Simple Wire Protocol Specification v0.1α

Revision History

v0.1α, 2011-10-03, Initial revision, RKO.

Introduction

General Notes

Byte Order

Data encoded and transmitted using SWP should always be in “big-endian”. This includes all multi-byte protocol data (codepoints, packet lengths, etc) and also primitive data including integer values, floating point values, and multi-byte character sets.

SWP Packets

Each SWP packet consists of a 16-bit length token followed by a number of codepoints. A typical SWP reply contains a single packet but multiple packets can be “chained” together. If the high-order bit of the length token is 1 then there is at least one more packet in the reply. The high-order bit is not used as part of the length and should be masked out.

Length Tokens

The lengths of packets (and also codepoints) are represented by a 16-bit value. The high-order bit is excluded from the length but indicates whether this packet is part of a “chain” or “string” of packets in the reply. If the high-order bit is not set then the reply contains only a single packet.

0xxx xxxx xxxx xxxx

0 - 32,767

1xxx xxxx xxxx xxxx

0 - 32, 767 with high-order bit indicating that an additional packet follows this one in the reply.

Alternate Proposals

Variable Width Lengths

0XXX XXXX

Single byte length.
7 significant bits.
Values 1 - 127.

10XX XXXX XXXX XXXX

Double byte length.
14 significant bits.
Values 128 - 16,383.

110X XXXX XXXX XXXX XXXX XXXX

Triple byte length.

21 significant bits.
Values 16384 - 2,097,151.

1110 XXXX XXXX XXXX XXXX XXXX XXXX XXXX

Quadruple byte length.
28 significant bits.
Values 2,097,152 - 268,435,455.

TODO: 5 (11110X…), 6 (111110X…), 7 (1111110X…), and 8 (11111110X…) byte lengths.

1111 1111

Single byte length.
0 significant bits.
Indicates string of 32,767 byte packets of unknown length.
A packet with a normal length token indicates the last in the string.

Notes

It is legal to use a longer length token than is required to represent a value. For example, the value 47 can be represented in 4 different ways:

> 0010 1111
> 1000 0000 0010 1111
> 1100 0000 0000 0000 0010 1111
> 1110 0000 0000 0000 0000 0000 0010 1111

A packet with a length of 1 (0000 0001) is legal and contains no payload. It can be used to "punctuate" the stream while debugging to make it more human readable. However, packet lengths of 2 and 3 are not legal. The smallest packet size greater than 1 is 4: 1 byte length token, 2 byte codepoint, 1 byte codepoint length token. See the section on Codepoints for more information.

2 or 4-byte Proposal

0xxx xxxx xxxx xxxx

0000 0000 0000 0000 - 0111 1111 1111 1111
0 - 32,767

1xxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

0000 0000 0000 0000 0000 0000 0000 0000 - 0111 1111 1111 1111 1111 1111 1111 1111
0 - 2,147,483,647

Notes

We still need to address an unknown packet length type token

Fixed 4-byte Proposal

0xxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Values 0 - 2,147,483,647

1xxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

Values 0 - 2,147,483,647 with high-order bit indicating that an additional packet follows this one in the reply.

Codepoints

A codepoint defines an object in the SWP stream. Codepoints are two or four bytes long. On the wire a codepoint value is preceded by a 16-bit length token which includes the length of itself and the codepoint value. If the high-order bit in the length is set then the codepoint fills up the remainder of the packet. If the packet is “chained” then the codepoint also spans into the next packet. Following the codepoint value is an 8-bit NULL indicator. A non-zero value indicates that the codepoint is NULL and contains no data.

Codepoint Classes

There are three codepoint classes with three different scopes. In addition, each class of codepoints is divided into Object Codepoints and Primitive Codepoints. Object Codepoints encapsulate other codepoints. Primitive Codepoints encapsulate, or box, raw data.

000x xxxx xxxx xxxx

Universal SWP codepoints. Reserved for general SWP protocol. Inherited by all sub-protocols.

Primitive Codepoints (0000 xxxx xxxx xxxx)

0000 0001 0000 0000 - 0000 1111 1111 1111
0x0100 - 0x0fff
256 - 4095

Object Codepoints (0001 xxxx xxxx xxxx)

0001 0000 0000 0000 - 0001 1111 1111 1111

0x1000 - 0x1fff
4096 - 8191

Notes

Values 0x0000 - 0x00ff are not valid codepoints and should never appear in an SWP stream. If a leading 0x00 appears in a context where a codepoint is expected it should be ignored. A series of 0x00 values in a SWP stream is a “no-op slide” and may be used as padding in some cases.

001x xxxx xxxx xxxx

Sub-protocol specific codepoints. Defined for a specific sub-protocol and may overlap with specifications from other sub-protocols.

Primitive Codepoints (0010 xxxx xxxx xxxx)

0010 0000 0000 0000 - 0010 1111 1111 1111
0x2000 - 0x2fff
8192 - 12287

Object Codepoints (0011 xxxx xxxx xxxx)

0011 0000 0000 0000 - 0011 1111 1111 1111
0x3000 - 0x3fff
12288 - 16383

010x xxxx xxxx xxxx

Private codepoints. Defined by a single developer/entity/etc and used to extend SWP or a sub-protocol with proprietary features.

Primitive Codepoints (0100 xxxx xxxx xxxx)

0100 0000 0000 0000 - 0100 1111 1111 1111
0x4000 - 0x4fff
16384 - 20479

Object Codepoints

0101 0000 0000 0000 - 0101 1111 1111 1111
0x5000 - 0x5fff
20480 - 24575

011x xxxx xxxx xxxx

Reserved codepoints.
0110 0000 0000 0000 - 0111 1111 1111 1111
0x6000 - 0x6fff
24576 - 28671

1xxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx

4-byte codepoints have the high-order bit set. They are divided into three classes with primitive and object types just like the double-byte codepoints:

1000 xxxx xxxx xxxx xxxx xxxx xxxx xxxx (SWP, primitive)
1001 xxxx xxxx xxxx xxxx xxxx xxxx xxxx (SWP, object)
1010 xxxx xxxx xxxx xxxx xxxx xxxx xxxx (sub-protocol, primitive)
1011 xxxx xxxx xxxx xxxx xxxx xxxx xxxx (sub-protocol, object)
1100 xxxx xxxx xxxx xxxx xxxx xxxx xxxx (private, primitive)
1101 xxxx xxxx xxxx xxxx xxxx xxxx xxxx (private, object)

SWP-Defined Primitives

SWP defines some basic primitive codepoints which suit most needs.

Unboxed Primitives

At times a primitive codepoint may be included as a sub-codepoint in an “unboxed” format. This means that the length and codepoint are excluded and only the raw data is included. Note that only fixed-length primitives (such as byte, int, double, etc) can be represented in an “unboxed” format. Variable length codepoints (such as bytes and strings) cannot be strictly “unboxed” because they also require a length component, however, in certain defined contexts (in a homogeneous array, for example) the length and data will appear without the identifying codepoint. Note that because the length is embedded in the value, VarInt primitives can appear “unboxed” even though they are variable in length.

SWP_CP_CP (0x0100)

A codepoint value used to transmit a codepoint as metadata.

Example (SWP_CP_BYTE):

00 07 01 00 00 01 01

Codepoint Length	Codepoint	Null Indicator	Data
00 07	01 00	00	01 01

SWP_CP_BYTE (0x0101)

8-bit unsigned value.

Example (32):

00 06 01 01 00 20

Codepoint Length	Codepoint	Null Indicator	Data
01 0100 06	01 01	00	20

SWP_CP_SBYTE (0x0102)

8-bit signed value.

Example (-2):

00 06 01 02 00 fe

Codepoint Length	Codepoint	Null Indicator	Data
00 06	01 02	00	fe

SWP_CP_USHORT (0x0103)

16-bit unsigned value.

Example (58,055):

00 07 01 03 00 e2 c7

Codepoint Length	Codepoint	Null Indicator	Data
00 07	01 03	00	e2 c7

SWP_CP_SHORT (0x0104)

16-bit signed value.

Example (-7,481):

00 07 01 04 00 e2 c7

Codepoint Length	Codepoint	Null Indicator	Data
00 07	01 04	00	e2 c7

SWP_CP_UINT (0x0105)

32-bit unsigned value.

Example (3,151,788,837):

00 09 01 05 00 bb dc 7b 25

Codepoint Length	Codepoint	Null Indicator	Data
00 09	01 05	00	bb dc 7b 25

SWP_CP_INT (0x0106)

32-bit signed value.

Example (-1,143,178,459):

00 09 01 06 00 bb dc 7b 25

Codepoint Length	Codepoint	Null Indicator	Data
00 09	01 06	00	bb dc 7b 25

SWP_CP_ULONG (0x0107)

64-bit unsigned value.

Example (934,878,509,234,847,509):

00 0d 01 07 00 0c f9 5b 0d 0b 16 ff 15

Codepoint Length	Codepoint	Null Indicator	Data
00 0d	01 07	00	0c f9 5b 0d 0b 16 ff 15

SWP_CP_LONG (0x0108)

64-bit signed value.

Example (-34,741,714,498,866,452):

00 0d 01 08 00 ff 84 92 98 40 68 32 ec

Codepoint Length	Codepoint	Null Indicator	Data
00 0d	01 08	00	ff 84 92 98 40 68 32 ec

SWP_CP_VARINT (0x0109)

A signed, variable-length integer value. See Variable Width Lengths for details.

SWP_CP_SINGLE (0x010a)

32-bit IEEE single precision floating point value.

Example (2.986939):

00 09 01 0a 00 01 2a 3f 40

Codepoint Length	Codepoint	Null Indicator	Data
00 09	01 0a	00	01 2a 3f 40

SWP_CP_DOUBLE (0x010b)

64-bit IEEE double precision floating point value.

Example (31.1640777590893):

00 0d 01 0b 00 10 4a 01 00 01 2a 3f 40

Codepoint Length	Codepoint	Null Indicator	Data
00 0d	01 0b	00	10 4a 01 00 01 2a 3f 40

SWP_CP_DFP32 (0x010c)

32-bit IEEE 754 (decimal32) encoded Decimal Floating Point value.

SWP_CP_DFP64 (0x010d)

64-bit IEEE 754 (decimal64) encoded Decimal Floating Point value.

SWP_CP_DFP128 (0x010e)

128-bit IEEE 754 (decimal128) encoded Decimal Floating Point value.

SWP_CP_BYTES (0x010f)

String of bytes.

Example (68 65 6c 6c 6f):

00 0a 01 0f 00 68 65 6c 6c 6f

Codepoint Length	Codepoint	Null Indicator	Data
00 0a	01 0f	00	68 65 6c 6c 6f

SWP_CP_STRING_ASCII (0x0110)

String of ASCII characters.

Example (“hello”):

00 0a 01 10 00 68 65 6c 6c 6f

Codepoint Length	Codepoint	Null Indicator	Data
00 0a	01 10	00	68 65 6c 6c 6f

SWP_CP_STRING_UTF8 (0x0111)

String of UTF-8 characters.

Example (“hello”):

00 0a 01 11 00 68 65 6c 6c 6f

Codepoint Length	Codepoint	Null Indicator	Data
00 0a	01 11	00	68 65 6c 6c 6f

SWP_CP_STRING_UTF16 (0x0112)

String of UTF-8 characters.

Example (“hello”):

00 0f 01 12 00 00 68 00 65 00 6c 00 6c 00 6f

Codepoint Length	Codepoint	Null Indicator	Data
00 0f	01 12	00	00 68 00 65 00 6c 00 6c 00 6f

TODO:

SWP_CP_TIME (0x0113)
SWP_CP_DATE (0x0114)
SWP_CP_DATETIME (0x0115)
SWP_CP_DATETIME_TZ (0x0116)

SWP Object Codepoints

SWP_CP_ARRAY (0x1000)

An SWP_CP_ARRAY Object codepoint represents an array of codepoints. It is possible for the array to contain a “mixed bag” of codepoints but this is generally discouraged.The element types can be either a Primitive or Object codepoints.

TODO: Consider excluding the CP for each element if the ElementType is non-null. This will make the entire value smaller

Instance Codepoints

ElementType (SWP_CP_CP)

A single SWP_CP_CP codepoint must be the first instance (or "sub-") codepoint of an SWP_CP_ARRAY codepoint. If this value is not NULL then the array must be a homogeneous array (i.e. all elements are the same type) and each element must be "unboxed". If this value is NULL then the type of elements may be mixed and each element must be properly "boxed".

ElementCount (SWP_CP_UINT)

A single SWP_CP_UINT codepoint must be the second instance codepoint of an SWP_CP_ARRAY codepoint. Again, it is not technically required to parse the codepoint but it is useful to know in advance how many elements are present in the array. It is possible for a NULL SWP_CP_UINT to be sent in which case the number of elements must be inferred.

Elements (Zero or more Codepoints)

Following the ElementType and ElementCount instance codepoints are zero or more element codepoints.

Example

Below is an example of an array of UTF-8 strings: { "hello", "world", "goodbye", "world", <null> }. Note that because this is a homogeneous array, the codepoint for each element is omitted.

```
00 40 10 00 00
00 07 01 00 00 01 10
00 09 01 05 00 00 00 00 05
00 07 00 68 65 6c 6c 6f
00 07 00 77 6f 72 6c 64
00 0a 00 67 6f 6f 64 6d 79 65
00 07 00 77 6f 72 6c 64
00 03 ff
```

Length	Codepoint	Null Indicator	Data	Comments
00 40	10 00	00	n/a	SWP_CP_ARRAY Codepoint
00 07	0100	00	01 10	ElementType
00 09	01 05	00	00 00 00 05	ElementCount

00 0a	n/a	00	00 68 65 6c 6c 6f	Element 0
00 0a	n/a	00	77 6f 72 6c 64	Element 1
00 0c	n/a	00	67 6f 6f 64 6d 79 65	Element 2
00 0a	n/a	00	77 6f 72 6c 64	Element 3
00 05	n/a	ff	<null>	Element 4

Example

Below is an example of a mixed array of UTF-8 strings and integers: { “hello”, 123, “world”, 456 }.

```
00 40 10 00 00
00 07 01 ff
00 09 01 05 00 00 00 00 04
00 0a 01 10 00 68 65 6c 6c 6f
00 09 01 06 00 00 00 00 7b
00 0a 01 10 00 77 6f 72 6c 64
00 09 01 06 00 00 00 01 c8
```

Length	Codepoint	Null Indicator	Data	Comments
00 40	10 00	00	n/a	SWP_CP_ARRAY Codepoint
00 07	0100	ff	n/a	ElementType
00 09	01 05	00	00 00 00 04	ElementCount
00 0a	01 10	00	00 68 65 6c 6c 6f	Element 0
00 09	01 06	00	00 00 00 7b	Element 1
00 0a	01 10	00	77 6f 72 6c 64	Element 2
00 09	01 06	00	00 00 01 c8	Element 3