Mashery I/O Docs

Configuration Guide

March 2014

Revised: 3/17/2014

www.mashery.com

Mashery, Inc.
717 Market Street, Suite 300
San Francisco, CA 94103

Part Number: IO-Docs-cfg-04-2012

Contents

Chapter 1.
About this Guide

Introduction

This guide describes how to configure Mashery I/O Docs using the Administration Dashboard. Mashery I/O Docs is an interactive API exploration and testing tool that runs on your Developer Portal. It enables your developer partners to perform API calls from within the documentation with their own API keys with described form-based parameter inputs fields and easy-to-read color-coded and formatted payload outputs.

For some APIs, I/O Docs may serve as a replacement for traditional long-form documentation. However, it can also serve as a useful compliment to regular documentation, especially in cases of APIs that have more advanced authentication and request methods.

Assumptions

This guide assumes that you have Administration Dashboard access and proper role privileges to configure content on the Developer Portal.

Chapter Overview

The Mashery I/O Docs Configuration Guide is divided into the following chapters:

- **Chapter 2: Prerequisites and Enablement Overview** - Provides you with a quick overview of the requirements and the process to activate I/O Docs on your Developer Portal via the Administration Dashboard.
- **Chapter 3: Top-Level Configuration Overview** - A high-level view of the global Developer Portal configuration settings and I/O Docs Definitions
- **Chapter 4: I/O Docs Definition JSON Schema** - A deep dive into the description schema that describes your resources, methods and parameters. This schema definition is used to render the interactive docs.

Chapter 2. Prerequisites and Enablement Overview

Prerequisites

I/O Docs is available on all default Mashery deployments, meaning that options to enable and configure I/O Docs are available in the Administrative Dashboard. If for any reason the options to enable and configure I/O Docs are not visible, please contact your Client Service Manager for more information.

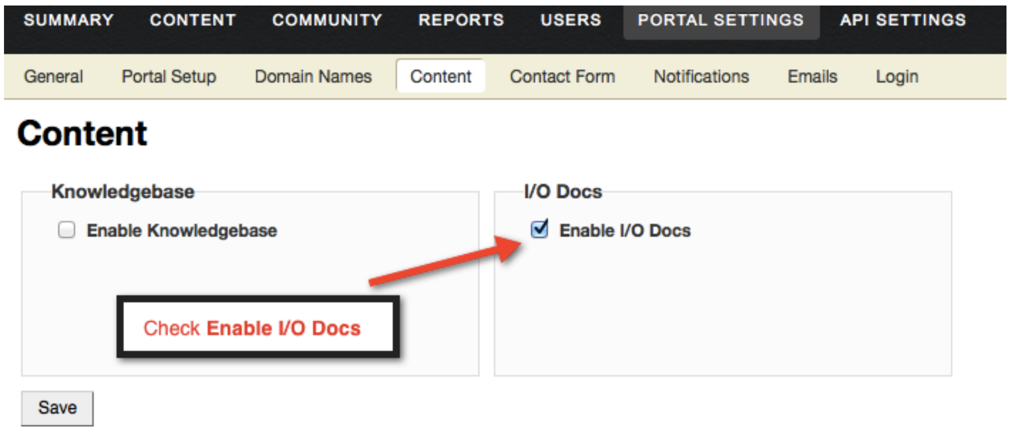
RESTful API Support

Currently, I/O Docs only supports RESTful APIs, using the GET, POST, DELETE and PUT methods. SOAP APIs are not supported because the resource, methods and parameters do not construct a SOAP request.

Enablement Overview

To enable I/O Docs:

1. Access to the Mashery Admin Dashboard.
2. Enable the I/O Docs Content Module by navigating to **Portal Settings** > **Content** and checking the box adjacent to **Enable I/O Docs:**



3. Click **Save**.

After completing these steps, I/O Docs is enabled. The URL for I/O Docs on your portal is: **http://*devhost.devdomain.com*/io-docs**, for example, if your portal URL is **http://developer.acme.com**, then the URL for I/O Docs would be **http://developer.acme.com/io-docs**. To configure I/O Docs properly for your APIs, please proceed to Chapter 3.

Chapter 3. Top-Level Configuration Overview

I/O Docs Definitions Overview

Definitions for I/O Docs are used to generate the rendered I/O Docs interface, including the groups of resources, methods and parameters. Definitions are associated with APIs configured in the API Settings section of the Administration Dashboard. An API is allowed only one I/O Docs definition.

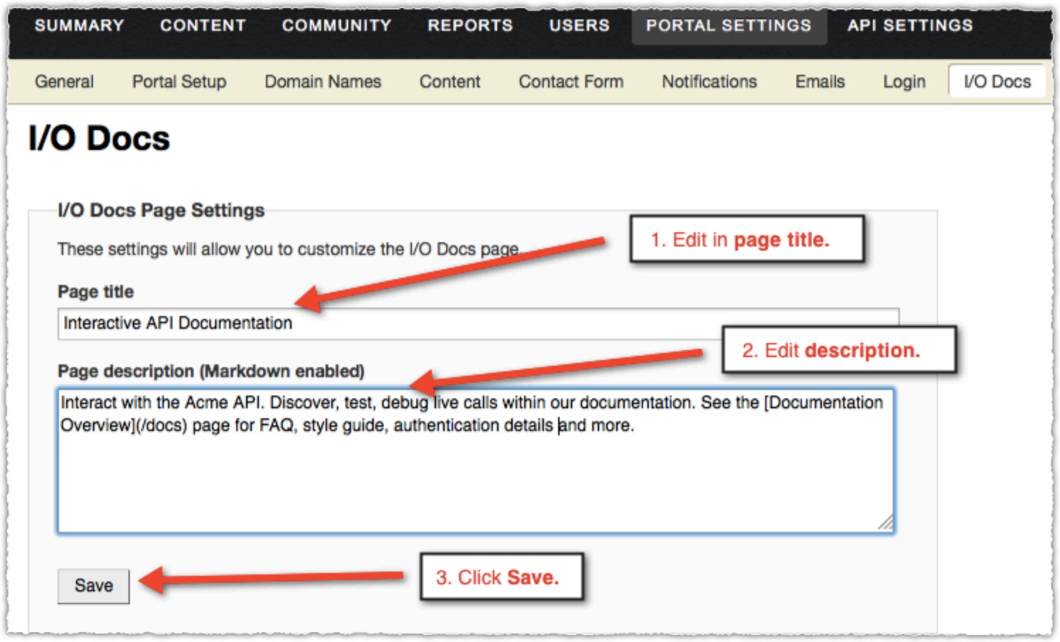
Note that although an I/O Docs definition is associated with an API, it is functionally separate. This means that you can freely configure your I/O Docs definition to highlight only certain resources, methods or parameters of your choice.

The definitions are simple JSON objects modeled after the Google Discovery Document format (GDDF). The schema format has been extended to address authentication and signature methods. More detailed information about the schema can be found in Chapter 4.

Global I/O Docs Page Settings

You can provide a page title header and description that appears at the top of every rendered I/O Docs page. These attributes can be set by following these steps:

1. Access the Mashery Admin Dashboard
2. Add page title and description by navigating to the **Portal Settings** > **I/O Docs** section, and editing these fields in the I/O Docs Page Settings. Click **Save**.



I/O Docs

I/O Docs Page Settings

These settings will allow you to customize the I/O Docs page.

Page title

Interactive API Documentation

Page description (Markdown enabled)

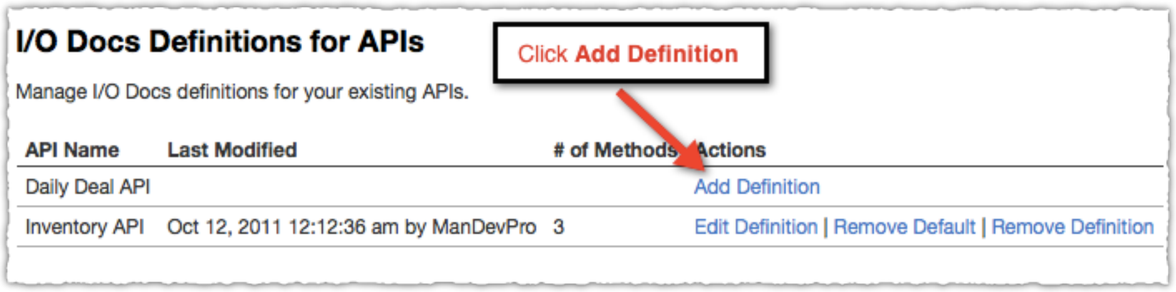
Interact with the Acme API. Discover, test, debug live calls within our documentation. See the [Documentation Overview](/docs) page for FAQ, style guide, authentication details and more.

Adding I/O Docs Definitions

If you have enabled I/O Docs by following directions in Chapter 2, the next step is to add a definition to one or more APIs.

To add I/O Docs definitions:

1. Access the Mashery Admin Dashboard
2. Add a definition to an API by navigating to the **Portal Settings** > **I/O Docs** and clicking **Add Definition** adjacent to the API Name that you wish to configure:

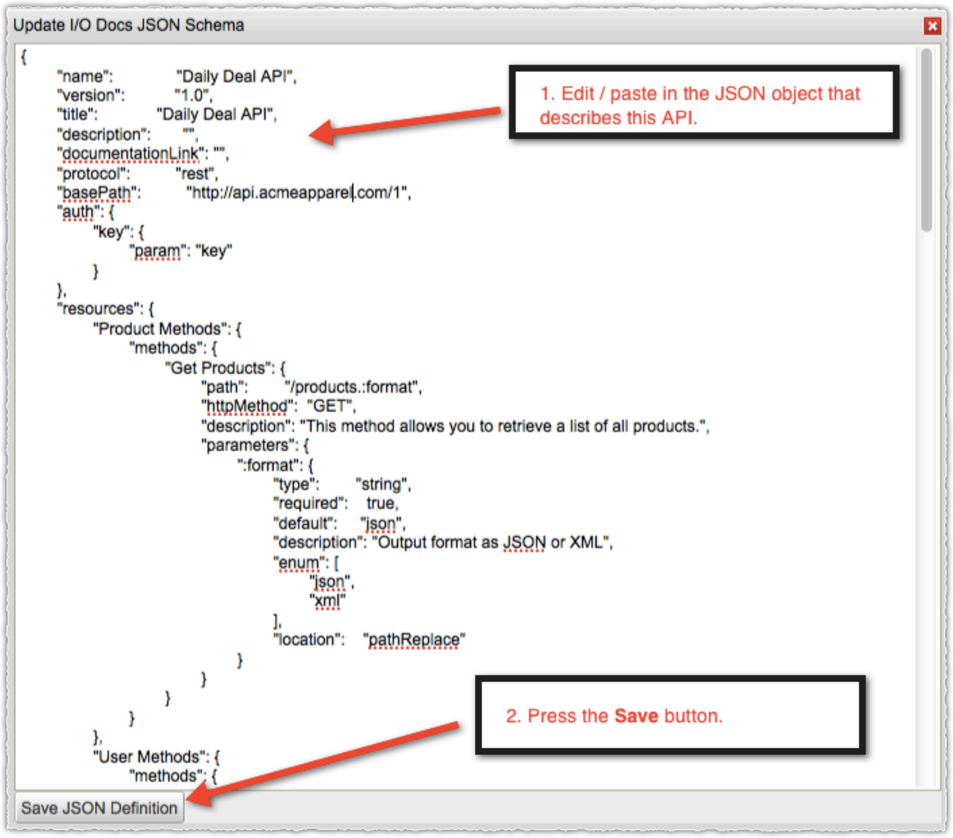


I/O Docs Definitions for APIs

Manage I/O Docs definitions for your existing APIs.

API Name	Last Modified	# of Methods	Actions
Daily Deal API			Add Definition
Inventory API	Oct 12, 2011 12:12:36 am by ManDevPro	3	Edit Definition | Remove Default | Remove Definition

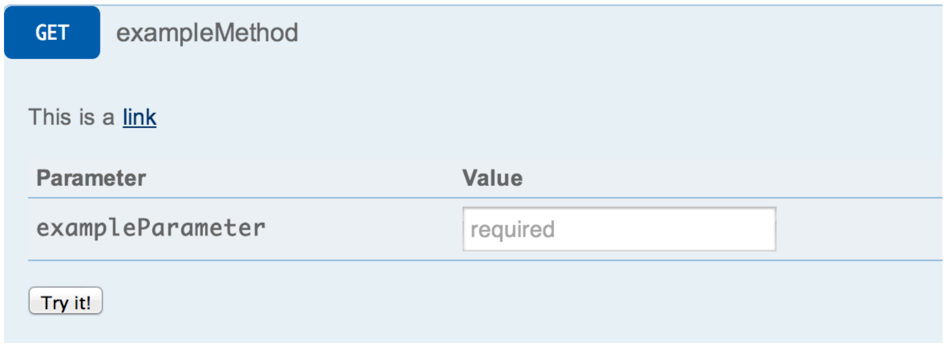
3. Provide the JSON schema that describes the entire API's set of resources, methods, and parameters that you wish to expose via I/O Docs. The JSON format details can be found in Chapter 4. Click **Save JSON Definition**:


Update I/O Docs JSON Schema
1. Edit / paste in the JSON object that describes this API.
2. Press the **Save** button.
Save JSON Definition

```
{
    "name":         "Daily Deal API",
    "version":      "1.0",
    "title":        "Daily Deal API",
    "description":     "",
    "documentationLink": "",
    "protocol":       "rest",
    "basePath":       "http://api.acmeapparel.com/1",
    "auth": {
        "key": {
            "param": "key"
        }
    },
    "resources": {
        "Product Methods": {
            "methods": {
                "Get Products": {
                    "path":       "/products.:format",
                    "httpMethod":  "GET",
                    "description": "This method allows you to retrieve a list of all products.",
                    "parameters": {
                        ":format": {
                            "type":       "string",
                            "required":   true,
                            "default":    "json",
                            "description": "Output format as JSON or XML",
                            "enum": [
                                "json",
                                "xml"
                            ],
                            "location":   "pathReplace"
                        }
                    }
                }
            }
        },
        "User Methods": {
            "methods": {
```

I/O Docs supports the use of **markdown** in API, reasource, and method level descriptions. For example, the **link markdown** in the method description renders as a link in I/O Docs :

```
…
"exampleMethod":{
        "description":"This is a [link](http://mashery.com)”,
        "httpMethod":"GET",
        "path":"/methodpath", …
```



This is a link

Parameter	Value
exampleParameter	required

Chapter 4. I/O Docs Definition Schema Details

JSON Schema Overview

The schema is a JSON object containing top-level API properties along with resources, methods and parameters that describe how API requests are formed and transmitted. The Mashery I/O Docs Definition Schema is based in part on the Google Discovery Document format. Some properties within the GDDF are disregarded by the I/O Docs schema processor and will not be referenced below or in the examples. The GDDF has been extended to address authentication and signature methods.

JSON Object Property Details

```
{
  "name": "value",
  "version": "value",
  "title": "value",
  "description": "value",
  "protocol": "rest",
  "basePath": "value",
  "auth": {
    "key": {
      "param": "value",
      "location": "value",
      "secret": {
        "param": "value",
        "type": "value"
      }
    },

    "basicAuth": "value",

    "oauth": {
      "version": "value",
      "base_uri": "value",
      "authorize_uri": "value",
      "access_token_uri": "value",
      "auth_flows": [
         "value"
      ],
      "options": { }
    }
  },
  "resources": {
    "value": {
      "methods": {
        "value": {
          "path": "value",
          "httpMethod": "value",
          "description": "value",
          "parameters": {
            "value": {
              "description": "value",
              "default": "value",
              "required": "value",
              "enum": [
                "value"
              ],
              "enumDescriptions": [
                "value"
              ],
              "location": "value"
            }
          }
        }
      }
    }
  }
}
```

I/O Docs Definition

The following table describes the structure of an I/O docs JSON definition. Note that this structure is largely based on the Google Discovery Document format.

Property Name	**Type**	**Require d**	**Description**
name	string	yes	The name of the API. This is not shown/displayed anywhere in the rendered page.
version	string	yes	The version of the API, e.g. "v1.0".

Property Name	Type	Require d	Description
title	string	yes	The title of the API that is displayed in the API selection drop-down menu, e.g. "Daily Deals API".
description	string	yes	The description of the API. This is displayed directly under the API selection drop-down menu when the corresponding API is selected, e.g. "The Daily Deals API provides a real-time look into the current daily deals based on ZIP code and radius". This value supports Markdown syntax.
protocol	string	yes	The protocol described by this document, e.g. "rest".
basePath	string	yes	The base URL path for REST requests, including scheme name. Port number is optional. e.g. "https://api.acme.mashery.com:443"
auth	object	no	Links to key and/or OAuth objects.
auth.key	object	no	Links to param and secret objects.

Property Name	Type	Require d	Description
auth.key.location	string	no	Location of API key. Default location is the query string for GET requests, and encoded parameter request body for POST requests. Also determines location of signature key/value pair. To override default location, valid values are "query", "pathReplace", "body" and "header". Additionally, you can supply one or more of the above values separated by a comma. (e.g. "query,header" would place the API key parameter in both the query string and as a request header value pair).
auth.key.param	string	no	Name of the API key parameter, e.g. "api_key" or "key". Defaults to "apiKey".
auth.key.secret	object	no	Links to the param and type objects.
auth.key.secret.param	string	no	Name of the API key secret parameter, e.g. "secret"
auth.key.secret.type	string	no	Type of secret signature method, e.g. "signed_md5" or "signed_sha256"
auth.basicAuth	boolean	no	Whether the basic authentication is required, either "true" or "false".
auth.oauth	object	no	OAuth version and flow definition.
auth.oauth.version	enum string	no	OAuth version (e.g. “2.0”). "1.0a" and "2.0"

Property Name	**Type**	**Required**	**Description**
auth.oauth.base_uri	string	no	The base URI for the OAuth service endpoints. This value prepends the *authorize_uri* and *access_token_uri* values below, e.g. "http://yourdomain.com" (optional)
auth.oauth.authorize_uri	string	no	The endpoint where the end-user is sent to authorize their account and grant permissions, e.g. "/oauth/authorize". If *base_uri* is not defined, a fully qualified URI is required.
auth.oauth.access_token_uri	string	no	The endpoint where the access token is granted, e.g. "/oauth/access_token". If *base_uri* is not defined, a fully qualified URI is required.
auth.oauth.access_token_location	enum type string	no	The location of the oauth access token. Possible values: "header", "querystring", "body (url-encoded)" to map to the RFC-6750 section 2 values
auth.oauth.auth_flows	array	no	Array of strings, each string containing one of the four supported flows, e.g. "auth_code", "client_cred", "password_cred", or "implicit"
auth.oauth.options	object	no	Links to OAuth options objects, e.g. "authorize" (often used to hold *authorize* and *scope* details – see examples #3 and #4 below). (optional)

Property Name	**Type**	**Require d**	**Description**
schemas	object	no	The schemas (body structure definitions) used by the web API.
schemas.*(key)*	object	no	An individual schema description. See JSON schema for more information.
schemas.*(key)*.id	string	no	Unique identifier for this schema. Example: URL
schemas.*(key)*.type	string	yes	The value type for this schema. A list of values can be found at the "type" section in the JSON Schema.
schemas.*(key)*.$ref	string	no	A reference to another schema. The value of this property is the ID of another schema.
schemas.*(key)*.description	string	no	A description of this object.
schemas.*(key)*.default	string	no	The default value of this property (if one exists).
schemas.*(key)*.required	boolean	no	Whether the parameter is required.
schemas.*(key)*.format	string	no	An additional regular expression or key that helps constrain the value. For more details see the Type and Format Summary.
schemas.*(key)*.pattern	string	no	The regular expression this parameter must conform to.
schemas.*(key)*.minimum	string	no	The minimum value of this parameter.
schemas.*(key)*.maximum	string	no	The maximum value of this parameter.
schemas.*(key)*.enum[]	array	no	Values this parameter may take (if it is an enum).

Property Name	**Type**	**Require d**	**Description**
schemas.*(key)*.enumDescri ptions[]	array	no	The descriptions for the enums. Each position maps to the corresponding value in theenum array.
schemas.*(key)*.properties	object	no	If this is a schema for an object, list the schema for each property of this object.
schemas.*(key)*.properties.*(k ey)*	object	no	A single property of this object. The value is itself a JSON Schema object describing this property.
schemas.*(key)*.additionalPr operties	object	no	If this is a schema for an object, this property is the schema for any additional properties with dynamic keys on this object.
schemas.*(key)*.items	object	no	If this is a schema for an array, this property is the schema for each element in the array.
resources	object	yes	The resources in the API.
resources.*(key)*	object	yes	An individual resources description or name. Contains methods related to this resource.
resources.*(key)*.methods	object	yes	Methods on this resource.
resources.*(key)*.methods.*(k ey)*	object	yes	Description for any methods on this resource. The value at this level (string) contains name of this method.
resources.*(key)*.methods.*(k ey)*.path	string	yes	The URI path of this REST method. Should be used in conjunction with the basePath property at the API-level.
resources.*(key)*.methods.*(k ey)*.httpMethod	string	yes	HTTP method used by this method, e.g. "GET", “POST”, “PUT”, “DELETE”.

Property Name	**Type**	**Required**	**Description**
resources.*(key)*.methods.*(key)*.description	string	yes	Description of this method. This value supports Markdown syntax.
resources.*(key)*.methods.*(key)*.parameters	object	yes	Details for all parameters in this method.
resources.*(key)*.methods.*(key)*.parameters.*(key)*	object	yes	Details for a single parameter in this method (object). The value assigned at this level (string) would be the name of the parameter.
resources.*(key)*.methods.*(key)*.parameters.*(key)*.description	string	yes	A description of the parameter. This value supports Markdown syntax.
resources.*(key)*.methods.*(key)*.parameters.*(key)*.default	string	yes	The default value of this property (if one exists).
resources.*(key)*.methods.*(key)*.parameters.*(key)*.required	boolean	yes	Whether the parameter is required, either "true" or "false".
resources.*(key)*.methods.*(key)*.parameters.*(key)*.enum	array	no	Values this parameter may take.
resources.*(key)*.methods.*(key)*.parameters.*(key)*.enumDescriptions	array	no	The descriptions for the enums. Each position maps to the corresponding value in the "enum" array.
resources.*(key)*.methods.*(key)*.parameters.*(key)*.location	enum type string	yes	Whether this parameter goes in the query, path (for REST requests) or header. Valid values are "query", "pathReplace", "body" and "header".

Property Name	**Type**	**Required**	**Description**
resources.*(key)*.methods.*(key)*.parameters.*(key)*.type	string	no	The parameter type, which the I/O docs UI uses to provide an optimal form field editor. Valid values are: "boolean", "date", "double", "integer", "number", "string", and "textarea".
resources.*(key)*.methods.*(key)*.request	object	no	Schema for the request body.
resources.*(key)*.methods.*(key)*.request.$ref	string	no	Schema ID for the request schema. Value is a string key reference to a schema defined in schemas.*(key)*.
resources.*(key)*.methods.*(key)*.response	object	no	Schema for the response body.
resources.*(key)*.methods.*(key)*.response.$ref	string	no	Schema ID for the response schema. Value is a string key reference to a schema defined in schemas.*(key)*.

Example #1 – Key Only Auth

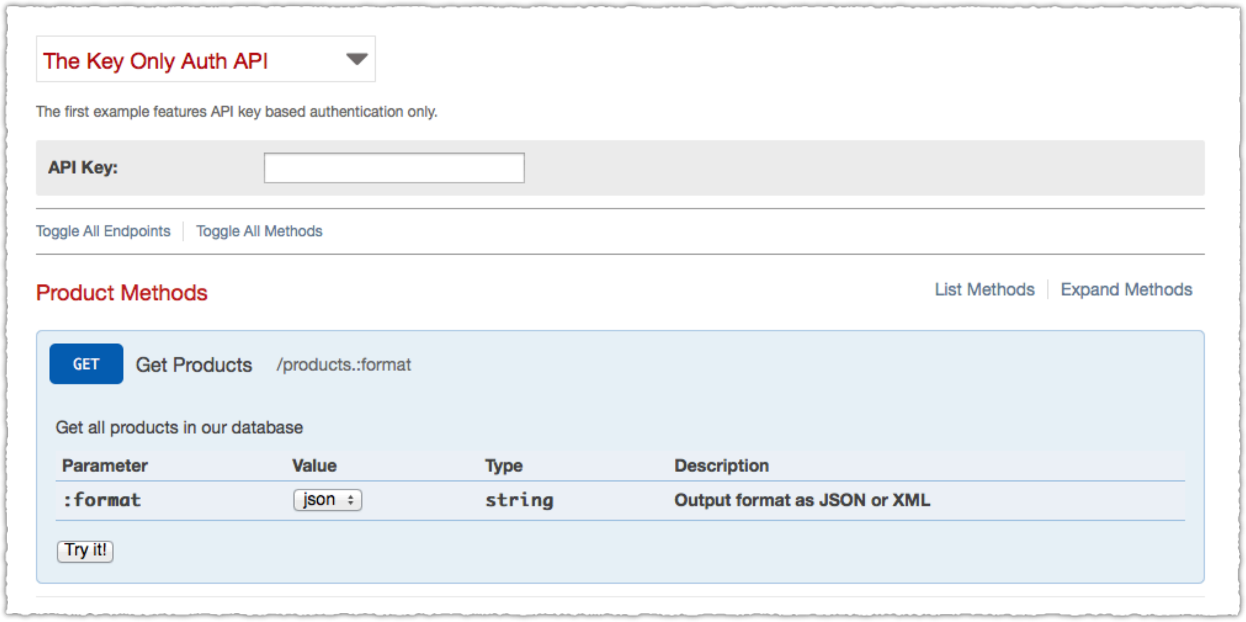
This is an example of an I/O Docs Definition for an API that uses key-based authorization, where the key is passed in a query string parameter named "api_key".

Example #1 - I/O Docs Definition

```
{
    "name": "Example #1 API",
    "version": "1.0",
    "title": "The Key Only Auth API",
    "description": "The first example features API key based authentication only.",
    "protocol": "rest",
    "basePath": " http://api.example1.com/v1",
    "auth": {
        "key": {
            "param": "api_key",
            "location": "query"

        }
    },
    "resources": {
        "Product Methods": {
            "methods": {
                "Get Products": {
                    "path": "/products.:format",
                    "httpMethod": "GET",
                    "description": "Get all products in our database",
                    "parameters": {
                        ":format": {
                            "type": "string",
                            "required": true,
                            "default": "json",
                            "description": "Output format as JSON or XML",
                            "enum": [
                                "json",
                                "xml"
                            ],
                            "location": "pathReplace"
                        }
                    }
                }
            }
        }
    }
}
```

Example #1 - I/O Docs Rendered



The first example features API key based authentication only.

Product Methods

Get Products /products.:format

Get all products in our database

Parameter	Value	Type	Description
:format	json	string	Output format as JSON or XML

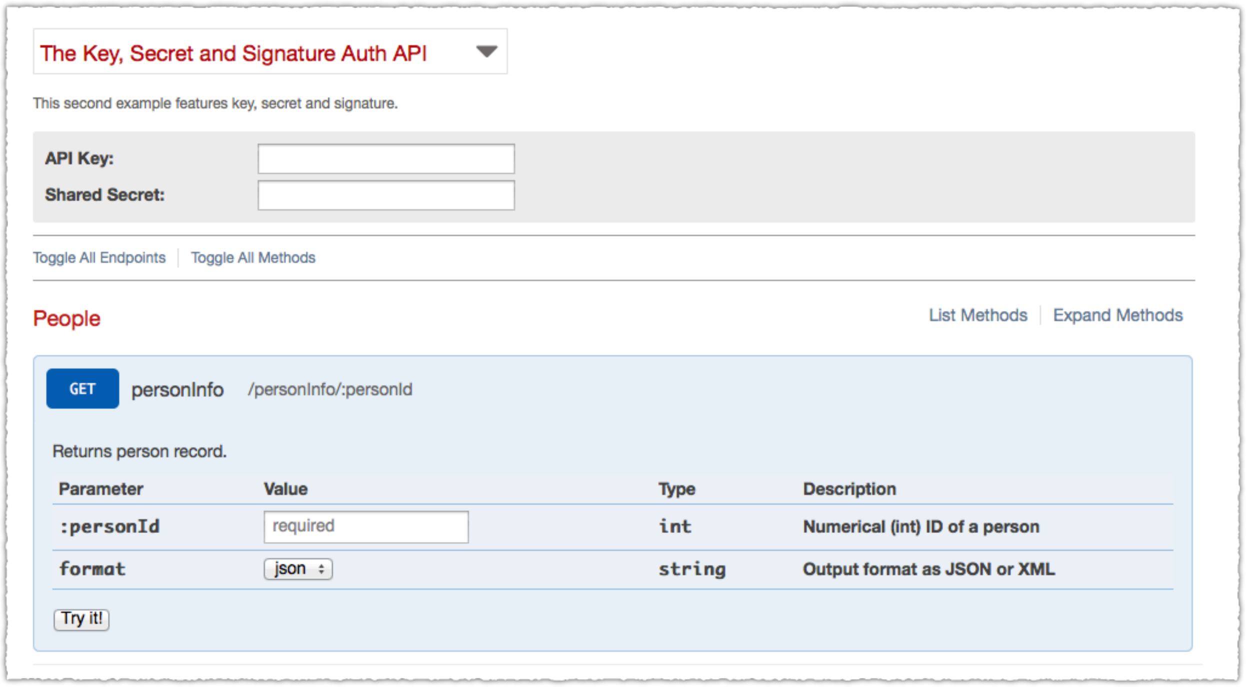
Example #2 – Key, Secret and Signature Auth

This is an example of an I/O Docs definition for an API that uses key-based authorization with an MD5 hash signature. The string that is hashed is a concatenation of the following values: API key, secret, epoch (POSIX time).

Example #2 - I/O Docs Definition

```
{
    "name": "Example #2 API",
    "version": "1.0",
    "title": "The Key, Secret and Signature Auth API",
    "description": "This second example features key, secret and signature.",
    "protocol": "rest",
    "basePath": "http://api.example2.com/v2",
    "auth": {
        "key": {
            "param": "api_key",
            "location": "query"
            "secret": {
               "param": "sig",
               "type": "signed_md5"
           }
      }
},
    "resources": {
        "People": {
            "methods": {
                "personInfo": {
                    "path": "/personInfo/:personId",
                    "httpMethod": "GET",
                    "description": "Returns person record. ",
                    "parameters": {
                        ":personId": {
                            "type": "int",
                            "required": true,
                            "default": "",
                            "description": "Numerical (int) ID of a person",
                            "location": "pathReplace"
                        },
                        "format": {
                            "type": "string",
                            "required": true,
                            "default": "json",
                            "description": "Output format as JSON or XML",
                            "enum": [
                                "json",
                                "xml"
                            ],
                            "location": "query"
                        }
                    }
                }
            }
        }
    }
}
```

Example #2 - I/O Docs Rendered



This second example features key, secret and signature.

API Key:
Shared Secret:

People

personInfo /personInfo/:personId

Returns person record.

Parameter	Value	Type	Description
:personId	required	int	Numerical (int) ID of a person
format	json	string	Output format as JSON or XML

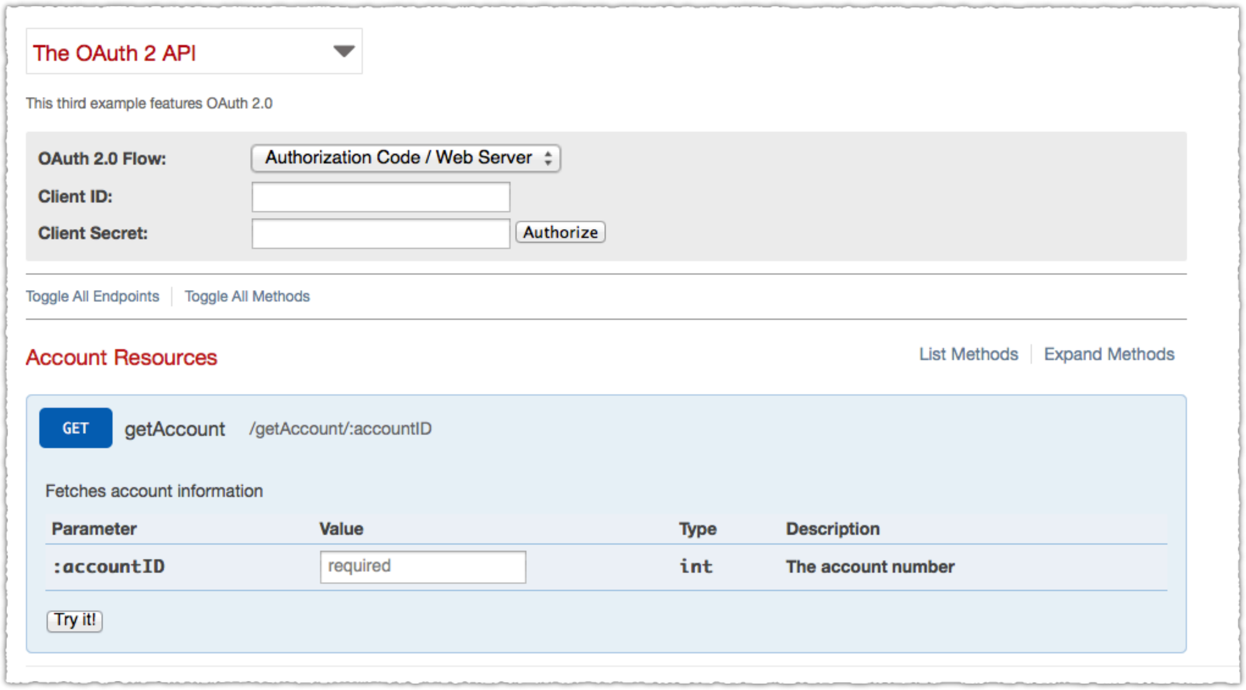
Example #3 – OAuth 2.0 API

This is an example of an I/O Docs Definition for an API that uses OAuth 2.0 and the authentication code flow. Additional options for scope are also included in this configuration. OAuth 1.0 and 1.0a are not supported by I/O Docs.

Example #3 - I/O Docs Definition

```
{
    "name": "Example #3 API",
    "version": "1.0",
    "title": "The OAuth 2 API",
    "description": "This third example features OAuth 2.0",
    "protocol": "rest",
    "basePath": " http://api.example3.com/v3",
    "auth": {
        "oauth": {
            "version": "2.0",
            "auth_flows": ["auth_code"],
            "base_uri": "http://api.example3.com",
            "authorize_uri": "/oauth/authorize",
            "access_token_uri": "/oauth/token",
            "access_token_location": "header",
            "options": {
                "authorize": {
                    "scope": [
                       "read",
                       "write",
                       "execute"
                    ]
                }
            }
        }
    },
    "resources": {
        "Account Resources": {
            "methods": {
                "getAccount": {
                    "path": "/getAccount/:accountID",
                    "httpMethod": "GET",
                    "description": "Fetches account information",
                    "parameters": {
                        ":accountID": {
                            "type": "int",
                            "required": true,
                            "default": "",
                            "description": "The account number",
                            "location": "pathReplace"
                        }
                    }
                }
            }
        }
    }
}
```

Example #3 - I/O Docs Rendered



Parameter	Value	Type	Description
:accountID	required	int	The account number

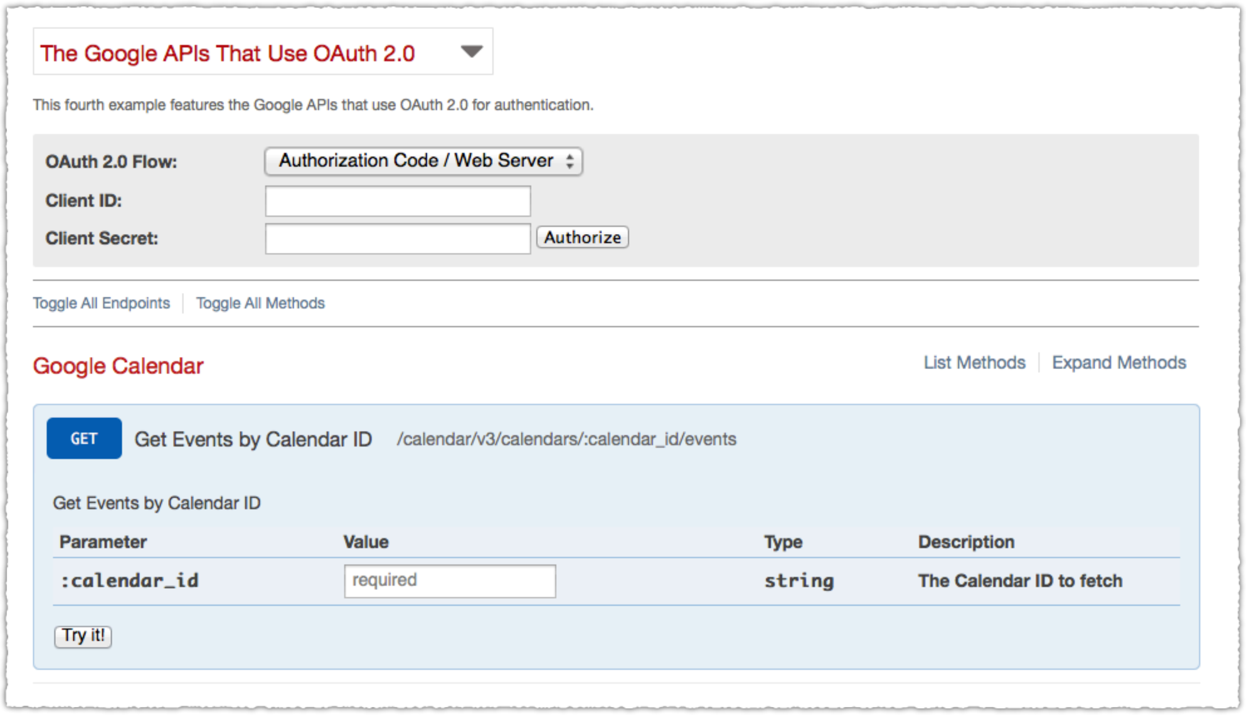
Example #4 – Google OAuth 2.0 API

This is an example of an I/O Docs definition for the Google APIs that use OAuth 2.0.

Example #4 - I/O Docs Definition

```
{
    "name": "Google APIs",
    "version": "1.0",
    "title": "The Google APIs That Use OAuth 2.0",
    "description": "This fourth example features the Google APIs that use OAuth 2.0
for authentication.",
    "protocol": "rest",
    "basePath": "https://www.googleapis.com",
    "auth": {
        "oauth": {
            "version": "2.0",
            "base_uri": "https://accounts.google.com",
            "auth_flows": [
                "auth_code",
                "implicit"
            ],
            "authorize_uri": "/o/oauth2/auth",
            "access_token_uri": "/o/oauth2/token",
            "options": {
                "authorize": {
                    "scope": ["https://www.googleapis.com/auth/calendar"],
                    "access_type": "online",
                    "approval_prompt": "force"
                },
                "auto_exchange_auth_code": true
            }
        }
    },
    "resources": {
        "Google Calendar": {
            "methods": {
                "Get Events by Calendar ID": {
                    "description": "Get Events by Calendar ID",
                    "httpMethod": "GET",
                    "path": "/calendar/v3/calendars/:calendar_id/events",
                    "parameters": {
                        ":calendar_id": {
                            "required": "true",
                            "default": "",
                            "type": "string",
                            "description": "The Calendar ID to fetch",
                            "location": "pathReplace"
                        }
                    }
                }
            }
        }
    }
}
```

Example #4 - I/O Docs Rendered



This fourth example features the Google APIs that use OAuth 2.0 for authentication.

OAuth 2.0 Flow: Authorization Code / Web Server
Client ID:
Client Secret:

Google Calendar

Get Events by Calendar ID /calendar/v3/calendars/:calendar_id/events

Get Events by Calendar ID

Parameter	Value	Type	Description
:calendar_id	required	string	The Calendar ID to fetch

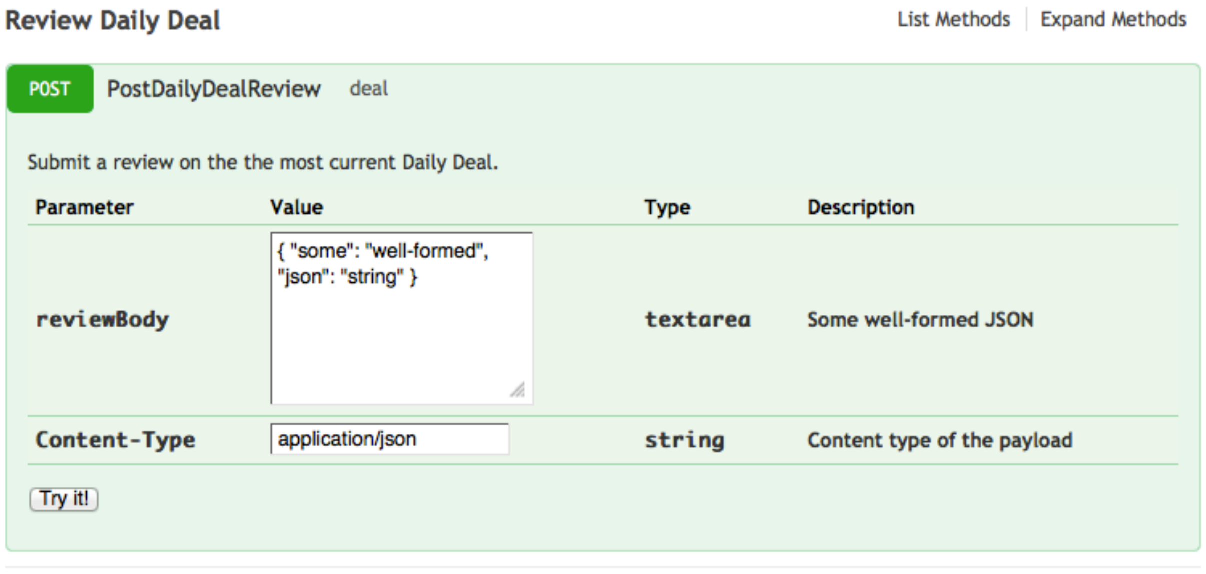
Example #5 – Post Body

This is an example of an I/O Docs Definition for an API that uses key-based authorization, where the key is passed in a query string parameter named "api_key".

Example #5 - I/O Docs Definition

```
{
   "name":"acme daily deal api definition",
   "version":"1.0",
   "title":"Acme Daily Deal API",
   "description":"This Daily Deal API is used to find the most current daily
          deals.",
   "protocol":"rest",
   "basePath":"http://api.acmeapparelstore.com/",
   "auth":{
      "key":{
         "param":"apikey",
         "location": "query"
      }
   },
   "resources":{
           "Review Daily Deal":{
            "methods": {
                           "PostDailyDealReview": {
                         "path":"deal",
                         "httpMethod":"POST",
                         "description":"Submit a review on the the most current
          Daily Deal.",
                         "parameters":{
                                      "reviewBody": {
                                 "description": "Some well-formed JSON",
                                 "default": "{ \"some\": \"well-formed\", \"json\":
          \"string\" }",
                                 "type": "textarea",
                                 "required": true,
                                 "location": "body"
                             },
                             "Content-Type":{
                                 "type":"string",
                                 "required":"true",
                                 "description":"Content type of the payload",
                                 "default":"application/json",
                                 "location": "header"
                             }
                                     }
                     }
                      }
        }
   }
}
```

Example #5 - I/O Docs Rendered



Review Daily Deal

POST PostDailyDealReview deal

Submit a review on the the most current Daily Deal.

Parameter	Value	Type	Description
reviewBody	{ "some": "well-formed", "json": "string" }	textarea	Some well-formed JSON
Content-Type	application/json	string	Content type of the payload