

[MS-ODATAJSON]: OData Protocol JSON Format Standards Support Document

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft [Open Specification Promise](#) or the [Community Promise](#). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
06/11/2013	1.0	New	Released new document.
08/08/2013	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
12/05/2013	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2014	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
05/20/2014	2.0	Major	Significantly changed the technical content.

Contents

1 Introduction	5
1.1 Glossary	5
1.2 References	5
1.2.1 Normative References	5
1.2.2 Informative References	5
1.3 Microsoft Implementations	5
1.4 Standards Support Requirements	6
1.5 Notation	6
2 Standards Support Statements	7
2.1 Normative Variations	7
2.1.1 Section 3.1.1 odata.metadata=minimal	7
2.1.2 Section 3.1.2 odata.metadata=full	7
2.1.3 Section 3.1.3 odata.metadata=none	7
2.1.4 Section 3.2 Controlling the Representation of Numbers	7
2.1.5 Section 4.1 Header Content-Type	8
2.1.6 Section 4.3 Relative URLs	8
2.1.7 Section 4.4 Payload Ordering Constraints	8
2.1.8 Section 4.5 Control Information	9
2.1.9 Section 4.5.1 Annotation odata.context	9
2.1.9.1 [OData-Protocol] Section 10.3 Entity	9
2.1.9.2 [OData-Protocol] Section 10.7 Collection of Projected Entities	9
2.1.9.3 [OData-Protocol] Section 10.8 Projected Entity	10
2.1.9.4 [OData-Protocol] Section 10.9 Collection of Projected Expanded Entities	10
2.1.9.5 [OData-Protocol] Section 10.10 Projected Expanded Entity	11
2.1.9.6 [OData-Protocol] Section 10.11 Collection of Entity References	11
2.1.9.7 [OData-Protocol] Section 10.12 Entity Reference	11
2.1.9.8 [OData-Protocol] Section 10.13 Property Value	12
2.1.9.9 [OData-Protocol] Section 10.16 Operation Result	12
2.1.10 Section 4.5.2 Annotation odata.metadataEtag	12
2.1.11 Section 4.5.3 Annotation odata.type	12
2.1.12 Section 4.5.6 Annotation odata.deltaLink	13
2.1.13 Section 4.5.7 Annotation odata.id	13
2.1.14 Section 4.5.8 Annotation odata.editLink and odata.readLink	13
2.1.15 Section 4.5.10 Annotation odata.navigationLink and odata.associationLink	14
2.1.16 Section 5 Service Document	14
2.1.17 Section 7.1 Primitive Value	15
2.1.18 Section 8.1 Navigation Link	15
2.1.19 Section 8.2 Association Link	15
2.1.20 Section 8.3 Expanded Navigation Property	16
2.1.21 Section 11 Individual Property	16
2.1.22 Section 12 Collection of Entities	16
2.1.23 Section 13 Entity Reference	17
2.1.24 Section 14 Delta Response	17
2.1.25 Section 17 Action Invocation	17
2.1.26 Section 18.1 Annotate a JSON Object	17
2.1.27 Section 18.2 Annotate a JSON Array or Primitive	18
2.1.28 Section 19 Error Response	18
2.2 Clarifications	18
2.3 Error Handling	18

2.4 Security.....	18
3 Change Tracking.....	19
4 Index	23

1 Introduction

OData Protocol JSON Format Standards Support Document provides a statement of standards support for the Open Data (OData) protocol. This document describes the variations in the **JavaScript Object Notation (JSON)** format between the specified Microsoft implementations and the OASIS OData JSON format standards document.

The OData protocol comprises a set of specifications for representing and interacting with structured content. The core specifications for the OData protocol are specified in "Open Data Protocol (OData)" [\[MS-ODATA\]](#). The OASIS "OData JSON Format Version 4.0" [\[ODataJSON4.0\]](#) standards document is an extension of the OData core protocol and defines representations for the OData requests and responses by using a JSON format.

1.1 Glossary

The following terms are defined in [\[MS-ODATA\]](#):

JavaScript Object Notation (JSON) property

The following terms are specific to this document:

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as described in [\[RFC2119\]](#). All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[MS-ODATA] Microsoft Corporation, "[Open Data Protocol \(OData\)](#)".

[OData-Protocol] OASIS, "OData Version 4.0 Part 1: Protocol", OASIS Standard, February 2014, <http://docs.oasis-open.org/odata/odata/v4.0/os/part1-protocol/odata-v4.0-os-part1-protocol.doc>

[ODataJSON4.0] OASIS, "OData JSON Format Version 4.0", OASIS Standard, February 2014, <http://docs.oasis-open.org/odata/odata-json-format/v4.0/os/odata-json-format-v4.0-os.doc>

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

1.2.2 Informative References

None.

1.3 Microsoft Implementations

Windows Server 2012 R2 operating system

1.4 Standards Support Requirements

An OData implementation that is fully compliant with the OData standard implements all mandatory features and optionally implements any optional features.

[\[ODataJSON4.0\]](#) defines conformance for the OData protocol. This conformance is defined as support for all features in the document except the following:

- Features that are called out as optional.

The following table lists the sections of [\[ODataJSON4.0\]](#) that are considered normative and that are considered informative.

Section	Normative/Informative
1 and 2	Informative
3 through 22	Normative
Appendices A and B	Informative

1.5 Notation

The following notations are used to identify clarifications in section 2.2.

Notation	Explanation
C####	This notation identifies a clarification of ambiguity in the target specification. This includes imprecise statements, omitted information, discrepancies, and errata. This does not include data formatting clarifications.
V####	This notation identifies an intended point of variability in the target specification, such as the use of MAY, SHOULD, or RECOMMENDED. This does not include extensibility points.
E####	Because the use of extensibility points (such as optional implementation-specific data) could impair interoperability, this notation identifies such points in the target specification.

2 Standards Support Statements

2.1 Normative Variations

The following subsections detail the normative variations from [\[ODataJSON4.0\]](#).

2.1.1 Section 3.1.1 odata.metadata=minimal

The specification states the following:

```
The odata.metadata=minimal format parameter indicates that the service SHOULD remove
computable control information from the payload wherever possible.
```

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The parameter value is `odata=minimalmetadata`.

2.1.2 Section 3.1.2 odata.metadata=full

The specification states the following:

```
The odata.metadata=full format parameter indicates that the service MUST include all control
information explicitly in the payload.
```

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The parameter value is `odata=fullmetadata`.

2.1.3 Section 3.1.3 odata.metadata=none

The specification states the following:

```
The odata.metadata=none format parameter indicates that the service SHOULD omit control
information other than odata.nextLink and odata.count.
```

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The parameter value is `odata=nometadata`.

2.1.4 Section 3.2 Controlling the Representation of Numbers

The specification states the following:

```
The IEEE754Compatible=true format parameter indicates that the service MUST serialize
Edm.Int64 and Edm.Decimal numbers (including the odata.count, if requested) as strings. If
not specified, or specified as IEEE754Compatible=false, all numbers MUST be serialized as
JSON numbers.
```

OData in the specified Microsoft implementations ([section 1.3](#)) does not support this feature. The `IEEE754Compatible` format parameter is not used, and numbers that are `Edm.Int64` or `Edm.Decimal` are always represented as strings.

2.1.5 Section 4.1 Header Content-Type

The specification states the following:

```
Responses MUST include the odata.metadata parameter to specify the amount of metadata
included in the response.
```

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The format parameter is named "odata" and has values "minimalmetadata", "fullmetadata", and "nometadata".

Section 4.1 of the specification also states the following:

```
Responses MUST include the IEEE754Compatible parameter if Edm.Int64 and Edm.Decimal numbers
are represented as strings.
```

OData in the specified Microsoft implementations (section 1.3) does not support this feature. The IEEE754Compatible format parameter is never set on the response, although Edm.Int64 and Edm.Decimal numbers are always represented as strings.

Section 4.1 of the specification also states the following:

```
Requests and responses MAY add the odata.streaming parameter with a value of true or false,
see section Payload Ordering Constraints.
```

OData in the specified Microsoft implementations (section 1.3) partially supports this feature. The name of the type parameter is "streaming" rather than "odata.streaming". For normative variations of payload ordering constraints, see section [2.1.7](#).

2.1.6 Section 4.3 Relative URLs

The specification states the following:

```
Within the odata.type annotation, relative URLs are relative to the base type URL, which is
-- the odata.type of the enclosing object, if one exists, otherwise
-- the odata.type of the next enclosing object, if one exists, etc. until the document root,
otherwise
-- the context URL of the document root, if one exists, otherwise
-- the request URL.
```

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. URLs within the odata.type annotation are always relative to the metadata URL.

2.1.7 Section 4.4 Payload Ordering Constraints

The specification states the following:

```
Clients can request that a JSON response conform to these ordering constraints by specifying
a media type of application/json with the odata.streaming=true parameter in the Accept header
or $format query option.
```


OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The name of the type parameter is "streaming" rather than "odata.streaming".

2.1.8 Section 4.5 Control Information

The specification states the following:

```
Clients that encounter unknown annotations in any namespace, including the odata namespace,
MUST NOT stop processing and MUST NOT signal an error.
```

OData in the specified Microsoft implementations ([section 1.3](#)) does not support this feature. An error will be raised if an unknown annotation in the odata namespace is encountered.

2.1.9 Section 4.5.1 Annotation odata.context

The specification states the following:

```
The odata.context annotation returns the context URL (see [OData-Protocol]) for the payload.
This URL can be absolute or relative.
```

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The name of the odata.context annotation is "odata.metadata" and is written without the leading "@". The contents of the odata.metadata annotation differs from that of the odata.context annotation prescribed in [\[OData-Protocol\]](#) as described below in sections [2.1.9.1](#) through [2.1.9.9](#).

Section 4.5.1 of the specification also states the following:

```
The odata.context annotation MUST also be included for entities whose entity set cannot
be determined from the context URL of the collection.
```

OData in the specified Microsoft implementations ([section 1.3](#)) does not support this feature. Entity sets can always be determined from the metadata URL of the collection and MUST NOT be present for individual entities within a collection or collections within an entity.

2.1.9.1 [OData-Protocol] Section 10.3 Entity

The [\[OData-Protocol\]](#) specification states the following:

Context URL template:

```
{context-url}#{entity-set}/$entity
```

```
If a response or response part is a single entity of the declared type of an entity set,
/$entity is appended to the context URL.
```

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The metadata URL for a single entity is the metadata URL for the entity set with "/@Element" appended.

2.1.9.2 [OData-Protocol] Section 10.7 Collection of Projected Entities

The [\[OData-Protocol\]](#) specification states the following:

Context URL templates:

```
{context-url}#{entity-set}/{type-name}{select-list}
```

If a result contains only a subset of properties, the parenthesized comma-separated list of the selected defined or dynamic properties, navigation properties, functions, and actions is appended to the {entity-set} after an optional type-cast segment. The shortcut * represents the list of all structural properties. Properties defined on types derived from the declared type of the entity set (or type specified in the type-cast segment if specified) are prefixed with the qualified name of the derived type as defined in [OData-ABNF].

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The metadata URL for a collection of projected entities has the \$select query option from the request appended to the metadata URL that would have been returned for the collection of entities if no projection had been applied.

2.1.9.3 [OData-Protocol] Section 10.8 Projected Entity

The [\[OData-Protocol\]](#) specification states the following:

Context URL templates:

```
{context-url}#{entity-set}/{type-name}{select-list}/$entity  
{context-url}#{singleton}{select-list}
```

If a single entity contains a subset of properties, the parenthesized comma-separated list of the selected defined or dynamic properties, navigation properties, functions, and actions is appended to the {entity-set} after an optional type-cast segment and prior to appending /\$entity. The shortcut * represents the list of all structural properties. Properties defined on types derived from the type of the entity set (or type specified in the type-cast segment if specified) are prefixed with the qualified name of the derived type as defined in [OData-ABNF]. Note that expanded properties are implicitly selected.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The metadata URL for a projected entity has the \$select query option from the request appended to the metadata URL that would have been returned for the single entity or derived entity if no projection had been applied.

2.1.9.4 [OData-Protocol] Section 10.9 Collection of Projected Expanded Entities

The [\[OData-Protocol\]](#) specification states the following:

Context URL template:

```
{context-url}#{entity-set}/{type-name}{select-list}
```

If a navigation property is explicitly selected, the parenthesized comma-separated list of properties includes the name of the selected navigation property with no parenthesis. If a \$expand contains a nested \$select, the navigation property appears suffixed with the parenthesized comma-separated list of properties selected (or expanded, containing a \$select) from the related entities. Additionally, if the expansion is recursive for nested children, a plus sign (+) is infix between the navigation property name and the list of properties.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The metadata URL for a collection of projected expanded entities is identical to the metadata URL for a collection of projected entities with no expand.

2.1.9.5 [OData-Protocol] Section 10.10 Projected Expanded Entity

The [\[OData-Protocol\]](#) specification states the following:

Context URL template:

```
{context-url}#{entity-set}/{type-name}{select-list}/$entity  
{context-url}#{singleton}{select-list}
```

If a single entity is expanded and projected (or contains a \$expand with a \$select expand option), the parenthesized comma-separated list of selected properties includes the name of the expanded navigation properties containing a nested \$select, each suffixed with the parenthesized comma-separated list of properties selected (or expanded with a nested \$select) from the related entities.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The metadata URL for a projected expanded entity is identical to the metadata URL for a projected entity with no expand.

2.1.9.6 [OData-Protocol] Section 10.11 Collection of Entity References

The [\[OData-Protocol\]](#) specification states the following:

Context URL template:

```
{context-url}#Collection($ref)
```

If a response is a collection of entity references, the context URL does not contain the type of the referenced entities.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The metadata URL for a collection of links from an entity is the metadata URL for the entity set that contains that entity, followed by "\$links/" and the name of the navigation property for which links are being returned.

2.1.9.7 [OData-Protocol] Section 10.12 Entity Reference

The [\[OData-Protocol\]](#) specification states the following:

Context URL template:

```
{context-url}#$ref
```

If a response is a single entity reference, \$ref is the context URL fragment.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The metadata URL for a single link from an entity is the same as for a collection of links as defined in [section 2.1.9.6](#).

2.1.9.8 [OData-Protocol] Section 10.13 Property Value

The [\[OData-Protocol\]](#) specification states the following:

Context URL template:

```
{context-url}#{entity}/{property-path}
```

If a response represents an individual property, the context URL specifies the canonical URL of the entity and the path to the structural property of that entity. The path MUST include cast segments for properties defined on types derived from the expected type of the previous segment.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The metadata URL for an individual property is the same as the metadata URL for the primitive or complex type, or collection of primitive or complex types, of the property.

2.1.9.9 [OData-Protocol] Section 10.16 Operation Result

The [\[OData-Protocol\]](#) specification states the following:

Context URL templates:

```
{context-url}#{entity-set}/{type-name}{select-list}
{context-url}#{entity-set}/{type-name}{select-list}/$entity
{context-url}#{entity}/{property-path}
{context-url}#Collection({type-name})
{context-url}#{type-name}
```

If the response from an action or function is a collection of entities or a single entity that is a member of an entity set, the context URL identifies the entity set. If the response from an action or function is a property of a single entity, the context URL identifies the entity and property. Otherwise, the context URL identifies the type returned by the operation. The context URL will correspond to one of the former examples.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The metadata URL for an operation result is the same as the metadata URL for the primitive or complex type, or collection of primitive or complex types, returned by the operation.

2.1.10 Section 4.5.2 Annotation `odata.metadataEtag`

The specification states the following:

The `odata.metadataEtag` annotation MAY appear in a response in order to specify the entity tag (ETag) that can be used to determine the version of the metadata of the response.

OData in the specified Microsoft implementations ([section 1.3](#)) does not support this feature. Requests and responses never include the `odata.metadataEtag` annotation.

2.1.11 Section 4.5.3 Annotation `odata.type`

The specification states the following:

The `odata.type` annotation specifies the type of a JSON object or name/value pair. Its value is a URI that identifies the type of the property or object. For built-in primitive types the value is the unqualified name of the primitive type, specified as a URI fragment. For all other types, the URI may be absolute or relative to the `odata.type` of the containing object. The root `odata.type` may be absolute or relative to the root context URL.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. For built-in primitive types, the name of the type MAY be unqualified or qualified with the "Edm." prefix.

Section 4.5.3 of the specification also states the following:

For non-built in primitive types, the URI contains the namespace-qualified or alias-qualified type, specified as a URI fragment. For properties that represent a collection of values, the fragment is the namespace-qualified or alias-qualified element type enclosed in parentheses and prefixed with `Collection`.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. For non-built in primitive types, the value of the annotation is simply the namespace-qualified or alias-qualified type name, not specified as a fragment and not written as an absolute or relative URL.

2.1.12 Section 4.5.6 Annotation `odata.deltaLink`

The specification states the following:

The `odata.deltaLink` annotation contains a URL that can be used to retrieve changes to the current set of results.

OData in the specified Microsoft implementations ([section 1.3](#)) does not support this feature. Responses never include the `odata.deltaLink` annotation.

2.1.13 Section 4.5.7 Annotation `odata.id`

The specification states the following:

If the `odata.id` is represented, it MAY be a relative URL.

If the entity is transient (i.e. cannot be read or updated), the `odata.id` annotation MUST appear and have the null value.

OData in the specified Microsoft implementations ([section 1.3](#)) does not support this feature. The `odata.id` is never represented as a relative URL and never has the null value.

2.1.14 Section 4.5.8 Annotation `odata.editLink` and `odata.readLink`

The specification states the following:

The `odata.editLink` annotation contains the edit URL of the entity; see [OData-Protocol].

The `odata.readLink` annotation contains the read URL of the entity or collection; see [OData-Protocol].

. . .

For updatable entities:

--The `odata.editLink` annotation is written if `odata.metadata=full` is requested or if the edit URL differs from the default value of the edit URL.

--The `odata.readLink` annotation is written if the read URL is different from the edit URL. If no `odata.readLink` annotation is present, the read URL is identical to the edit URL.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The `editlink` is always written in full metadata, and the `readlink` is never written.

2.1.15 Section 4.5.10 Annotation `odata.navigationLink` and `odata.associationLink`

The specification states the following:

The `odata.navigationLink` annotation contains a navigation URL that can be used to retrieve an entity or collection of entities related to the current entity via a navigation property.

. . .

The `odata.associationLink` annotation contains an association URL that can be used to retrieve a reference to an entity or a collection of references to entities related to the current entity via a navigation property.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The annotations are `odata.navigationLinkUrl` and `odata.associationLinkUrl`.

2.1.16 Section 5 Service Document

The specification states the following:

Each element MUST be a JSON object with at least two name/value pairs, one with name `name` containing the name of the entity set, function import, or singleton, and one with name `url` containing the URL of the entity set, which may be an absolute or a relative URL. It MAY contain a name/value pair with name `title` containing a human-readable, language-dependent title for the object.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. Entries in the Service Document do not contain the title name/value pair.

Section 5 of the specification also states the following:

JSON objects representing an entity set MAY contain an additional name/value pair with name `kind` and a value of `EntitySet`. If the `kind` name/value pair is not present, the object MUST represent an entity set.

JSON objects representing a function import MUST contain the `kind` name/value pair with a value of `FunctionImport`.

JSON objects representing a singleton MUST contain the kind name/value pair with a value of Singleton.

JSON objects representing a related service document MUST contain the kind name/value pair with a value of ServiceDocument.

Clients that encounter unknown values of the kind name/value pair not defined in this version of the specification MUST NOT stop processing and MUST NOT signal an error.

OData in the specified Microsoft implementations (section 1.3) partially supports this feature. Only JSON objects representing an entity set are supported. The kind **property** is not supported on objects within the representation of a Service Document, and an error will be raised if such a property is present.

2.1.17 Section 7.1 Primitive Value

The specification states the following:

Values of type Edm.Binary, Edm.Date, Edm.DateTimeOffset, Edm.Duration, Edm.Guid, and Edm.TimeOfDay as well as enumeration values are represented as JSON strings whose content satisfies the rules binaryValue, dateValue, dateTimeOffsetValue, durationValue, guidValue, timeOfDayValue, and enumValue, respectively, in [OData-ABNF].

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The Date, Duration, and TimeOfDay primitive types, as well as enumeration values, are not supported. The following additional primitive types, which are not defined in the specification, are supported:

- Time
- Float
- DateTime

The serialization rules for these types are defined in [\[MS-ODATA\]](#). **Edm.Float** is defined in [MS-ODATA] section [2.2.2](#) to be the same as **Edm.Single**. Serialization rules for **Edm.Time**, **Edm.DateTime**, and **Edm.Single** are defined in [MS-ODATA] section [2.2.6.3.1](#).

2.1.18 Section 8.1 Navigation Link

The specification states the following:

The navigation link for a navigation property is represented as a name/value pair. The name is the name of the property, followed by @odata.navigationLink.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The name of the name/value pair is the name of the property followed by @odata.navigationLinkUrl.

2.1.19 Section 8.2 Association Link

The specification states the following:

The association link for a navigation property is represented as a name/value pair. The name

is the name of the property, followed by @odata.associationLink.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The name of the name/value pair is the name of the property followed by @odata.associationLinkUrl.

2.1.20 Section 8.3 Expanded Navigation Property

The specification states the following:

If a collection of entities can be related, it is represented as a JSON array. Each element is the representation of an entity or the representation of an entity reference.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. Expanded navigation properties never contain entity references.

Section 8.3 of the specification also states the following:

The navigation property MAY be annotated with odata.context, odata.count or odata.nextLink.

OData in the specified Microsoft implementations (section 1.3) partially supports this feature. Navigation properties never contain the odata.context or odata.count annotations.

2.1.21 Section 11 Individual Property

The specification states the following:

A single-valued property that has the null value does not have a representation; see [OData-Protocol].

OData in the specified Microsoft implementations ([section 1.3](#)) does not support this feature. Any request for an individual property with a null value is returned as a JSON object that contains an odata.metadata annotation whose value is the metadata URL with "/Edm.Null" appended, and a second name/value pair named "odata.null" that has a Boolean value of "True".

2.1.22 Section 12 Collection of Entities

The specification states the following:

A collection of entities is represented as a JSON object containing a name/value pair named value. It MAY contain odata.context, odata.count, odata.nextLink, or odata.deltaLink annotations.

If present, the odata.context annotation MUST be the first name/value pair in the response.

The odata.count name/value pair represents the number of entities in the collection. If present, it MUST come before the value name/value pair.

The value of the value name/value pair is a JSON array where each element is representation of an entity or a representation of an entity reference. An empty collection is represented as an empty JSON array.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. The collection of entities does not support the odata.deltaLink and MUST NOT contain entity references.

2.1.23 Section 13 Entity Reference

The specification states the following:

```
An entity reference (see [OData-Protocol]) MAY take the place of an entity instance in a JSON payload, based on the client request.
```

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. Entity references are not supported. Links are represented by using a name/value pair named "url", as shown in the following example:

```
{
  "url": http://host/service/Orders(10643)
}
```

2.1.24 Section 14 Delta Response

The specification states the following:

```
Responses from a delta request are returned as a JSON object.
```

OData in the specified Microsoft implementations ([section 1.3](#)) does not support the representation of Delta Responses.

2.1.25 Section 17 Action Invocation

The specification states the following:

```
Action parameter values are encoded in a single JSON object in the request body.
```

OData in the specified Microsoft implementations ([section 1.3](#)) does not support this feature.

2.1.26 Section 18.1 Annotate a JSON Object

The specification states the following:

```
When annotating a name/value pair for which the value is represented as a JSON object, each annotation is placed within the object and represented as a single name/value pair.
```

```
The name always starts with the "at" sign (@), followed by the namespace- or alias-qualified name of the annotation, i.e. the namespace or alias of the schema that defines the term, followed by a dot (.), followed by the name of the term.
```

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. Names never start with the "at" sign (@). The name is always just the namespace- or alias-qualified term name.

2.1.27 Section 18.2 Annotate a JSON Array or Primitive

The specification states the following:

When annotating a name/value pair for which the value is represented as a JSON array or primitive value, each annotation that applies to this name/value pair MUST be placed next to the annotated name/value pair and represented as a single name/value pair.

The name is the same as the name of the name/value pair being annotated, followed by the "at" sign (@), followed by the namespace- or alias-qualified name of the annotation, followed by a dot (.), followed by the name of the term.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. Names never start with the "at" sign (@). The name is always just the namespace- or alias-qualified term name.

2.1.28 Section 19 Error Response

The specification states the following:

This object MUST contain name/value pairs with the names code and message, and it MAY contain name/value pairs with the names target, details and innererror.

OData in the specified Microsoft implementations ([section 1.3](#)) partially supports this feature. Name/value pairs for target and details are not supported.

Section 19 of the specification also states the following:

The value for the message name/value pair MUST be a human-readable, language-dependent representation of the error. The Content-Language header MUST contain the language code from [RFC5646] corresponding to the language in which the value for message is written.

OData in the specified Microsoft implementations (section 1.3) partially supports this feature. The value for the message name/value pair is a JSON object that contains two name/value pairs: "lang" and "value". The value of the "lang" name/value pair indicates the language code for the message, and the value of the "value" name/value pair is the human-readable message written in that language.

2.2 Clarifications

None.

2.3 Error Handling

None.

2.4 Security

None.

3 Change Tracking

This section identifies changes that were made to the [MS-ODATAJSON] protocol document between the February 2014 and May 2014 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.

- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated**.

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.1 Glossary	Removed the term "resource" from the list of terms that are defined in [MS-ODATA].	Y	Content updated.
1.2.1 Normative References	Moved reference [MS-ODATA] from the Informative References section, added reference [OData-Protocol], and updated reference [ODataJSON4.0].	Y	Content updated.
1.2.2 Informative References	Moved reference [MS-ODATA] to the Normative References section.	Y	Content updated.
2.1.4 Section 3.2 Controlling the Representation of Numbers	Added section.	Y	New content added.
2.1.5 Section 4.1 Header Content-Type	Revised content for adopted OASIS standard.	Y	Content updated.
2.1.6 Section 4.3 Relative URLs	Revised content for adopted OASIS standard.	Y	Content updated.
2.1.7 Section 4.4 Payload Ordering Constraints	Added section.	Y	New content added.
2.1.8 Section 4.5 Control Information	Added section.	Y	New content added.
2.1.9 Section 4.5.1 Annotation odata.context	Revised title and content for adopted OASIS standard.	Y	Content updated.
2.1.9.1 [OData-Protocol] Section 10.3 Entity	Added section.	Y	New content added.
2.1.9.2	Added section.	Y	New

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
[OData-Protocol] Section 10.7 Collection of Projected Entities			content added.
2.1.9.3 [OData-Protocol] Section 10.8 Projected Entity	Added section.	Y	New content added.
2.1.9.4 [OData-Protocol] Section 10.9 Collection of Projected Expanded Entities	Added section.	Y	New content added.
2.1.9.5 [OData-Protocol] Section 10.10 Projected Expanded Entity	Added section.	Y	New content added.
2.1.9.6 [OData-Protocol] Section 10.11 Collection of Entity References	Added section.	Y	New content added.
2.1.9.7 [OData-Protocol] Section 10.12 Entity Reference	Added section.	Y	New content added.
2.1.9.8 [OData-Protocol] Section 10.13 Property Value	Added section.	Y	New content added.
2.1.9.9 [OData-Protocol] Section 10.16 Operation Result	Added section.	Y	New content added.
2.1.10 Section 4.5.2 Annotation odata.metadataEtag	Added section.	Y	New content added.
2.1.11 Section 4.5.3 Annotation odata.type	Revised content for adopted OASIS standard.	Y	Content updated.
2.1.12 Section 4.5.6 Annotation odata.deltaLink	Clarified that responses never include the odata.deltaLink annotation.	Y	Content updated.
2.1.13 Section 4.5.7 Annotation odata.id	Added section.	Y	New content added.
2.1.14 Section 4.5.8 Annotation odata.editLink and odata.readLink	Updated quote from OASIS standard.	Y	Content updated.
2.1.17 Section 7.1 Primitive Value	Added references for serialization rules defined in [MS-ODATA].	Y	Content updated.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
2.1.18 Section 8.1 Navigation Link	Added section.	Y	New content added.
2.1.19 Section 8.2 Association Link	Added section.	Y	New content added.
2.1.20 Section 8.3 Expanded Navigation Property	Added section.	Y	New content added.
2.1.21 Section 11 Individual Property	Added section.	Y	New content added.
2.1.22 Section 12 Collection of Entities	Added section.	Y	New content added.
2.1.26 Section 18.1 Annotate a JSON Object	Added section.	Y	New content added.
2.1.27 Section 18.2 Annotate a JSON Array or Primitive	Added section.	Y	New content added.
2.1.28 Section 19 Error Response	Added discussion of requirements for the value of the message name/value pair.	Y	Content updated.
2.1.5 Section 4.2 Message Body	Removed section.	Y	Content removed.
2.1.11 Section 4.5.9 Annotation odata.kind	Removed section.	Y	Content removed.

4 Index

C

[Change tracking](#) 19

[Clarifications](#) 18

E

[Error handling](#) 18

G

[Glossary](#) 5

I

[Implementations](#) 5

[Informative references](#) 5

[Introduction](#) 5

M

[Microsoft implementations](#) 5

N

[Normative references](#) 5

[Normative variations](#) 7

[Notation](#) 6

R

References

[informative](#) 5

[normative](#) 5

S

[Security](#) 18

[Standards support requirements](#) 6

T

[Tracking changes](#) 19

V

Variations

[normative](#) 7