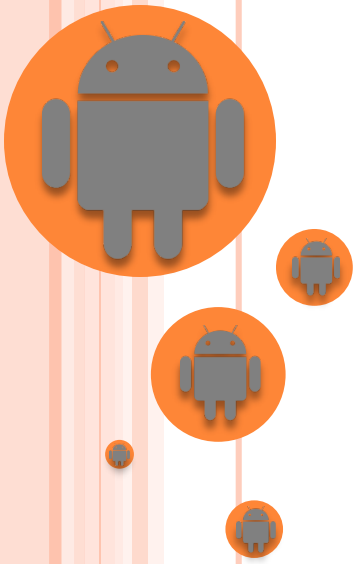
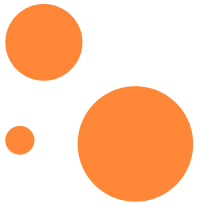


Permisos y Requisitos



Permisos

- Como medida de seguridad, las aplicaciones Android deben pedir permiso explícito para hacer uso de muchas de las características de los dispositivos:
  - Vibración
  - Internet
  - Cámara de fotos
  - Lector de huellas
  - ...
- Los permisos se dividen en:
  - Normales
  - Tiempo de ejecución (Runtime) o Peligrosos
  - De firma
  - Especiales

Permisos

- Los permisos normales son aquellos que no implican un riesgo para el usuario del dispositivo
- El sistema operativo los concede directamente, sin preguntar al usuario
- El usuario no puede revocarlos
- Por ejemplo:
  - El acceso a Internet
  - El acceso a la vibración
  - El acceso al lector de huella dactilar

Permisos

- Los permisos peligrosos son aquellos que implican cierto riesgo para el usuario del dispositivo
  - Acceso a datos privados (contactos, localización, etc.)
  - Acceso a servicios que cuestan dinero (envío de SMS, de llamadas, etc.)
  - ...

Permisos

- Los permisos (de ambos tipos) se declaran explícitamente en el manifiesto de la aplicación a través del elemento *uses-permission*

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.tema6">
    <uses-permission android:name="android.permission.READ_SMS"/>
    <application
        ...
</manifest>
```

- Listado de permisos disponibles:
  - https://developer.android.com/reference/android/Manifest.permission.html

Permisos

- Se le pregunta al usuario explícitamente si desea conceder cada uno de esos permisos a la aplicación
- En Android 5.1.1 o inferior
  - Se solicita la aprobación de todos los permisos peligrosos en el momento de instalarla
    - Si el usuario no los acepta, la aplicación no se instala
- En Android 6 (API 23) o superior
  - Se solicita la aprobación de cada permiso peligroso en tiempo de ejecución cuando se vaya a usar
    - El usuario puede aceptarlo o denegarlo
    - También puede cambiar su decisión a través de los ajustes

Hay que gestionarlos correctamente y comprobar siempre si en ese momento tenemos concedido el permiso o no

Permisos

- Los permisos peligrosos están agrupados
  - ACTIVITY_RECOGNITION
  - CALENDAR
  - CALL_LOG
  - CONTACTS
  - CAMERA
  - LOCATION
  - MICROPHONE
  - PHONE
  - SENSORS
  - SMS
  - STORAGE

Al usuario no se le pregunta por un permiso concreto, si no por un grupo de permisos. Si el usuario acepta, no se le pregunta por más permisos del mismo grupo.
Aún así, hay que pedir y gestionar todos los permisos individualmente

PERMISOS

- Comprobar si un permiso está concedido

```
 if (ContextCompat.checkSelfPermission(this, Manifest.permission.READ_SMS)!=
                                                    PackageManager.PERMISSION_GRANTED) {
        //EL PERMISO NO ESTÁ CONCEDIDO, PEDIRLO
        …
    }
    else {
        //EL PERMISO ESTÁ CONCEDIDO, EJECUTAR LA FUNCIONALIDAD
        …
    }
}
```

PERMISOS

- Pedir un permiso

Devuelve True si el usuario denegó el permiso anteriormente SIN pulsar la opción de "No preguntar de nuevo"

```
if (ActivityCompat.shouldShowRequestPermissionRationale(this,
Manifest.permission.READ_SMS))
{
    // MOSTRAR AL USUARIO UNA EXPLICACIÓN DE POR QUÉ ES NECESARIO EL PERMISO Y QUÉ
CARACTERÍSTICAS SE QUEDAN DESACTIVADAS SI NO SE ACEPTAN. AÑADIR BOTÓN "NO, GRACIAS" O
"CANCEL" PARA QUE EL USUARIO CONTINUE SIN LOS PERMISOS
    …
}
else{
    //EL PERMISO NO ESTÁ CONCEDIDO TODAVÍA O EL USUARIO HA INDICADO
    //QUE NO QUIERE QUE SE LE VUELVA A SOLICITAR
    …
}
//PEDIR EL PERMISO
ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.READ_SMS},
                                                                    CODIGO_DE_PERMISO);
```

Contexto

CODIGO_DE_PERMISO es un código numérico decidido por el desarrollador que identifica qué permiso se ha solicitado

Permisos

- Si el usuario ha indicado que no se le vuelva a preguntar e insistimos, podemos guiarle hasta la pantalla de los ajustes.
  - Pero no va en favor de la usabilidad

```
Intent intent = new Intent(Settings.ACTION_APPLICATION_DETAILS_SETTINGS);
intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
Uri uri = Uri.fromParts("package", getPackageName(), null);
intent.setData(uri);
startActivity(intent);
```

Permisos

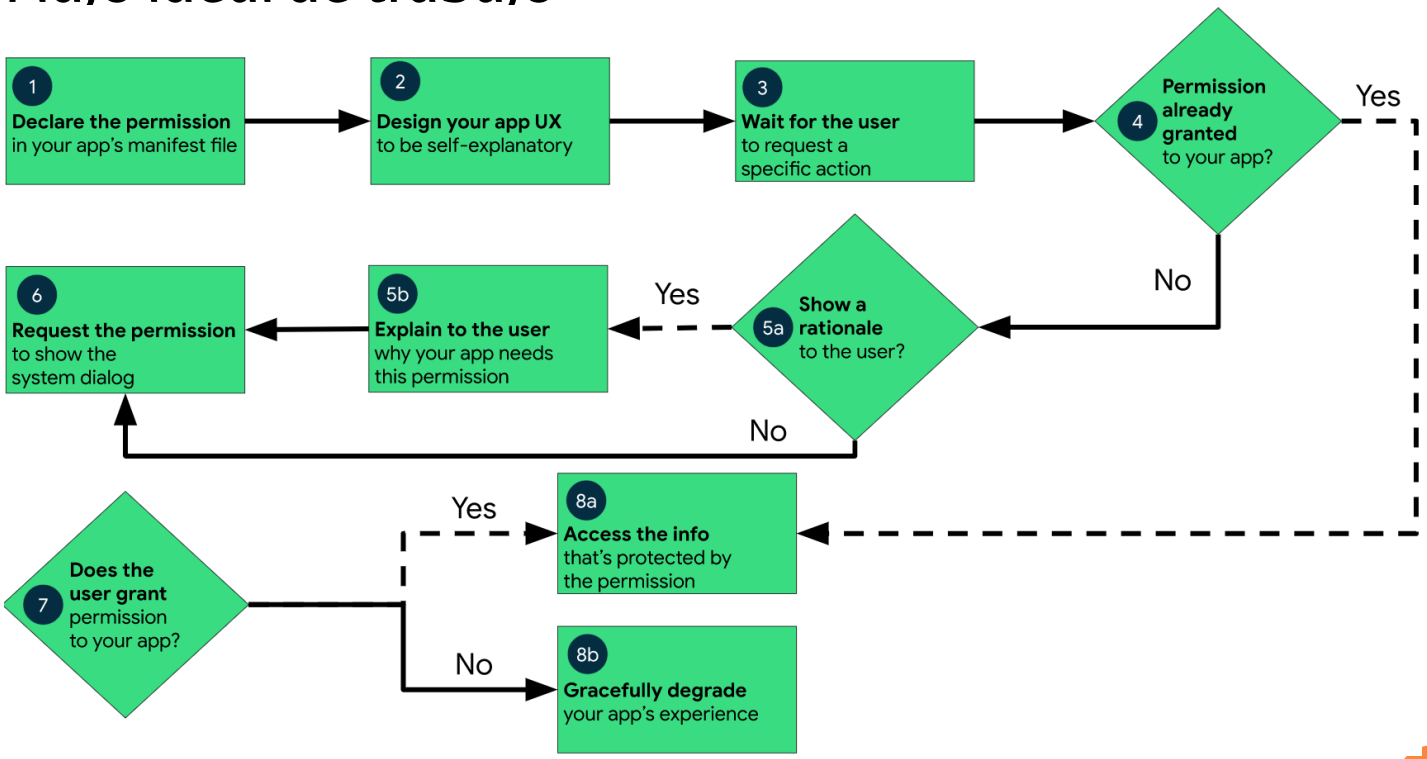
- Una vez que el usuario responde al diálogo de permisos, el sistema invoca el siguiente método para gestionar la decisión del usuario.

Método a sobreescribir

```
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
                                                          @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    switch (requestCode){
        case CODIGO_DE_PERMISO:{
            // Si la petición se cancela, granResults estará vacío
            if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                // PERMISO CONCEDIDO, EJECUTAR LA FUNCIONALIDAD
                …
            }
            else {
                // PERMISO DENEGADO, DESHABILITAR LA FUNCIONALIDAD O EJECUTAR ALTERNATIVA
                …
            }
            return;
            }
        case OTRO_CODIGO_DE_PERMISO:{
            …
        }
    }
}
```

Permisos

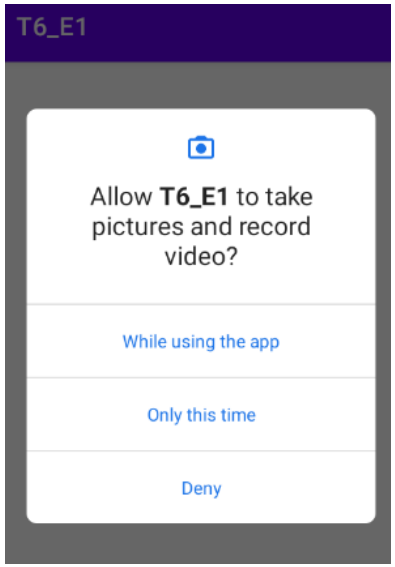
- Gestionar la usabilidad
  - Flujo ideal de trabajo



  - Mas información:
    - https://developer.android.com/training/permissions/requesting#workflow_for_requesting_permissions

Permisos

- Cambios en Android 11
  - El sistema muestra la opción "sólo esta vez" cuando se solicitan permisos relacionados con:
    - Ubicación
    - Micrófono
    - Cámara
  - Si el usuario elige esta opción, la aplicación obtiene el permiso temporalmente.
  - No implica cambios en el código.
  - Más información:
    - https://developer.android.com/about/versions/11/privacy/permissions#one-time



Permisos

- Cambios en Android 11
  - Si el usuario elige “Rechazar” un permiso más de una vez durante la vida de una aplicación*, no se le vuelve a mostrar el diálogo de solicitud de permiso de nuevo
    - Se interpreta automáticamente como “no preguntar de nuevo”.
  - En versiones anteriores, el usuario tendría que haber elegido “no preguntar de nuevo”.
  - No implica cambios en el código.
  - Más información:
    - https://developer.android.com/about/versions/11/privacy/permissions#dialog-visibility
  - *Se entiende ”vida de aplicación” como todo el tiempo desde que se instala en el teléfono, independientemente del número de veces que se use.

Permisos

- Cambios en Android 11
  - Si la aplicación no se usa durante unos meses*, Android automáticamente elimina los permisos peligrosos que el usuario hubiera concedido.
    - Equivale a que el usuario hubiera entrado en los ajustes y seleccionado "denegar" de forma manual.
  - No implica cambios en el código.
  - Más información:
    - https://developer.android.com/training/permissions/requesting#auto-reset-permissions-unused-apps
  - *La documentación no especifica cuántos.

Permisos

- Ejercicio 1: Cread una aplicación que muestre el histórico de llamadas del dispositivo
  - El permiso necesario es READ_CALL_LOG.
  - La función *historicoLlamadas()* disponible en el fichero de comandos devuelve un String con toda la información del histórico de llamadas.
    - Necesita los permisos antes de ejecutarse

Requisitos

- Sirven para declarar funciones hardware o software que usa la aplicación
  - Cámara de fotos
  - Bluetooth
  - …
- Se puede indicar que esa función es obligatorio tenerla o que se usa, pero no es obligatorio tenerla

Requisitos

- Los requisitos se declaran explícitamente en el manifiesto de la aplicación a través del elemento *uses-feature*

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.tema6">
    <uses-permission android:name="android.permission.READ_SMS"/>
    <uses-feature
        android:name="android.hardware.bluetooth"
        android:required="true"/>
    …
</manifest>
```

Si es "true", Android supone que la app requiere el HW de manera obligatoria y puede que no se instale si no existe el HW, si es "false" es recomendable.

- Listado de funciones disponibles https://developer.android.com/guide/topics/manifest/uses-feature-element.html?hl=es-419

Requisitos

- Se puede comprobar si el dispositivo tiene alguna característica hardware determinada

```
boolean tieneBluetooth =
getPackageManager().hasSystemFeature(PackageManager.FEATURE_BLUETOOTH);
boolean tieneCamera =
getPackageManager().hasSystemFeature(PackageManager.FEATURE_CAMERA);
```

- Listado de features disponibles https://developer.android.com/reference/android/content/pm/PackageManager

Requisitos

- Google Play filtra las aplicaciones para cada dispositivo en función de las características obligatorias que tenga
- Si está definida como requerida (true) y el dispositivo no dispone de ese elemento, no podrá instalar la aplicación