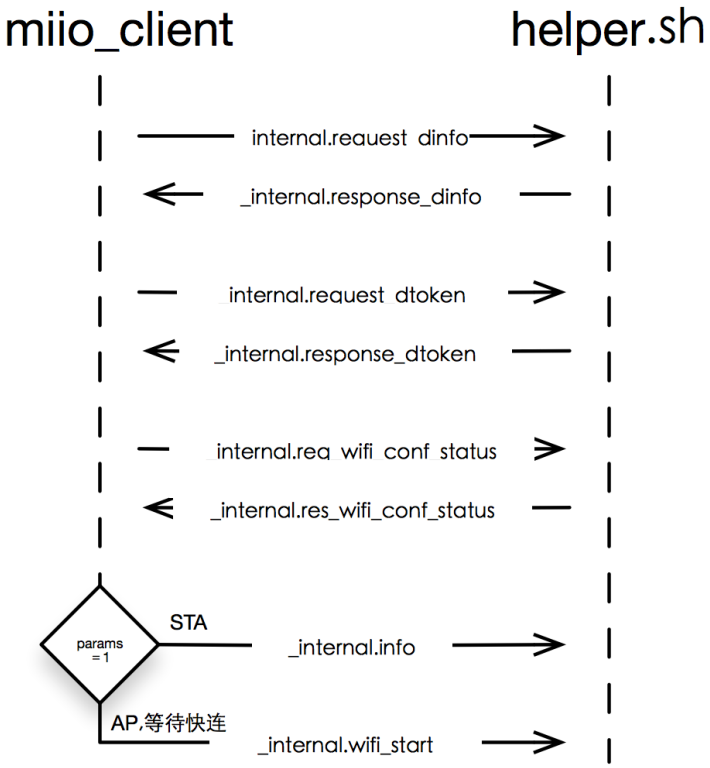
Helper 脚本使用说明

一，helper 和 miio_client 如何通信

helper 脚本和 miio_client 通过 miio_client_send_line、miio_client_recv_line 进行通信。前者负责发送，后者负责接收。

二、helper method 说明

miio_client 上电启动的时候，会通过几个 method 和 helper 进行信息交互，流程如下：



method 定义如下：

1、_internal.request_dinfo

请求设备信息，包括设备 did、key、mac 等。Helper 脚本会通过 method _internal.response_dinfo 进行回复

2、_internal.request_dtoken

请求设备 token，用来加密上报云端的信息，helper 回复 method 为：
_internal.response_dtoken

3、_internal.req_wifi_conf_status

请求 wifi 的配置状态，helper 脚本返回格式为：
"{\"method\":\"_internal.res_wifi_conf_status\",\"params\":x}"
params 为 1，表示 wifi 为 STATION 模式，为 0 表示 AP 模式。

4、_internal.info

请求设备状态信息，包括 ip、gateway、sw_version 等

5、_internal.wifi_start

进入快连时，miio_client 会将配置信息下发给 helper 脚本，格式为：

```
{"method":"_internal.wifi_start","params":{"datadir":"%s","ssid":"xx
x","passwd":"yyy","uid":"zzz", "country_domain":"xx", "tz":"xxx"}}
```

helper 脚本会解析对应参数，然后通过 wifi_start.sh 对 WiFi 进行配置。成功会向 miio_client 发送消息：

```
    "{\"method\":\"_internal.wifi_connected\"}"
```

失败重新回到 AP 模式，并发送消息：

```
    "{\"method\":\"_internal.wifi_ap_mode\",\"params\":null}"
```

6、_internal.config_tz

单独的时区配置接口，可选。

三，helper 设备配置文件说明

MIIO 用到的配置文件有

- /etc/miio/device.conf
- /etc/miio/device.token

- /etc/miio/wifi.conf
- /etc/miio/otd_donot_trust_peer
- /etc/os-release

1. /etc/miio/device.conf

设备配置文件，保存着设备独有的信息，例如：

```
# cat /etc/miio/device.conf
# did must be a unsigned int
# key must be a string
#
did=10000
key=EsrtdeaInabcPQM0
mac=8C:BE:BE:AA:bb:59
vendor=coolvendor
# model max len 23
model=coolvendor.prod.v1
```

其中的 did，key，mac 由小米智能家居项目组分配，请联系小米相关负责人。

2. /etc/miio/device.token

设备 token，每次设备重置（reset）的时候生成，主要用于设备跟用户绑定，快连，同一局域网下手机对设备直接控制等。第三方开发者不用关心这个文件。

3. /etc/miio/wifi.conf

设备快连完成之后，WiFi 用户名和密码存放文件。同时作为设备是否完成快连的标志文件存在。
第三方开发者注意：设备重置时，必须删除这个文件。

4. /etc/miio/otd_donot_trust_peer

手机快连配置成功，设备跟某个手机绑定之后，可以通过下面这条命令让设备不再信任别的手机的配置命令。

```
{"method":"miIO.config","params":{"enauth":1}}
{
        "method": "miIO.config",
       "params": {
```

```
                "enauth": 1
        }
}
```

/etc/miio/otd_donot_trust_peer 就是这个标志文件，当 miio_client 收到 enauth 配置命令之后，建立这个文件。第三方开发者不用关心这个文件。

5. /etc/os-release

/etc/os-release 是 Linux 标准的软件版本存放文件，我们用它来存放设备固件版本号。
注：你们可以根据设备的具体情况，将以上信息存储到合适的位置。

四、wifi_start.sh 作用

wifi_start.sh 的作用是根据 wifi 的配置是否完成，使 wifi 模块进入 AP 或者 STATION 模式，里面用到了两个重要的函数 wifi_ap_mode 以及 wifi_sta_mode，分别控制 WiFi 模块进入 AP 或者 STATION 模式。由于实际硬件千差万别，这里只能提供一个参考，你们可以根据自己设备的具体情况完成这两个功能。

五、AP 模式下 ssid 命名规则

启动 AP 快连的时候，会设置 AP 的 ssid，这个 ssid 的格式有一定规则，否则米家 app 不能自动发现这个设备。ssid 名字规则如下：
假设你的设备 model 是：vendor.camera.v1，那么 ssid 的名字必须是：

    vendor-camera-v1_miapAABB

其中 AABB 是设备 MAC 地址最后两个 bytes，比如 MAC 是 34:17:eb:9b:a7:47，那么 ssid 的名字是：

    vendor-camerav1_miapA747

六，时区信息配置

另外，你还需要修改 miio_client_helper.sh，以使 miio_client 支持时区配置。考虑到不同系统使用的文件系统格式不一样，对于只读文件系统，我们不能动态修改 /etc/localtime(glibc)或者/etc/TZ(uclibc)，所以我们这里给出一个通用的方式。

首先您需要创建一个/etc/localtime 或/etc/TZ（根据系统使用的 libc 库选择）到一个实际可读写的位置(YOUR_LINK_TIMEZONE_FILE)的软链接，(这一步需要在制作根文件的时候完成)，然后根据实际使用的 libc 库选择时区配置的路径：

```
GLIBC_TIMEZONE_DIR="/usr/share/zoneinfo"
UCLIBC_TIMEZONE_DIR="/usr/share/zoneinfo/uclibc"
```

假如使用的是 uclibc 库，链接位置为/mnt/TZ，则脚本实际配置如下：

```
YOUR_LINK_TIMEZONE_FILE="/mnt/TZ"
YOUR_TIMEZONE_DIR=$UCLIBC_TIMEZONE_DIR
```

helper 会根据实际下发的时区信息将/mnt/TZ 链接到对应时区。

七，如何运行

把下面这段脚本放入到你系统的启动脚本中：

```
/some/path/wifi_start.sh (or equivalent some other scripts)
/usr/bin/mosquitto -c /etc/mosquitto.conf –d (如果你使用 mqtt 版本的话)
/usr/bin/miio_client -D
/usr/bin/miio_client_helper.sh &
```

注：可以通过 miio_client –h 查看使用帮助

八，如何快连

为了方便用户将设备与米家 APP 进行绑定，MIIO SDK 提供了以下两种方式。

1．AP 方式快连

系统启动的时候，会调用 wifi_start.sh，这个脚本会检查设备是否已经配置好 wifi（通过检查文件/etc/miio/wifi.conf）。然后根据 wifi 是否配置好分两条路径。

1. 如果已经配置好，wifi_start.sh 会直接驱动 wifi 进入 STA 模式并且连入网络。在 wifi_start.sh 之后启动的 miio_client 尝试连接小米云。
2. 如果 wifi 没有配置好，wifi_start.sh 会驱动 wifi 进入 AP 模式。在 wifi_start.sh 之后启动的 miio_client 暂时不会跟小米云连接，而是监听特定 UDP 端口（54321）的包。如果收到本地手机的配置请求，miio_client 先发送{did, token, timestamp} 建立信任关系，然后手机把路由器的{ssid, passwd}发给 miio_client，格式为：

{"id":xx,"method":"miIO.config_router","params":{"ssid":"xx","passwd":"xx","uid":xx, "country_domain":"xx","tz":"Asia/Shanghai"}}


```
{
        "id": xx,
        "method": "miIO.config_router",
        "params": {
                "ssid": "xx",
                "passwd": "xx",
                "uid": xx,
                "country_domain": "xx",
                "tz": "Asia/Shanghai"
        }
}
```


其中，ssid、passwd、uid 为必选项，country_domain 和 tz 可以根据需要进行传递，如果设备在中国大陆，则 country_domain 可以忽略。

miio_client 收到{ssid,passwd}之后把他们写入 WiFi 配置文件/etc/miio/wifi.conf，调用 wifi_start.sh 切换 WiFi 到 STA 模式，开始尝试连接小米云。

2．蓝牙方式快连

蓝牙方式快连与 AP 方式类似，不同的地方是步骤 2 时，蓝牙方式快连会监听 UDP 端口（54323）的包。
米家 APP 下发的命令格式为：

{"id":xx,"method":"local.ble.config_router","params":{"ssid":"xx","passwd":"xx","uid":xx,"country_domain":"xx","tz":"Asia/Shanghai"}}


```
{
        "id": xx,
        "method": "local.ble.config_router",
        "params": {
                "ssid": "xx",
                "passwd": "xx",
                "uid": xx,
                "country_domain": "xx",
                "tz": "Asia/Shanghai"
        }
}
```

3．快连出错和重试

手机侧会验证用户输入的{ssid, passwd}，尽量保证没问题（可以正常连接路由器）；同时，设备端 miio_client 也会有出错重试次数（默认是 5 次，每次间隔 3s），重试完如果还是不成功，则回到 AP 快连状态。手机/用户可以重新开始快连。

注：在进行快连之前，首先要在米家开放平台创建新硬件，并将开发人员的 uid(小米 ID)添加到白名单，然后快连的时候米家 APP 才能发现设备。