APPARENT

Doxie Go Wi-Fi

HTTP/JSON API Developer Guide

Revision 2: March 31, 2015

Introduction

Doxie Go Wi-Fi is Apparent's battery-powered, wireless mobile document scanner. Doxie is intended for use with the Doxie app, but it can also be accessed programmatically to develop custom workflow solutions. Doxie's API is available on port 8080 and uses HTTP and JSON. This document explains how to communicate with the scanner, obtain status information, access scans, and delete scans. The `curl` command is used extensively in this documentation to demonstrate use of the API, but any HTTP socket implementation may be used.

Use of this API is provided subject to the terms of the Doxie license agreement found in the Doxie desktop app. This API is provided as-is and is subject to change without notice.

Use of this API requires specific knowledge of HTTP/JSON APIs and requires a suitable environment for programmatically executing commands – this API is not intended for typical users of the Doxie Go Wi-Fi product. Doxie Customer Care provides limited email support for this API. Contact help@getdoxie.com for assistance.

Network Modes

By default Doxie Go Wi-Fi creates its own Wi-Fi network with a name such as "Doxie_042D6A". The last six characters of the Wi-Fi network name are based on the unique MAC address assigned to each unit. After joining your device to Doxie's network, Doxie's API will be accessible at 192.168.1.100 on port 8080.

Doxie Go Wi-Fi can also be configured with the Doxie desktop app to join an existing network. Typical configurations will have Doxie lease an IP via DHCP, but Doxie can also be configured to use a static IP. Devices joined to the same network as Doxie can access the API via the scanner's IP (pulled via DHCP or statically assigned) on port 8080.

Push the Wi-Fi button on the back of Doxie Go Wi-Fi to turn on Wi-Fi. When Doxie's light turns solid blue, the scanner's memory is now network accessible.

Auto Discovery

Doxie Go Wi-Fi supports the Simple Service Discovery Protocol (SSDP), which is a part of Universal Plug and Play (UPnP). Doxie periodically broadcasts its services via `NOTIFY` messages as well as responding to matching `M-SEARCH` search requests.

SSDP is supported in both networking modes. When using Doxie's direct network, it may be easier to attempt to communicate with the scanner at 192.168.1.100:8080 rather than relying on SSDP for discovery.

Authentication

By default Doxie Go Wi-Fi does not require authentication for API access, but a password can be configured by using the Doxie desktop app. If a password is set on the scanner, most API commands will require the password to be provided via HTTP basic access authentication with "doxie" as the username:

```
$ curl -D - -u doxie:MyPassword http://192.168.1.100:8080/scans/recent.json
HTTP/1.1 200 OK
Date: Thu, 01 Jan 1970 00:05:52 GMT
Content-Type: application/json
Content-Length: 35
Connection: close

{
  "path":"/DOXIE/JPEG/IMG_0001.JPG"
}
```

Omitting the authorization header or providing an incorrect password will return a `401 Unauthorized` error:

```
$ curl -D - http://192.168.1.100:8080/scans/recent.json
HTTP/1.1 401 Unauthorized
Date: Thu, 01 Jan 1970 00:05:58 GMT
Content-Length: 0
Connection: close
WWW-Authenticate: Basic realm="DoxieDX250 - Doxie_042D6A"
```

Reset Mechanism

When Doxie Go Wi-Fi is turned on, holding down the Wi-Fi button for 10 seconds will reset all Wi-Fi settings including the network configuration, scanner name, and scanner password. Resetting Doxie will not remove any existing scans.

General Commands

Scanner Status

`GET /hello.json` returns status information for the scanner, firmware, network mode, and password configuration. Accessing this command does not require a password if one has been set. The values returned depend on whether the scanner is creating its own network or joining an existing network.

```
$ curl http://192.168.1.100:8080/hello.json
{
  "model": "DX250",
  "name": "Doxie_042D6A",
  "firmwareWiFi": "1.29",
  "hasPassword": false,
  "MAC": "00:11:E5:04:2D:6A",
  "mode": "AP"
}
```

```
$ curl http://10.0.0.100:8080/hello.json
{
  "model": "DX250",
  "name": "Doxie_042D6A",
  "firmwareWiFi": "1.29",
  "hasPassword": false,
  "MAC": "00:11:E5:04:2D:6A",
  "mode": "Client",
  "network": "Apparent",
  "ip": "10.0.0.100"
}
```

- **model:** Always DX250.
- **name:** The name of the scanner, which defaults to the form "Doxie_XXXXXX". The name of a scanner can be changed by using the Doxie desktop app.
- **firmwareWiFi:** The Wi-Fi firmware version.
- **hasPassword:** Indicates whether a password has been set to authenticate API access. Passwords can be set and removed by using the Doxie desktop app.
- **MAC:** The MAC address of the scanner as shown on the scanner's bottom label.
- **mode:** "AP" if the scanner is creating its own network or "Client" if the scanner is joining an existing network.
- **network:** If the scanner is in "Client" mode, this is the name of the network it has joined.
- **ip:** If the scanner is in "Client" mode, this is the IP of the scanner on the network it has joined.

Scanner Status – Additional Information

`GET /hello_extra.json` returns additional status values. These values are accessed separately from those in `/hello.json` because there can be a delay of several seconds in loading them. Accessing this command does not require a password if one has been set.

```
$ curl http://192.168.1.100:8080/hello_extra.json
{
  "firmware": "0.26",
  "connectedToExternalPower": true
}
```

- **firmware:** The scanner firmware version.
- **connectedToExternalPower:** Indicates whether the scanner is connected to its AC adapter versus running on battery power. This value is not cached, so it immediately reflects any state changes.

Restart Wi-Fi System

`GET /restart.json` returns `204 No Content` and then restarts the scanner's Wi-Fi system. The scanner's status light blinks blue during the restart.

```
$ curl -D - http://192.168.1.100:8080/restart.json
HTTP/1.1 204 No Content
Date: Thu, 01 Jan 1970 00:03:54 GMT
Content-Length: 0
Connection: close
```

Scan Commands

List All Scans

`GET /scans.json` returns an array of all scans currently in the scanner's memory. After scanning a document, the scan will available via the API several second later.

```
$ curl http://192.168.1.100:8080/scans.json
[
  {
    "name":"/DOXIE/JPEG/IMG_0001.JPG",
    "size":241220,
    "modified":"2010-05-01 00:10:06"
  },
  {
    "name":"/DOXIE/JPEG/IMG_0002.JPG",
    "size":265085,
    "modified":"2010-05-01 00:09:26"
  },
  {
    "name":"/DOXIE/JPEG/IMG_0003.JPG",
    "size":273522,
    "modified":"2010-05-01 00:09:44"
  }
]
```

Calling this function immediately after scanning something may return a blank result, even if there are other scans on the scanner, due to the scanner's memory being in use. Consider retrying if a successful HTTP status code is returned along with a blank body.

Check for New Scans

`GET /scans/recent.json` returns the path to the last scan if available. Monitoring this value for changes provides a simple way to detect new scans without having to fetch the entire list of scans:

```
$ curl -D - http://192.168.1.100:8080/scans/recent.json
HTTP/1.1 200 OK
Date: Thu, 01 Jan 1970 00:04:26 GMT
Content-Type: application/json
Content-Length: 35
Connection: close

{
  "path":"/DOXIE/JPEG/IMG_0003.JPG"
}
```

If there is no recent scan available (e.g., the scanner was just turned on or there are no scans), this function will return `204 No Content`:

```
$ curl -D - http://192.168.1.100:8080/scans/recent.json
HTTP/1.1 204 No Content
Date: Thu, 01 Jan 1970 00:02:59 GMT
Content-Length: 0
```

```
Connection: close
```

Get Scan

`GET /scans/DOXIE/JPEG/IMG_XXXX.JPG` returns the scan at the specified path or `404 Not Found`. Use a path found via `/scans.json` or `/scans/recent.json` prepended with `/scans`.

```
$ curl -O http://192.168.1.100:8080/scans/DOXIE/JPEG/IMG_0001.JPG
```

Get Thumbnail

`GET /thumbnails/DOXIE/JPEG/IMG_XXXX.JPG` returns a thumbnail of the scan at the specified path or `404 Not Found`. Use a path found via `/scans.json` or `/scans/recent.json` prepended with `/thumbnails`.

```
curl -O http://192.168.1.100:8080/thumbnails/DOXIE/JPEG/IMG_0001.JPG
```

Thumbnails are constrained to fit within 240x240 pixels. Thumbnails for new scans are not generated until after the scan has been made available in `/scans.json` and `/scans/recent.json`. This function will return `404 Not Found` if the thumbnail has not yet been generated. Retrying after a delay is recommended to handle such cases.

Delete Scan

`DELETE /scans/DOXIE/JPEG/IMG_XXXX.JPG` deletes the scan at the specified path and returns `204 No Content` if successful or `403 Forbidden` on error.

```
$ curl -D - -X DELETE http://192.168.1.100:8080/scans/DOXIE/JPEG/IMG_0001.JPG
HTTP/1.1 204 No Content
Date: Thu, 01 Jan 1970 00:15:24 GMT
Content-Length: 0
Connection: close
```

```
$ curl -D - -X DELETE http://192.168.1.100:8080/scans/DOXIE/JPEG/IMG_9999.JPG
HTTP/1.1 403 Forbidden
Date: Thu, 01 Jan 1970 00:15:54 GMT
Content-Length: 0
Connection: close
```

Deleting takes several seconds because a lock on the internal storage must be obtained and released. Deleting may fail if the lock cannot be obtained (e.g., the scanner is busy), so consider retrying on failure conditions. When deleting multiple scans, use `/scans/delete.json` for best performance.

Delete Multiple Scans

`POST /scans/delete.json` deletes multiple scans in a single operation. This is much faster than deleting each scan individually. Multiple scans are referenced using a JSON array of paths:

```
[
  "/DOXIE/JPEG/IMG_0001.JPG",
  "/DOXIE/JPEG/IMG_0002.JPG",
  "/DOXIE/JPEG/IMG_0003.JPG"
]
```

The array of paths must be provided as the body of the POST request:

```
$ curl -D - -d '["/DOXIE/JPEG/IMG_0001.JPG","/DOXIE/JPEG/IMG_0002.JPG","/DOXIE/JPEG/
IMG_0003.JPG"]' http://192.168.1.100:8080/scans/delete.json
HTTP/1.1 204 No Content
Date: Thu, 01 Jan 1970 00:17:55 GMT
Content-Length: 0
Connection: close
```

This function returns `204 No Content` if successful or `403 Forbidden` on error.