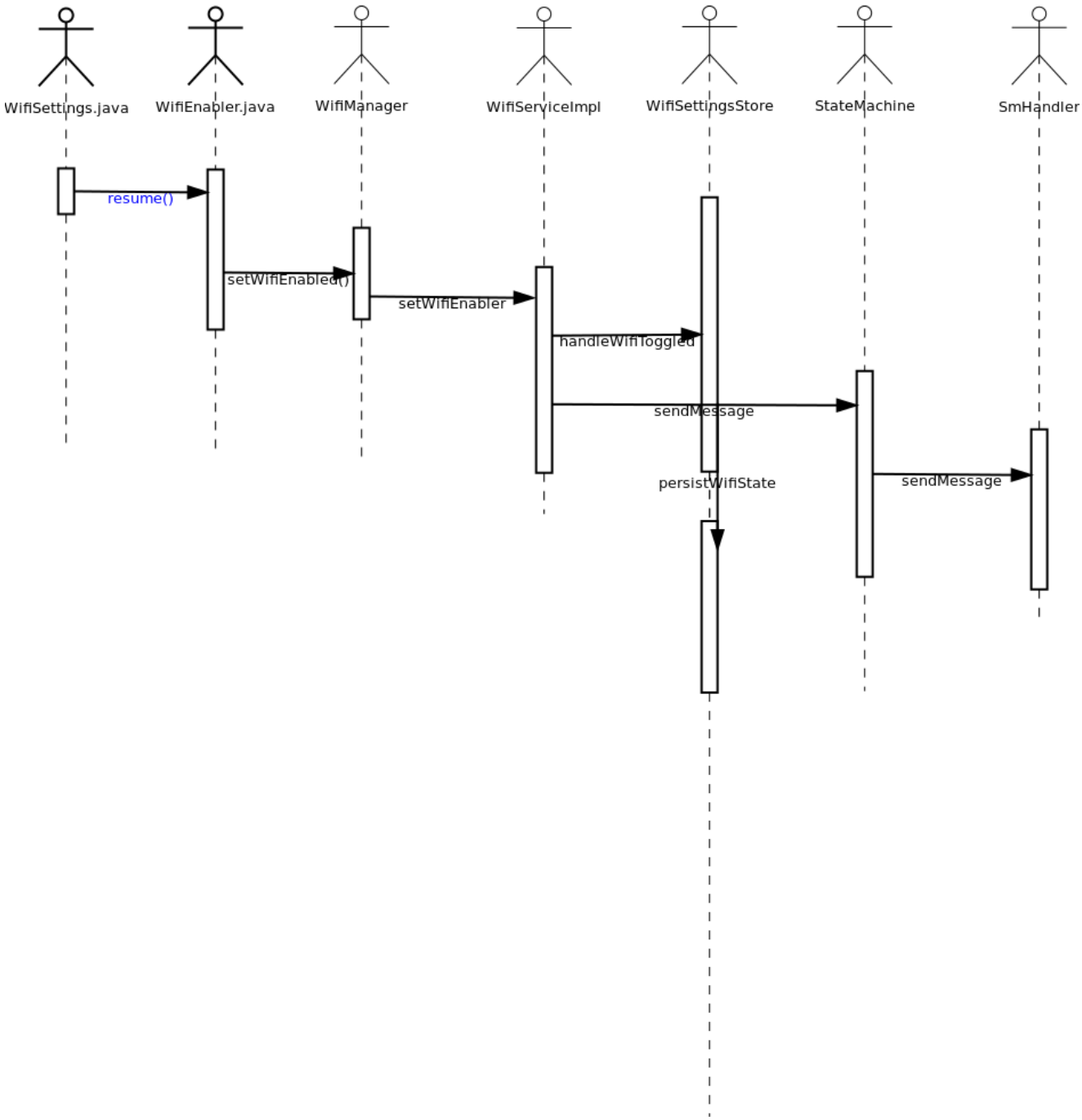
Wifi打开/关闭流程简析

涉及类

1. com.android.settings.wifi.WifiSettings
2. com.android.settings.wifi.WifiEnabler
3. android.net.wifi.WifiManager
4. com.android.server.wifi.WifiServiceImpl
5. com.android.server.wifi.WifiSettingsStore
6. com.android.internal.util.StateMachine
7. com.android.server.wifi.WifiController
8. com.android.server.wifi.WifiStateMachine
9. com.android.settingslib.wifi.WifiTracker

时序图



Settings内部流程

```
//WifiSetting.java:: onResume
if (mWifiEnabler != null) {
    //监听打开和关闭的SwitchBar
    mWifiEnabler.resume(activity);
}

//WifiEnabler.java
public void resume(Context context) {
    mContext = context;
    //注册开启/关闭后的广播
    // Wi-Fi state is sticky, so just let the receiver update UI
    mContext.registerReceiver(mReceiver, mIntentFilter);
    if (!mListeningToOnSwitchChange) {
        //设置SwitchBar的监听
        mSwitchBar.addOnSwitchChangeListener(this);
        mListeningToOnSwitchChange = true;
    }
}
//WifiEnabler.java
public WifiEnabler(Context context, SwitchBar switchBar) {
    ...
    mIntentFilter = new IntentFilter(WifiManager.WIFI_STATE_CHANGED_ACTION);
    // The order matters! We really should not depend on this. :(
    //这个很有可能是高通加的注释
    mIntentFilter.addAction(WifiManager.SUPPLICANT_STATE_CHANGED_ACTION);
    mIntentFilter.addAction(WifiManager.NETWORK_STATE_CHANGED_ACTION);
    ...
}

//WifiEnabler.java
@Override
public void onSwitchChanged(Switch switchView, boolean isChecked) {
    ...
        if (!mWifiManager.setWifiEnabled(isChecked)) {
            // Error
            mSwitchBar.setEnabled(true);
            Toast.makeText(mContext, R.string.wifi_error, Toast.LENGTH_SHORT).show();
        }
        return;
    ...
}
```

framework层分析

```
//WifiManager.java
public boolean setWifiEnabled(boolean enabled) {
    try {
        //mService是IWifiManager的实例
        return mService.setWifiEnabled(enabled);
    } catch (RemoteException e) {
        return false;
    }
}

//WifiServiceImpl.java
public synchronized boolean setWifiEnabled(boolean enable) {
    //检测是否有权限

    if(isStrictOpEnable()) {
        //严格模式的一些处理
    }

    /*
    * Caller might not have WRITE_SECURE_SETTINGS,
    * only CHANGE_WIFI_STATE is enforced
    */

    long ident = Binder.clearCallingIdentity();
    try {
        //存储WiFi打开/关闭的值
        if (! mSettingsStore.handleWifiToggled(enable)) {
            // Nothing to do if wifi cannot be toggled
            return true;
        }
    } finally {
        Binder.restoreCallingIdentity(ident);
    }

    if (!mIsControllerStarted) {
        Slog.e(TAG,"WifiController is not yet started, abort setWifiEnabled");
        return false;
    }
    //发送打开和关闭的消息给WifiController
    mWifiController.sendMessage(CMD_WIFI_TOGGLED);
    return true;
}

//WifiSettingsStore.java
synchronized boolean handleWifiToggled(boolean wifiEnabled) {
    // Can Wi-Fi be toggled in airplane mode ?
    if (mAirplaneModeOn && !isAirplaneToggleable()) {
        return false;
    }

    if (wifiEnabled) {
        //飞行模式

        if (mAirplaneModeOn) {
```

```
            persistWifiState(WIFI_ENABLED_AIRPLANE_OVERRIDE);
        } else {
            persistWifiState(WIFI_ENABLED);
        }
    } else {
        // When wifi state is disabled, we do not care
        // if airplane mode is on or not. The scenario of
        // wifi being disabled due to airplane mode being turned on
        // is handled handleAirplaneModeToggled()
        persistWifiState(WIFI_DISABLED);
    }
    return true;
}

//WifiSettingsStore.java
private void persistWifiState(int state) {
    final ContentResolver cr = mContext.getContentResolver();
    mPersistWifiState = state;
    Settings.Global.putInt(cr, Settings.Global.WIFI_ON, state);
}

//StateMachine.java
public final void sendMessage(int what) {
    // mSmHandler can be null if the state machine has quit.
    SmHandler smh = mSmHandler;
    if (smh == null) return;

    smh.sendMessage(obtainMessage(what));
}

//StateMachine.java::SmHandler
@Override
public final void handleMessage(Message msg) {
    if (!mHasQuit) {
    if (mDbg) mSm.log("handleMessage: E msg.what=" + msg.what);

    /** Save the current message */
    mMsg = msg;

    /** State that processed the message */
    State msgProcessedState = null;
    //状态机是否已启动
    if (mIsConstructionCompleted) {
        /** Normal path */
        //走这
        msgProcessedState = processMsg(msg);
    } else if (!mIsConstructionCompleted && (mMsg.what == SM_INIT_CMD)
            && (mMsg.obj == mSmHandlerObj)) {
        /** Initial one time path. */
        mIsConstructionCompleted = true;
        invokeEnterMethods(0);
    } else {

        throw new RuntimeException("StateMachine.handleMessage: "
```

```
                + "The start method not called, received msg: " + msg);
        }
        performTransitions(msgProcessedState, msg);

        // We need to check if mSm == null here as we could be quitting.
        if (mDbg && mSm != null) mSm.log("handleMessage: X");
        }
    }
```