# imobiledevice

### lockdown.h

1. lockdownd\_error\_t **lockdownd\_client\_new**(idevice\_t device, lockdownd\_client\_t \*client, const char \*label); 创建一个lockdown client

* device为已经建立连接的设备，client为需要建立lockdown所返回的句柄，label为方法；

1. lockdownd\_error\_t **lockdownd\_client\_new\_with\_handshake**(idevice\_t device, lockdownd\_client\_t \*client, const char \*label); 创建一个lockdown client并进行初始化的handshake

* device为已经建立连接的设备，client为需要建立lockdown所返回的句柄，label为方法；

1. lockdownd\_error\_t **lockdownd\_client\_free**(lockdownd\_client\_t client); 关闭lockdown client。
2. lockdownd\_error\_t **lockdownd\_query\_type**(lockdownd\_client\_t client, char \*\*type); 询问lockdown service的类型
3. lockdownd\_error\_t **lockdownd\_start\_service**(lockdownd\_client\_t client, const char \*identifier, lockdownd\_service\_descriptor\_t \*service); 开启一个lockdown service，

* Identifier 为需要开启的lockdown service的标识符，service判断是否成功；

1. lockdownd\_error\_t **lockdownd\_start\_session**(lockdownd\_client\_t client, const char \*host\_id, char \*\*session\_id, int \*ssl\_enabled); 在lockdown上打开一个session，如果有必要转为ssl模式；
2. lockdownd\_error\_t **lockdownd\_stop\_session**(lockdownd\_client\_t client, const char \*session\_id); 关闭lockdown session
3. lockdownd\_error\_t **lockdownd\_send**(lockdownd\_client\_t client, plist\_t plist);
4. lockdownd\_error\_t lockdownd\_receive(lockdownd\_client\_t client, plist\_t \*plist);
5. lockdownd\_error\_t **lockdownd\_pair**(lockdownd\_client\_t client, lockdownd\_pair\_record\_t pair\_record); 查看client是否在record中有匹配项
6. lockdownd\_error\_t **lockdownd\_get\_value**(lockdownd\_client\_t client, const char \*domain, const char \*key, plist\_t \*value); 检索key请求所对应的数据

* client 建立了lockdown的设备；domain请求发往的域；key请求；value返回的数据；

1. lockdownd\_error\_t **lockdownd\_set\_value**(lockdownd\_client\_t client, const char \*domain, const char \*key, plist\_t value); 设置key请求所对应的数据
2. lockdownd\_error\_t **lockdownd\_remove\_value**(lockdownd\_client\_t client, const char \*domain, const char \*key); 删除key请求所对应的数据