ETSI ES 201 873-11 V4.8.1 (2018-05)

Methods for Testing and Specification (MTS);

The Testing and Test Control Notation version 3;

Part 11: Using JSON with TTCN-3

**ETSI Standard**

Reference

RES/MTS-201873-11ed481

Keywords

JSON, language, testing, TTCN-3

***ETSI***

650 Route des Lucioles

F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C

Association à but non lucratif enregistrée à la

Sous-Préfecture de Grasse (06) N° 7803/88

***Important notice***

The present document can be downloaded from:  
<http://www.etsi.org/standards-search>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the only prevailing document is the print of the Portable Document Format (PDF) version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at <https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>

If you find errors in the present document, please send your comment to one of the following services:  
<https://portal.etsi.org/People/CommiteeSupportStaff.aspx>

***Copyright Notification***

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.  
The content of the PDF version shall not be modified without the written authorization of ETSI.  
The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2018.

All rights reserved.

**DECT**TM, **PLUGTESTS**TM, **UMTS**TM and the ETSI logo are trademarks of ETSI registered for the benefit of its Members.  
**3GPP**TM and **LTE**TM are trademarks of ETSI registered for the benefit of its Members and  
of the 3GPP Organizational Partners.  
**oneM2M** logo is protected for the benefit of its Members.  
**GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

### 6.4.2 JSON Strings

A JSON string is a sequence of zero or more Unicode characters, enclosed in a pair of quotation mark characters (""", **char**(U22)). Any characters may be escaped by the escape sequence: "\u<HHHH>", where <HHHH> represents four hexadecimal digits, but the icharacters: quotation mark (""", **char**(U22)), reverse solidus ("\", **char**(U5C)) and all C0 control characters (**char**(U0) through **char**(U1F)) shall be escaped.

Alternatively, the short, two-character escape sequences defined in table 1 can be used to escape some of the characters.

Table 1: Short character escape sequences

|  |  |  |
| --- | --- | --- |
| Character's name | Character code | Short escape sequence |
| quotation mark | **char**(U22) | \" |
| reverse solidus | **char**(U5C) | \\ |
| solidus | **char**(U2F) | \/ |
| backspace | **char**(U8) | \b |
| form feed | **char**(UC) | \f |
| line feed | **char**(UA) | \n |
| carriage return | **char**(UD) | \r |
| horizontal tab | **char**(U9) | \t |

By default, it is a tool implementation option which form of escaping is used, which may be overridden by the "escape as …" encoding instruction.

NOTE 1: Note that the JSON module in Annex A defines useful TTCN-3 constants for the characters listed above.

The following TTCN‑3 type shall be used to map JSON strings to TTCN-3:

**type** **universal** **charstring** String **with** {

**variant** "JSON:string"

}

NOTE 2: Though Unicode and ISO/IEC 10646 [3] do not necessarily contain the same set of characters at all points in time, JSON strings are expressed using the TTCN-3 universal charstring type.

In addition to the generic encoding instructions like "normalize" and "name as …", the following specific encoding instructions are applicable to **JSON.String** types:

* escape as … see clause B.3.7

EXAMPLE: String encoding examples

If:

**const** JSON.String c\_string1 := <actual value> **with** {**variant** "escape as short"};

then

|  |  |  |  |
| --- | --- | --- | --- |
| <actual value> | JSON character sequence | UTF-8 serialization of the JSON value | Note |
| "abcd" | "abcd" | 226162636422 |  |
| "ab\cd" | "ab\\cd" | 2261625C5C636422 |  |
| "ab/cd" | "ab\/d" | 2261625C2F636422 |  |
| "ab" & char(U7) & "cd" | "ab\u0007cd" | 2261625C75303030 37636422 |  |
| "ab" & char(U7) & cu\_ht & "cd" | "ab\u0007\tcd" | 2261625C7530303037 5C74636422 |  |

If:

**const** JSON.String c\_string1 := <actual value> **with** {**variant** "escape as usi"};

then

|  |  |  |  |
| --- | --- | --- | --- |
| <actual value> | JSON character sequence | UTF-8 serialization of the JSON value | Note |
| "abcd" | "abcd" | 226162636422 |  |
| "ab\cd" | "ab\u005Ccd" | 2261625C75303035 43636422 |  |
| "ab/cd" | "ab/cd" | 2261622F636422 | JSON doesn't require to escape the solidus character |
| "ab" & char(U7) & "cd" | "ab\u0007cd" | 2261625C75303030 37636422 |  |
| "ab" & char(U7) & cs\_ht & "cd" | "ab\u0007\u0009cd" | 2261625C7530303037 5C7530303039636422 |  |

If:

**const** JSON.String c\_string1 := <actual value> **with** {**variant** "escape as transparent"};

then

|  |  |  |  |
| --- | --- | --- | --- |
| <actual value> | JSON character sequence | UTF-8 serialization of the JSON value | Note |
| "abcd" | "abcd" | 226162636422 |  |
| "ab\cd" | "ab\cd" | 2261625C636422 | Note that the resulting sequence is an invalid JSON encoding |
| "ab/cd" | "ab/cd" | 2261622F636422 |  |
| "ab" & char(U7) & cs\_ht & "cd" | "ab\u0007\tcd" | 2261625C7530303037 5C74636422 | Note that the BELL and HT C0 control characters are escaped by the encoder |

## B.3.7 Escape as

***Syntactical structure(s)***

**variant** """ escape as ( short | usi | transparent ) """,

***Applicable to (TTCN-3)***

Types and fields of *JSON.Number* and *JSON.String* types.

***Description***

The "escape as short" encoding instruction tells the encoder that all characters in the TTCN-3 value, which has short escape sequences defined (see IETF® RFC 7159 [2] and clause 6.4.2), shall be encoded using the short escape sequence.

The "escape as usi" encoding instruction tells the encoder that quotation mark (""", **char**(U22)), reverse solidus ("\", **char**(U5C)), and the control characters (**char**(U0) through **char**(U1F)), in the TTCN-3 value shall be encoded using the USI-like escape sequence "\u<HHHH>" (see IETF® RFC 7159 [2] and clause 6.4.2).

The "escape as transparent" encoding instruction tells the encoder that characters in the TTCN-3 value shall not be escaped in their JSON representation, except the C0 control characters (present in the TTCN-3 value in the **char**(…) representation).

NOTE: This instruction is useful, when a character string is copied from a JSON string, where the needed characters are already have been replaced by their escape sequences, into a TTCN-3 code.

This instruction has no effect at decoding, i.e. all escaped characters, using either the short or the USI-like escaping shall be decoded to and evaluated in its (abstract) character representation in TTCN-3 (e.g. at matching or in any other operations).