REST API of REXYGEN

Reference Manual

REX Controls s.r.o.

Version 2.50.10
Plzeň (Pilsen), Czech Republic
2020-09-03

Contents

Chapter 1

Introduction

Starting in version 2.50, the REXYGEN system provides a simple communication interface utilizing HTTP protocol and other Web technologies that is called REST API. The REST API is integrated directly into REXYGEN and no special applications or tools are necessary for putting it into operation. The REST API makes it possible to command the REXYGEN system from third party tools that support HTTP protocol or even directly from Web browser without the need for any dedicated tool.

The API utilizes a simple lightweight built-in Web server, that has also been integrated into REXYGEN starting from version 2.11. The Web server runs HTTP protocol on port 8008 and HTTPS protocol (HTTP over SSL) on port 8009 in default configuration. Both ports may be configured and the Web server can by enabled or disabled in configuration. For detailed information about the configuration see [1].

Chapter 2

Common principles

The REST API has been developed with awareness of following key principles: performance, simplicity, browseability and security. Those principles form the final structure and mechanism of the API.

The performance principle is assured by a highly integrated and embedded Web server without any auxiliary framework between the server and the core of the REXYGEN system. Although integrated directly into the core, the Web server is separated from the core by a thin interface so that possibility of interference with the real-time execution core is minimized and eventual separation of Web server into a stand-alone application is possible. The Web server runs in a dedicated thread with adjustable priority utilizing asynchronous socket communication.

From the structure point of view, the performance is achieved by distinguishing between *data* and *meta-data*, so that re-sending of unnecessary or already transmitted data is minimized. A short and a long format of response is distinguished, so in cases where the short format is sufficient, the amount of data transmitted may be reduced. A user should use the JSON data format in requests to gain maximal performance. The JSON data format is currently the only format supported within write (POST) requests, although the data may be retrieved from the Web server in HTML, XML and plain text format.

The simplicity and browseability principles are reflected in the structure of the API. The structure is described in chapter 4. The API reflects as much as possible an already defined structure and links of a real-time control algorithm. Almost all rules and convections that the user already knows from using the REXYGEN system, REXYGEN Studio or REXYGEN Diagnostics may be applied. The browseability makes it easy for any user to start using the REST API of the REXYGEN system without the need for detailed study of user manual and without the need for specialized tools. Anyone with basic knowledge of HTTP protocol and modern Web browser can start using REST API of REXYGEN. Every single resource provided by the interface can be found, listed and inspected using a standard modern Web browser.

The Web server has been also designed with a respect to the safety and security. Although the Web server is an integral part of the control system, the thin interface

between the core and Web server minimizes the possibility of interference. All data is checked for consistency, all memory is zeroed and memory allocations are minimized. The security is ensured by utilizing standard HTTPS protocol (HTTP over SSL) and by providing (yet limited) basic user authentication.

Chapter 3

Location

The REST API is located on a URL in format `http://host:port/api` when a HTTP web server is enabled and on a URL `https://host:port/api` when a HTTPS web server is enabled. See [1] for more information about configuration of the server. The default port is 8008 for HTTP an 8009 for HTTPS connections.

A used may immediately start using and interacting with the REST API by typing the proper URL into a web browser.

Chapter 4

Structure of resources

All records available within the REST API are located into a resource tree as a resources. Resources are identified by their path or rather by a URL. Resources may be read or updated by standard HTTP requests. Resources are available by a GET request in HTML format and so they may be directly viewed and browsed by a standard web browser. Every resource may carry any of following types of information:

- **data** – represents an actual *value*, *quality* and *time-stamp* of a configuration parameter or a signal which may be input, output, parameter, etc.,
- **meta-data** – represents information about properties of a *data*, which may be a type, range, etc.,
- **children list** – represents information about another resources that belongs to the resource as children resources, the *children list* is a part of *meta-data*.

4.1 Resource tree

A resource tree is located on `http://host/api`. The first level represents the target device and provides no data. It has following children resources:

- **modules** – provides list of modules loaded by a target executive,
- **archives** – provides list of archives used by a target executive,
- **drivers** – provides list of archives used by a target executive,
- **levels** – provides list of priority levels in a target executive,
- **tasks** – provides list of tasks in a target executive,
- **data** – provides an access to the signals in a target executive in a target-specific namespace,
- **system** – provides information about a target device,
- **info** – provides information about a target executive.

4.1.1 modules

The *modules* resource provides only list of loaded modules. No other information is available in current version.

4.1.2 archives

The *archives* resource provides only list of archives which are user by a target executive. No other information is available in current version.

4.1.3 drivers

The *drivers* resource provides only list of drivers which are user by a target executive. No other information is available in current version.

4.1.4 levels

The *levels* resource provides only list of priority levels defined in a target executive. No other information is available in current version.

4.1.5 tasks

The *tasks* resource provides only list of tasks in a target executive. Each task is a parent resource with its function blocks as a children resources. Each subsystem resource is also a parent resource with its function blocks as children.

4.1.6 data

The *data* resource provides an access to all resources in a target-specific namespace, ie. in a naming convention that is specific to the target. A standard REXYGEN signal naming convention is used in a case of the REXYGEN system.

4.1.7 system

The *system* resource provides miscellaneous information about the target device including software and hardware version and platform identification.

4.1.8 info

The *info* resource provides information about target executive including name and description, a name of author and company and several checksums and ids that uniquely identify the target executive.

Chapter 5

Requests

This chapter describes HTTP requests supported by the REST API.

5.1 Request Types

In the current version, GET and POST requests are supported. The GET request is used for reading of resources. It has no impact on the state of the target device or on the running executive. The same value is returned if multiple GET requests are processed while no state change in the target device or executive is performed. The POST request is used for setting value of a resource. Only **data** may be set by the POST request. It is not possible to change **meta-data** anyhow. The modified resource must be modifiable and a user has to be authorized and have proper permissions while modifying a resource by the POST request.

5.2 Supported Data Formats (MIME types)

The REST API supports MIME types that are listed in following paragraph. A supported MIME type may be specified in the `Accept` header field of a HTTP request.

- `text/html` – HTML,
- `application/json` – JSON,
- `text/xml` – XML
- `text/plain` – plain text,
- `text/csv` – comma-separated list of values.

Currently, only `application/json` and `application/x-www-form-urlencoded` formats are supported in POST requests. Those formats may be specified in the `Content-Type` header field.

Chapter 6

Queries

This chapter contains information about queries in the REST API.

6.1 About Queries

A query string is an additional information appended to an URL that may affect a type and a format of data returned and how the request is processed on the server. A URL with a query string has following format:

```
http://host/api/resource?query
```

A query may be a parameter with a value. A query with a parameter and value has following format:

```
http://host/api/resource?parameter=value
```

Multiple parameters may be present. A query with multiple parameters has following format:

```
http://host/api/resource?parameter1=value1&parameter2=value2
```

6.2 Supported Queries

This section contains information about queries that are supported in the REST API.

6.2.1 data

The `data` query has no value. It specifies, that *data* information of a resource specified should be returned.

6.2.2 meta

The `meta` query has no value. It specifies, that *meta-data* information of a resource specified should be returned.

6.2.3 show

The `show` query has no value. It specifies that the data of a resource should be shown in a user-friendly format. This query is currently only supported on `TRND` function blocks where the `show` query causes that a graphical representation of data is returned.

6.2.4 export

The `export` query has no value. It specifies that the data of a resource should be returned in a format that is suitable for subsequent processing. This query is currently only supported on `TRND` function blocks where the `export` query causes that all the data from buffer are returned in a selected file format.

6.2.5 override

The `override` query has no value. It causes that the signal being written is switched to a *Local override* ie. *manual* mode. It may be applied to inputs of function block to put an input signal into fixed state. The `override` query is available only in a POST request.

6.2.6 format

The `format` query has a value which may be `long` or `short`. It defines the detail of information being returned. In case of a signal, only actual value is returned if `format=short` but signal quality and time-stamp are also included if `format=long`.

6.2.7 robot

The `robot` query has no value. The query notifies the server that the client is not a user. With this query specified, the server sends no responses that need interaction with a user and basic authentication is assumed. This query should be always specified when using `wget` of `curl` for interaction with the REST API.

6.2.8 mime

The `mime` query has a value that specifies the MIME format of data returned from the server. See section 5.2 for supported MIME types. The MIME specified by a `mime` query has always a precedence over the `Accept` HTTP header field.

Chapter 7

Authentication

An authentication is also supported by the integrated Web server of the REXYGEN system. See [1] for more information on how to secure the target device and enable user authentication.

Two distinct mechanisms are available for user authentication:

- **HTTP Cookie** – this mechanism is used by the integrated HTML web interface,
- **HTTP Basic Authentication** – this mechanism should be used when interacting with the REST API programatically – see section 6.2.7. A URL with a user authentication has a following format:

  ```
  http://user:password@host/api/resource
  ```

  Authentication mechanisms should not be mixed together. A request fails if both types of authentication are used and any of them is incorrect.

Bibliography

[1] REX Controls s.r.o.. *RexCore – User manual*, 2020. →.

Documentation reference number: 12481