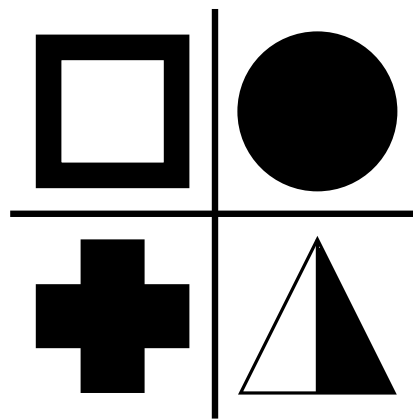


# The SymbolEditor Handbook

Stephen P. Allewell



## The SymbolEditor Handbook

# Contents

|          |                                          |           |
|----------|------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                      | <b>5</b>  |
| <b>2</b> | <b>The User Interface</b>                | <b>6</b>  |
| 2.1      | The SymbolEditor Main Window . . . . .   | 6         |
| 2.1.1    | Editor Window . . . . .                  | 6         |
| 2.1.2    | Library Window . . . . .                 | 7         |
| <b>3</b> | <b>Symbols</b>                           | <b>8</b>  |
| <b>4</b> | <b>Command Reference</b>                 | <b>9</b>  |
| 4.1      | SymbolEditor Menus . . . . .             | 9         |
| 4.1.1    | The File Menu . . . . .                  | 9         |
| 4.1.2    | The Edit Menu . . . . .                  | 10        |
| 4.1.3    | The Rendering Menu and Toolbar . . . . . | 10        |
| 4.1.4    | The Tools Menu and Toolbar . . . . .     | 10        |
| 4.1.5    | The Settings and Help Menu . . . . .     | 12        |
| <b>5</b> | <b>Dialogs</b>                           | <b>13</b> |
| 5.1      | The Configuration Dialog . . . . .       | 13        |
| <b>6</b> | <b>Credits and License</b>               | <b>14</b> |

### **Abstract**

SymbolEditor is an application to allow the creation and editing of symbol libraries for the KXStitch application.

# Chapter 1

## Introduction

SymbolEditor is used to create cross stitch symbols for the KXStitch application. Originally KXStitch relied on standard character fonts to supply these symbols, but differences in user languages and the quality of the fonts available along with the alignment of the characters and the limited number available has driven the need to create a dedicated symbol set.

The symbol set will be stored in a file and several files can be created depending on the users needs, for example having a Halloween symbol set for Halloween themed patterns.

Each of the files will contain a series of symbols which will be displayed in the symbol library tab. Editing of existing symbols can be done by clicking the entry in the library which will populate the editor with this symbol. Alternatively new symbols can be created and added to the library.

Please report any problems or feature requests to the SymbolEditor KDE mailing list [symboleditor@kde.org](mailto:symboleditor@kde.org), you can subscribe [here](#) or the KDE bug tracking system at <http://bugs.kde.org/>

The SymbolEditor home page is at <http://userbase.kde.org/SymbolEditor>

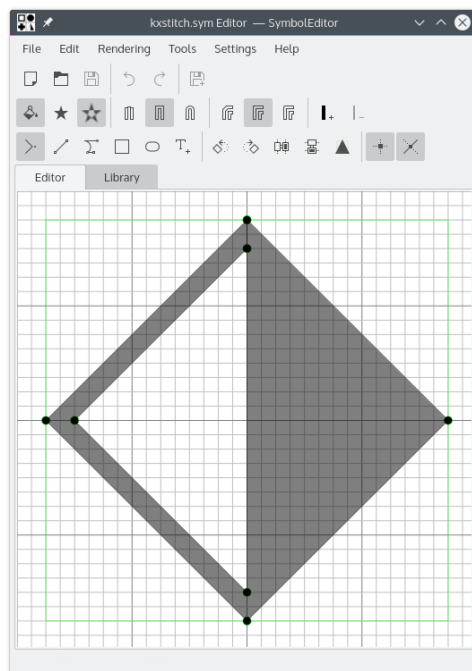
## Chapter 2

# The User Interface

### 2.1 The SymbolEditor Main Window

The main window consists of a tabbed widget containing the symbol editor and a library view. There is a menu bar and several tool bars containing standard actions, editing tools and rendering options. The status bar gives prompts for using the various tools.

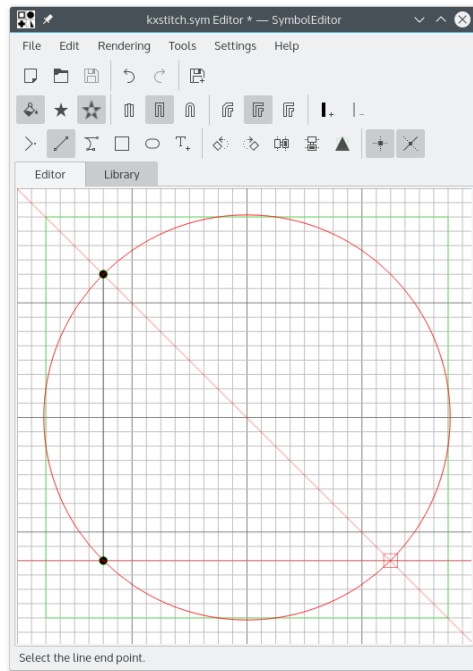
#### 2.1.1 Editor Window



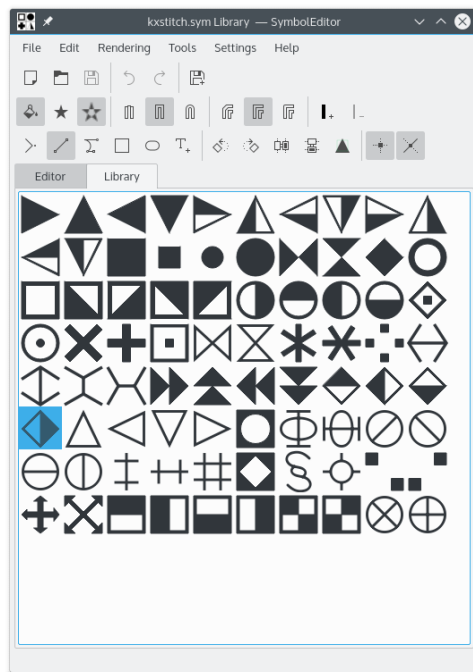
The Editor window allows editing of a symbol using the available tools. It consists of a grid providing snap points and shows a preferred size guide which allows symbols to be created to a consistent size. The size of the grid, the number of elements and the preferred size guide are all configurable.

The editor provides guidelines for positioning points relative to existing points. The guides are positioned on the horizontal, vertical and diagonal vectors from an existing point and also on a circular reference around the center of the grid.

## The SymbolEditor Handbook



### 2.1.2 Library Window



The library window shows representations of the symbols contained in the library. The symbol currently being edited is shown highlighted. Selecting a symbol will open it in the editor so that it can be modified.

It is possible to open several instances of the SymbolEditor application and copy symbols from one library to another using drag and drop. Ensure that both library windows are visible, then click and drag the required symbol from one library window to the other. The symbol will then be appended to the target library.

## Chapter 3

# Symbols

Symbols are created as QPainterPath objects with additional information defining attributes of the path. A QPainterPath is made up of sub paths such as rectangles, ellipses or chains of lines and bezier curves. The paths may intersect each other forming complex shapes.

The symbol may be filled using either the winding option which will fill the entire path, or the odd even option which fill alternate sections of the path.

If the path is not filled, the outline will have a defined line width, end cap type and line join type. See the QPainterPath documentation for more detail.



# Chapter 4

## Command Reference

### 4.1 SymbolEditor Menus

#### 4.1.1 The File Menu

**New (Ctrl-N)**

Creates a new library

**Open (Ctrl-O)**

Opens an existing library

**Open Recent**

Open a recently used library

**Save (Ctrl-S)**

Saves the library

**Save As (Ctrl-Shift-S)**

Saves the library under a new name

**Save Symbol**

Save the symbol being edited

If the symbol being edited was selected from the library, saving will update that symbol. If it is a new symbol, it will be appended to the library and subsequent saves will append another new symbol to the library.

**Save Symbol as New**

Save the symbol being edited as a new symbol

When a symbol is saved as new it will be appended to the library regardless of it having originally been selected from the library for editing. Subsequent saves will append another new symbol to the library.

**Import Library**

Import a library appending the contained symbols into the current library

**Close (Ctrl-W)**

Close the library

**Quit (Ctrl-Q)**

Quits SymbolEditor

### 4.1.2 The Edit Menu

SymbolEditor has the common KDE **Edit** menu items, for more information read the sections about the [Edit Menu](#) of the KDE Fundamentals.

### 4.1.3 The Rendering Menu and Toolbar



#### Fill Path

Enable filling of the path

#### Winding Fill

Select the winding fill method

#### Odd Even Fill

Select the odd even fill method

#### Flat Cap

Select the flat cap line end type

#### Square Cap

Select the square cap line end type

#### Round Cap

Select the round cap line end type

#### Bevel Join

Select the bevel line join type

#### Miter Join

Select the miter line join type

#### Round Join

Select the round line join type

#### Increase Line Width

Increase the path line width

#### Decrease Line Width

Decrease the path line width

### 4.1.4 The Tools Menu and Toolbar

A number of tools are available to aid the design of the symbols. The symbols are composed of a series of sub paths and each sub path is composed of a move to the start position (this defaults to 0,0 for new symbols) followed by lines and curves. The curves are cubic splines having a start, an end and two control points defining the curve. There are convenience tools to create rectangles and ellipses, but these will be broken down into lines and curves.

All points created for the elements are moveable by dragging them to their new position. All points can be snapped to the grid intersections if the snap option is turned on, otherwise they can be positioned anywhere.

## The SymbolEditor Handbook

The symbol can be rotated clockwise and counter clockwise and also flipped vertically and horizontally. This allows multiple symbols to be easily created based on the same design. Remember to use the save symbol as new option for this.



### Move To

Move the starting position of the sub path

If an existing sub path is being created, this will be closed and a new sub path started at the new position

### Draw To

Draw a line to the next selected point

### Cubic To

Draw a bezier curve using two control points and an end point

### Rectangle

Draw a rectangle between two opposite corners

This closes any existing sub path and creates a new sub path consisting of the rectangle.

### Ellipse

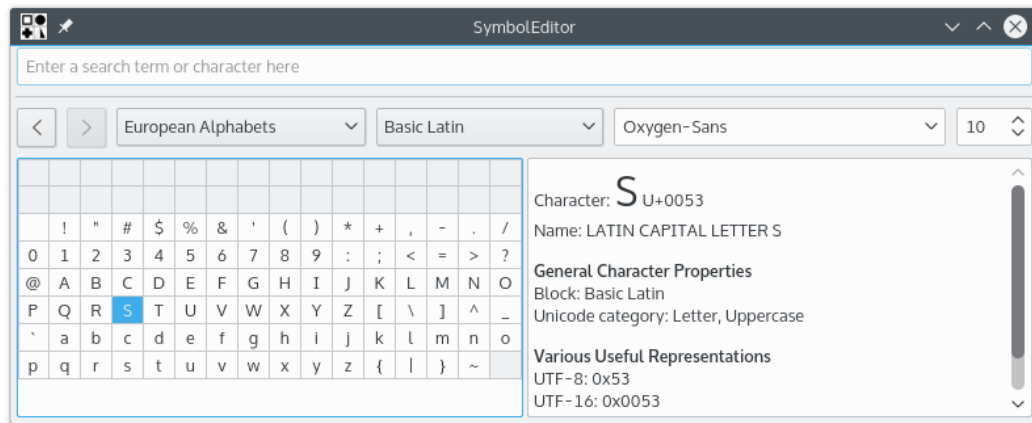
Draw an ellipse between two opposite corners

This closes any existing sub path and creates a new sub path consisting of the ellipse.

### Insert Character

Select a character from a font and insert it into the editor

This will overwrite any existing paths



The character selector dialog is the standard KDE dialog and allows selection of characters from any font. The size is not used as the character is scaled to fit within the preferred size guide.

Double clicking a character will insert that character into the editor. The dialog will stay open so that another character could be inserted, although this will overwrite the first. This does allow an insert character, save, insert character, save workflow to quickly build up a library of symbols. Ensure that the symbol being edited is a new symbol and not one previously selected from the library, otherwise saving would keep overwriting the one in the library.

**Rotate Left**

Rotate the whole symbol counter clockwise 90 degrees

**Rotate Right**

Rotate the whole symbol clockwise 90 degrees

**Flip Horizontal**

Flip the symbol across the vertical axis

**Flip Vertical**

Flip the symbol across the horizontal axis

**Scale to Preferred Size**

Scale the symbol to fit within the defined preferred size

Note that this only affects the points; where a non filled symbol is created and a line width is defined, this may extend beyond the guide size.

**Enable Snap**

Enable snapping of points to the grid

**Guide Lines**

Enable the generation of guide lines

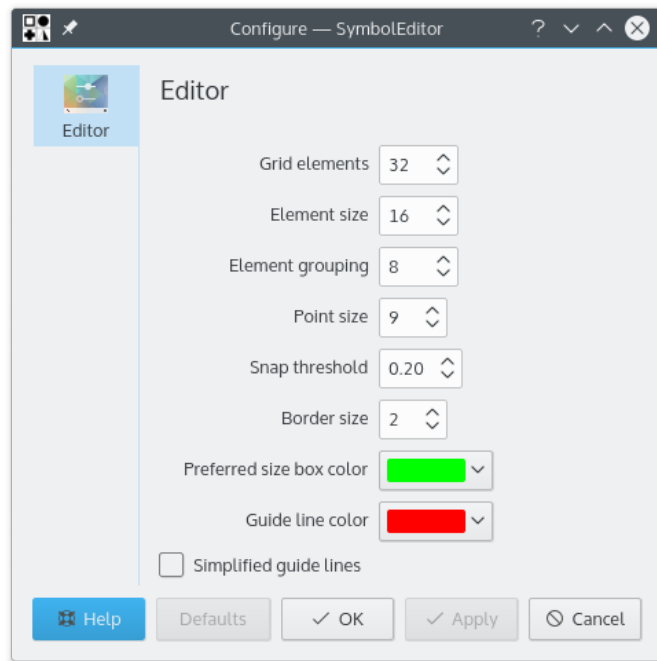
## 4.1.5 The Settings and Help Menu

SymbolEditor has the common KDE **Settings** and **Help** menu items, for more information read the sections about the [Settings Menu](#) and [Help Menu](#) of the KDE Fundamentals.

## Chapter 5

# Dialogs

### 5.1 The Configuration Dialog



The configuration dialog allows a number of options to be set.

## Chapter 6

# Credits and License

SymbolEditor

Program copyright 2012-2015 Stephen P. Allewell [steve.allewell@gmail.com](mailto:steve.allewell@gmail.com)

Contributors:

- Translations provided by various contributors. See the translation file for details.

Documentation copyright 2012-2015 Stephen P. Allewell [steve.allewell@gmail.com](mailto:steve.allewell@gmail.com)

This documentation is licensed under the terms of the [GNU Free Documentation License](#).

This program is licensed under the terms of the [GNU General Public License](#).