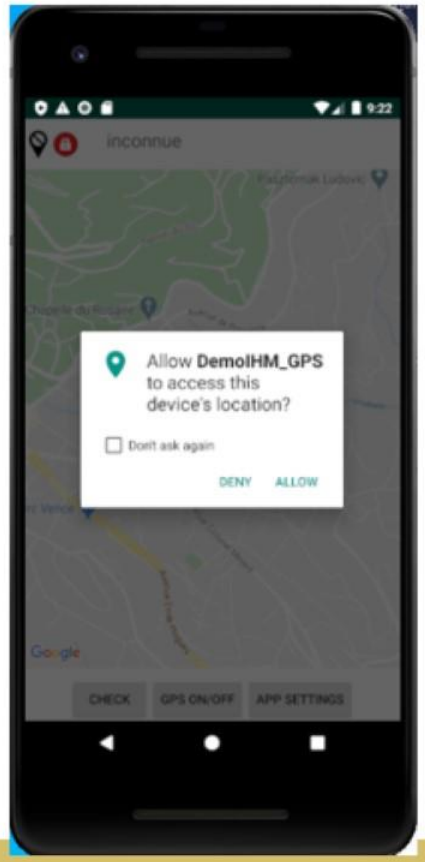
- **Avant Android 6.0 (API niveau 23) l'utilisateur est simplement informé des autorisations requises au moment de l'installation. Il doit seulement les approuver pour autoriser l'application à s'installer**

- **Si l'appareil exécute Android 6.0 (API niveau 23) ou plus, l'utilisateur n'est plus informé des autorisations d'application au moment de l'installation. mais application doit demander à l'utilisateur d'accorder les autorisations dangereuses lors de l'exécution.**

  - **Lorsque votre application demande une autorisation, l'utilisateur voit une boîte de dialogue système indiquant à l'utilisateur à quel groupe d'autorisations votre application tente d'accéder.**
  - **La boîte de dialogue comprend un bouton Refuser et Autoriser .**

Même si l'utilisateur accorde à votre application l'autorisation demandée, il garde la possibilité de révoquer une ou plusieurs autorisations dans les paramètres système.

Vous devez toujours vérifier et demander des autorisations lors de l'exécution pour vous prémunir contre les erreurs d'exécution



COMMENT UTILISER LES AUTORISATIONS

- **Vérifier que l'on dispose bien du matériel requis dans le Manifest**
- **Déclarer les autorisations dans le Manifest (pour informer à l'installation)**
- **Demander l'autorisation si l'application ne dispose pas déjà de l'autorisation**

- **En cas de refus,**
  - Expliquer pourquoi l'application à besoin d'autorisation
  - (Re)demander ?!? les autorisations « dangereuses » à l'utilisateur
- À partir d'Android 6.0 (API niveau 23), les utilisateurs peuvent révoquer les autorisations de n'importe quelle application à tout moment, même si l'application cible un niveau d'API inférieur. Donc, même si l'application a utilisé la caméra hier, elle ne peut pas supposer qu'elle dispose toujours de cette autorisation aujourd'hui.

VÉRIFIER QUE LE MATÉRIEL NÉCESSAIRE EST BIEN DISPONIBLE

- **L'accès à certaines fonctionnalités matérielles (telles que Bluetooth ou l'appareil photo) nécessite une autorisation d'application.**
- **Cependant, tous les appareils Android n'ont pas réellement ces fonctionnalités matérielles.**
- **Donc, si l'application demande l'autorisation CAMERA, il est important de vérifier la possibilité via <uses-feature> dans le manifeste**

```
<uses-feature android:name="android.hardware.camera" android:required="false" />
```

APPROBATION DES AUTORISATIONS DANS LE MANIFEST

- **Une application doit publier les autorisations dont elle a besoin en incluant des balises <uses-permission> dans le manifeste de l'application.**

- **Par exemple, une application qui doit envoyer des messages SMS aurait cette ligne dans le manifeste:**

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package=« edu.demo.ihm">

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

    <application ...>
        ...
    </application>
</manifest>
```

- autorisations *normales* → le système accorde automatiquement ces autorisations

- autorisations dangereuses → l'utilisateur doit explicitement accepter d'accorder ces autorisations.

DEMANDER LES AUTORISATIONS (DANGEREUSES)

- Il faut demander l'autorisation seulement si celle-ci n'a pas déjà été octroyée (PERMISSION_GRANTED)
- La méthode asynchrone ContextCompat.checkSelfPermission() vérifie que l'activité déclarée dans le Manifest n'est pas révoquée par l'utilisateur.

```
If (ActivityCompat.checkSelfPermission(getContext(), Manifest.permission.ACCESS_FINE_LOCATION)
            != PackageManager.PERMISSION_GRANTED ) {
            ActivityCompat.requestPermissions(context,
                        new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, REQUEST_CODE); }

else {
            //Do the stuff that requires permission...
     }
```

- `REQUEST_CODE` est le code de retour unique que l'on veut associer à cette demande (on peut en faire d'autres)
- La demande concerne un tableau de permissions (chacune étant un String)
- Le résultat de la vérification sera alors retourné dans la méthode `onRequestPermissionsResult()`

DEMANDER LES AUTORISATIONS (DANGEREUSES)

- Il s'agit ensuite d'interpréter le résultat de la décision de l'utilisateur dans onRequestPermissionsResult()
  - Si l'application dispose de l'autorisation, la méthode renvoie PERMISSION_GRANTED et l'application peut poursuivre l'opération.
  - Si l'application ne dispose pas de l'autorisation, la méthode renvoie PERMISSION_DENIED et l'application doit explicitement demander l'autorisation à l'utilisateur.

```
@Override
public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {
    switch (requestCode) {
        case REQUEST_CODE : // check user decision here!
                            break;
    }
}
```

- `REQUEST_CODE` est le code de retour associé à une demande
- `permissions` est un tableau de string représentant chacune une permission qui a été demandée
- `grantResults` est un tableau de string représentant chacune une réponse à une permission qui a été demandée

EXPLIQUER POURQUOI L'APPLICATION A BESOIN D'AUTORISATIONS (DANGEREUSES)

- **Lorsqu'une autorisation a été refusée, il est de bon usage d'expliquer pourquoi l'application a besoin d'autorisation.**

```
@Override
public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {
  switch (requestCode) {
    case REQUEST_CODE :
      if (grantResults.length > 0) {
        if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {
            //Do the stuff that requires permission...
        } else if (grantResults[0] == PackageManager.PERMISSION_DENIED) {
            // display an explanation
            if (ActivityCompat.shouldShowRequestPermissionRationale( getApplicationContext(),
                                    Manifest.permission.PERMISSION_DENIED)) {
            } else {
                //Never ask again selected, or device policy prohibits the app from having that
permission.
                //So, disable that feature, or fallut back to another situation...
            }
        }//end decision is DENIED
      }
   }; break; //end case
  }//end switch
}
```

Ne jamais oublier que l'utilisateur peut changer les autorisations à tout moment !

INFORMATIONS COMPLÉMENTAIRES

- **Attentions, en plus des autorisations, il se peut que vous soyez amené à contrôler si un capteur est allumé ou non (par exemple le Wifi, Bluetooth, GPS…)**