

# **CMSC436: Programming Handheld Systems**

# Permissions

# Today's Topics

Android permissions

Defining and using permissions

Component permissions and related APIs

# Permissions

Permissions protects resources and data

For instance, they limit access to:

- User information – e.g, Contacts

- Cost-sensitive API's – e.g., SMS/MMS

- System resources – e.g., Camera

# Permissions

Permissions are represented as strings

Apps describe relevant permissions in  
AndroidManifest.xml, including

- Permissions they use

- Permissions required of components that want to interact with them

# Using Permissions

Applications specify permissions they use through a `<uses-permission>` tag

These permissions must be granted before access is allowed

Apps must check at runtime that all required permissions have been granted

# Using Permissions

```
<manifest ... >
```

```
...
```

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

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

```
<uses-permission
```

```
    android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

```
...
```

```
</manifest >
```

See: <https://developer.android.com/training/permissions/index.html>

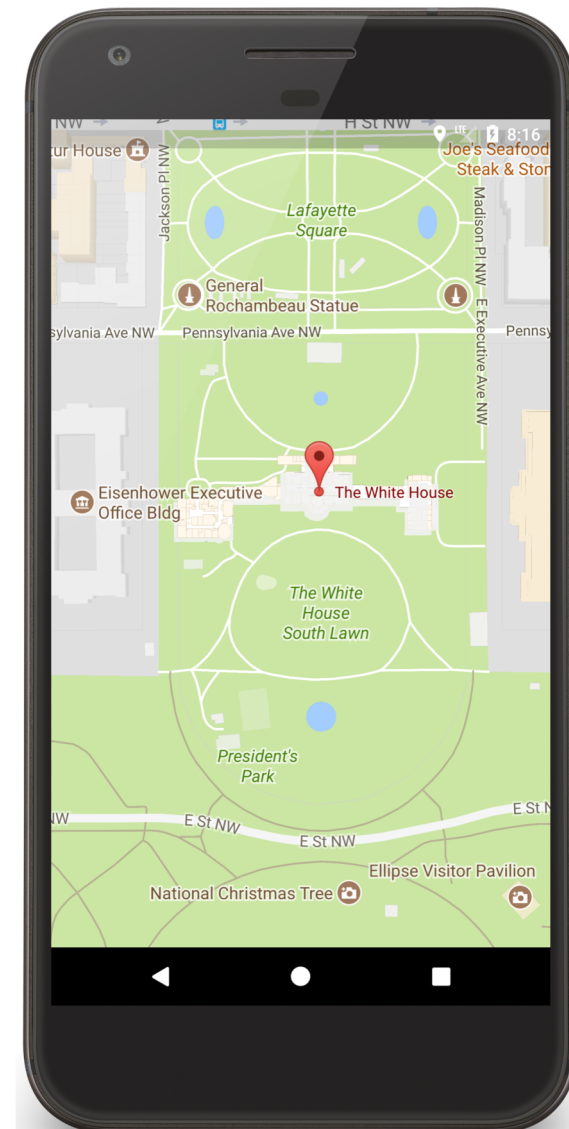
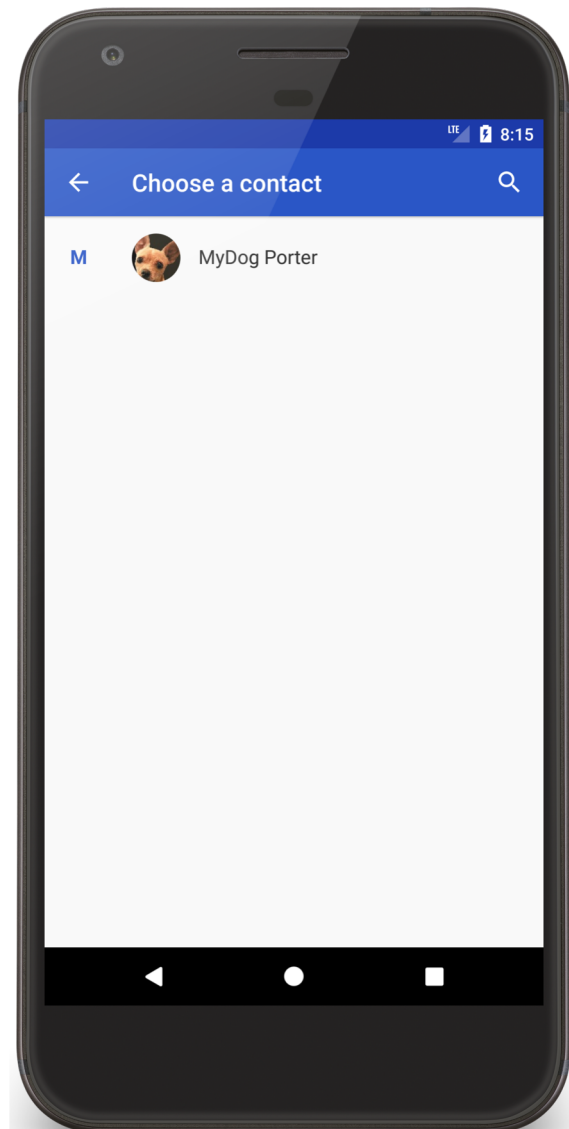
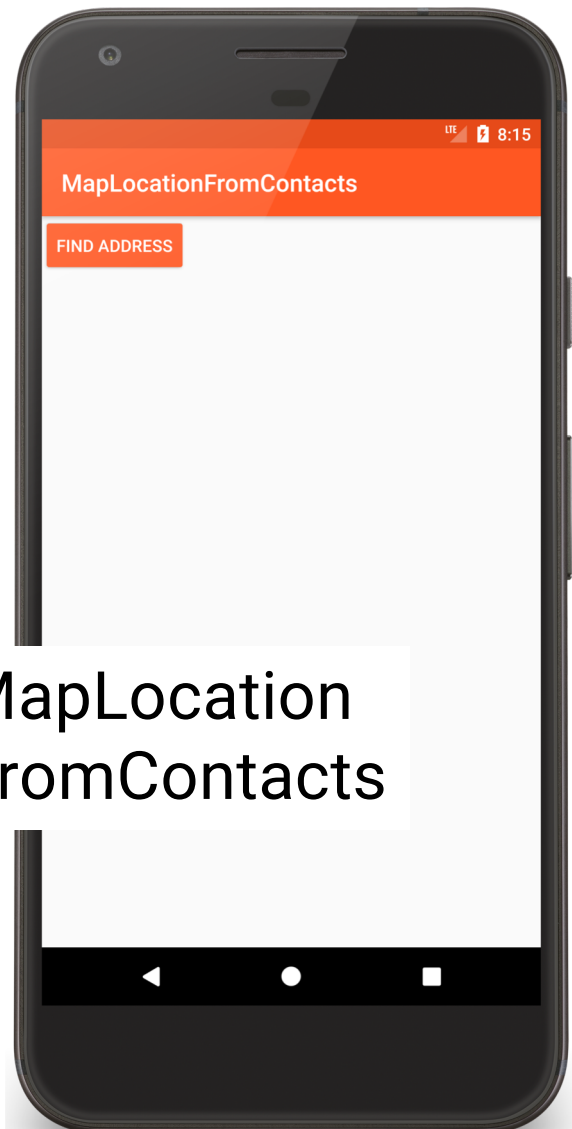
# MapLocationFromContacts

Selects a contact from contacts database

Displays a map centered on selected contact's address



# MapLocation FromContacts



```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="course.examples.maplocationfromcontacts"
  ... ">

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

  <application
    android:allowBackup="false"
    android:icon="@mipmap/ic_launcher"
    android:label="MapLocationFromContacts"
    android:theme="@style/MaterialTheme">
    <activity
      android:name="MapLocationFromContactsActivity"
      android:label="MapLocationFromContacts" >
      ...
    </activity>
  </application>
</manifest>
```

```
private void getContact() {  
    // Step 1: Ensure permissions  
    if (needsRuntimePermission(Manifest.permission.READ_CONTACTS)) {  
        requestPermissions(new String[]{Manifest.permission.READ_CONTACTS},  
                            PERMISSIONS_PICK_CONTACT_REQUEST);  
    } else {  
        // App has permissions. Get the contact from the Contacts app  
        startContactsApp();  
    }  
}
```

```
private boolean needsRuntimePermission(String permission) {  
    // Check the SDK version and whether the permission is already granted.  
    return (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M &&  
            checkSelfPermission(permission) != PackageManager.PERMISSION_GRANTED);  
}
```

*// Callback after user has been asked to grant a permission*

**@Override**

```
public void onRequestPermissionsResult(int requestCode,  
                                         String[] permissions, int[] grantResults) {  
    if (requestCode == PERMISSIONS_PICK_CONTACT_REQUEST) {  
        if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {  
            // Permission is granted  
            startContactsApp();  
        } else {  
            Toast.makeText(this, "This app requires access to your contact list",  
                          Toast.LENGTH_SHORT).show();  
        }  
    }  
}
```

# Defining Permissions

Apps can also define and enforce their own permissions

# Defining Permissions

Suppose your application performs a potentially dangerous operation

You might not want to allow just any application to invoke yours

So you can define & enforce your own permission

# PermissionExampleBoom

Simple Application that performs a (pretend) dangerous action





# Define & Enforcing Permissions

You don't want just application to run  
PermissionExampleBoom

Define & enforce an application-specific  
permission

*<!-- Defines a custom permission -->*

**<permission**

```
    android:name="course.examples.permissionexample.BOOM_PERM"  
    android:description="@string/boom_perm_string"  
    android:label="@string/boom_permission_label_string"  
    android:protectionLevel="dangerous"/>
```

*<!-- Enforces the BOOM\_PERM permission on users of this application -->*

**<application**

```
    android:allowBackup="true"  
    android:icon="@mipmap/ic_launcher"  
    android:theme="@style/MaterialTheme"  
    android:label="@string/app_name"  
    android:permission="course.examples.permissionexample.BOOM_PERM" >
```

# ProtectionLevel

Normal – Low risk

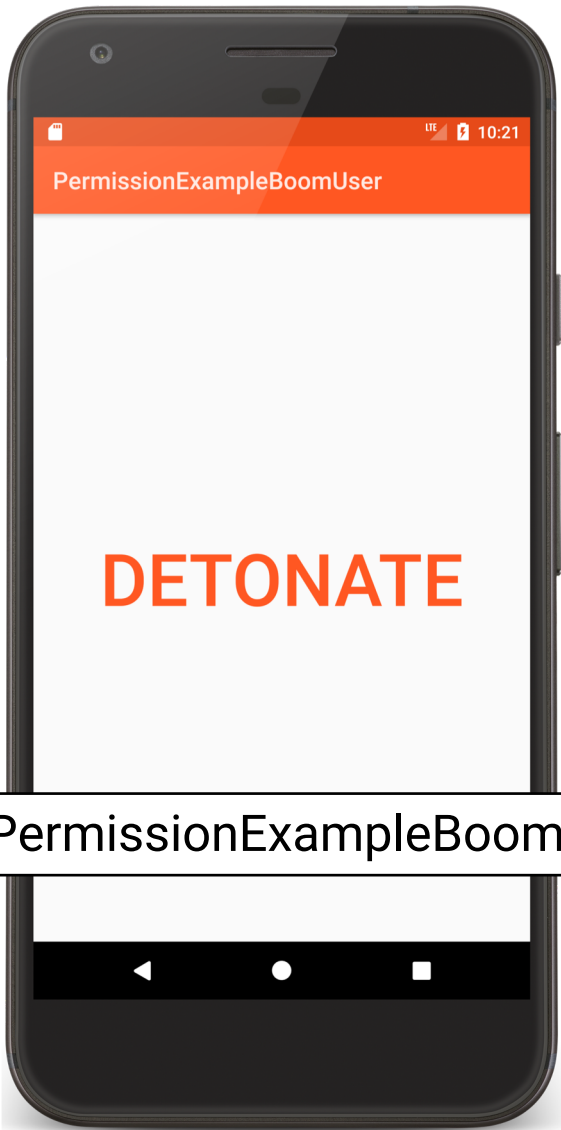
System automatically grants permission

Dangerous– High risk

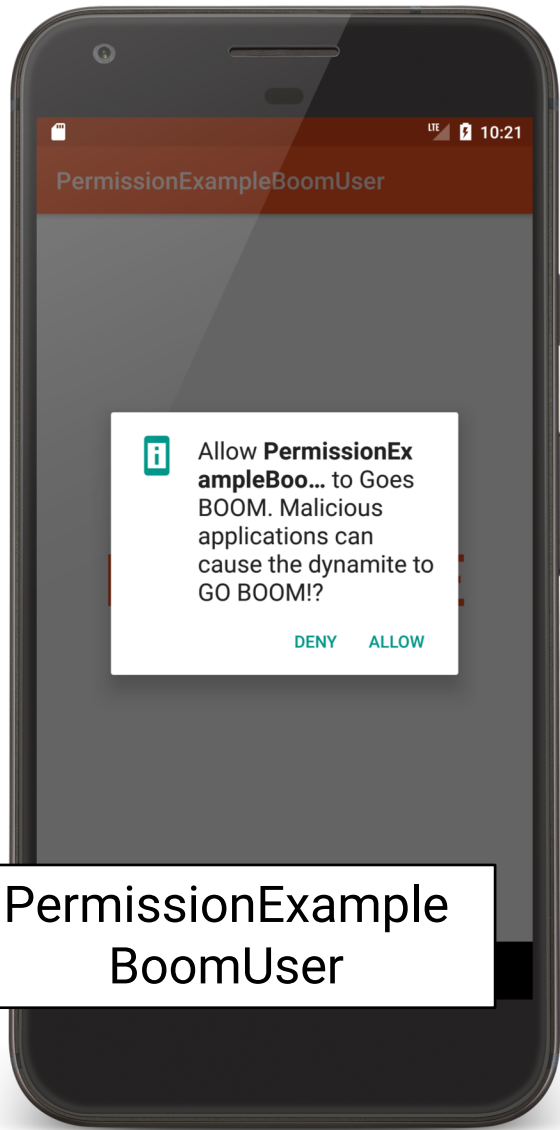
User must explicitly grant permission

# Using the Permission

Apps that want to use `PermissionExampleBoom` must acquire the required permission



PermissionExampleBoom



PermissionExample BoomUser



# Uses-Permission

Applications declare the permissions required by the Applications it uses

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
  package="course.examples.permissionexample.boomuser"  
  android:versionCode="1"  
  android:versionName="1.0" >
```

```
<!-- App needs the "...BOOM_PERM permission -->
```

```
<uses-permission android:name="course.examples.permissionexample.BOOM_PERM" />
```

# Component Permissions

Individual components can set their own permissions, restricting which other components can access them

Component permissions take precedence over application-level permissions



# Activity Permissions

Restricts which components can start the associated Activity

Checked within execution of

`startActivity()`

`startActivityForResult()`

Throws `SecurityException` on permissions failure

# Service Permissions

Restricts which components can start or bind to the associated service

Checked within execution of

`Context.startService()`

`Context.stopService()`

`Context.bindService()`

Throws `SecurityException` on permissions failure

# BroadcastReceiver Permissions

Restricts which components can send & receive broadcasts

Permissions checked in multiple places

More on this when we discuss  
BroadcastReceivers

# ContentProvider Permissions

Restrict which components can read & write the data in a ContentProvider

More on this when we discuss ContentProviders

**Next**

The Fragment Class

# Example Applications

MapLocationFromContacts

PermissionBoom

PermissionBoomUser