[MS-HTTPE-Diff]:

Hypertext Transfer Protocol (HTTP) Extensions

Revision Summary

Date	Revision History	Revision Class	Comments
11/14/2013	1.0	New	Released new document.
2/13/2014	2.0	Major	Updated and revised the technical content.
5/15/2014	2.0	None	No changes to the meaning, language, or formatting of the technical content.
6/30/2015	3.0	Major	Significantly changed the technical content.
10/16/2015	3.0	None	No changes to the meaning, language, or formatting of the technical content.
7/14/2016	3.0	None	No changes to the meaning, language, or formatting of the technical content.
3/16/2017	4.0	Major	Significantly changed the technical content.
6/1/2017	4.0	None	No changes to the meaning, language, or formatting of the technical content.
9/15/2017	5.0	Major	Significantly changed the technical content.
12/1/2017	5.0	None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

The Hypertext Transfer Protocol (HTTP) Extensions Protocol specifies a set of extensions to Hypertext Transfer Protocol (HTTP) dealing with the internationalization of host names and query strings.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

American National Standards Institute (ANSI) character set: A character set defined by a code page approved by the American National Standards Institute (ANSI). The term "ANSI" as used to signify Windows code pages is a historical reference and a misnomer that persists in the Windows community. The source of this misnomer stems from the fact that the Windows code page 1252 was originally based on an ANSI draft, which became International Organization for Standardization (ISO) Standard 8859-1 [ISO/IEC-8859-1]. In Windows, the ANSI character set can be any of the following code pages: 1252, 1250, 1251, 1253, 1254, 1255, 1256, 1257, 1258, 874, 932, 936, 949, or 950. For example, "ANSI application" is usually a reference to a non-Unicode or code-page-based application. Therefore, "ANSI character set" is often misused to refer to one of the character sets defined by a Windows code page that can be used as an active system code page; for example, character sets defined by code page 1252 or character sets defined by code page 950. Windows is now based on Unicode, so the use of ANSI character sets is strongly discouraged unless they are used to interoperate with legacy applications or legacy data.

ASCII: The American Standard Code for Information Interchange (ASCII) is an 8-bit character-encoding scheme based on the English alphabet. ASCII codes represent text in computers, communications equipment, and other devices that work with text. ASCII refers to a single 8-bit ASCII character or an array of 8-bit ASCII characters with the high bit of each character set to zero.

Augmented Backus-Naur Form (ABNF): A modified version of Backus-Naur Form (BNF), commonly used by Internet specifications. ABNF notation balances compactness and simplicity with reasonable representational power. ABNF differs from standard BNF in its definitions and uses of naming rules, repetition, alternatives, order-independence, and value ranges. For more information, see [RFC5234].

code page: An ordered set of characters of a specific script in which a numerical index (code-point value) is associated with each character. Code pages are a means of providing support for character sets and keyboard layouts used in different countries. Devices such as the display and keyboard can be configured to use a specific code page and to switch from one code page (such as the United States) to another (such as Portugal) at the user's request.

host name: The name of a host on a network that is used for identification and access purposes by humans and other computers on the network.

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol Secure (HTTPS): An extension of HTTP that securely encrypts and decrypts web page requests. In some older protocols, "Hypertext Transfer Protocol over Secure Sockets Layer" is still used (Secure Sockets Layer has been deprecated). For more information, see [SSL3] and [RFC5246].

Internationalized Domain Names for Applications (IDNA): An encoding process that transforms a string of Unicode characters into a smaller, restricted character set. IDNA encoding is commonly used for creating domain names that can be represented in the ASCII character set that is supported in the Domain Name System (DNS) of the Internet. IDNA uses the Punycode algorithm [RFC3492] and ACE (ASCII-compatible encoding) prefix [RFC5890] for the transformation.

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [RFC3986].

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [RFC1738].

UTF-8: A byte-oriented standard for encoding Unicode characters, defined in the Unicode standard. Unless specified otherwise, this term refers to the UTF-8 encoding form specified in [UNICODE5.0.0/2007] section 3.9.

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

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

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-UCODEREF] Microsoft Corporation, "Windows Protocols Unicode Reference".

[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

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.rfc-editor.org/rfc/rfc2616.txt

[RFC3629] Yergeau, F., "UTF-8, A Transformation Format of ISO 10646", STD 63, RFC 3629, November 2003, http://www.ietf.org/rfc/rfc3629.txt

[RFC3986] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005, http://www.rfc-editor.org/rfc/rfc3986.txt

[RFC5234] Crocker, D., Ed., and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008, http://www.rfc-editor.org/rfc/rfc5234.txt

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, August 2010, http://rfc-editor.org/rfc/rfc5890.txt

[TR46] Davis, M., and Suignard, M., "Unicode IDNA Compatibility Processing", Unicode Technical Standard #46, September 2012, "", http://www.unicode.org/reports/tr46/

1.2.2 Informative References

[ISO/IEC-8859-1] International Organization for Standardization, "Information Technology -- 8-Bit Single-Byte Coded Graphic Character Sets -- Part 1: Latin Alphabet No. 1", ISO/IEC 8859-1, 1998, http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=28245

Note There is a charge to download the specification.

[MS-NETOD] Microsoft Corporation, "Microsoft .NET Framework Protocols Overview".

[RFC2396] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998, http://www.rfc-editor.org/rfc/rfc2396.txt

[RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011, http://www.rfc-editor.org/rfc/rfc6066.txt

[RFC6943] Thaler, D., Ed., "Issues in Identifier Comparison for Security Purposes", RFC 6943, May 2013, http://www.rfc-editor.org/rfc/rfc6943.txt

1.3 Overview

This document specifies a set of extensions to the Hypertext Transfer Protocol (HTTP) [RFC2616] dealing with internationalization of host names, with query strings, and with the path syntax.

Originally, the HTTP protocol was defined only in terms of the ASCII character set. However, there quickly became a demand to support languages other than English. Among other things, this notably affected two types of data. First, it affected strings passed in the query component of a Uniform Resource Locator (URL) (later called a Uniform Resource Identifier (URI) in [RFC3986]) for use in fields in forms, doing lookups in search engines, and so on. Second, a demand arose to give servers names in the native language, thus resulting in an internationalized host name.

A mechanism known as Internationalized Domain Names for Applications (IDNA), for supporting internationalized host names in protocols defined for ASCII, was later standardized and is now specified in [RFC5890] and [TR46]. In the meantime, the extensions in this document were already in use, which include:

- The transport of query string parameters in URIs without being percent-encoded.
- The use of characters in the HTTP Host header without being limited to the ASCII subset of characters, as opposed to requiring IDNA encoding to get an ASCII string to include.

A second extension is that the syntax of the path component of a URI is extended to allow square brackets "[" and "]" without being percent encoded.

1.4 Relationship to Other Protocols

This document specifies extensions to HTTP, and retains the same relationships to other protocols as the base HTTP protocol does. For encoding formats, the query extension in this document is an alternative to the encoding format specified in [RFC2616] and the Host header extension in this document is an alternative to the IDNA encoding format.

1.5 Prerequisites/Preconditions

These extensions assume that the client and the server have both been configured to use the same code page.

1.6 Applicability Statement

The extensions in this document are applicable only to environments where clients and servers all use the same code page. Furthermore, they are also applicable only to environments where either no HTTP proxy is present between the client and the server, or any HTTP proxies support the more liberal URI syntax defined in this document.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

Messages are transported as specified in [RFC2616].

2.2 Message Syntax

The syntax is as specified in [RFC2616], except as follows.

2.2.1 Request-URI

The URI requested appears in the Request-URI field as specified in [RFC2616] section 5.1.2. It used the syntax restrictions in [RFC2396], which specifies that the query component of a URI can use only the ASCII subset of characters and requires other characters to be percent-escaped as specified in [RFC2396] section 2.4.1.

Note Although [RFC2396] was later obsoleted by [RFC3986], that restriction is unchanged. Specifically, [RFC3986] section 3.4 states:

```
query       = *( pchar / "/" / "?" )
```

This specification extends the Augmented Backus-Naur Form (ABNF) [RFC5234] for the query portion of the Request-URI field as follows:

```
query       = *(<any CHAR except CTLs or "#">)
```

Furthermore, [RFC3986] section 3.3 specifies the syntax of the path component and states:

```
pchar       = unreserved / pct-encoded / sub-delims / ":" / "@"
```

This specification extends this syntax to allow the "[" and "]" characters as follows:

```
pchar       = unreserved / pct-encoded / sub-delims / ":" / "@" / "[" / "]"
```

2.2.2 Host Header

HTTP is defined in [RFC2616] as using text encoded in ISO-8859-1 [ISO/IEC-8859-1]. The Host header is specified in [RFC2616] section 14.23 with a more restricted syntax, however. It uses the syntax restrictions specified in [RFC2396], which specifies that the Host header value can use only a limited set of characters, all within the ASCII character set. When using HTTPS and the server name indication extension specified in [RFC6066], the host name specified in the Host header SHOULD<1> match the host name specified in the server name indication extension.

This specification extends the Host header syntax to permit the value to be encoded in UTF-8 [RFC3629] or in the client's code page rather than requiring the use of IDNA to generate an ASCII string. This means that characters might be encoded using octets that are not allowed in ISO-8859-1.

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

CodePage: The American National Standards Institute (ANSI) code page that the client is configured to use. See [MS-UCODEREF] section 2.2.1 for more details.

3.1.2 Timers

None beyond what is specified in [RFC2616].

3.1.3 Initialization

None beyond what is specified in [RFC2616].

3.1.4 Higher-Layer Triggered Events

3.1.4.1 Sending an HTTP Request

When a higher-layer protocol or application requests the content for a given URI, the HTTP implementation MUST construct the HTTP request as specified in [RFC2616], except as follows.

Characters not legal in the standard query syntax SHOULD<2> be escaped as described in [RFC2396] section 2.4.1 but MAY instead be encoded (unescaped) in the configured **CodePage** as specified in [MS-UCODEREF] section 3.1.5.1.1.2.

Characters that are not legal in the standard Host header syntax SHOULD be encoded by using the IDNA algorithm as specified in [RFC5890] and [TR46], but MAY<3> instead be encoded in UTF-8 [RFC3629] or encoded in the configured **CodePage** as specified in [MS-UCODEREF] section 3.1.5.1.1.2.

3.1.5 Message Processing Events and Sequencing Rules

None beyond what is specified in [RFC2616].

3.1.6 Timer Events

None beyond what is specified in [RFC2616].

3.1.7 Other Local Events

None beyond what is specified in [RFC2616].

3.2 Server Details

3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

CodePage: The ANSI code page that the server is configured to use. See [MS-UCODEREF] section 2.2.1 for more details.

3.2.2 Timers

None beyond what is specified in [RFC2616].

3.2.3 Initialization

None beyond what is specified in [RFC2616].

3.2.4 Higher-Layer Triggered Events

None beyond what is specified in [RFC2616].

3.2.5 Message Processing Events and Sequencing Rules

3.2.5.1 Receiving an HTTP Request

When an HTTP Request message is received, the HTTP server MUST validate it and process it as specified in [RFC2616], except as follows.

If the Request-URI contains a query component that does not conform to the standard syntax, but does conform to the extended syntax specified in section 2.2.1, the server MUST interpret it as being encoded in the server's **CodePage** (see [MS-UCODEREF] section 3.1.5.1.1.3 for details). If the Request-URI contains a query component containing a percent (%) character, the server SHOULD interpret it as being escaped as specified in [RFC2396] section 2.4.1. The server MAY<4> instead interpret it literally; that is, where the percent character represents itself rather than indicating the beginning of an escape sequence.

The Host header MUST be validated using the extended syntax specified in section 2.2.2. If the value contains characters that would not be valid in the standard syntax, the server SHOULD interpret it as follows: attempt to interpret it as UTF-8 and if it is not a valid UTF-8 sequence, then interpret it in as being encoded in the server's **CodePage** (see [MS-UCODEREF] section 3.1.5.1.1.3 for details). The server MAY<5> instead reverse the order of checks; that is, first attempt to interpret it as being encoded in the server's **CodePage** and if it is not a valid string in that **CodePage**, then interpret it as being encoded in UTF-8.

3.2.6 Timer Events

None beyond what is specified in [RFC2616].

3.2.7 Other Local Events

None beyond what is specified in [RFC2616].

4 Protocol Examples

In the following examples, the client and server are both configured to use the ANSI Baltic code page (code page 1257), and an application requests that "http://bønne.contoso.com/path?søster" be retrieved. Both the host component and the query component of this URI contain a LATIN SMALL LETTER O WITH STROKE ("ø" which is Unicode U+00F8). Note that in neither example does it appear encoded in ISO-8859-1 (that is, with octet value 248 = 0xF8) as HTTP would normally require for all headers.

4.1 Request Sent Through a Proxy

In this example, the request is being sent through a proxy, so the Request-URI includes a host name. Since the host and query portions of the URI both contain a non-ASCII character, the client has chosen to use the extended syntax, and the HTTP request appears as follows (possibly along with other HTTP headers).

```
GET http://xn--bnne-gra.contoso.com/?s%C3%B8ster HTTP/1.1
Host: bønne.contoso.com
```

In this request, the LATIN SMALL LETTER O WITH STROKE is encoded as follows:

- In the host component of the URI in the request line, it appears in the host name's IDNA form (xn--bnne-gra) as in normal HTTP without any extensions.
- In the query component of the URI in the request line, it appears in the escaped form of the UTF-8 encoding (U+00F8 encoded in UTF-8 is 0xC3 0xB8) as in normal HTTP without any extensions.
- In the Host header, the client chooses to use the ANSI Baltic code page (octet value 184 = 0xB8) instead of the IDNA form. As such, other HTTP utilities might misinterpret the "ø" as being (in ISO-8859-1) a CEDILLA ("¸" which is Unicode U+00B8) and display it as "b¸nne.contoso.com".

4.2 Request Not Sent Through a Proxy

In this example, the request is not sent through a proxy, so the Request-URI does not contain a host name. Since the host and query portions of the URI both contain a non-ASCII character, the client has chosen to use the extended syntax, and the HTTP request appears as follows (possibly along with other HTTP headers).

```
GET /?søster HTTP/1.1
Host: bønne.contoso.com
```

In this request, the LATIN SMALL LETTER O WITH STROKE is encoded as follows:

- In the query component of the URI in the request line, it appears encoded in the ANSI Baltic code page (octet value 184 = 0xB8). As such, other HTTP utilities might misinterpret the "ø" as being (in ISO-8859-1) a CEDILLA ("¸" which is Unicode U+00B8) and display it as "s¸ster".
- In the Host header, the client chooses to use UTF-8 and encodes the "ø" as 0xC3 0xB8. As such, other HTTP utilities might misinterpret the "ø" as being (in ISO-8859-1) a LATIN CAPITAL LETTER A WITH TILDE ("Ã" which is Unicode U+00C3) followed by CEDILLA ("¸" which is Unicode U+00B8) and display it as "bÃ¸nne.contoso.com".

5 Security

5.1 Security Considerations for Implementers

Security considerations are discussed in [RFC2616] section 15. Since the query component and the Host header are often used for comparison against expected strings, and since the extensions in this document allow additional ways to encode the same strings, take care to ensure that matching algorithms operate correctly, typically by normalizing a string to some common form before comparison. For further discussion of security considerations for comparison algorithms, see [RFC6943].

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

This document specifies version-specific details in the Microsoft .NET Framework. For information about which versions of .NET Framework are available in each released Windows product or as supplemental software, see [MS-NETOD] section 4.

- Microsoft .NET Framework 2.0
- Microsoft .NET Framework 3.5
- Microsoft .NET Framework 4.0
- Microsoft .NET Framework 4.5
- Microsoft .NET Framework 4.6
- Microsoft .NET Framework 4.7
- Windows 2000 operating system
- Windows XP operating system
- Windows Server 2003 operating system
- 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
- Windows 10 operating system
- Windows Server 2016 operating system
- Windows Server operating system

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates 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.

<1> Section 2.2.2: Except in Windows 10, servers in applicable Windows Server releases that are configured to select certificates using server name indication (SNI) returned a status code of 400 to a

request in which the Host header did not match the host name specified in the SNI extension. Servers configured to select certificates based on the network interface receiving the request do not require that the Host header match the SNI extension.

<2> Section 3.1.4.1: In applicable Windows releases, the ANSI code page with the extended query syntax is used.

<3> Section 3.1.4.1: In applicable Windows releases, UTF-8 is used by default for the Host header when sending to destinations in the Intranet zone. In the .NET Framework, the ANSI code page for the Host header value is used by default. When the <idn> configuration flag is enabled, the .NET Framework uses the IDNA form.

<4> Section 3.2.5.1: Windows does not decode the sequence but returns the string directly to the higher-layer protocol or application. It is thus the responsibility of the higher-layer protocol or application to determine how to interpret the string.

<5> Section 3.2.5.1: Windows allows the order to be configured.

7 Change Tracking

~~This section identifies~~ No table of changes ~~that were made to this~~ is available. The document is either new or has had no changes since ~~the~~its last release. ~~Changes are classified as Major, Minor, or None.~~

~~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.~~
- ~~A document revision that captures changes to protocol functionality.~~

~~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 **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.~~

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

~~**Section**~~	~~**Description**~~	~~**Revision class**~~
~~6 Appendix A: Product Behavior~~	~~Added Windows Server to the applicable products list.~~	~~Major~~

8 Index

V