GDG DevFest

Wi-Fi Direct

Frederic Barthelery

GENYMOBILE

Let IT be mobile

Who am I ?

Developer on Android since 2009

Software engineer at **Genymobile**

Lead developer of **Beem**

Wi-Fi Direct

Certification from the Wi-Fi Alliance in 2010

Improved Ad Hoc mode with WPS to simplify setup

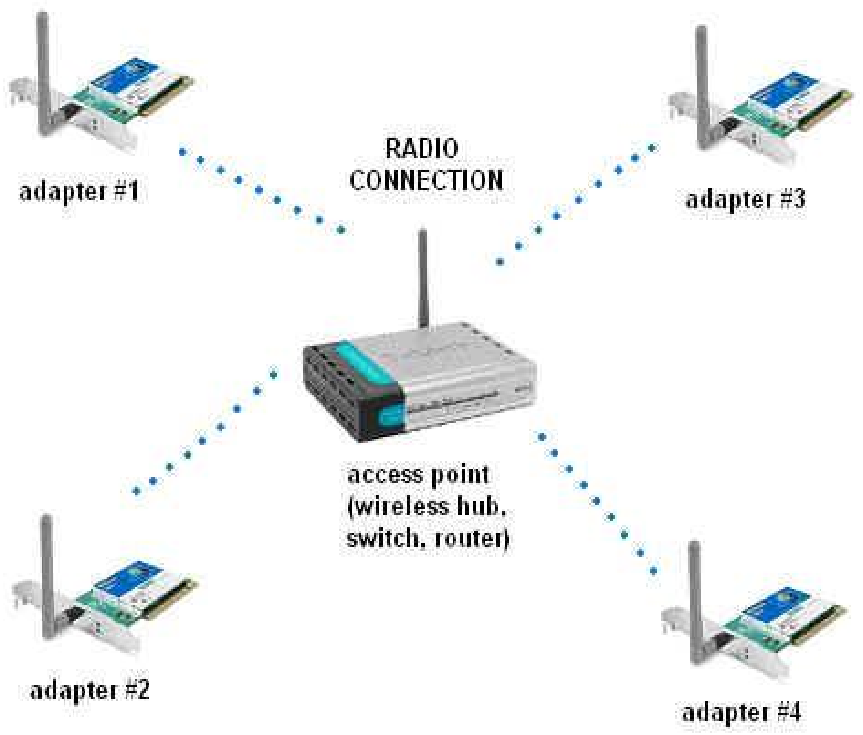
Compatible with legacy device

Available since Android Ice Cream Sandwich (4.0)

DevFest Nantes 2012

Wi-Fi Direct

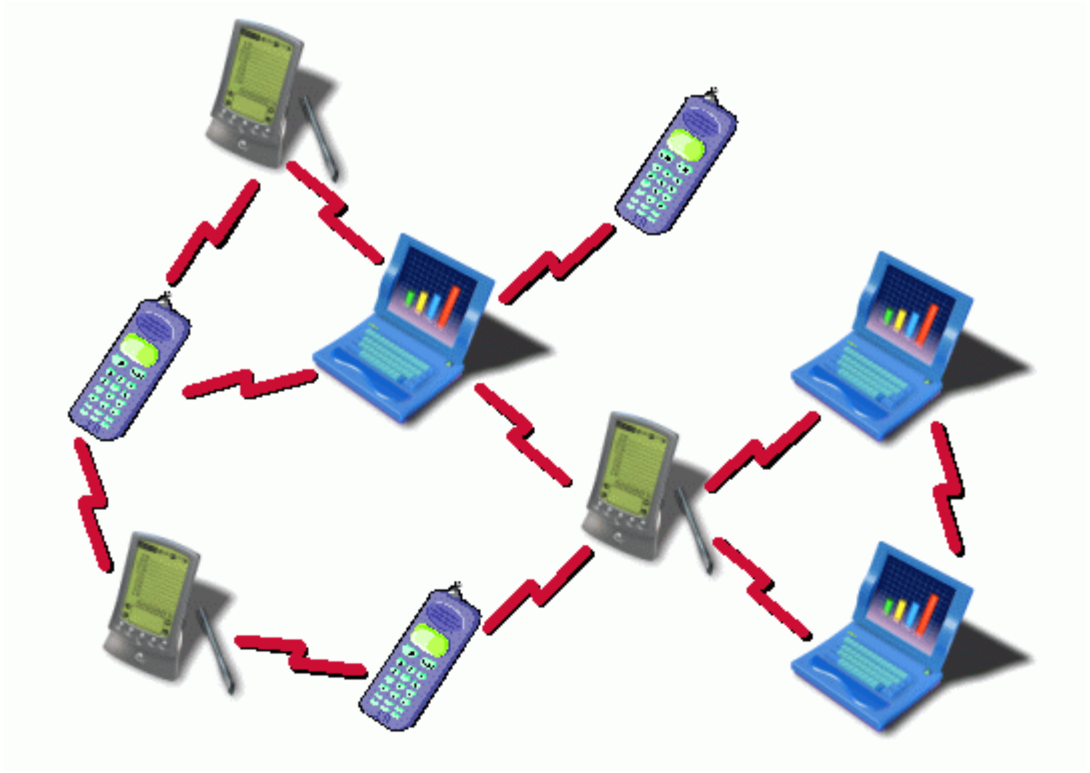
GENYMOBILE Let IT be mobile



Infrastructure mode

DevFest
Nantes 2012

Wi-Fi Direct



Ad hoc mode

GENYMOBILE
Let IT be mobile

Permissions

AndroidManifest.xml

```
<uses-feature
   android:name="android.hardware.wifi.direct" />

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

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

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

DevFest Nantes 2012

Wi-Fi Direct API

Package android.net.wifi.p2p

System service like the WIFI_SERVICE

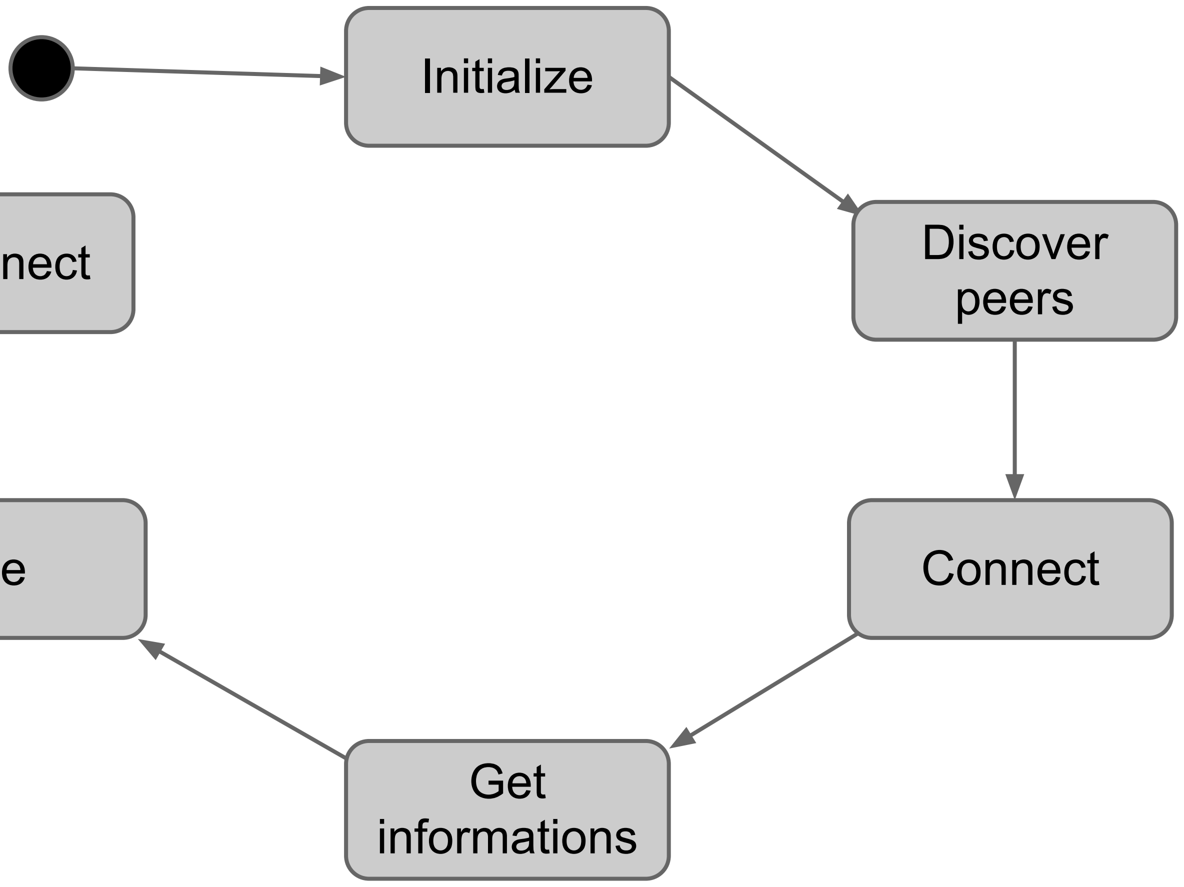
Asynchronous API

All methods need a Channel

Most methods take an optionnal ActionListener

GENYMOBILE *Let IT be mobile*

Workflow


Initialize
Discover peers
Connect
Get informations
Use
Disconnect

DevFest Nantes 2012

Wi-Fi Direct API

android.net.wifi.p2p.WifiP2pManager

```
// initialize
Channel initialize(Context, Looper, ChannelListener);

// discover
void discoverPeers(Channel, ActionListener);

// connection management
void connect(Channel, WifiP2pconfig, ActionListener);
void cancelConnect(Channel, ActionListener);
void createGroup(Channel, ActionListener);
void removeGroup(Channel, ActionListener);

// request informations
void requestConnectionInfo(Channel, ConnectionInfoListener);
void requestGroupInfo(Channel, GroupInfoListener);
void requestPeers(Channel, PeerListListener);
```

GENYMOBILE Let IT be mobile

DevFest Nantes 2012

Wi-Fi Direct API

Broadcast intents

All operations are asynchronous. Their results are broadcasted

- `WifiP2pManager.WIFI_P2P_STATE_CHANGED_ACTION`
- `WifiP2pManager.WIFI_P2P_CONNECTION_CHANGED_ACTION`
- `WifiP2pManager.WIFI_P2P_THIS_DEVICE_CHANGED_ACTION`
- `WifiP2pManager.WIFI_P2P_PEERS_CHANGED_ACTION`

GENYMOBILE *Let IT be mobile*

Wi-Fi Direct API

Initialize

```
WifiP2pManager wifiMgr;
Channel channel;

wifiMgr = (WifiP2pManager)
              getSystemService(Context.WIFI_P2P_SERVICE);

channel = wifiMgr.initialize(this,
                            getMainLooper(),
                            new ChannelListener() {
           public void onChannelDisconnected() {
           // deal with the error or try to reinitialize.
                channel = null;
           }
});
```

Wi-Fi Direct API

Discover

```
wifiMgr.discoverPeers(channel, actionListener);
```

Discovery result broadcast

```
//WifiDirectBroadcastReceiver
public void onReceive(Intent intent) {
    String action = intent.getAction();
    if (WifiP2pManager.WIFI_P2P_PEERS_CHANGED_ACTION
            .equals(action)) {
            onPeersChanged(intent);
    }
}
```

Wi-Fi Direct API

```
private void onPeersChanged(Intent intent) {
       wifiMgr.requestPeers(channel, myPeerListListener);
}

PeerListListener myPeerListListener =
   new PeerListListener() {
       @Override
       public void onPeersAvailable(WifiP2pDeviceList
peers) {
           Collection<WifiP2pDevice> devices =
                                      peers.getDeviceList();
           // get informations on devices
           // and choose one to connect to
       }
}
```

DevFest Nantes 2012

Wi-Fi Direct API

Connect

```
private void connect(WifiP2pDevice device) {
    WifiP2pConfig config = new WifiP2pConfig();
    config.deviceAddress = device.deviceAddress;
    config.wps.setup = WpsInfo.PBC; // choose between what is available on the device.
    wifiMgr.connect(channel, config, actionListener);
}
```

For legacy devices

```
wifiMgr.createGroup(channel,  actionListener);
```

GENYMOBILE Let IT be mobile

Wi-Fi Direct API

Connection result in broadcast

```
public void onReceive(Intent intent) {
    String action = intent.getAction();
    ...
    if (WifiP2pManager.WIFI_P2P_CONNECTION_CHANGED_ACTION
                .equals(action)) {
        onConnectionChanged(intent);
    }
}
```

Wi-Fi Direct API

```
private void onConnectionChanged(Intent intent) {
    WifiP2pInfo p2pInfo =
intent.getParcelableExtra(WifiP2pManager.
EXTRA_WIFI_P2P_INFO);
    NetworkInfo netInfo =
intent.getParcelableExtra(WifiP2pManager.EXTRA_NETWORK_INFO);
    if (netInfo.isConnected()) {
        updateInfos();
        useNetwork(p2pInfo);
    } else {
        resetInfos();
    }
}
```

Wi-Fi Direct API

```
private void updateInfos() {
    wifiMgr.requestGroupInfo(channel,
        new GroupInfoListener() {
        @Override
        public void onGroupInfoAvailable(WifiP2pGroup
group)
        {
            String name = group.getNetworkName();
            String passphrase = group.getPassphrase();
            Collection<WifiP2pDevice> devices =
                    group.getClientList();
                    // do stuff with devices
                    // but ... No way to get their IP
addresses :(
        }
    });
}
```

DevFest Nantes 2012

Wi-Fi Direct API

```
private void useNetwork(WifiP2pInfo p2pInfo) {
    if (!p2pInfo.isGroupOwner()) {
        InetAddress addr  = p2pInfo.groupOwnerAddress;
        Socket s  = new Socket(addr, 1234);
        // use the socket
    } else {
        // groupOwnerAddress is our local address
        ServerSocket serverSocket = new ServerSocket(1234);
        Socket s  = serverSocket.accept();
        // use the socket
    }
}
```

GENYMOBILE Let IT be mobile

Wi-Fi Direct API

Disconnect

```
wifiMgr.removeGroup(channel, actionListener);
```

Wi-Fi Direct API

Warnings

ActionListener.onSuccess() don't signal success of the action but only successful send of the command to the service

Maybe the API is too much asynchronous.
No events for a connection failure

Only knows the group owner IP address.

Network service discovery

Set of protocols to discover service offered on a network

DNS-SD, UPNP

DNS-SD available since Android JellyBean (4.1)

DevFest
Nantes 2012

DNS-SD

DNS Service Discovery

Multicast DNS with SRV, TXT and PTR records

<instanceName>.<instanceType>.local.

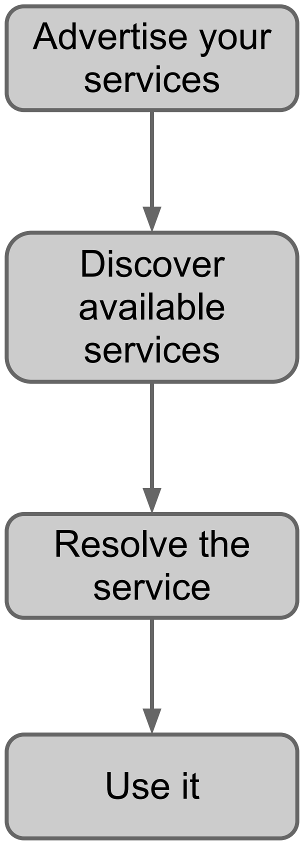
instanceType = _<proto>._tcp or _<proto>._udp

LX42PRINTER._ipp._tcp.local -> 192.168.42.42:631

GENYMOBILE
Let IT be mobile

DNS-SD

Workflow



DNS-SD

android.net.nsd.NsdManager

```
// advertise
void registerService(NsdServiceInfo serviceInfo,
       int protocolType, RegistrationListener listener);
void unregisterService(RegistrationListener listener);

// discover
void discoverServices(String serviceType,
       int protocolType, DiscoveryListener listener);
void stopServiceDiscovery(DiscoveryListener listener);

// resolve
void resolveService(NsdServiceInfo serviceInfo,
                         ResolveListener listener);
```

Let IT be mobile

DNS-SD

Advertise your service

Describe your service

- service name
- service type
- port

```
nsdMgr.registerService(serviceInfo,
                                NsdManager.PROTOCOL_DNS_SD,
                                myRegistrationListener);
```

Service name can change to solve conflicts, so save the resulting service name

DNS-SD

Discover services

```
List<NsdServiceInfo> services =
        new ArrayList<NsdServiceInfo>();

nsdMgr.discoverService(serviceType,
                       NsdManager.PROTOCOL_DNS_SD,
                       myDiscoveryListener);
```

Maintain list of discovered services by using the DiscoveryListener

```
nsdMgr.stopServiceDiscovery(myDiscoveryListener);
```

DNS-SD

Use the NsdServiceInfo to get informations about the service

```
class NsdServiceInfo {
    String getServiceName();
     String getServiceType();
    InetAddress getHost();
     int getPort();
}
```

getHost() and getPort() return invalid result because the service is not resolved

DNS-SD

Resolve the service

Dns request to get IP address and port

```
nsdMgr.resolveService(serviceInfo, myResolveListener);

ResolveListener myResolveListener = new ResolveListener()
{
    public void onServiceResolved(NsdServiceInfo
serviceInfo) {
        InetAddress addr = serviceInfo.getHost();
        int port = serviceInfo.getPort();
        // create a socket and use it
    }
...
```

Network Service Discovery

Nice to know

Not limited to Wi-Fi Direct, works also in a LAN.
Cool feature but NsdService is very instable.

Evolutions

UPNP service discovery

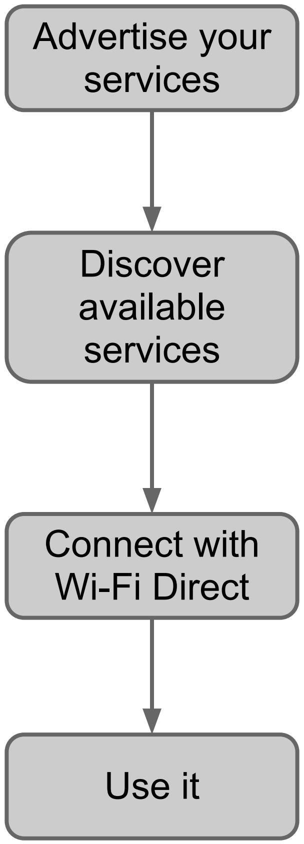
Pre-association service discovery

Discover which Wi-Fi Direct device offers a service before connecting

Available since Android JellyBean (4.1)

Use DNS-SD, UPNP, custom vendor protocol

Workflow


Advertise your services
Discover available services
Connect with Wi-Fi Direct
Use it

Pre-Association Service Discovery

android.net.wifi.p2p.WifiP2pManager

```
// advertise
void addLocalService(Channel c, WifiP2pServiceInfo servInfo,
                     ActionListener listener);

// discover
void addServiceRequest(Channel c, WifiP2pServiceRequest req,
                       ActionListener listener);
// method depends of the discovery protocol DNS-SD or UPNP
void set*ResponseListener(...);
void discoverServices(Channel c, ActionListener listener);
```

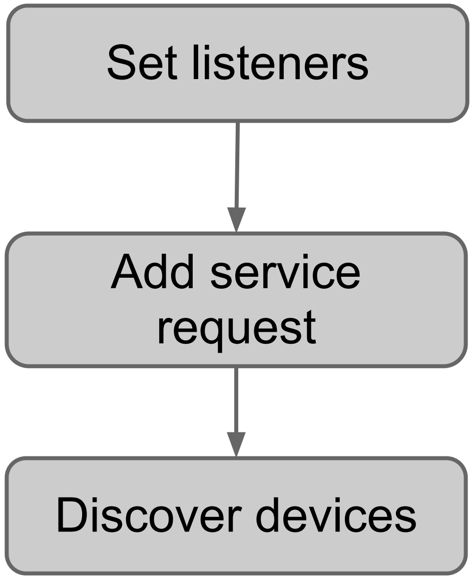
Pre-association service advertising

```
WifiP2pServiceInfo info;

info = WifiP2pDnsSdServiceInfo.newInstance(instanceName,
               serviceType, Map<String, String> txtMap);
// or
info = WifiP2pUpnpServiceInfo.newInstance(uuid,
                         device, List<String> services);

wifiMgr.addLocalService(channel, info, actionListener);
```

Pre-association service discovery



Pre-Association Service Discovery

UPNP

```
WifiP2pServiceRequest req;

req = WifiP2pUpnpServiceRequest.newInstance(ssdp);  //
various newInstance() method

wifiMgr.setUpnpServiceResponseListener(channel, new
UpnpServiceResponseListener() {
    public void onUpnpServiceAvailable(List<String>
uniqueServiceName, WifiP2pDevice srcDevice) {
        // connect to the device with wifi direct
        // and use the service
    }
});
wifiMgr.discoverServices(channel, actionListener);
```

Benefits

Simplify user workflow

Better user experience

It just works

Conclusion

Conclusion

Benefits of high speed transfer for every one

Base for futur usage and projects like :

- Mesh Networking (see the Serval Project)
- Wireless Display (Wi-Di, Miracast)
- Real Peer 2 Peer
- Yours ?

DevFest Nantes 2012

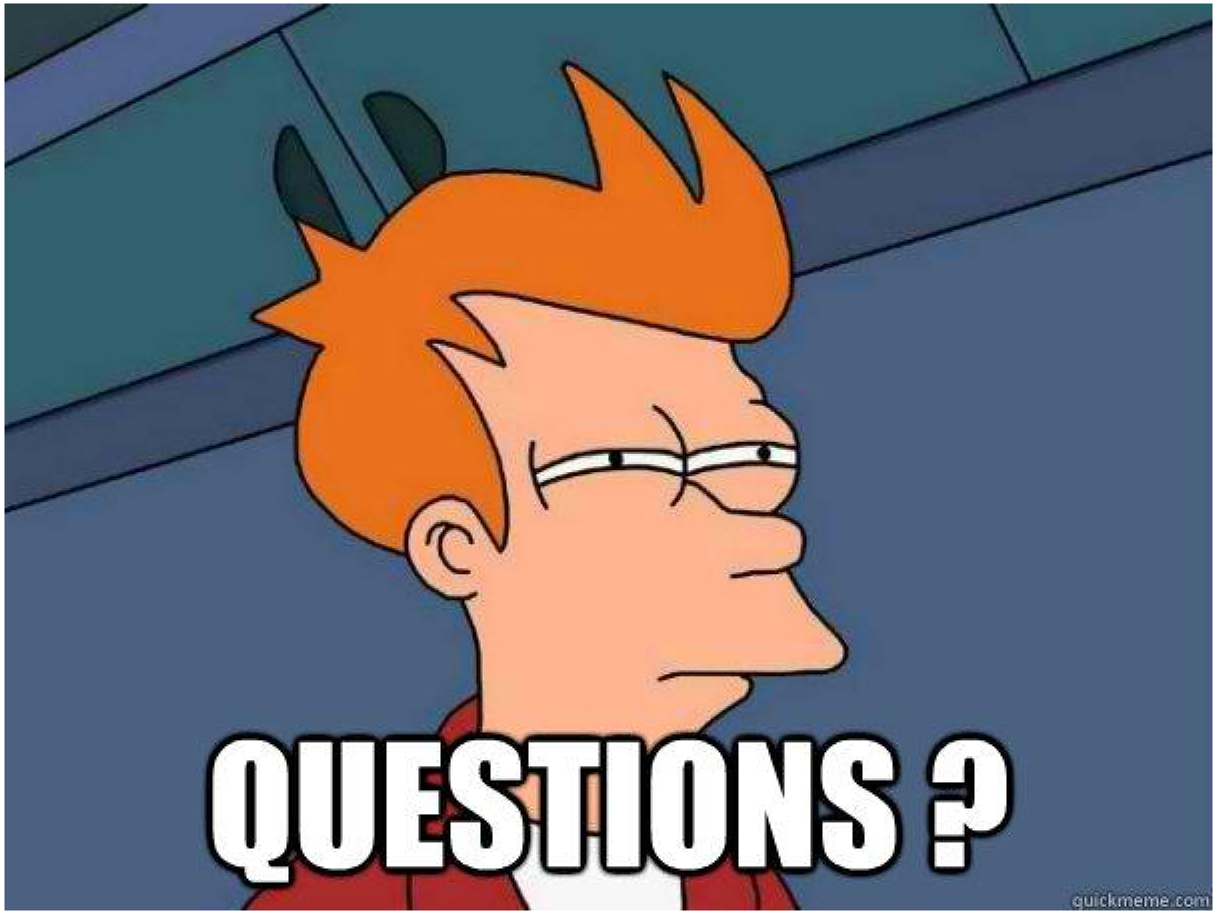
Bonus

APK: http://bit.ly/YWvXbX
CODE: http://hg.geekorum.com/P2PShare
CODE: http://github.com/fbarthelery/P2PShare

GENYMOBILE
Let IT be mobile

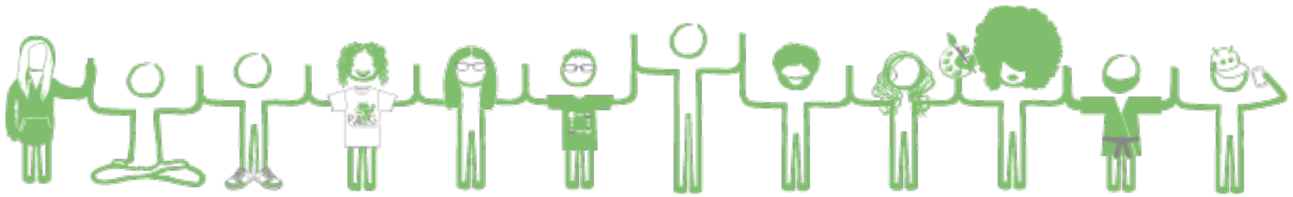
DevFest Nantes 2012

Questions ?



GENYMOBILE
Let IT be mobile

Thanks !



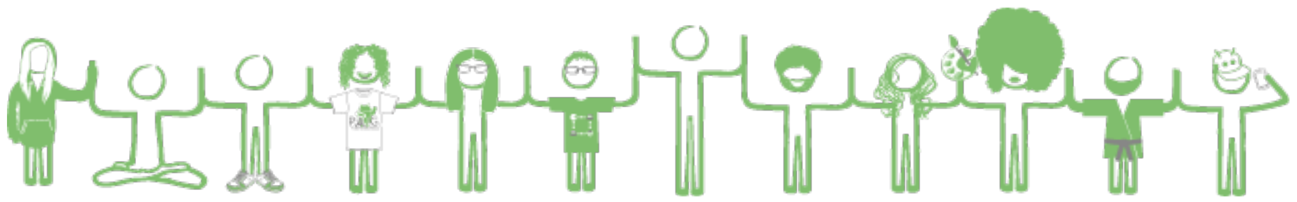
Thanks !

Contact

Frederic Barthelery
fbarthelery@genymobile.com
bart@geekorum.com

http://www.genymobile.com
http://www.beem-project.com

Extras



You want more ?

DNS-SD

Service advertising

```
private void registerService() {
    NsdServiceInfo serviceInfo = new NsdServiceInfo();
    serviceInfo.setPort(port);
    serviceInfo.setServiceName(myServiceName);
    serviceInfo.setServiceType(SERVICE_TYPE);
    nsdMgr.registerService(serviceInfo,
                           NsdManager.PROTOCOL_DNS_SD,
                           myRegistrationListener);
}

private void unregisterService() {
    nsdMgr.unregisterService(myRegistrationListener);
}
```

DNS-SD

```
RegistrationListener myRegistrationListener =
    new RegistrationListener() {
    @Override
    public void onServiceRegistered(NsdServiceInfo
serviceInfo) {
        // cool save the serviceName
        serviceName = serviceInfo.getServiceName();
    }

    @Override
    public void onRegistrationFailed(NsdServiceInfo
serviceInfo, int errorCode) {
        if (errorCode == NsdManager.FAILURE_INTERNAL_ERROR)
{
        // euh, what can I do ...
        }
    }
```

DNS-SD

```
    @Override
    public void onServiceUnregistered(String serviceType) {
        // cool
    }

    @Override
    public void onUnregistrationFailed(NsdServiceInfo
serviceInfo, int errorCode) {
        if (errorCode == NsdManager.FAILURE_MAX_LIMIT) {
            // TODO
        } else if (errorCode == NsdManager.
FAILURE_INTERNAL_ERROR) {
            // euh, what can I do ...
        }
    }
};
```

DNS-SD

Service discovery workflow

```
List<NsdServiceInfo> services =
       new ArrayList<NsdServiceInfo>();

nsdMgr.discoverService(serviceType,
                       NsdManager.PROTOCOL_DNS_SD,
                       myDiscoveryListener);

nsdMgr.stopServiceDiscovery(myDiscoveryListener);
```

DNS-SD

```
DiscoveryListener myDiscoveryListener =
    new DiscoveryListener() {
    @Override
    public void onDiscoveryStarted(String serviceType) {
        // cool
    }

    @Override
    public void onStartDiscoveryFailed(String serviceType,
int errorCode) {
        if (errorCode == NsdManager.FAILURE_MAX_LIMIT) {
        } else if (errorCode == NsdManager.
FAILURE_INTERNAL_ERROR) {
        // retry ? later ? abort ?
        }
    }
...
```

DNS-SD

```
    @Override
    public void onDiscoveryStopped(String serviceType) {
        // cool
    }

    @Override
    public void onStopDiscoveryFailed(String serviceType,
int errorCode) {
        if (errorCode == NsdManager.FAILURE_INTERNAL_ERROR)
{
        // euh, what can I do ...
        }
    }

...
```

DNS-SD

```
    // maintain list of available services
    @Override
    public void onServiceLost(NsdServiceInfo serviceInfo) {
        services.remove(serviceInfo);
    }

    @Override
    public void onServiceFound(NsdServiceInfo serviceInfo)
{
        services.add(serviceInfo);
    }
};
```

DNS-SD

```
    public void onResolveFailed(NsdServiceInfo
serviceInfo, int errorCode) {
        // check errorCode
        if (errorCode == NsdManager.
FAILURE_ALREADY_ACTIVE) {
            // a previous resolve request waiting for
answer.
        } else if (errorCode == NsdManager.
FAILURE_INTERNAL_ERROR) {
            // could be a lot of things...
            // but the NsdService has not been able to
solve the request
        }
    }
```

Pre-association service discovery

```
WifiP2p Service Discovery
android 4.1 JellyBean

Permet de decouvrir les peers qui offrent des services.
Permet d'annoncer ses services sans qu'une connexion soit
etablies.
```

Pre-association service discover workflow

```
WifiP2pServiceRequest req;
wifiMgr.set*Listener();
wifiMgr.addServiceRequest(channel, req, actionListener);
wifiMgr.discoverServices(channel, actionListener);
```

DNS-SD

```
req = WifiP2pDnsSdServiceRequest.newInstance(instanceName,
         serviceType); // various newInstance() method
wifiMgr.setDnsSdResponseListener(channel,
    new DnsSdServiceResponseListener() {
        public void onDnsSdServiceAvailable(string
instanceName, String registrationtype, WifiP2pDevice
srcDevice) {
            // connect to the device with wifi direct
        }
    },
    new DnsSdTxtRecordListener() {
        public void onDnsSdTxtRecordAvailable(String
fullDomainName, Map<String, String> txtRecordMap,
WifiP2pDevice srcDevice) {
            // connect to the device with wifi direct
        }
```