ionCube24 User API

Document version 0.1

ionCube Ltd.

19 Jan 2017.

Introduction

The ionCube24 REST API provides users a means to automate their interactions with the web service. The purpose of this document is to describe how to use the API, the actions available in the API and how to best interpret unexpected responses.

Reporting Bugs, Feature Requests and Discussion

The ionCube24 section of the ionCube Helpdesk at https://support.ioncube.com should be used to report bugs, raise feature requests and for general discussion related to ionCube24 so that these topics can be tracked.

API Description

API Keys

API operations require a key (see *Request Authorisation* below). Keys may be issued and managed via the ionCube24 web interface.

HTTP Requests

The API uses `GET` and `POST` operations as appropriate to the request.

HTTP Status codes

HTTP status codes indicate the outcome of a request. The use of codes is described below, and in general they relate to the processing of the request itself and not the application state. As such, a valid request may produce a `200 OK` response even though the operation has failed at the application level, whereas an operation that could not be attempted would return a different response. Application level errors are indicated in a request's response body. See the Appendix for more information on the possible causes of the following error codes.

Interpretation of HTTP codes

Code	**Used when**
`200`	The request was accepted for processing. An error property will be present in the response if an error occurred.
`400`	The request was recognised as a potential request, but was invalid in some respects. The reason is indicated in the error property of the response.
`401`	The API key was missing or did not match the key of an account. The response will include the header `WWW-Authenticate: API`
`403`	The requested operation was not allowed.
`404`	The endpoint or action does not exist or is unavailable for the API key.
`415`	The API payload was invalid JSON.
`500`	A server side failure prevented processing of the

	request. Further information may be available in the response error property.

Error response

A successful operation is indicated by the lack of any error in the response. A failed operation would return a response with an `errors` key whose value is an array of one or more objects with the following fields.

Field	**Meaning**	**Type**
`code`	Error code.	`int`
`reason`	Error reason.	`string`

Optional fields

Field	**Purpose**	**Type**
`reason_extra`	Additional detail for the error.	`string`

Example:

```
{
      "errors": [{
            "code": 20002,
            "reason": "Password did not match criteria: Your password is
too short (minimum 6 characters)"
      }, ...]
}
```

Error codes

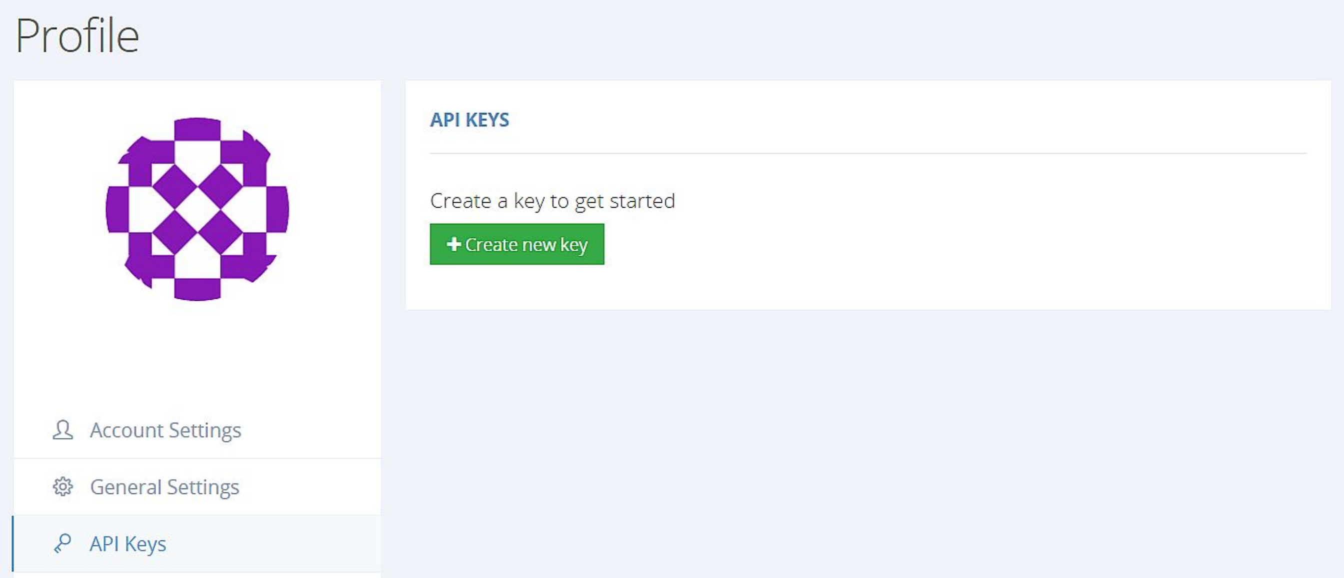
Please see the Appendix section at the end of this document.

Request Authorisation

API key

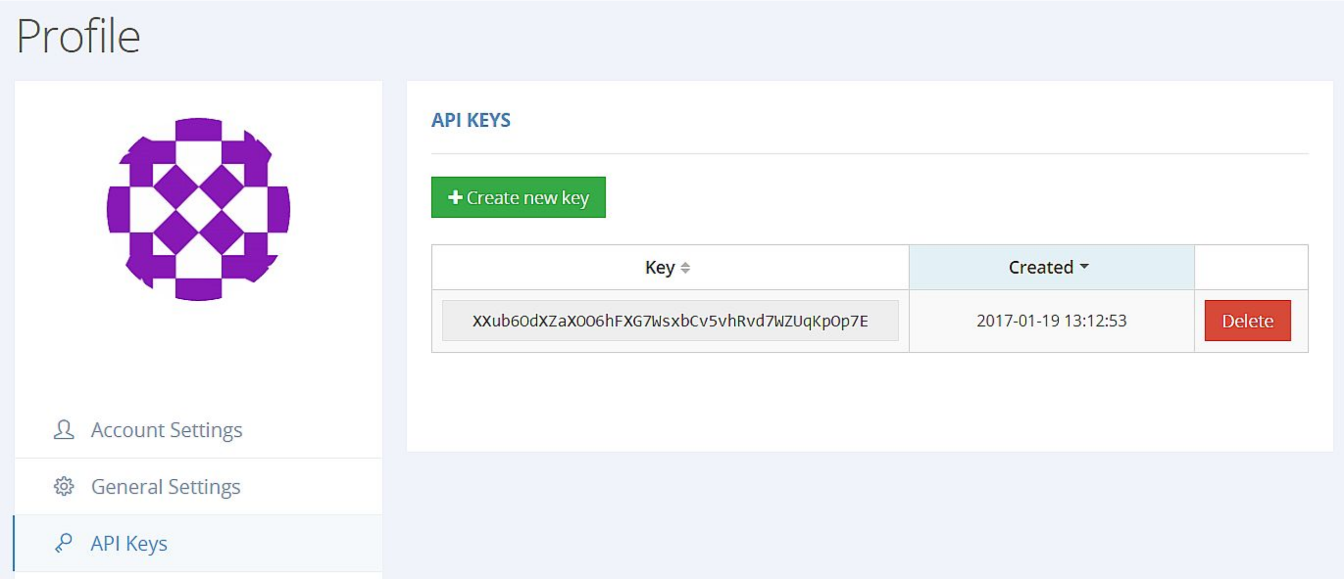
An API key is required for all requests, passed via an `Access-Token` header. API keys can be generated and managed by logged-in users at https://ioncube24.com/#/userprofile/api

The API Keys section of the interface should appear as follows:



Create a key to get started

After selecting “Create new key”, the key should immediately appear in the interface:



Key ⇕	Created ▾	
XXub6OdXZaXOO6hFXG7WsxbCv5vhRvd7WZUqKpOp7E	2017-01-19 13:12:53	Delete

Required header fields

Header Field	Purpose	Example
`Authorization`	API key.	`API key="pjshdTyd…"`

Authorisation performs the following tests:

- *Authorisation header present*
  If the header is missing, a `401` status is returned.

- *Valid header*
  If the authorisation header is invalid, a `400` status is returned.

- *Valid and active key*
  If the API is invalid, a `400` status is returned.

- *Endpoint authorisation*
  An API key is authorised for specific endpoints. If an API key does not authorise the requested resource, but the root of a requested endpoint does match the root of at least one authorised endpoint, a `403` will be returned. If the root of a requested endpoint is not permissioned for any resources, a `404` will be returned.

  Examples for request `/api/v1/abc` where only `/abc` is authorised.

API Key Allows	**HTTP Status**
`bananas`	`404`
`/api/v1/xyz`	`403`
`/api/v1/abc`	`200`

API Operations

Fetch Trustpoint

```
GET /api/v1/domains/<domain name>/trustpoint
```

Retrieve the trustpoint of the specified domain as a UNIX timestamp.

Required query parameters

None.

Optional query parameters

Field	**Purpose**	**Type**
`server_name`	Server name. Will be used to validate that the domain name is related.	`string`
`timeout`	Request timeout in seconds. Default is 10 seconds. Min 1 second, max 60 seconds.	`int`

Response fields

```
{
      "use_trustpoint: true, // bool. True if trustpoint is enabled
      "trustpoint" : 1484840815 // Trustpoint as a UNIX timestamp
}
```

Update Trustpoint

```
POST /api/v1/domains/<domain name>/trustpoint
```

Update the trustpoint for the specified domain.

Required fields

Field	Purpose	Type
`trustpoint`	The server trustpoint as an integer UNIX timestamp. Can not be set more than one week into the future or the past. If the trustpoint has previously been disabled via the web interface, the trustpoint will be re-enabled. Either `trustpoint` or `trustpoint_delta` must be specified.	`int`
`trustpoint_delta`	The number of minutes into the future from the current time that the trustpoint will be set. Min 0, max 10080 (one week). If the trustpoint has previously been disabled via the web interface, the trustpoint will be re-enabled. Either `trustpoint` or `trustpoint_delta` must be specified.	`int`

Optional fields

Field	Purpose	Type
`server_name`	Server name. Will be used to validate that the domain name is related.	`string`
`timeout`	Request timeout in seconds. Default is 10 seconds. Min 1 second, max 60 seconds.	`int`

Response fields

None

Appendix

HTTP Error Codes - Details

Code	Name	Message	Custom Status
200	MODEL_ERROR	Invalid, non-unique or missing parameters	1
	UNFULFILLED_PASSWORD_CRITERIA	Password did not match criteria	2
	UNFULFILLED_MODEL_CRITERIA	Invalid parameters	3
	INVALID_RESOURCE	Requested resource does not exist	4
	TRUSTPOINT_FETCH_FAILED	Fetching the trustpoint failed	5
	TRUSTPOINT_UPDATE_FAILED	Updating the trustpoint failed	6
	UNKNOWN_ERROR	An unknown error occured	7
	RESOURCE_CONFLICT	Conflicting resource identifiers	8
	RESOURCE_AMBIGUOUS	The specified resource is ambiguous	9

400	INVALID_TOKEN	Permission denied, recheck access key	1
	MISSING_PAYLOAD_PARAMETER	Could not process request due to empty, invalid or missing parameters	2
401	PERMISSION_DENIED	Permission denied	1
	MISSING_HEADER	Permission denied, expected Authorization header	2
	EMPTY_HEADER	Permission denied, expected non-empty Authorization header	3
	INVALID_HEADER	Permission denied, recheck Authorization header format	4
	NON_EXISTENT_TOKEN	Permission denied, access key does not exist (anymore)	5
403	OPERATION_NOT_ALLOWED	Operation is not allowed	1
	ACCESS_DENIED	Permission to resource with this ID was denied	2
	INVALID_INI_TOKEN	Permission denied, please generate a token first	3

	INVALID_SCRIPT_TOKEN	Permission denied, please generate a token first	4
405	UNSUPPORTED_POST_ID	Specifying ID in POST request is not supported	1
	MISSING_URL_PARAMETER	Could not process request, please specify ID in url	2
415	INVALID_JSON	Invalid JSON Format	1
500	INTERNAL_ERROR	Internal server error	1
	NOT_YET_IMPLEMENTED	Parts of this request are not yet implemented	2
	REMOTE_SERVER_ERROR	The request could not be completed due to problems contacting the specified remote server	3
501	UNSUPPORTED_REQUEST_TYPE	Unsupported request type	1

Document Change History

19 Jan 2017	0.1	Initial version.