

1、【000】 Authorization Verification

1. Request Data Format

JSON string, no encryption/decryption required.

2. Permission Authentication

The token returned from the login interface needs to be included in the request header for every API call.

Field name: `Authorization-Token`

Token validity period is one day. After expiration, you need to re-login to apply for a new token.

3. Response Format

JSON string, no encryption/decryption required.

```
{"code":200,"message":"sucess"}
```

Base URL

- <https://doordeer-usa.goolink.org>

2、【001】 Property Login

Brief Description:

- Administrator login interface
- Login credentials: Login name: [userxxx@email.com](#), Password: A123456b

Request URL:

- `/adminv2/Login/getLogIn`

Request Method:

- POST

Parameters:

Parameter	Required	Type	Description
account	Yes	string	User account
passwd	Yes	string	Password
phone_lang	Yes	int	Language value, set to: 2

Response Example

```
{
  "code": 200,
  "message": "sucess",
  "data": {
    "token": "92bb7fcf4cdd56ce255d29669ad436de",
    "buils_arr": [
      {
        "build_id": "10014303",
        "build_name": "Information Hub Building Office_1",
        "pay_model": "1",
        "builds_exptime": "2024-12-30 16:00:00",
        "timezone": "8",
        "rooms_arr": [
          {
            "unit_id": "10016000",
            "unit_label": "201"
          },
          {
            "unit_id": "10016001",
            "unit_label": "301"
          },
          {
            "unit_id": "10016002",
            "unit_label": "401"
          },
          {
            "unit_id": "10016003",
            "unit_label": "501"
          },
          {
            "unit_id": "10016004",
            "unit_label": "601"
          },
          {
            "unit_id": "10016005",
            "unit_label": "701"
          },
          {
            "unit_id": "10016006",
            "unit_label": "801"
          }
        ]
      }
    ]
  }
}
```

```

        "unit_id": "10016007",
        "unit_label": "0"
    },
    {
        "unit_id": "10016459",
        "unit_label": "303"
    }
],
"_0_room": {
    "id": "20028392",
    "gid_id": "10017117",
    "gid": "qqamzz62be531a",
    "building_id": "10014303",
    "account_id": "10011231",
    "main_or_sub_flag": "3",
    "ring_passwd_id": "10034106",
    "unit_id": "10016007",
    "vir_chn_no": "10016007",
    "unit_label": "0",
    "status": "1",
    "exptime": "2038-01-19 11:14:07",
    "gid_passwd": "295031",
    "devname": "qqamzz62be531a-test",
    "tmp_enabled": "1",
    "open_delay": "eyJCMDMiOjE0LCJCMDQiOjE1fQ==",
    "exptime_timestamp": "2147483647",
    "lock_passwd": "775376",
    "account": "",
    "passwd": "",
    "lock_id": [
        {
            "io_name": "B03",
            "io_other_name": "lock 1",
            "io_id": "10003785",
            "Enable": "1"
        },
        {
            "io_name": "B04",
            "io_other_name": "lock 2",
            "io_id": "10003786",
            "Enable": "1"
        }
    ]
}
},
"TokenArr": {
    "1": "423f4e5d41994cbbaed25c80cffff4a2",
    "5": "ffffffff-897e-a16e-ffff-ffffca01fdf4"
},
"comid": "10000220"
}
}

```

Response Status Code Description

code	message
200	success
403	Account and password do not match
400	Missing parameters
405	Request method not supported
407	Only property administrators are allowed to login
501	Login attempts exceeded 20 times, login restricted

Response Parameter Description

Parameter	Type	Description
build_id	int	Building ID, also called: building_id in other interfaces
unit_id	int	Individual room ID

3、【002】Property Get Room List

Brief Description:

- Property administrator gets room data list

Request URL:

- /adminv2/Rooms/getList

Request Method:

- POST

Parameters:

Parameter	Required	Type	Description
page	Yes	int	Page number
limit	Yes	int	Number of items to fetch
userid	No	string	User account
unit_label	No	int	User unit number
building_id	No	int	Building ID
status	No	int	Service status. 1=Active, 0=Inactive
zone_id	No	int	Agent ID
name	No	string	Building name

Request Example

```
{
```

```
"page": 1,  
"limit": 10  
}
```

Response Example

```
{  
  "code": 200,  
  "data": {  
    "total": 62,  
    "items": [{  
      "id": "10013982",  
      "status": "0",  
      "exptime": "2038-01-19 11:14:07",  
      "building_id": "10013853",  
      "unit_type": "Reside",  
      "unit_label": "202",  
      "floor_number": "2",  
      "owner_firstname": "",  
      "owner_lastname": "",  
      "unit_owner_phone": "",  
      "mail_address": "13420568677",  
      "tenant_email": "",  
      "vir_channel_no": "10013982",  
      "update_timestamp": "1700190997",  
      "update_localtime": "2024-02-01 15:29:52",  
      "comid": "10000005",  
      "name": "Test Property Push Notification Building",  
      "adminid": null,  
      "room_type": "1"  
    }],  
    "buildingsArr": {  
      "10000002": "Information Hub West Gate",  
      "10000003": "FaceID-qqazz020d47d3",  
      "10000004": "Langtao Test-qqazz0235a654",  
      "10000005": "H.Park C",  
      "10000006": "Liu-qqazz026352fa",  
      "10000008": "Test-qqazz02d0e16d",  
      "10000009": "Ma-qqazz02e25bba",  
      "10000010": "Test-qqamzz040b0d79",  
      "10000011": "test-qqaszz05583656",  
      "10000012": "test-qqamzz0452d3cd",  
      "10000013": "test-qqaszz0501d9bd",  
      "10000014": "test-qqaszz055f0bcd",  
      "10000015": "test-qqaszz0561dcfd",  
      "10000035": "test-qqamzz07235ae9",  
      "10000036": "test-qqamzz0726ec00",  
      "10000037": "test-qqamzz0733745b",  
      "10000049": "FaceID-qqamzz0752dc07",  
      "10000050": "FaceID-qqamzz07530252",  
      "10000051": "FaceID-qqamzz07544c46",  
      "10001168": "Management Office Test 09",  
      "10001659": "qqamzz182bb1de- Test",  
      "10001855": "test-qqamzz0751f9dd",  
      "10001978": "Test Building A",  
    }  
  }  
}
```

```
        "10013852": "asd",
        "10013853": "Test Property Push Notification Building",
        "10013858": "123",
        "10001793": "Valero Tower",
        "10001883": "alan test",
        "10013857": "111111"
    },
    "zone_arr": {
        "1": "XXXXaaa",
        "25": "PTS Corporation",
        "26": "GBF test111",
        "29": "Test aaaa",
        "31": "This is a test building 888"
    }
}
```

Response Status Code Description

code	message
200	success
500	Network error

Response Parameter Description

Parameter	Type	Description
total	int	Total count
items	array	Building array
buildingsArr	array	Building array
zone_arr	array	Community array

Remarks

4、【003】Property Create Room

Brief Description:

- Property administrator creates a room

Request URL:

- /adminv2/Rooms/CrateRoom

Request Method:

- POST

Parameters:

Parameter	Required	Type	Description
building_id	Yes	int	Building ID
unit_type	No	string	Room type
unit_label	Yes	string	Room number
floor_number	No	int	Floor number
owner_firstname	No	string	First name
owner_lastname	No	string	Last name
unit_owner_phone	No	int	Owner phone
mail_address	No	string	Owner account
name	No	string	Building name

Response Example

```
{
  "code": 200,
  "message": "sucess"
}
```

Response Status Code Description

code	message
200	success
500	Error message

Remarks

5、【004】Property Edit Room

Brief Description:

- Property administrator edits a room

Request URL:

- /adminv2/Rooms/updateRoom

Request Method:

- POST

Parameters:

Parameter	Required	Type	Description
id	Yes	int	Room ID
unit_type	No	string	Room type
unit_label	No	string	Room number
floor_number	No	int	Floor number
owner_firstname	No	string	First name
owner_lastname	No	string	Last name
display_owner_firstname	No	string	Display first name
display_owner_lastname	No	string	Display last name
unit_owner_phone	No	int	Owner phone
mail_address	No	string	Owner account
name	No	string	Building name
func_flag	No	int	Property switch

Response Example

```
{
  "code": 200,
  "message": "sucess"
}
```

Response Status Code Description

code	message
200	success
500	Error message

Remarks

6、【005】Property Delete Room

Brief Description:

- Property administrator deletes a room

Request URL:

- /adminv2/Rooms/DeleteRoom

Request Method:

- POST

Parameters:

Parameter	Required	Type	Description
id	Yes	int	Room ID

Response Example

```
{
  "code": 200,
  "message": "sucess"
}
```

Response Status Code Description

code	message
200	success
500	Error message

Remarks

7、 Demo

7.1、 Demo.py

```
# -*- coding: utf-8 -*-
import requests
import json
import sys
import io

BASE_URL = "https://doordeer-usa.goolink.org"

# Token obtained from test_getLogIn is automatically cached for reuse by other methods
_token = ""

def test_getLogIn(account, passwd):
    """Login interface, returns (token, response data)"""
    url = BASE_URL + "/adminv2/Login/getLogIn"
    params = {"account": account, "passwd": passwd, "phone_lang": 2}
    resp = requests.get(url, params=params)
    data = resp.json()
    print(json.dumps(data, indent=2, ensure_ascii=False))

    global _token
    if data.get("code") == 200:
        _token = data["data"]["token"]
    return _token, data
```

```

def test_getRoomList(page=1, limit=10, building_id=""):
    """Get room list"""
    url = BASE_URL + "/adminv2/Rooms/getList"
    headers = {"Authorization-Token": _token}
    params = {"page": page, "limit": limit, "building_id": building_id}
    resp = requests.get(url, params=params, headers=headers)
    data = resp.json()
    print(json.dumps(data, indent=2, ensure_ascii=False))
    return data

def test_CrateRoom(building_id, unit_label, **kwargs):
    """Create room
    Optional parameters: unit_type, floor_number, owner_firstname, owner_lastname,
                        unit_owner_phone, mail_address, name
    """
    url = BASE_URL + "/adminv2/Rooms/CrateRoom"
    headers = {"Authorization-Token": _token}
    params = {"building_id": building_id, "unit_label": unit_label}
    params.update(kwargs)
    resp = requests.get(url, params=params, headers=headers)
    data = resp.json()
    print(json.dumps(data, indent=2, ensure_ascii=False))
    return data

def test_updateRoom(room_id, unit_label, mail_address, **kwargs):
    """Update room
    Required: room_id
    Optional parameters: unit_type, unit_label, floor_number, owner_firstname, owner_lastname,
                        display_owner_firstname, display_owner_lastname, unit_owner_phone,
                        mail_address, name
    """
    url = BASE_URL + "/adminv2/Rooms/updateRoom"
    headers = {"Authorization-Token": _token}
    params = {"id": room_id, "unit_label": unit_label, "mail_address": mail_address}
    params.update(kwargs)
    resp = requests.get(url, params=params, headers=headers)
    data = resp.json()
    print(json.dumps(data, indent=2, ensure_ascii=False))
    return data

def test_DeleteRoom(room_id):
    """Delete room"""
    url = BASE_URL + "/adminv2/Rooms/DeleteRoom"
    headers = {"Authorization-Token": _token}
    params = {"id": room_id}
    resp = requests.get(url, params=params, headers=headers)
    data = resp.json()
    print(json.dumps(data, indent=2, ensure_ascii=False))
    return data

test_getRoomList(1, 10, 32556)

```