CS5530
Mobile/Wireless Systems
Android Bluetooth Interface

Yanyan Zhuang

Department of Computer Science

http://www.cs.uccs.edu/~yzhuang

Bluetooth Communication

- Bluetooth-enabled devices must first complete a **pairing** process
  - A discoverable device makes itself available for incoming connection
  - Another device finds the discoverable device using a discovery process
- After the discoverable device accepts the pairing request, the two devices complete a **bonding** process
  - After pairing and bonding are complete, two devices exchange info
- Two devices remain bonded
  - So they can reconnect automatically during a future session as long as they're in range of each other and neither device has removed the bond

Android Bluetooth Interface

- Using Bluetooth APIs, an Android app can perform the following
  - **Scan for other Bluetooth devices**
  - **Query local Bluetooth adapter for paired Bluetooth devices**
  - Establish communication channels
  - Transfer data to and from other devices
  - Manage multiple connections

Bluetooth Permissions

- "android.permission.BLUETOOTH"
  - To use Bluetooth features, app must declare the permission BLUETOOTH
  - For any Bluetooth communication, such as requesting a connection, accepting a connection, and transferring data
- "android.permission.BLUETOOTH_ADMIN"
  - If app wants to initiate device discovery or manipulate Bluetooth settings, must declare BLUETOOTH_ADMIN in addition to the BLUETOOTH permission

Types of Bluetooth Devices

- A Bluetooth device can either be
  - Paired
    - Paired devices are aware of each other's existence and share a link key, which can be used to authenticate
  - Connected
    - Connected devices share a channel, allowing them to send and receive data
  - Unknown
    - Remote devices that visible but the current device isn't paired with yet

Setting up Bluetooth

- Creating a BluetoothAdapter (similar to WiFiManager)
  - BluetoothAdapter bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    if (bluetoothAdapter == null) {
    // Device does not support Bluetooth
    }

Querying Paired Devices

- To see if the desired device is already known
  - ```
    Set<BluetoothDevice> pairedDevices =
    bluetoothAdapter.getBondedDevices();
    if (pairedDevices.size() > 0) {
        for (BluetoothDevice device : pairedDevices) {
            Log.d(TAG, "Device name: " + device.getName());
            Log.d(TAG, "Device address: " + device.getAddress());
        }
    }
    else {
        Log.d(TAG, "No paired device found :(");
    }
    ```
    ```

Discovering Devices (Scan)

- Asynchronous
  - The discovery process usually involves an inquiry scan of about 12 seconds
  - App must register a BroadcastReceiver for the ACTION_FOUND intent
    - The system broadcasts this intent for **each** discovered device (WiFi scan returns all)
    - The intent contains the extra fields EXTRA_DEVICE and EXTRA_CLASS, which in turn contain a BluetoothDevice and a BluetoothClass, respectively

Discovering Devices (Scan)

- Create an IntentFilter
  - bluetoothFilter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
- Register the receiver
  - by calling registerReceiver(BroadcastReceiver, IntentFilter)
    - E.g., Intent bluetoothIntent = getApplicationContext().registerReceiver(bluetoothReceiver, bluetoothFilter);

Discovering Devices (Scan)

- Create an instance of BroadcastReceiver
  - public static BroadcastReceiver bluetoothReceiver = new BroadcastReceiver() {

```
    @Override
    public void onReceive(Context context, Intent intent) {
   // define what to do when receive the broadcast
  }
};
```

- Stop receiving broadcasts
  - unregisterReceiver()

Discovering Devices (Scan)

- App request start/stop scan
  - bluetoothAdapter.startDiscovery(); // needs permission
    // BLUETOOTH_ADMIN
  - bluetoothAdapter.cancelDiscovery();

Discovering Devices (Scan)

- What to do when receive the broadcast?
  - public void onReceive(Context context, Intent intent) {
    String action = intent.getAction();
    if (BluetoothDevice.ACTION_FOUND.equals(action)) {
    BluetoothDevice device =
    intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
    Log.d(TAG, "Device name: " + device.getName());
    Log.d(TAG, "Device address: " + device.getAddress());
    short rssi = intent.getShortExtra(BluetoothDevice.EXTRA_RSSI,
    Short.MIN_VALUE);
    Log.d(TAG, "RSSI: " + rssi);

    BluetoothClass deviceClass =
    intent.getParcelableExtra(BluetoothDevice.EXTRA_CLASS);
    Log.d(TAG, "Class: " + deviceClass.getMajorDeviceClass());
    }
    }

Discovering Devices (Scan)

- What to do when receive the broadcast?
  - public void onReceive(Context context, Intent intent) {

```
String action = intent.getAction();
if (BluetoothDevice.ACTION_FOUND.equals(action)) {
// we found some device
}
}
```

Discovering Devices (Scan)

- Getting the device (name and MAC address)
  - BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
    Log.d(TAG, "Device name: " + device.getName());
    Log.d(TAG, "Device address: " + device.getAddress());
- Getting signal strength
  - short rssi = intent.getShortExtra(BluetoothDevice.EXTRA_RSSI, Short.MIN_VALUE);
    Log.d(TAG, "RSSI: " + rssi);
- Getting device class (profile)
  - BluetoothClass deviceClass = intent.getParcelableExtra(BluetoothDevice.EXTRA_CLASS);
    Log.d(TAG, "Class: " + deviceClass.getMajorDeviceClass());

Bluetooth Permission (Android 6.0 or later)

- **AndroidManifest.xml** permission *and* **runtime** permission in code
- AndroidManifest.xml:
  - <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    or <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

Bluetooth Permission (Android 6.0 or later)

- Runtime check and request permission
  - String[] PERMS_INITIAL={
    Manifest.permission.ACCESS_FINE_LOCATION,
    };

    if (ContextCompat.checkSelfPermission(this,
    Manifest.permission.ACCESS_FINE_LOCATION)
    != PackageManager.PERMISSION_GRANTED) {
    ActivityCompat.requestPermissions(this, PERMS_INITIAL,
    MY_PERMISSIONS_REQUEST);
    }

Devices Class/Profile

- AUDIO_VIDEO
  - Constant value: 1024 (0x00000400)
- COMPUTER
  - Constant value: 256 (0x00000100)
- HEALTH
  - Constant value: 2304 (0x00000900)
- IMAGING
  - Constant value: 1536 (0x00000600)
- UNCATEGORIZED
  - Constant value: 7936 (0x00001f00)

My Scan Result

- See text file!