Sutton-Slevinski Collaboration
Internet-Draft
Intended status: Informational
Expires: 3 August 2022

S. Slevinski
www.sutton-signwriting.io
30 January 2022

Formal SignWriting
draft-slevinski-formal-signwriting-09

Abstract

Sutton SignWriting is the universal and complete solution for written sign language, ISO 15924 script code "Sgnw". It has been applied by a wide and deep international community of sign language users. Sutton SignWriting is an international standard for writing sign languages by hand or with computers. From education to research, from entertainment to religion, SignWriting has proven useful because people are using it to write signed languages.

Formal SignWriting is one particular computerized design for Sutton SignWriting that envisions a sign as a two part word. Each word is written as a string of characters that can be recognized and processed by regular expressions. The design has been optimized for display, searching, sorting, text flow, and other character processing.

Where as American Sign Language is a natural language, Formal SignWriting is a formal language. A formal language uses words and punctuation to form text. Each word is expressed as a string of characters. Well-formed words are governed by the structural rules of the grammar. A formal language is useful in mathematics, computer science, and linguistics.

This memo defines a conceptual character encoding map for the Internet community. It is published for reference, examination, implementation, and evaluation. Distribution of this memo is unlimited.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 3 August 2022.

Table of Contents

1. Sutton SignWriting

M548x535S10019452x474S10011476x465S2ea04481x501S2ea48459x509S29b0b514x500S15a0a

U+1D803 U+1D936 U+1D929 U+4001A U+1D8D6 U+1D8EC U+40012 U+1D8EE
U+1D8E3 U+4B7C5 U+1D8F3 U+1D907 U+4B809 U+1D8DD U+1D90F U+49A2C
U+1D914 U+1D906 U+421CB U+1D915 U+1D8EB U+45841 U+1D91E U+1D8FB
("01D803 01D936 01D929 04001A 01D8D6 01D8EC 040012 01D8EE 01D8E3 04B7C5 01D8F3 01D907 04B809 01D8DD 01D90F 049A2C 01D914 01D906 042 1CB 01D915 01D8EB 045841 01D91E 01D8FB")

Sutton SignWriting is the universal and complete solution for written sign language. It has been applied by a wide and deep international community of sign languages including: American Sign Language, Arabian Sign Languages, Australian Sign Language, Bolivian Sign Language, Brazilian Sign Language, British Sign Language, Catalan Sign Language, Colombian Sign Language, Czech Sign Language, Danish Sign Language, Dutch Sign Language, Ethiopian Sign Language, Finnish Sign Language, Flemish Sign Language, French-Belgian Sign Language, French Sign Language, German Sign Language, Greek Sign Language, Irish Sign Language, Italian Sign Language, Japanese Sign Language, Malawi Sign Language, Malaysian Sign Language, Maltese Sign Language, Mexican Sign Language, Nepalese Sign Language, New Zealand Sign Language, Nicaraguan Sign Language, Norwegian Sign Language, Peruvian Sign Language, Philippines Sign Language, Polish Sign Language, Portuguese Sign Language, Quebec Sign Language, South African Sign Language, Spanish Sign Language, Swedish Sign Language, Swiss Sign Language, Taiwanese Sign Language, and Tunisian Sign Language.

Sutton SignWriting is an international standard for writing sign languages by hand or with computers. From education to research, from entertainment to religion, SignWriting has proven useful because people are using it to write signed languages.

1.1. Script

Sign language is vastly different than spoken language. Instead of the sequential sounds of the voice, there is a 3-dimensional space with simultaneous action. Sutton SignWriting creates 2-dimensional writing that is visually iconic and full of featural information. This is true on the symbol level and on the sign level. A symbol represents phonemic information and is full of featural information to better understand the phonemes of the symbols. A sign is a 2-dimensional arrangement of symbols and is full of featural information to better understand the morphemes of the signs.

Punctuation is represented by a single symbol and separates a series of signs into structured sentences. Line breaks should not occur before punctuation.

When written vertically, SignWriting can use 3 different lanes: left, middle, and right. The middle lane is the default lane and punctuation is always used in the middle lane. No matter the lane, the center of a sign is aligned with the center of the lane. The left and right lanes are used to represent body weight shifts and are represented by a horizontal offset from the middle lane. Body weight shifts are important to the grammar of sign languages, used for two different grammatical aspects: 1) role shifting during sign language storytelling, and 2) spatial comparisons of two items under discussion. One "role" or "item" is placed on the right side of the body (right lane), and the other on the left side of the body (left lane), and the weight shifts back and forth between the two, with the narrator in the middle (middle lane).

1.2. Symbols

The Sutton SignWriting Symbols are the building blocks of Sutton SignWriting. The symbols are arranged in 2 dimensions to create the sign images. The symbols are organized with a 16-bit coded character set and a layered hierarchy. The symbols are defined in the International SignWriting Alphabet 2010 (ISWA 2010). The ISWA 2010 is a product of the Sutton-Slevinski collaboration.

2. Formal SignWriting

Formal SignWriting is one particular computerized encoding for Sutton SignWriting. The design is based on character processing with regular expressions. With Formal SignWriting, each sign is written as a two-part word of time and space.

Where as American Sign Language is a natural language, Formal SignWriting is a formal language. A formal language uses words and punctuation to form text. Each word is expressed as a string of characters. Well-formed words are governed by the structural rules of the grammar. A formal language is useful in mathematics, computer science, and linguistics.

2.1. Design Principles

Formal SignWriting was created using four design principles: completeness, universality, empowerment, and possibility.

2.1.1. Complete

Sutton SignWriting is a complex script with unique requirements and processing. Formal SignWriting supports all of the structures inherent to the script.

2.1.2. Universal

Sutton SignWriting can be used to write any sign language, natural or constructed. Formal SignWriting supports all sign languages without requiring the addition of new characters or updated fonts. Whereas chinese encoded text is an ever expanding set of ideographs which require new fonts and possiblly new characters, SignWriting uses a closed set for characters with completed fonts that do not need to be updated.

2.1.3. Empowering

Sutton SignWriting is flexible enough to let each writer decide how they want to write their signs. Formal SignWriting enable the writers to decide for themselves the spelling of their respective

signs.

2.1.4. Possible

Sutton SignWriting is a practical script that makes it possible to write sign language. Formal SignWriting is a practical encoding because it works with existing font technologies across operating systems.

2.2. Characters

Any sign can be written as a string of characters. Formal SignWriting has two sets of characters that can be used: Formal SignWriting in ASCII (FSW) and SignWriting in Unicode (SWU). These sets are isomorphic with an easy bi-directional conversion between the two sets.

Description	FSW Characters	SWU Characters
Sequence Marker	A	U+1D800
Signbox Markers	B, L, M, R	U+1D801 to U+1D804
Numbers	250 to 749	U+1D80C to U+1D9FF
Symbols	S10000 to S38b07	U+40001 to U+4F428

Table 1

2.2.1. Formal SignWriting in ASCII (FSW)

Formal SignWriting in ASCII (FSW) was released in January 2012 and has been stable since. FSW only uses characters from the ASCII subset of "ABLMRS0123456789xabcdef".

2.2.2. SignWriting in Unicode (SWU)

SignWriting in Unicode (SWU) was first published in October 2016 and officially submitted to the Unicode Technical Committee in July 2017. SWU is not part of the Unicode standard.

SignWriting in Unicode (SWU) is an experimental Unicode design that is supported by the Sutton SignWriting resources. This alternate encoding overwrites the Sutton SignWriting block in Unicode and uses plane 4 for the SignWriting symbols.

2.3. Building Blocks

The mathematical words of Formal SignWriting are plain text strings of characters from either character set: Formal SignWriting in ASCII (FSW) or SignWriting in Unicode (SWU).

2.3.1. Regular Expressions

Regular Expressions define string matching criteria. Regular Expressions offer fast processing and wide support on the various platforms.

Formal SignWriting is defined with regular expressions. Formal languages and regular expressions are used to solve fundamental problems.

Regular Expression Basics

Characters	Description	Example
*	Match a literal 0 or more times	ABC* matches AB, ABC, ABCC, ...
+	Match a literal 1 or more times	ABC+ matches ABC, ABCC, ABCCC, ...
?	Match a literal 0 or 1 times	ABC? matches AB or ABC
{#}	Match a literal "#" times	AB{2} matches ABB

[ ]	Match any single literal from a list	[ABC] matches A, B, or C
[-]	Match any single literal in a range	[A-C] matches A, B, or C
( )	Creates a group for matching	A(BC)+ matches ABC, ABCBC, ABCBCBC, ...
(|)	Matches one of several alternatives	(AB|BC|CD) will match AB, BC, or CD
(?:)	Creates a non-capturing group	A(?:BC) will match ABC as one group

Table 2

2.3.2. Token Patterns

The Formal SignWriting encoding model makes explicit those features which can be effectively and efficiently processed. The mathematical names are structured with 11 different tokens. They can be grouped in 4 layers: the 5 structural makers (A, B, L, M, R), the 3 base symbol ranges (w, s, P), the 2 modifier indexes (i, o), and the numbers (n).

The Tokens of Formal SignWriting

Token	Description
A	Sequence Marker
B	Signbox Marker
L	Left Lane Marker
M	Middle Lane Marker
R	Right Lane Marker
w	Writing BaseSymbols
s	Detailed Location BaseSymbols

P	Punctuation BaseSymbols
i	Fill Modifiers
o	Rotation Modifiers
n	Number from 250 to 749

Table 3

These tokens are used in patterns to form written sign language.

2.3.3. Symbols

Symbols can be described with 3 tokens: base symbol, fill modifier, and rotation modifier.

Symbol Tokens

Token Pattern	Description
w	Writing BaseSymbols.
s	Detailed Location BaseSymbols.
P	Punctuation BaseSymbols.
i	Fill Modifiers.
o	Rotation Modifiers.
wio	A writing symbol as 3 tokens of writing base, fill modifier and rotation modifier. Writing symbols can be used in the spatial signbox or the temporal prefix.
[ws]io	A writing symbol or a detailed location symbol as 3 tokens of base, fill modifier, and rotation modifier. Writing symbols and detail location symbols can be used in the temporal prefix.
Pio	A punctuation symbol as 3 tokens of punctuation base, fill modifier, and rotation modifier. Punctuation symbols divide signs into sentences.

Table 4

There are a variety of symbol types that are used for different purposes.

Symbol Types and Descriptions

Type	Description

all symbols	All symbols used in Formal SignWriting.
writing	Symbols that can be used in the spatial signbox or the temporal prefix.
hand	Various handshapes
movement	Contact symbols, small finger movements, straight arrows, curved arrows and circles.
dynamic	Dynamic symbols are used to give the "feeling" or "tempo" to movement.
head	Symbols for the head and face.
hcenter	Used to determine the horizontal center of a sign. Same as the head type.
vcenter	Use to determine the vertical center of a sign. Includes the head and trunk types.
trunk	Symbols for torso movement, shoulders, and hips.
limb	Symbols for limbs and fingers.
location	Detailed location symbols can only be used in the temporal prefix.
punctuation	Punctual symbols are used to divide signs into sentences.

Table 5

Symbol types occur in specific ranges depending on the characters involved.

Symbol Types and Ranges

Type	FSW	SWU
all symbols	S100 - S38b	U+40001 -U+4F480
writing	S100 - S37e	U+40001 -U+4EFA0
hand	S100 - 204	U+40001 -U+461E0
movement	S205 - S2f6	U+461E1 -U+4BCA0
dynamic	S2f7 - S2fe	U+4BCA1 -U+4BFA0
head	S2ff - S36c	U+4BFA1 -U+4E8E0
hcenter	S2ff - S36c	U+4BFA1 -U+4E8E0

vcenter	S2ff - S375	U+4BFA1 -U+4EC40
trunk	S36d - S375	U+4E8E1 -U+4EC40
limb	S376 - S37e	U+4EC41 -U+4EFA0
location	S37f - S386	U+4EFA1 -U+4F2A0
punctuation	S387 - S38b	U+4F2A1 -U+4F480

Table 6

2.3.3.1. FSW Symbols

Symbol keys are 6 characters long. The first character of a symbol key is always "S". The next 3 characters identify the symbol base. The last two characters identify the fill and rotation modifiers respectively.

Symbol Key Definition

Regular Expression	Description

S	Start of symbol key
[123][0-9a-f]{2}	Symbol key base
[0-5]	Fill modifier
[0-9a-f]	Rotation modifier
S[123][0-9a-f]{2}[0-5][0-9a-f]	Symbol key definition

Table 7

2.3.3.2. SWU Symbols

The 37,811 symbols of the International SignWriting Alphabet 2010 are uniquely identified with Unicode characters in the range U+40001 to U+4F428.

A simple formula transforms a symbol key into a codepoint. Given a symbol key as variable "key", in JavaScript the function is defined as:


```
var code = ((parseInt(key.slice(1,4),16) - 256) * 96) +
((parseInt(key.slice(4,5),16))*16) + parseInt(key.slice(5,6),16) +
1;
```

2.3.4. Numbers

The numbers encode the ruler principle with characters. The ruler principle is built in automatically for scripts written sequentially in one dimension. The number characters are needed to specify the spatial relationship between symbols.

Both FSW and SWU use a restricted range of 500 numbers between 250 and 749.

Cartesian Coordinates can be described with 2 tokens: number and number. These numbers represent the X and Y coordinates respectively.

Coordinate Tokens

Token Patterns	Description
n	Number from 250 to 749
nn	Coordinate with X and Y values as 2 numbers

Table 8

2.3.4.1. FSW Numbers

Formal SignWriting in ASCII has two definitions for a number. The more general definition simply defines 3 digits together with a potential range of 1000. A more explicit definition correctly restricts the numbers to 500 possibilities in the 250 to 749 range. The general coordinate definition is adequate for processing.

An X,Y coordinate is created by using the letter "x" to join two FSW numbers.

General 3 digit number definition: [0-9]{3}

General coordinate definition: [0-9]{3}x[0-9]{3}

Explicit number definition from 250 to 749: (2[5-9][0-9]|[3-6][0-9]{2}|7[0-4][0-9])

Explicit coordinate definition:
(2[5-9][0-9]|[3-6][0-9]{2}|7[0-4][0-9])x(2[5-9][0-9]|[3-6][0-9]{2}|7[0-4][0-9])

2.3.4.2. SWU Numbers

SignWriting in Unicode has a single definition for a number. Each number is uniquely identified with Unicode characters in the range U+1D80C to U+1D9FF. A coordinate is defined as 2 numbers together.

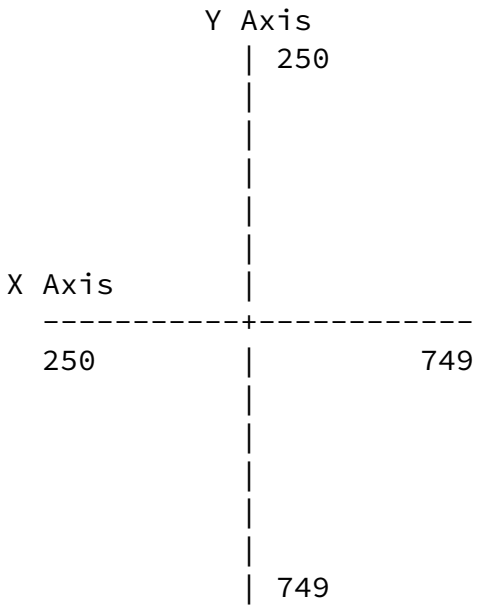
2.4. Two-Part Word

Formal SignWriting envisions a sign as a two-part word of time and space. The two-dimensional appearance of a sign is written in the spatial signbox as an objective arrangement. The one-dimensional order of a sign is written in the temporal prefix as a subjective analysis.

2.4.1. Spatial Signbox

The spatial signbox is a two-dimensional cluster of symbols. The position of each symbol is determined by the writer and defined using Cartesian Coordinates that represent the top-left of the symbol image. Formal numbers range from 250 to 749.

2-dimensional space does not have a normative 1-dimensional order. When symbols overlap, the relative order of the overlapping symbols is important. Symbols written first appear underneath symbols that are written later. Otherwise, the exact string order of the spatial symbols is unpredictable. The spatial signbox is neither formatting nor style and represents meaning that is beyond the temporal prefix.



The Spatial Signbox can be described with 8 tokens.

Spatial Signbox Tokens

Token Pattern	Description
B	Signbox Marker
L	Left Lane Marker
M	Middle Lane Marker
R	Right Lane Marker
w	Writing BaseSymbols
i	Fill Modifiers
o	Rotation Modifiers
n	Number from 250 to 749
wio	A writing symbol as 3 tokens of writing base, fill modifier and rotation modifier
nn	Coordinate with X and Y values as 2 numbers
wionn	A spatial symbol as 5 tokens, with 3 tokens for a writing symbol and 2 tokens for coordinates of top left placement
(wionn)*	Zero or more spatial symbols
Bnn(wionn)*	A Signbox with a preprocessed maximum coordinate and a list of spatial symbols used for horizontal writing
[LMR]	A lane marker: either left, middle or right.
[LMR]nn(wionn)*	A Signbox in either the left, middle, or right lane with a preprocessed maximum coordinate and a list of spatial symbols used for vertical writing

Table 9

The spatial signbox is assigned to a lane, has a preprocessed maximum coordinate and zero or more writing symbols with X and Y coordinates for each symbol.

2.4.1.1. Bounding Box

The symbols do not have a consistent width or height. The center of a symbol can be safely assumed to be at half-width and half-height. A bounding box for a symbol is based on the symbol width and height. Each symbol has a defined width and height [SWFontSource] in a text file with 37,811 lines. Alternately, the symbol width and height can be calculated by analyzing the glyphs in a TTF font file, using JavaScript or other language.

The bounding box of a sign is a tight box around the symbols. The bounding box is used to determine the width and height of a sign.

The bounding box of a sign consists of four values: Minimum X, Minimum Y, Maximum X and Maximum Y. The values of the bounding box is taken straight from the coordinates in a Formal SignWriting word.

2.4.1.2. Maximum Coordinate

The maximum coordinate for a Signbox is pre-calculated to simplify layout for width, height, and center. For each symbol, the width of height of that symbol is added to the coordinate position of that symbol. These new coordinate values represent the bottom-right coordinate of each symbol bounding box. The maximum X value is joined with the maximum Y value to determine the maximum coordinate.

2.4.1.3. Centering a Sign

To simplify layout and improve 2-dimensional searching, every sign has a normalized center based on symbol type, size, and mathematical formula. The vertical center is based on the center of the bounding box around the head symbols. The horizontal center is based on the center of the bounding box around the head and trunk symbols. If a sign doesn't contain head or trunk symbols, then the bounding box of all symbols is used. For the symbol ranges see Table 6

Once the center of a sign has been determined, the symbols are moved so that the center is coordinate 500,500.

2.4.2. Temporal Prefix

The temporal prefix is a one-dimensional list of symbols that is written by an author. The arrangement of the symbols is based on a particular theory of sorting. The order of the symbols in the temporal prefix is significant because sorting is possible with a binary string comparison. The temporal prefix is neither formatting nor style and represents meaning not found in the spatial signbox.

Signs are written in 2-dimensional space which does not have a normative 1-dimensional order. Any 1-dimensional order of 2-dimensional space is subjective. Some 1-dimensional orders may be canonical according to a particular theory, but there are a variety of theories on setting a 1-dimensional order.

The temporal prefix will use the same symbols that are used in the spatial signbox, but it does not need to use all of them and it is not limited to only those symbols. The temporal prefix is a list of writing symbols and/or detailed location symbols that identify temporal order and additional analysis. A valid sequence must contain at least one symbol and can not contain punctuation.

The temporal prefix allows for sorting that is universally supported through binary string comparison.

There are several theories on the best way to structure a temporal prefix. The most productive is based on the SignSpelling Sequence theory of Valerie Sutton. A temporal prefix is structured as a series of beginning handshapes, followed by transitional movements and dynamics that lead to the next set of handshapes. This pattern continues until the end of the sign. The last section of the temporal prefix should contain symbols of type "head", "trunk", and "limb".

Detailed location symbols of type "location" can be used in a temporal prefix, but are rarely (if ever) needed for general writing.

A temporal prefix can be described with 5 tokens.

Temporal Prefix Tokens

Token Patterns	Description
A	Sequence Marker
w	Writing BaseSymbols
s	Detailed Location BaseSymbols
i	Fill Modifiers
o	Rotation Modifiers
(A([ws]io)+)?	An optional temporal prefix to be used as a prefix for a Signbox

Table 10

The temporal prefix starts with a sequence marker and includes an ordered list of writing symbols and detailed locations.

2.5. Styling String

The styling string of Formal SignWriting uses a lite markup to define a variety of styling options. The styling string is the same for FSW and SWU. The entire sign can be customized for padding, coloring, and size. Individual symbols within a sign can be customized for coloring and size. For SVG output, class names and IDs can be defined. A styling string can be added to the end of any Formal SignWriting string to style a particular sign.

Colors can be written as CSS color names or as color hex values.

CSS Color Names: [a-zA-Z]+

Color Hex Values: [0-9a-fA-F]{3}([0-9a-fA-F]{3})?

The styling string is divided into 3 sections: one for the entire sign, one for individual symbols, and one for SVG class names and ID. The styling string starts with a single dash, after which is the section about the entire sign. A second dash, if present, marks the start of the section about the individual symbols. A third dash, if present, marks the start of the section about the SVG class names and ID. The order of the styling options is important.

Styling String:

```
-C?(P[0-9]{2})?(G_([0-9a-fA-F]{3}([0-9a-fA-F]{3})?|[a-zA-
Z]+)_)?(D_([0-9a-fA-F]{3}([0-9a-fA-F]{3})?|[a-zA-Z]+)(,([0-9a-fA-
F]{3}([0-9a-fA-F]{3})?|[a-zA-
Z]+))?_)?(Z([0-9]+(\.[0-9]+)?|x))?(-(D[0-9]{2}_([0-9a-fA-F]{3}([0-
9a-fA-F]{3})?|[a-zA-Z]+)(,([0-9a-fA-F]{3}([0-9a-fA-F]{3})?|[a-zA-
Z]+))?_)*)?(--?[_a-zA-Z][_a-zA-Z0-9-]{0,100}( -?[_a-zA-Z][_a-zA-
Z0-9-]{0,100})*!([a-zA-Z][_a-zA-Z0-9-]{0,100}!)?)?
```


2.5.1. Entire Sign

There are several options for styling an entire sign.

C Colorize

P Padding

G Background

D Detail colors

Z Zoom level

2.5.1.1. Colorize

Colorizing a sign will set the color of each symbol based on its classification.

Hand 0000CC

Movement CC0000

Dynamic FF0099

Head 006600

Body 000000

Detailed Location 884411

Punctuation FF9900

Styling String	Description
-C	Colorize the symbols of the sign

Table 11

2.5.1.2. Padding

Padding is applied around the entire sign. A two-digit number is used to set the padding.

Styling String	Description
-P01	A padding of 1 around the sign

Table 12

2.5.1.3. Background

By default, the background of a sign is transparent. The background color can be set with a CSS color name or with a color hex value.

The color name or value must be surrounded by underscores.

Styling String	Description
-G_lightblue_	Background color of light blue.
-G_f00_	Background color as 3 hex values.
-G_ff0000_	Background color as 6 hex values.

Table 13

2.5.1.4. Detail Colors

By default, each symbol has a line color of black and a fill color of white. The line color for all of the symbols can be set with a CSS color name or with a color hex value. The color name or value must be surrounded by underscores. Setting the fill color is optional. To set the fill color, put a comma and the fill color after the line color but before the closing underscore.

Styling String	Description
-D_red_	Line color of red.
-D_red,yellow_	Line color of red with a fill color of yellow.

Table 14

2.5.1.5. Zoom Level

By default, a sign is set to zoom level 1. The zoom level can be set with an integer or a decimal number.

Alternatively, the zoom level can be set to lower-case 'x', for extendable. The SVG created will not specify the width or height, so that the sign image will fill whatever container it is placed inside.

Styling String	Description
-Z2	Zoom level of 2
-Z15.7	Zoom level of 15.7
-Zx	Zoom level of extendable

Table 15

2.5.2. Individual Symbols

There is one option for styling individual symbols. Individual symbols are identified by a two-digit number, which identifies the order the symbol appears in the Signbox.

D Detail colors

2.5.2.1. Detail Colors

By default, each symbol has a line color of black and a fill color of white. The line color for an individual symbol can be set with a CSS color name or with a color hex value. The color name or value must be surrounded by underscores. Setting the fill color is optional. To set the fill color, put a comma and the fill color after the line color but before the closing underscore.

Styling String	Description
--D01_red_	First symbol line color of red.
--D01_red,yellow_	First symbol line color of red with a fill color of yellow.
--D01_red_D02_green_	First symbol line color of red and second symbol line color of green.

Table 16

2.5.3. SVG Class Names and ID

When using SVG, there are two additional styling options of class names and ID.

{class names}! SVG Class Names

{ID}! SVG ID

Both class names and ID use a restricted ASCII subset.

class names -?[_a-zA-Z][_a-zA-Z0-9-]{0,100}(-?[_a-zA-Z][_a-zA-Z0-9-]{0,100})*

ID [a-zA-Z][_a-zA-Z0-9-]{0,100}

Each SVG can be created with a list of class names separated by spaces, ending in an exclamation (!) mark. After the class names exclamation mark, an ID can be written followed by another exclamation mark.

Styling String	Description
---glowing!	A class name of "glowing"
---flashing primary!	Two class names of "flashing" and "primary".
---!cursor!	SVG created with an ID of "cursor"
---flashing!cursor!	SVG created with a class name of "flashing" and an ID of "cursor"

Table 17

2.6. Query Language

The query language of Formal SignWriting allows for precise searching of signs written in either FSW or SWU. A query string is a concise representation for a much larger and detailed set of regular expressions. The regular expressions can be used to quickly and accurately search large files and databases containing Formal SignWriting.

A filter and repeat pattern of searching is used as a series of match criteria. A file, database, or text input is searched using a sequence of steps. Each step applies a single match criteria. Matching results are collated and the next search criteria is applied. The pattern of searching the previous results continues until all regular expressions have been used.

The query language of Formal SignWriting is different for FSW and SWU, but allows for the same searching options. Query strings contain three major sections: prefix, signbox, and styling. The prefix section and the signbox section mostly use the same elements.

2.6.1. Major Sections

There are three major sections of the query string: the prefix, the signbox, and the styling string.

Query Language Sections

Token	Name	Description

Q	Query marker	The start of a query string
P	Prefix marker	The searching of the temporal prefix
X	Signbox marker	The searching of the spatial signbox
Y	Styling string marker	Include the styling string in search results

Table 18

A query string always starts with the query marker (Q), followed by an optional prefix marker (P), followed by an optional signbox marker (X), followed by an optional styling string marker.

Full query string definition: QP?X?Y?

2.6.2. Common Elements

There are several common elements used for searching the prefix and the signbox.

Query Common Elements

Token	Name	Description
B	Symbol base marker	A symbol indicator that doesn't specify fill or rotation
f	Fill modifier	A modifier that specifies a symbol fill
r	Rotation modifier	A modifier that specifies a

		symbol rotation
S	Symbol marker	A marker that indicates a symbol
R	Range marker	A marker that starts a range definition
I	Item marker	An item can be either a symbol or a range definition
O	Or marker	A marker that connects a series of items

L	List marker	A marker that indicates a list of connected items

Table 19

Common definitions used in the temporal prefix and the spatial signbox.

Symbol definition: S = Bfr

Range definition: RBB

Item definition: I = (S|RBB)

List definition: L = I(OI)*

2.6.3. Searching the Temporal Prefix

Searching the temporal prefix requires two additional tokens.

Prefix Elements

Token	Name	Description
A	Prefix start marker	A marker that indicates the

		start of prefix searching
T	Prefix end marker	A marker that indications the enf of prefix searching

Table 20

Definition of temporal prefix searching.

Prefix searching definition: P = ((AL+)?T)?

2.6.4. Searching the Spatial Signbox

Searching the spatial signbox requires two additional tokens.

Signbox Elements

Token	Name	Description
C	Coordinate marker	A marker that indicates a coordinate
V	Variance marker	A marker that indicates a custom variance for location searching

Table 21

Definition of spatial signbox searching.

Signbox searching definition: X = (LC?)*V?

2.6.5. Regular Expressions

2.6.5.1. FSW Query Regular Expressions

FSW Query Elements

Token	Variable	Regular Expression

Q	Q	Q
B	base	[123][0-9a-f]{2}
f	fill	[0-5u]
r	rotation	[0-9a-fu]
S	symbol	S${base}${fill}${rotation}
R	range	R${base}${base}
I	item	(?:${symbol}\|${range})
O	or	o
L	list	${item}(?:${or}${item})*
A	A	A
T	T	T
P	prefix	(?:${A}(?:${list})+)?${T}
C	coord	(?:[0-9]{3}x[0-9]{3})?

X	signbox	(?:${list}${coord})*
V	var	V[0-9]+
Y	style	-
F	full	${Q}(${prefix})?(${signbox})?(${var})?(${style}?)

Table 22

2.6.5.2. SWU Query Regular Expressions

SWU Query Elements

Token	Variable	Regular Expression
Q	Q	Q
B	base	(?:(?:\uD8C0[\uDC01-\uDFFF])|(?:[\uD8C1-\uD8FC][\uDC00-\uDFF
f	fill	f?
r	rotation	r?
S	symbol	${base}${fill}${rotation}
R	range	R${base}${base}
I	item	(?:${symbol}\|${range})
O	or	o
L	list	${item}(?:${or}${item})*
A	A	A
T	T	T
P	prefix	(?:${A}(?:${list})+)?${T}
C	coord	(?:(?:\uD836[\uDC0C-\uDDFF]){2})?
X	signbox	(?:${list}${coord})*
V	var	V[0-9]+

Y	style	-
F	full	${Q}(${prefix})?(${signbox})?(${var})?(${style}?)

Table 23

3. Technology Integration

Formal SignWriting has been specifically designed to integrate with standard technology on the phone, tablet, and desktop.

3.1. Fonts

The Sutton SignWriting Fonts are available as source SVG and as three TrueType Font files.

> Sutton SignWriting Fonts
> Copyright (c) 1974-2017, Center for Sutton Movement Writing, inc
> Licensed under the SIL Open Font License v1.1

The Sutton SignWriting TrueType fonts are available for download and installation.

Installing the fonts using the instructions below is not required, but it will improve the user experience. If the fonts are not installed on the system, CSS declarations will install the fonts in the browser cache.

3.1.1. Windows, Linux, and Mac

Installation is straight forward for Windows, Linux and Mac. Simply download the TrueType fonts and install as usual.

Sutton SignWriting Line TrueType Font [SWFontLine]

Sutton SignWriting Fill TrueType Font [SWFontFill]

Sutton SignWriting One-D TrueType Font [SWFontOneD]

3.1.2. Mac and iOS

Installation is possible for Mac OS X and iOS with a configuration profile. The Sutton SignWriting Symbol configuration profile includes 2 fonts for SVG: SuttonSignWritingLine and SuttonSignWritingFill. The Sutton SignWriting One configuration profile includes the font SuttonSignWritingOneD. With the configuration profile installed, the SignWriting in Unicode (SWU)

characters can be used throughout the operating system, even as file

and folder names.

Sutton SignWriting Symbol Configuration Profile [SWFontSymbol]

Sutton SignWriting One Configuration Profile [SWFontOne]

3.1.3. Android

Android can not install the fonts directly onto the system. The CSS declarations below will install the fonts in the browser cache.

3.2. Fonts and CSS

The TrueType Fonts can be used without installing the fonts on any platform by defining two font-face statements. Simply include the following CSS in any HTML page to access the fonts. Make sure to replace the URLs with the fully qualified links for the fonts.

```
@font-face {
  font-family: "SuttonSignWritingLine";
  src:
    local('SuttonSignWritingLine'),
    url('https://.../SuttonSignWritingLine.ttf') format('truetype');
}
@font-face {
  font-family: "SuttonSignWritingFill";
  src:
    local('SuttonSignWritingFill'),
    url('https://.../SuttonSignWritingFill.ttf') format('truetype');
}
@font-face {
  font-family: "SuttonSignWritingOneD";
  src:
    local('SuttonSignWritingOneD'),
    url('https://.../SuttonSignWritingOneD.ttf') format('truetype');
}
```

If the fonts are installed, then the system fonts will be used. If the fonts are not installed when a SignWriting Font page is opened, the CSS will cause the fonts to be automatically downloaded to the browser's cache on the first visit. Once the fonts are installed in the browser cache, they will remain there until the browser cache is emptied. Any webside that uses this CSS can access the browser installed font without requesting a new copy. The fonts are 18 MB, so the first page view make take a few seconds or longer depending on your download speed and processor.

3.3. Scalar Vector Graphics

Sutton SignWriting is a 2-dimensional script. The sign images are composed using Scalar Vector Graphic (SVG).

3.3.1. Font Based SVG

The conversion of Formal SignWriting to Scalar Vector Graphics requires three parts: header, text, and symbols. Consider the FSW string "M518x533S1870a489x515S18701482x490S20500508x496S2e734500x468".

3.3.1.1. SVG Header

The header section contains the SVG definition along with the width, height, and viewbox. The viewbox is a combination of the minimum X, minimum Y, width, and height.

Minimum X: 482

Maximum X: 518

Width: 36

Minimum Y: 468

Maximum Y: 533

Height: 65

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
  width="36" height="65" viewBox="482 468 36 65">
```

If the width and height properties are not included, then the resulting SVG will automatically expand in size to fill the containing element on the screen.

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg"
  viewBox="482 468 36 65">
```

3.3.1.2. SVG Text

The SVG text section is included to make it possible to copy and paste Formal SignWriting strings. The font-size is set to zero to make the text invisible.

```
<text style="font-size:0%;">
M518x533S1870a489x515S18701482x490S20500508x496S2e734500x468
</text>
```

3.3.1.3. SVG Symbols

Each symbol in the Signbox is a combination of the symbol key and the positioning coordinate.

Symbol 1: S1870a 489x515

Symbol 2: S18701 482x490

Symbol 3: S20500 508x496

Symbol 4: S2e734 500x468

Each spatial symbol is written as an SVG group and positioned by the transformation translate.

```
  <g transform="translate(489,515)">...</g>
  <g transform="translate(482,490)">...</g>
  <g transform="translate(508,496)">...</g>
  <g transform="translate(500,468)">...</g>
```

Inside of each group, 2 text elements are written. The symbol fill is written first using the SuttonSignWritingFill font with a plane 16 character. The symbol line is written second using the SuttonSignWritingLine font with a plane 15 character. See Section 2.3.3.2 for the formula to convert symbol keys to codepoints.

```
<text class="sym-fill"
 style="font-family:'SuttonSignWritingFill';font-size:30px;fill:white;">
 {plane 16 codepoint}
</text>
<text class="sym-line"
 style="font-family:'SuttonSignWritingLine';font-size:30px;fill:black;">
 {plane 15 codepoint}
</text>
```

3.3.2. Stand Alone SVG

It is possible to request completed SVG images from SignPuddle 3. The SVG images created by the SignWriting Server are stand-alone graphics that do not use the TrueType Fonts. The SVG images use path elements to define the symbol lines and curves.

The SVG header and SVG text for the server-side images are the same as the standard FSW to SVG transformation. See Section 3.3.1

The SVG symbols section is structured differently. Multiple SVG elements are contained within each sign SVG image. Each sub-SVG element uses X and Y coordinates to place each symbol. Consider the FSW string "M518x533S1870a489x515S18701482x490S20500508x496S2e734500x468".

Symbol 1: S1870a 489x515

Symbol 2: S18701 482x490

Symbol 3: S20500 508x496

Symbol 4: S2e734 500x468

```
<svg x="489" y="515">...</svg>
<svg x="482" y="490">...</svg>
<svg x="508" y="496">...</svg>
<svg x="500" y="468">...</svg>
```

Inside of each sub-SVG element is a group (g) element with one or two path elements. This inside information can only be requested from the SignWriting Server or some other source of the symbol image data.

```
<g transform="translate(0.146473559361,17.7697467366) ... ">
  <path class="sym-fill" fill="white" d="M700 1493 ... "/>
  <path class="sym-line" fill="black" d="M1826 1480 ... "/>
</g>
```

3.4. HTML and CSS

Basic HTML structures and CSS rules can be used with Formal SignWriting for customization and layout.

3.4.1. Centering and Sizing

It is possible to center a symbol or sign within a div with a few CSS rules. The symbol or sign will automatically shrink in size if the containing div is smaller than the SVG image. Additionally, if the SVG is created with the zoom level of extendable (styling string "-Zx"), the symbol or sign will grow in size to fill as much of the containing div as possible.

```
<div class="centered">
  <svg version="1.1" xmlns="http://www.w3.org/2000/svg" ...
</div>
```



```
div.centered {
  position: relative;
  width: 10%;
  height: 10%;
  border: 1px solid black;
}

div.centered svg {
  position: absolute;
  display: block;
  top:2.5%;
  bottom: 2.5%;
  left: 2.5%;
  right: 2.5%;
  margin: auto;
  max-width: 95%;
  max-height: 95%;
}
```

3.4.2. Coloring Symbols and Signs

Individual signs can be colored with CSS rules. The individual classes of 'sym-line' and 'sym-fill' can be used to isolate each part of a symbol, both positive and negative spaces, or the classes can be ignored to create the shadow of a symbol that includes both aspects

of a symbol.

```
<svg class="primary" ...
<svg class="success" ...
<svg class="info" ...
<svg class="warning" ...
<svg class="danger" ...
<svg class="shadow" ...
<svg class="inverse" ...

svg.primary g text.sym-line { fill: #337ab7 !important; }
svg.success g text.sym-line { fill: #5cb85c !important; }
svg.info g text.sym-line { fill: #5bc0de !important; }
svg.warning g text.sym-line { fill: #f0ad4e !important; }
svg.danger g text.sym-line { fill: #d9534f !important; }
svg.shadow g text { fill: grey !important; }
svg.inverse g text.sym-line { fill: white !important; }
svg.inverse g text.sym-fill { fill: black !important; }
```


3.4.3. Other Effects

Other CSS rules can be used for other effect. Please note that transform property does not effect the document flow and should not be used for general layout.

```
svg.shadowed {
  text-shadow: -1px -1px 1px #fff, 1px 1px 1px #000;
}
svg.rotate {
  transform: rotate(0.5turn);
}
svg.bigger {
  transform: scale(2);
}
svg.skewed {
  transform: skewX(30deg);
}
```

3.4.4. Sentences

SignWriting is written vertically using the vertical writing mode of CSS. To create the center lane and to visually divide the columns of text, several span elements are used. Each sign is contained in a div with a width and height that matches the enclosed sign. To properly align each sign with the center of its lane, the containing div will either use "margin-right" or "border-left". With "border-left", the rule must include "solid transparent" after the size.

```
<div class="signtext">
 <span class="outside"><span class="middle"><span class="inside">
  <div style="width:42px;height:77px;margin-right:2px;"><svg ...
  <div style="width:38px;height:48px;margin-right:2px;"><svg ...
  <div style="width:25px;height:9px;border-left:7px solid transparent;">
```



```
div.signtext {
  -webkit-writing-mode: vertical-lr;
  writing-mode: vertical-lr;
  font-size: 0%;
  border-left: 1px solid blue;
  height: 100%;
}

span.outside { border-left: 1px solid blue; vertical-align: top; }
span.middle { vertical-align: bottom; }
span.inside { border-left: 1px dashed red; }
```

```
div.signtext div {
  writing-mode: horizontal-tb;
  display: inline-block;
  vertical-align: middle;
  padding: 20px;
  box-sizing: content-box;
}
```

3.5. JavaScript Packages

For modern web and app development, several packages are available on Github and NPM.

3.5.1. @sutton-signwriting/core

a javascript package for node and browsers that supports general processing of the Sutton SignWriting script

Source [SSWCoreSrc]

Documentation [SSWCoreDoc]

Distribution [SSWCoreDist]

```
npm install @sutton-signwriting/core
```

3.5.2. @sutton-signwriting/font-ttf

a javascript package for the web components and browser that generates SVG and PNG images for individual symbols and complete signs

Source [SSWTTFSrc]

Documentation [SSWTTFDoc]

Distribution [SSWTTFDist]

```
npm install @sutton-signwriting/font-ttf
```

3.5.3. @sutton-signwriting/font-db

a javascript package for node that generates SVG and PNG images for individual symbols and complete signs

Source [SSWFontDBSrc]

Documentation [SSWFontDBDoc]

Distribution [SSWFontDBDist]

npm install @sutton-signwriting/font-db

3.5.4. @sutton-signwriting/sgnw-components

a javascript package of Web Components for use with the SignWriting script

Source [SSWSgnwSrc]

Documentation [SSWSgnwDoc]

Distribution [SSWSgnwDist]

npm install @sutton-signwriting/sgnw-components

4. Transformations

Formal SignWriting and the surrounding technologies have been created to facilitate easy transformations between the various forms.

4.1. Formal SignWriting to Query String

Formal SignWriting strings have several natural transformations to query string. The transformation can use the temporal prefix and/or the spatial signbox. For each symbol, the query can include the exact symbol key, or the query can use a general symbol key where the fill and rotation modifiers are not explicitly defined. Consider the Formal SignWriting string "AS14c20S27106M518x529S14c20481x471S27106503x489".

Exact Temporal Prefix Symbols: QAS14c20S27106T

General Temporal Prefix Symbols: QAS14cuuS271uuT

Exact Spatial Signbox Symbols: QS14c20S27106

General Spatial Signbox Symbols: QS14cuuS271uu

Exact Spatial Signbox Symbols with Location: QS14c20481x471S27106503x489

General Spatial Signbox Symbols with Location: QS14cuu481x471S271uu503x489

4.2. Query String to Regular Expression

The transformation from query string to regular expressions has been fully implemented in the Sutton SignWriting JavaScript Library and the SignWriting Server.

The query language to regular expressions generator uses the following regular expression structures as building blocks.

Temporal Prefix Prefix: (A(S[123][0-9a-f]{2}[0-5][0-9a-f])+)

Signbox Prefix: [BLMR]([0-9]{3}x[0-9]{3})

Spatial Symbols: (S[123][0-9a-f]{2}[0-5][0-9a-f][0-9]{3}x[0-9]{3})*

The Temporal Prefix Prefix is a structural marker followed by one or more symbols. For the query string "QT", the prefix is required. For the general "Q", the prefix is optional so "?" is appended to the Temporal Prefix Prefix regular expression.

The Signbox Prefix is a combination of structural marker and preprocessed maximum coordinate. Every constructed regular expression will include the Signbox Prefix.

The Spatial Symbols is zero or more symbol definitions and associated coordinates. The Spatial Symbols regular expression is used for every search. For both "Q" and "QT", it is the only symbol matching used. When searching for specific symbols and ranges, the general Spatial Symbols definition will sandwich the specific search definitions.

Searching for number ranges with regular expressions requires a unique technique. This technique requires five steps.

Find a number between 122 and 455

1) 10's don't match and the min 1's are not zero (last number to

9): Match 12[2-9]

2) Bring up the 10's if hundreds are different: Match 1[3-9][0-9]

3) Bring up the 100's if different: Match [2-3][0-9][0-9]

4) Bring up the 10's: Match 4[0-4][0-9]

5) Bring up the 1's: Match 45[0-5]

Final Match (12[2-9]|1[3-9][0-9]|[2-3][0-9][0-9]|4[0-4][0-9]|45[0-5])

For the styling string regular expression, see Section 2.5.

5. Unicode Considerations

"The plan for encoding Sutton SignWriting in Unicode is for there to be two separate Unicode proposals. The first is for the symbol set covered by [the] ISWA 2010... The second is for an encoding that takes symbols and turns them into signs." -ScriptSource [UnicodeScript]

5.1. Unicode Technical Committee

In 2011, two documents were submitted: N4015 L2/11-101 [UnicodeN4015] and N4090 L2/11-217 [UnicodeN4090].

In 2012, one document was submitted: N4342 L2/12-321 [UnicodeN4342].

In 2015, the Sutton SignWriting Block was officially added to the Unicode standard. A documement was submitted: L2/15-194 [Unicode15194]. In July, Steve Slevinski attended UTC #144. After the meeting, another document was submitted: L2/15-219 [Unicode15219].

In 2016, one document was submitted: L2/16-225 [Unicode16225]. In August, Steve Slevinski attended UTC #148.

In 2017, two documents were submitted: L2/17-220 [Unicode17220] and L2/17-282 [Unicode17282]. In August, Steve Slevinski attended UTC #152.

Further discussions with the Unicode Technical Committee are dependent on the support of a voting member.

5.2. SignWriting in Unicode 8

The Sutton SignWriting symbol set based on the ISWA 2010 was encoded in the Unicode Standard version 8.0.

This encoding is based on the Unicode proposal from section 5.1 of draft-slevinski-iswa-2010 [UnicodeProposal], first published in January 2011.

The first draft officially submitted to the Unicode Technical Committee was N4015, a compromise with the Unicode committee that removed two-dimensional layout by dropping five structural markers and 500 number characters.

The second draft N4090 was under protest because it broke sorting and introduced variable length symbol names.

The third draft N4342 was rejected by the Center for Sutton Movement Writing. A new facial diacritic model was forced into the proposal that was neither defined nor tested.

In 2020, Google released Noto Sans SignWriting [UnicodeGoogle], a font for Sutton SignWriting that fully implements the official Unicode 8 design with modifying characters and facial diacritics. The font can be used to view any symbol of the International SignWriting Alphabet 2010 (ISWA 2010) as well as create complex facial expressions with diacritics. General layout of SignWriting is not part of the Unicode 8 design. Outside of facial expressions, layout requires a higher level protocol such as SVG.

5.2.1. Official Characters

In 2015, the symbols of Sutton SignWriting (Section 1.2 and Section 2.3.3) were added to Unicode version 8.

Description	Unicode 8 Range

Base Characters	U+1D800 to U+1DA8B
Fill Modifiers 2 to 6	U+1DA9B to U+1DA9F
Rotation Modifiers 2 to 16	U+1DAA1 to U+1DAAF

Table 24

Each symbol key can be rewritten using 1 to 3 Unicode characters of a base, optional fill, and optional rotation. Given a symbol key as variable "key", in JavaScript the 3 characters can be derived with simple formulas. Both Fill Modifier 1 (U+1DA9A) and Rotation Modifier 1 (U+1DAA0) are inherent characters and should be not be written in the character string.


```
var base = parseInt(key.substr(1,3),16) + parseInt('1D700',16);

var fill = parseInt(key.substr(4,1),16) + parseInt('1DA9A',16);

var rotation = parseInt(key.substr(5,1),16) +
parseInt('1DAA0',16);
```

The Sutton SignWriting Resources do not support these characters as defined in the Unicode standard. The presentation Issues with SignWriting in Unicode 8 [Unicode8Issues] details why this encoding is incomplete, broken, and fictional. Alternatively, the Sutton SignWriting Resources support the character sets (Section 2.2) of Formal SignWriting in ASCII (FSW) and SignWriting in Unicode (SWU).

5.2.2. 17 New Characters

The addition of 17 Unicode characters to the official Unicode standard can complete the script encoding and cover 2-dimensional layout.

Description	Formal SignWriting	Proposed Unicode
Fill Modifier 1	0	U+1DA9A
Rotation Modifier 1	0	U+1DAA0

Numbers	0 to 9	U+1DAB0 to U+1DAB9
Sequence Marker	A	U+1DABA
Signbox Markers	B	U+1DABB
Left Lane Markers	L	U+1DABC
Middle Lane Markers	M	U+1DABD
Right Lane Markers	R	U+1DABE

Table 25

Fill Modifier 1 and Rotation Modifier 1 are included to fix sorting and simplify processing.

The 10 number characters express the concept of distance, important for use with 2-dimensional scripts.

The 5 structural markers define cohesive units of the script.

5.3. SignWriting in Unicode (SWU)

Characters are used to name signs. Fonts are used to view signs.

SignWriting in Unicode (SWU) was first published in October 2016 and officially submitted to the Unicode Technical Committee in July 2017. SWU is not part of the Unicode standard. SignWriting in Unicode is an experimental Unicode design that is supported by the Sutton SignWriting Resources. This alternate encoding overwrites the Sutton SignWriting block in Unicode and uses plane 4 for the SignWriting symbols.

6. IANA Considerations

None.

7. Security Considerations

None.

8. References

[SignSpelling]
Sutton, V.S., "SignSpelling Guidelines", <https://www.signwriting.org/archive/docs6/sw0534-SignSpellingGuidelines-2008.pdf>.

[SLlegal] "Legal Recognition of Sign Language", <https://en.wikipedia.org/wiki/Legal_recognition_of_sign_languages>.

[SSWCoreDist]
Slevinski, S.S., "Sutton SignWriting Core Distribution", <https://unpkg.com/browse/@sutton-signwriting/core/>.

[SSWCoreDoc]
Slevinski, S.S., "Sutton SignWriting Core Documentation", <https://sutton-signwriting.github.io/core/>.

[SSWCoreSrc]
Slevinski, S.S., "Sutton SignWriting Core Source", <https://github.com/sutton-signwriting/core>.

[SSWFontDBDist]
Slevinski, S.S., "Sutton SignWriting Font DB Distribution", <https://unpkg.com/browse/@sutton-signwriting/font-db/>.

[SSWFontDBDoc]
Slevinski, S.S., "Sutton SignWriting Font DB Documentation", <https://sutton-signwriting.github.io/font-db/>.

[SSWFontDBSrc]
Slevinski, S.S., "Sutton SignWriting Font DB Source",

<https://github.com/sutton-signwriting/font-db>.

[SSWSgnwDist]
Slevinski, S.S., "Sutton SignWriting Web Components Distribution", <https://unpkg.com/browse/@sutton-signwriting/sgnw-components/>.

[SSWSgnwDoc]
Slevinski, S.S., "Sutton SignWriting Web Components Documentation", <https://sutton-signwriting.github.io/sgnw-components/>.

[SSWSgnwSrc]
Slevinski, S.S., "Sutton SignWriting Web Components Source", <https://github.com/sutton-signwriting/sgnw-components>.

[SSWTTFDist]
Slevinski, S.S., "Sutton SignWriting Font TTF Distribution", <https://unpkg.com/browse/@sutton-signwriting/font-ttf/>.

[SSWTTFDoc]
Slevinski, S.S., "Sutton SignWriting Font TTF Documentation", <https://sutton-signwriting.github.io/font-ttf/>.

[SSWTTFSrc]
Slevinski, S.S., "Sutton SignWriting Font TTF Source", <https://github.com/sutton-signwriting/font-ttf>.

[SteveSlevinski]
Slevinski, S.S., "Steve Slevinski's Homepage", <https://SteveSlevinski.me>.

[SW2010] Slevinski, S.S., "Sutton SignWriting Standard of 2010", <https://steveslevinski.me/#standard2010>.

[SW2012] Slevinski, S.S., "Sutton SignWriting Standard of 2012", <https://steveslevinski.me/#standard2012>.

[SW2017] Slevinski, S.S., "Sutton SignWriting Standard of 2017", <https://steveslevinski.me/#standard2017>.

[SWbyHand] Sutton, V.S., "Writing SignWriting by Hand", <https://www.signwriting.org/lessons/cursive>.

[SWChat] "Sutton SignWriting Chat on Gitter", <https://gitter.im/sutton-signwriting/community>.

[SWFacebook] "Sutton SignWriting Facebook Group", <https://www.facebook.com/groups/SuttonSignWriting/>.

[SWFontFill] Slevinski, S.S., "Sutton SignWriting Fill TypeType Font", <https://unpkg.com/@sutton-signwriting/font-ttf@1.0.0/font/SuttonSignWritingFill.ttf>.

[SWFontLine] Slevinski, S.S., "Sutton SignWriting Line TypeType Font", <https://unpkg.com/@sutton-signwriting/font-ttf@1.0.0/font/SuttonSignWritingLine.ttf>.

[SWFontOne] Slevinski, S.S., "Sutton SignWriting One Configuration Profile", <https://cdn.jsdelivr.net/gh/slevinski/signwriting_2010_fonts/fonts/SuttonSignWritingOne.mobileconfig>.

[SWFontOneD] Slevinski, S.S., "Sutton SignWriting One-D TypeType Font", <https://unpkg.com/@sutton-signwriting/font-ttf@1.0.0/font/SuttonSignWritingOneD.ttf>.

[SWFontSource] Slevinski, S.S., "Sutton SignWriting Font Source", <https://github.com/Slevinski/signwriting_2010_fonts/tree/master/source>.

[SWFontSymbol]
Slevinski, S.S., "Sutton SignWriting Symbol Configuration Profile", <https://cdn.jsdelivr.net/gh/slevinski/signwriting_2010_fonts/fonts/SuttonSignWritingSymbol.mobileconfig>.

[SWio] Slevinski, S.S., "Sutton SignWriting Resources", <https://www.sutton-signwriting.io>.

[SWList] "SignWriting Email List", <http://www.signwriting.org/forums/swlist/>.

[SWMobile] Gleaves, R.G., "SignWriting Mobile Site", <https://m.signwriting.org>.

[SWPatrons]
Slevinski, S.S., "Support SignWriting on Patreon", <https://patreon.com/signwriting>.

[SWScript] Slevinski, S.S., "SignWriting Script: an ode to writing by hand", <https://www.signpuddle.net/wiki/index.php/SignWriting_Script/>.

[SWWeb] Sutton, V.S., "SignWriting Website", <https://signwriting.org>.

[SWWikipedia]
"SignWriting Wikipedia Page", <https://en.wikipedia.org/wiki/SignWriting>.

[TwoDFont] Slevinski, S.S., "Two-Dimensional Font Prototype", <https://www.signwriting.org/symposium/presentation0019.html>.

[TwoDFontProject]
Slevinski, S.S., "Two-Dimensional Font Project", <https://meta.wikimedia.org/wiki/Grants:Project/slevinski/ASL_Wikipedia_2-D_Font_Development_for_SignWriting>.

[Unicode] "The Unicode Standard: A Technical Introduction", <http://unicode.org/standard/principles.html>.

[Unicode15194]
Slevinski, S.S., "Addressing SignWriting Collation in DUCET", <https://www.unicode.org/L2/L2015/15194-signwriting-response.pdf>.

[Unicode15219]
Anderson, D.A., Slevinski, S.S., and K.W. Whistler, "SignWriting Design, With Three Examples and Their Representation", <https://www.unicode.org/L2/L2015/15219-signwriting-design.pdf>.

[Unicode16225]
Slevinski, S.S., "SignWriting in Unicode Next", <http://www.unicode.org/cgi-bin/GetMatchingDocs.pl?L2/16-225>.

[Unicode17220]
Slevinski, S.S., "Design Options for Sutton SignWriting with examples and fonts", <https://www.unicode.org/L2/L2017/17220-signwriting-design-opt.pdf>.

[Unicode17282]
Slevinski, S.S., "Design Options for Sutton SignWriting Auxiliary", <https://www.unicode.org/L2/L2017/17282-signwriting-design-aux.pdf>.

[Unicode8Issues]
Slevinski, S.S., "Issues with SignWriting in Unicode 8", <https://www.slideshare.net/StephenSlevinski/sign-writing-in-unicode-8-issues>.

[UnicodeAnalysis]
Aznar, G.A., "Analysis of the different methods to encode SignWriting in Unicode", <https://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=A5A55FFC4B59332074F2B0670B758D4F?doi=10.1.1.188.7470&rep=rep1&type=pdf>.

[UnicodeChinese]
Zhang, S.Z., "Chinese Characters Are Futuristic and the Alphabet Is Old News", <https://www.theatlantic.com/technology/archive/2016/11/chinese-computers/504851/>.

[UnicodeGoogle]
Google, G., "Noto Sans SignWriting", <https://scriptsource.org/cms/scripts/

page.php?item_id=entry_detail&uid=wb9v9lchat>.

[UnicodeGraphite]
Hosken, M.H., "New engine feature proposal", <https://sourceforge.net/p/silgraphite/mailman/silgraphite-devel/thread/20121120152328.5984b784%40sil-mh6/#msg30121143>.

[UnicodeIdeas]
Hosken, M.H., "SignWriting Layout Discussion v2", <https://scriptsource.org/cms/scripts/render_download.php?format=file&media_id=..%2Fsites%2Fs%2Fmedia%2Fdatabase%2Fsproto%2Fentries%2Fjt%2F4f%2Fjt4fblkvhu_Signwriting_layout_discuss_v2.pdf&filename=Signwriting_layout_discuss_v2.pdf>.

[UnicodeN4015]
Everson, M.E., Slevinski, S.S., and V.S. Sutton, "Preliminary proposal for encoding the SignWriting script in the SMP of the UCS", <http://www.unicode.org/L2/L2011/11101-n4015-signwriting.pdf>.

[UnicodeN4090]
Everson, M.E., Slevinski, S.S., and V.S. Sutton, "Revised proposal for encoding the SignWriting script in the SMP of the UCS", <https://www.unicode.org/L2/L2011/11217-n4090-signwriting.pdf>.

[UnicodeN4342]
Everson, M.E., Hosken, M.H., Slevinski, S.S., and V.S. Sutton, "Proposal for encoding Sutton SignWriting in the UCS", <https://www.unicode.org/L2/L2012/12321-n4342-signwriting.pdf>.

[UnicodePragmatic]
Batchelder, N.B., "Pragmatic Unicode, or, How do I stop the pain?", <https://www.youtube.com/watch?v=sgHbC6udIqc>.

[UnicodeProposal]
Slevinski, S.S., "Encoding the graphemes of the SignWriting Script with the x-ISWA-2010", <https://tools.ietf.org/html/draft-slevinski-iswa-2010-00#section-5.1>.

[UnicodeScript]
"Unicode Status (SignWriting)", <https://scriptsource.org/cms/scripts/page.php?item_id=entry_detail&uid=jt4fblkvhu>.

Appendix A. A brief history

A.1. SignWriting

Year(s)	Event(s)
1966	Valerie Sutton invented DanceWriting
1974	Valerie Sutton invented SignWriting
1974 to 1986	SignWriting is written exclusively by hand
1981 to 1984	publishing efforts include stencils and wax transfers
1986 to 1995	SignWriting is successfully computer encoding with keyboarding support
2002	advanced sorting of SignWriting dictionaries is available
2004	a drag-and-drop interface is created for SignWriting
2006	SignWriting received the ISO 15924 script code "Sgnw"
2010	the International SignWriting Alphabet 2020 (ISWA

	2010) is released	
2012	the Formal SignWriting in ASCII (FSW) specification is released	
2015	the Sutton SignWriting Block is added to the Unicode Standard	
2017	the SignWriting in Unicode (SWU) specification is released	
2020 to present	Steve Slevinski assumed full responsibility for administering and financially supporting the SignWriting websites	
2020 to present	Valerie Sutton continues her personal support for SignWriting with plans for a new series of SignWriting instruction books and a future SignWriting Trust for the long-term support of SignWriting	

2021	SignWriting is expanding into the machine learning space with a focus on video analysis and translation	

Table 26

A.2. Steve Slevinski

Year(s)	Event(s)
1994	Steve graduated from Grove City College with a Bachelor of Science in Mathematics, computer science applied
1994	Steve was hired by the New York State Education Department as a Senior Computer Programmer / Analyst
1996	Steve was hired by Danet Inc in telecommunications for quality assurance and in 1998 he was promoted to

	the maintainer of the internal business systems
1999	Steve raised his first son with sign language, who learned to use several dozen signs before he spoke his first word
2004	Steve raised his daughter with sign language, but she only learned a few signs before she started talking
2002	Steve became a friend of the deaf through his wife's work in Pittsburgh and through connections at the Western Pennsylvania School for the Deaf (WPSD)
2004 to present	Steve has been actively working with Valerie Sutton on a weekly basis with Valerie sharing her invention and Steve trying to make it a reality with technology
2004 to 2019	Steve was a paid consultant to the Center for Sutton Movement Writing non-profit
2020 to present	Steve was hired by the University of Iowa as a data manager for psychiatry research and neuroimaging
2020 forward	Steve continues his work with SignWriting on the weekends and early mornings

Table 27

A.3. Financial Support

Year(s)	Event(s)
1974 to 2019	SignWriting was financially supported and promoted through the Center for Sutton Movement Writing (CSMW) non-profit
2018	the Center for Sutton Movement Writing (CSMW) lost a major funding source

2019	the Center for Sutton Movement Writing closed the non-profit status of the organization due to excessive government paperwork and a focus on fundraising rather than productive work
2019	Steve Slevinski started a Patreon campaign [SWPatrons] to support current and future work with SignWriting
2022	Added crypto options to support current and future work with SignWriting * Bitcoin: bc1qj0fxphz4qsdt6xd4f3cr4lc96rxyp2ngh0npdp * Ethereum: 0x68733c1Dd7CB5A76d38b0711d937FBb953928357 * Dogecoin: DC8K7qWC7WdTrbDSpAJhFtp3kr9owzMF8j

Table 28

Appendix B. SignWriting General Interest

The Sutton SignWriting resources are free to use by anyone for any purpose. Sutton SignWriting supports free culture and the creation of free culture works.

Sutton SignWriting Resources [SWio]

SignWriting Website [SWWeb]

SignWriting Mobile Site [SWMobile]

SignWriting Wikipedia Page [SWWikipedia]

SignWriting Email List [SWList]

SignWriting Group on Facebook [SWFacebook]

SignWriting Online Chat [SWChat]

Appendix C. Proposed Null Symbol

For consideration and comment, a new null symbol is proposed. This new symbol is for use in the temporal prefix and not in the spatial signbox. The null symbol can be used for one-handed signs so that they will be sorted before two-handed signs.

Type	FSW	SWU
null symbol	S00000	U+40000

Table 29

Sorting uses a binary string comparison. The null symbol will sort before all other symbols. Following the SignSpelling guidelines [SignSpelling], the temporal prefix starts with the dominant hand and is followed by the non-dominant hand. For one-handed signs, the null symbol can be used in place of the non-dominant hand.

Sort Order	Symbol Key	Definition
1	S00000	Null Symbol
2	S10000	Index
3	S10100	Index on Circle
4	S20500	Touch Single

Table 30

C.1. Sorting without the Null Symbol

Without the null symbol, the one-handed signs may be sorted after two-handed signs. This is suboptimal.

Sort Order	Temporal Prefix	Description
1	AS10000	Single hand
2	AS10000S10100	Two hands
3	AS10000S20500	Single hand and contact

Table 31

C.2. Sorting with the Null Symbol

With the null symbol, all one-handed signs can be sorted before two-handed signs. This is ideal.

Sort Order	Temporal Prefix	Description
1	AS10000	Single hand
2	AS10000S00000S20500	Single hand, null symbol, and contact
3	AS10000S10100	Two hands

Table 32

Author's Address

Steve Slevinski
www.sutton-signwriting.io

Email: slevinski@signwriting.org