

DID URL Resource Parameter Specification

- ***** IMPORTANT NOTICE *****
- Document Status
- Introduction
- Terminology
- Purpose
- Motivations
- The Resource Parameter
- Syntax
- Examples
 - Without a Path Component
 - With a Path Component
- DID Resolver and Verifiable Data Registry (VDR) Requirements
- DID Method Requirements
- Registration with the W3C DID Specification Registries
- Contributors
- Acknowledgements
- Licensing

*** IMPORTANT NOTICE ***

THIS SPECIFICATION IS BEING SUPERSEDED by the [DID-Linked Resources Specification](#) hosted by the [Utility Foundry Working Group](#) — please see that specification for all future work.

Document Status

This document is a Draft Deliverable of the [ToIP Technical Stack Working Group](#).

The current version is Working Draft 03.

Introduction

This is a specification for an extension to the [W3C Decentralized Identifiers \(DIDs\) 1.0 specification](#) to support the [resource](#) parameter as listed in the [W3C DID Specification Registries 1.0](#). It specifies the syntax and normative behavior for usage of the parameter and the requirements for [DID methods](#) that support the parameter.

Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL", when appearing in ALL CAPITALS, are to be interpreted as described in [RFC 2119](#).

All other terms are linked to their definitions in the [W3C Decentralized Identifiers \(DIDs\) 1.0 specification](#).

Purpose

The purpose of this specification is to specify how a [DID URL](#) may include a parameter that instructs a [DID resolver](#) to request the associated [verifiable data registry](#) (VDR) to directly return a digital resource identified by a [decentralized identifier \(DID\)](#). This parameter is only available for [DID URLs](#) whose [DID method](#) supports the parameter. The parameter also supports requesting the resource in a particular [media type](#).

Motivations

The process of resolving a [DID](#) and dereferencing a [DID URL](#) (meaning a DID that includes an additional path, query, and/or fragment component as defined by the ABNF in [section 3.2 of the DID 1.0 specification](#)) is shown in figure 1 from [section 7.2](#) of the spec:

[blocked URL](#)

Figure 1: The relationship of DIDs, DID URLs, DID documents, and Resources

The normal pattern for processing a [DID URL](#) consists of two steps:

1. **Resolution:** Resolving the plain [DID](#) (defined by the ABNF from [section 3.1 of the DID spec](#)) to a [DID document](#).
2. **Dereferencing:** Processing the [DID document](#) in order to determine how to dereference the remainder of the [DID URL](#) (path, query, fragment, etc.)

The first of these two steps—the DID resolution step—is shown in figure 2.

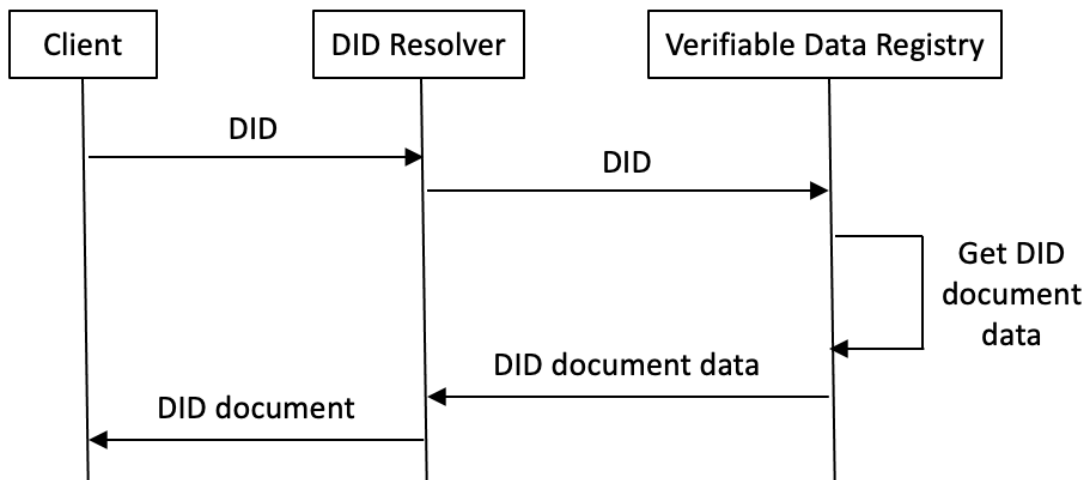


Figure 2: The normal DID resolution process

For a [DID URL](#), the return of the [DID document](#) to the resolver commences the second step of [dereferencing](#) the identified resource based on the portion of the [DID URL](#) that follows the [DID](#). An example is processing a [DID fragment](#) to return a specific public key from the [DID document](#). How [dereferencing](#) operates depends on the [DID URL](#) and the [DID method](#).

However an exception to this normal 2-step resolution/dereferencing process is when **the DID itself identifies a digital resource that can be returned directly by the VDR of the associated DID method**. This behavior may be desirable:

- When the [DID](#) serves as a persistent identifier of a machine-readable digital resource that the client wishes to consume directly, such as a data schema, interface definition, or policy definition.
- When the [DID](#) serves as a persistent identifier of a human-readable document that needs a long-lived, cryptographically verifiable identifier such as a legal document (e.g., title, deed, will, regulatory filing), a governance framework, or a [non-fungible token](#) (NFT) or any other type of digital asset.

In this case, the client MAY wish to use a [DID URL](#) to request that a [DID resolver](#) return the identified digital resource in a single step as shown in Figure 3:

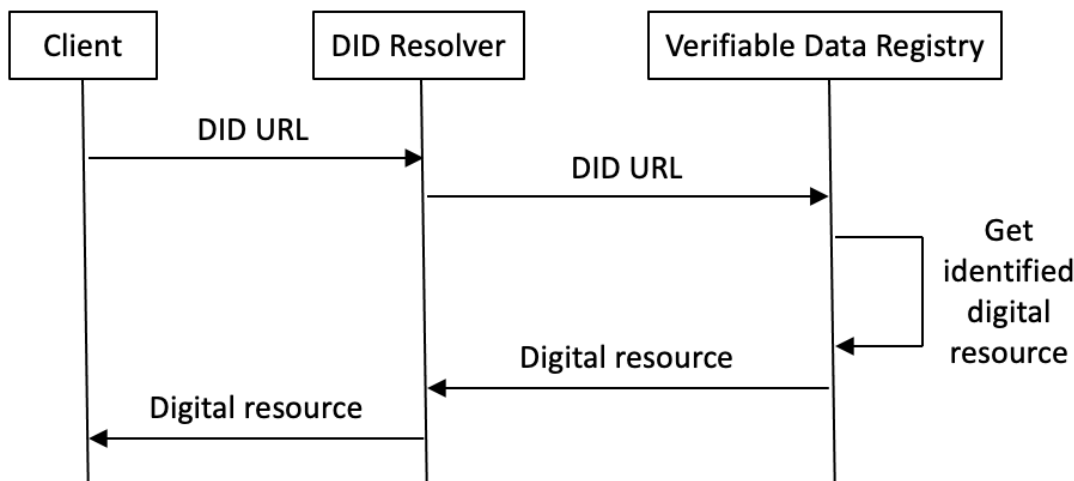


Figure 3: Using a DID URL to request a digital resource in a single step

Two important notes about this process:

1. The [DID document](#) is **not** retrieved by the resolver as part of the dereferencing process. Rather the resolver makes a call to the [VDR](#) with the [DID URL](#) including a [resource](#) parameter conformant with this specification. The [VDR](#) then follows the specification of the associated [DID method](#) to retrieve the identified digital resource and return that resource to the resolver directly.
2. The [resource](#) parameter serves a different purpose than the [alsoKnownAs](#) parameter. The purpose of the [alsoKnownAs](#) parameter is to map a [DID](#) to an alternative identifier for the same [DID subject](#). This alternative identifier can be any other URI, e.g., a URL, or a URN, or another DID. When the [DID](#) is resolved, the value of the [alsoKnownAs](#) parameter is returned as a property of the [DID document](#). It is then up to the client to decide what action to take with this alternative identifier. For example, if the value is a URL, the client may choose to [dereference](#) this URL to retrieve the identified Web resource. In this case, the [alsoKnownAs](#) parameter essentially serves as a redirection mechanism managed by the [DID controller](#). By contrast, the [resource](#) parameter:

- a. Does not involve an alternative identifier for the **DID subject**. The **DID** is the only identifier needed. (Although the DID document may still include an **alsoKnownAs** parameter if desired.)
- b. Does not involve any redirection. The **DID** directly identifies the target digital resource.
- c. Does not require a separate dereferencing step. The **DID URL** is effectively resolved and dereferenced in one step.
- d. Does not involve a service endpoint in the associated DID document. In fact the use of any content in the DID document (other than the DID itself) is an anti-pattern because the purpose of the **resource** parameter is to enable a DID URL to resolve **directly to the digital resource**—including the ability to specify the requested media type of the resource—so that developers are completely abstracted from the mechanics of how the VDR actually returns the identified resource to the requesting DID resolver.

The Resource Parameter

To enable this combined resolution/dereferencing behavior, this specification defines a **DID URL** parameter named **resource**. If a **DID method** specification supports this parameter, and if a **DID URL** using that method includes the parameter with a valid value (either **true** or a valid **media type name**), then when a resolver calls the associated **VDR** using that **DID URL**, the VDR returns the identified digital resource, **not** the **DID document**.

IMPORTANT: A **DID URL** that includes the **resource** parameter does still have an associated **DID document** like any other **DID**. However if the DID URL includes the **resource** parameter, the associated **DID document** is **not involved** in the combined resolution/dereferencing step. If the **DID** alone is resolved (without the **resource** parameter), it will still return the associated **DID document**. *There is nothing special about this DID document.* It still describes and controls interactions with the identified digital resource just like any other **DID document**. For example:

- If the **DID document** contains only an **id** property (whose value must be the **DID**) and no verification methods, then the **DID document** cannot be updated and the identified digital resource is a static resource that cannot be versioned.
- If the **DID document** contains one or more **verification methods**, then the **DID controller** has a method for versioning the identified digital resource and/or the **DID document** itself.

If the **DID document** contains one or more **alsoKnownAs** property values, those values still identify other instances of the identified digital resource available via other URIs, URLs, or **DIDs**. (See also the recommendation below to include an **alsoKnownAs** property whose value is a **hashlink** to the identified digital resource.)

Syntax

The parameter name **MUST** be **resource**. The parameter value **MUST** be one of the following:

1. A Boolean value of **true**. The parameter value of **true** **MUST** be case-sensitive and **MUST** be spelled out exactly. No other renderings of **true**, i.e. T, TRUE, 1, -1, and so forth — are equivalent.
2. A valid **media type name**. If the **VDR** supports the storage of the resource in multiple media types, this option enables the client to request return of a specific media type.

Examples

Without a Path Component

Following are examples of DID URLs that use the **resource** parameter but do not include a path component. These examples are based on the fictional **did:example: DID method** that supports the **resource** parameter:

```
did:example:21tDAKCERh95uGgKbJNHYP?resource=true
```

```
did:example:21tDAKCERh95uGgKbJNHYP?resource=json
```

```
did:example:21tDAKCERh95uGgKbJNHYP?resource=ld+json
```

```
did:example:21tDAKCERh95uGgKbJNHYP?resource=cbor
```

With a Path Component

The next set of examples are **DID URLs** that use the **resource** parameter and include a path component. In this case the target resource identified by the path component is relative to the resource identified by the DID itself. This pattern supports the ability of a **VDR** to organize digital resources in structures such as directory trees, collections, databases, and so on.

```
did:example:21tDAKCERh95uGgKbJNHYP/some/path?resource=true
```

```
did:example:21tDAKCERh95uGgKbJNHYP/some/longer/path?resource=json
```

```
did:example:21tDAKCERh95uGgKbJNHYP/uuid:33ad7beb-1abc-4a26-b892-466df4379a51/?resource=ld+json
```

```
did:example:21tDAKCERh95uGgKbJNHYP/resources/uuid:33ad7beb-1abc-4a26-b892-466df4379a51/?resource=cbor
```

DID Resolver and Verifiable Data Registry (VDR) Requirements

1. If a **DID URL** includes the **resource** parameter with a value of **true**, a conforming **DID resolver** MUST return the **default media type** of the digital resource identified by the **DID** from the **VDR** specified by the associated **DID method** provided such resource is available.
 - a. If the **DID resolver** is unable to return the identified resource, the resolver MUST return the error "Resource not found".
2. If a **DID URL** includes the **resource** parameter with a value of a valid **media type name**, a conforming **DID resolver** MUST return the requested **media type** of the digital resource identified by the **DID** from the **VDR** specified by the associated **DID method** provided such resource is available.
 - a. If the **DID resolver** is unable to return the identified resource in the requested media type, the resolver MUST return the error "Resource not found".
3. If a **DID URL** includes the **resource** parameter with a value of **false**, a conforming **DID resolver** SHOULD ignore the parameter.
4. If the **DID** alone is resolved **without** the **resource** parameter, it MUST return the authoritative **DID document** as defined in **W3C Decentralized Identifiers (DIDs) 1.0**. This specification adds no additional requirements to a conforming **DID document**.

DID Method Requirements

A **DID method** specification conforming to this specification to include support for the **resource** parameter:

1. MUST define:
 - a. How the associated **VDR** shall map the **DID** to the identified digital resource and the identified media type.
 - b. If the **DID URL** includes a path component, how the **VDR** shall process this path component in mapping to the identified resource.
 - c. If the **resource** parameter has a value of **true**, how the **VDR** shall determine the default **media type**.
 - d. How the **VDR** shall return the resource in response to a request from a conforming **DID resolver**.
2. MUST define how this parameter will interact with the following additional DID parameters defined in **section 3.2.1 of the DID Core Specification**.
 - a. **versionId**
 - i. If the **versionId** parameter is included in the **DID URL**, the resolver MUST return the identified version of the identified digital resource.
 - ii. If the identified version does not exist, the resolver MUST return the error: "Version not found".
 - b. **versionTime**
 - i. If the **versionTime** parameter is included in the **DID URL**, the resolver MUST return the identified version of the identified digital resource.
 - ii. If the identified version does not exist, the resolver MUST return the error: "Version not found".
3. SHOULD include in the associated **DID document** an **alsoKnownAs** property containing at least one value that is a **hashlink** to the identified digital resource. This provides additional cryptographic verification of the binding between the **DID** and the identified digital resource.
4. SHOULD NOT define any additional **DID document** properties—such as a special **service endpoint**—to enable dereferencing to the target digital resource. **This is an anti-pattern** because it requires DID controllers to add custom properties to their DID documents in order to use the **resource** parameter. A much better solution is to store the mapping between the **DID** and the identified digital resource within the **VDR** itself (and specify how this works in the **DID method** specification). This way the **VDR** can evolve its internal storage mechanisms without requiring **DID controllers** to make any changes to their **DID documents**.

Registration with the W3C DID Specification Registries

The **resource** parameter defined by this specification is registered with the **W3C DID Specification Registries 1.0** at the following URL:

<https://www.w3.org/TR/did-spec-registries/#resource>

Contributors

To comply with the intellectual property rights protections in the **charter of the ToIP Foundation** (as required by all Joint Development Foundation projects hosted the Linux Foundation), all contributors to this Pre-Draft Deliverable **MUST be current members of the ToIP Foundation**. The following contributors each certify that they meet this requirement:

- **Drummond Reed**, Evernym
- **Brent Zundel**, Evernym
- **Daniel Hardman**, SICPA
- **Stephen Curran**, Cloud Compass Computing

Acknowledgements

The authors wish to thank the editors and contributors to the **W3C Decentralized Identifiers (DIDs) 1.0 specification** and in particular co-editor Amy Guy for her help. Also thanks to the cheqd team of Ankur Banerjee and **Alex Tweeddale** for their suggestions about including support for the DID path component.

Licensing

This is a publicly available specification published by the ToIP Foundation under the following licenses:

- Copyright: **Creative Commons Attribution 4.0**.
- Patent: **Joint Development Foundation W3C Mode** (based on the **W3C Patent Policy**).

