Android中的WiFi P2P

Android中的WiFi P2P能够允许一定范围内的设备通过Wifi直接互连而不必通过热点或互联网。

使用WiFi P2P需要**Android API Level >= 14**才可以，而且不要忘记在Manifest文件中加入下面5个权限:

- android.permission.ACCESS_WIFI_STATE
- android.permission.CHANGE_WIFI_STATE
- android.permission.ACCESS_NETWORK_STATE
- android.permission.CHANGE_NETWORK_STATE
- android.permission.INTERNET (WiFi P2P并不需要连接互联网，但是因为要用到Java Socket，所以要加这个权限)

关于WiFi P2P的操作几乎都靠WifiP2pManager来进行，所以如果你的程序要用到WiFi P2P功能，可以设置一个全局变量wifiP2pManager，然后在onCreate()生命函数中获取系统WifiP2pManager对象:

```
wifiP2pManager = (WifiP2pManager) getApplicationContext().getSystemService(Context.WIFI_P2P_SERVICE);
```

WifiP2pManager有如下方法可以很方便的进行P2P操作:

方法	功能
initialize()	在使用WiFi P2P功能时必须先调用这个方法，用来通过WiFi P2P框架注册我们的应用
connect()	根据配置(WifiP2pConfig对象)与指定设备(WifiP2pDevice对象)进行P2P连接
cancelConnect()	关闭某个P2P连接
requestConnectInfo()	获取设备的连接信息
createGroup()	以当前的设备为组长创建P2P小组
removeGroup()	移除当前的P2P小组
requestGroupInfo()	获取P2P小组的信息
discoverPeers()	初始化peers发现操作
requestPeers()	获取当前的peers列表(通过discoverPeers发现来的)

每当WifiP2pManager执行某个P2P操作时，可以通过不同的监听器来检测这些操作的反馈结果，这些监听器的类型如下:

监听器	关联的行为
WifiP2pManager.ActionListener	connect(), cancelConnect(), createGroup(), removeGroup()和discoverPeers()
WifiP2pManager.ChannelListener	initialize()
WifiP2pManager.ConnectionInfoListener	requestConnectInfo()
WifiP2pManager.GroupInfoListener	requestGroupInfo()
WifiP2pManager.PeerListListener	requestPeers()

那么接下来就可以讲解如何使用WiFi P2P了。

一. 准备工作

准备阶段需要让我们的应用可以接受P2P信号，这就需要一个意图广播接收器，要创建一个符合我们要求的广播接收器需要准备下面3个要素:

(1) 一个意图广播接收器，用来接受意图广播。

(2) 通过WifiP2pManager的initialize()初始化操作来获取一个Channel对象，用于以后和WiFi P2P框架保持通信。

(3) 自己包装BroadcastReceiver类，实现一个接收器。

意图过滤器使用如下方法创建:

```
//  设置intent过滤器
intentFilter  =  new  IntentFilter();
//一个全局的intentFilter对象
intentFilter.addAction(WifiP2pManager.WIFI_P2P_STATE_CHANGED_ACTION);
//  WiFi  P2P是否可用
intentFilter.addAction(WifiP2pManager.WIFI_P2P_PEERS_CHANGED_ACTION);
  //  peers列表发生变化
  intentFilter.addAction(WifiP2pManager.WIFI_P2P_CONNECTION_CHANGED_ACTION);
  //  WiFi  P2P连接发生变化
  intentFilter.addAction(WifiP2pManager.WIFI_P2P_THIS_DEVICE_CHANGED_ACTION);
    //  WiFi  P2P设备信息发生变化(比如更改了设备名)
```

Channel的获取方法:

```
channel  =  wifiP2pManager.initialize(getApplicationContext(),  getMainLooper(),  null);
//  channel是一个全局的Channel对象
```

意图广播接收器的创建需要通过继承BroadcastReceiver类来自己实现，下面给出了一段示例:

```
class  MyBroadcastReceiver  extends  BroadcastReceiver
  {
  //  使用WiFi  P2P时，自己定义的BroadcastReceiver的构造函数一定要包含下面这三个要素
        private  WifiP2pManager  wifiP2pManager;
        private  Channel  channel;
        private  Activity  activity;
        public  MyBroadcastReceiver(WifiP2pManager  wifiP2pManager,  Channel  channel,  Activity  activity)
        {
        this.wifiP2pManager  =  wifiP2pManager;
        this.channel  =  channel;
        this.activity  =  activity;
                }
                //  onReceiver对相应的intent进行处理
                @Override
                public  void  onReceive(Context  context,  Intent  intent)
                {
                String  action  =  intent.getAction();
                        if  (WifiP2pManager.WIFI_P2P_STATE_CHANGED_ACTION.equals(action))
                        {
                        int  state  =  intent.getIntExtra(WifiP2pManager.EXTRA_WIFI_STATE,  -1);
                                if  (state  ==  WifiP2pManager.WIFI_P2P_STATE_ENABLED)
                                  {
                                  //  WiFi  P2P  可以使用
                                        }  else
                                          {
                                          //  WiFi  P2P  不可以使用
                                                }
                                                        }
                                                else  if  (WifiP2pManager.WIFI_P2P_PEERS_CHANGED_ACTION.equals(action))
                                                  {
                                                        }
                                                        else  if(WifiP2pManager.WIFI_P2P_CONNECTION_CHANGED_ACTION.equals(action))  {
                                                          }
                                                          else  if  (WifiP2pManager.WIFI_P2P_THIS_DEVICE_CHANGED_ACTION.equals(action))
                                                                  }
                                                                          }
                                                                          }
```

定义完了之后就可以再需要的地方实例化一个意图广播接收器对象了:

```
broadcastReceiver  =  new  MyBroadcastReceiver(wifiP2pManager,  channel,  this);
```

那么三个要素都具备了之后就要向应用注册我们的接收器，好让它可以开始工作:

```
registerReceiver(broadcastReceiver,  intentFilter);
```

二.启动peers发现

```
public  void  discoverPeers(View  view)
{
        peerListListener  =  new  WifiP2pManager.PeerListListener()
        {
                @Override
                public  void  onPeersAvailable(WifiP2pDeviceList  peerList)
                {
                        peers.clear();
                        //  peers是一个全局ArrayList对象，用于保存发现的peers的信息
                        peers.addAll(peerList.getDeviceList());
                        //  如果你有一个控件用来显示这些peers的信息，就可以再这里更新了
                  }
            };
            wifiP2pManager.discoverPeers(channel,  new  WifiP2pManager.ActionListener()
                      {
                              @Override
                              public  void  onSuccess()
                              {
                                      Toast.makeText(getApplicationContext(),  "discover  peers!",  Toast.LENGTH_SHORT);
                              }
                              @Override
                              public  void  onFailure(int  arg0)
                                {
                                }
                      });
              //  discoverPeers是异步执行的，调用了之后会立刻返回，但是发现的过程一直在进行，
                //  直到发现了某个设备时就会通知你}
```

当发现了设备之后就会触发WifiP2pManager.WIFI_P2P_PEERS_CHANGED_ACTION意图广播，你可以再之前自己包装的BroadcastReceiver类中加入对这一意图的处理:

```
else  if  (WifiP2pManager.WIFI_P2P_PEERS_CHANGED_ACTION.equals(action))  {      if  (wifiP2pManager  !=  null)  {                wifiP2pManager
```

三.与peers建立连接

```
public  void  connect()
{
        final  WifiP2pDevice  device  =  (WifiP2pDevice)  peers.get(0);
        //从peers列表中获取发现来的第一个设备
        WifiP2pConfig  config  =  new  WifiP2pConfig();
        config.deviceAddress  =  device.deviceAddress;
        config.wps.setup  =  WpsInfo.PBC;
        wifiP2pManager.connect(channel,  config,  new  WifiP2pManager.ActionListener()
                                        {
                                                @Override
                                                public  void  onSuccess()  {
                                                //  连接成功
                                                        Toast.makeText(getApplicationContext(),  "与设备"  +  device.deviceName  +  "连接成功"
                                                }
                                                        @Override
                                                        public  void  onFailure(int  arg0)
                                                        {
                                                                //  连接失败
                                                                Toast.makeText(getApplicationContext(),  "与设备"  +  device.deviceName  +  "j
                                                          }
                                        });
      }
```

每当有P2P连接状态发生改变时，就会触发WifiP2pManager.WIFI_P2P_CONNECTION_CHANGED_ACTION意图广播，可以在之前你自己包装的BroadcastReceiver类中加入对这一意图的处理:

```
else  if  (WifiP2pManager.WIFI_P2P_CONNECTION_CHANGED_ACTION.equals(act
```