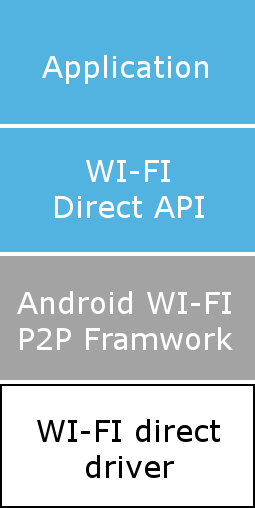
# Android WI-FI P2P

Since Android 4.0, devices with appropriate hardware are allowed to connect directly to each other over WI-FI P2P without an access point between them. Android P2P framework complies with the WI-FI Alliance’s WI-FI Direct certification program. With the usage of this API you are able to discover and connect to other devices when they support WI-FI P2P. According to documentations the advantage of WI-FI P2P beside Bluetooth or similar connection types is a fast connection across distances much longer than others. This allows applications a fast exchange of data between multiple users, which could be useful for applications such as multiplayer games, photosharing applications and in general, all applications which are relying on a fast connection between a long distance.

The main parts of the Androids WI-FI P2P APIs offer following methods and features.

* Methods for discovering, requesting and connecting to peers.
* Listeners which allows your application to get notified about the results of important WI-FI P2P methods, like discover, request, connect.
* Intents which notifies your application of specific events detected by the WI-FI P2P framework.

The picture below describes a typical Android WI-FI P2P architecture.



**Android Application and API**

**WI-FI direct package**

**Available since Android 4.0. Preinstalled with the WI-FI P2P packages and drivers.**

**Chip vender**

Figure 1: Android WI-FI P2P architecture

## Android API Description

In general the “WifiP2pManager” class provides all methods which allow your application to interact with the WI-FI hardware on your device. This class offers following actions to establish a connection:

|  |  |
| --- | --- |
| Method | Description |
| initialize() | This method registers the application with the WI-FI hardware. It is necessary to call this method before any other “WifiP2pManager” method. |
| connect() | This method starts a P2P connection with another device. |
| cancelConnect() | Cancels all running connection processes. |
| requestConnectInfo() | Requests a device connection Info |
| createGroup() | Creates a P2P group with the caller device as Group owner. |
| removeGroup() | Removes the current P2P group. |
| requestGroupInfo() | Requests the P2P a group info |
| discoverPeers() | Calling this method initiates a peer discovery. |
| requestPeers() | Requests the current list discovered peers. |

All the “WifiP2pManager” methods let you pass in a listener object. These listeners can be notified by the WI-FI P2P framework about the status of a call. The below table describes the available listeners and the corresponding “WifiP2pManager” methods which use the listeners.

|  |  |
| --- | --- |
| Listeners | Associated methods |
| WifiP2pManager.ActionListener | connect(), cancelConnect(), createGroup(), removeGroup(), discoverPeers() |
| WifiP2pManager.ChannelListener | initialize() |
| WifiP2pManager.ConnectionInfoListener | requestConnectInfo() |
| WifiP2pManager.GroupInfoListener | requestGroupInfo() |
| WifiP2pManager.PeerListListener | requestPeers() |

Furthermore, Androids WI-FI P2P APIs offers intents which notify your application when certain WI-FI P2P events happen, for example when new peers are discovered or the WI-FI state changes. If you want to use these intents it is necessary to create a broadcast receiver (<http://developer.android.com/guide/topics/connectivity/wifip2p.html#creating-br>) in your Android application that handles the specific intents.

|  |  |
| --- | --- |
| Intents | Description |
| WIFI\_P2P\_CONNECTION\_CHANGED\_ACTION | Get notified when the device’s WI-FI P2P state changes. |
| WIFI\_P2P\_PEERS\_CHANGED\_ACTION | Get notified when “discoverPeers()” is called. In this Intent you usually call “requestPeers()” and pass in a “PeerListListener” which get then a list of available peers. |
| WIFI\_P2P\_STATE\_CHANGED\_ACTION | Get notified when the WI-FI P2P is enabled or disabled on the device. |
| WIFI\_P2P\_THIS\_DEVICE\_CHANGED\_ACTION | Get notified when some specific device details have changed, like the device name. |