Internet Engineering Task Force (IETF)
Request for Comments: 9727
Category: Standards Track
ISSN: 2070-1721

K. Smith
Vodafone

June 2025

api-catalog: A Well-Known URI and Link Relation to Help Discovery of APIs

Abstract

This document defines the "api-catalog" well-known URI and link relation. It is intended to facilitate automated discovery and usage of published Application Programming Interfaces (APIs). A request to the api-catalog resource will return a document providing information about, and links to, the Publisher’s APIs.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9727.

Table of Contents

1. Introduction

An application may publish APIs to encourage requests for interaction from external parties. Such APIs must be discovered before they may be used, i.e., the external party needs to know what APIs a given Publisher exposes, their purpose, any policies for usage, and the endpoint to interact with each API. To facilitate automated discovery of this information and automated usage of the APIs, this document proposes:

* a well-known URI [WELL-KNOWN], "api-catalog", that is encoded as a URI reference to an API catalog document describing a Publisher's API endpoints.

* a link relation [WEB-LINKING], "api-catalog", of which the target resource is the Publisher's API catalog document.

1.1. Goals and Non-Goals

The primary goal of this document is to facilitate the automated discovery of a Publisher's public API endpoints, along with metadata that describes the purpose and usage of each API, by specifying a well-known URI that returns an API catalog document. The API catalog document is primarily machine-readable to enable automated discovery and usage of APIs, and it may also include links to human-readable documentation (see the example in Appendix A.1).

Non-goals: This document does not mandate paths for API endpoints, i.e., it does not mandate that my_example_api's endpoint should be https://www.example.com/.well-known/api-catalog/my_example_api, nor even to be hosted at www.example.com (although it is not forbidden to do so).

1.2. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

The terms "content negotiation" and "status code" are from [HTTP]. The term "well-known URI" is from [WELL-KNOWN]. The term "link

relation" is from [WEB-LINKING].

The term "Publisher" refers to an organisation, company, or individual that publishes one or more APIs for use by external third parties. A fictional Publisher named "example" is used throughout this document. The examples use the Fully Qualified Domain Names (FQDNs) "www.example.com", "developer.example.com", "apis.example.com", "apis.example.net", "gaming.example.com", and "iot.example.net", where the .com and .net Top-Level Domains (TLDs) and various subdomains are simply used to illustrate that the "example" Publisher may have their API portfolio distributed across various domains for which they are the authority. Scenarios where the Publisher "example" is not the authority for a given _.example._ domain are made explicit in the text.

In this document, "API" refers to the specification resources required for an external party (or in the case of "private" APIs, an internal party) to implement software that uses the Publisher's API.

The specification recommends the use of TLS. Hence, "HTTPS" and "https://" are used throughout.

2. Using the "api-catalog" Well-Known URI

The api-catalog well-known URI is intended for HTTPS servers that publish APIs.

* The API catalog MUST be named "api-catalog" in a well-known location as described by [WELL-KNOWN].

* The location of the API catalog document is decided by the Publisher. The /.well-known/api-catalog URI provides a convenient reference to that location.

A Publisher supporting this URI:

* SHALL resolve an HTTPS GET request to /.well-known/api-catalog and return an API catalog document (as described in Section 4).

* SHALL resolve an HTTPS HEAD request to /.well-known/api-catalog with a response including a Link header with the relation(s) defined in Section 3.

3. The api-catalog Link Relation

This document introduces a new link relation [WEB-LINKING], "api-catalog". This identifies a target resource that represents a list of APIs available from the Publisher of the link context. The target resource URI may be /.well-known/api-catalog or any other URI chosen by the Publisher. For example, the Publisher "example" could include the api-catalog link relation in the HTTP header and/or content payload when responding to a request to https://www.example.com:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Location: /index.html
Link: </my_api_catalog.json>; rel=api-catalog
Content-Length: 356

<!DOCTYPE HTML>
  <html>
    <head>
      <title>Welcome to Example Publisher</title>
    </head>
    <body>
      <p>
```

```
        <a href="my_api_catalog.json" rel="api-catalog">
         Example Publisher's APIs
        </a>
       </p>
       <p>(remainder of content)</p>
     </body>
   </html>
```

3.1. Using Additional Link Relations

When used in an API catalog document, the "item" [RFC6573] link relation identifies a target resource that represents an API that is a member of the API catalog.

Other link relations may be utilised in an API catalog to convey metadata descriptions for API links.

4. The API Catalog Document

The API catalog is a document listing a Publisher's APIs. The Publisher may host the API catalog document at any URI(s) they choose. For example, the API catalog document URI of https://www.example.com/my_api_catalog.json can be requested directly or via a request to https://www.example.com/.well-known/api-catalog, which the Publisher will resolve to https://www.example.com/my_api_catalog.

4.1. API Catalog Contents

The API catalog MUST include hyperlinks to API endpoints. It is RECOMMENDED that the API catalog also includes useful metadata, such as usage policies, API version information, links to the OpenAPI Specification [OAS] definitions for each API, etc. If the Publisher does not include that metadata directly in the API catalog document, they SHOULD make that metadata available at the API endpoint URIs they have listed (see Appendix A.2 for an example).

4.2. API Catalog Formats

The Publisher MUST publish the API catalog document in the Linkset format application/linkset+json (Section 4.2 of [RFC9264]). The Linkset SHOULD include a profile parameter (Section 5 of [RFC9264]) with a Profile URI [RFC7284] value of "https://www.rfc-editor.org/info/rfc9727" to indicate the Linkset is representing an API catalog document as defined above. Appendix A includes example API catalog documents based on the Linkset format.

The Publisher MAY make additional formats available via content negotiation (Section 12 of [HTTP]) to their /.well-known/api-catalog location. A non-exhaustive list of such formats that support the automated discovery and machine (and human) usage of a Publisher's APIs is listed at Appendix A.3. If a Publisher already lists their APIs in a format other than Linkset, but wishes to utilise the /.well-known/api-catalog URI, then:

* They MUST also implement a Linkset with, at minimum, hyperlinks to API endpoints; see Appendix A.2.

* They MAY support content negotiation at the /.well-known/api-catalog URI to allow for the return of their existing format.

4.3. Nesting API Catalog Links

An API catalog may itself contain links to other API catalogs by using the "api-catalog" relation type for each link. An example of this is given in Appendix A.4.

5. Operational Considerations

5.1. Accounting for APIs Distributed Across Multiple Domains

A Publisher ("example") may have their APIs hosted across multiple domains that they manage, e.g., at www.example.com, developer.example.com, apis.example.com, apis.example.net, etc. They may also use a third-party API hosting provider that hosts APIs on a distinct domain.

To account for this scenario, it is RECOMMENDED that:

* The Publisher also publish the api-catalog well-known URI at each of their API domains, e.g., https://apis.example.com/.well-known/api-catalog, https://developer.example.net/.well-known/api-catalog, etc.

* An HTTPS GET request to any of these URIs returns the same result, namely, the API catalog document.

* The Publisher choose one of their instances of /.well-known/api-catalog as a canonical reference to the location of the latest API catalog since the physical location of the API catalog document is decided by the Publisher and may change. The Publisher’s other instances of /.well-known/api-catalog should redirect to this canonical instance of /.well-known/api-catalog to ensure the latest API catalog is returned.

For example, if the Publisher’s primary API portal is https://apis.example.com, then https://apis.example.com/.well-known/api-catalog should resolve to the location of the Publisher’s latest API catalog document. If the Publisher is also the domain authority for www.example.net, which also hosts a selection of their APIs, then a request to https://www.example.net/.well-known/api-catalog should redirect to https://apis.example.com/.well-known/api-catalog.

If the Publisher is not the domain authority for www.example.net, then the Publisher’s API Catalog MAY include a link to the API catalog of the third-party that is the domain authority for www.example.net. For example, the API catalog available at https://apis.example.com/.well-known/api-catalog may list APIs hosted at apis.example.com and also link to the API catalog hosted at https://www.example.net/.well-known/api-catalog using the "api-catalog" link relation:

```
{
 "linkset": [
  {
   "anchor": "https://www.example.com/.well-known/api-catalog",
   "item": [
    {
     "href": "https://developer.example.com/apis/foo_api"
    },
    {
     "href": "https://developer.example.com/apis/bar_api"
    },
    {
     "href": "https://developer.example.com/apis/cantona_api"
    }
   ],
   "api-catalog": "https://www.example.net/.well-known/api-catalog"
  }
 ]
}
```

5.2. Internal Use of api-catalog for Private APIs

A Publisher may wish to use the api-catalog well-known URI on their internal network to signpost authorised users (e.g., company employees) towards internal/private APIs not intended for third-party use. This scenario may incur additional security considerations as noted in Section 8.

5.3. Scalability Guidelines

In cases where a Publisher has a large number of APIs potentially deployed across multiple domains, two challenges may arise:

* Maintaining the catalog entries to ensure they are up to date and correcting any errors.

* Restricting the catalog size to help reduce network and client-processing overheads.

In both cases, a Publisher may benefit from grouping their APIs, providing an API catalog document for each group and using the main API catalog hosted at /.well-known/api-catalog to provide links to these. For example, a Publisher may decide to group their APIs according to a business category (e.g., "gaming APIs", "anti-fraud APIs", etc.), a technology category (e.g., "IOT", "networks", "AI", etc.), or any other criterion. This grouping may be implicit where the Publisher has already published their APIs across multiple domains, e.g., at gaming.example.com, iot.example.net, etc.

Section 4.3 shows how the API catalog at /.well-known/api-catalog can use the api-catalog link relation to point to other API catalogs.

The Publisher SHOULD consider caching and compression techniques to reduce the network overhead of large API catalogs.

5.4. Monitoring and Maintenance

Publishers are RECOMMENDED to follow operational best practice when hosting API catalog(s), including, but not limited to:

* Availability. The Publisher should monitor availability of the API catalog and consider alternate means to resolve requests to /.well-known/api-catalog during planned downtime of hosts.

* Performance. Although the performance of APIs listed in an API catalog can demand high transactions per second and low-latency response, the retrieval of the API catalog itself to discover those APIs is less likely to incur strict performance demands. That said, the Publisher should monitor the response time to fulfil a request for the API catalog and determine any necessary improvements (as with any other Web resource the Publisher serves). For large API catalogs, the Publisher should consider the techniques described in Section 5.3.

* Usage. Since the goal of the api-catalog well-known URI is to facilitate discovery of APIs, the Publisher may wish to correlate requests to the /.well-known/api-catalog URI with subsequent requests to the API URIs listed in the catalog.

* Current data. The Publisher should include the removal of stale API entries from the API catalog as part of their API release lifecycle. The Publisher MAY decide to include metadata regarding legacy API versions or deprecated APIs to help users of those APIs discover up-to-date alternatives.

* Correct metadata. The Publisher should include human and/or

automated checks for syntax errors in the API catalog. Automated checks include format validation (e.g., to ensure valid JSON syntax) and linting to enforce business rules, such as removing duplicate entries and ensuring descriptions are correctly named with valid values. A proofread of the API catalog as part of the API release lifecycle is RECOMMENDED to detect any errors in business grammar (for example, an API entry that is described with valid syntax, but has been allocated an incorrect or outdated description.)

* Security best practice. See Section 8.

5.5. Integration with Existing API Management Frameworks

A Publisher may already utilise an API management framework to produce their API portfolio. These frameworks typically include the publication of API endpoint URIs, deprecation and redirection of legacy API versions, API usage policies and documentation, etc. The api-catalog well-known URI and API catalog document are intended to complement API management frameworks by facilitating the discovery of the framework's outputs -- API endpoints, usage policies, and documentation -- and are not intended to replace any existing API discovery mechanisms the framework has implemented.

Providers of such frameworks may include the production of an API catalog and the publication of the /.well-known/api-catalog URI as a final pre-release (or post-release) step in the release management workflow. The following steps are recommended.

If the /.well-known/api-catalog URI has not been published previously, the framework provider should:

* Collate and check the metadata for each API that will be included in the API catalog. This metadata is likely to already exist in the framework.

* Determine which metadata to include in the API catalog following the requirements set out in Section 4.1 and the considerations set out in Section 5.

* Map the chosen metadata to the format(s) described in Section 4.2. The structure suggested in Appendix A.2 may be followed where only the hyperlinks to APIs are to be included in the API catalog. Where possible, the API catalog should include further metadata per the guidance in Section 4.1; in which case, the structure suggested in Appendix A can be utilised and adapted (ensuring compliance to [RFC9264]) to reflect the nature of the chosen metadata.

* Publish the /.well-known/api-catalog URI following the guidance set out in Section 2.

If the /.well-known/api-catalog URI has previously been published, the framework provider should:

* Include a step in the release management lifecycle to refresh the API catalog following any changes in API hyperlinks or published metadata. This could include placing triggers on certain metadata fields, so that as they are updated in pre-production on the API framework, the updates are pushed to a pre-production copy of the API catalog to be pushed live when the release is published by the framework.

6. Conformance to RFC 8615

The requirements in Section 3 of [WELL-KNOWN] for defining Well-Known

URIs are met as described in the following subsections.

6.1. Path Suffix

The api-catalog URI SHALL be appended to the /.well-known/ path-prefix for "well-known locations".

6.2. Formats and Associated Media Types

A /.well-known/api-catalog location MUST support the Linkset [RFC9264] format of application/linkset+json and MAY also support the other formats via content negotiation.

7. IANA Considerations

7.1. The api-catalog Well-Known URI

This specification registers the "api-catalog" well-known URI in the "Well-Known URIs" registry as defined by [WELL-KNOWN].

URI Suffix: api-catalog
Reference: RFC 9727
Status: permanent
Change Controller: IETF

7.2. The api-catalog Link Relation

This specification registers the "api-catalog" link relation in the "Link Relation Types" registry by following the procedures per Section 2.1.1.1 of [WEB-LINKING].

Relation Name: api-catalog
Description: Refers to a list of APIs available from the Publisher of the link context.
Reference: RFC 9727

7.3. The api-catalog Profile URI

This specification registers "https://www.rfc-editor.org/info/rfc9727" in the "Profile URIs" registry according to [RFC7284].

Profile URI: https://www.rfc-editor.org/info/rfc9727
Common Name: API catalog
Description: A Profile URI to request or signal a Linkset representing an API catalog.
Reference: RFC 9727

8. Security Considerations

For all scenarios:

* TLS SHOULD be used, i.e., make /.well-known/api-catalog available exclusively over HTTPS, to ensure no tampering of the API catalog.

* The Publisher SHOULD take into account the security considerations from Section 4 of [WELL-KNOWN].

* The Publisher SHOULD perform a security and privacy review of the API catalog prior to deployment to ensure it does not leak personal, business, or other sensitive metadata, nor expose any vulnerability related to the APIs listed.

* The Publisher SHOULD enforce read-only privileges for external requests to .well-known/api-catalog and for internal systems and roles that monitor the .well-known/api-catalog URI. Write privileges SHOULD only be granted to roles that perform updates to

the API catalog and/or the forwarding rewrite rules for the .well-known/api-catalog URI.

* As with any Web offering, it is RECOMMENDED to apply rate-limiting measures to help mitigate abuse and prevent denial-of-service attacks on the API catalog endpoint.

For the public-facing APIs scenario, security teams SHOULD additionally audit the API catalog to ensure no APIs intended solely for internal use have been mistakenly included. For example, a catalog hosted on https://developer.example.com should not expose unnecessary metadata about any internal domains (e.g., https://internal.example.com).

For the internal/private APIs scenario, the Publisher SHOULD take steps to ensure that appropriate controls, such as Cross-Origin Resource Sharing (CORS) policies and access control lists, are in place to ensure only authorised roles and systems may access an internal api-catalog well-known URI.

A comprehensive API catalog that is regularly audited may assist the Publisher in decommissioning "zombie" APIs, i.e., legacy/obsolete APIs that should no longer be available. Such APIs represent a security vulnerability as they are unlikely to be supported, monitored, patched, or updated.

Note the registration of domain names and associated policies is out of scope of this document.

9. References

9.1. Normative References

[HTTP] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/info/rfc9110>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC6573] Amundsen, M., "The Item and Collection Link Relations", RFC 6573, DOI 10.17487/RFC6573, April 2012, <https://www.rfc-editor.org/info/rfc6573>.

[RFC7284] Lanthaler, M., "The Profile URI Registry", RFC 7284, DOI 10.17487/RFC7284, June 2014, <https://www.rfc-editor.org/info/rfc7284>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC9264] Wilde, E. and H. Van de Sompel, "Linkset: Media Types and a Link Relation Type for Link Sets", RFC 9264, DOI 10.17487/RFC9264, July 2022, <https://www.rfc-editor.org/info/rfc9264>.

[WEB-LINKING] Nottingham, M., "Web Linking", RFC 8288, DOI 10.17487/RFC8288, October 2017, <https://www.rfc-editor.org/info/rfc8288>.

[WELL-KNOWN]

Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <https://www.rfc-editor.org/info/rfc8615>.

9.2. Informative References

[APIsjson] Lane, K. and S. Willmott, "API Discovery Format", 6 November 2024, <https://apisjson.org/format/apisjson_0.19.txt>. Latest version available at <https://apisjson.org/>.

[HAL] Kelly, M., "JSON Hypertext Application Language", Work in Progress, Internet-Draft, draft-kelly-json-hal-11, 19 October 2023, <https://datatracker.ietf.org/doc/html/draft-kelly-json-hal-11>.

[OAS] Miller, D., Ed., Andrews, H., Ed., Whitlock, J., Ed., Mitchell, L., Ed., Gardiner, M., Ed., Quintero, M., Ed., Kistler, M., Ed., Handl, R., Ed., and R. Ratovsky, Ed., "OpenAPI Specification v3.1.0", 24 October 2024, <https://spec.openapis.org/oas/latest>. Latest version available at <https://spec.openapis.org/oas/latest.html>.

[RESTdesc] Verborgh, R., Mannens, E., Van de Walle, R., and T. Steiner, "RESTdesc", 2025, <https://restdesc.org/about/descriptions>.

[RFC8631] Wilde, E., "Link Relation Types for Web Services", RFC 8631, DOI 10.17487/RFC8631, July 2019, <https://www.rfc-editor.org/info/rfc8631>.

[WebAPIext] Ralphson, M., Ed. and N. Evans, Ed., "WADG0001 WebAPI type extension", Draft Community Group Report, 8 July 2020, <https://webapi-discovery.github.io/rfcs/rfc0001.html>.

Appendix A. Example API Catalog Documents

This section is informative and provides and example of an API catalog document using the Linkset format.

A.1. Using Linkset with Link Relations Defined in RFC 8631

This example uses the Linkset format [RFC9264] and the following link relations defined in [RFC8631]:

"service-desc": Used to link to a description of the API that is primarily intended for machine consumption (for example, the [OAS] specification, YAML, or JSON file).

"service-doc": Used to link to API documentation that is primarily intended for human consumption (an example of human-readable documentation is the IETF Internet-Draft submission API instructions <https://datatracker.ietf.org/api/submission>).

"service-meta": Used to link to additional metadata about the API and is primarily intended for machine consumption.

"status": Used to link to the API status (e.g., API "health" indication) for machine and/or human consumption.

Client request:

```
GET .well-known/api-catalog HTTP/1.1
Host: example.com
Accept: application/linkset+json
```

Server response:

```
HTTP/1.1 200 OK
Date: Mon, 01 Jun 2023 00:00:01 GMT
Server: Apache-Coyote/1.1
Content-Type: application/linkset+json;
    profile="https://www.rfc-editor.org/info/rfc9727"

{
 "linkset": [
  {
   "anchor": "https://developer.example.com/apis/foo_api",
   "service-desc": [
    {
     "href": "https://developer.example.com/apis/foo_api/spec",
     "type": "application/yaml"
    }
   ],
   "status": [
    {
     "href": "https://developer.example.com/apis/foo_api/status",
     "type": "application/json"
    }
   ],
   "service-doc": [
    {
     "href": "https://developer.example.com/apis/foo_api/doc",
     "type": "text/html"
    }
   ],
   "service-meta": [
    {
     "href": "https://developer.example.com/apis/foo_api/policies",
     "type": "text/xml"
    }
   ]
  },
  {
   "anchor": "https://developer.example.com/apis/bar_api",
   "service-desc": [
    {
     "href": "https://developer.example.com/apis/bar_api/spec",
     "type": "application/yaml"
    }
   ],
   "status": [
    {
     "href": "https://developer.example.com/apis/bar_api/status",
     "type": "application/json"
    }
   ],
   "service-doc": [
    {
     "href": "https://developer.example.com/apis/bar_api/doc",
     "type": "text/plain"
    }
   ]
  },
  {
   "anchor": "https://apis.example.net/apis/cantona_api",
   "service-desc": [
    {
     "href": "https://apis.example.net/apis/cantona_api/spec",
     "type": "text/n3"
    }
```

```
      ],
      "service-doc": [
       {
        "href": "https://apis.example.net/apis/cantona_api/doc",
        "type": "text/html"
       }
      ]
     }
    ]
   }
```

A.2. Using Linkset with Bookmarks

This example also uses the Linkset format [RFC9264] and lists the API endpoints in an array of bookmarks. Each link shares the same context anchor (the well-known URI of the API catalog) and "item" [RFC9264] link relation (to indicate they are an item in the catalog). The intent is that by following a bookmark link, a machine client can discover the purpose and usage policy for each API; hence, the document targeted by the bookmark link should support this.

Client request:

```
GET .well-known/api-catalog HTTP/1.1
Host: example.com
Accept: application/linkset+json
```

Server response:

```
HTTP/1.1 200 OK
Date: Mon, 01 Jun 2023 00:00:01 GMT
Server: Apache-Coyote/1.1
Content-Type: application/linkset+json;
    profile="https://www.rfc-editor.org/info/rfc9727"

{ "linkset":
 [
   { "anchor": "https://www.example.com/.well-known/api-catalog",
     "item": [
       {"href": "https://developer.example.com/apis/foo_api"},
       {"href": "https://developer.example.com/apis/bar_api"},
       {"href": "https://developer.example.com/apis/cantona_api"}
     ]
   }
 ]
}
```

A.3. Other API Catalog Formats

A non-exhaustive list of other API catalog document formats includes:

* An APIs.json document [APIsjson].

* A RESTDesc semantic description for hypermedia APIs [RESTdesc].

* A Hypertext Application Language document [HAL].

* An extension to the Schema.org WebAPI type [WebAPIext].

A.4. Nesting API Catalog Links

In this example, a request to the /.well-known/api-catalog URI returns an array of links of relation type "api-catalog". This can be useful to Publishers with a large number of APIs who wish to group them in smaller catalogs (as described in Section 5.3).

Client request:

```
GET .well-known/api-catalog HTTP/1.1
Host: example.com
Accept: application/linkset+json
```

Server response:

```
HTTP/1.1 200 OK
Date: Mon, 01 Jun 2023 00:00:01 GMT
Server: Apache-Coyote/1.1
Content-Type: application/linkset+json;
    profile="https://www.rfc-editor.org/info/rfc9727"

{
  "linkset": [
    {
      "anchor": "https://www.example.com/.well-known/api-catalog",
      "api-catalog": [
        {
          "href": "https://apis.example.com/iot/api-catalog"
        },
        {
          "href": "https://ecommerce.example.com/api-catalog"
        },
        {
          "href": "https://developer.example.com/gaming/api-catalog"
        }
      ]
    }
  ]
}
```

Acknowledgements

Thanks to Jan Algermissen, Phil Archer, Tim Bray, Ben Bucksch, Sanjay Dalal, David Dong, Erik Kline, Mallory Knodel, Murray Kucherawy, Max Maton, Darrel Miller, Mark Nottingham, Roberto Polli, Joey Salazar, Rich Salz, Herbert Van De Sompel, Orie Steele, Tina Tsou, Gunter Van de Velde, ric Vyncke, and Erik Wilde for their reviews, suggestions, and support.

Author’s Address

Kevin Smith
Vodafone
Email: kevin.smith@vodafone.com
URI: https://www.vodafone.com