WAC and Modern Forms Smart Fans IP API Definition for 3rd Party Integrations

The API allows for retrieving fan information, as well as sending commands to the device.

This file should only be considered accurate for the firmware version it is tagged with in WAC source control.

Introduction

This document is intended to assist third parties with integrating WAC and Modern Forms smart fans into their control systems. The document is divided into three sections:

- Commissioning and Setup
- Discovery
- Status and Control

WAC and Modern Forms use standard RESTful API's and JSON data formatting.

APIs are tested using CURL (https://curl.haxx.se/)

Commissioning and Setup

Fans are delivered from the factory ready to be commissioned to the WAC/Modern Forms cloud through the Modern Forms or WAC app. The fan will start up as a Wi-Fi access point named:

ModernFormsFan_XXXXXX //For Modern Forms fans

WACFan_XXXXXX //For WAC Brand fans

In either case XXXXXX is the last six hexadecimal digits of the Wi-Fi station MAC address for the fan.

NOTE: Generation 3 fan receivers have different MAC Addresses when operating in Station mode versus Access Point mode. All operations using the fan use the station MAC address regardless of operating mode.

Standard procedure is to use the WAC or Modern Forms app, available on the Apple App Store or Google Play to configure the fan to connect to the local Wi-Fi network and register it to the cloud. It is strongly recommended that the fan be setup using the app.

A fan can be commissioned manually using a REST API but it is not supported by WAC/Modern Forms. For information on manual commissioning, please contact WAC technical support.

Discovery

Once commissioned, the fan will connect to the local Wi-Fi network in station mode and will respond to mDNS queries with its service information. WAC and Modern Forms fans can be identified by filtering on type `"easylink"` and protocol `"tcp"`.

Modern Forms will return an `fqdn` value of `WAC Windermier Fan#XXXXXX._easylink._tcp.local` for Generation 1 and 2 Fans, and `MF_Fan_XXXXXX._easylink._tcp.local` for Generation 3 Fans.

WAC Fans will return an `fqdn` value of `WAC_Fan_XXXXXX._easylink._tcp.local`.

Example response for Generation 3 Fan using Bonjour as a service discovery

```
{
    addresses: ["192.168.1.241"],
    fqdn: "MF_Fan_98E5AC._easylink._tcp.local",
    host: "MF_Fan_98E5AC.local",
    name: "MF_Fan_98E5AC",
    port: 80,
    protocol: "tcp",
    rawTxt: <Buffer(75)>,
    referer: {
        address: "192.168.1.241",
        family: "IPv4",
        port: 5353,
        size: 197,
    },
    subtypes: [],
    txt: {
        firmware rev: "02.99.0011",
        mac: "C8:2B:96:98:E5:AC",
        protocol: "com.modernforms.fan"
    },
    type: "easylink"
}
```


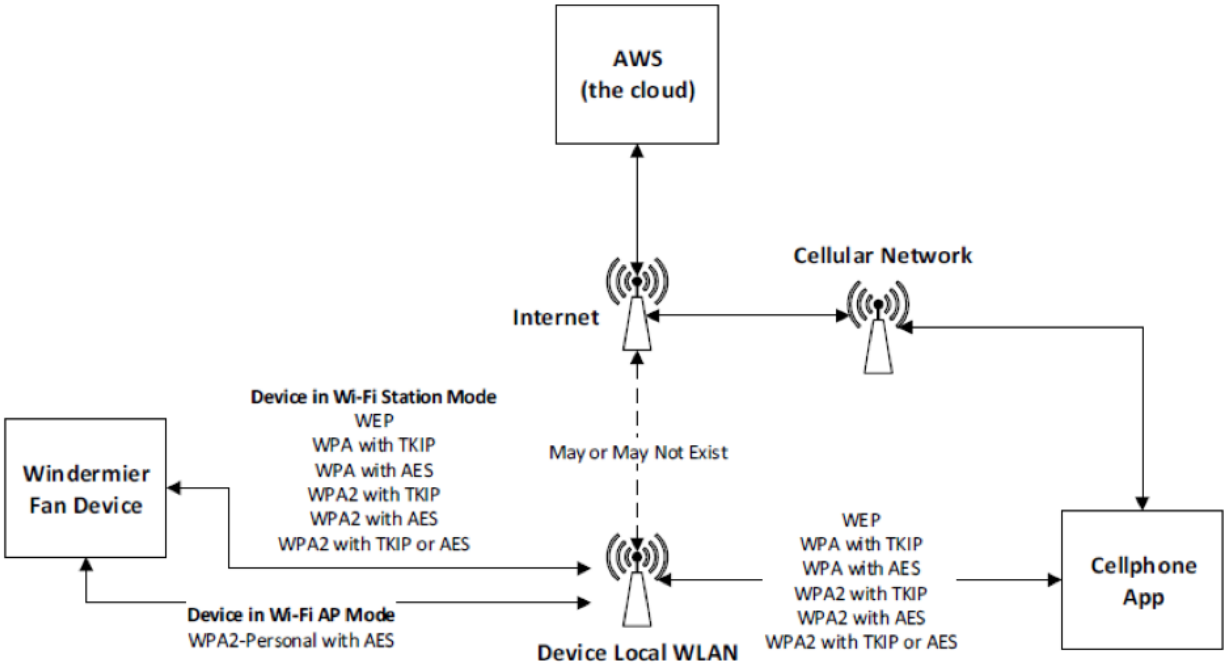

Figure 1: High Level Connections

Status and Control

Read the Fan Status

Retrieve configuration information about the device, such as firmware and hardware versions

Generation 1, 2 Fans

```
ANY /config-read
```

Example request

```
$ curl -X GET 192.168.1.85/config-read
```

Example response


```
{
    "T": "Current Configuration",
    "N": "WAC Windermier Fan(83DEF0)",
    "C": [
        {
            "N": "APPLICATION",
            "C": []
        }
    ],
    "PO": "com.modernforms.fan",
    "HD": "WAC_WINDERMIER_REV_5",
    "FW": "01.03.0021",
    "RF": "wl0: Oct  6 2016 01:32:44 version 5.90.230.15 ",
    "certificateId": "6v6amxh5vbb2qjnkrp2av8i8r1tk1svzwn4ktrr9ds2ljz65ycfq1
    "Wi-Fi strength": 100
}
```

Property	Description
`T`	Static string with the value of "Current Configuration"
`N`	Device Name associated with the Receiver
`C`	Unused.
`PO`	Protocol used for mDNS discovery
`HD`	Hardware revision
`FW`	Firmware Version
`RF`	RF library version
`certificateId`	Identifier associated with the certificates on the deivce
`Wi-Fi strength`	WiFi signal strength represented as signal quality percentage

Generation 3 Fans

```
POST /config-read
```

Example request

```
$ curl -X POST 192.168.1.85/config-read
```

Example response

```
{
    "Name": "MF_Fan_98E5AC",
    "Protocol": "com.modernforms.fan",
    "Firmware Rev": "02.00.0003",
    "RF Rev": "v3.2.2",
    "certificateId": "6v6amxh5vbb2qjnkrp2av8i8r1tk1svzwn4ktrr9ds2ljz65ycfq1
    "Wi-Fi strength": "-48"
}
```

Property	Description
`Name`	Device Name associated with the Receiver
`Protocol`	Protocol used for mDNS discovery
`Firmware Rev`	Firmware Version
`RF Rev`	RF library version
`certificateId`	Identifier associated with the certificates on the deivce
`Wi-Fi strength`	WiFi signal strength represented as a dB value

Read the Fan Static Shadow Data

Returns the shadow data properties that are not expected to be changed often, such as owner, or never, such as fanType.

Generation 1, 2, 3 Fans

```
POST /mf
```

Example request

```
$ curl -H "Content-Type: application/json" -d '{"queryStaticShadowData":1
}' -X POST 192.168.1.85/mf
```

Example request headers

```
{
  "Content-Type": application/json
}
```

Example request body

```
{
  "queryStaticShadowData": 1
}
```

Property	Required/Optional	Data Type	Range
`queryStaticShadowData`	Required	Integer	[1]

Example response for Generation 1, 2 Fans


```
{
    "clientId": "MF_C8934683DEF0",
    "mac": "C8:93:46:83:DE:F0",
    "lightType": "F4IN-120V-R1-30",
    "fanType": "1803-52",
    "fanMotorType": "DC125X20",
    "productionLotNumber": "",
    "productSku": "",
    "owner": "developer@waclighting.com",
    "federatedIdentity": "us-east-1:ef3pcxfz-2fngy-fn1cth-0kw0q0-ziitsbiz2w
    "deviceName": "Living Room",
    "firmwareVersion": "01.03.0021",
    "mainMcuFirmwareVersion": "01.03.3008",
    "firmwareUrl": ""
}
```


Property	Description	Data Type	Range
`clientId`	Unique identifer specific to a single device	String	MF_xxxxxxxxxxxx or WAC_xxxxxxxxxxxx
`mac`	The Media Access Control address of the fan, used to uniquely identify this device on the network. The x's are the ASCII hex representation of the device's 6 MAC address bytes.	String	xx:xx:xx:xx:xx:xx
	The type of light assembly installed		

`lightType`	in the device, if applicable	String	printable ASCII
`fanType`	Combination string of fan model number and blade length	String	printable ASCII
`fanMotorType`	Identifying string for the motor type in the fan	String	printable ASCII
`productionLotNumber`	The production lot that the device was manufactured in	String	printable ASCII
`productSku`	The SKU # of the device	String	printable ASCII
`owner`	The email address associated with the account used to pair the device	String	printable ASCII
`federatedIdentity`	Unique identifier from AWS specific to a single user	String	printable ASCII
`deviceName`	The name given to the device when paired or edited in the app	String	printable ASCII
`firmwareVersion`	The current firmware version the device is using	String	printable ASCII
	The current version of firmware that is the application-specific MCU that electrically controls		

`mainMcuFirmwareVersion`	the fan. This is not needed by anything right now but may be used in the future for things like RMA units. The d's are digits 0-9, where the first section represents the major version, second is the minor version, and last section is the build version	String	dd.dd.dddd
`firmwareUrl`	URL that tells the device where to download the latest version of firmware	String	printable ASCII

Example response for Generation 3 Fans


```
{
    "clientId": "MF_C82B9698E5AC",
    "mac": "C8:2B:96:98:E5:AC",
    "lightType": "",
    "fanType": "2003-52",
    "fanMotorType": "DC125X12",
    "brand": 0,
    "dateCode": "",
    "owner": "developer@waclighting.com",
    "federatedIdentity": "us-east-1:ef3pcxfz-2fngy-fn1cth-0kw0q0-ziitsbiz2w
    "deviceName": "Living Room",
    "firmwareVersion": "02.00.0003",
    "mainMcuFirmwareVersion": "02.01.0000",
    "firmwareUrl": ""
}
```


Property	Description	Data Type	Range

`clientId`	Unique identifer specific to a single device	String	MF_xxxxxxxxxxx or WAC_xxxxxxxxxxx
`mac`	The Media Access Control address of the fan, used to uniquely identify this device on the network. The x's are the ASCII hex representation of the device's 6 MAC address bytes.	String	xx:xx:xx:xx:xx:xx
`lightType`	The type of light assembly installed in the device, if applicable	String	printable ASCII
`fanType`	Combination string of fan model number and blade length	String	printable ASCII
`fanMotorType`	Identifying string for the motor type in the fan	String	printable ASCII
`brand`	Integer used to indicate a device's brand	Integer	[0 - 2]
`dateCode`	Date that certificates were loaded onto the device	String	printable ASCII
`owner`	The email address associated with the account used to	String	printable ASCII

	pair the device		
`federatedIdentity`	Unique identifier from AWS specific to a single user	String	printable ASCII
`deviceName`	The name given to the device when paired or edited in the app	String	printable ASCII
`firmwareVersion`	The current firmware version the device is using	String	printable ASCII
`mainMcuFirmwareVersion`	The current version of firmware that is the application-specific MCU that electrically controls the fan. This is not needed by anything right now but may be used in the future for things like RMA units. The d's are digits 0-9, where the first section represents the major version, second is the minor version, and last section is the build version	String	dd.dd.dddd
`firmwareUrl`	URL that tells the device where to download the	String	printable ASCII

	latest version of firmware		

Read the Fan Dynamic Shadow Data

Returns the shadow data properties that change often during normal operation, such as the light brightness or fan speed value.

Generation 1, 2, 3 Fans

```
POST /mf
```

Example request

```
$ curl -H "Content-Type: application/json" -d '{"queryDynamicShadowData": 1}' -X POST 192.168.1.85/mf
```

Example request headers

```
{
  "Content-Type": application/json
}
```

Example request body

```
{
  "queryDynamicShadowData": 1
}
```

Property	Required/Optional	Data Type	Range
`queryDynamicShadowData`	Required	Integer	[1]

Example response for Generation 1, 2 Fans

```
{
    "clientId": "MF_C8934683DEF0",
    "lightOn": false,
    "fanOn": false,
    "lightBrightness": 25,
    "fanSpeed": 1,
    "fanDirection": "forward",
    "rfPairModeActive": false,
    "resetRfPairList": false,
    "factoryReset": false,
    "awayModeEnabled": false,
    "fanSleepTimer": 0,
    "lightSleepTimer": 0,
    "decommission": false,
    "schedule": "AAAAAPwDAGSgBQAAnAkAZEALAAA8DwBk4BAAANwUAGSAFgAAfBoAZCAcAA
    "adaptiveLearning": false
}
```


In the following table, the use of the word "device" refers to the entire product as a whole, while "fan" and "light" refer to the individual components themselves.

Property	Description	Data Type	Range
`clientId`	Unique identifer specific to a single device	String	MF_xxxxxxxxxxxx or WAC_xxxxxxxxxxxx
`lightOn`	Indicates if the light is On or Off	Boolean	[true, false]
`lightBrightness`	Controls the brightness of the light	Integer	[1-100]
`fanOn`	Indicates if fan is On or Off	Boolean	[true, false]
`fanSpeed`	Controls the speed of the fan	Integer	[1-6]
`fanDirection`	Indicates if fan is in Summer Mode (`forward`) or Winter	String	["forward", "reverse"]

	Mode (`reverse`)		
`rfPairModeActive`	Indicates if the device is in Pairing Mode. Set to `true` to pair devices such as remotes or wall controls	Boolean	[true, false]
`resetRfPairList`	Boolean value indicating if device should clear all paired devices. Setting to `true` would clear all paired devices	Boolean	[true, false]
`factoryReset`	Boolean value indicating if device should be factory reset. Setting to `true` would initiate factory reset process. This type of reset erases all home AP credentials (SSID and password), puts the device back into Wi-Fi AP mode, decommissions the device from the cloud, and clears all previously paired devices	Boolean	[true, false]
`awayModeEnabled`	Boolean value indicating if device should be operate in Away Mode. Away Mode will control the device to simulate that a user is present	Boolean	[true, false]
`fanSleepTimer`	Epoch timestamp at which the fan should turn off	Integer	[0-2147483647]
`lightSleepTimer`	Epoch timestamp at which the light should turn	Integer	[0-2147483647]

	off		
`decommission`	Boolean value indicating if device should be decommissioned. Setting to `true` would initiate the decommission process. The device will decommission itself from the cloud, if it has access, then set Wi-Fi credentials back to factory default settings and enter AP mode.	Boolean	[true, false]
`schedule`	Base64 Encoded string value representing scheduled events to control fan power and speed, and light power and brightness	String	All Base64 Bytes
`adaptiveLearning`	Boolean value indicating if device should be operate in Adaptive Learning Mode. Adaptive Learning Mode will study a user's behavior and control the device to mimic their habits	Boolean	[true, false]

Example response for Generation 3 Fans

```
{
    "clientId": "MF_C82B9698E5AC",
    "lightOn": true,
    "fanOn": true,
    "lightBrightness": 100,
    "fanSpeed": 1,
    "fanDirection": "forward",
    "wind": false,
    "windSpeed": 2,
    "rfPairModeActive": false,
    "resetRfPairList": false,
    "factoryReset": false,
    "awayModeEnabled": false,
    "fanTimer": 0,
    "lightTimer": 0,
    "decommission": false,
    "schedule": "AAAAAPwDAGSgBQAAnAkAZEALAAA8DwBk4BAAANwUAGSAFgAAfBoAZCAcAA
    "adaptiveLearning": false,
    "userData": "cloud"
}
```


In the following table, the use of the word "device" refers to the entire product as a whole, while "fan" and "light" refer to the individual components themselves.

Property	Description	Data Type	Range
`clientId`	Unique identifer specific to a single device	String	MF_xxxxxxxxxxxx or WAC_xxxxxxxxxxxx
`lightOn`	Indicates if the light is On or Off	Boolean	[true, false]
`lightBrightness`	Controls the brightness of the light	Integer	[1-100]
`fanOn`	Indicates if fan is On or Off	Boolean	[true, false]
`fanSpeed`	Controls the speed of the fan	Integer	[1-6]
	Indicates if fan is in		

`fanDirection`	Summer Mode (`forward`) or Winter Mode (`reverse`)	String	["forward", "reverse"]
`wind`	Boolean value indicating if device is in operating in Wind Mode (`true`) or not (`false`). Wind Mode mimics a natural breeze by setting a base speed, and then increasing the fan speed at randomly timed intervals to simulate a gust of wind, then returning to the base speed	Boolean	[true, false]
`windSpeed`	Integer value ranging from 1-3 indicating the strength of the base and gust fan speeds	Integer	[1-3]
`rfPairModeActive`	Indicates if the device is in Pairing Mode. Set to `true` to pair devices such as remotes or wall controls	Boolean	[true, false]
`resetRfPairList`	Boolean value indicating if device should clear all paired devices. Setting to `true` would clear all paired devices	Boolean	[true, false]
	Boolean value indicating if device should be factory reset. Setting to `true` would initiate factory reset process. This type of		

`factoryReset`	reset erases all home AP credentials (SSID and password), puts the device back into Wi-Fi AP mode, decommissions the device from the cloud, and clears all previously paired devices	Boolean	[true, false]
`awayModeEnabled`	Boolean value indicating if device should be operate in Away Mode. Away Mode will control the device to simulate that a user is present	Boolean	[true, false]
`fanTimer`	Integer value indicting how many second until the fan should turn off	Integer	[0-2147483647]
`lightTimer`	Integer value indicting how many second until the light should turn off	Integer	[0-2147483647]
`decommission`	Boolean value indicating if device should be decommissioned. Setting to `true` would initiate the decommission process. The device will decommission itself from the cloud, if it has access, then set Wi-Fi credentials back to factory default settings and enter AP mode.	Boolean	[true, false]
`schedule`	Base64 Encoded string value representing scheduled events to control fan power and	String	All Base64 Bytes

	speed, and light power and brightness		
`adaptiveLearning`	Boolean value indicating if device should be operate in Adaptive Learning Mode. Adaptive Learning Mode will study a user's behavior and control the device to mimic their habits	Boolean	[true, false]
`userData`	Indicates the last method used to control the device	String	["none", "local_wifi", "cloud", "ble", "sleep_timer", "away_mode", "schedule"]

Control the Fan

Send commands to the device, such as turning the fan on or setting the light brightness to 100.

Generation 1, 2, 3 Fans

```
POST /mf
```

Example request

```
$ curl -H "Content-Type: application/json" -d '{"lightOn": true, "lightBr
ightness": 100}' -X POST 192.168.1.85/mf
```

Example request headers

```
{
  "Content-Type": application/json
}
```

Example request body

```
{
  "lightOn": true,
  "lightBrightness": 100
}
```

Property	Required/Optional	Data Type	Range
`lightOn`	Optional	Boolean	[true, false]
`lightBrightness`	Optional	Integer	[1-100]
`fanOn`	Optional	Boolean	[true, false]
`fanSpeed`	Optional	Integer	[1-6]
`fanDirection`	Optional	String	["forward", reverse"]
`wind`	Optional *(Gen 3+ Only)*	Boolean	[true, false]
`windSpeed`	Optional *(Gen 3+ Only)*	Integer	[1-3]

Example response

Reponse will be the dynamic shadow data, with the updated values

```
{
    "clientId": "MF_C82B9698E5AC",
    "lightOn": true,
    "fanOn": true,
    "lightBrightness": 100,
    "fanSpeed": 1,
    "fanDirection": "forward",
    "wind": false,
    "windSpeed": 2,
    "rfPairModeActive": false,
    "resetRfPairList": false,
    "factoryReset": false,
    "awayModeEnabled": false,
    "fanTimer": 0,
    "lightTimer": 0,
    "decommission": false,
    "schedule": "",
    "adaptiveLearning": false,
    "userData": "local_wifi"
}
```