Ready Label

RESTful API Version 1.0

May 27, 2016

Biosero

San Diego, CA

# Overview

This document covers the essential information to get working with the 1.0 version of the Ready Label Restful API as quickly as possible. This is a living document and will be updated in conjunction with the API. Versioning support will be added in a future update to the API.

## Getting Started

The API in Ready Label is turned off by default so the first thing you are going to want to do is turn it on. You can turn the API on in the program settings window at *Other > API*. Depending on the version of the program you are running it may ask for admin permissions. This is for a temporary registration process (not application wide) so go ahead and click yes to continue. Once the API has been turned on it will remain on until turned off even if the program is closed and reopened.

The API is a Restful service and the call/response format used in the http request body is JSON. The service is hosted internally by the application and as such the application must be running in order to use the API service. The root http address is http://<hostname>:5300/readylabel where the hostname is either the domain name or the IP address of the host computer. The currently available web requests are listing in the document below.

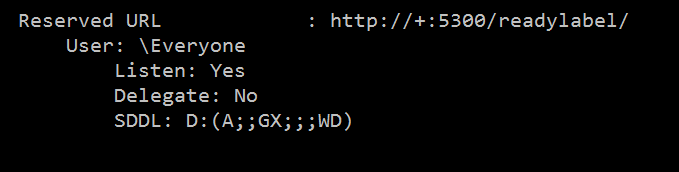
## Common Issues

### Http Registration

In order to pass communication via HTTP protocols Ready Label must be registered with the local IIS client. In most cases this will happen seamlessly during the installation process or when the API is manually turned on by the user. In some instances, however, both these mechanisms can fail especially in highly locked down environments. This will manifest itself as server connection issues when trying to access the API from anywhere but the localhost. To validate that the API http path has been registered properly run the following command from the command line:

**netsh htttp show urlacl**

If your program has been registered properly you should be able to find the following registration information in the command output:



If for some reason the registration doesn’t appear simply run the following command from the command line in administrative mode:

**netsh http add urlacl http://+:5300/readylabel/ sddl=D:(A;;GX;;;WD)**

### Firewalls

The Ready Label API listens on port 5300. In some cases, firewalls may block communication across this (or any) port. In this instance there are two ways to handle this issue. The first to turn off the firewall completely. The second is to add a rule to the firewall to allow communication across port 5300. There is currently no support for changing the port the API listens on but it will likely be added in a future update.

## Working with Label Data

One of the key aspects of Ready Label is its ability to handle data in a transparent way at runtime. This component has been kept consistent in the API. In the background, when a label is generated from a label template in Ready Label there is a search and replace algorithm that is run that replaces field data surrounded by squiggly brackets { } with matching values. In other words the program treats internal data like a dictionary of key-value pairs where fields in a template that are bounded by { and } respectively are treated as keys that are replaced with the corresponding data at runtime (irrespective of the data source that data comes from). So a sequence called “Sample” will replace any template field marked as {Sample} when a new label is generated.

From the perspective of the API this means that print commands (and any other command concerned with generating labels from template data) need a way to replace data in the corresponding template. This is handled in the JSON body of the request via the “data” field. This should look like this when sent via a http request:

"data"**:**

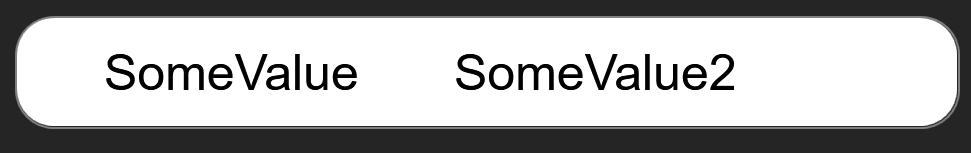
**{**

"Key"**:** "SomeValue",

"Key2"**:** "SomeValue2"

**}**

This data will replace any fields marked as {Key} or {Key2} with the respective data. In practicality this looks like the following template and result:



# API Requests

## GET /labels

Returns the collection of labels located in the *Home* folder.

### Resource Information

|  |  |
| --- | --- |
| Response Format | JSON |

### Paramaters

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Importance | Description | Example Value |
| fields | Optional | Limits the fields returned for each resulting object. Fields need to be specified as comma separated values | filePath,name |
| sort\_ascending | Optional | Sorts the objects returned. Values must be comma separating in and in order of sort importance.  **Note**: Field names are case-sensitive | name,totalFields |
| sort\_descending | Optional | Sorts the objects returned in descending order. Values must be comma separated and in order of sort importance.  **Note**: Field names are case-sensitive | name,totalFields |

### Example Request

#### URL

<http://[ip]:[port]/readylabel/labels>

#### Body

*Empty*

### Example Result

**[**

**{**

"filePath"**:** "C:\\Users\\<username>\\Documents\\Ready Label\\API TEST 3.label"**,**

"templateName"**:** "API TEST 3"**,**

"isOpen"**:** false**,**

"width"**:** 24.0**,**

"height"**:** 9.0**,**

"xOffset"**:** 0.0**,**

"yOffset"**:** 0.0**,**

"totalElements"**:** 3

**},**

**{**

"filePath"**:** "C:\\Users\\<username>\\Documents\\Ready Label\\API TEST.label"**,**

"templateName"**:** "API TEST"**,**

"isOpen"**:** false**,**

"width"**:** 24.0**,**

"height"**:** 9.0**,**

"xOffset"**:** 0.0**,**

"yOffset"**:** 0.0**,**

"totalElements"**:** 3

**},**

**{**

"filePath"**:** "C:\\Users\\<username>\\Documents\\Ready Label\\EGecko Test.label"**,**

"templateName"**:** "EGecko Test"**,**

"isOpen"**:** false**,**

"width"**:** 50.0**,**

"height"**:** 6.0**,**

"xOffset"**:** 0.0**,**

"yOffset"**:** 0.0**,**

"totalElements"**:** 1

**}**

**]**

## GET /printers/<id>

Returns the configuration of the printer with the specified ID.

Please note that the responses returned from each printer will vary based on the configuration capabilities of each printer.

### Resource Information

|  |  |
| --- | --- |
| Response Format | JSON |

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Importance | Description | Example Value |
| id | Required | The printer name.  **Note**: Currently accepted values are egecko, gecko, & vcode | egecko |
| fields | Optional | Limits the fields returned for each resulting object. Fields need to be specified as comma separated values | applynorth,applysouth,  \*Known bug: this param may not work with some printers |

### Example Request

#### URL

<http://[ip]:[port]/readylabel/printers/egecko>

#### Body

*Empty*

### Example Result

**{**

"pickupHeight"**:** 0.0**,**

"applyHeight"**:** **-**1.0**,**

"applyDepth"**:** 0.0**,**

"applyNorth"**:** true**,**

"applySouth"**:** true**,**

"applyEast"**:** true**,**

"applyWest"**:** true

**}**

## GET /printers/<id>/status

Returns the current status of the printer with the specified ID. Note that this is used to determine if the printer is doing something. As such in an automated scenario this command should be used in prior to sending new print commands to determine if the printer is already doing something.

### Resource Information

|  |  |
| --- | --- |
| Response Format | JSON |

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Importance | Description | Example Value |
| id | Required | The printer name.  **Note**: Currently accepted values are egecko, gecko, & vcode | egecko |
| fields | Optional | Limits the fields returned for each resulting object. Fields need to be specified as comma separated values | accessmode,state |

### Example Request

#### URL

<http://[ip]:[port]/readylabel/printer/egecko/status>

#### Body

*Empty*

### Example Result

{

"accessMode": "Local",

"state": "Ready"

}

## POST /printers/<id>

Updates the configuration of the printer specified with the id.

Please note that the configuration object for each printer is different based on their properties and capabilities. It is recommended you retrieve the config object (see above) for the printer you are working with to design your POST object.

### Resource Information

|  |  |
| --- | --- |
| Response Format | JSON |

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Importance | Description | Example Value |
| id | Required | The printer name.  **Note**: Currently accepted values are egecko, gecko, & vcode | egecko |

### Example Request

#### URL

<http://[ip]:[port]/readylabel/printers/egecko/status>

#### Body

**{**

"pickupHeight"**:** 0.0**,**

"applyHeight"**:** 4.0**,**

"applyDepth"**:** 0.0**,**

"applyNorth"**:** true**,**

"applySouth"**:** true**,**

"applyEast"**:** false**,**

"applyWest"**:** false

**}**

### Example Result

**{**

"pickupHeight"**:** 0.0**,**

"applyHeight"**:** 4.0**,**

"applyDepth"**:** 0.0**,**

"applyNorth"**:** true**,**

"applySouth"**:** true**,**

"applyEast"**:** false**,**

"applyWest"**:** false

**}**

## POST /printers/<id>/command

### Resource Information

|  |  |
| --- | --- |
| Response Format | JSON |

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Importance | Description | Example Value |
| id | Required | The printer name.  **Note**: Currently accepted values are egecko, gecko, & vcode | egecko |
| name | Required | The name of the command | print |

### Example Request

#### URL

<http://[ip]:[port]/readylabel/printers/egecko/command?name=print>

#### Body

See *Printer Commands* section for details.

### Example Result

See *Printer Commands* section for details.

# Printer Commands

## EGecko

### Initialize

#### URL

<http://[ip]:[port]/printers/egecko/command?name=>intialize

#### Body

*Empty*

#### Example Result

Status Code 200

Body: *Empty*

### Print and Apply

#### URL

<http://[ip]:[port]/printers/egecko/command?name=>printandapply

#### Body

**{**

"labelFile"**:** "C:\Users\<username>\Documents\Ready Label\API TEST.label"**,**

"data"**:**

**{**

"Key"**:** "SomeValue"

**}**

**}**

\*Note: The label file path is the path from the root drive of the machine Ready Label is installed.

\*Known Bug: The data section must have a value even if that value is not used internally in Ready Label. We recommend always at least using the above key/value pair to prevent this from occurring.

#### Example Result

Status Code 200

Body: *Empty*

### Rotate Stage

#### URL

<http://[ip]:[port]/printers/egecko/command?name=>rotate

#### Body

**{**

"rotation"**:** 90**,**

"movementMode"**:** "absolute" //absolute or relative

**}**

#### Example Result

Status Code 200

Body: *Empty*