API & SDK information

API

For using the API, you need credentials. Please, contact to Omnitec to obtain your credentials.

As you can see in the API web https://api.rentandpass.com/explorer all functions are implemented, with the use and necessary parameters. You can test the function from this web to see results. By now, only format "application/json" is implemented for results. The rest are not implemented.

There you can get the necessary token after doing "Sign in" with your clientId, clientSecret and one User and Password you must create. Find below steps:

1.- Creating one user:

https://api.rentandpass.com/explorer/#!/user/user_register

2.- Sign in with your user:

https://api.rentandpass.com/api/signin/token?clientId=7e9b8c39aa11ec9afd45ecbe12d3eec8&clientSecret=abf1567fdb168c4edfb0f48fc4246c49&username=[**USUARIO**]&password=[**PASSWORD**]

And you receive a response similar to this:

```
{
  "access_token": "70dd7894ebc8999f52e842df7765e3c0",
  "refresh_token": "2ef7ecfad927dc2ecfadce36e1d2d853",
  "openid": 1211344576,
  "scope": "user,key,room",
  "expires_in": 3118226
}
```

This access_token is the one that must be sent to the application so that the user can operate. Remember that the client_secret should NEVER be used outside of your server. That is, it should never reach the user's application, because if the user accessed it in some way, it would have access to all your locks.

3.- Now you can use the token with the clientId to, for example, list the locks of that user.

https://api.rentandpass.com/api/lock/list?clientId=7e9b8c39aa11ec9afd45ecbe12d3eec8&token=70dd7894ebc8999f52e842df7765e3c0

Response:

```
{
  "list": [
    {
```

```
        "lockId": 1246554,
        "date": 1530345366340,
        "specialValue": 4337,
        "electricQuantity": 85,
        "lockAlias": "Lock",
        "keyboardPwdVersion": 4,
        "lockMac": "F3:BC:A0:17:48:0F",
        "lockName": "M301X_425234"
      }
    ],
    "pageNo": 1,
    "pageSize": 20,
    "pages": 1,
    "total": 1
}
```


SDKs

From the following link you can download all the information related to the SDK:

https://upkey.app/sdkfolder

Notes

Now our SDKs are restricted to the package name of each app. We need you to tell us what package name you are going to use to give yourself access. If you don't have it yet or want to do tests, you can use the package name "com.justfor.test".

When initializing a lock, the endpoint / lock / initialize must be called, sending the string returned by the onLockInitialize callback as the data parameter, and to collect the ekeys, the endpoint / key / list must be called with the sdkVersion = 3 parameter.

Quickstart guide

First of all, you need to create or register a new user, this is mandatory to initialize locks or send eKeys: POST /user/register

All the users are the same, there is no difference in permissions. These permissions are assigned through eKeys, those electronic keys may have admin privileges or just user privileges. All the lock's management options are limited to users with an eKey with admin privileges on that lock.

When a lock is initialized with an user, that user will receive an eKey with admin privileges that can't be deleted until that lock is reset.

Admin user process to initialize locks:

1. GET **/signin/token** – Store the *accessToken* parameter for the consecutive calls.
2. **SDK** --> startScanLock

3. **SDK** --> initLock
4. **POST /lock/initialize**

Process to send an eKey to another user:

1. **GET /signin/token** (OPTIONAL - Use this only if you haven't used it previously; this request is done from the lock's owner.)
2. Create user for the eKey receiver if it is not an already existing User (see /user/register)
3. **PUT /key/send** – *keyRight* specifies if the eKey has admin privileges.

Process to open a lock:

1. **GET /signin/token** (OPTIONAL - Only if it was not used previously; this request is done from the User that wants to open the lock)
2. **GET /key/list - *sdkVersion* = 3 – Stores the *lockData* parameter, this is the required parameter for all the SDK requests.**
3. **SDK** --> unlock

Before initializing the lock, please make sure the lock was not already initialized with another app and therefore, it is under factory settings, ready to be initialized.

To make a list for the locks nearby (in BLE range) that are in factory settings (see previous step), the request is:

```
device.isSettingMode(); //Android SDK

scanModel.isInited == NO //iOS SDK
```

This parameter will show if the device is ready to be initialized, apart from the previously explained, the lock must be "awake", you can "awake" the lock just by using a card on the device (even if it doesn't open) or by pushing a button on the device at that moment, when you are scanning the lock with that SDK request.

Before making the initLock request, it is mandatory to specify the lock's firmware version. The device object is picked from the ScanLock callback:

```
device.setManufacturerId(specialStr); // Android SDK

dict[@"clientPara"] = specialStr; //iOS SDK

specialStr = "XBY9E68A" //Rent&Pass type device

specialStr = "XBYMIYAVE" //Miyave type device
```

Any questions, we are at your disposal in soporte@omnitecsystems.com