Programming in Android

Session: 11

Wireless and Networking

Objectives

- Explain Bluetooth, Network, Wi-Fi, and NFC
- Explain the process of using Bluetooth
- Explain network management and Internet connectivity
- Explain working with Wi-Fi
- Explain working with NFC

Introduction

- High speed data transfer is essential for Smart Phones
- Wireless was the chosen medium as mobile devices cannot be tied down with a wired connection
- The Wireless Data Transfer standards currently supported by Android are:
  - Bluetooth
  - Network Connections
  - Wi-Fi
  - NFC

Bluetooth

- Bluetooth operates in the 2.4 GHz wireless spectrum
- It provides a means for short distance data transfer at reasonable speeds
- Android provides Bluetooth APIs, with the following functionalities:
  - Scan for Bluetooth devices which are present nearby
  - Query for paired Bluetooth devices using the local Bluetooth adapter
  - Establishing the RFCOMM channels
  - Connect through service discovery to other devices
  - Exchange data to and from other devices
  - Manage multiple connections

Bluetooth API Classes 1-5

- **BluetoothAdapter**:

  Is a representation of the local Bluetooth adapter and helps to perform the following functionalities:

  - Discover all Bluetooth devices
  - Query a list of devices
  - Create a Bluetooth device instance using the MAC address
  - Create a BluetoothServerSocket which will enable it to listen for communications from other Bluetooth connected devices

- **BluetoothDevice**:

  Is a representation of a remote Bluetooth device and performs the following functionalities:

  - Request a connection with remote device using BluetoothSocket
  - Retrieve information about the device such as name, address, and state of the device

Bluetooth API Classes 2-5

- **BluetoothSocket**:

  Is a representation of the interface like TCP socket and performs the following functionalities:

  - Allows data exchange between the Bluetooth connected devices via InputStream and OutputStream can occur through BluetoothSocket

- **BluetoothServerSocket**:

  Is a representation of an open server socket and performs the following functionalities:

  - Listens for the incoming requests
  - Responds to the received requests by returning a Bluetooth socket when the connection is accepted

Bluetooth API Classes 3-5

- **Bluetooth Class**:

  Is a read only set of properties that describe the characteristics and capabilities of a Bluetooth device and all Bluetooth profiles and services supported by the particular device

- **Bluetooth Profile**:

  Is a wireless interface specification for communication between Bluetooth based devices

- **Bluetooth Handset**:

  It provides support for Bluetooth headsets that are connected with the mobile device

Bluetooth API Classes 4-5

- **BluetoothA2dp**:

  It stands for Advanced Audio Distribution Profile and performs the following functionality:

  - Defines the way of streaming a high quality audio from one Bluetooth connected device to another

- **Bluetooth Health**:

  It represents a Health Device profile proxy and is used for controlling the Bluetooth service

- **BluetoothHealthCallback**:

  Is an abstract class that the callback methods must be implemented for the following functionality:

  - Receive updates about the changes in the Bluetooth channel state and application registration

Bluetooth API Classes 5-5

- **BluetoothHealthAppConfiguration**:

  It represents a configuration of the Bluetooth Health third party application that is used for communicating with a remote Bluetooth health device

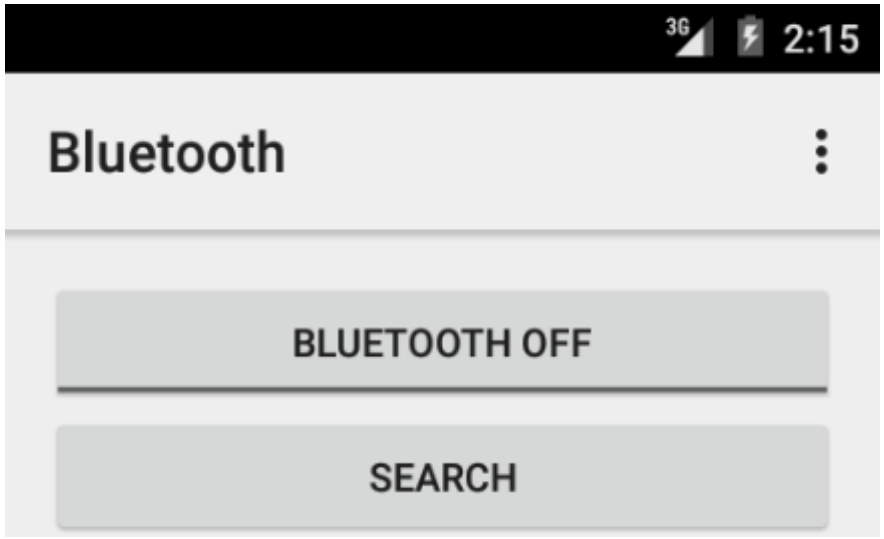
- **BluetoothProfile.Servicelistener**:

  Is an interface that performs the following functionality:

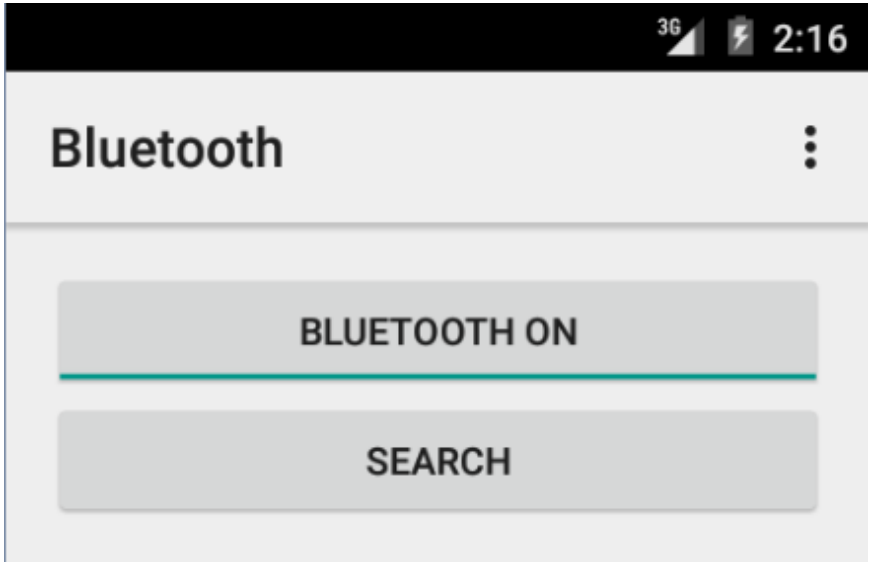
  - Notifies the IPC clients whenever they are connected or disconnected from the service

Bluetooth Example Application 1-2

- Using the classes, an application for demonstrating Bluetooth functionality is created as shown in the following figure:

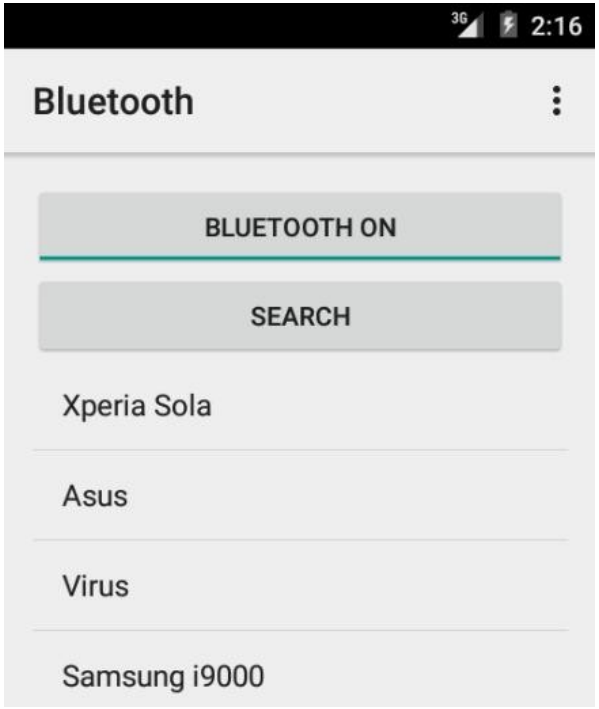


- When the BLUETOOTH OFF button is clicked the output will be as shown in the following figure:



Bluetooth Example Application 2-2

- When SEARCH button is clicked the output will be as shown in the following figure:



- The application logic is as follows:

- Developer creates an Activity with a Toggle Button and a normal Button
- The Toggle Button is used to Turn BLUETOOTH ON and OFF
- When the SEARCHButton is clicked, the search() method is called
- The method retrieved the Bonded devices using the BluetoothAdapter's getBondedDevices() method
- The retrieved list is displayed to the user

Network

- In Android, the type of the connection is irrelevant and hidden from the application
- As long as a network connection is established, it will be available to access using the classes HttpClient and HttpUrlConnection
- The Network Connection could be:
  - Mobile data connection
  - Wi-Fi network connection
  - Wired connection
  - Bluetooth tether

HttpClient

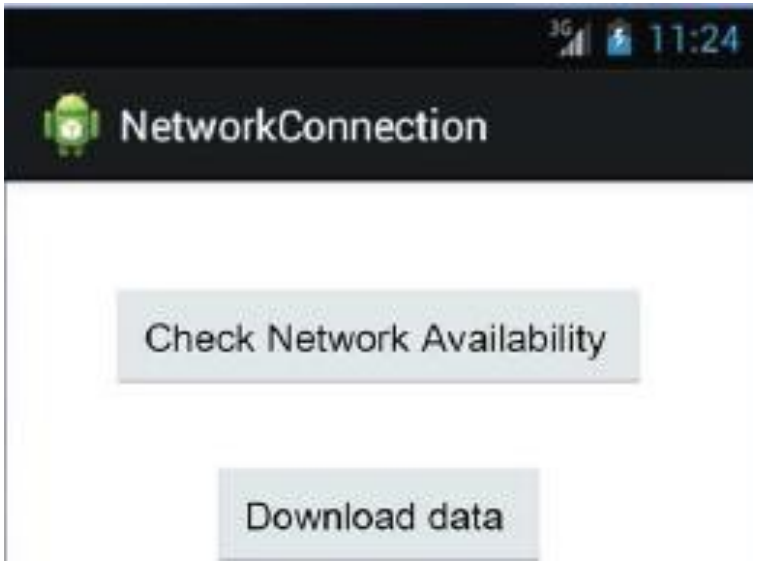
- HttpClient encapsulates a smorgasbord objects which are required to execute HTTP requests to send and receive the data over the network
- HttpClient simplifies the handling of Http Requests
- The thread safety of HttpClient depends on the way the configuration of a specific client is being made

HttpURLConnection

- HttpUrlConnection is used to make a single request to the Http Server
- Connection for the particular URL can be established by invoking the openConnection() method and the data from the URL can be read by invoking the method getInputStream()
- It is used for sending and receiving data over the Web
- The class uses the GET and POST method to send and receive data
- By default it uses the GET method
- The other HTTP methods are used with setRequest() method

Network Example Application 1-2

- Using the classes, an application for demonstrating Network functionality is created as shown in the following figure:

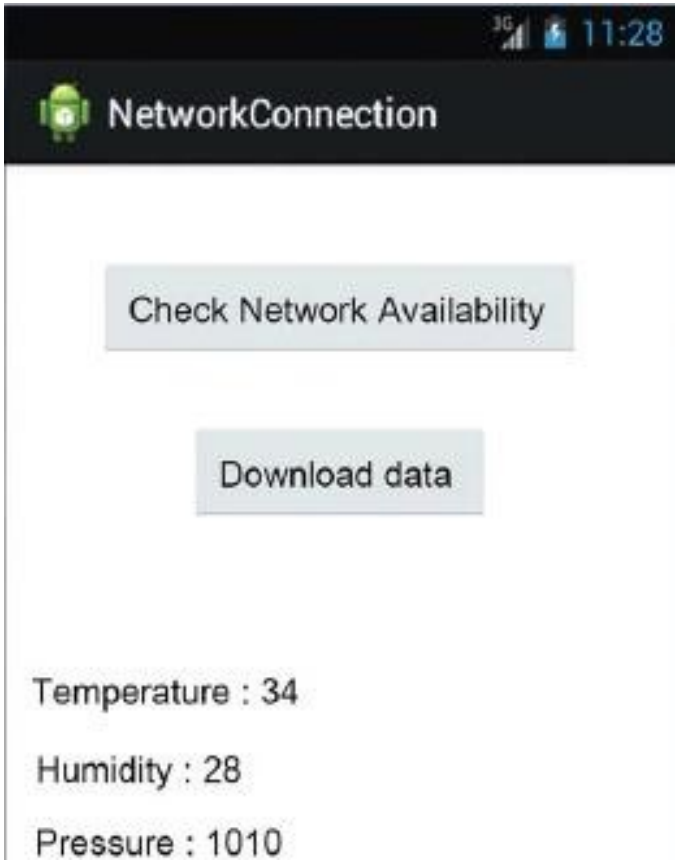


- When Check Network Availability button is clicked the output will be as shown in the following figure:

Network Example Application 2-2

- When Download data button is clicked the output will be as shown in the following figure:

- Downloaded data displays the final output as shown in the following figure:



Network Example Application Logic

- The application logic is as follows:

- Developer creates an Activity with two Buttons
- When the Check Network Availability button is clicked the Network availability is checked using the ConnectivityManager API
- A Toast is displayed to the user with the Network state
- When the Download data button is clicked, the Remote server is contacted using an HttpClient Object
- The data transfer takes place using JSON
- The returned data is displayed to the user

Wi-Fi

- Wi-Fi is the most commonly used standard for wireless communication for high bandwidth transfer
- Wi-Fi runs in the same 2.4 Ghz spectrum as Bluetooth
- It has a reasonably long range with high transfer rates reaching upto 300 Mbps and higher
- It is also possible to share a mobile data connection with other devices using Android's 'tether' feature

Wi-Fi API Classes

- **WifiManager**:
  - This class is used to manage the Wi-Fi connectivity by invoking the method, Context.getSystemService(Context.WIFI_SERVICE)
  - The Wifi Manager can deal with the following:
    - List of configured networks
    - Currently available active networks
    - Provide enough information to the developer regarding which network one has to choose and connect to

- **ConnectivityManager**:
  - ConnectivityManager notifies the application whenever any network changes happen
  - Some of the tasks performed by the ConnectivityManager are as follows:
    - Scans Wi-Fi, GPRS and UMTS, and other network connections
    - Sends broadcast intents whenever the connectivity changes
    - Queries the state of available network using an API

Wi-Fi Example Application

- Using the classes, an application for demonstrating Wi-Fi functionality is created as shown in the following figure:

- When the Turn Wifi On button is clicked, the output will be as shown in the following figure:

Wi-Fi Example Application Logic

- The application logic is as follows:

- Developer creates an Activity with two Buttons
- When the Turn Wifi On button is clicked, Wi-Fi is enabled on the device
- The converse happens when the Turn Wifi Off button is clicked
- The status is displayed to the user

NFC

- NFC is the abbreviation for Near Field Communication
- It is a new wireless communication standard for short range communication
- It operates at 13.56 Mhz frequency and can achieve speeds up to 424 Kbit/s
- NFC devices are activated on proximity using a touch or tap gesture among the devices
- NFC primarily consists of three standards/utilities:
  - Card emulation mode
  - Read/Write mode
  - Peer-to-Peer mode

NFC API

- **NDEF Content Type**
  - NDEF is the data format used by Android for NFC messages and records. The data formats are implemented in the classes NdefMessage and NdefRecord
  - An NDEF object can be retrieved using the get(Tag) method. The tag type needs to be mentioned before the NDEF object is retrieved. There are currently four supported tag types:
    - NFC_FORUM_TYPE_1
    - NFC_FORUM_TYPE_2
    - NFC_FORUM_TYPE_3
    - NFC_FORUM_TYPE_4
- **Relevant Classes**
  - This section describes the relevant classes needed to use the NFC API:
    - NdefMessage
    - NdefRecord
    - NfcAdapter
    - NfcEvent
    - NfcManager

NFC Permissions

- The application needs to explicitly request permission to access NFC hardware
- The manifest code is shown in the following Code Snippet:

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

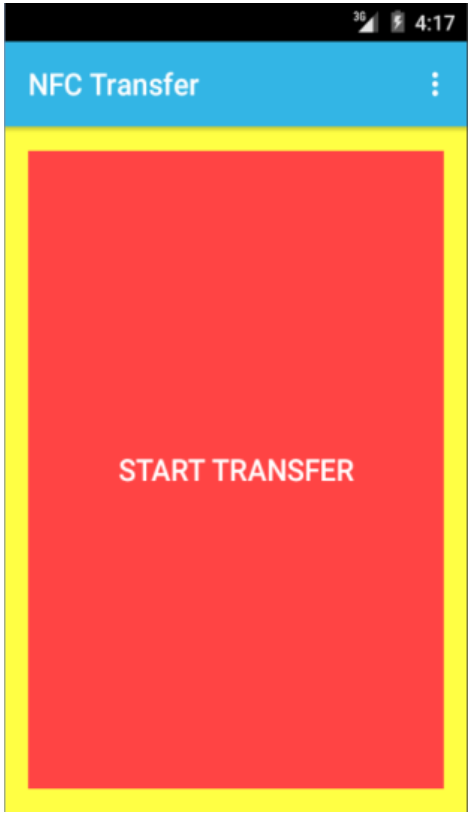
Verifying NFC Support

- It is a good practice to ensure that the device has the hardware and API support to perform NFC operation
- This can be done using one of the following methods:
  - Manifest Code
  - System Feature
  - API Support
- The code to check the System Feature and API Support is shown in the following Code Snippet:

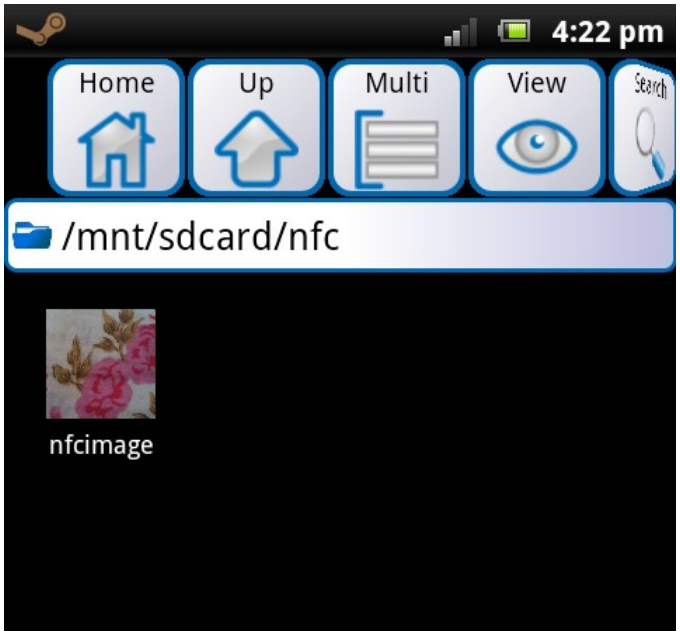
```
//System Feature
boolean systemSupport =
getPackageManager.getSystemFeature(PackageManager.FEATURE_NFC;


//API Support
int level = Build.VERSION.SDK_INT;
```

NFC Example Application

- Using the classes, an application for demonstrating NFC functionality is created as shown in the following figure:



- Once the START TRANSFER button is clicked, the transferred file can be viewed as shown in the following figure:



NFC Example Application Logic

- The application logic is as follows:

- Developer creates an Activity with a Button
- When the START TRANSFER button is clicked, the file nfcimage.jpg is transferred from one device to another
- This is done using the NFCAdapter's setBeamPushUris() method
- The Transferred file can be viewed on the other device using any File Manager Application

Summary

- Bluetooth is a network stack that transfers data between devices wirelessly. Android provides all the Bluetooth APIs' under android.bluetooth package
- Bluetooth Adapter plays the role of discovering all the Bluetooth enabled devices and querying for request
- Android applications performing network operations uses HTTP to send and receive data
- HttpClient can be used to send and receive data from the server by creating a DefaultHttpClient() which would help in sending and receiving of data
- Android provides Wi-Fi APIs, through which applications can communicate with the wireless stack that provides Wi-Fi network access
- NFC is a short range wireless data transfer protocol
- The API for NFC is encapsulated in the android.nfc package