[MS-PROPSTORE]:
Property Store Binary File Format

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
07/16/2010	1.0	New	First Release.
08/27/2010	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
03/25/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
05/06/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
06/17/2011	1.1	Minor	Clarified the meaning of the technical content.
09/23/2011	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
12/16/2011	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
03/30/2012	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
07/12/2012	1.2	Minor	Clarified the meaning of the technical content.
10/25/2012	1.2	No change	No changes to the meaning, language, or formatting of the technical content.
01/31/2013	1.2	No change	No changes to the meaning, language, or formatting of the technical content.
08/08/2013	2.0	Major	Significantly changed the technical content.

Contents

1 Introduction

This document specifies the Microsoft Property Store Binary File Format. This file format is a persistence format for a set of properties. Implementers can use this file format to store a set of properties in a file or within another structure.

Sections 1.7 and 2 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

globally unique identifier (GUID)
little-endian
Unicode

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

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

A reference marked "(Archived)" means that the reference document was either retired and is no longer being maintained or was replaced with a new document that provides current implementation details. We archive our documents online [Windows Protocol].

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. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-DTYP] Microsoft Corporation, "Windows Data Types".

[MS-SHLLINK] Microsoft Corporation, "Shell Link (.LNK) Binary File Format".

[MS-OLEPS] Microsoft Corporation, "Object Linking and Embedding (OLE) Property Set Data Structures".

[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

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

1.3 Overview

This structure provides a compact way to serialize one or more property sets. Each property set consists of a property set identifier and one or more property values. Each property value consists of a unique property name and an associated value. Each property name can be either an unsigned integer or, in the case of a special property set identifier, a **Unicode** string.

This structure does not specify the semantics of properties or the assignment of property set identifiers or property names.

Data in this file format is stored in **little-endian** format.

1.4 Relationship to Protocols and Other Structures

This structure is used by the Shell Link (.LNK) Binary File Format, as specified in [MS-SHLLINK].

1.5 Applicability Statement

This document specifies a persistence format for one or more sets of property identifiers and associated property values. This persistence format is applicable when each property set can be identified by a **globally unique identifier (GUID)**, and when each property within a property set can be identified by an unsigned integer or a Unicode string name and can be persisted as a TypedPropertyValue structure, as specified in [MS-OLEPS] section 2.15.

1.6 Versioning and Localization

None.

1.7 Vendor-Extensible Fields

Implementers are free to define new **Format ID**s within the Serialized Property Storage structure, as defined in section 2.2, and to define new property identifiers within a Serialized Property Value structure, as defined in section 2.3.

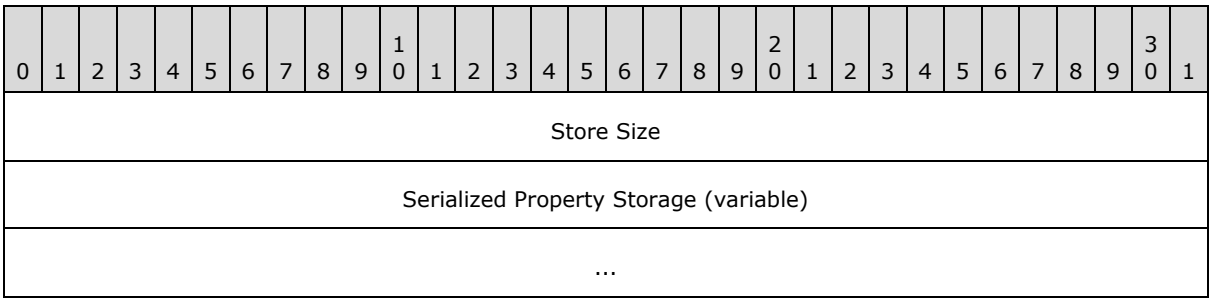
2 Structures

This document references commonly used data types as defined in [MS-DTYP].

Unless otherwise qualified, instances of **GUID** in this section refer to [MS-DTYP] section 2.3.4.

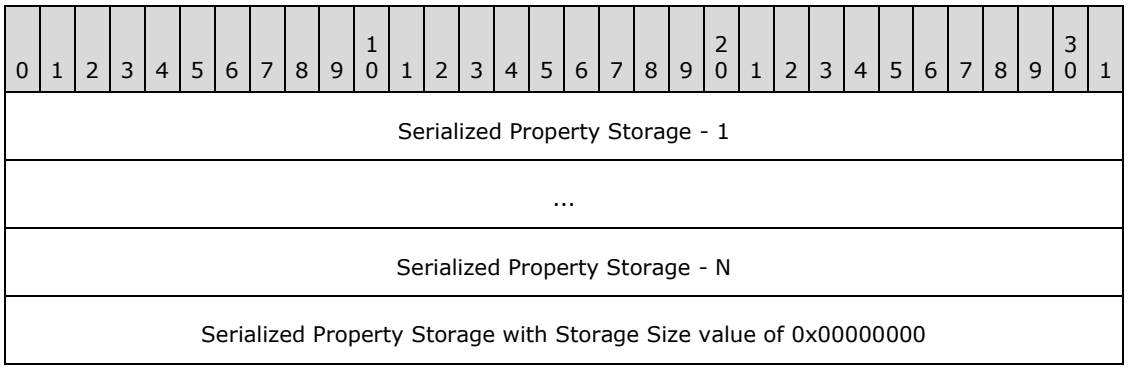
2.1 Serialized Property Store

The Property Store Binary File Format is a sequence of Serialized Property Storage structures. The sequence MUST be terminated by a Serialized Property Storage structure that specifies 0x00000000 for the **Storage Size** field.



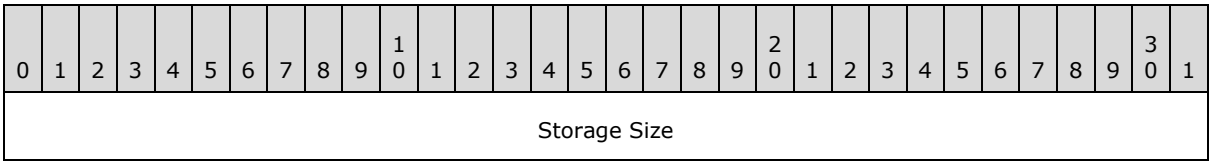
Store Size (4 bytes): An unsigned integer that specifies the total size, in bytes, of this structure, excluding the size of this field.

Serialized Property Storage (variable): A sequence of one or more Serialized Property Storage structures, as specified in section 2.2.



2.2 Serialized Property Storage

The Serialized Property Storage structure is a sequence of Serialized Property Value structures. The sequence MUST be terminated by a Serialized Property Value structure that specifies 0x00000000 for the **Value Size** field.



Version
Format ID
...
...
...
Serialized Property Value (variable)
...

Storage Size (4 bytes): An unsigned integer that specifies the total size, in bytes, of this structure. It MUST be 0x00000000 if this is the last Serialized Property Storage in the enclosing Serialized Property Store.

Version (4 bytes): MUST be equal to 0x53505331.

Format ID (16 bytes): A GUID that specifies the semantics and expected usage of the properties contained in this Serialized Property Storage structure. It MUST be unique in the set of serialized property storage structures.

Serialized Property Value (variable): A sequence of one or more property values. If the **Format ID** field is equal to the GUID {D5CDD505-2E9C-101B-9397-08002B2CF9AE}, then all values in the sequence MUST be Serialized Property Value (String Name) structures, as specified in section 2.3.1; otherwise, all values MUST be Serialized Property Value (Integer Name) structures, as specified in section 2.3.2. The last Serialized Property Value in the sequence MUST specify 0x00000 for the **Value Size**.

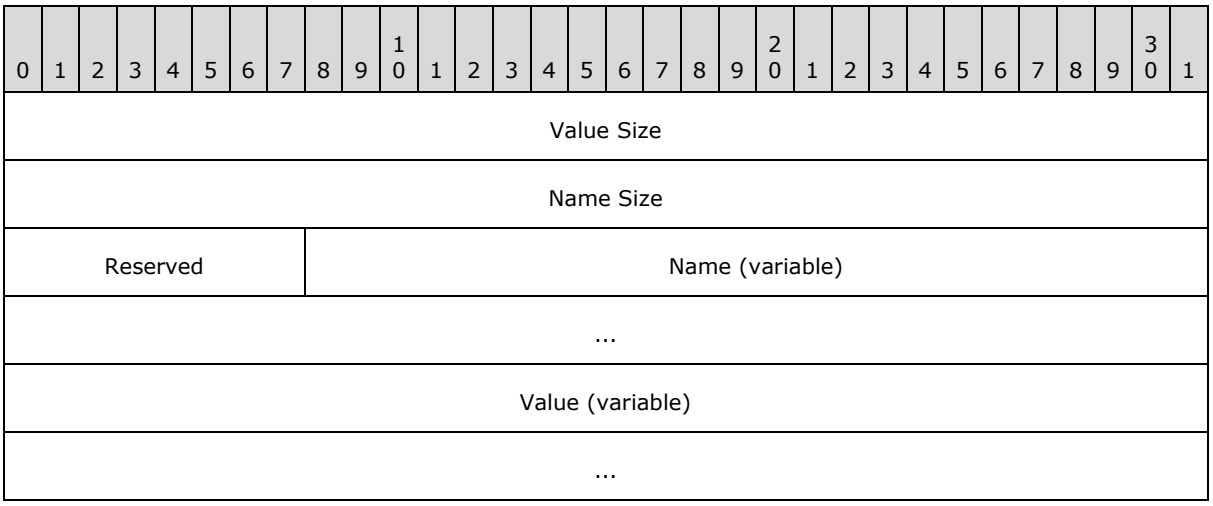
0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
Serialized Property Value - 1																															
...																															
Serialized Property Value - N																															
Serialized Property Value with Value Size of 0x00000000																															

2.3 Serialized Property Value

There are two types of Serialized Property Value structures: Serialized Property Value (String Name) structures and Serialized Property Value (Integer Name) structures.

2.3.1 Serialized Property Value (String Name)

The Serialized Property Value (String Name) structure specifies a single property within a Serialized Property Storage structure, where the property is identified by a unique Unicode string.



Value Size (4 bytes): An unsigned integer that specifies the total size, in bytes, of this structure. It MUST be 0x00000000 if this is the last The Serialized Property Value in the enclosing Serialized Property Storage structure.

Name Size (4 bytes): An unsigned integer that specifies the size, in bytes, of the **Name** field, including the null-terminating character.

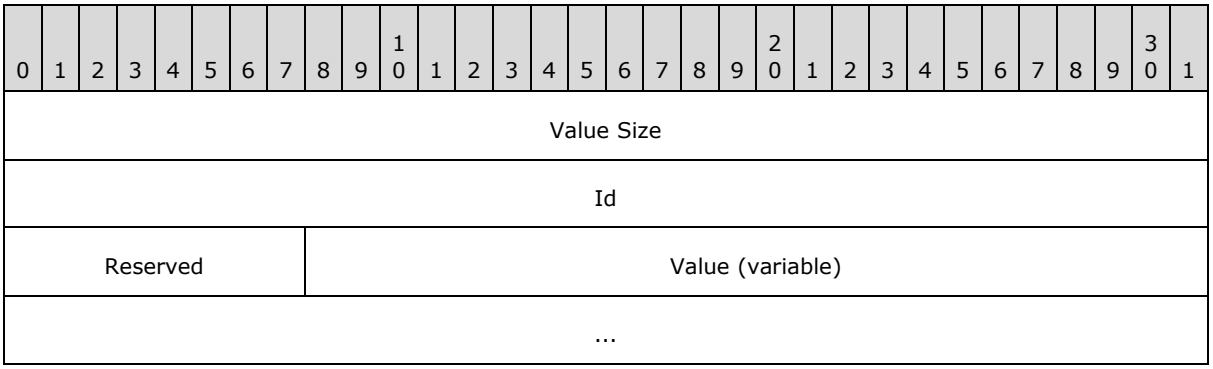
Reserved (1 byte): MUST be 0x00.

Name (variable): A null-terminated Unicode string that specifies the identity of the property. It MUST be unique within the enclosing Serialized Property Storage structure.

Value (variable): A TypedPropertyValue structure, as specified in [MS-OLEPS] section 2.15.

2.3.2 Serialized Property Value (Integer Name)

The Serialized Property Value (Integer Name) structure specifies a single property within a Serialized Property Storage structure, where the property is identified by a unique unsigned integer.



Value Size (4 bytes): An unsigned integer that specifies the total size, in bytes, of this structure. It MUST be 0x00000000 if this is the last Serialized Property Value in the enclosing Serialized Property Storage structure.

Id (4 bytes): An unsigned integer that specifies the identity of the property. It MUST be unique within the enclosing Serialized Property Storage structure.

Reserved (1 byte): MUST be 0x00.

Value (variable): A TypedPropertyValue structure, as specified in [MS-OLEPS] section 2.15.

3 Structure Examples

None.

4 Security Considerations

None.

5 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Windows Vista operating system
- Windows Server 2008 operating system
- Windows 7 operating system
- Windows Server 2008 R2 operating system
- Windows 8 operating system
- Windows Server 2012 operating system
- Windows 8.1 operating system
- Windows Server 2012 R2 operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

6 Change Tracking

This section identifies changes that were made to the [MS-PROPSTORE] protocol document between the January 2013 and August 2013 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.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

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 language and 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 or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

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.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- 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 protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
5 Appendix A: Product Behavior	Modified this section to include references to Windows 8.1 operating system and Windows Server 2012 R2 operating system.	Y	Content updated.

7 Index