Sutton-Slevinski Collaboration S. Slevinski
Internet-Draft www.sutton-signwriting.io
Intended status: Informational 12 October 2025
Expires: 15 April 2026

Formal SignWriting
draft-slevinski-formal-signwriting-10

Abstract

Formal SignWriting is a character encoding model for Sutton SignWriting, a script for writing sign languages recognized under ISO 15924 script code "Sgnw". It defines a sign as a two-part word, represented as a string of characters processable by regular expressions, optimized for display, searching, sorting, and text processing. This document specifies the character sets, grammar, token patterns, and technical integration for Formal SignWriting in ASCII (FSW) and SignWriting in Unicode (SWU), including styling, query language, and transformations. The specification is intended for implementation and evaluation, with unrestricted distribution.

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 15 April 2026.

Table of Contents

1. Formal SignWriting

M548x535S10019452x474S10011476x465S2ea04481x501S2ea48459x509S29b0b514x500S15a0a515x473S1eb20524x489

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
("ざぉ ▯ ▯▯３ー▯▯")

Formal SignWriting is a character encoding model for Sutton SignWriting, representing a sign as a two-part word: a temporal prefix (one-dimensional symbol sequence) and a spatial signbox (two-dimensional symbol arrangement). Each word is a string of characters governed by a defined grammar, processable by regular expressions, and optimized for display, searching, sorting, and text processing.

1.1. Character Sets

Formal SignWriting uses two isomorphic character sets: Formal SignWriting in ASCII (FSW) and SignWriting in Unicode (SWU), with bi-directional conversion between them.

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
Null Symbol	S00000	U+40000
Symbols	S10000 to S38b07	U+40001 to U+4F428

Table 1

1.1.1. Formal SignWriting in ASCII (FSW)

FSW uses ASCII characters: A, B, L, M, R, S, 0-9, a-f, x. Released in January 2012, it has been stable since.

1.1.2. SignWriting in Unicode (SWU)

SWU, published in October 2016 and submitted to the Unicode Technical Committee in July 2017, is an experimental encoding using the Sutton SignWriting block and plane 4 for symbols. It is not part of the Unicode standard but is supported by Sutton SignWriting resources.

1.2. Building Blocks

Formal SignWriting strings are composed of tokens forming a grammar for signs, processable by regular expressions.

1.2.1. Regular Expressions

Regular expressions define the grammar for Formal SignWriting strings.

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

1.2.2. Token Patterns

Formal SignWriting uses 11 tokens, grouped into four layers: structural markers (A, B, L, M, R), base symbols (u, w, s, P), modifiers (i, o), and numbers (n).

Tokens of Formal SignWriting

Token	Description
A	Sequence Marker
B	Signbox Marker
L	Left Lane Marker
M	Middle Lane Marker
R	Right Lane Marker
u	Null BaseSymbol
w	Writing BaseSymbols
s	Detailed Location BaseSymbols
P	Punctuation BaseSymbols
i	Fill Modifiers
o	Rotation Modifiers
n	Number from 250 to 749

Table 3

1.2.3. Symbols

Symbols are defined by three tokens: base symbol (w, s, P), fill modifier (i), and rotation modifier (o), based on the International SignWriting Alphabet 2010 (ISWA 2010).

Symbol Tokens

Token Pattern	Description
w	Writing BaseSymbols
s	Detailed Location BaseSymbols
P	Punctuation BaseSymbols
i	Fill Modifiers
o	Rotation Modifiers
wio	Writing symbol (base, fill, rotation)
[ws]io	Writing or detailed location symbol
Pio	Punctuation symbol

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	Punctuation 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+4F428
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+4F428

Table 6

1.2.3.1. FSW Symbols

FSW symbol keys are 6 characters: "S", followed by a 3-character base (100-38b), a fill modifier (0-5), and a rotation modifier (0-9 and a-f).

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

1.2.3.2. SWU Symbols

SWU symbols (37,811) are identified by Unicode characters U+40001 to U+4F428. Conversion from FSW symbol key to SWU codepoint is defined as:

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

1.2.4. 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

1.2.4.1. 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])

1.2.4.2. SWU Numbers

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

1.3. Two-Part Word

A sign is a two-part word of time and space: a temporal prefix (subjective, one-dimensional symbol sequence) and a spatial signbox (objective, two-dimensional symbol arrangement).

1.3.1. Temporal Prefix

The temporal prefix is a one-dimensional sequence of symbols (writing, null, or detailed location) starting with "A", enabling sorting via binary string comparison.

Temporal Prefix Tokens

Token Patterns	Description
A	Sequence Marker
u	Null BaseSymbol
w	Writing BaseSymbols
s	Detailed Location BaseSymbols
i	Fill Modifiers
o	Rotation Modifiers
(A([uws]io)+)?	Optional temporal prefix

Table 9

1.3.2. Spatial Signbox

The spatial signbox is a two-dimensional cluster of writing symbols with Cartesian coordinates for top-left placement. Symbol order affects overlap (earlier symbols are underneath).

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
wionn	Spatial symbol (writing symbol + coordinates)
(wionn)*	Zero or more spatial symbols
Bnn(wionn)*	Signbox for horizontal writing
[LMR]nn(wionn)*	Signbox for vertical writing

Table 10

1.3.2.1. 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 are taken straight from the coordinates in a Formal SignWriting word.

1.3.2.2. 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.

1.3.2.3. 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.

1.4. 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 only. 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}!)?)?
```

1.4.1. Entire Sign

There are several options for styling an entire sign.

C Colorize

P Padding

G Background

D Detail colors

Z Zoom level

1.4.1.1. 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

1.4.1.2. 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

1.4.1.3. 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

1.4.1.4. 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

1.4.1.5. 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

1.4.2. 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

1.4.2.1. 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

1.4.3. 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

1.5. 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.

1.5.1. 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 (Y).

Full query string definition: QP?X?Y?

1.5.2. 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)*

1.5.3. 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 indicates the end of prefix searching

Table 20

Definition of temporal prefix searching.

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

1.5.4. 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?

1.5.5. Regular Expressions

1.5.5.1. 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}t${base}
I	item	(?:${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})?

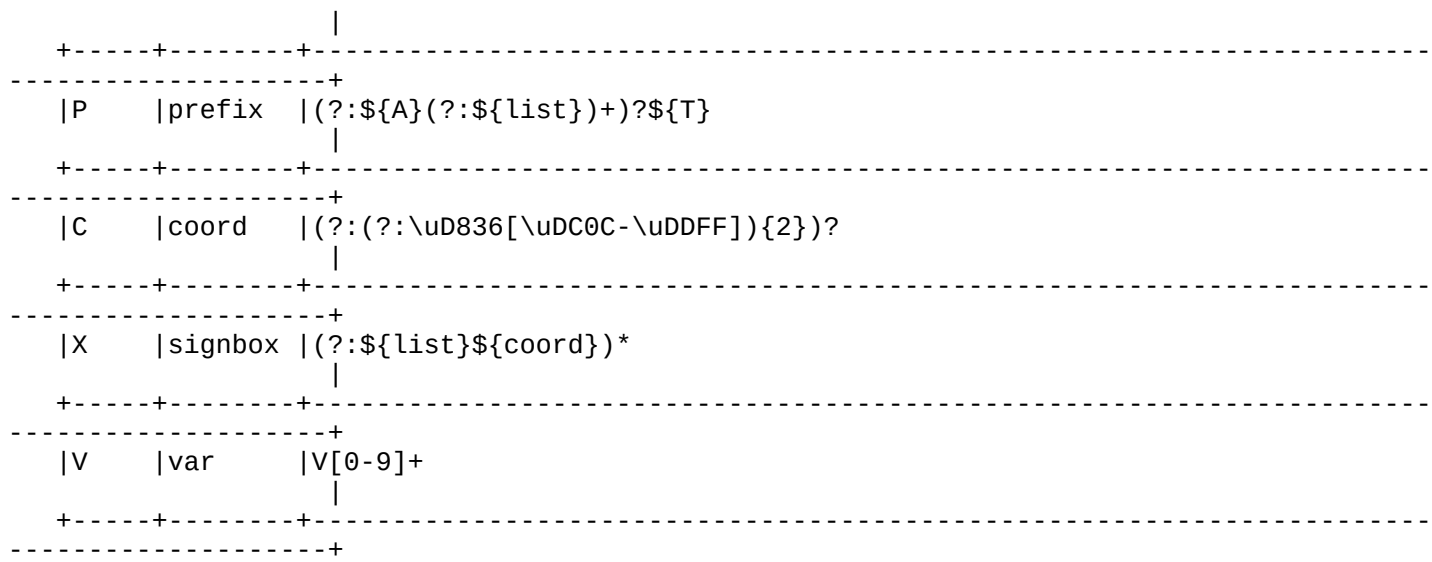
X	signbox	(?:${list}${coord})*
V	var	V[0-9]+
Y	style	-
F	full	${Q}(${prefix})?(${signbox})?(${var})?(${style}?)

Table 22

1.5.5.2. SWU Query Regular Expressions

SWU Query Elements

Token	Variable	Regular Expression
Q	Q	Q
B	base	(?:(?:\uD8C0[\uDC01-\uDFFF])|(?:[\uD8C1-\uD8FC][\uDC00-\uDFFF])|(?:\uD8FD[\uDC00-\uDC80]))
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

1.6. Transformations

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

1.6.1. 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

1.6.2. Query String to Regular Expression

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

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

Spatial Signbox Structure: [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 Structure 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 Structure regular expression.

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

The Spatial Symbols are 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) 10s don’t match and the min 1s are not zero (last number to 9): Match 12[2-9]

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

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

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

5) Bring up the 1s: 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 1.4.

2. Technology Integration

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

2.1. 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.

2.1.1. 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]

2.1.2. 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]

2.1.3. Android

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

2.2. 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 website 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.

2.3. Scalable Vector Graphics

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

2.3.1. Font Based SVG

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

2.3.1.1. 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">
```

2.3.1.2. 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>
```

2.3.1.3. 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 1.2.3.2 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>
```

2.3.2. Stand Alone SVG

It is possible to create stand-alone SVG images from @sutton-signwriting/font-db. These SVG graphics 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 2.3.1

Within stand-alone SVG, multiple SVG elements are used. Each SVG elements uses X and Y coordinates to place the symbols. 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 be requested from a server or downloaded in a database.

```
<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>
```

2.4. HTML and CSS

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

2.4.1. 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%;
}
```

2.4.2. 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; }
```

2.4.3. Other Effects

Other CSS rules can be used for other effect. Please note that transform property does not affect 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);
}
```

2.4.4. 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;
   }
```

2.5. JavaScript Packages

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

2.5.1. @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
```

2.5.2. @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

2.5.3. @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

2.5.4. @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

3. Unicode Considerations

3.1. Unicode 8 Encoding

Sutton SignWriting symbols are encoded in Unicode 8 (U+1D800 to U+1DAAF), with base characters, fill modifiers (2-6), and rotation modifiers (2-16).

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

FSW symbol key conversion to Unicode 8:

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

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

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

3.2. Proposed Unicode Extensions

17 additional characters are proposed to support 2D layout and sorting.

Description	FSW	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

4. IANA Considerations

None.

5. Security Considerations

None.

6. References

[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>.

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

[SWFontLine]
Slevinski, S.S., "Sutton SignWriting Line TrueType 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 TrueType 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>.

Author's Address

Steve Slevinski
www.sutton-signwriting.io
Email: slevinski@signwriting.org